Bioinformatics and Systems Biology: Expression, Regulation and Networks - Exam project

By Anouk Vinckier Louis Desmet Robbe Naert

The correct formation of heart valves during embryonic development is critical for normal functioning of the heart. Abnormally formed heart valves are responsible for 1/3 of cardiovascular birth defects and valve defects are correlated with an increased disease susceptibility.

Heart valves develop at two locations in the embryonic heart. The first location is between the atria and ventricles also known as the atrioventricular canal (AVC) where the mitral and tricuspud valves are formed. The second location is at the base of the aorta and pulmonary trunk also known as the outflow tract (OFT) where the aortic and pulmonary valves are formed. During heart valve development endothelial cells transform into mesenchymal cells and move to the cardiac cushions. Here these mesenchymal cells proliferate and differentate to eventually form valve leaflets.

Sox9 is highly expressed in the mesenchyme of these cardiac cushions. The essential role of sox9 in valve development in mice has been shown by several research groups. The loss of sox9 interferes with normal valve development and is embryonically lethal. In humans, several sox9 mutations have been described resulting in a condition called campomelic dysplasia. This disease is characterized by skeletal abnormalties, sex reversal and multiple organ defects such as defects in the pancreas and heart.

Sox9 is also highly expressed in cartilage containing tissues such as the limb that differentiates from sox9-positive mesenchyme. Loss of sox9 leads to loss of bone formation in this case. Despite the importance of sox9 in valve development, not much is known about its transcription targets or action mechanism. To study this, reseachers have combined ChIP-Seq with RNA-seq to determine DNA-binding sites in mouse embryonic AVC and limb. Moreover they also determined which genes in the embryonic heart are affected by knocking-out sox9 via RNA-seq.

We will first use the RNA sequencing data provided to see which genes are down - and upregulated in the AVC when sox9 is knocked out in the heart. Then we will also use the provided ChIP sequencing data to try to determine binding targets for sox9 in the heart and limb.

Article used: https://www.ncbi.nlm.nih.gov/pubmed/26525672 (https://www.ncbi.nlm.nih.gov/pubmed/26525672)

Dataset used: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE73225 (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE73225)

All the commands and data were performed under the account of Robbe Naert (r0305394)

Path to the original notebook: /mnt/storage/r0305394/jupyternotebooks/groepswerk/Exam_project_SOX9.ipynb

Path to the R notebook: /mnt/storage/r0305394/jupyternotebooks/groepswerk/Exam_project_SOX9_R.ipynb

RNA sequencing

We first made a folder (data) to store the sra files in.

In [1]:
```
pwd
cd /mnt/storage/r0305394/jupyternotebooks/Groepwerk
mkdir -p /mnt/storage/r0305394/jupyternotebooks/Groepwerk/data
```

```
/mnt/storage/r0305394/jupyternotebooks/Groepwerk
```

Following command was used to convert the sra files into fastaq files. This generates two fastq files per SRA files since these were paired-end reads. The commands are shown below. This wasn't performed in jupyter but in the terminal because it took some time and in this way we could detach the process.

fastq-dump --split-files SRR2479801 fastq-dump --split-files SRR2479802 fastq-dump --split-files SRR2479803 fastq-dump --split-files SRR2479804 fastq-dump --split-files SRR2479805 fastq-dump --split-files SRR2479806 fastq-dump --split-files SRR2479807 fastq-dump --split-files SRR2479808

Then we made a new folder to store the results obtained from the RNA-sequencing.

In [1]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk
```

We also made a new folder to store all the output from the quality control of the fastq files generated earlier (this folder was later removed because there were storage problems).

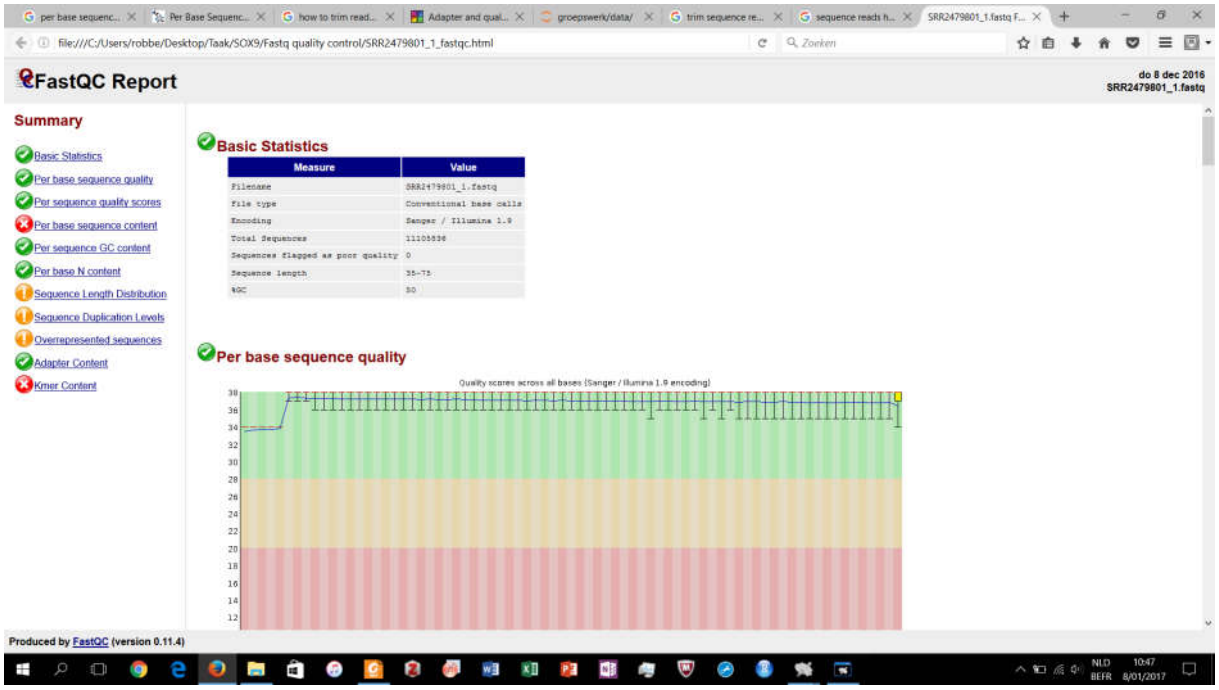
Then we performed quality control on all the fastq files. In class we have seen that the three most important parameters during quality analysis are: per base sequence quality, sequence duplication levels and overrepresented sequences.

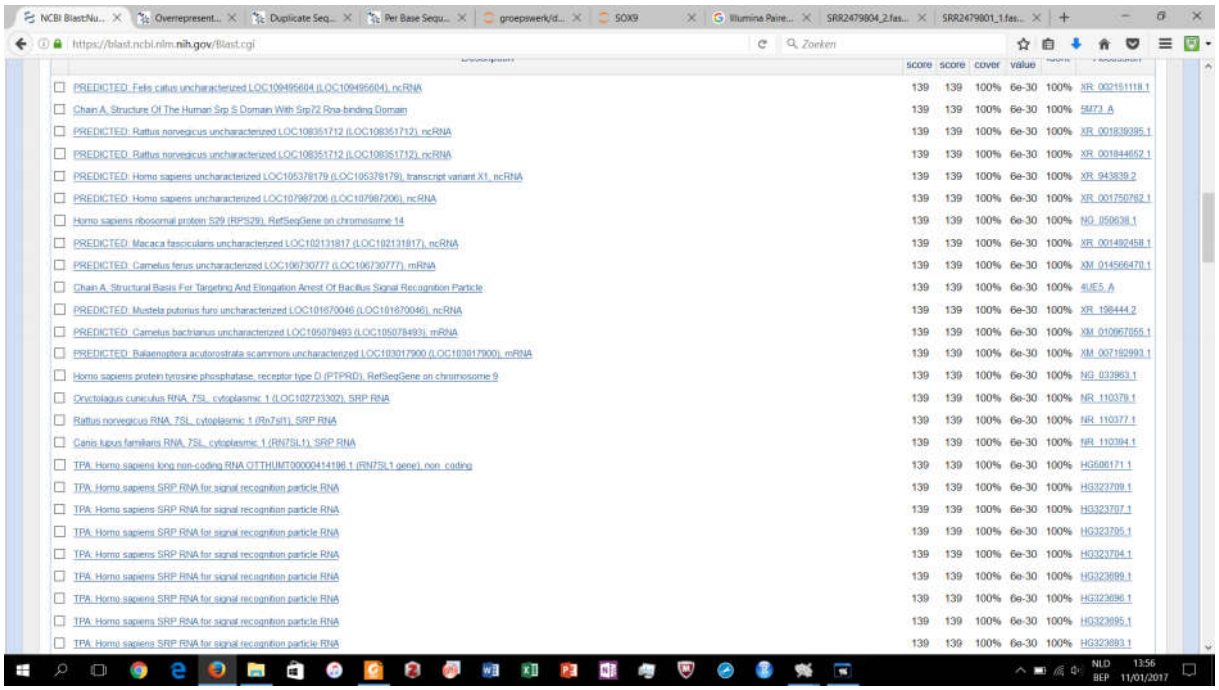
In [9]:
```
```

In [3]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/
```

```
Started analysis of SRR2479801_1.fastq
Approx 5% complete for SRR2479801_1.fastq
Approx 10% complete for SRR2479801_1.fastq
Approx 15% complete for SRR2479801_1.fastq
Approx 20% complete for SRR2479801_1.fastq
Approx 25% complete for SRR2479801_1.fastq
Approx 30% complete for SRR2479801_1.fastq
Approx 35% complete for SRR2479801_1.fastq
Approx 40% complete for SRR2479801_1.fastq
Approx 45% complete for SRR2479801_1.fastq
Approx 50% complete for SRR2479801_1.fastq
Approx 55% complete for SRR2479801_1.fastq
Approx 60% complete for SRR2479801_1.fastq
Approx 65% complete for SRR2479801_1.fastq
Approx 70% complete for SRR2479801_1.fastq
Approx 75% complete for SRR2479801_1.fastq
Approx 80% complete for SRR2479801_1.fastq
Approx 85% complete for SRR2479801_1.fastq
Approx 90% complete for SRR2479801_1.fastq
Approx 95% complete for SRR2479801_1.fastq
Analysis complete for SRR2479801_1.fastq
```

The overall quality of the reads seems to be good. The per base sequence quality is good although there is warning for sequence duplication levels and overrepresented sequences. There is a warning for the sequence duplication levels. This warning is generated when non-unique sequences make up for more than 20% of the total number of sequences. In this case the percentage of sequences that remain after removing duplicates is 62,56%. The sequence duplication levels are most likely the result of oversequencing to be able to detect lowly expressed transcripts. There is also a warning for overrepresented sequences. This warning is triggered when a sequence is found that represent more than 0.1% of the total amount of sequences. These overrepresented sequences are either very important for biological functioning or are caused by contamination. We blasted all the overrepresented sequences. All of them led us to practically the same results (shown under the fastq quality output). The blast result seems to indicate that this overrepresented sequence is human or rat contamination.

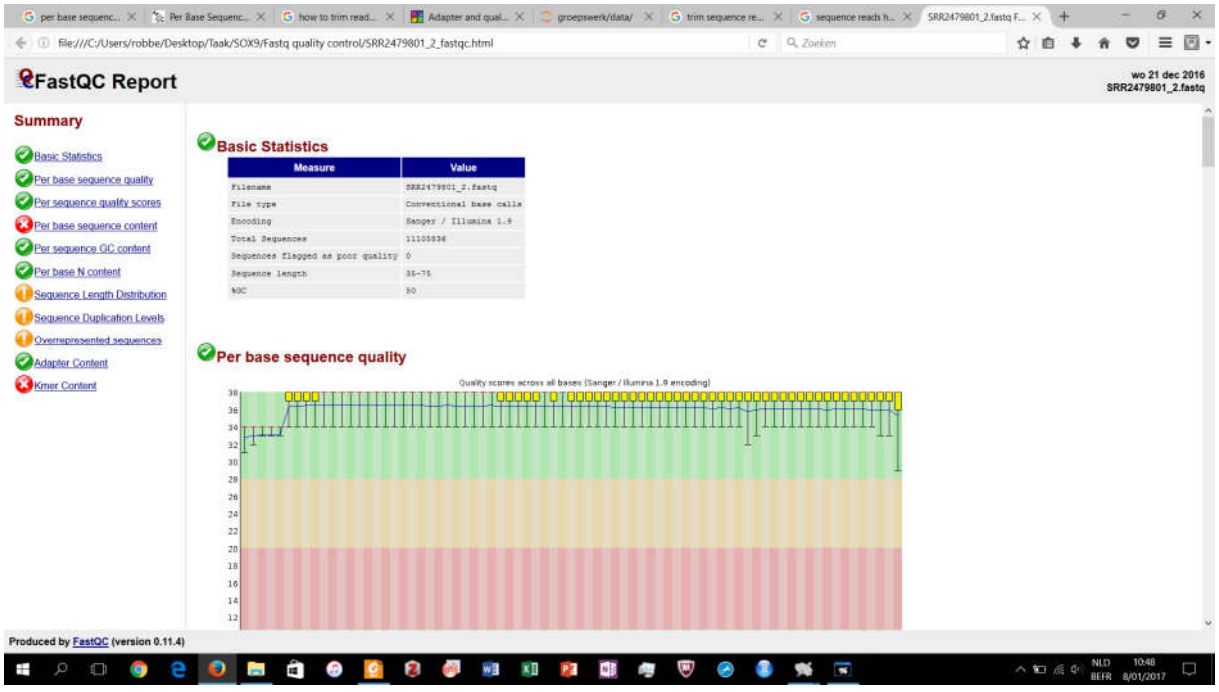

FastQC Report
do 8 dec 2016
SRR2479801_1.fastq
Summary
Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Kmer Content
Basic Statistics
Measure
Value
Filename
SRR2479801_1.fastq
File type
Conventional base calls
Encoding
Sanger / Illumina 1.9
Total Sequences
11105836
Sequences flagged as poor quality
0
Sequence length
35-75
%GC
50
Per base sequence quality
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Produced by FastQC (version 0.11.4)

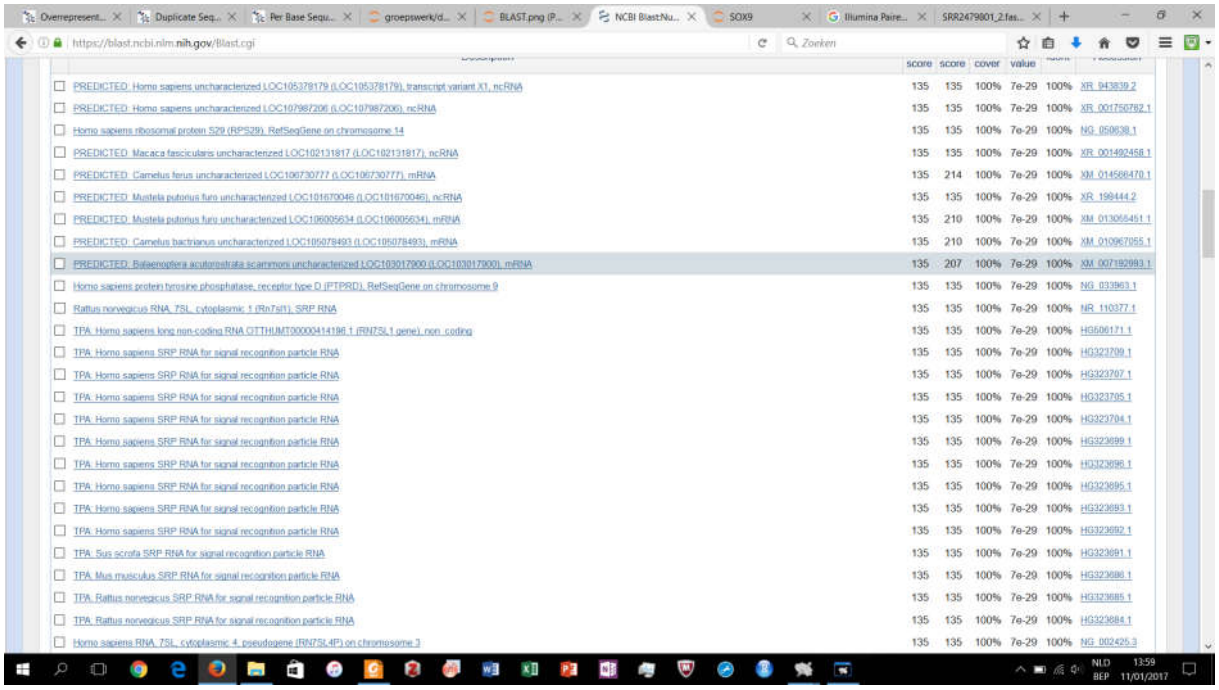

https://blast.ncbi.nlm.nih.gov/Blast.cgi
score
cover
value

In [4]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/
```

```
Started analysis of SRR2479801_2.fastq
Approx 5% complete for SRR2479801_2.fastq
Approx 10% complete for SRR2479801_2.fastq
Approx 15% complete for SRR2479801_2.fastq
Approx 20% complete for SRR2479801_2.fastq
Approx 25% complete for SRR2479801_2.fastq
Approx 30% complete for SRR2479801_2.fastq
Approx 35% complete for SRR2479801_2.fastq
Approx 40% complete for SRR2479801_2.fastq
Approx 45% complete for SRR2479801_2.fastq
Approx 50% complete for SRR2479801_2.fastq
Approx 55% complete for SRR2479801_2.fastq
Approx 60% complete for SRR2479801_2.fastq
Approx 65% complete for SRR2479801_2.fastq
Approx 70% complete for SRR2479801_2.fastq
Approx 75% complete for SRR2479801_2.fastq
Approx 80% complete for SRR2479801_2.fastq
Approx 85% complete for SRR2479801_2.fastq
Approx 90% complete for SRR2479801_2.fastq
Approx 95% complete for SRR2479801_2.fastq
Analysis complete for SRR2479801_2.fastq
```

The per base sequence quality is ok but there are two warnings again concerning the duplication levels and overrepresented sequences. In this case the percentage of sequences that remain after removing duplicates is 62.2%. The sequence duplication levels are most likely the result of oversequencing to be able to detect lowly expressed transcripts. Several overrepresented sequences were detected. We blasted these sequences and all of them lead us to practically the same results (shown under the fastq quality output). BLAST output also seem to point to human or rat contamination as in the previous fastq file.

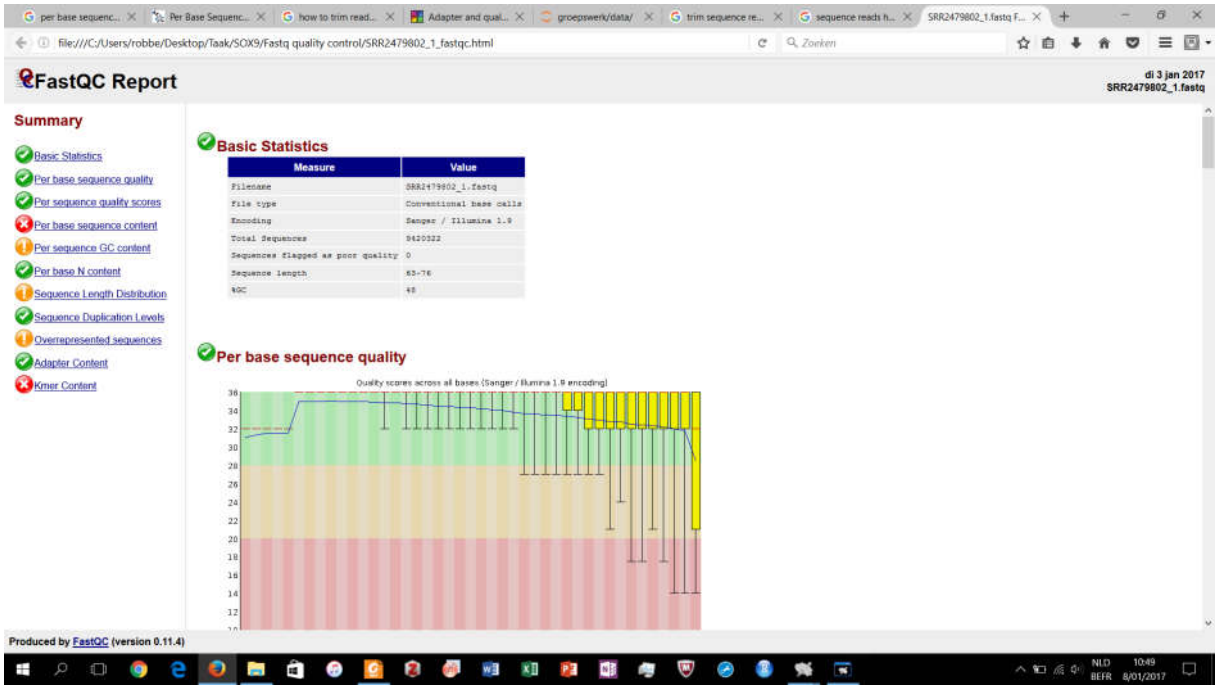




In [162]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479802_1.fastq
Approx 5% complete for SRR2479802_1.fastq
Approx 10% complete for SRR2479802_1.fastq
Approx 15% complete for SRR2479802_1.fastq
Approx 20% complete for SRR2479802_1.fastq
Approx 25% complete for SRR2479802_1.fastq
Approx 30% complete for SRR2479802_1.fastq
Approx 35% complete for SRR2479802_1.fastq
Approx 40% complete for SRR2479802_1.fastq
Approx 45% complete for SRR2479802_1.fastq
Approx 50% complete for SRR2479802_1.fastq
Approx 55% complete for SRR2479802_1.fastq
Approx 60% complete for SRR2479802_1.fastq
Approx 65% complete for SRR2479802_1.fastq
Approx 70% complete for SRR2479802_1.fastq
Approx 75% complete for SRR2479802_1.fastq
Approx 80% complete for SRR2479802_1.fastq
Approx 85% complete for SRR2479802_1.fastq
Approx 90% complete for SRR2479802_1.fastq
Approx 95% complete for SRR2479802_1.fastq
Analysis complete for SRR2479802_1.fastq
```

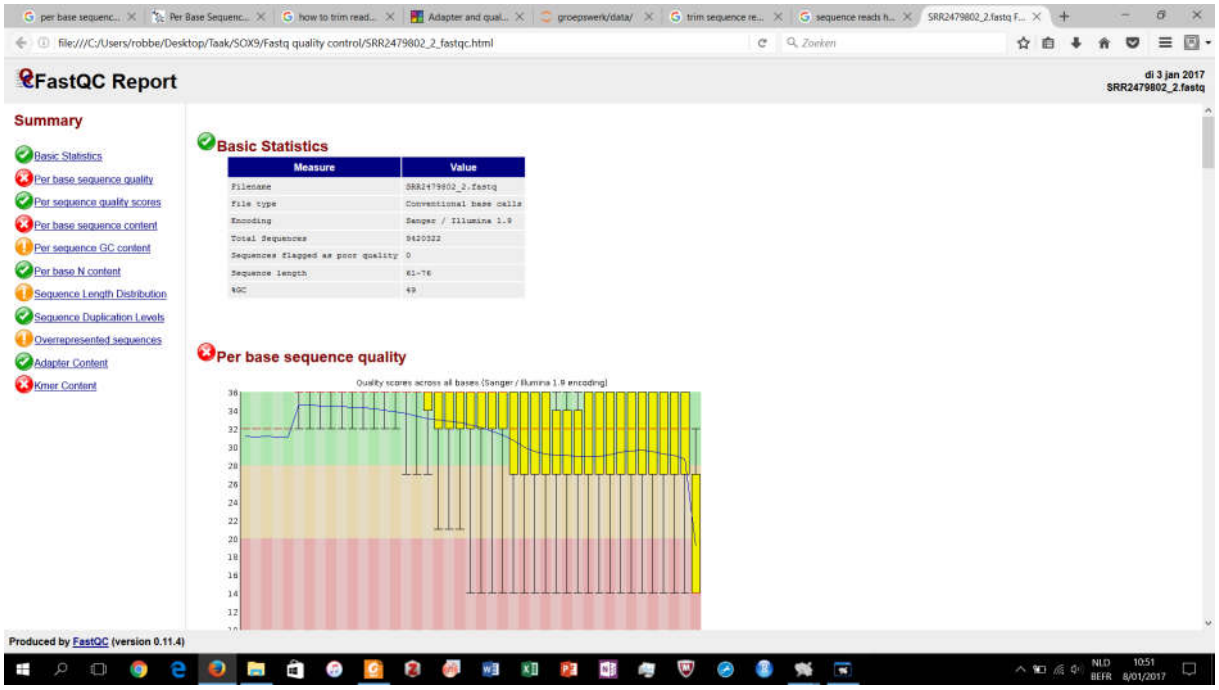
The per base sequence quality and duplication levels are both good. There is warning about overrepresented sequences. One of sequences was an adapter. Ideally we would trim the reads to remove the adapter and re-run the fastq quality control again. But since in this case the overrepresented sequence was less then 0.9% and because we didn't want to risks trimming to much, we decided not to do this. There was another overrepresented sequence with no hit which we blasted. BLAST was unable to match this sequence.

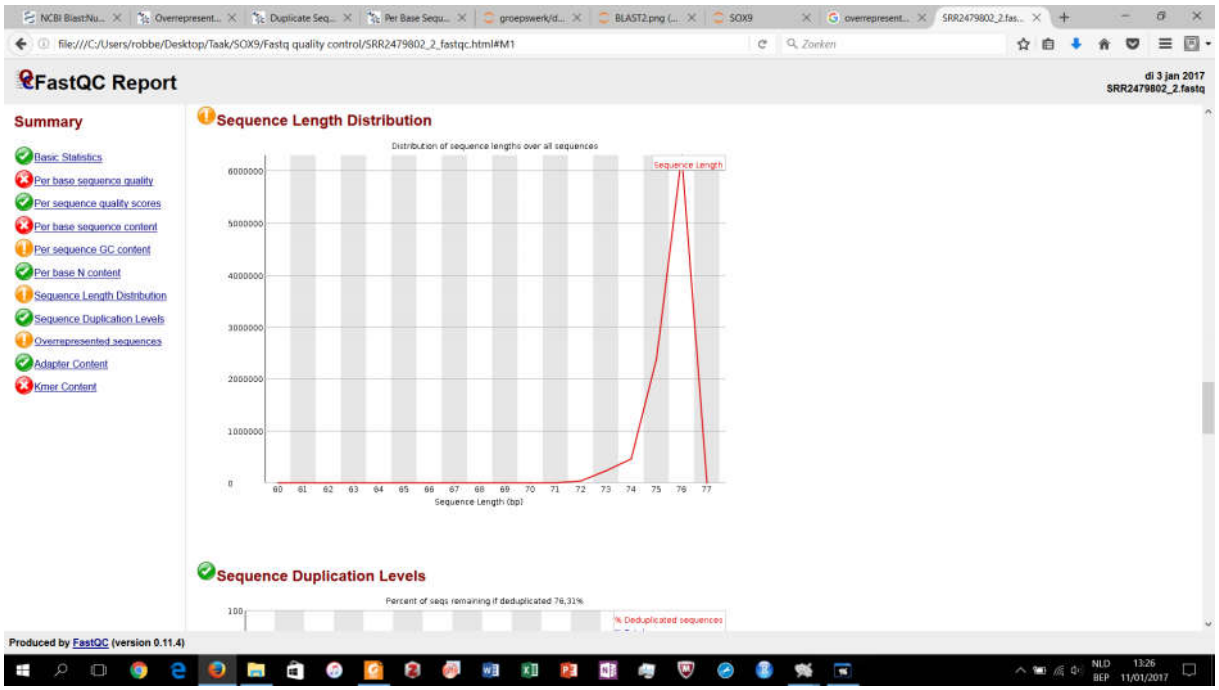


In [163]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479802_2.fastq
Approx 5% complete for SRR2479802_2.fastq
Approx 10% complete for SRR2479802_2.fastq
Approx 15% complete for SRR2479802_2.fastq
Approx 20% complete for SRR2479802_2.fastq
Approx 25% complete for SRR2479802_2.fastq
Approx 30% complete for SRR2479802_2.fastq
Approx 35% complete for SRR2479802_2.fastq
Approx 40% complete for SRR2479802_2.fastq
Approx 45% complete for SRR2479802_2.fastq
Approx 50% complete for SRR2479802_2.fastq
Approx 55% complete for SRR2479802_2.fastq
Approx 60% complete for SRR2479802_2.fastq
Approx 65% complete for SRR2479802_2.fastq
Approx 70% complete for SRR2479802_2.fastq
Approx 75% complete for SRR2479802_2.fastq
Approx 80% complete for SRR2479802_2.fastq
Approx 85% complete for SRR2479802_2.fastq
Approx 90% complete for SRR2479802_2.fastq
Approx 95% complete for SRR2479802_2.fastq
Analysis complete for SRR2479802_2.fastq
```

As you can see below, this fastq file has a fail for per base sequence quality. This module gives a fail when the lower quartile for any base is less than 5 or if the median for any base is less than 20. In our case the fail was generated because the median of the last base was lower then 20. We believe that is caused because the reads have different lengths leading to a lower coverage for this base position. We then checked the sequence length distribution graph. As you can see there are quite some sequence that have only 74 and 75 reads. This leads to a lower converage of the 76th base location leading to the drop in quality. Therefore we believe it is not necessary to trim this last base location. There is also a warning concerning overrepresented sequences. There was one overrepresented sequence with no hit. We blasted this sequence but this also gave us no hit.

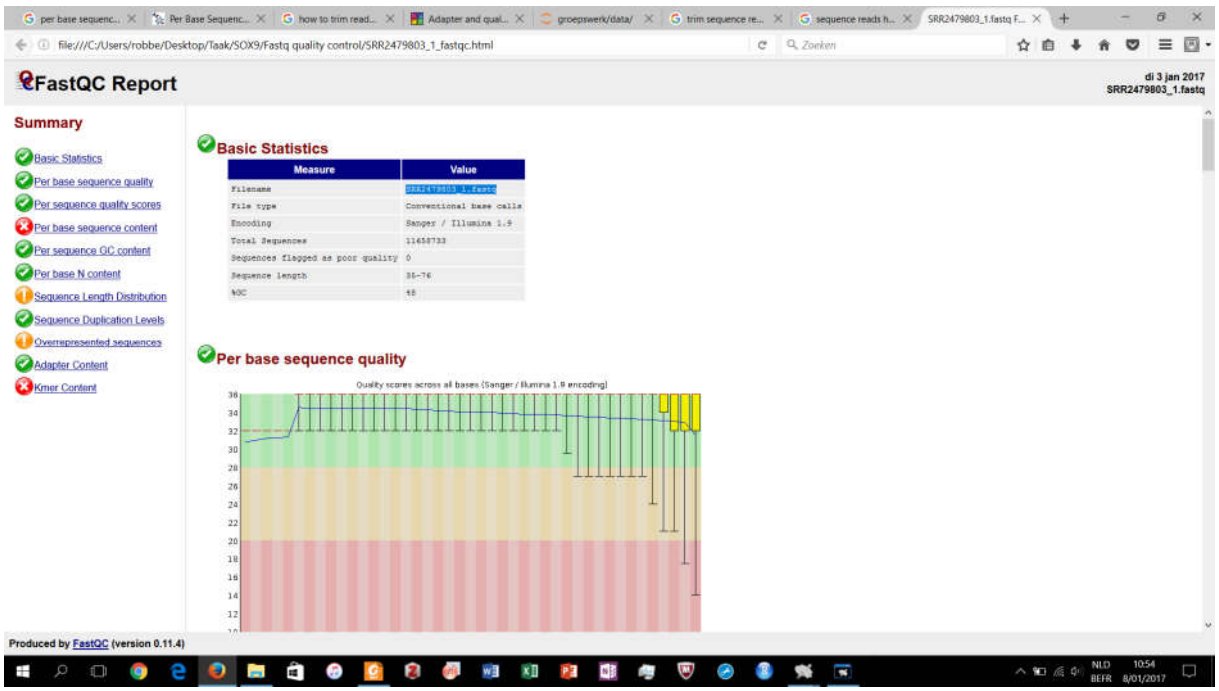

FastQC Report
di 3 jan 2017
SRR2479802_2.fastq
Summary
Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Kmer Content
Basic Statistics
Measure
Value
Filename
SRR2479802_2.fastq
File type
Conventional base calls
Encoding
Sanger / Illumina 1.9
Total Sequences
9420322
Sequences flagged as poor quality
0
Sequence length
61-76
%GC
49
Per base sequence quality
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Produced by FastQC (version 0.11.4)


FastQC Report
di 3 jan 2017
SRR2479802_2.fastq
Summary
Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Kmer Content
Sequence Length Distribution
Distribution of sequence lengths over all sequences
Sequence Length
Sequence Length (bp)
Sequence Duplication Levels
Percent of seqs remaining if deduplicated 76.31%
% Deduplicated sequences
Produced by FastQC (version 0.11.4)

In [164]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479803_1.fastq
Approx 5% complete for SRR2479803_1.fastq
Approx 10% complete for SRR2479803_1.fastq
Approx 15% complete for SRR2479803_1.fastq
Approx 20% complete for SRR2479803_1.fastq
Approx 25% complete for SRR2479803_1.fastq
Approx 30% complete for SRR2479803_1.fastq
Approx 35% complete for SRR2479803_1.fastq
Approx 40% complete for SRR2479803_1.fastq
Approx 45% complete for SRR2479803_1.fastq
Approx 50% complete for SRR2479803_1.fastq
Approx 55% complete for SRR2479803_1.fastq
Approx 60% complete for SRR2479803_1.fastq
Approx 65% complete for SRR2479803_1.fastq
Approx 70% complete for SRR2479803_1.fastq
Approx 75% complete for SRR2479803_1.fastq
Approx 80% complete for SRR2479803_1.fastq
Approx 85% complete for SRR2479803_1.fastq
Approx 90% complete for SRR2479803_1.fastq
Approx 95% complete for SRR2479803_1.fastq
Analysis complete for SRR2479803_1.fastq
```

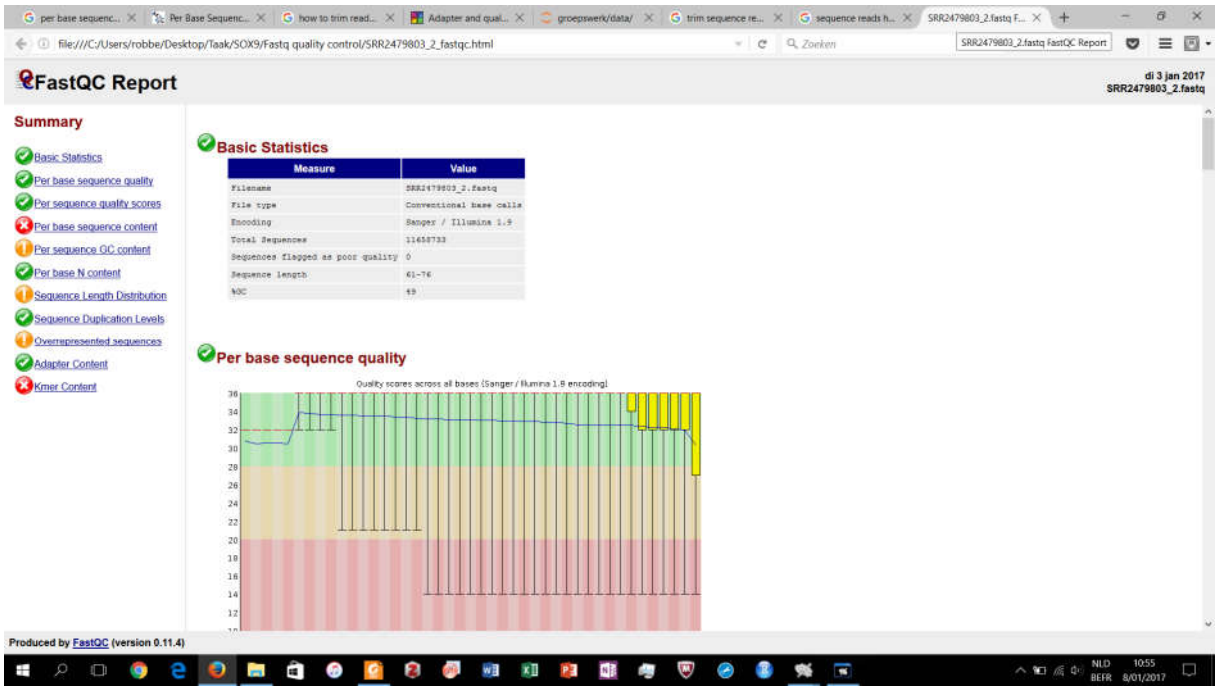
The per base sequence quality and sequence duplication levels are good. There is a warning about the overrepresented sequences. One of them was an adapter and accounted for less then 0.9% of the total. We decided not to trim this because we didn't want to risk overtrimming. There was another overrepresented sequence with no hit. We blasted this sequence but we found no hit.



In [165]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479803_2.fastq
Approx 5% complete for SRR2479803_2.fastq
Approx 10% complete for SRR2479803_2.fastq
Approx 15% complete for SRR2479803_2.fastq
Approx 20% complete for SRR2479803_2.fastq
Approx 25% complete for SRR2479803_2.fastq
Approx 30% complete for SRR2479803_2.fastq
Approx 35% complete for SRR2479803_2.fastq
Approx 40% complete for SRR2479803_2.fastq
Approx 45% complete for SRR2479803_2.fastq
Approx 50% complete for SRR2479803_2.fastq
Approx 55% complete for SRR2479803_2.fastq
Approx 60% complete for SRR2479803_2.fastq
Approx 65% complete for SRR2479803_2.fastq
Approx 70% complete for SRR2479803_2.fastq
Approx 75% complete for SRR2479803_2.fastq
Approx 80% complete for SRR2479803_2.fastq
Approx 85% complete for SRR2479803_2.fastq
Approx 90% complete for SRR2479803_2.fastq
Approx 95% complete for SRR2479803_2.fastq
Analysis complete for SRR2479803_2.fastq
```

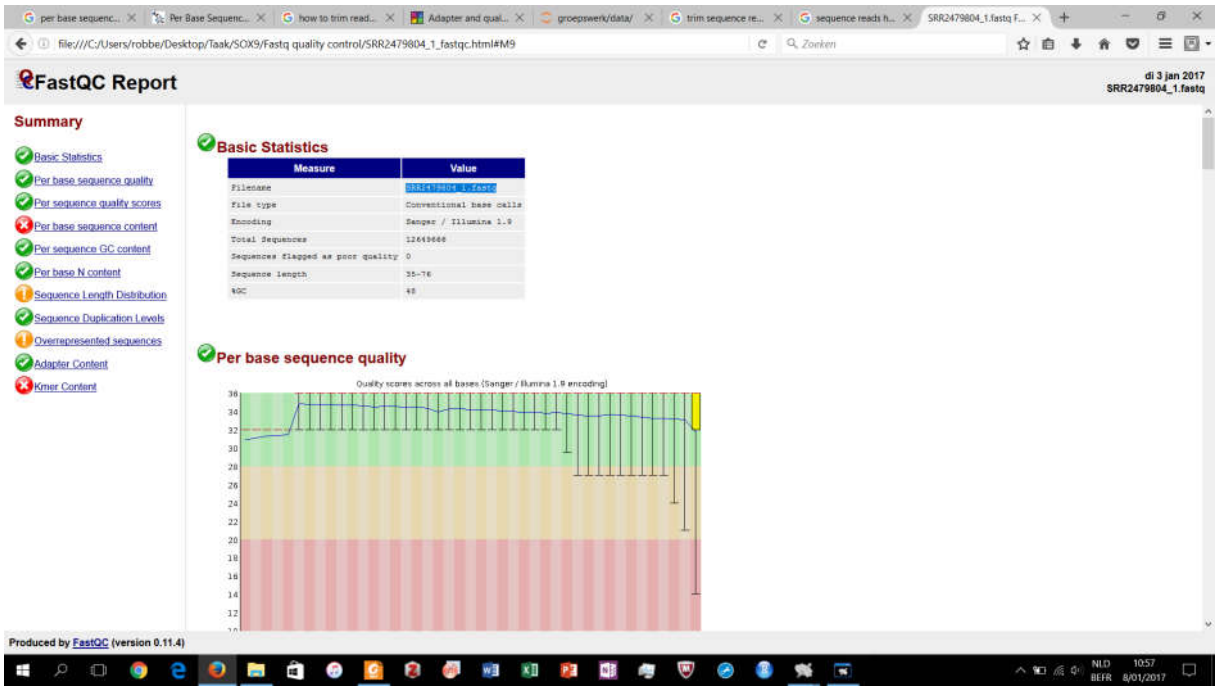
The per base sequence quality and duplication levels look fine. But again we get a warning for overrepresented sequences. This overrepresented sequence had no hit so we decided to blast the sequence. BLAST was also unable to find a hit for the overrepresented sequence.



In [166]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479804_1.fastq
Approx 5% complete for SRR2479804_1.fastq
Approx 10% complete for SRR2479804_1.fastq
Approx 15% complete for SRR2479804_1.fastq
Approx 20% complete for SRR2479804_1.fastq
Approx 25% complete for SRR2479804_1.fastq
Approx 30% complete for SRR2479804_1.fastq
Approx 35% complete for SRR2479804_1.fastq
Approx 40% complete for SRR2479804_1.fastq
Approx 45% complete for SRR2479804_1.fastq
Approx 50% complete for SRR2479804_1.fastq
Approx 55% complete for SRR2479804_1.fastq
Approx 60% complete for SRR2479804_1.fastq
Approx 65% complete for SRR2479804_1.fastq
Approx 70% complete for SRR2479804_1.fastq
Approx 75% complete for SRR2479804_1.fastq
Approx 80% complete for SRR2479804_1.fastq
Approx 85% complete for SRR2479804_1.fastq
Approx 90% complete for SRR2479804_1.fastq
Approx 95% complete for SRR2479804_1.fastq
Analysis complete for SRR2479804_1.fastq
```

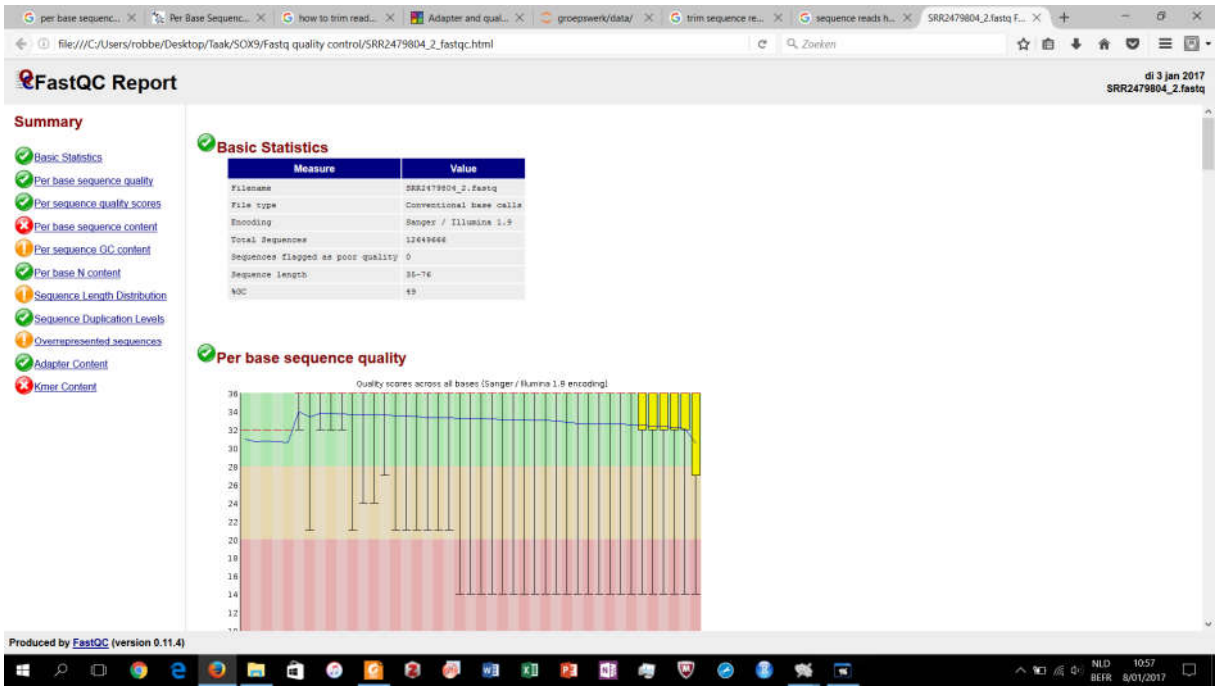
The per base sequence quality and sequence duplication levels are good. But this time there is also a warning for overrepresented sequences. One of the overrepresented sequences is an adaptor which represent less than 0.8% of the total sequences. We didn't trim this because of reasons mentioned earlier (only 0.8% and fear of overtrimming). The other overrepresented sequence had no hit. We blasted this sequence and also found no hits.



In [167]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479804_2.fastq
Approx 5% complete for SRR2479804_2.fastq
Approx 10% complete for SRR2479804_2.fastq
Approx 15% complete for SRR2479804_2.fastq
Approx 20% complete for SRR2479804_2.fastq
Approx 25% complete for SRR2479804_2.fastq
Approx 30% complete for SRR2479804_2.fastq
Approx 35% complete for SRR2479804_2.fastq
Approx 40% complete for SRR2479804_2.fastq
Approx 45% complete for SRR2479804_2.fastq
Approx 50% complete for SRR2479804_2.fastq
Approx 55% complete for SRR2479804_2.fastq
Approx 60% complete for SRR2479804_2.fastq
Approx 65% complete for SRR2479804_2.fastq
Approx 70% complete for SRR2479804_2.fastq
Approx 75% complete for SRR2479804_2.fastq
Approx 80% complete for SRR2479804_2.fastq
Approx 85% complete for SRR2479804_2.fastq
Approx 90% complete for SRR2479804_2.fastq
Approx 95% complete for SRR2479804_2.fastq
Analysis complete for SRR2479804_2.fastq
```

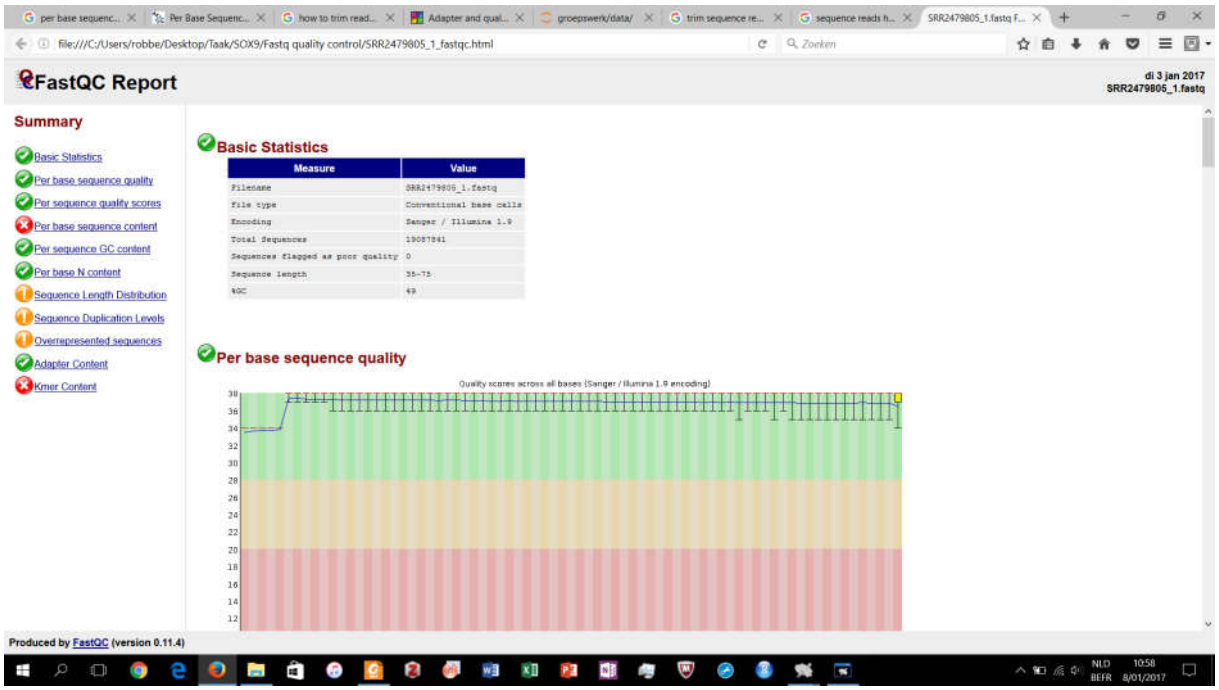
The per base sequence quality and duplication levels are okay. There is a warning concerning overrepresented sequences. This overrepresented sequence had no hit so we blasted the sequence. Blast was unable to find any hits.

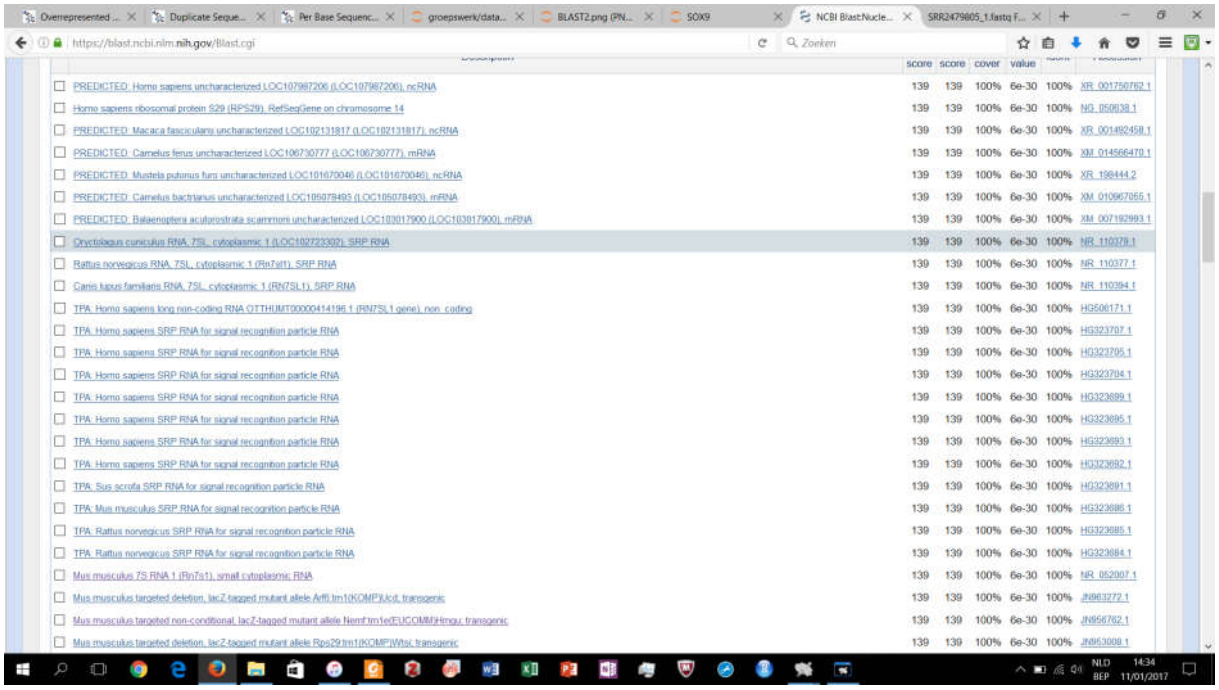


In [168]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479805_1.fastq
Approx 5% complete for SRR2479805_1.fastq
Approx 10% complete for SRR2479805_1.fastq
Approx 15% complete for SRR2479805_1.fastq
Approx 20% complete for SRR2479805_1.fastq
Approx 25% complete for SRR2479805_1.fastq
Approx 30% complete for SRR2479805_1.fastq
Approx 35% complete for SRR2479805_1.fastq
Approx 40% complete for SRR2479805_1.fastq
Approx 45% complete for SRR2479805_1.fastq
Approx 50% complete for SRR2479805_1.fastq
Approx 55% complete for SRR2479805_1.fastq
Approx 60% complete for SRR2479805_1.fastq
Approx 65% complete for SRR2479805_1.fastq
Approx 70% complete for SRR2479805_1.fastq
Approx 75% complete for SRR2479805_1.fastq
Approx 80% complete for SRR2479805_1.fastq
Approx 85% complete for SRR2479805_1.fastq
Approx 90% complete for SRR2479805_1.fastq
Approx 95% complete for SRR2479805_1.fastq
Analysis complete for SRR2479805_1.fastq
```

The per base sequence quality is good but there are warnings about the duplication levels and overrepresented sequences. In this case the percentage of sequences that remain after removing duplicates is 58.09%. The sequence duplication levels are most likely the result of oversequencing to be able to detect lowly expressed transcripts. There are also several overrepresented sequences without hit. We blasted these sequences and it possible that there is contamination from human, rat or wolf.

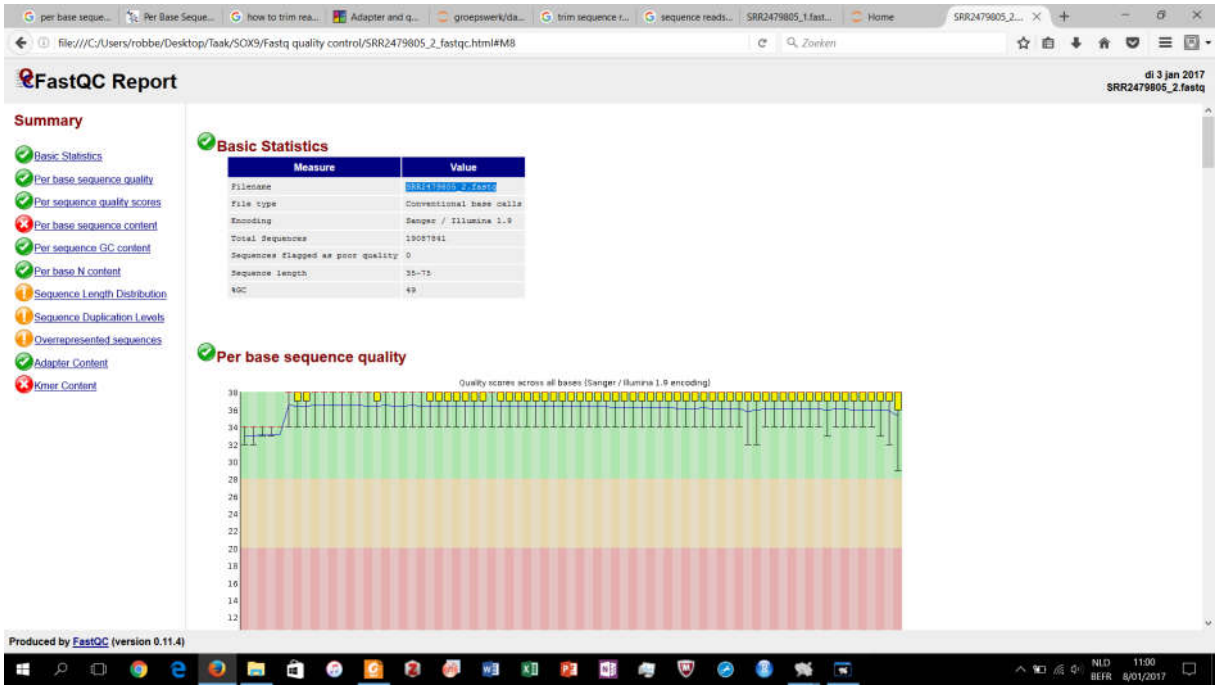


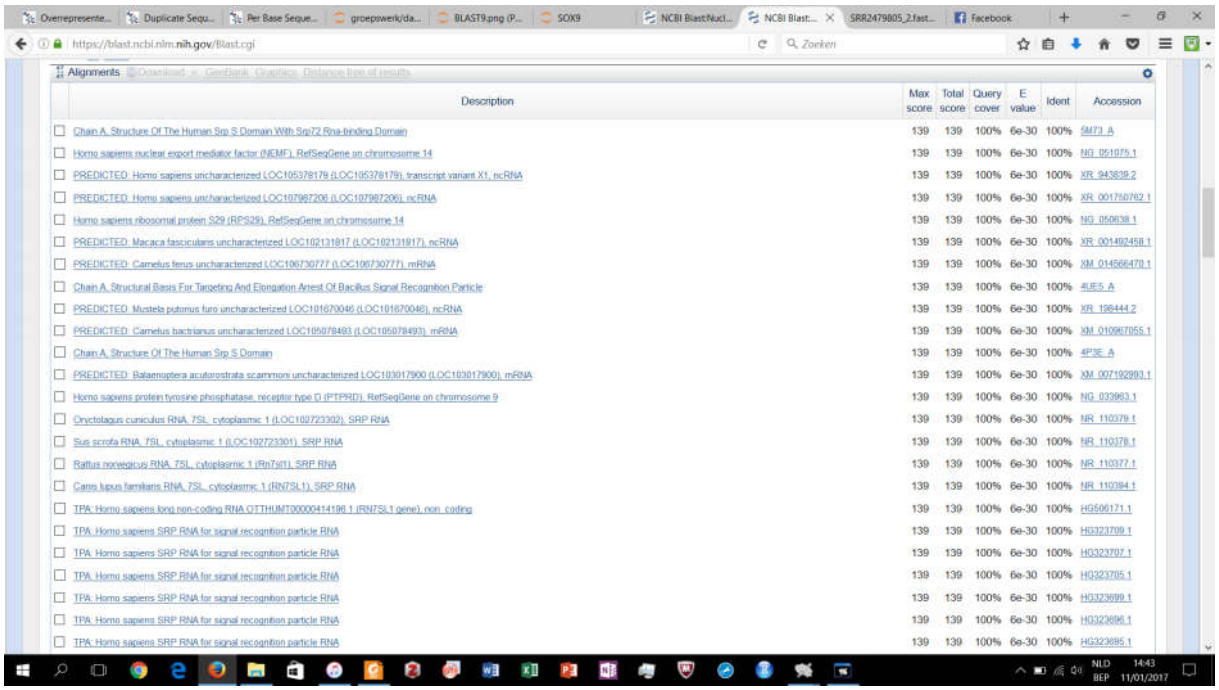


In [169]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479805_2.fastq
Approx 5% complete for SRR2479805_2.fastq
Approx 10% complete for SRR2479805_2.fastq
Approx 15% complete for SRR2479805_2.fastq
Approx 20% complete for SRR2479805_2.fastq
Approx 25% complete for SRR2479805_2.fastq
Approx 30% complete for SRR2479805_2.fastq
Approx 35% complete for SRR2479805_2.fastq
Approx 40% complete for SRR2479805_2.fastq
Approx 45% complete for SRR2479805_2.fastq
Approx 50% complete for SRR2479805_2.fastq
Approx 55% complete for SRR2479805_2.fastq
Approx 60% complete for SRR2479805_2.fastq
Approx 65% complete for SRR2479805_2.fastq
Approx 70% complete for SRR2479805_2.fastq
Approx 75% complete for SRR2479805_2.fastq
Approx 80% complete for SRR2479805_2.fastq
Approx 85% complete for SRR2479805_2.fastq
Approx 90% complete for SRR2479805_2.fastq
Approx 95% complete for SRR2479805_2.fastq
Analysis complete for SRR2479805_2.fastq
```

The per base sequence quality is good but there are issues with the duplication levels and overrepresented sequences. In this case the percentage of sequences that remained after removing duplicates is 59.33%. The sequence duplication levels are most likely the result of oversequencing to be able to detect lowly expressed transcripts. There were several overrepresented sequences with no hit. We therefore blasted these sequences and found evidence pointing to contamination with human, rat or wolf.

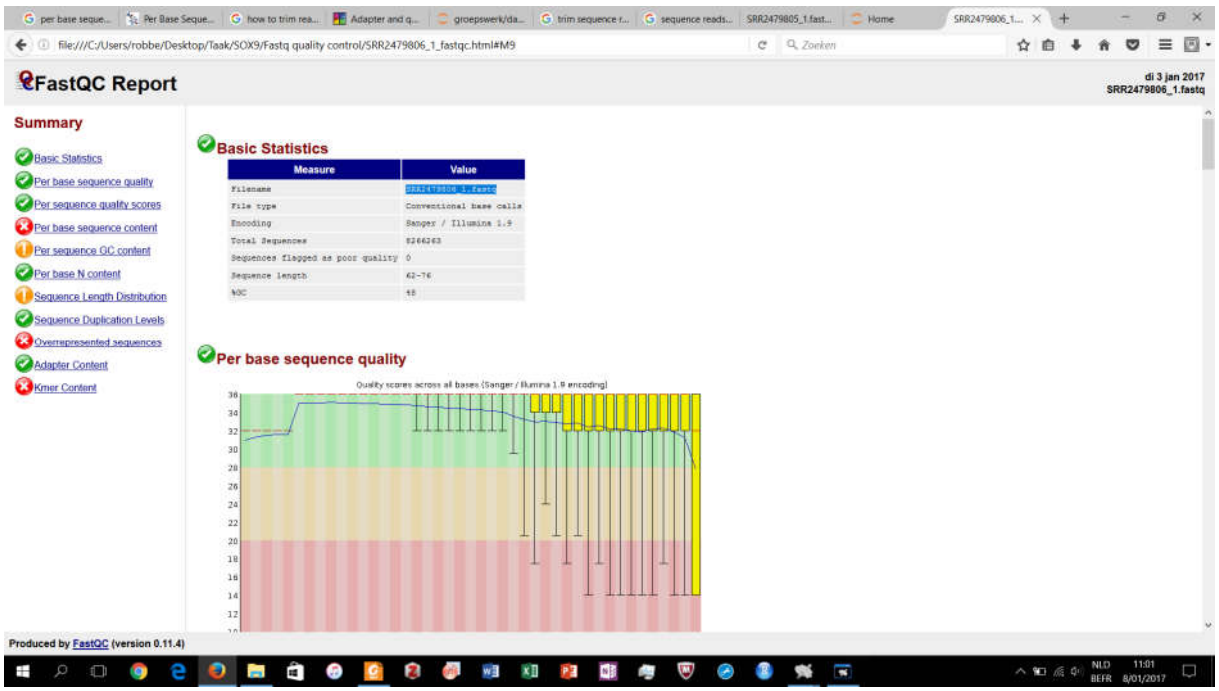

FastQC Report
di 3 jan 2017
SRR2479805_2.fastq
Summary
Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Kmer Content
Basic Statistics
Measure
Value
Filename
SRR2479805_2.fastq
File type
Conventional base calls
Encoding
Sanger / Illumina 1.9
Total Sequences
19087841
Sequences flagged as poor quality
0
Sequence length
35-75
%GC
49
Per base sequence quality
Quality scores across all bases (Sanger / Illumina 1.9 encoding)
Produced by FastQC (version 0.11.4)


https://blast.ncbi.nlm.nih.gov/Blast.cgi
Alignments
Description
Max score
Total score
Query cover
E value
Ident
Accession

In [170]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479806_1.fastq
Approx 5% complete for SRR2479806_1.fastq
Approx 10% complete for SRR2479806_1.fastq
Approx 15% complete for SRR2479806_1.fastq
Approx 20% complete for SRR2479806_1.fastq
Approx 25% complete for SRR2479806_1.fastq
Approx 30% complete for SRR2479806_1.fastq
Approx 35% complete for SRR2479806_1.fastq
Approx 40% complete for SRR2479806_1.fastq
Approx 45% complete for SRR2479806_1.fastq
Approx 50% complete for SRR2479806_1.fastq
Approx 55% complete for SRR2479806_1.fastq
Approx 60% complete for SRR2479806_1.fastq
Approx 65% complete for SRR2479806_1.fastq
Approx 70% complete for SRR2479806_1.fastq
Approx 75% complete for SRR2479806_1.fastq
Approx 80% complete for SRR2479806_1.fastq
Approx 85% complete for SRR2479806_1.fastq
Approx 90% complete for SRR2479806_1.fastq
Approx 95% complete for SRR2479806_1.fastq
Analysis complete for SRR2479806_1.fastq
```

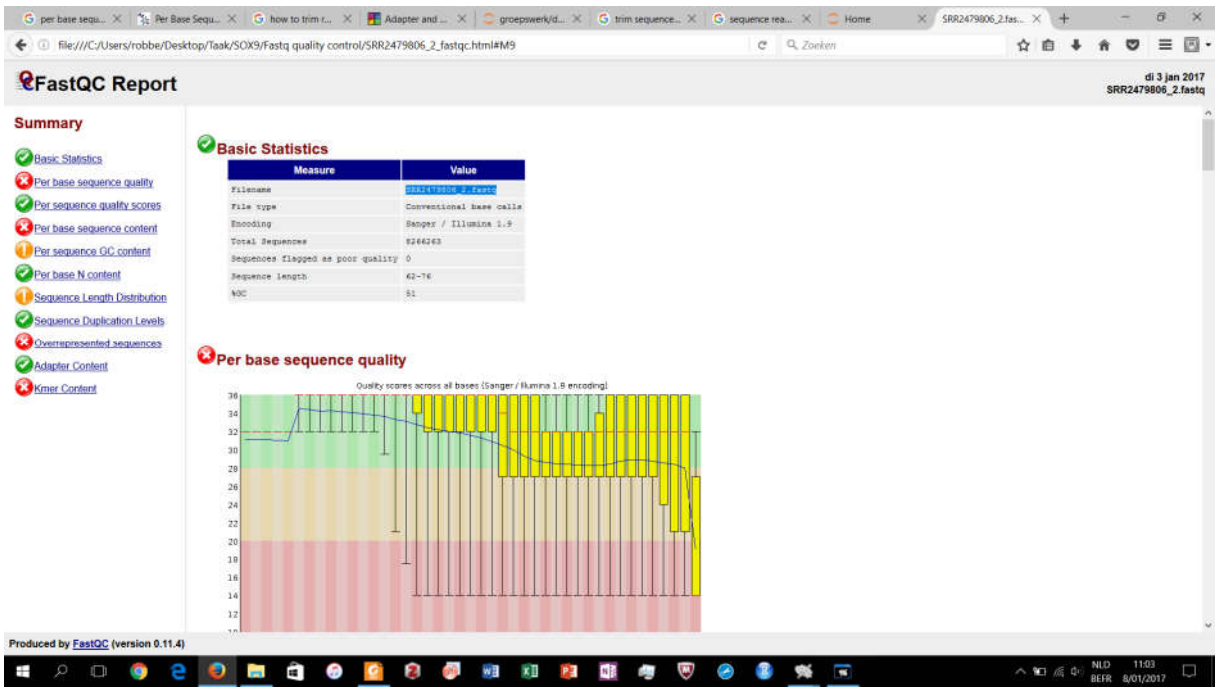
The per base sequence quality and duplication levels good. But now we have a failure in the overrepresented sequences because there is an overrepresented sequence that represent more than 1% of the total sequences. This overrepresented sequence is an adaptor and we decide not to trim this because we didn't want to risk overtrimming.

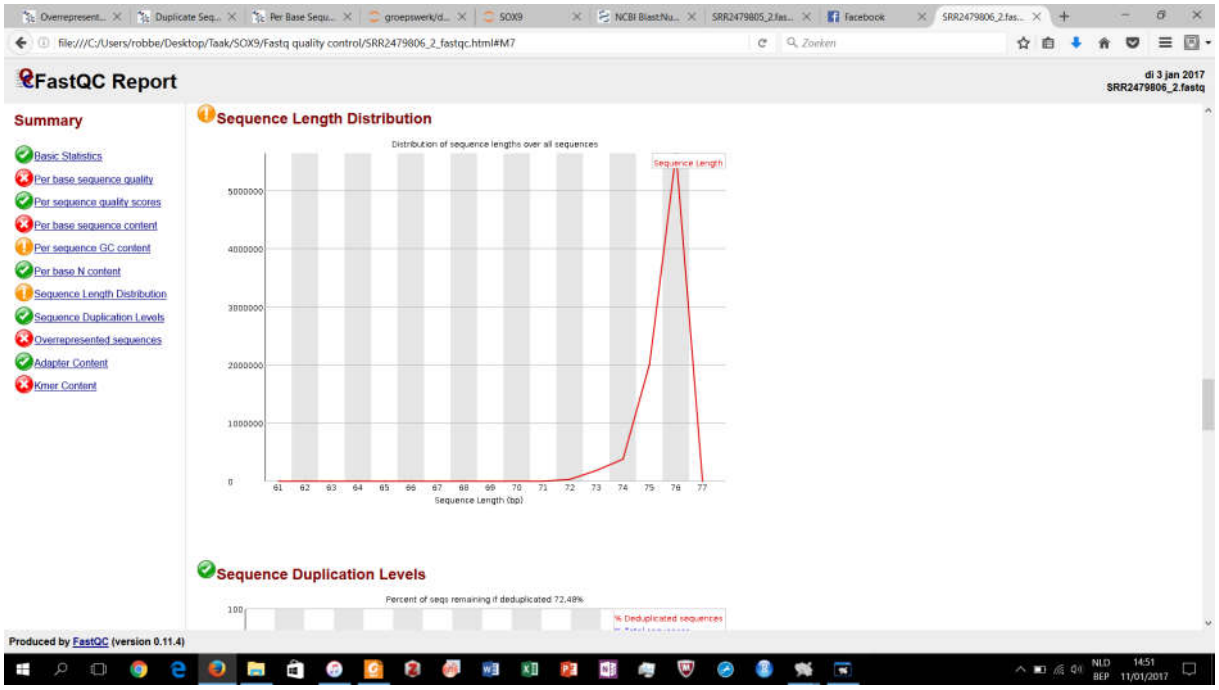


In [171]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479806_2.fastq
Approx 5% complete for SRR2479806_2.fastq
Approx 10% complete for SRR2479806_2.fastq
Approx 15% complete for SRR2479806_2.fastq
Approx 20% complete for SRR2479806_2.fastq
Approx 25% complete for SRR2479806_2.fastq
Approx 30% complete for SRR2479806_2.fastq
Approx 35% complete for SRR2479806_2.fastq
Approx 40% complete for SRR2479806_2.fastq
Approx 45% complete for SRR2479806_2.fastq
Approx 50% complete for SRR2479806_2.fastq
Approx 55% complete for SRR2479806_2.fastq
Approx 60% complete for SRR2479806_2.fastq
Approx 65% complete for SRR2479806_2.fastq
Approx 70% complete for SRR2479806_2.fastq
Approx 75% complete for SRR2479806_2.fastq
Approx 80% complete for SRR2479806_2.fastq
Approx 85% complete for SRR2479806_2.fastq
Approx 90% complete for SRR2479806_2.fastq
Approx 95% complete for SRR2479806_2.fastq
Analysis complete for SRR2479806_2.fastq
```

As you can see the per base sequence quality and overrepresented sequences both have a fail. There were four represented sequences which differed in only one amino acid. We blasted the one which caused the failure but we found no similarities. The fail concerning the per base sequence quality is generated because the mean value of the last base position is below 20. We believe that this is caused because the reads have different lengths leading to a lower coverage for this base position. We then checked the sequence lenght distribution graph. As you can see there quit some sequence that have only 74 and 75 reads. This leads to a lower converage of the 76th base location leading to the drop in quality. Therefore we believe it is not necessary to trim this last base location.

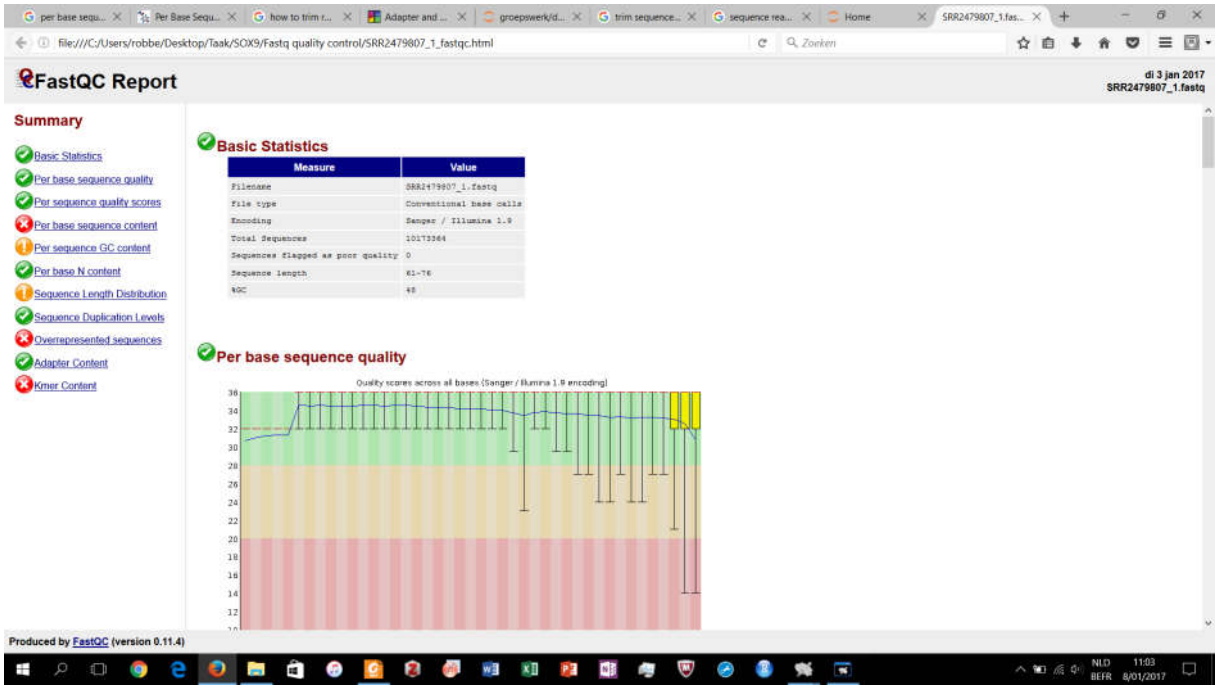




In [172]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479807_1.fastq
Approx 5% complete for SRR2479807_1.fastq
Approx 10% complete for SRR2479807_1.fastq
Approx 15% complete for SRR2479807_1.fastq
Approx 20% complete for SRR2479807_1.fastq
Approx 25% complete for SRR2479807_1.fastq
Approx 30% complete for SRR2479807_1.fastq
Approx 35% complete for SRR2479807_1.fastq
Approx 40% complete for SRR2479807_1.fastq
Approx 45% complete for SRR2479807_1.fastq
Approx 50% complete for SRR2479807_1.fastq
Approx 55% complete for SRR2479807_1.fastq
Approx 60% complete for SRR2479807_1.fastq
Approx 65% complete for SRR2479807_1.fastq
Approx 70% complete for SRR2479807_1.fastq
Approx 75% complete for SRR2479807_1.fastq
Approx 80% complete for SRR2479807_1.fastq
Approx 85% complete for SRR2479807_1.fastq
Approx 90% complete for SRR2479807_1.fastq
Approx 95% complete for SRR2479807_1.fastq
Analysis complete for SRR2479807_1.fastq
```

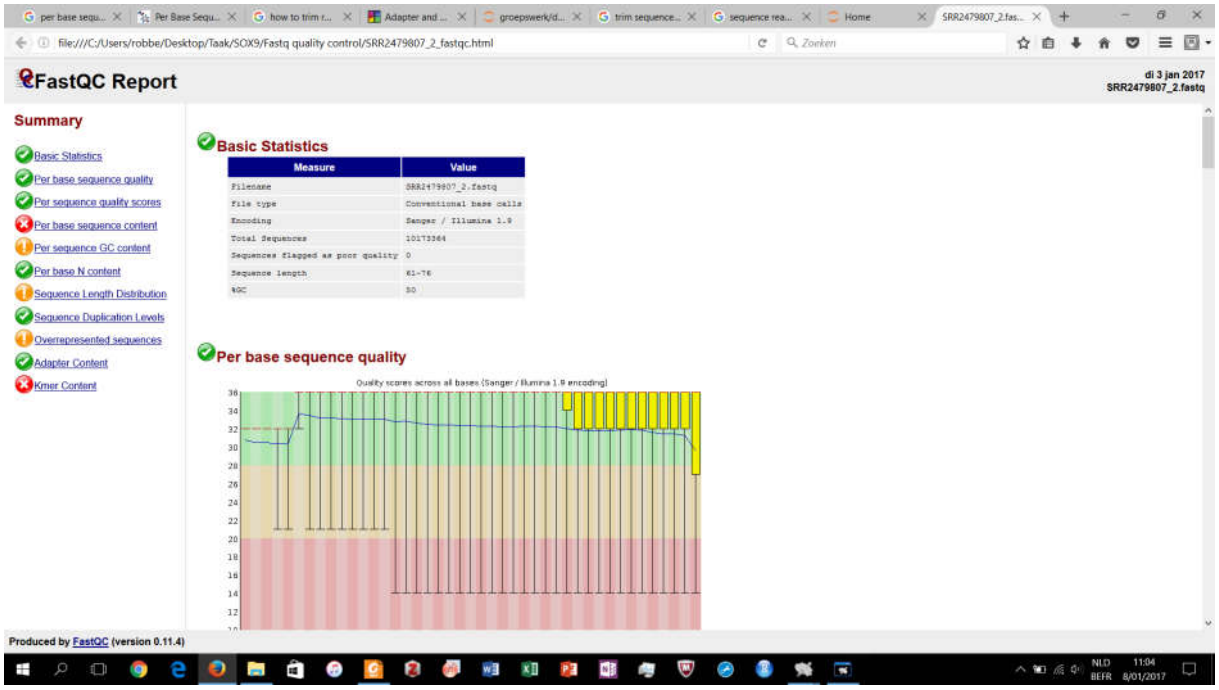
The per base sequence quality and duplication levels are good. But there is a failure concering overrepresented sequences. Four of these overrepresented sequences are adaptors and we decided not to trim this because we didn't want to risk overtrimming (one of these adaptars generated the failure). Another overrepresented sequence without hit was blasted. BLAST was also unable to find any hits.



In [173]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479807_2.fastq
Approx 5% complete for SRR2479807_2.fastq
Approx 10% complete for SRR2479807_2.fastq
Approx 15% complete for SRR2479807_2.fastq
Approx 20% complete for SRR2479807_2.fastq
Approx 25% complete for SRR2479807_2.fastq
Approx 30% complete for SRR2479807_2.fastq
Approx 35% complete for SRR2479807_2.fastq
Approx 40% complete for SRR2479807_2.fastq
Approx 45% complete for SRR2479807_2.fastq
Approx 50% complete for SRR2479807_2.fastq
Approx 55% complete for SRR2479807_2.fastq
Approx 60% complete for SRR2479807_2.fastq
Approx 65% complete for SRR2479807_2.fastq
Approx 70% complete for SRR2479807_2.fastq
Approx 75% complete for SRR2479807_2.fastq
Approx 80% complete for SRR2479807_2.fastq
Approx 85% complete for SRR2479807_2.fastq
Approx 90% complete for SRR2479807_2.fastq
Approx 95% complete for SRR2479807_2.fastq
Analysis complete for SRR2479807_2.fastq
```

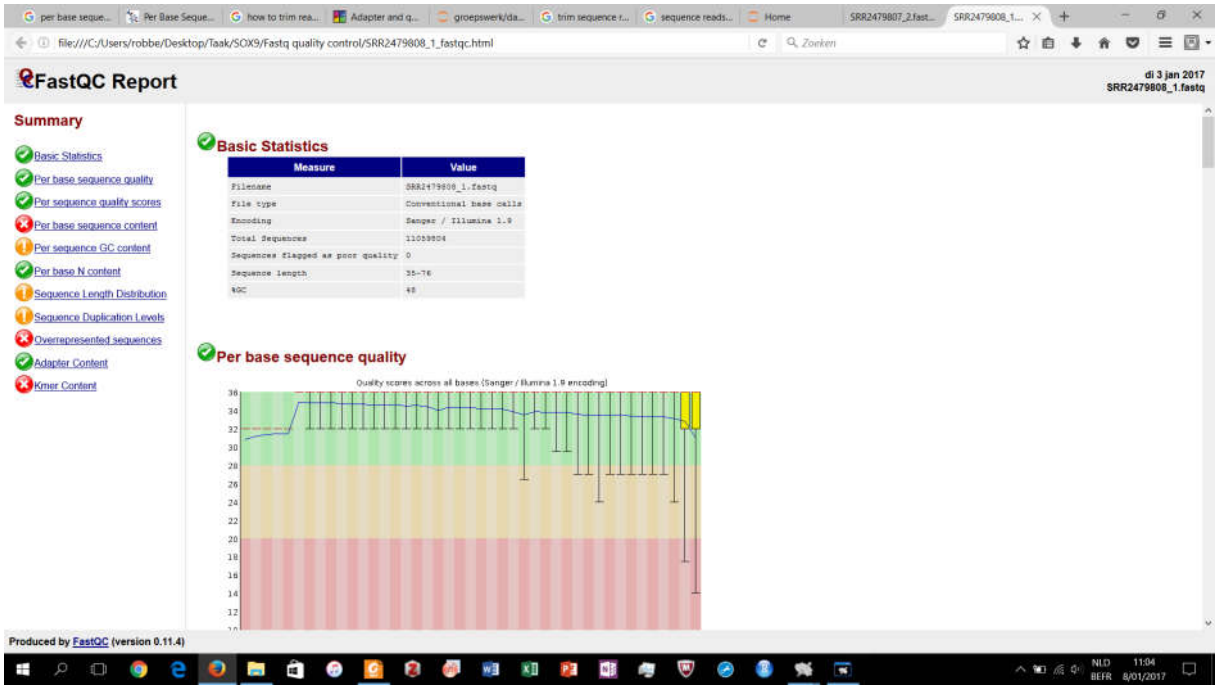
The per base sequence quality score and duplication levels are fine. There is a warning about overrepresented sequences. One of the two overrepresented sequences is an adaptor. But since this overrepresented sequence only represent 0.4% of the total sequences and because of the risk for overtrimming we decided not to trim this. The other overrepresented sequence had no hit and was blasted. This blast step did not found any hits.



In [189]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479808_1.fastq
Approx 5% complete for SRR2479808_1.fastq
Approx 10% complete for SRR2479808_1.fastq
Approx 15% complete for SRR2479808_1.fastq
Approx 20% complete for SRR2479808_1.fastq
Approx 25% complete for SRR2479808_1.fastq
Approx 30% complete for SRR2479808_1.fastq
Approx 35% complete for SRR2479808_1.fastq
Approx 40% complete for SRR2479808_1.fastq
Approx 45% complete for SRR2479808_1.fastq
Approx 50% complete for SRR2479808_1.fastq
Approx 55% complete for SRR2479808_1.fastq
Approx 60% complete for SRR2479808_1.fastq
Approx 65% complete for SRR2479808_1.fastq
Approx 70% complete for SRR2479808_1.fastq
Approx 75% complete for SRR2479808_1.fastq
Approx 80% complete for SRR2479808_1.fastq
Approx 85% complete for SRR2479808_1.fastq
Approx 90% complete for SRR2479808_1.fastq
Approx 95% complete for SRR2479808_1.fastq
Analysis complete for SRR2479808_1.fastq
```

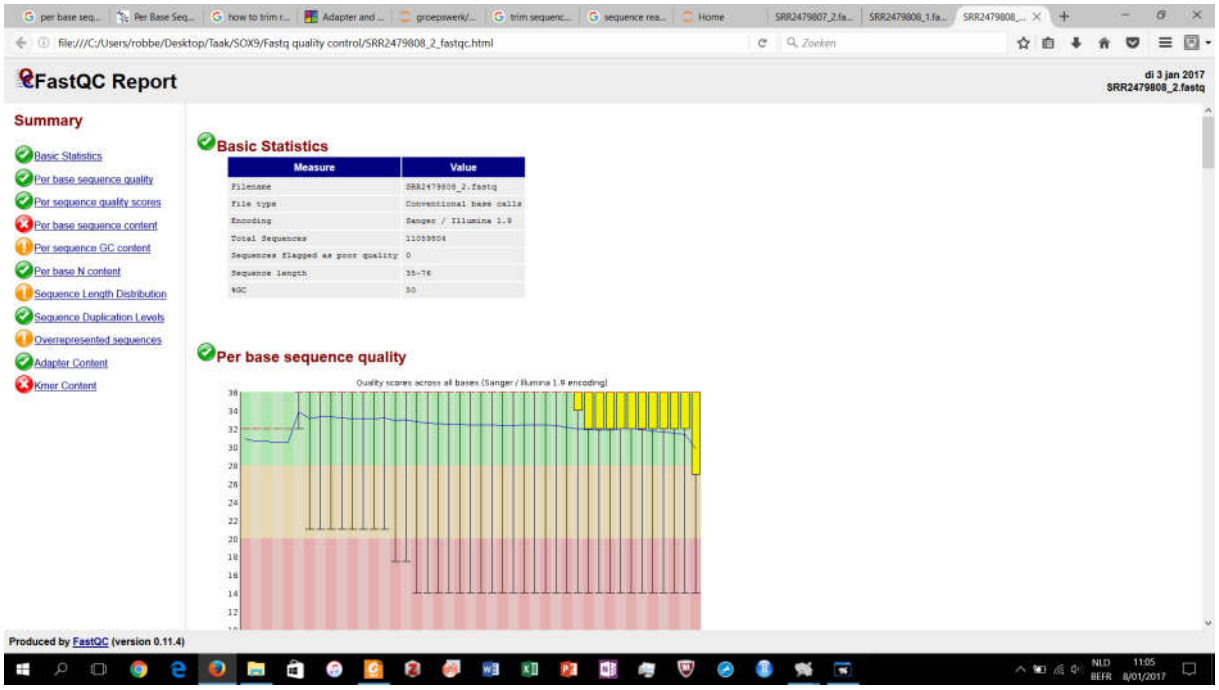
The per base sequence quality is good. There is warning for duplication levels. In this case the percentage of sequences that remained after removing duplicates is 69.89%. The sequence duplication levels are most likely the result of oversequencing to be able to detect lowly expressed transcripts. There is also a failure regarding overrepresented sequences. Three of them are adaptors. These weren't trimmed for reasons mentioned earlier. The other overrepresented sequence had no hit and BLAST was unable to find any hits.



In [178]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/data/quality
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/data
```

```
Started analysis of SRR2479808_2.fastq
Approx 5% complete for SRR2479808_2.fastq
Approx 10% complete for SRR2479808_2.fastq
Approx 15% complete for SRR2479808_2.fastq
Approx 20% complete for SRR2479808_2.fastq
Approx 25% complete for SRR2479808_2.fastq
Approx 30% complete for SRR2479808_2.fastq
Approx 35% complete for SRR2479808_2.fastq
Approx 40% complete for SRR2479808_2.fastq
Approx 45% complete for SRR2479808_2.fastq
Approx 50% complete for SRR2479808_2.fastq
Approx 55% complete for SRR2479808_2.fastq
Approx 60% complete for SRR2479808_2.fastq
Approx 65% complete for SRR2479808_2.fastq
Approx 70% complete for SRR2479808_2.fastq
Approx 75% complete for SRR2479808_2.fastq
Approx 80% complete for SRR2479808_2.fastq
Approx 85% complete for SRR2479808_2.fastq
Approx 90% complete for SRR2479808_2.fastq
Approx 95% complete for SRR2479808_2.fastq
Analysis complete for SRR2479808_2.fastq
```

The per base sequence quality and duplication levels are good. There is a warning concerning overrepresented sequences. One of those hits is an adaptor. For reasons earlier mentioned, this wasn't trimmed. The other overrepresented sequence had no hit. We decided to BLAST this sequence but this give no hits.



Now that we have the reads and checked the quality, we need to look at what we are going to use to map them. In the paper they have used Tophat2 but we opted to go with STAR since STAR delivers the best results according to several papers and fora on the internet and is really fast. In class we used STAR but there we already had a human reference genome to work with. We are working with mus musculus so we need to generate our own reference genome. In the paper they use the mm9 genome. We have opted to work with the mm10 as this is the most recent version. We obtained a mus musculus genome and corresponding annotation file from NCBI (link: https://www.ncbi.nlm.nih.gov/genome?term=mus%20musculus (https://www.ncbi.nlm.nih.gov/genome?term=mus%20musculus)).

The first step was to unpack the reference genome.

In [190]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Sequence
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Sequence
```

Then we also unpacked the annotation file.

In [13]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Sequence
```

Then using the command below, we used the mus musculus genome and corresponding annotation file to generate a reference genome.

runThreadN is the number of processors that will be used.

genomeDir is the directory in which all the output will be stored.

genomeFastaFiles shows the path to the mus musculus genome.

sjdbGTFfile shows the path to the mus musculus annotation file.

sjdgOverhang tells STAR how many bases to link from donor and acceptor sides of the junctions. We used the standard value of 100 since we read on fora that this should work fine for us.

In [191]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR
pwd
STAR --runThreadN 8 --runMode genomeGenerate --genomeDir /mnt/storage/r0305394/jupy
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR
Jan 03 21:15:25 ..... started STAR run
Jan 03 21:15:25 ... starting to generate Genome files
Jan 03 21:16:44 ... starting to sort Suffix Array. This may take a long time...
Jan 03 21:17:02 ... sorting Suffix Array chunks and saving them to disk...
Jan 03 21:40:18 ... loading chunks from disk, packing SA...
Jan 03 21:43:49 ... finished generating suffix array
Jan 03 21:43:49 ... generating Suffix Array index
Jan 03 21:48:06 ... completed Suffix Array index
Jan 03 21:48:06 ..... processing annotations GTF
Jan 03 21:48:29 ..... inserting junctions into the genome indices
Jan 03 21:52:04 ... writing Genome to disk ...
Jan 03 21:52:07 ... writing Suffix Array to disk ...
Jan 03 21:52:52 ... writing SAindex to disk
Jan 03 21:52:58 ..... finished successfully
```

Then the different fasta files containing the reads can be mapped to the reference genome generated in the previous step. We created a new folder for every samples (e.g. SSR2479801). For each samples two fastq files were mapped at the same time since these files are paired-end reads.

genomeDir is the path to the mus musculus reference genome generated earlier.

runThreadN is the number of processors used.

readFilesIn is the path to the fastq files that we want map.

outFileNamePrefix is the name that will be given to output file.

In [192]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
Jan 03 21:53:54 ..... started STAR run
Jan 03 21:53:54 ..... loading genome
Jan 03 21:54:22 ..... started mapping
Jan 03 21:57:22 ..... finished successfully
```

In [193]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479802
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479802
Jan 03 21:58:02 ..... started STAR run
Jan 03 21:58:02 ..... loading genome
Jan 03 21:58:18 ..... started mapping
Jan 03 22:01:22 ..... finished successfully
```

In [194]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479803
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479803
Jan 03 22:01:41 ..... started STAR run
Jan 03 22:01:41 ..... loading genome
Jan 03 22:02:17 ..... started mapping
Jan 03 22:06:36 ..... finished successfully
```

In [195]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479804
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479804
Jan 03 22:06:57 ..... started STAR run
Jan 03 22:06:57 ..... loading genome
Jan 03 22:07:14 ..... started mapping
Jan 03 22:11:46 ..... finished successfully
```

In [196]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479805
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479805
Jan 03 22:23:25 ..... started STAR run
Jan 03 22:23:25 ..... loading genome
Jan 03 22:23:39 ..... started mapping
Jan 03 22:28:27 ..... finished successfully
```

In [197]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479806
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479806
Jan 03 22:29:26 ..... started STAR run
Jan 03 22:29:26 ..... loading genome
Jan 03 22:29:39 ..... started mapping
Jan 03 22:32:37 ..... finished successfully
```

In [198]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479807
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479807
Jan 03 22:33:26 ..... started STAR run
Jan 03 22:33:26 ..... loading genome
Jan 03 22:33:40 ..... started mapping
Jan 03 22:37:09 ..... finished successfully
```

In [199]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479808
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479808
Jan 03 22:38:09 ..... started STAR run
Jan 03 22:38:09 ..... loading genome
Jan 03 22:38:22 ..... started mapping
Jan 03 22:42:27 ..... finished successfully
```

The next step was to convert these sam files to bam files using samtools.

In [200]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
[bam_sort_core] merging from 11 files...
```

In [201]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479802
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479802
[bam_sort_core] merging from 8 files...
```

In [202]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479803
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479803
[bam_sort_core] merging from 10 files...
```

In [203]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479804
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479804
[bam_sort_core] merging from 11 files...
```

In [204]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479805
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479805
[bam_sort_core] merging from 19 files...
```

In [205]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479806
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479806
[bam_sort_core] merging from 7 files...
```

In [206]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479807
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479807
[bam_sort_core] merging from 9 files...
```

In [207]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479808
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479808
[bam_sort_core] merging from 9 files...
```

Then we also generated an index file of each bam file in case we wanted to check our bam file in IGV.

In [208]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
```

In [209]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479802
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479802
```

In [210]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479803
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479803
```

In [211]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479804
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479804
```

In [212]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479805
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479805
```

In [213]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479806
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479806
```

In [214]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479807
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479807
```

In [215]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479808
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479808
```

We then looked at one of the bam files to check whether everything looked good.

In [216]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
SRR2479801.6451707      419     NC_000067.6     3069571 0       68M     =
3069571 68      TCTATCCAGGCTCTTCTGGCTTTCATAGTCTCTGGTGAAAAGTCTGGTGTAATTCTGATAGGCC
TGCC    <CC9CGFCFCFGGFCCF@FF88CCEGGGGGC,CCFCF,F9<FFEFCA@FGGGFFEFEDF<FFGGGGGF
NH:i:5  HI:i:4  AS:i:134        nM:i:0
SRR2479801.6451707      339     NC_000067.6     3069571 0       68M     =
3069571 -68     TCTATCCAGGCTCTTCTGGCTTTCATAGTCTCTGGTGAAAAGTCTGGTGTAATTCTGATAGGCC
TGCC    A@CFFDGGEFFD8FF<FFFDGFCFGF<CEGFFC;,@<GGEEF,EFC,FFGFGCGGGFCGCC@FCCB@9
NH:i:5  HI:i:4  AS:i:134        nM:i:0
SRR2479801.7305973      99      NC_000067.6     3093063 255     75M     =
3242712 149724  GGAGTTTGGCTGGGTATAGTAGCCTGGGCTGCAGTTTGTGTTCTCTTAGTGTCTGTATAACATC
TGTCCAGGCTC      CCCCCGGCFGGGGGGGCF<EFFGGGGGCFGGEGGFGGGGGGFGGGGGGGGGFGGGGGGGGGGGGG
GGGGGCC8FGG      NH:i:1  HI:i:1  AS:i:134        nM:i:6
samtools view: writing to standard output failed: Broken pipe
samtools view: error closing standard output: -1
```

We then used idxstats to see how many reads were mapped per chromosome.

From left to right: reference sequence name, sequence length, # mapped reads and # unmapped reads.

Al the NC files seen below represent a chromosome. The exact chromosome can be found by looking up the NC number on https://www.ncbi.nlm.nih.gov/genome/?term=mus+musculus+9 (https://www.ncbi.nlm.nih.gov/genome/?term=mus+musculus+9).

As you can see there are also a lot of NT numbers. These files are can be looked up in the nucleotide database of NCBI. For example NT_166280.1, the first NT number, is an unlocalized genomic contig. This means that it is a sequence found in an assembly that is associated with a specific chromosome but cannot be ordered or oriented on that chromosome (explanation was found on https://www.ncbi.nlm.nih.gov/grc/help/definitions (https://www.ncbi.nlm.nih.gov/grc/help/definitions)).

There are also some NW numbers. These are patches. Patches are accessioned scaffold sequences that represent assembly updates (explanation was found on https://www.ncbi.nlm.nih.gov/grc/help/patches (https://www.ncbi.nlm.nih.gov/grc/help/patches)).

As you can see the largest number of mapped reads corresponds to the chromosomes while only a small number of reads are mapped to the NT and NW numbers.

In [217]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/SRR2479801
NC_000067.6     195471971       1438010 0
NT_166280.1     169725  62      0
NT_166281.1     241735  38      0
NT_166282.1     153618  42      0
NT_166283.1     39340   20      0
NT_162750.1     206961  62      0
NC_000068.7     182113224       2029458 0
NC_000069.6     160039680       937147  0
NC_000070.6     156508116       1134691 0
NT_166291.1     66673   262     0
NT_187052.1     14945   12      0
NT_166434.1     227966  540     0
NT_187053.1     207968  764     0
NT_187054.1     191905  1174    0
NT_187055.1     1976    28      0
NC_000071.6     151834684       1191860 0
NT_187056.1     199368  78      0
NT_187057.1     205776  82      0
```

We then stored all the bam files in one folder because it is easier to work with during the upcoming steps.

Path to the bam files: /mnt/storage/r0305394/jupyternotebooks/r0305394/groepswerk/Rnaseq/Bam

The next step was to count all the mapped reads per gene. First we looked at the annotation file we used earlier when generating the reference genome so that we were able to see what word we had to use to filter the gene names with the according mapped reads.

In [218]:

```
##gff-version 3
#!gff-spec-version 1.21
#!processor NCBI annotwriter
#!genome-build GRCm38.p5
#!genome-build-accession NCBI_Assembly:GCF_000001635.25
##sequence-region NC_000067.6 1 195471971
##species https://www.ncbi.nlm.nih.gov/Taxonomy/Browser/wwwtax.cgi?id=10090 (htt
ps://www.ncbi.nlm.nih.gov/Taxonomy/Browser/wwwtax.cgi?id=10090)
NC_000067.6     RefSeq  region  1       195471971       .       +       .
ID=id0;Dbxref=taxon:10090;Name=1;chromosome=1;gbkey=Src;genome=chromosome;mol_ty
pe=genomic DNA;strain=C57BL/6J
NC_000067.6     BestRefSeq%2CGnomon     gene    3199733 3672278 .       -
.       ID=gene0;Dbxref=GeneID:497097,MGI:MGI:3528744;Name=Xkr4;description=X-li
nked Kx blood group related 4;gbkey=Gene;gene=Xkr4;gene_biotype=protein_coding;g
ene_synonym=AY534250,Gm210,mKIAA1889,XRG4
```

The command below compiles the number of mapped reads per gene of all the samples together in one file.

Q is the quality controle. We used the same filter as in class.

g shows us the mapped reads per gene,

a is the path to the annotation file,

o determines the output file followed by all the bam files that are the input of this command.

In [219]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/Bam
featureCounts -Q 10 -g gene -a /mnt/storage/r0305394/jupyternotebooks/groepswerk/Se
```

```
        ==========     _____ _     _ _____  _____  _______      _____
        =====         / ____| |   | |  _ \|  __ \|  ____|    /\   |  __ \
          =====      | (___ | |   | | |_) | |__) | |__      /  \  | |  | |
            ====      \___ \| |   | |  _ <|  _  /|  __|    / /\ \ | |  | |
              ====    ____) | |__| | |_) | | \ \| |____  / ____ \| |__| |
        ==========   |_____/ \____/|____/|_|  \_\______/_/    \_\_____/
        v1.5.0-p1

//========================== featureCounts setting ===========================\\
||                                                                            ||
||             Input files : 8 BAM files                                      ||
||                           P SRR2479801.bam                                 ||
||                           P SRR2479802.bam                                 ||
||                           P SRR2479803.bam                                 ||
||                           P SRR2479804.bam                                 ||
||                           P SRR2479805.bam                                 ||
||                           P SRR2479806.bam                                 ||
||                           P SRR2479807.bam                                 ||
||                           P SRR2479808.bam                                 ||
||                                                                            ||
||             Output file : all.counts                                       ||
||             Annotations : /mnt/storage/r0305394/jupyternotebooks/groep ... ||
||                                                                            ||
||                 Threads : 1                                                ||
||                   Level : meta-feature level                               ||
||              Paired-end : no                                               ||
||         Strand specific : no                                               ||
||      Multimapping reads : not counted                                      ||
|| Multi-overlapping reads : not counted                                      ||
||       Read orientations : fr                                               ||
||                                                                            ||
\\====================== http://subread.sourceforge.net/ (http://subread.sourcefo
rge.net/) ======================//

//================================= Running ==================================\\
||                                                                            ||
|| Load annotation file /mnt/storage/r0305394/jupyternotebooks/groepswerk ... ||
||    Features : 1225991                                                      ||
||    Meta-features : 41575                                                   ||
||    Chromosomes/contigs : 185                                               ||
||                                                                            ||
|| Process BAM file SRR2479801.bam...                                         ||
||    Paired-end reads are included.                                          ||
||    Assign reads to features...                                             ||
||    Total reads : 26017423                                                  ||
||    Successfully assigned reads : 12297493 (47.3%)                          ||
||    Running time : 0.50 minutes                                             ||
||                                                                            ||
|| Process BAM file SRR2479802.bam...                                         ||
||    Paired-end reads are included.                                          ||
||    Assign reads to features...                                             ||
||    Total reads : 19182638                                                  ||
||    Successfully assigned reads : 11259603 (58.7%)                          ||
||    Running time : 0.41 minutes                                             ||
||                                                                            ||
|| Process BAM file SRR2479803.bam...                                         ||
||    Paired-end reads are included.                                          ||
||    Assign reads to features...                                             ||
||    Total reads : 23767113                                                  ||
||    Successfully assigned reads : 13872095 (58.4%)                          ||
||    Running time : 0.46 minutes                                             ||
||                                                                            ||
|| Process BAM file SRR2479804.bam...                                         ||
||    Paired-end reads are included.                                          ||
||    Assign reads to features...                                             ||
```

We then took a look at the output file generated by featurecounts. As you can see it looks really chaotic. The following steps are to clean up this file.

In [220]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/Bam
```

```
# Program:featureCounts v1.5.0-p1; Command:"featureCounts" "-Q" "10" "-g" "gene"
"-a" "/mnt/storage/r0305394/jupyternotebooks/groepswerk/Sequence/GCF_000001635.2
5_GRCm38.p5_genomic.gff" "-o" "all.counts" "SRR2479801.bam" "SRR2479802.bam" "SR
R2479803.bam" "SRR2479804.bam" "SRR2479805.bam" "SRR2479806.bam" "SRR2479807.bam
" "SRR2479808.bam"
Geneid  Chr     Start   End     Strand  Length  SRR2479801.bam  SRR2479802.bam
SRR2479803.bam  SRR2479804.bam  SRR2479805.bam  SRR2479806.bam  SRR2479807.bam
SRR2479808.bam
Xkr4    NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6     3199733;
3213439;3322816;3421702;3670552 3207317;3216968;3323760;3421901;3672278 -;-;-;-;
-       13987   47      68      76      118     66      24      36      32
Gm39585 NC_000067.6;NC_000067.6 3361552;3368342 3366505;3377812 +;+     14425
49      4       4       6       36      0       2       0
LOC108167595    NC_000067.6     3671524 3672264 -       741     0       0
0       0       0       0       0       0
Gm38717 NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;
NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_00006
7.6;NC_000067.6;NC_000067.6     4115673;4119668;4120015;4142612;4147812;4148612;
4163855;4170205;4197534;4206660;4226611;4228443;4231053;4233184;4243133 4115948;
4119712;4120073;4142766;4147963;4148744;4163941;4170404;4197641;4206837;4226823;
4228619;4231144;4233728;4243365 -;-;-;-;-;-;-;-;-;-;-;-;-;-;-   2653    0
8       1       0       3       0       0       1
Rp1     NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6
4290846;4343507;4351910;4352202;4360200;4409170 4293012;4350091;4352081;4352837;
4360314;4409266 -;-;-;-;-;-     9772    4       4       3       9       0
2       8       2
Sox17   NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6 4490928;4493100;4493772;
4496291 4492668;4493490;4495942;4497386 -;-;-;- 5399    37      95      71
88      95      78      94      103
Gm7357  NC_000067.6;NC_000067.6 4522817;4524432 4523591;4526931 +;+     3275
4       12      10      14      2       4       14      12
LOC108167609    NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6 4583127;4583406;
4598113;4608720 4583243;4583478;4598958;4608781 -;-;-;- 1098    0       1
0       0       1       0       1       0
```

The first we did was to generate a file that gives us the exact location of each gene without looking at the total amount of mapped reads. To do this we only kept the first six rows and cut the rest.

In [221]:

In [222]:

```
# Program:featureCounts v1.5.0-p1; Command:"featureCounts" "-Q" "10" "-g" "gene"
"-a" "/mnt/storage/r0305394/jupyternotebooks/groepswerk/Sequence/GCF_000001635.2
5_GRCm38.p5_genomic.gff" "-o" "all.counts" "SRR2479801.bam" "SRR2479802.bam" "SR
R2479803.bam" "SRR2479804.bam" "SRR2479805.bam" "SRR2479806.bam" "SRR2479807.bam
" "SRR2479808.bam"
Geneid  Chr     Start   End     Strand  Length
Xkr4    NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6     3199733;
3213439;3322816;3421702;3670552 3207317;3216968;3323760;3421901;3672278 -;-;-;-;
-       13987
Gm39585 NC_000067.6;NC_000067.6 3361552;3368342 3366505;3377812 +;+     14425
LOC108167595    NC_000067.6     3671524 3672264 -       741
Gm38717 NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;
NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_00006
7.6;NC_000067.6;NC_000067.6     4115673;4119668;4120015;4142612;4147812;4148612;
4163855;4170205;4197534;4206660;4226611;4228443;4231053;4233184;4243133 4115948;
4119712;4120073;4142766;4147963;4148744;4163941;4170404;4197641;4206837;4226823;
4228619;4231144;4233728;4243365 -;-;-;-;-;-;-;-;-;-;-;-;-;-;-   2653
Rp1     NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6
4290846;4343507;4351910;4352202;4360200;4409170 4293012;4350091;4352081;4352837;
4360314;4409266 -;-;-;-;-;-     9772
Sox17   NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6 4490928;4493100;4493772;
4496291 4492668;4493490;4495942;4497386 -;-;-;- 5399
Gm7357  NC_000067.6;NC_000067.6 4522817;4524432 4523591;4526931 +;+     3275
LOC108167609    NC_000067.6;NC_000067.6;NC_000067.6;NC_000067.6 4583127;4583406;
4598113;4608720 4583243;4583478;4598958;4608781 -;-;-;- 1098
```

As you can see we now have file containing the gene name, the chromosome on which the gene is located, the start and end of the gene, which strand is located on and the length of the gene.

We also want a file containing the gene names with the corresponding number of mapped reads. To do this we only take the first row containing the gene name and all the rows starting from row 7 which contain the number of mapped reads per sample.

In [223]:

In [2]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/Rnaseq/Bam
```

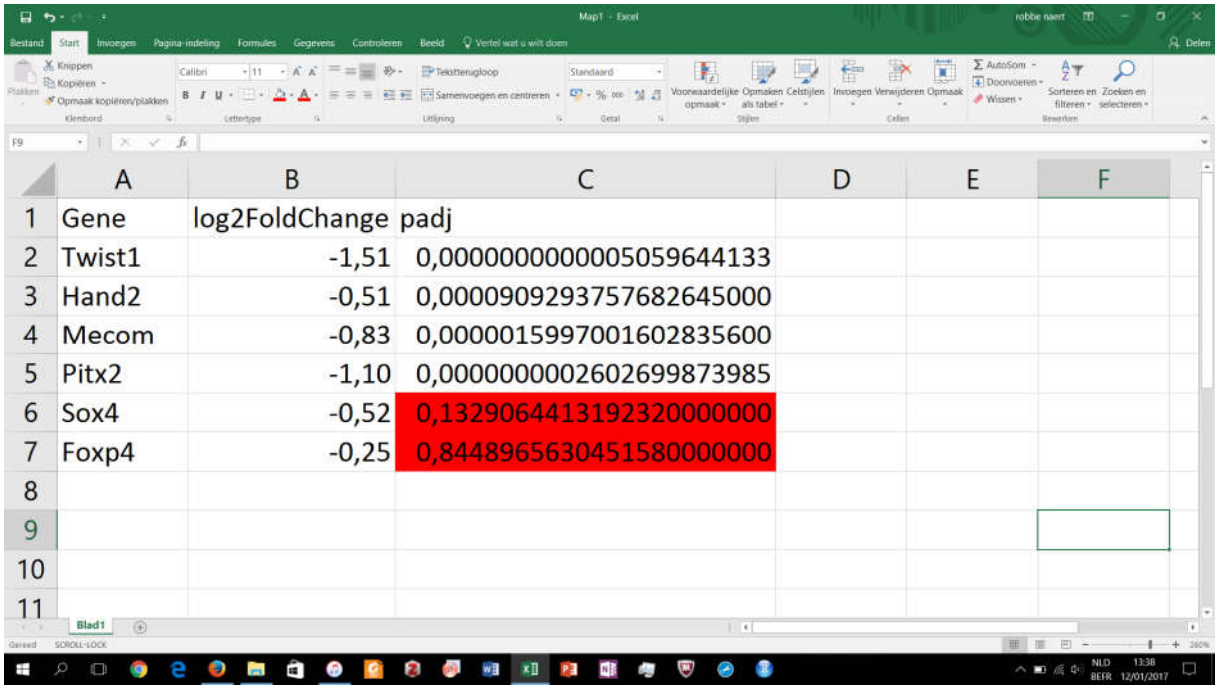
```
Geneid          SRR2479801.bam  SRR2479802.bam  SRR2479803.bam  SRR2479804.bam  SR
R2479805.bam    SRR2479806.bam  SRR2479807.bam  SRR2479808.bam
Xkr4            47              68              76              118             66
24              36              32
Gm39585         49              4               4               6               36
0               2               0
LOC108167595    0               0               0               0               0
0               0               0
Gm38717         0               8               1               0               3
0               0               1
Rp1             4               4               3               9               0
2               8               2
Sox17           37              95              71              88              95
78              94              103
Gm7357          4               12              10              14              2
4               14              12
LOC108167609    0               1               0               0               1
0               1               0
Gm39586         1               6               2               1               6
0               0               4
```

As you can see this file now shows the gene names with their corresponding number of mapped reads per sample.

DESeq2 analysis

The next step is to use this output to look at the differential expression using DESeq2. This will be done in R so this has to be done in a new notebook which is named Exam_project_SOX9_R.ipynb (path: /mnt/storage /r0305394/jupyternotebooks/groepswerk/Exam_project_SOX9_R.ipynb).

This generated a file with 666 differentially expressed genes. In the paper they mention the downregulation of several important transcription factors that are known to play a role in heart valve development such as Twist 1, Sox4, Hand2, Foxp4, Pitx2 and Mecom. We were able to confirm the downregulation of Twist1, Hand2, Mecom and Pitx2 in our results. Sox4 was also downregulated in our samples but it had a p-value of 0.13 so it was not significant. Foxp4 was also downregulated but had a p-value of 0.84 so it was not significant. It is possible that these results differ due a different alignment method (STAR vs. Tophat) and a different version of the mus musculus genome (mm10 vs. mm9).



	A	B	C	D	E	F
1	Gene	log2FoldChange	padj			
2	Twist1	-1,51	0,00000000000005059644133			
3	Hand2	-0,51	0,0000902937576826450000			
4	Mecom	-0,83	0,0000015997001602835600			
5	Pitx2	-1,10	0,0000000002602699873985			
6	Sox4	-0,52	0,1329064413192320000000			
7	Foxp4	-0,25	0,8448965630451580000000			
8						
9						
10						
11						

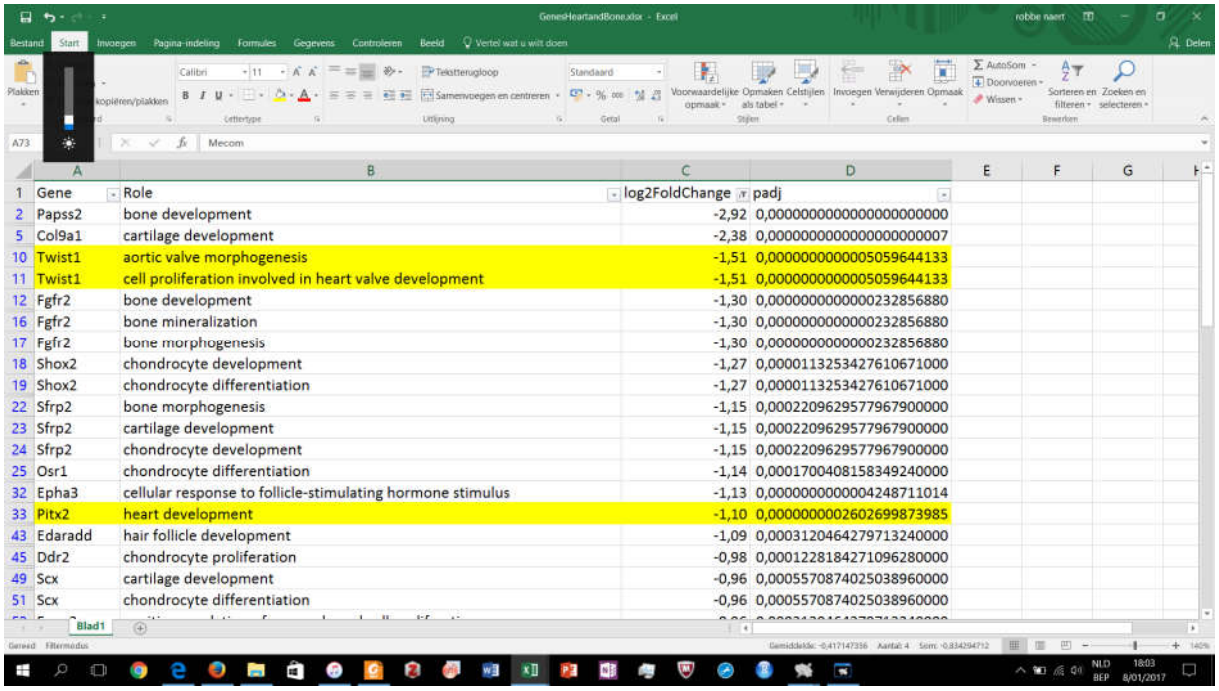
Next we determined the ontology of all the differently expressed by genes via humanmine.org. We extracted the results as a tsv file and used excel to study them.

The first thing we did was to check several processes in which sox9 is involved (heart valve development and limb formation).

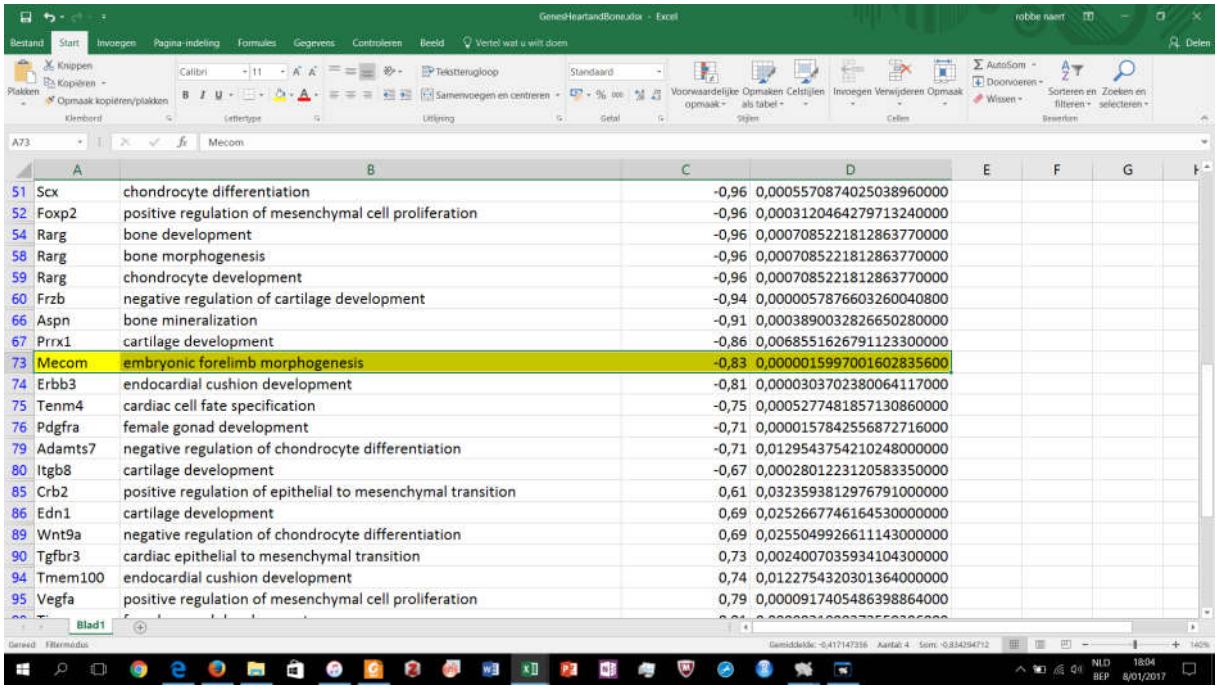
The processes we checked included epithelial-to-mesenchymal transition, cardiac cushion, valve development, chondrogenesis, limb formation, cartilage formation and mesenchyme development,.... These are all processes involved in heart valve formation and/or limb formation and/or bone development and have been linked to sox9. We further also found a paper in which dysfunctional sox9 has been linked to disorders linked to sex development (https://www.ncbi.nlm.nih.gov/pubmed/24115316 (https://www.ncbi.nlm.nih.gov/pubmed/24115316)).

We then copied the genes that played a role in these processes and made a new list in excel. We then selected only those genes that had a 2log fold change equal to/higher than 0.58 or equal to/lower than -0.58. In this way we were able to generate a list of 111 up - and downregulated genes that play a role in processes such as aortic valve morphogenesis (Twist1), chondrocyt development/differentation (Shox2), endocardial cushion formation (Errb2),female gonad development (Serpine1) and cartilage development (Sfrp2).

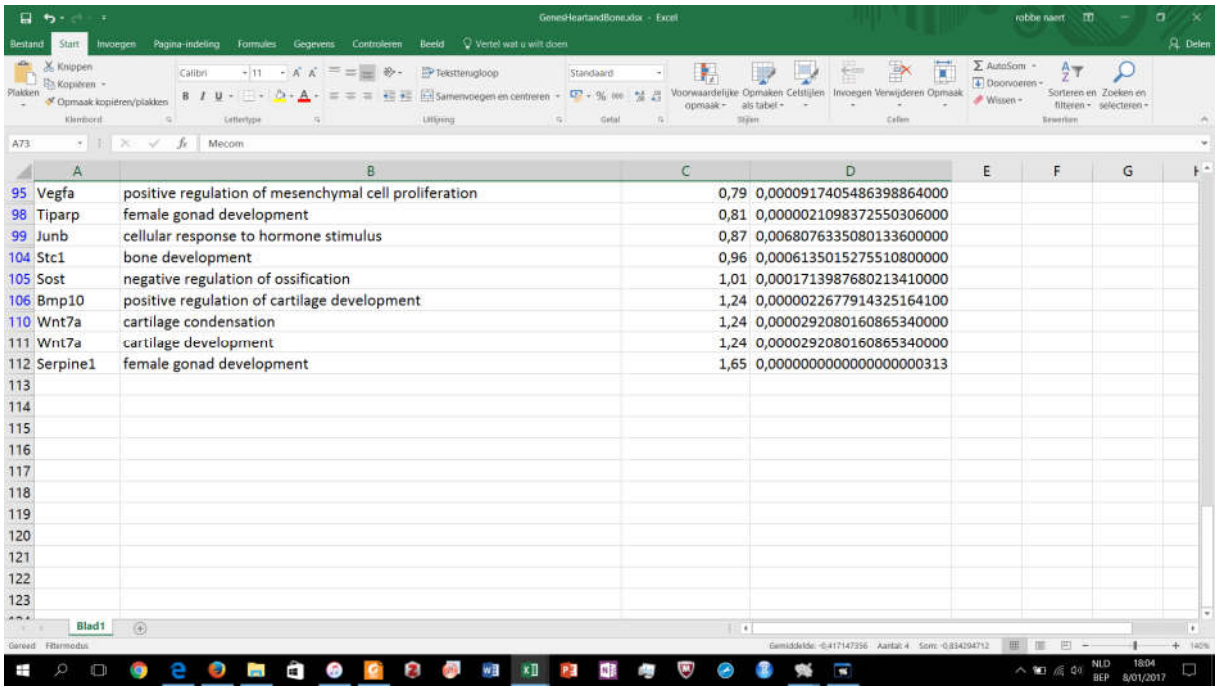
These results seem to indicate that sox9 KO affects processes linked to sex development, heart valve development and cartilage formation. As mentioned in the previous markdown box, we were able to confirm three important downregulated TFs that are mentioned in the paper (Twist1, Hand2, Pitx2 and Mecom). Because Hand2 had log2 fold changes higher that -0.58 we discarded it from the list shown above. According to our gene ontology Mecom has no role in heart development although in the paper it is mentioned as an important TF for heart valve development. In our gene ontology report Mecom was shown to play a role in embryonic forelimb morphogenesis. We therefore searched pubmed and found a paper in which Mecom is shown to regulate cardiac development(https://www.ncbi.nlm.nih.gov/pubmed/?term=Mice+Carrying+a+Hypomorphic+Evi1+Allele+Are+Embryonic+Viable+but+Exhibit+Severe+Congenital+He (https://www.ncbi.nlm.nih.gov/pubmed/?term=Mice+Carrying+a+Hypomorphic+Evi1+Allele+Are+Embryonic+Viable+but+Exhibit+Severe+Congenital+He



	A	B	C	D
1	Gene	Role	log2FoldChange	padj
2	Papss2	bone development	-2,92	0,000000000000000000000
5	Col9a1	cartilage development	-2,38	0,000000000000000000007
10	Twist1	aortic valve morphogenesis	-1,51	0,000000000000059644133
11	Twist1	cell proliferation involved in heart valve development	-1,51	0,000000000000059644133
12	Fgfr2	bone development	-1,30	0,000000000000232856880
16	Fgfr2	bone mineralization	-1,30	0,000000000000232856880
17	Fgfr2	bone morphogenesis	-1,30	0,000000000000232856880
18	Shox2	chondrocyte development	-1,27	0,000011325342761067100
19	Shox2	chondrocyte differentiation	-1,27	0,000011325342761067100
22	Sfrp2	bone morphogenesis	-1,15	0,000220962957796790000
23	Sfrp2	cartilage development	-1,15	0,000220962957796790000
24	Sfrp2	chondrocyte development	-1,15	0,000220962957796790000
25	Osr1	chondrocyte differentiation	-1,14	0,000170040815834924000
32	Epha3	cellular response to follicle-stimulating hormone stimulus	-1,13	0,000000000000424871101
33	Pitx2	heart development	-1,10	0,000000000260269987398
43	Edaradd	hair follicle development	-1,09	0,000312046427971324000
45	Ddr2	chondrocyte proliferation	-0,98	0,000122818427109628000
49	Scx	cartilage development	-0,96	0,000557087402503896000
51	Scx	chondrocyte differentiation	-0,96	0,000557087402503896000



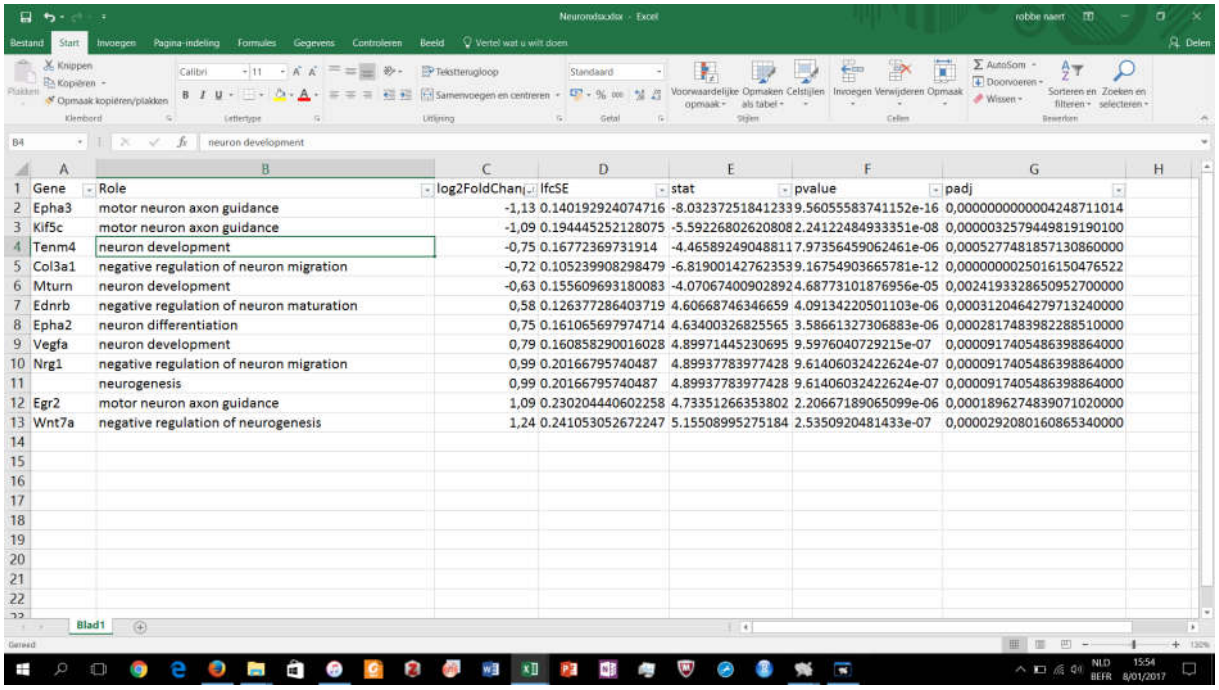
	A	B	C	D
51	Scx	chondrocyte differentiation	-0,96	0,0005570874025038960000
52	Foxp2	positive regulation of mesenchymal cell proliferation	-0,96	0,0003120464279713240000
54	Rarg	bone development	-0,96	0,0007085221812863770000
58	Rarg	bone morphogenesis	-0,96	0,0007085221812863770000
59	Rarg	chondrocyte development	-0,96	0,0007085221812863770000
60	Frzb	negative regulation of cartilage development	-0,94	0,0000057876603260040800
66	Aspn	bone mineralization	-0,91	0,0003890032826650280000
67	Prrx1	cartilage development	-0,86	0,0068551626791123300000
73	Mecom	embryonic forelimb morphogenesis	-0,83	0,0000015997001602835600
74	Erbb3	endocardial cushion development	-0,81	0,0000303702380064117000
75	Tenm4	cardiac cell fate specification	-0,75	0,0005277481857130860000
76	Pdgfra	female gonad development	-0,71	0,0000157842556872716000
79	Adamts7	negative regulation of chondrocyte differentiation	-0,71	0,0129543754210248000000
80	Itgb8	cartilage development	-0,67	0,0002801223120583350000
85	Crb2	positive regulation of epithelial to mesenchymal transition	0,61	0,0323593812976791000000
86	Edn1	cartilage development	0,69	0,0252667746164530000000
89	Wnt9a	negative regulation of chondrocyte differentiation	0,69	0,0255049926611143000000
90	Tgfbr3	cardiac epithelial to mesenchymal transition	0,73	0,0024007035934104300000
94	Tmem100	endocardial cushion development	0,74	0,0122754320301364000000
95	Vegfa	positive regulation of mesenchymal cell proliferation	0,79	0,0000917405486398864000



	A	B	C	D
95	Vegfa	positive regulation of mesenchymal cell proliferation	0,79	0,0000917405486398864000
98	Tiparp	female gonad development	0,81	0,0000021098372550306000
99	Junb	cellular response to hormone stimulus	0,87	0,0068076335080133600000
104	Stc1	bone development	0,96	0,0006135015275510800000
105	Sost	negative regulation of ossification	1,01	0,0001713987680213410000
106	Bmp10	positive regulation of cartilage development	1,24	0,0000022677914325164100
110	Wnt7a	cartilage condensation	1,24	0,0000292080160865340000
111	Wnt7a	cartilage development	1,24	0,0000292080160865340000
112	Serpine1	female gonad development	1,65	0,0000000000000000000313
113				
114				
115				
116				
117				
118				
119				
120				
121				
122				
123				

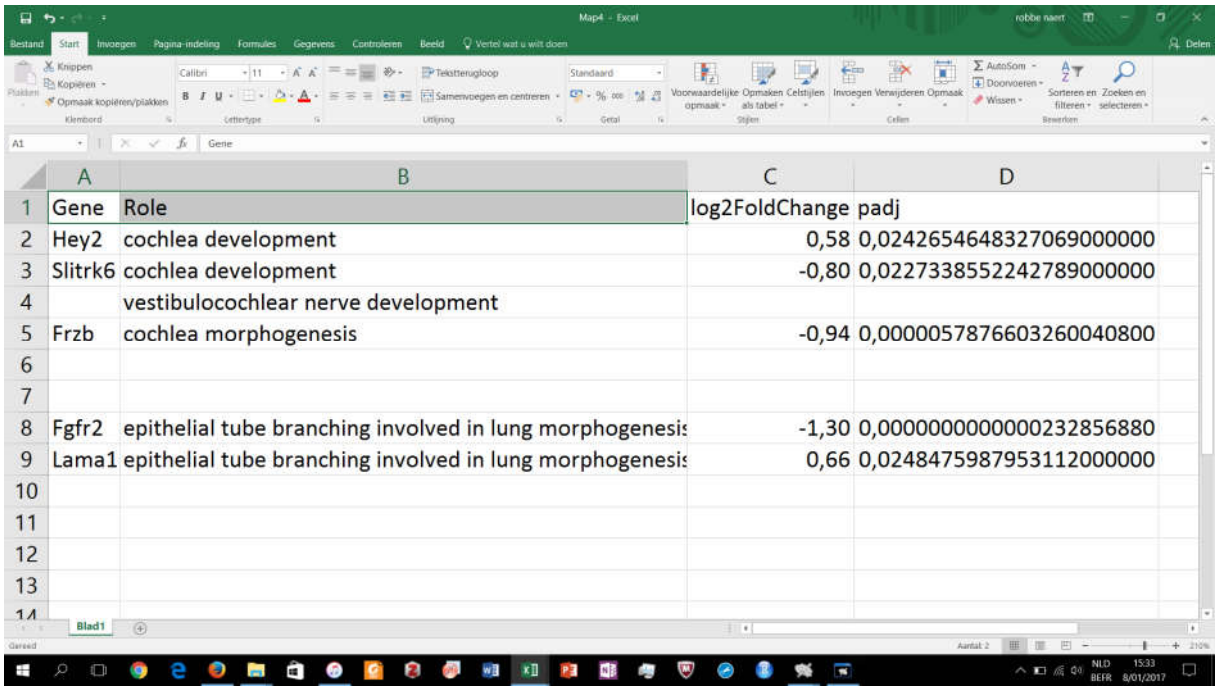
When we went through the original gene ontology list (containing the 666 up - and downregulated genes), we noticed that a lot of the up - and downreglated genes seemed to play a role in neuronal development and functioning. We searched pubmed to whether we could find any role for sox9 in neuronal development. We found an article that indicated that sox9 to be involved in neurogenesis (https://www.ncbi.nlm.nih.gov/pubmed/25888505 (https://www.ncbi.nlm.nih.gov/pubmed/25888505)).

A lot of genes were also involved in astrocyte development, formation and functioning. We searched pubmed and found a paper in which researchers state that sox9 plays a role in de start of gliogenesis, a process in which glial cells are formed. These glial cells have the potential to differentiate into astrocytes (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3543821/ (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3543821/)).



Gene	Role	log2FoldChan	lfcSE	stat	pvalue	padj
Epha3	motor neuron axon guidance	-1,13	0.140192924074716	-8.032372518412339	9.56055583741152e-16	0,000000000004248711014
Kif5c	motor neuron axon guidance	-1,09	0.194445252128075	-5.592268026208082	2.24122484933351e-08	0,000003257944981910 0
Tenm4	neuron development	-0,75	0.16772369731914	-4.465892490488117	7.97356459062461e-06	0,000527748185713 0860000
Col3a1	negative regulation of neuron migration	-0,72	0.105239908298479	-6.819001427623539	8.16754903665781e-12	0,000000002501615 0476522
Mturn	neuron development	-0,63	0.155609693180083	-4.070674009028924	4.68773101876956e-05	0,002419332865095 2700000
Ednrb	negative regulation of neuron maturation	0,58	0.126377286403719	4.60668746346659	4.09134220501103e-06	0,000312046279713 240000
Epha2	neuron differentiation	0,75	0.161065697974714	4.63400326825565	3.58661327306883e-06	0,000281748398228 8510000
Vegfa	neuron development	0,79	0.160858290016028	4.89971445230695	9.5976040729215e-07	0,000091740548639 8864000
Nrg1	negative regulation of neuron migration	0,99	0.20166795740487	4.89937783977428	9.61406032422624e-07	0,000091740548639 8864000
	neurogenesis	0,99	0.20166795740487	4.89937783977428	9.61406032422624e-07	0,000091740548639 8864000
Egr2	motor neuron axon guidance	1,09	0.230204440602258	4.73351266353802	2.20667189065099e-06	0,000189627483907 1020000
Wnt7a	negative regulation of neurogenesis	1,24	0.241053052672247	5.15508995275184	2.5350920481433e-07	0,000029208016086 5340000

In this paper, they refer to two papers that show that sox9 also is important during lung development in the lung epithelium during branching morphogenesis (https://www.ncbi.nlm.nih.gov/pubmed/?term=sox9+plays+multiple+roles+in+the+lung+epithelium (https://www.ncbi.nlm.nih.gov/pubmed/?term=sox9+plays+multiple+roles+in+the+lung+epithelium)) and cochlea development (https://www.ncbi.nlm.nih.gov/pubmed/?term=loss+of+sox9+in+the+periotic+mesenchyme (https://www.ncbi.nlm.nih.gov/pubmed/?term=loss+of+sox9+in+the+periotic+mesenchyme)). We checked the gene ontology list to see if any genes were affected that play a role in these processes.



Gene	Role	log2FoldChange	padj
Hey2	cochlea development	0,58	0,0242654648327069000000
Slitrk6	cochlea development	-0,80	0,0227338552242789000000
	vestibulocochlear nerve development		
Frzb	cochlea morphogenesis	-0,94	0,0000057876603260040800
Fgfr2	epithelial tube branching involved in lung morphogenesis	-1,30	0,0000000000000232856880
Lama1	epithelial tube branching involved in lung morphogenesis	0,66	0,0248475987953112000000

Overall, we can conclude from this RNA sequencing that KO of sox9 affects a number of genes that play a role in heart valve formation, cartilage development but also in neuronal differentation. We were able to confirm some of the results obtained in the paper. We were able to show a downregulation of Twist1, Pitx2 and Mecom which are all TF that are important for normal heart valve and heart development.

ChIP sequencing

We have two ChIP seq samples. The first sample is from the AVC and will tell us something about the genes that are regulated by sox9 in the heart. The second samples is from the limb mesenchyme and will tell which genes are bound and regulated by sox9 in the limb. This will allows us to see whether the downregulated genes found in the heart during RNA seq are due to a direct effect of sox9 which means that sox9 binds directly to the transcription site of this gene. Or whether this gene is downregulated through an indirect effect meaning that sox9 binds a gene X that controls the transcription of a gene Y. Secondly this data will allow to determine binding motifs in the heart and in the limb and will shows us if there are any similar binding motifs in the heart and limb.

Fist we made a new folder to store all data related to the ChIP sequencing. Then we made two folders inside this folder where we stored the original ChIP seq data that we downloaded from https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE73225 (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE73225).

In [146]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk
pwd
mkdir -p ChIPseq
cd ChIPseq
pwd
mkdir -p AVC
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq
```

In [337]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
pwd
```

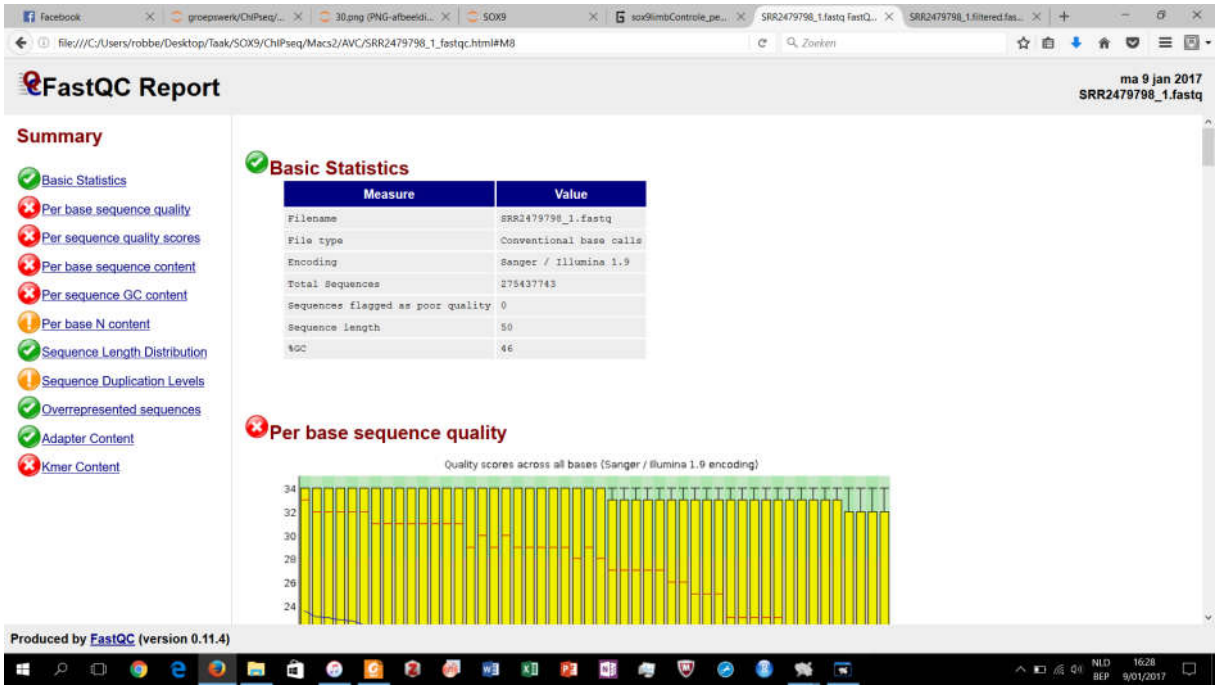
```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
Read 275437743 spots for SRR2479798
Written 275437743 spots for SRR2479798
```

We then checked the quality of the ChIP seq data (AVC sample).

In [314]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIP
```

```
Started analysis of SRR2479798_1.fastq
Approx 5% complete for SRR2479798_1.fastq
Approx 10% complete for SRR2479798_1.fastq
Approx 15% complete for SRR2479798_1.fastq
Approx 20% complete for SRR2479798_1.fastq
Approx 25% complete for SRR2479798_1.fastq
Approx 30% complete for SRR2479798_1.fastq
Approx 35% complete for SRR2479798_1.fastq
Approx 40% complete for SRR2479798_1.fastq
Approx 45% complete for SRR2479798_1.fastq
Approx 50% complete for SRR2479798_1.fastq
Approx 55% complete for SRR2479798_1.fastq
Approx 60% complete for SRR2479798_1.fastq
Approx 65% complete for SRR2479798_1.fastq
Approx 70% complete for SRR2479798_1.fastq
Approx 75% complete for SRR2479798_1.fastq
Approx 80% complete for SRR2479798_1.fastq
Approx 85% complete for SRR2479798_1.fastq
Approx 90% complete for SRR2479798_1.fastq
Approx 95% complete for SRR2479798_1.fastq
Analysis complete for SRR2479798_1.fastq
```

As you can see this quality report is bad. The per sequence quality is bad for every base position. Trimming this would mean to trim the entire reads. In the next steps we will work with this fastq file. Later on we will try to clean this fastq file and will repeat everything with this cleaned fastq file to see the effect.



The fastq file generated during the step above was over 50 GB large. We therefore deleted the file after we had used it to map in the following steps.

Next we downloaded the ChIP seq data for the second sample (limb sample).

In [319]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
pwd
```

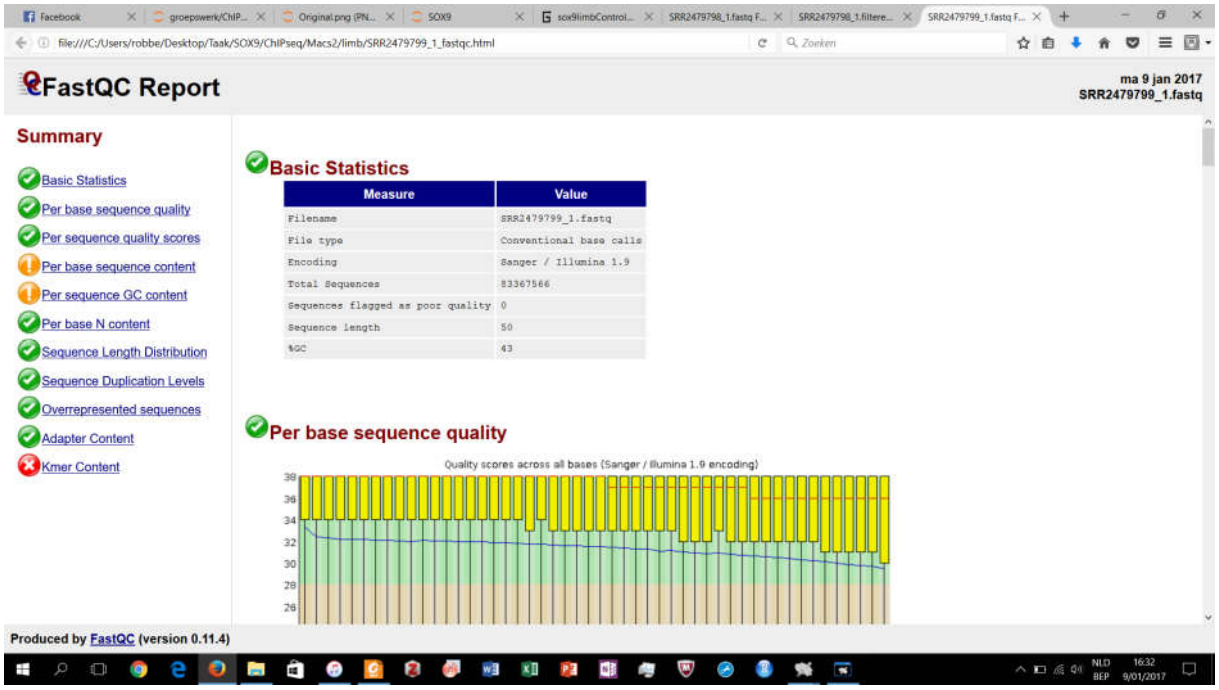
```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
Read 83367566 spots for SRR2479799
Written 83367566 spots for SRR2479799
```

Quality control was performed on the second ChIP seq data set.

In [321]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIP
```

```
Started analysis of SRR2479799_1.fastq
Approx 5% complete for SRR2479799_1.fastq
Approx 10% complete for SRR2479799_1.fastq
Approx 15% complete for SRR2479799_1.fastq
Approx 20% complete for SRR2479799_1.fastq
Approx 25% complete for SRR2479799_1.fastq
Approx 30% complete for SRR2479799_1.fastq
Approx 35% complete for SRR2479799_1.fastq
Approx 40% complete for SRR2479799_1.fastq
Approx 45% complete for SRR2479799_1.fastq
Approx 50% complete for SRR2479799_1.fastq
Approx 55% complete for SRR2479799_1.fastq
Approx 60% complete for SRR2479799_1.fastq
Approx 65% complete for SRR2479799_1.fastq
Approx 70% complete for SRR2479799_1.fastq
Approx 75% complete for SRR2479799_1.fastq
Approx 80% complete for SRR2479799_1.fastq
Approx 85% complete for SRR2479799_1.fastq
Approx 90% complete for SRR2479799_1.fastq
Approx 95% complete for SRR2479799_1.fastq
Analysis complete for SRR2479799_1.fastq
```

The per base sequence, dupplication levels and overrepresented sequences are looking good.



The fastq file generated during the step above was over 15 GB large. We therefore deleted the file after we had used it to map in the following steps.

The next step is to align the reads to the reference genome. In this case we will use STAR to align to the reference genome we made earlier in RNA sequencing. The first file that we are mapping contains Chip seq data from the AVC from mice hearts.

In [266]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
Jan 06 13:49:46 ..... started STAR run
Jan 06 13:49:50 ..... loading genome
Jan 06 13:52:58 ..... started mapping
Jan 06 15:06:23 ..... finished successfully
```

The sam file generated in the step above was over 30 GB large. We therefore deleted the sam file after converting it into a bam file.

Next we converted the sam file into a bam file using samtools.

In [267]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
[bam_sort_core] merging from 40 files...
```

We then repeated the same steps for the second Chip Sequencing fasta file. This file contains the ChIP seq data from limb mesenchyme.

In [269]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
Jan 06 15:44:42 ..... started STAR run
Jan 06 15:44:42 ..... loading genome
Jan 06 15:47:38 ..... started mapping
Jan 06 16:04:03 ..... finished successfully
```

The sam file generated in the step above was over 15 GB large. We therefore deleted the sam file after converting it into a bam file.

In [270]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
[bam_sort_core] merging from 21 files...
```

Next we generated two index files, one for each bam file generated in the previous steps.

In [271]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
```

In [272]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
```

Peak calling with macs2.

We performed peak calling two times on each bam file (once without control and once without controle) to see the effect of the control files wich normally should filter out false positives.

In [236]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
```

In [273]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
pwd
macs2 callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/SRR
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
INFO  @ Fri, 06 Jan 2017 16:21:42:
# Command line: callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/Ch
IPseq/AVC/SRR2479798.bam --outdir /mnt/storage/r0305394/jupyternotebooks/groepsw
erk/ChIPseq/AVC/macs2 -n sox9AVC -g mm -q 0.01
# ARGUMENTS LIST:
# name = sox9AVC
# format = AUTO
# ChIP-seq file = ['/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AV
C/SRR2479798.bam']
# control file = None
# effective genome size = 1.87e+09
# band width = 300
# model fold = [5, 50]
# qvalue cutoff = 1.00e-02
# Larger dataset will be scaled towards smaller dataset.
# Range for calculating regional lambda is: 10000 bps
# Broad region calling is off

INFO  @ Fri, 06 Jan 2017 16:21:42: #1 read tag files...
INFO  @ Fri, 06 Jan 2017 16:21:42: #1 read treatment tags...
INFO  @ Fri, 06 Jan 2017 16:21:42: Detected format is: BAM
INFO  @ Fri, 06 Jan 2017 16:21:42: * Input file is gzipped.
INFO  @ Fri, 06 Jan 2017 16:21:46:  1000000
INFO  @ Fri, 06 Jan 2017 16:21:50:  2000000
INFO  @ Fri, 06 Jan 2017 16:21:53:  3000000
INFO  @ Fri, 06 Jan 2017 16:21:57:  4000000
INFO  @ Fri, 06 Jan 2017 16:22:01:  5000000
INFO  @ Fri, 06 Jan 2017 16:22:04:  6000000
INFO  @ Fri, 06 Jan 2017 16:22:08:  7000000
INFO  @ Fri, 06 Jan 2017 16:22:12:  8000000
INFO  @ Fri, 06 Jan 2017 16:22:16:  9000000
INFO  @ Fri, 06 Jan 2017 16:22:19:  10000000
INFO  @ Fri, 06 Jan 2017 16:22:23:  11000000
INFO  @ Fri, 06 Jan 2017 16:22:27:  12000000
INFO  @ Fri, 06 Jan 2017 16:22:31:  13000000
INFO  @ Fri, 06 Jan 2017 16:22:34:  14000000
INFO  @ Fri, 06 Jan 2017 16:22:38:  15000000
INFO  @ Fri, 06 Jan 2017 16:22:41:  16000000
INFO  @ Fri, 06 Jan 2017 16:22:44:  17000000
INFO  @ Fri, 06 Jan 2017 16:22:47:  18000000
INFO  @ Fri, 06 Jan 2017 16:22:51:  19000000
INFO  @ Fri, 06 Jan 2017 16:22:55:  20000000
INFO  @ Fri, 06 Jan 2017 16:22:58:  21000000
INFO  @ Fri, 06 Jan 2017 16:23:02:  22000000
INFO  @ Fri, 06 Jan 2017 16:23:06:  23000000
INFO  @ Fri, 06 Jan 2017 16:23:10:  24000000
INFO  @ Fri, 06 Jan 2017 16:23:13:  25000000
INFO  @ Fri, 06 Jan 2017 16:23:17:  26000000
INFO  @ Fri, 06 Jan 2017 16:23:21:  27000000
INFO  @ Fri, 06 Jan 2017 16:23:25:  28000000
INFO  @ Fri, 06 Jan 2017 16:23:28:  29000000
INFO  @ Fri, 06 Jan 2017 16:23:32:  30000000
INFO  @ Fri, 06 Jan 2017 16:23:36:  31000000
INFO  @ Fri, 06 Jan 2017 16:23:40:  32000000
INFO  @ Fri, 06 Jan 2017 16:23:44:  33000000
INFO  @ Fri, 06 Jan 2017 16:23:47:  34000000
INFO  @ Fri, 06 Jan 2017 16:23:51:  35000000
INFO  @ Fri, 06 Jan 2017 16:23:55:  36000000
INFO  @ Fri, 06 Jan 2017 16:23:59:  37000000
INFO  @ Fri, 06 Jan 2017 16:24:02:  38000000
INFO  @ Fri, 06 Jan 2017 16:24:05:  39000000
INFO  @ Fri, 06 Jan 2017 16:24:09:  40000000
INFO  @ Fri, 06 Jan 2017 16:24:13:  41000000
INFO  @ Fri, 06 Jan 2017 16:24:17:  42000000
```

We then checked the amount of peaks that have been called in the AVC sample.

In [287]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
```

```
52848 sox9AVC_peaks.narrowPeak
```

We now want to perform another peak calling on the same sample but now we use a control sample to filter out false positives. This control sample contains DNA from the whole heart and has been crosslinked and sonicated but has not been immunoprecipitated. The first step is to download and convert the control files with fastq dump.

In [250]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq
```

In [251]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/controle
```

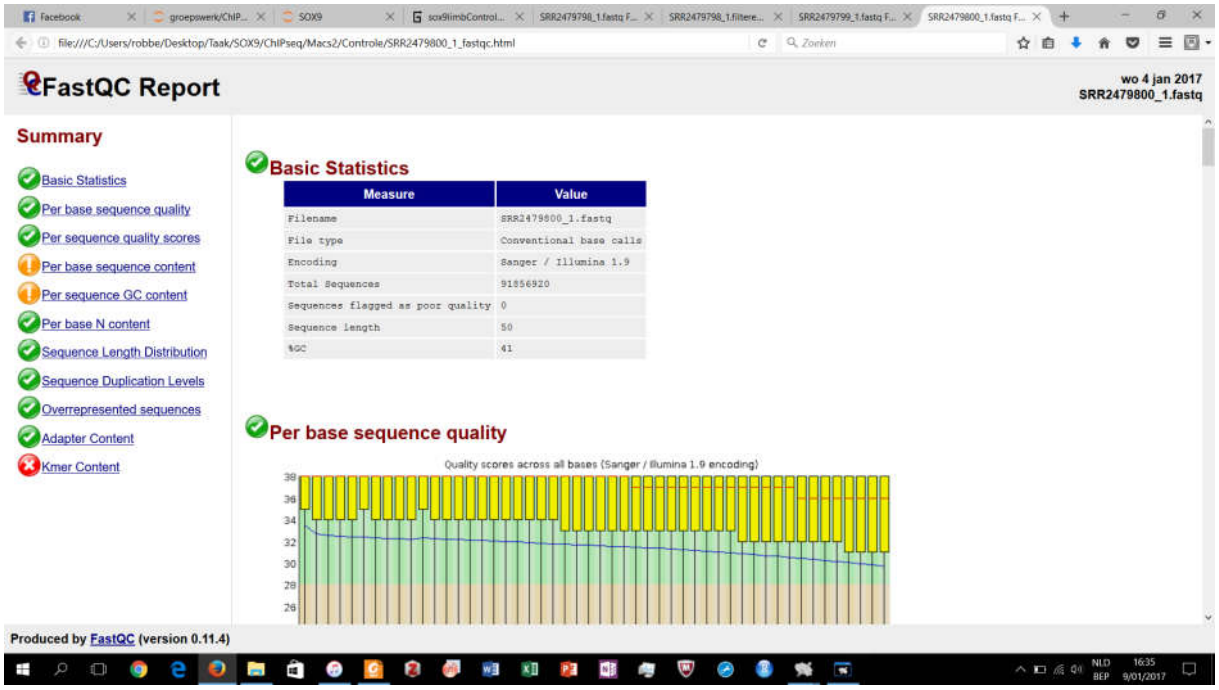
```
Read 91856920 spots for SRR2479800
Written 91856920 spots for SRR2479800
```

We also checked the quality of the control sample.

In [252]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/controle
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIP
```

```
Started analysis of SRR2479800_1.fastq
Approx 5% complete for SRR2479800_1.fastq
Approx 10% complete for SRR2479800_1.fastq
Approx 15% complete for SRR2479800_1.fastq
Approx 20% complete for SRR2479800_1.fastq
Approx 25% complete for SRR2479800_1.fastq
Approx 30% complete for SRR2479800_1.fastq
Approx 35% complete for SRR2479800_1.fastq
Approx 40% complete for SRR2479800_1.fastq
Approx 45% complete for SRR2479800_1.fastq
Approx 50% complete for SRR2479800_1.fastq
Approx 55% complete for SRR2479800_1.fastq
Approx 60% complete for SRR2479800_1.fastq
Approx 65% complete for SRR2479800_1.fastq
Approx 70% complete for SRR2479800_1.fastq
Approx 75% complete for SRR2479800_1.fastq
Approx 80% complete for SRR2479800_1.fastq
Approx 85% complete for SRR2479800_1.fastq
Approx 90% complete for SRR2479800_1.fastq
Approx 95% complete for SRR2479800_1.fastq
Analysis complete for SRR2479800_1.fastq
```

The per base sequence quality, duplication levels and overrepresented sequences look good.



Next the control sample is mapped to the reference genome using STAR.

In [275]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/controle
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
Jan 06 16:48:45 ..... started STAR run
Jan 06 16:48:49 ..... loading genome
Jan 06 16:49:02 ..... started mapping
Jan 06 17:05:29 ..... finished successfully
```

We then converted the sam file to a bam file using samtools.

In [276]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/controle
```

```
[bam_sort_core] merging from 26 files...
```

Next we performd a second peak calling on the AVC bam file to see the effect of the control file on the peaks called.

In [277]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
pwd
macs2 callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/SRR
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
INFO  @ Fri, 06 Jan 2017 17:26:30:
# Command line: callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/Ch
IPseq/AVC/SRR2479798.bam -c /mnt/storage/r0305394/jupyternotebooks/groepswerk/Ch
IPseq/controle/SRR2479800.bam --outdir /mnt/storage/r0305394/jupyternotebooks/gr
oepswerk/ChIPseq/AVC/macs2 -n sox9AVCControle -g mm -q 0.01
# ARGUMENTS LIST:
# name = sox9AVCControle
# format = AUTO
# ChIP-seq file = ['/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AV
C/SRR2479798.bam']
# control file = ['/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/con
trole/SRR2479800.bam']
# effective genome size = 1.87e+09
# band width = 300
# model fold = [5, 50]
# qvalue cutoff = 1.00e-02
# Larger dataset will be scaled towards smaller dataset.
# Range for calculating regional lambda is: 1000 bps and 10000 bps
# Broad region calling is off

INFO  @ Fri, 06 Jan 2017 17:26:30: #1 read tag files...
INFO  @ Fri, 06 Jan 2017 17:26:30: #1 read treatment tags...
INFO  @ Fri, 06 Jan 2017 17:26:30: Detected format is: BAM
INFO  @ Fri, 06 Jan 2017 17:26:30: * Input file is gzipped.
INFO  @ Fri, 06 Jan 2017 17:26:34:  1000000
INFO  @ Fri, 06 Jan 2017 17:26:37:  2000000
INFO  @ Fri, 06 Jan 2017 17:26:40:  3000000
INFO  @ Fri, 06 Jan 2017 17:26:43:  4000000
INFO  @ Fri, 06 Jan 2017 17:26:46:  5000000
INFO  @ Fri, 06 Jan 2017 17:26:49:  6000000
INFO  @ Fri, 06 Jan 2017 17:26:52:  7000000
INFO  @ Fri, 06 Jan 2017 17:26:55:  8000000
INFO  @ Fri, 06 Jan 2017 17:26:58:  9000000
INFO  @ Fri, 06 Jan 2017 17:27:00:  10000000
INFO  @ Fri, 06 Jan 2017 17:27:03:  11000000
INFO  @ Fri, 06 Jan 2017 17:27:06:  12000000
INFO  @ Fri, 06 Jan 2017 17:27:09:  13000000
INFO  @ Fri, 06 Jan 2017 17:27:12:  14000000
INFO  @ Fri, 06 Jan 2017 17:27:15:  15000000
INFO  @ Fri, 06 Jan 2017 17:27:18:  16000000
INFO  @ Fri, 06 Jan 2017 17:27:20:  17000000
INFO  @ Fri, 06 Jan 2017 17:27:23:  18000000
INFO  @ Fri, 06 Jan 2017 17:27:25:  19000000
INFO  @ Fri, 06 Jan 2017 17:27:28:  20000000
INFO  @ Fri, 06 Jan 2017 17:27:31:  21000000
INFO  @ Fri, 06 Jan 2017 17:27:34:  22000000
INFO  @ Fri, 06 Jan 2017 17:27:37:  23000000
INFO  @ Fri, 06 Jan 2017 17:27:40:  24000000
INFO  @ Fri, 06 Jan 2017 17:27:43:  25000000
INFO  @ Fri, 06 Jan 2017 17:27:46:  26000000
INFO  @ Fri, 06 Jan 2017 17:27:49:  27000000
INFO  @ Fri, 06 Jan 2017 17:27:52:  28000000
INFO  @ Fri, 06 Jan 2017 17:27:55:  29000000
INFO  @ Fri, 06 Jan 2017 17:27:58:  30000000
INFO  @ Fri, 06 Jan 2017 17:28:01:  31000000
INFO  @ Fri, 06 Jan 2017 17:28:04:  32000000
INFO  @ Fri, 06 Jan 2017 17:28:07:  33000000
INFO  @ Fri, 06 Jan 2017 17:28:10:  34000000
INFO  @ Fri, 06 Jan 2017 17:28:13:  35000000
INFO  @ Fri, 06 Jan 2017 17:28:16:  36000000
INFO  @ Fri, 06 Jan 2017 17:28:19:  37000000
INFO  @ Fri, 06 Jan 2017 17:28:22:  38000000
INFO  @ Fri, 06 Jan 2017 17:28:24:  39000000
INFO  @ Fri, 06 Jan 2017 17:28:27:  40000000
```

We then again checked the amount of peaks called. As you can see, the amount of peaks called has decreased from 52848 to 15486 because the control file has filtered out false positives.

In [289]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
```

```
15486 sox9AVCControle_peaks.narrowPeak
```

In the following commands we do the same for the limb samples as we have done for the AVC sample. We will call peaks two times: once without control samples and once with control samples.

In [239]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb
```

In [278]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
pwd
macs2 callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/SR
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
INFO  @ Fri, 06 Jan 2017 17:46:17:
# Command line: callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/Ch
IPseq/limb/SRR2479799.bam --outdir /mnt/storage/r0305394/jupyternotebooks/groeps
werk/ChIPseq/limb/macs2 -n sox9limb -g mm -q 0.01
# ARGUMENTS LIST:
# name = sox9limb
# format = AUTO
# ChIP-seq file = ['/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/li
mb/SRR2479799.bam']
# control file = None
# effective genome size = 1.87e+09
# band width = 300
# model fold = [5, 50]
# qvalue cutoff = 1.00e-02
# Larger dataset will be scaled towards smaller dataset.
# Range for calculating regional lambda is: 10000 bps
# Broad region calling is off

INFO  @ Fri, 06 Jan 2017 17:46:17: #1 read tag files...
INFO  @ Fri, 06 Jan 2017 17:46:17: #1 read treatment tags...
INFO  @ Fri, 06 Jan 2017 17:46:17: Detected format is: BAM
INFO  @ Fri, 06 Jan 2017 17:46:17: * Input file is gzipped.
INFO  @ Fri, 06 Jan 2017 17:46:21:  1000000
INFO  @ Fri, 06 Jan 2017 17:46:25:  2000000
INFO  @ Fri, 06 Jan 2017 17:46:29:  3000000
INFO  @ Fri, 06 Jan 2017 17:46:33:  4000000
INFO  @ Fri, 06 Jan 2017 17:46:37:  5000000
INFO  @ Fri, 06 Jan 2017 17:46:41:  6000000
INFO  @ Fri, 06 Jan 2017 17:46:45:  7000000
INFO  @ Fri, 06 Jan 2017 17:46:49:  8000000
INFO  @ Fri, 06 Jan 2017 17:46:52:  9000000
INFO  @ Fri, 06 Jan 2017 17:46:55:  10000000
INFO  @ Fri, 06 Jan 2017 17:46:59:  11000000
INFO  @ Fri, 06 Jan 2017 17:47:03:  12000000
INFO  @ Fri, 06 Jan 2017 17:47:07:  13000000
INFO  @ Fri, 06 Jan 2017 17:47:11:  14000000
INFO  @ Fri, 06 Jan 2017 17:47:15:  15000000
INFO  @ Fri, 06 Jan 2017 17:47:19:  16000000
INFO  @ Fri, 06 Jan 2017 17:47:24:  17000000
INFO  @ Fri, 06 Jan 2017 17:47:28:  18000000
INFO  @ Fri, 06 Jan 2017 17:47:32:  19000000
INFO  @ Fri, 06 Jan 2017 17:47:36:  20000000
INFO  @ Fri, 06 Jan 2017 17:47:39:  21000000
INFO  @ Fri, 06 Jan 2017 17:47:43:  22000000
INFO  @ Fri, 06 Jan 2017 17:47:47:  23000000
INFO  @ Fri, 06 Jan 2017 17:47:51:  24000000
INFO  @ Fri, 06 Jan 2017 17:47:55:  25000000
INFO  @ Fri, 06 Jan 2017 17:47:59:  26000000
INFO  @ Fri, 06 Jan 2017 17:48:03:  27000000
INFO  @ Fri, 06 Jan 2017 17:48:07:  28000000
INFO  @ Fri, 06 Jan 2017 17:48:11:  29000000
INFO  @ Fri, 06 Jan 2017 17:48:15:  30000000
INFO  @ Fri, 06 Jan 2017 17:48:19:  31000000
INFO  @ Fri, 06 Jan 2017 17:48:23:  32000000
INFO  @ Fri, 06 Jan 2017 17:48:27:  33000000
INFO  @ Fri, 06 Jan 2017 17:48:31:  34000000
INFO  @ Fri, 06 Jan 2017 17:48:35:  35000000
INFO  @ Fri, 06 Jan 2017 17:48:39:  36000000
INFO  @ Fri, 06 Jan 2017 17:48:41:  37000000
INFO  @ Fri, 06 Jan 2017 17:48:44:  38000000
INFO  @ Fri, 06 Jan 2017 17:48:48:  39000000
INFO  @ Fri, 06 Jan 2017 17:48:52:  40000000
INFO  @ Fri, 06 Jan 2017 17:48:56:  41000000
INFO  @ Fri, 06 Jan 2017 17:49:00:  42000000
```

We then determined the amount of peaks called by macs2.

In [288]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
```

```
7948 sox9limb_peaks.narrowPeak
```

We then did a second peak calling on the limb sample but now with control samples.

In [279]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
pwd
macs2 callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/SR
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
INFO  @ Fri, 06 Jan 2017 17:54:57:
# Command line: callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/Ch
IPseq/limb/SRR2479799.bam -c /mnt/storage/r0305394/jupyternotebooks/groepswerk/C
hIPseq/controle/SRR2479800.bam --outdir /mnt/storage/r0305394/jupyternotebooks/g
roepswerk/ChIPseq/limb/macs2 -n sox9limbControle -g mm -q 0.01
# ARGUMENTS LIST:
# name = sox9limbControle
# format = AUTO
# ChIP-seq file = ['/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/li
mb/SRR2479799.bam']
# control file = ['/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/con
trole/SRR2479800.bam']
# effective genome size = 1.87e+09
# band width = 300
# model fold = [5, 50]
# qvalue cutoff = 1.00e-02
# Larger dataset will be scaled towards smaller dataset.
# Range for calculating regional lambda is: 1000 bps and 10000 bps
# Broad region calling is off

INFO  @ Fri, 06 Jan 2017 17:54:57: #1 read tag files...
INFO  @ Fri, 06 Jan 2017 17:54:57: #1 read treatment tags...
INFO  @ Fri, 06 Jan 2017 17:54:57: Detected format is: BAM
INFO  @ Fri, 06 Jan 2017 17:54:57: * Input file is gzipped.
INFO  @ Fri, 06 Jan 2017 17:55:01:  1000000
INFO  @ Fri, 06 Jan 2017 17:55:05:  2000000
INFO  @ Fri, 06 Jan 2017 17:55:09:  3000000
INFO  @ Fri, 06 Jan 2017 17:55:13:  4000000
INFO  @ Fri, 06 Jan 2017 17:55:17:  5000000
INFO  @ Fri, 06 Jan 2017 17:55:21:  6000000
INFO  @ Fri, 06 Jan 2017 17:55:26:  7000000
INFO  @ Fri, 06 Jan 2017 17:55:29:  8000000
INFO  @ Fri, 06 Jan 2017 17:55:32:  9000000
INFO  @ Fri, 06 Jan 2017 17:55:36:  10000000
INFO  @ Fri, 06 Jan 2017 17:55:40:  11000000
INFO  @ Fri, 06 Jan 2017 17:55:44:  12000000
INFO  @ Fri, 06 Jan 2017 17:55:48:  13000000
INFO  @ Fri, 06 Jan 2017 17:55:52:  14000000
INFO  @ Fri, 06 Jan 2017 17:55:56:  15000000
INFO  @ Fri, 06 Jan 2017 17:56:00:  16000000
INFO  @ Fri, 06 Jan 2017 17:56:05:  17000000
INFO  @ Fri, 06 Jan 2017 17:56:09:  18000000
INFO  @ Fri, 06 Jan 2017 17:56:13:  19000000
INFO  @ Fri, 06 Jan 2017 17:56:17:  20000000
INFO  @ Fri, 06 Jan 2017 17:56:21:  21000000
INFO  @ Fri, 06 Jan 2017 17:56:25:  22000000
INFO  @ Fri, 06 Jan 2017 17:56:29:  23000000
INFO  @ Fri, 06 Jan 2017 17:56:33:  24000000
INFO  @ Fri, 06 Jan 2017 17:56:37:  25000000
INFO  @ Fri, 06 Jan 2017 17:56:41:  26000000
INFO  @ Fri, 06 Jan 2017 17:56:45:  27000000
INFO  @ Fri, 06 Jan 2017 17:56:49:  28000000
INFO  @ Fri, 06 Jan 2017 17:56:53:  29000000
INFO  @ Fri, 06 Jan 2017 17:56:57:  30000000
INFO  @ Fri, 06 Jan 2017 17:57:01:  31000000
INFO  @ Fri, 06 Jan 2017 17:57:05:  32000000
INFO  @ Fri, 06 Jan 2017 17:57:09:  33000000
INFO  @ Fri, 06 Jan 2017 17:57:13:  34000000
INFO  @ Fri, 06 Jan 2017 17:57:17:  35000000
INFO  @ Fri, 06 Jan 2017 17:57:21:  36000000
INFO  @ Fri, 06 Jan 2017 17:57:24:  37000000
INFO  @ Fri, 06 Jan 2017 17:57:27:  38000000
INFO  @ Fri, 06 Jan 2017 17:57:31:  39000000
INFO  @ Fri, 06 Jan 2017 17:57:35:  40000000
```

Next the amount of peaks is determined again. The amount of called peaks has decreased from 7948 to 4877 by using the control files to filter out false positives.

In [290]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
```

```
4877 sox9limbControle_peaks.narrowPeak
```

In [296]:
```
pwd
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
NT_039247.2     78931   79068   sox9limb_peak_1 58      .       5.24696 9.51248
5.80235 64
NT_039257.4     135028  136204  sox9limb_peak_2 622     .       6.12638 67.69081
        62.23330        565
NT_039257.4     136278  136496  sox9limb_peak_3 377     .       4.59856 42.64519
        37.76172        172
NT_039257.4     137679  137895  sox9limb_peak_4 382     .       4.51395 43.10199
        38.20818        58
NT_039257.4     138803  139177  sox9limb_peak_5 530     .       5.19123 58.31994
        53.05914        102
NT_039257.4     140814  141012  sox9limb_peak_6 460     .       5.24723 51.15353
        46.04527        71
NT_039257.4     142563  142700  sox9limb_peak_7 338     .       4.48069 38.66857
        33.89186        63
NT_039257.4     144792  144900  sox9limb_peak_8 467     .       6.11834 51.91118
        46.78747        54
NT_039257.4     145044  145517  sox9limb_peak_9 742     .       7.86744 79.94833
        74.28191        93
NT_039257.4     146669  146770  sox9limb_peak_10        377     .       5.85187
42.59619        37.71386        60
```

We then converted the peaks.narrowPeak files into bed files so they could be loaded into IGV. We first used head to determine which columns we needed to conserve into the bed file. We need to conserve the first four columns. The first column is the name of the chromosome, the second column is the start of the peak, third column is the end position of the peak and the fourth column is the name of the peak.

In [281]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
```

```
NT_039195.1     1095    1162    sox9AVC_peak_1  64      .       6.83351 10.17349
        6.43060 28
NT_039195.1     136741  136787  sox9AVC_peak_2  51      .       6.07423 8.71880
5.13959 13
NT_039223.3     19614   19660   sox9AVC_peak_3  51      .       6.07423 8.71880
5.13959 26
NT_039223.3     287225  287271  sox9AVC_peak_4  51      .       6.07423 8.71880
5.13959 41
NT_039223.3     324155  324201  sox9AVC_peak_5  39      .       5.31495 7.31485
3.95571 8
NT_039247.2     78934   79096   sox9AVC_peak_6  257     .       15.97444
30.52283        25.71295        28
NT_039248.1     252817  252866  sox9AVC_peak_7  106     .       9.11135 14.79398
        10.69148        10
NT_039256.1     45543   45590   sox9AVC_peak_8  51      .       6.07423 8.71880
5.13959 25
NT_039256.1     107149  107197  sox9AVC_peak_9  77      .       7.59279 11.67366
        7.79515 21
NT_039257.4     68154   68204   sox9AVC_peak_10 64      .       6.83351 10.17349
        6.43060 29
```

In [282]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
```

In [284]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
```

In [285]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
```

In [286]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
```

We now have bed files that only contain the first four columns.

In [283]:

```
NT_039195.1     1095    1162    sox9AVC_peak_1
NT_039195.1     136741  136787  sox9AVC_peak_2
NT_039223.3     19614   19660   sox9AVC_peak_3
NT_039223.3     287225  287271  sox9AVC_peak_4
NT_039223.3     324155  324201  sox9AVC_peak_5
NT_039247.2     78934   79096   sox9AVC_peak_6
NT_039248.1     252817  252866  sox9AVC_peak_7
NT_039256.1     45543   45590   sox9AVC_peak_8
NT_039256.1     107149  107197  sox9AVC_peak_9
NT_039257.4     68154   68204   sox9AVC_peak_10
```

We will use this bed files later to look at our peaks in IGV. First we'll perform a GREAT analysis of the peaks.

GREAT analysis

GREAT allows to determine biological processes that are connected to the peaks determined in ChIP sequencing. We first have to remove the NT and NW lines of our bed files because GREAT does not recognize this and will not the process the files.

GREAT was also not recognizing the chromosome names used in this file (the NC names to be exactly).

To solve this we go to the STAR folder (path:/mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR) and click on the chrName.txt file. Here we then changed all the NC numbers into the correct chromosomes using the identifier on https://www.ncbi.nlm.nih.gov/genome/?term=mus+musculus (https://www.ncbi.nlm.nih.gov/genome/?term=mus+musculus). We then had to perform all the above steps again to obtain the correct chromosome identifier.

In [297]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC/macs2
```

```
NT_039195.1     1100    1146    sox9AVCControle_peak_1
NT_039247.2     78936   79095   sox9AVCControle_peak_2
NT_039248.1     252817  252865  sox9AVCControle_peak_3
NT_039257.4     135025  136207  sox9AVCControle_peak_4
NT_039257.4     136275  136510  sox9AVCControle_peak_5
NT_039257.4     137673  137897  sox9AVCControle_peak_6
NT_039257.4     138800  139325  sox9AVCControle_peak_7
NT_039257.4     140793  141012  sox9AVCControle_peak_8
NT_039257.4     142561  142704  sox9AVCControle_peak_9
NT_039257.4     143691  143849  sox9AVCControle_peak_10
```

In [304]:

In [306]:

```
NW_004058051.1  4089    4154    sox9AVCControle_peak_523
NW_004058051.1  29038   29117   sox9AVCControle_peak_524
NW_004058051.1  29221   29275   sox9AVCControle_peak_525
NW_004058053.1  142510  142556  sox9AVCControle_peak_526
NW_004450263.1  212967  213013  sox9AVCControle_peak_527
NW_004450263.1  294202  294250  sox9AVCControle_peak_528
NW_004450263.1  476588  476638  sox9AVCControle_peak_529
NW_004450263.1  486947  486993  sox9AVCControle_peak_530
NW_004450263.1  730720  730766  sox9AVCControle_peak_531
NW_006763129.1  142985  143036  sox9AVCControle_peak_532
```

In [307]:

In [308]:

```
chr1    3015811 3015857 sox9AVCControle_peak_551
chr1    3041287 3041353 sox9AVCControle_peak_552
chr1    3043189 3043249 sox9AVCControle_peak_553
chr1    3139059 3139111 sox9AVCControle_peak_554
chr1    3241581 3241638 sox9AVCControle_peak_555
chr1    3440727 3440776 sox9AVCControle_peak_556
chr1    3715894 3715965 sox9AVCControle_peak_557
chr1    3764919 3764965 sox9AVCControle_peak_558
chr1    3767731 3767778 sox9AVCControle_peak_559
chr1    3768768 3768819 sox9AVCControle_peak_560
```

We then did the same for the second file but now we did everything in one command.

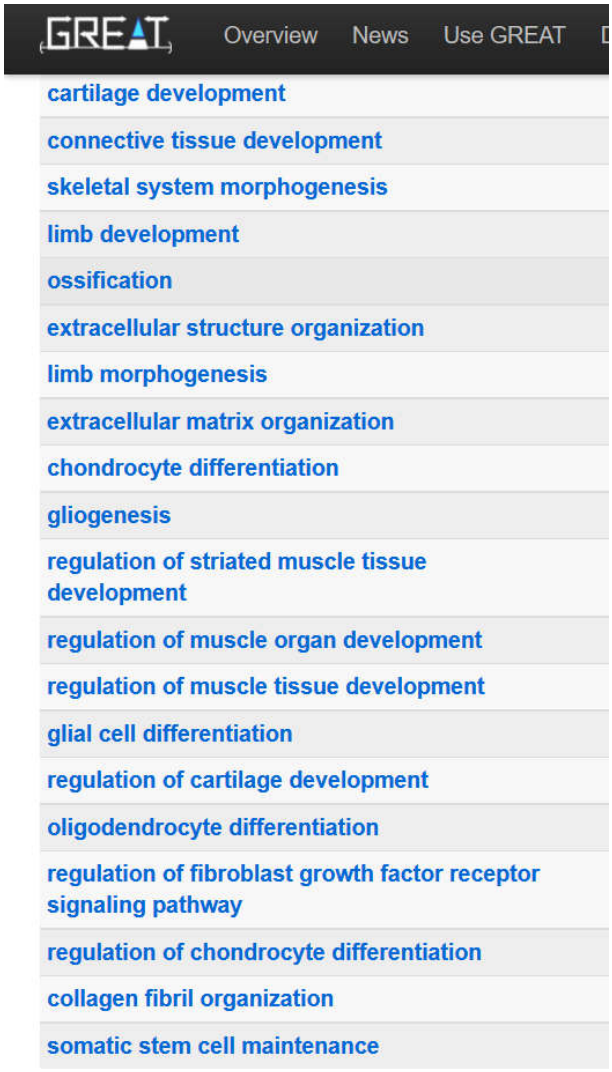
In [311]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/limb/macs2
```

In [312]:

```
chr1    9564522 9564679 sox9limbControle_peak_66
chr1    10037881        10038177        sox9limbControle_peak_67
chr1    11774853        11774976        sox9limbControle_peak_68
chr1    12000626        12000891        sox9limbControle_peak_69
chr1    12192818        12192954        sox9limbControle_peak_70
chr1    12644765        12644871        sox9limbControle_peak_71
chr1    12692906        12693022        sox9limbControle_peak_72
chr1    12836210        12836320        sox9limbControle_peak_73
chr1    13771092        13771193        sox9limbControle_peak_74
chr1    14513616        14513759        sox9limbControle_peak_75
```

As you can see, we now have to bed files that contain lines that GREAT will be able to read. The NT and NW lines have been removed and the NC lines have been converted to the correct chromosome name. We proceeded to upload both bed files to see which processes were linked to the peaks.

The sox9limbControle_peaksgrep2.bed contains the peaks called in limb mesenchyme and shows results that match with what we know about sox9. As you can see the sox9 peaks in the limb are connected to processes such as cartilage development, limb development, gliogenesis and chondrocyt differentation.



Unfortunately we did not find anything for sox9AVCControle_peaksgrep2.bed because there were to many peaks according to the site. Also remember that this was the fasta file with the overall very bad quality. In the next part we tried to improve the quality of this fastq file to re-run the MACS peak calling.

Using the following command we filter sequences of which 80% of the bases have a quality score of 30 or higher. This will allow us to increase the per base sequence quality.

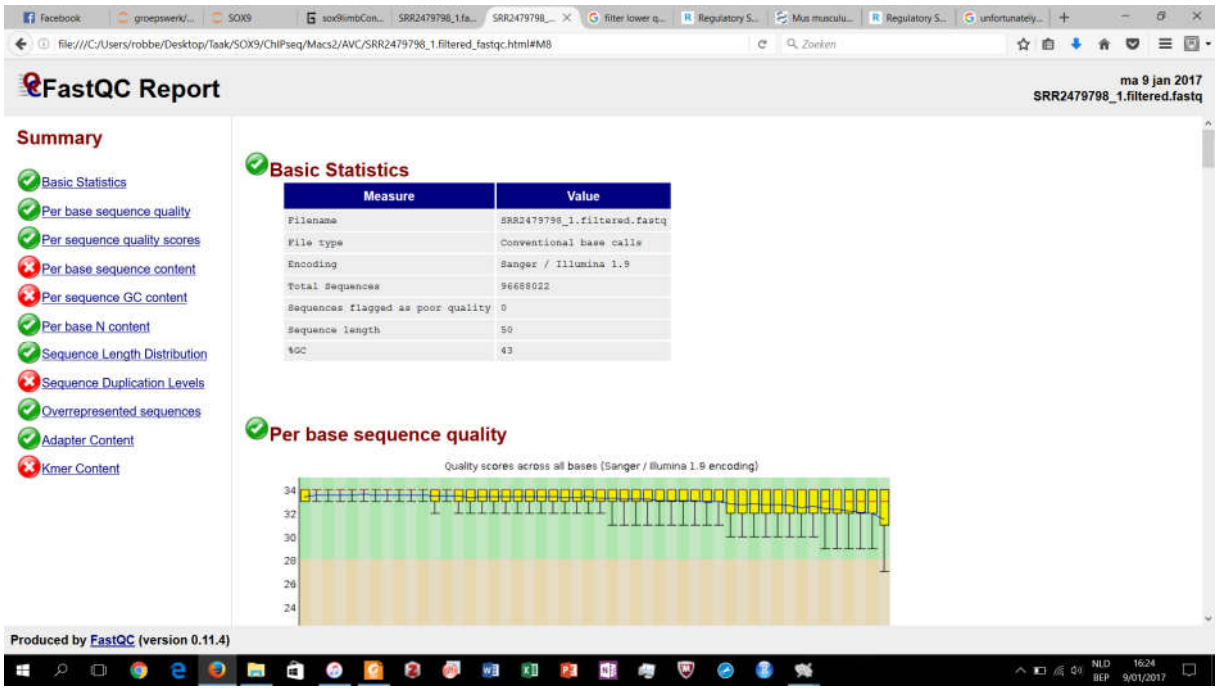
In [336]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
```

```
fastq_quality_filter: failed to open input file 'SRR2479798_1.fastq': No such fi
le or directory
```

In [324]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIP
```

```
Started analysis of SRR2479798_1.filtered.fastq
Approx 5% complete for SRR2479798_1.filtered.fastq
Approx 10% complete for SRR2479798_1.filtered.fastq
Approx 15% complete for SRR2479798_1.filtered.fastq
Approx 20% complete for SRR2479798_1.filtered.fastq
Approx 25% complete for SRR2479798_1.filtered.fastq
Approx 30% complete for SRR2479798_1.filtered.fastq
Approx 35% complete for SRR2479798_1.filtered.fastq
Approx 40% complete for SRR2479798_1.filtered.fastq
Approx 45% complete for SRR2479798_1.filtered.fastq
Approx 50% complete for SRR2479798_1.filtered.fastq
Approx 55% complete for SRR2479798_1.filtered.fastq
Approx 60% complete for SRR2479798_1.filtered.fastq
Approx 65% complete for SRR2479798_1.filtered.fastq
Approx 70% complete for SRR2479798_1.filtered.fastq
Approx 75% complete for SRR2479798_1.filtered.fastq
Approx 80% complete for SRR2479798_1.filtered.fastq
Approx 85% complete for SRR2479798_1.filtered.fastq
Approx 90% complete for SRR2479798_1.filtered.fastq
Approx 95% complete for SRR2479798_1.filtered.fastq
Approx 100% complete for SRR2479798_1.filtered.fastq
Analysis complete for SRR2479798_1.filtered.fastq
```

As you can see the per base sequence quality is much better but now we seem to have a lot of duplicated sequences. We thought about using fastx_collapser to remove duplicated sequences but this also converts the fastq file into a fasta upon which it impossible to apply fastq quality control. We therefore filtered the original fastq files again but with a lower threshhold (24) in the hope of obtaining high quality reads with a lower percentage of replicated sequences.



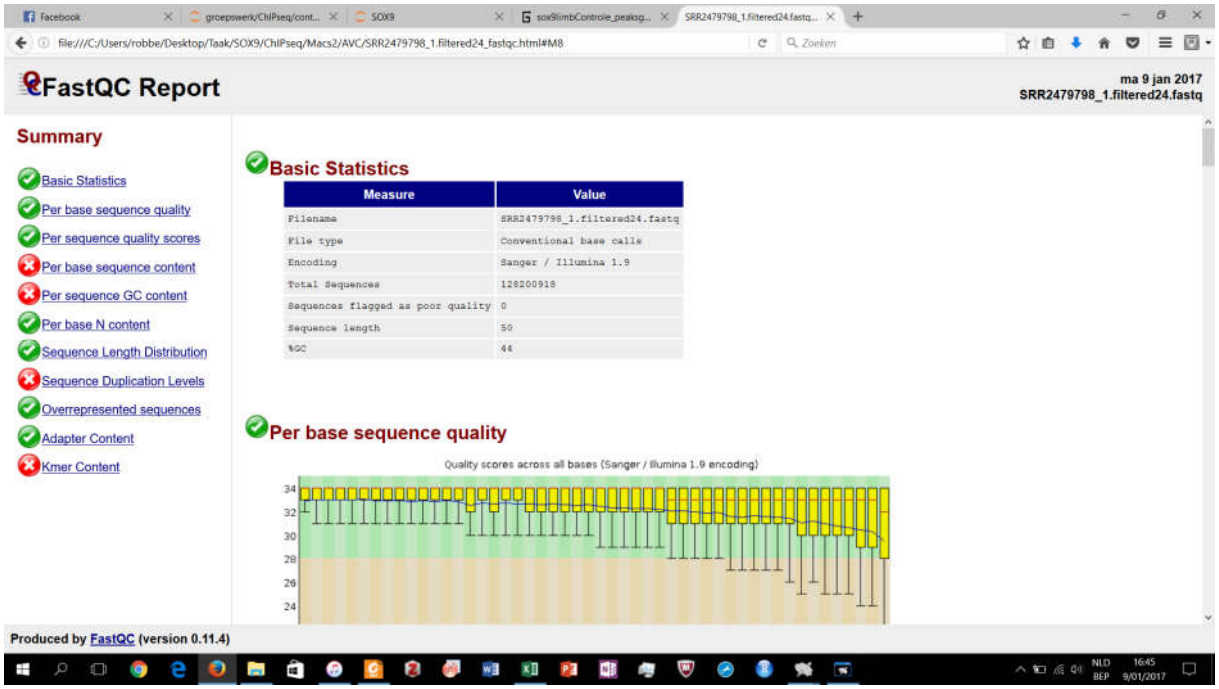
Using the next command, we filter the original fastq file again but now we apply a less strict filter of 24 instead of 30 which we used during the first filter step.

In [339]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
```

In [343]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC
/usr/local/bin/FastQC/fastqc /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIP
```

```
Started analysis of SRR2479798_1.filtered24.fastq
Approx 5% complete for SRR2479798_1.filtered24.fastq
Approx 10% complete for SRR2479798_1.filtered24.fastq
Approx 15% complete for SRR2479798_1.filtered24.fastq
Approx 20% complete for SRR2479798_1.filtered24.fastq
Approx 25% complete for SRR2479798_1.filtered24.fastq
Approx 30% complete for SRR2479798_1.filtered24.fastq
Approx 35% complete for SRR2479798_1.filtered24.fastq
Approx 40% complete for SRR2479798_1.filtered24.fastq
Approx 45% complete for SRR2479798_1.filtered24.fastq
Approx 50% complete for SRR2479798_1.filtered24.fastq
Approx 55% complete for SRR2479798_1.filtered24.fastq
Approx 60% complete for SRR2479798_1.filtered24.fastq
Approx 65% complete for SRR2479798_1.filtered24.fastq
Approx 70% complete for SRR2479798_1.filtered24.fastq
Approx 75% complete for SRR2479798_1.filtered24.fastq
Approx 80% complete for SRR2479798_1.filtered24.fastq
Approx 85% complete for SRR2479798_1.filtered24.fastq
Approx 90% complete for SRR2479798_1.filtered24.fastq
Approx 95% complete for SRR2479798_1.filtered24.fastq
Analysis complete for SRR2479798_1.filtered24.fastq
```

The per base sequence quality is good but there is still a problem with the sequence duplication levels. We choose the continue working with SRR2479798_1.filtered24.fastq because it has the lowest duplication level. We had thought about fastx_collapser to remove duplicated sequences but this also converts the fastq file into a fasta upon which it impossible to apply fastq quality control.



We made a new folder in which we stored the output of the AVC sample after it has been filtered.

In [326]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq
```

In [345]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC_Filtered
pwd
STAR --genomeDir /mnt/storage/r0305394/jupyternotebooks/groepswerk/STAR --runThread
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC_Filtered
Jan 09 16:51:48 ..... started STAR run
Jan 09 16:51:48 ..... loading genome
Jan 09 16:55:19 ..... started mapping
Jan 09 17:20:49 ..... finished successfully
```

We converted the sam file into a bam using samtools.

In [346]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC_Filtered
```

```
[bam_sort_core] merging from 26 files...
```

We then generated an index file in orde to be able to read the bam file in IGV.

In [347]:

We then performed another peak calling but this time we use the filtered fastq file of the AVC.

In [348]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC_Filtered
pwd
macs2 callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC_Fil
```

```
/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC_Filtered
INFO  @ Mon, 09 Jan 2017 17:38:28:
# Command line: callpeak -t /mnt/storage/r0305394/jupyternotebooks/groepswerk/Ch
IPseq/AVC_Filtered/SRR2479798_1.filtered24.bam -c /mnt/storage/r0305394/jupytern
otebooks/groepswerk/ChIPseq/controle/SRR2479800.bam --outdir /mnt/storage/r03053
94/jupyternotebooks/groepswerk/ChIPseq/AVC_Filtered -n sox9AVCFilter24 -g mm -q
0.01
# ARGUMENTS LIST:
# name = sox9AVCFilter24
# format = AUTO
# ChIP-seq file = ['/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AV
C_Filtered/SRR2479798_1.filtered24.bam']
# control file = ['/mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/con
trole/SRR2479800.bam']
# effective genome size = 1.87e+09
# band width = 300
# model fold = [5, 50]
# qvalue cutoff = 1.00e-02
# Larger dataset will be scaled towards smaller dataset.
# Range for calculating regional lambda is: 1000 bps and 10000 bps
# Broad region calling is off

INFO  @ Mon, 09 Jan 2017 17:38:28: #1 read tag files...
INFO  @ Mon, 09 Jan 2017 17:38:28: #1 read treatment tags...
INFO  @ Mon, 09 Jan 2017 17:38:28: Detected format is: BAM
INFO  @ Mon, 09 Jan 2017 17:38:28: * Input file is gzipped.
INFO  @ Mon, 09 Jan 2017 17:38:32:  1000000
INFO  @ Mon, 09 Jan 2017 17:38:36:  2000000
INFO  @ Mon, 09 Jan 2017 17:38:39:  3000000
INFO  @ Mon, 09 Jan 2017 17:38:43:  4000000
INFO  @ Mon, 09 Jan 2017 17:38:47:  5000000
INFO  @ Mon, 09 Jan 2017 17:38:50:  6000000
INFO  @ Mon, 09 Jan 2017 17:38:54:  7000000
INFO  @ Mon, 09 Jan 2017 17:38:58:  8000000
INFO  @ Mon, 09 Jan 2017 17:39:01:  9000000
INFO  @ Mon, 09 Jan 2017 17:39:05:  10000000
INFO  @ Mon, 09 Jan 2017 17:39:08:  11000000
INFO  @ Mon, 09 Jan 2017 17:39:11:  12000000
INFO  @ Mon, 09 Jan 2017 17:39:15:  13000000
INFO  @ Mon, 09 Jan 2017 17:39:18:  14000000
INFO  @ Mon, 09 Jan 2017 17:39:22:  15000000
INFO  @ Mon, 09 Jan 2017 17:39:26:  16000000
INFO  @ Mon, 09 Jan 2017 17:39:29:  17000000
INFO  @ Mon, 09 Jan 2017 17:39:33:  18000000
INFO  @ Mon, 09 Jan 2017 17:39:37:  19000000
INFO  @ Mon, 09 Jan 2017 17:39:41:  20000000
INFO  @ Mon, 09 Jan 2017 17:39:44:  21000000
INFO  @ Mon, 09 Jan 2017 17:39:48:  22000000
INFO  @ Mon, 09 Jan 2017 17:39:51:  23000000
INFO  @ Mon, 09 Jan 2017 17:39:55:  24000000
INFO  @ Mon, 09 Jan 2017 17:39:58:  25000000
INFO  @ Mon, 09 Jan 2017 17:40:02:  26000000
INFO  @ Mon, 09 Jan 2017 17:40:06:  27000000
INFO  @ Mon, 09 Jan 2017 17:40:09:  28000000
INFO  @ Mon, 09 Jan 2017 17:40:13:  29000000
INFO  @ Mon, 09 Jan 2017 17:40:17:  30000000
INFO  @ Mon, 09 Jan 2017 17:40:20:  31000000
INFO  @ Mon, 09 Jan 2017 17:40:24:  32000000
INFO  @ Mon, 09 Jan 2017 17:40:28:  33000000
INFO  @ Mon, 09 Jan 2017 17:40:31:  34000000
INFO  @ Mon, 09 Jan 2017 17:40:35:  35000000
INFO  @ Mon, 09 Jan 2017 17:40:38:  36000000
INFO  @ Mon, 09 Jan 2017 17:40:41:  37000000
INFO  @ Mon, 09 Jan 2017 17:40:45:  38000000
INFO  @ Mon, 09 Jan 2017 17:40:48:  39000000
```

As you can see the AVC sample has lot less peaks compared to the sample that was not filtered.

In [353]:

```
1092 sox9AVCFilter24_peaks.narrowPeak
```

The file is then converted into a bed file.

In [349]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC_Filtered
```

In [3]:
```
cd /mnt/storage/r0305394/jupyternotebooks/groepswerk/ChIPseq/AVC_Filtered
```

```
NT_039247.2     78960   79057   sox9AVCFilter24_peak_1
NT_039257.4     135051  136004  sox9AVCFilter24_peak_2
NT_039257.4     136146  136208  sox9AVCFilter24_peak_3
NT_039257.4     136286  136358  sox9AVCFilter24_peak_4
NT_039257.4     136410  136489  sox9AVCFilter24_peak_5
NT_039257.4     137674  137899  sox9AVCFilter24_peak_6
NT_039257.4     138800  139136  sox9AVCFilter24_peak_7
NT_039257.4     139252  139325  sox9AVCFilter24_peak_8
NT_039257.4     140794  141012  sox9AVCFilter24_peak_9
NT_039257.4     142559  142706  sox9AVCFilter24_peak_10
```

Using the following command, we removed the NT and NW lines out of bed files. These lines prevent GREAT from running an analysis. Since the lines with chromosomes are kept this doesn't influence the results that much.

In [350]:

In [352]:

```
chr1    8828433 8828479 sox9AVCFilter24_peak_187
chr1    9021840 9021886 sox9AVCFilter24_peak_188
chr1    10037889        10037987        sox9AVCFilter24_peak_189
chr1    10105763        10105809        sox9AVCFilter24_peak_190
chr1    12644761        12644846        sox9AVCFilter24_peak_191
chr1    17014076        17014143        sox9AVCFilter24_peak_192
chr1    23146780        23146860        sox9AVCFilter24_peak_193
chr1    31129934        31129982        sox9AVCFilter24_peak_194
chr1    37063233        37063284        sox9AVCFilter24_peak_195
chr1    38757840        38757911        sox9AVCFilter24_peak_196
```

For some reason that is unknown to us, GREAT also gives us nothing when we upload the filtered AVC file. It is possible that the filtering was to harsh and filtered away most of the results. In the following part we use IGV to analyze our peaks. This will allow us to say something about the peaks in the AVC sample, the sample that did not give us in results in GREAT.

IGV analysis of the peaks

We now have three bed files containing the peaks: 2 files with peaks in the AVC (without filter and with filter) and 1 file with peaks in the limb mesenchyme. We can now upload these files in IGV together with their corresponding bam files to study the peaks in IGV.

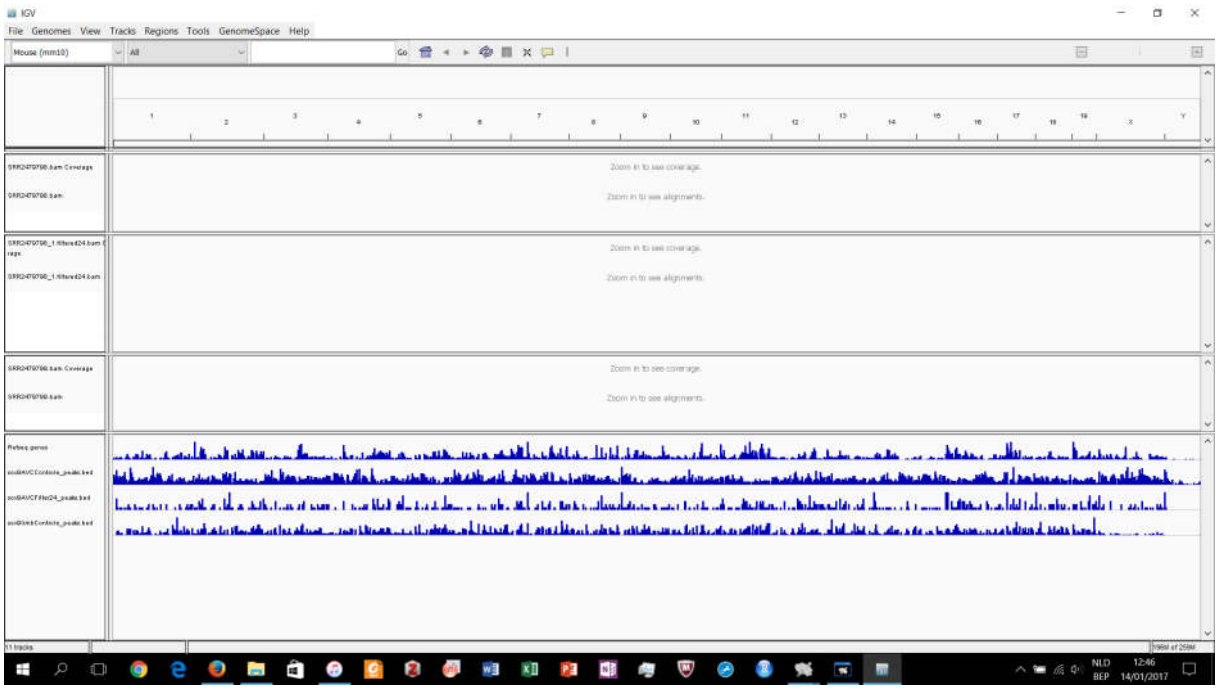
The top three lines are our bam files:

1. SRR247979.bam (= AVC file without filtering)
2. SRR2479798_1.filtered24.bam (= AVC file with filtering)
3. SRR2479799.bam (= limb mesenchyme file)

The bottom three lines represent the bed fils containing the peaks:

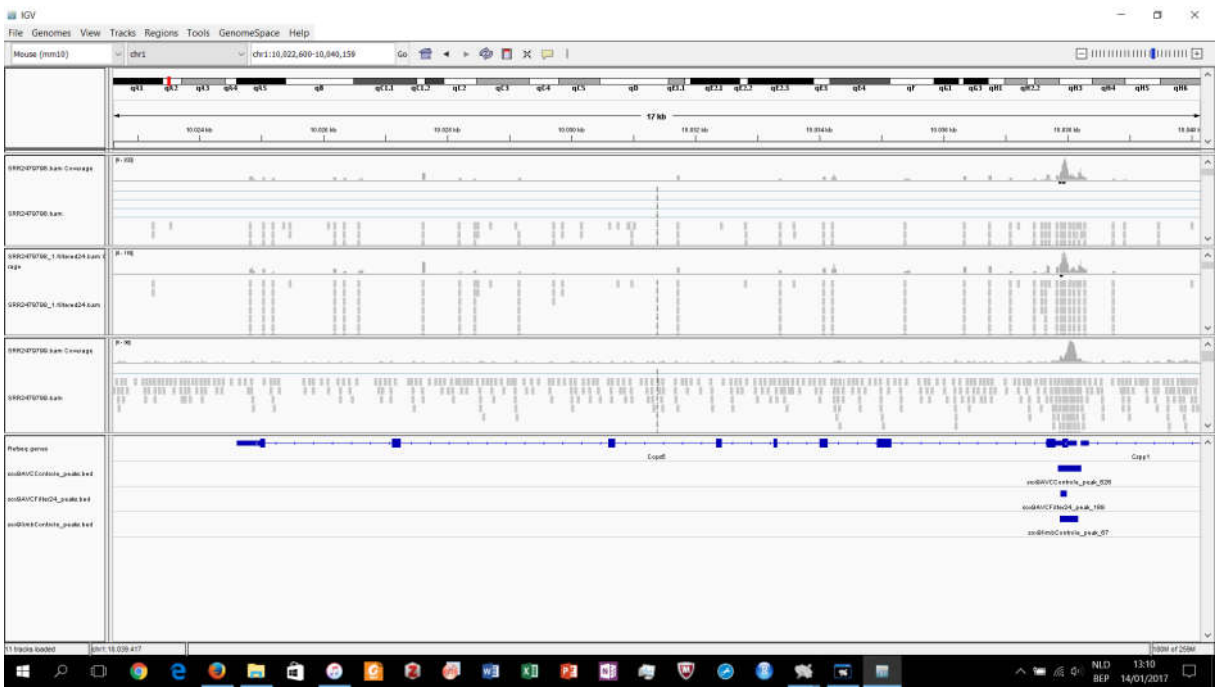
1. sox9AVCControle_peaks.bed (= peaks from the AVC file without filtering)
2. sox9AVCFilter24_peaks.bed (= peaks from the AVC file with filtering)
3. sox9limbControle_peaks.bed (= peaks from the limb file)

As you can see the filtered AVC file shows a lot less peaks than the unfiltered AVC file. The limb file also contains a lot less peaks than the unfiltered AVC file.

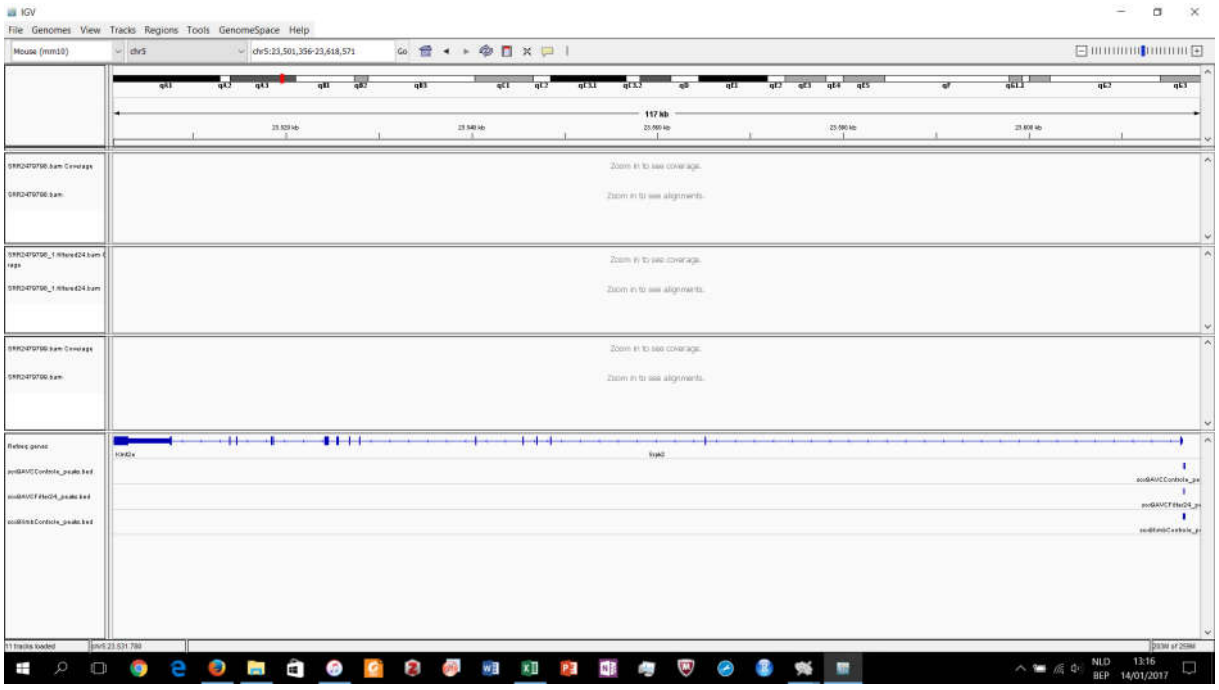


In the paper, the researchers mentioned several genes that have peaks. Although we have used a different mapping method (STAR) and a different genome (mm10), we first decided to see whether we were able to see peaks that are mentioned in the paper.

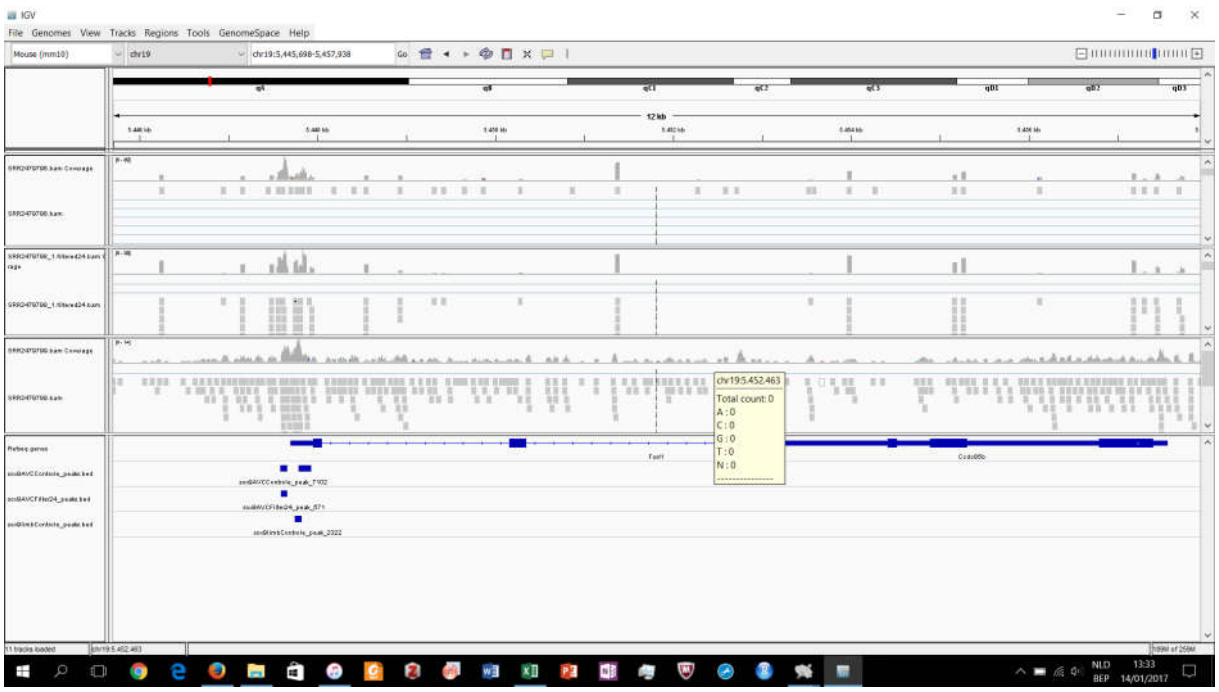
The first gene that we are going to look at is Cops5 also known as Jab1. It has been shown that a loss of Cops5 leads to shortened limbs and this is the result of impaired chondrogenesis. This loss of Cops5 resulted in a decreased expression of several genes including sox9 indicating some sort of relationship between sox9 and cops5 important for bone development (https://www.ncbi.nlm.nih.gov/pubmed/24604556 (https://www.ncbi.nlm.nih.gov/pubmed/24604556)). As you can see in the image below, there are peaks in all of our bed files at the transcriptional site of Cops5. This shows us that sox9 binds this site and regulates the expression of Cops5 in both heart and limb.



The next gene we looked at is Srpk2. This gene can promote cell proliferation and regulates progression of cell cycle (https://www.ncbi.nlm.nih.gov/pubmed/19592491 (https://www.ncbi.nlm.nih.gov/pubmed/19592491)). Again we can see peaks in all samples located at transcription site of Srpk2 indicating that sox9 regulates the expression of this gene in both heart and limb mesenchyme. This is also indicates that sox9 is an regulator of cell proliferation and cell cycle progression.

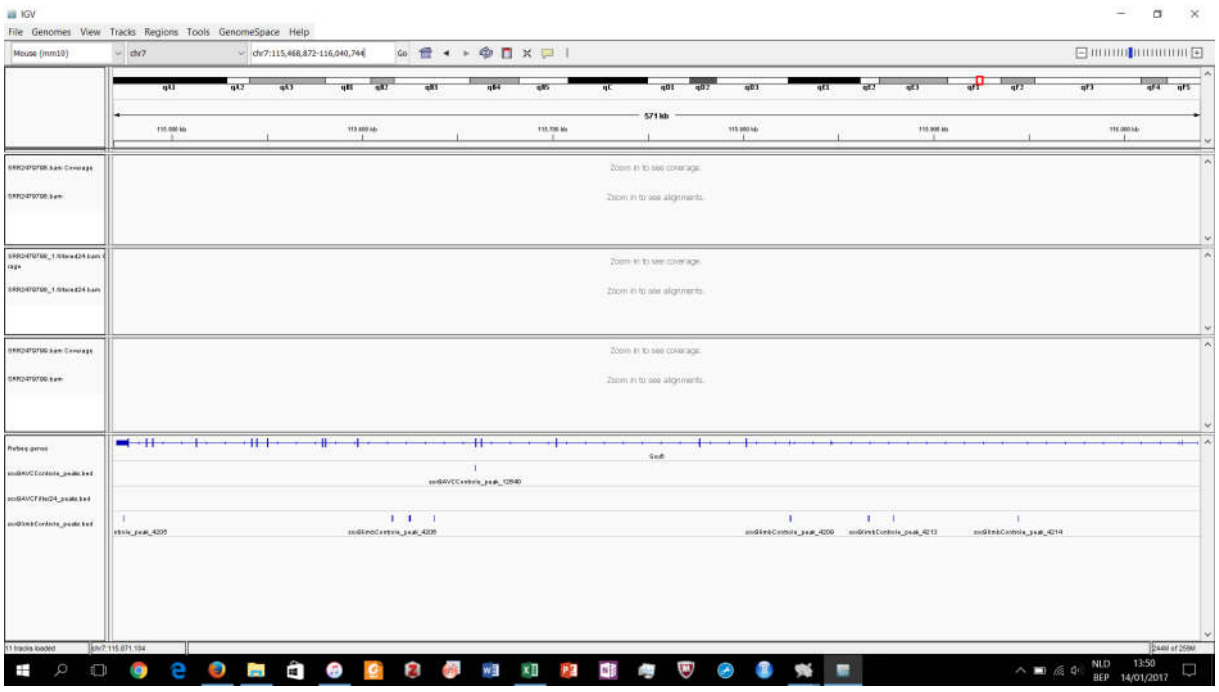


Another gene that was checked is Fols1. This gene generates a protein that is part of the AP-1 complex. This complex has a role in cell proliferation and cell survival (https://www.ncbi.nlm.nih.gov/pubmed/20060892 (https://www.ncbi.nlm.nih.gov/pubmed/20060892)). Again we see peaks in all samples that are located at the transcription site of Fosl1.

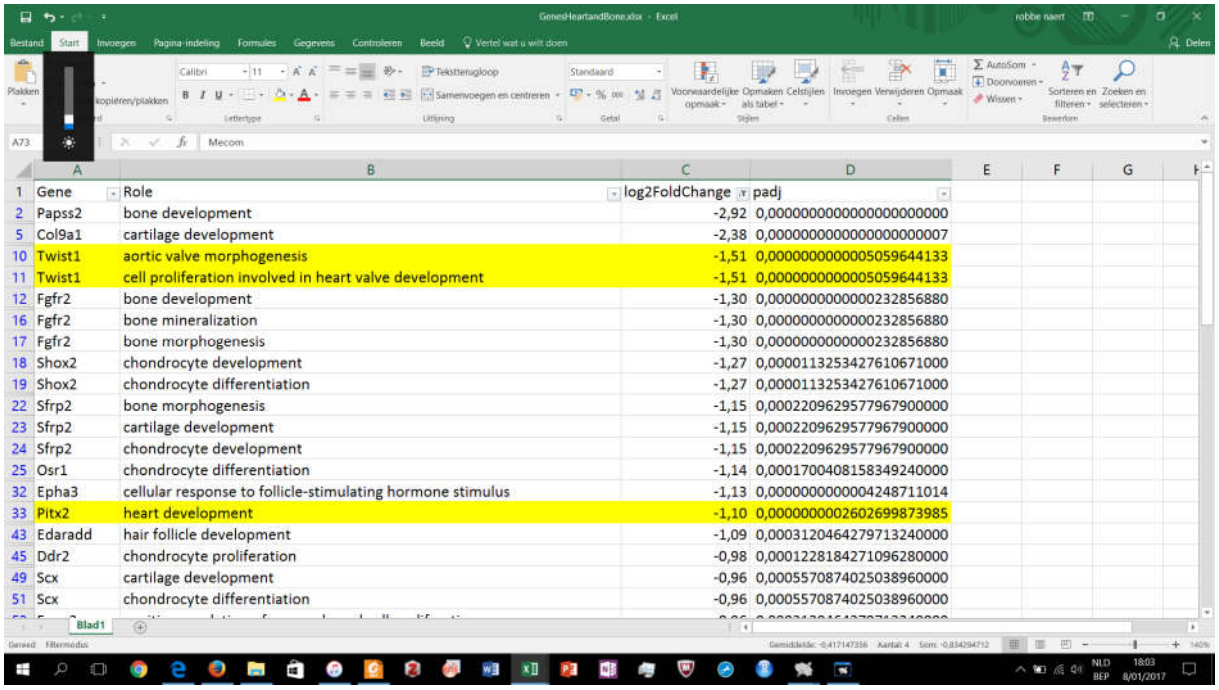


Several other genes that have been implicated in cell proliferation also had peaks in their transcriptional start site such as Eed, Hdac2, Trp53 (data not shown). Together with the data shown above, this indicates that Sox9 is an important regulator of cell proliferation. Moreover this function doesn't seem to be confined to a specific tissue as peaks were seen in both the heart (AVC) and limb.

Based on the GREAT analysis of the limb, we know that the peaks are associated with limb development and chondrogenesis. It has been shown that Sox9 regulates sox6 to regulate target genes in the developing limb (https://www.ncbi.nlm.nih.gov/pubmed/?term=L-sox5+and+sox+6+drive+expression (https://www.ncbi.nlm.nih.gov/pubmed/?term=L-sox5+and+sox+6+drive+expression)). We therefore looked if we could find any peaks in sox6 to demonstrate that sox9 regulates sox6. We able to identify several peaks in sox6 in the limb showing that this gene has several binding locations for sox9.

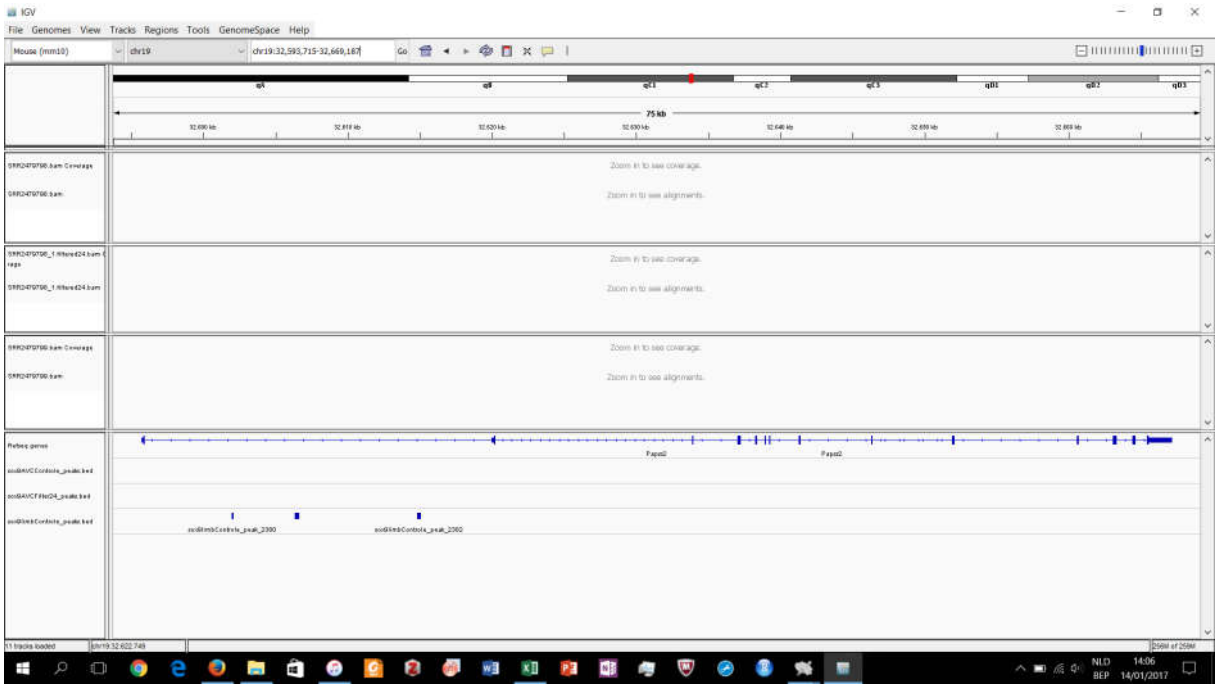


Another thing we did was to check for peaks in some of the genes that were downregulated (as shown in the RNA sequencing; see image below).

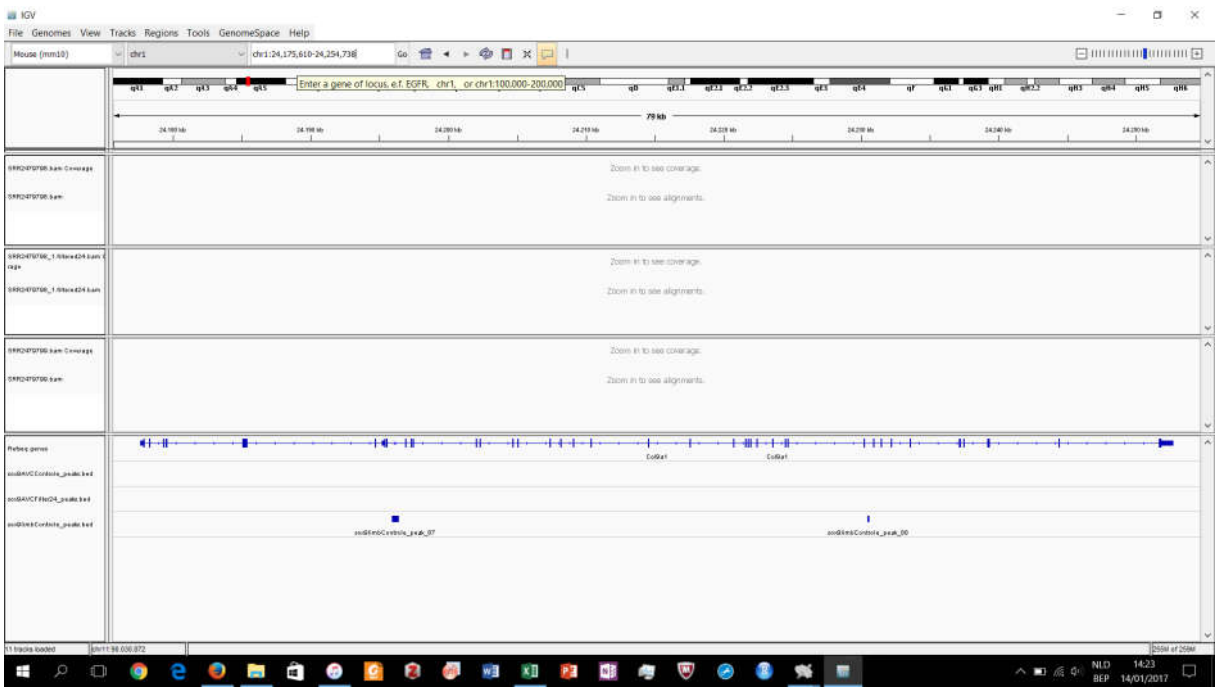


	Gene	Role	log2FoldChange	padj
2	Papss2	bone development	-2,92	0,000000000000000000000
5	Col9a1	cartilage development	-2,38	0,000000000000000000007
10	Twist1	aortic valve morphogenesis	-1,51	0,000000000000059644133
11	Twist1	cell proliferation involved in heart valve development	-1,51	0,000000000000059644133
12	Fgfr2	bone development	-1,30	0,000000000000232856880
16	Fgfr2	bone mineralization	-1,30	0,000000000000232856880
17	Fgfr2	bone morphogenesis	-1,30	0,000000000000232856880
18	Shox2	chondrocyte development	-1,27	0,000011325342761067 1000
19	Shox2	chondrocyte differentiation	-1,27	0,000011325342761067 1000
22	Sfrp2	bone morphogenesis	-1,15	0,000220962957796790 0000
23	Sfrp2	cartilage development	-1,15	0,000220962957796790 0000
24	Sfrp2	chondrocyte development	-1,15	0,000220962957796790 0000
25	Osr1	chondrocyte differentiation	-1,14	0,000170040815834924 0000
32	Epha3	cellular response to follicle-stimulating hormone stimulus	-1,13	0,000000000000424871 1014
33	Pitx2	heart development	-1,10	0,000000000260269987 3985
43	Edaradd	hair follicle development	-1,09	0,000312046427971324 0000
45	Ddr2	chondrocyte proliferation	-0,98	0,000122818427109628 0000
49	Scx	cartilage development	-0,96	0,000557087402503896 0000
51	Scx	chondrocyte differentiation	-0,96	0,000557087402503896 0000

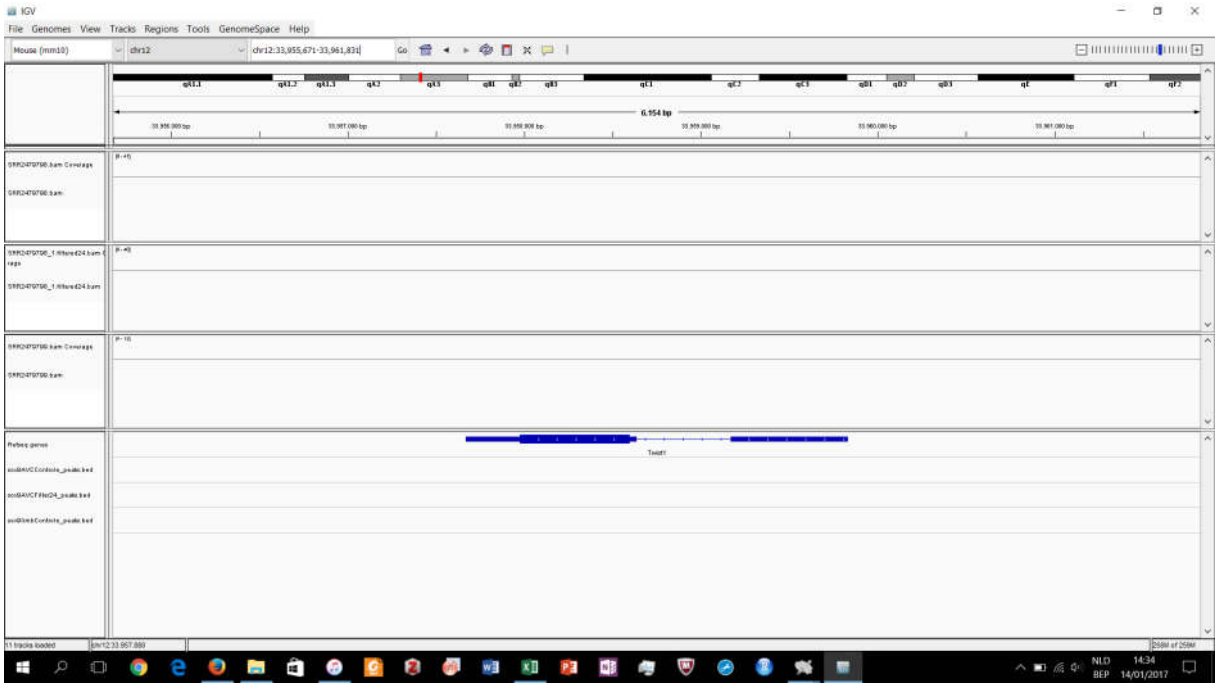
We first looked at Papss2. There are several peaks in the limb. However these peaks are not located close to a transcriptional start site. It is possible that sox9 has no effect on the transciptional start of this gene but influences the elongation or splicing.



Another gene we checked is Col9a1. This gene is important for cartilage development. As you can see we have two peaks in Col9a1. These peaks are located between exon 7 and 8 and between exon 29-30.



We also checked for peaks for Twist1. Twist1 plays an important role in heart valve development. unfortunately our data did not show any peaks for Twist although the researchers describe finding peaks in their data. Based on our data we would have to conclude that the decreased Twist1 expression we saw in the sox9KO (RNa sequencing data) is not due to a direct effect of sox9 on Twist1. It seems that sox9 regulates the expression of Twist 1 in an indirect way by downregulating an unknown transcription factor x which in turn influences Twist1 expression.



We also checked peaks for some other important TF mentioned earlier such as Mecom, Pitx2 and Hand2 (data not shown).

Mecom had two peaks: one peak in the AVC (without filtering) and one in the limb. This peak wasn't there in the AVC (with filtering) meaning that the filter step removed the peak.

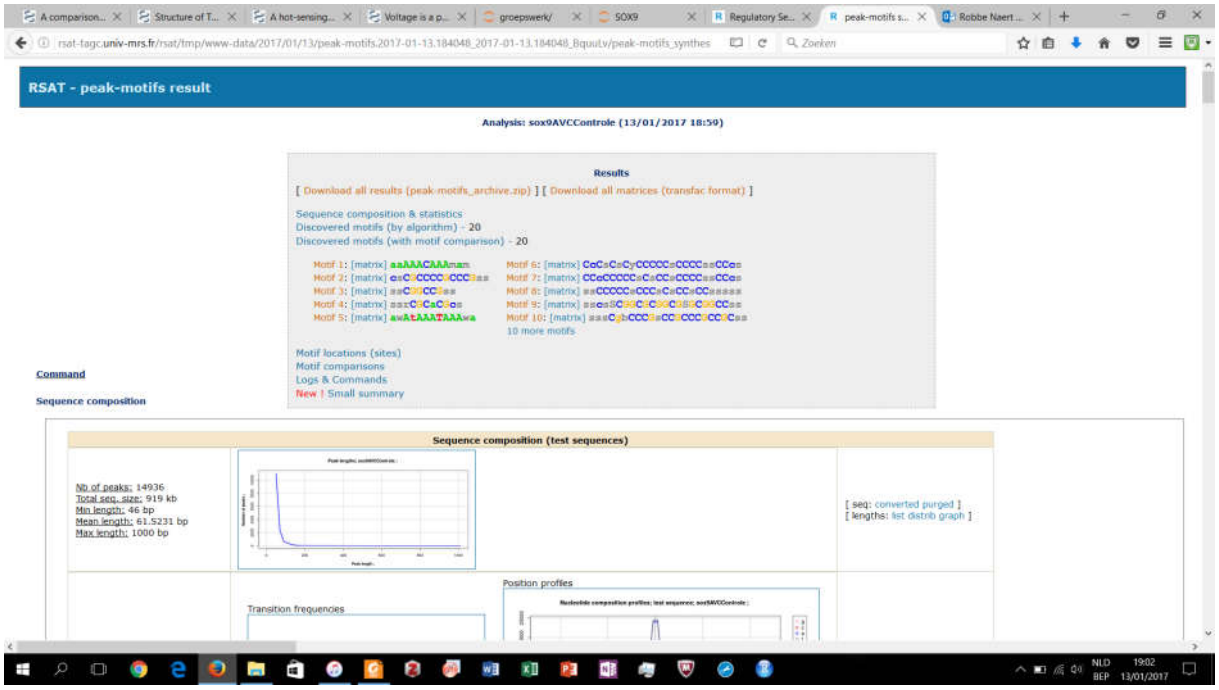
We did not find any peaks in Pitx2 or Hand2.

Motif discovery

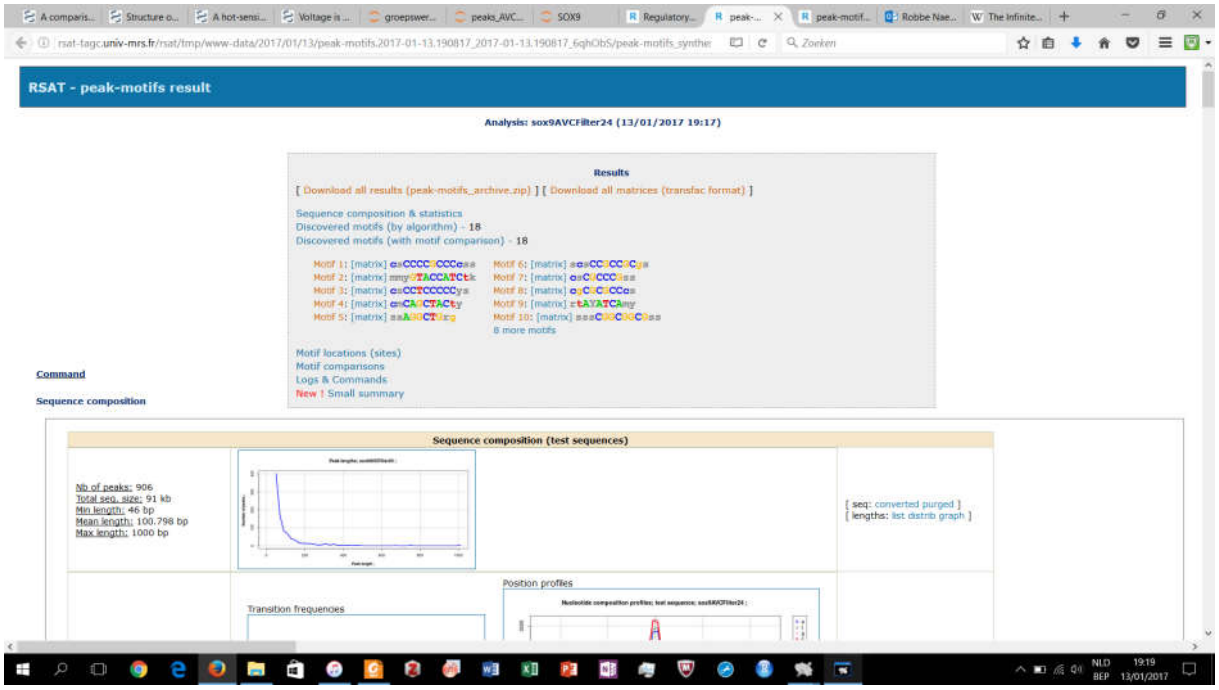
Another thing that can be done now that we have the bed files, is motif discovery to look at the binding motifs of sox9 in AVC and limb. To do this we first have to use fetch-sequence to the data of the bed file into a fasta file. This tool skipped the NT and NW files but since these files are only small areas we chose to go ahead and work with the fasta file only containing the chromosomes. After obtaining the faste files for both the AVC and limb we uploaded these to http://rsat-tagc.univ-mrs.fr/ (http://rsat-tagc.univ-mrs.fr/).

We determined the binding motifs in AVC (before and after filtering) and limb.

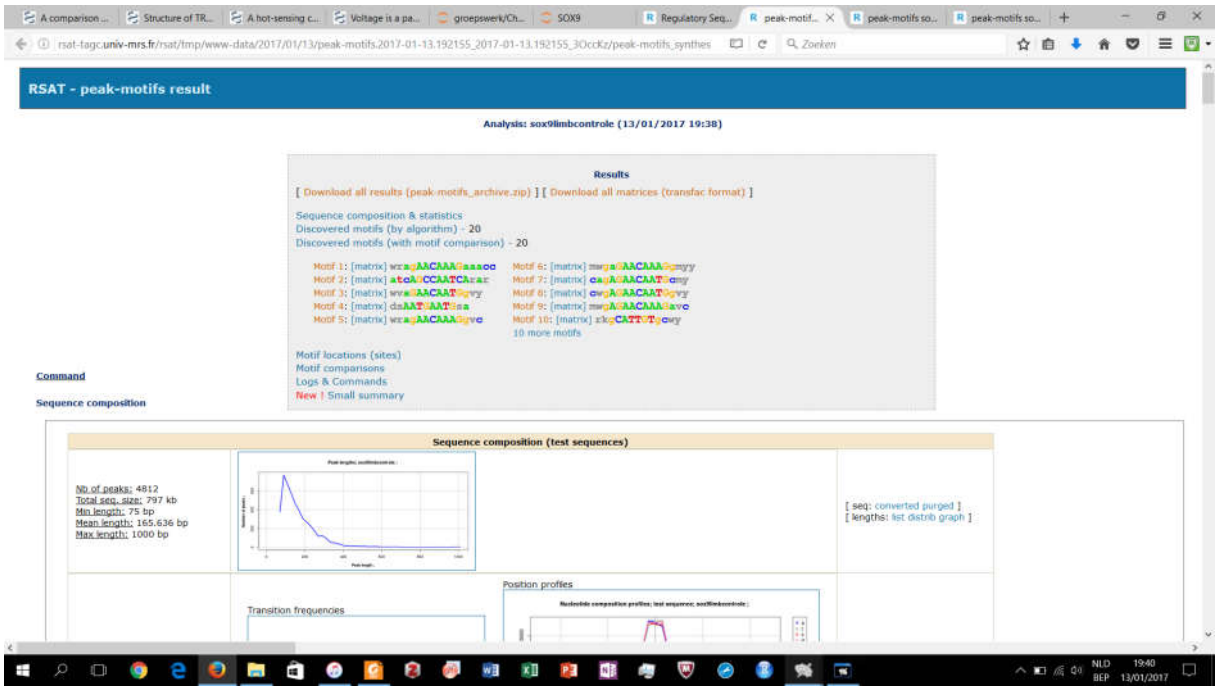
We will first take a look at the output generated by RSAT from the AVC file without filtering (AVCControle). RSAT gave us 20 motifs. The first 10 are shown in the image below. (http://rsat-tagc.univ-mrs.fr/rsat/tmp/www-data/2017/01/13/peak-motifs.2017-01-13.184048_2017-01-13.184048_BquuLv/peak-motifs_synthesis.html#discovered_motifs_10 (http://rsat-tagc.univ-mrs.fr/rsat/tmp/www-data/2017/01/13/peak-motifs.2017-01-13.184048_2017-01-13.184048_BquuLv/peak-motifs_synthesis.html#discovered_motifs_10))



We then determined binding motifs for sox9 in the AVC file with filtering (ACVFilter24). RSAT give us 18 motifs of which the first 10 are shown below. (http://rsat-tagc.univ-mrs.fr/rsat/tmp/www-data/2017/01/13/peak-motifs.2017-01-13.190817_2017-01-13.190817_6qhObS/peak-motifs_synthesis.html (http://rsat-tagc.univ-mrs.fr/rsat/tmp/www-data/2017/01/13/peak-motifs.2017-01-13.190817_2017-01-13.190817_6qhObS/peak-motifs_synthesis.html))



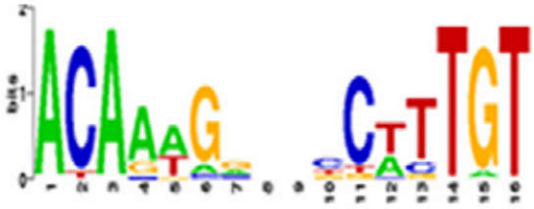
We then looked at the binding motifs for sox9 in the limb mesenchyme. RSAT was able to find 20 motifs. The first 10 are shown below. (http://rsat-tagc.univ-mrs.fr/rsat/tmp/www-data/2017/01/13/peak-motifs.2017-01-13.192155_2017-01-13.192155_3OccKz/peak-motifs_synthesis.html (http://rsat-tagc.univ-mrs.fr/rsat/tmp/www-data/2017/01/13/peak-motifs.2017-01-13.192155_2017-01-13.192155_3OccKz/peak-motifs_synthesis.html))



Now that we have binding motifs of all of our samples, we first decided to compare what we found with what the researcher found in their paper. In the paper they show two binding motifs that they found: SOX monomer motif and SOX dimer motif. They do no mention if these binding motifs were found in the limb or in the AVC or in both. The binding motifs mentioned in the paper are shown in the image below.

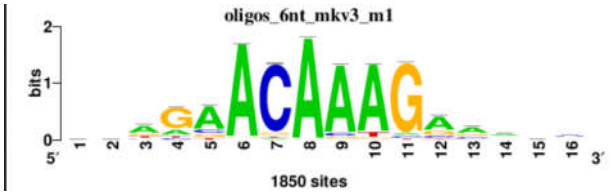


SOX monomer motif

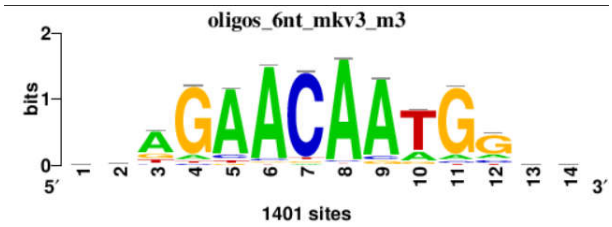


SOX dimer motif

We then checked the binding motifs we had found to see if any were the same. We did not find any match for the SOX monomer motif in our AVCControle and AVCFilter24 samples. We did however find a motif similar to the SOX monomer motif in the limbControle sample. As you can see the motif belows, which was obtained from the limb mesenchyme, shows big similarities with SOX monomer motif. They both have AACAAAG motif. Moreover motif 5, 6 and 9 showed the same similar motif.



The limb mesenchyme had another predicted motif binding site that was similar to the SOX monomer. As you can see this limb binding motif (image below) shares the AACAATG binding motif with the SOX monomer. Moreover motif 7 and 8 showed the same similar motif.



We were unable to verify the SOX dimer binding motif in our samples as none of the predicted binding motifs that we found showed a dimer pattern. We also checked for identical binding motifs in the AVC and limb. We did not find any which leads us to the conclusion that based on our data is seems that SOX9 has different binding motifs in different tissues.

In conclusion, we were able to show the role of sox9 in the limb as we discovered several peaks linked to functions such as cartilage development, chondorocyte differentation and bone development. However we were unable to find similar results in the AVC sample.

IGV allowed us to check the peaks in the AVC samples. We were able to confirm some results obtained in the paper but we did not find any peaks for the important genes involved in heart development such as Twist1, Pitx2, Hand2 and Sox4.

Motif discovery also showed no similar binding motifs in the limb and heart leading us to the conclusion that sox9 binds different motifs depending on the tissue.