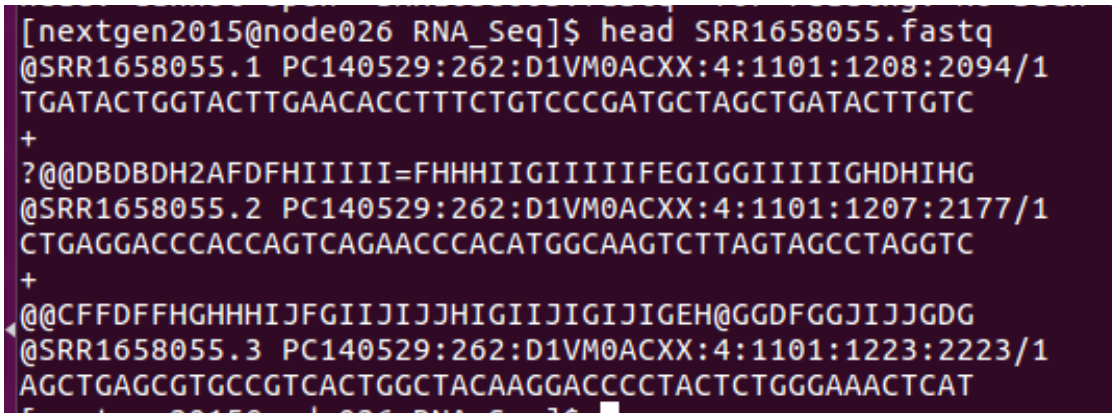
Pre-Practical Data Processing:

Downloaded initial files in .sra format to local machine using the “GEO” accession numbers.
I then proceeded to download sratools and utilised this to convert all files to fasta format.
I transfered all of these files to the cluster using “sftp”

I made a new RNA Seq directory: cd /data4/nextgen2015/users/17232658/
mkdir RNA_Seq
I transferred these files to the cluster to my directory:
/data4/nextgen2015/users/17232658/RNA_Seq/

Biological Background: *Sprouty* genes (*Spry1,2,3,4*) encode RTK feedback inhibitors required for development of the kidney, inner ear, and other organs. Spry proteins have tumor suppressor activity, and their expression levels are commonly downregulated in cancer, leading to aberrant amplification of RTK pathways, while their re-expression inhibits malignant growth . The Ras-ERK (RTK pathway) signaling axis in part regulates gene expression through control of activating and repressive epigenetic mechanisms.
Sample’s were obtained from mouse embroyonic fibroblasts, and seperated into a wild-type and *Sr* deleted cohort. The WT had an empty adenovirus vector transmitted where delted sampkes used a CRE recombinase deletion procedure to knock out the *Sprty* gene’s. These cohort’s were subdivided in 3 to “Unsync” (Freely growing = U), “Starve” (Serum starved = S) & FGF-treated (F). *FGF* is a fibroblast growth factor, which contains a “TK” domain and subsequently triggers tigeering of the “Ras-Erk” axis.
In this study, they found that chronic Ras-Erk signaling mediated by *Spry* loss leads to inappropriate gene activation, which correlates with dynamic changes in H3K27ac at S’es (super enhancer’s) and Te’s (typical enhacer’s). The below analysis will compare the results to this analysis.

Data Processing:
Performed “sanity check” on first .fastq file:



To obtain the mouse genome I proceeded to the USC Genome browser and download the mm9.2bit
I then downloaded the “twoBitToFa” tool, made it executable and converted the .2bit format to fasta.
I then proceeded to “sftp” this to the cluster to my directory

I performed a “sanity check” on this genome:

```
[nextgen2015@node026 RNA_Seq]$ head mm9.fa
>chr1
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
```

One would expect the“N” nucleotidestoo occur as this is a telomeric region with highly repetitive sequences, and therefore the bases are particularly diffiult to call.

Therefore, I selected the to view the 1,000,000 – 1,000,010 line of text using:
sed -n 1000000,1000010p mm9.fa

```
[nextgen2015@node026 RNA_Seq]$ sed -n 1000000,1000010p mm9.fa
TTTATATTGATGAAAAGCTATAGATAACATTAGGAAGAACTACTATAAAT
GAAGTCCGCAAAGTGTGGTGGAATAACTGTCAACATTTTTGTGTGGTATA
AAATATTTCCATGATACTATAGTACCAAAATGCTAATTCATTATAAAATG
AAAGTGTTAACTTTCAAATGATAAACCATAAACAAAATCTGAGACACACA
ACTACAGATAATGTCATTGGAATTTAGAATTTACATCATACTGGTGTTCT
CATATTATGCCTAATAACAATGATATTGTGTCTGATATGTTTTAATCCAA
ATATTAGCATGTGACATAATTAGAATAACAAAATTTTGGTGATATGAAAA
TTCTTGACTTGTTATCTTTAAGAGTCTTGCTAAAGATGCAGTCTCATTAA
AGTCAGTGGGTGATGAAAAATGTTTACAAGTTGTGGAAAATGAGGTAAGT
GTGTAGGATTATGAAAATGTGATAACTAAATGGAATATAAAAGTCTGGAT
GAAATCTAGTACAATAAAGATAATTGAAACTCATATAAAGTCAGGTCATT
```

1. QC:
Commands:
cd /data4/nextgen2015/users/17232658/
mkdir RNA_Seq
cd /data4/nextgen2015/users/17232658/RNA_Seq/

nano do_qc.sh
#!/bin/bash
for f in *.fastq;
do
fastqc $f;
done
multiqc .;

chmod a+x do_qc.sh

module load fastqc

./do_qc.sh

After this:

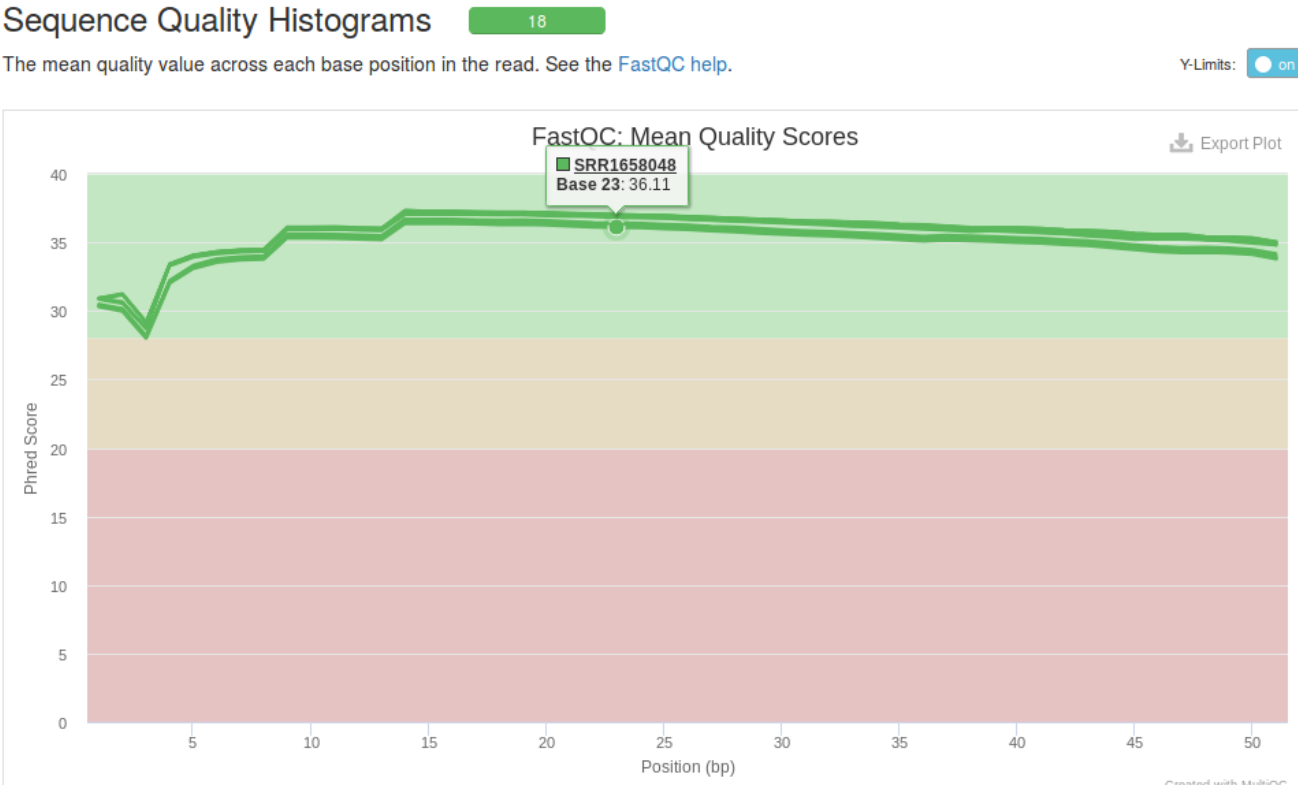
multiqc_data
multiqc_report.html
are generated.

Copy “multiqc_report.html” to local machine:

cd /home/nextgen2015/users/17232658/
cp /data4/nextgen2015/users/17232658/RNA_Seq/multiqc_report.html ./

Terminal 2:
scp nextgen2015@syd:/home/nextgen2015/users/17232658/multiqc_report.html ./

Output – Analysis of MultiQC data:



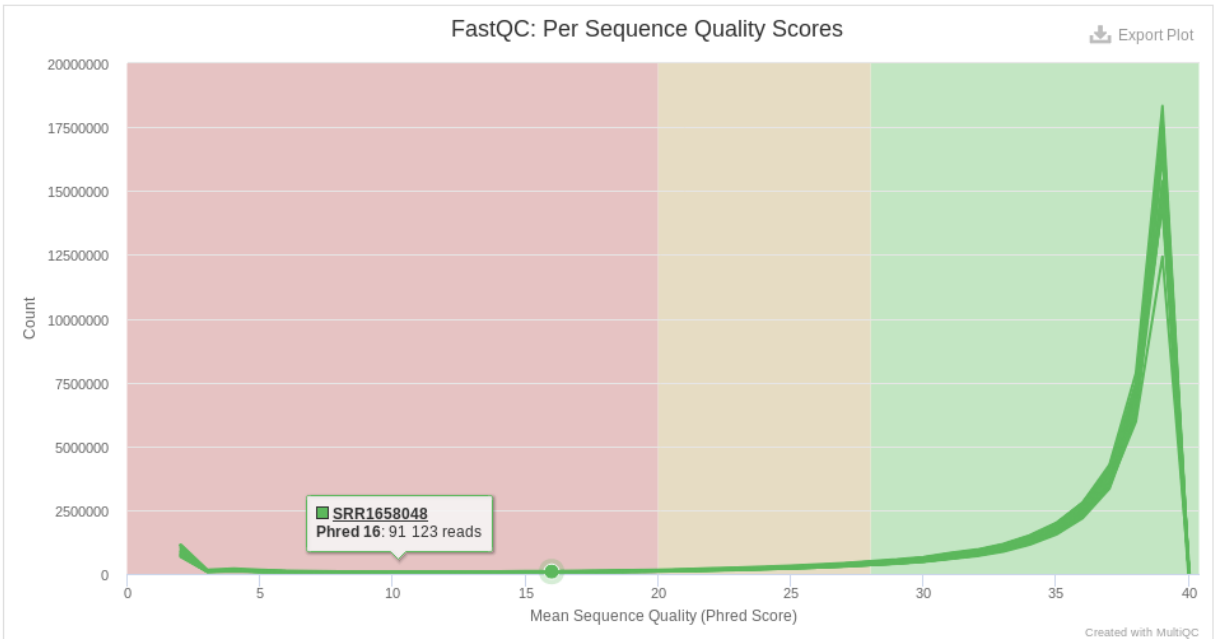
This “sequence quality histogram” exhibits the mean quality value across each base position in the read. It is apparent that all samples analysed have high mean quality scores. This is evident by the large Phred scores (occupying the “y-axis”), coupled with the fact the samples all exists within the high quality “green” zone. The “Phred scores” are a measure of the uality of the identification of the nucleobases generated by automated DNA sequencing. As is observed, the mean quality scores are between the 30 and 40 mark. A “Phred score of 40 is representative of a 1 in 10,000 probability of incorrect base call, or a base 99.99% base call accuracy, with 30 meaning a 1 in 1,000 probability of incorrect base call, or a base 99.9% base call accuracy . It is thererfore intuituve to think that these are good quality base calls, and can be utilised for downstream analysis.

Per Sequence Quality Scores

18

The number of reads with average quality scores. Shows if a subset of reads has poor quality. See the FastQC help.

Y-Limits: on



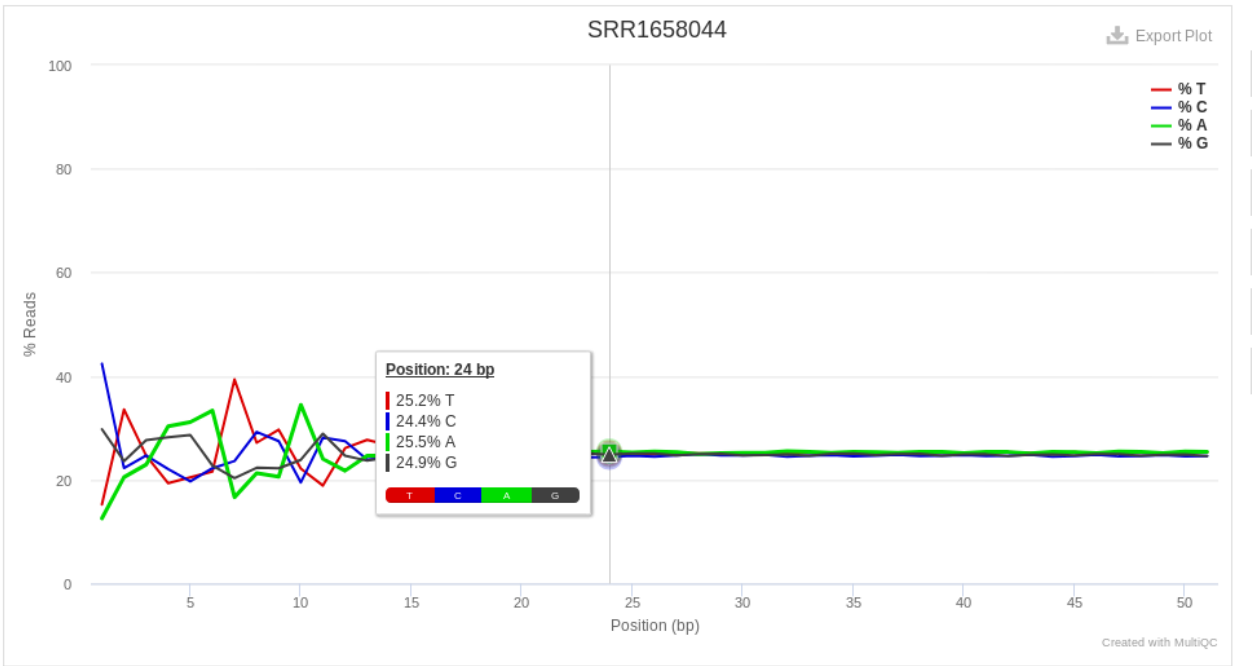
The per sequence quality score report allows you to see if a subset of your sequences have universally low quality values. It is evident therefore that all of the sample have universally high large “Phred” sequence scores. The preaks appear to assemble al in the “green” zone which correlates with large “Phred” scores, close to 40. In addition, the “peaks” within this green region is high, indicating that lare numbers of counts have large quality scores and therefore indicative of great sample quality. However, there are a few reads exisitng in the “red zone”, in the extreme left of the graph. However, these samples only have small read counts in this region, and therefore should not substantially impact results.

Per Base Sequence Content

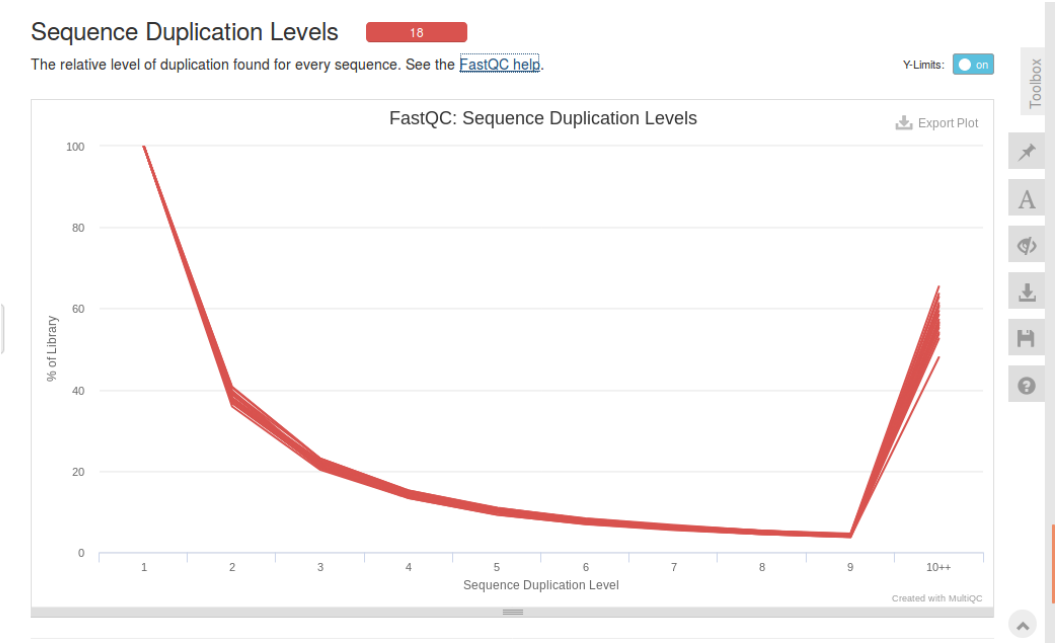
18

The proportion of each base position for which each of the four normal DNA bases has been called. See the FastQC help.

Back to overview heatmap | « Prev | Next »



Overall, the “per base sequence content” metric was failed, with all 18 samples failing this QC step. The above graph shows the “SRR1658044” which is the sample exhibitng the lowest peak in the “green zone” in the previous sample. The “Per Base Sequence Content” plots out the proportion of each base position in a file for which each of the four normal DNA bases has been called. In a random library you would expect that there would be little to no difference between the different bases of a sequence run, so the lines in this plot should run parallel with each other. This is not the case for all of our samples as the ratio of nucleotide’s called fluctuates substantially as evidenced by the above diagram. However, libraries produced by priming using random hexamers (including nearly all RNA-Seq libraries) and those which were fragmented using transposases inherit an intrinsic bias in the positions at which reads start. Therefore, this wouldn’t be a major concern and one would proceed with the analysis as ususal.



The above plot represents the relative level of duplication found for every sequence. None of the samples fared well with this QC analysis with all 18 samples faling the “Sequence Duplication Levels” QC. This would suggest some kind of enrichment bias with the data (eg PCR over amplification). However, in RNA-Seq libraries sequences from different transcripts will be present at wildly different levels in the starting population. In order to be able to observe lowly expressed transcripts it is therefore common to greatly over-sequence high expressed transcripts, and this would therefore explain the large set of duplicates. Therefore, I will proceed with the normal protocol and not remove duplicates.

Protocol Description:
Recently, a new software suite has been created to perform RNA_seq whilst running much faster, using substantially less memory and providing more accurate overall esults than previous protocols. This involves the sequential use of *Hisat*, *Stringtie* & *Ballgown:*

- *HISAT* aligns RNA-seq reads to a genome and discovers transcript splice sites, while running far faster than TopHat2 and requiring much less computer memory than other methods. The user can provide a file of annotated gene positions as an option, and HISAT will use that file however this is not the protocol I adhered to.

- *StringTie* assembles the alignments into full and partial transcripts, creating multiple isoforms as necessary and estimating the expression levels of all genes and transcripts. A key part of this protcol is to implement the "merge" function After assembling each sample, the full set of which merges together all the gene structures found in any of the samples. This is necessary as, after the initial *Stringtie* assembly, some of the samples might be only partially covered by reads. This ultimately creates a set of transcripts that is consistent across all samples. *Stringtie* then estimates the new ttranscript abundances using the merged structures aswell as additional read count information.

- *Ballgown* takes the transcripts and expression levels from *StringTie* and applies rigorous statistical methods to determine which transcripts are differentially expressed between experimental conditions, in our case between the "case" ("CRE") and "control" ("VEC")

Protocol:
Indexing is required to build a "model" for the mouse fasta genome sequence. I preoceeded to index mine as "mm9"

Indexing:
mkdir indexes
cd indexes
hisat2-2.1.0/hisat2-build -p 8 -f mm9.fa mm9

For the protocol I created a range of bash scripts and "qsubb'ed" these to ensure smooth running on the cluster.

Create a "Hisat" Bash Script:
```
# Your job name
#$ -N APCL

# The job should be placed into the queue 'all.q'
#$ -q all.q

# Running in the current directory
#$ -cwd

# Export some necessary environment variables
#$ -v PATH
#$ -v LD_LIBRARY_PATH
#$ -v PYTHONPATH
```

#$ -S /bin/bash

#Finally, put your command here
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658055.fastq -S SRR1658055.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658054.fastq -S SRR1658054.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658053.fastq -S SRR1658053.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658052.fastq -S SRR1658052.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658051.fastq -S SRR1658051.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658050.fastq -S SRR1658050.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658049.fastq -S SRR1658049.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658048.fastq -S SRR1658048.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658047.fastq -S SRR1658047.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658046.fastq -S SRR1658046.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658045.fastq -S SRR1658045.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658044.fastq -S SRR1658044.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658043.fastq -S SRR1658043.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658042.fastq -S SRR1658042.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658041.fastq -S SRR1658041.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9_tran -U SRR1658040.fastq -S SRR1658040.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9 -U SRR1658039.fastq -S SRR1658039.sam
hisat2-2.1.0/hisat2 -p 8 --dta -x indexes/mm9 -U SRR1658038.fastq -S SRR1658038.sam

Description:
-p = The number of threads that will run on separate processors/cores and synchronize when parsing reads and outputting alignments. The default is “1”, but we want to speed up the proccessing speed and therefore I used “8”.

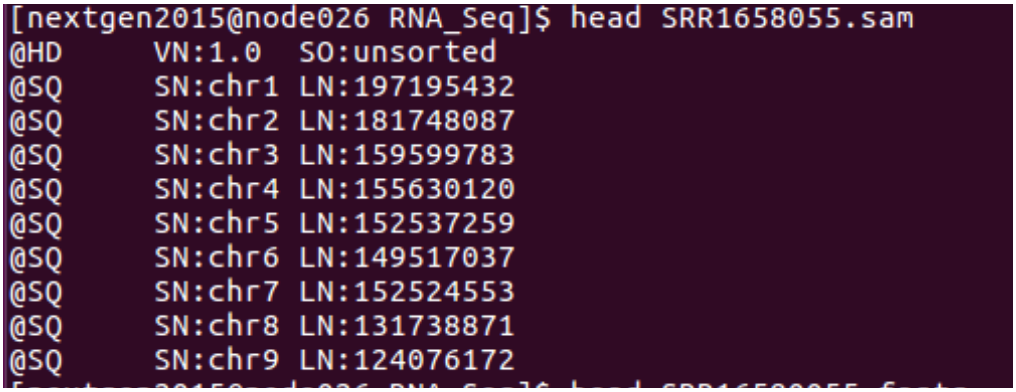
-dta = **Downstream Transcriptome Assembly**. This report alignments tailored for the downstream transcript assembling *Stringtie* package.

-U = The files that containunpaired reads to be aligned.

-S = The file to write SAM alignments to. “SAM” stands for “Sequence Alignment/Map” format. They are tab-delimited text format consisting of a header section, which is optional, and an alignment section.

Command:
qsub ./hisat_alignAP.sh

Performed “sanity check” on .sam file:



Create a “Samtools” Bash Script:

```
#!/bin/bash

# Your job name
#$ -N APsam

# The job should be placed into the queue 'all.q'
#$ -q all.q

# Running in the current directory
#$ -cwd

# Export some necessary environment variables
#$ -v PATH
#$ -v LD_LIBRARY_PATH
#$ -v PYTHONPATH
#$ -S /bin/bash

#Finally, put your command here

samtools view -bS SRR1658055.sam > SRR1658055_unsorted.bam
samtools view -bS SRR1658054.sam > SRR1658054_unsorted.bam
samtools view -bS SRR1658053.sam > SRR1658053_unsorted.bam
samtools view -bS SRR1658052.sam > SRR1658052_unsorted.bam
samtools view -bS SRR1658051.sam > SRR1658051_unsorted.bam
samtools view -bS SRR1658050.sam > SRR1658050_unsorted.bam
samtools view -bS SRR1658049.sam > SRR1658049_unsorted.bam
samtools view -bS SRR1658048.sam > SRR1658048_unsorted.bam
samtools view -bS SRR1658047.sam > SRR1658047_unsorted.bam
```

-bS = “SAM” file to be processed
Using this Samtools version, the files have to be converted in a 2 step process. The first sorts the “SAM” files and the second converts to binary or “BAM” format with the second step not visible in the bash script above.

Commands:

```
chmod a+x samtoolsAP.sh

module load samtools
./samtoolsAP.sh
```

Perform “sanity check” on .bam file:

This is the expected output of this binary “.bam” format and can be read by computers.

At this point I had the option to remove duplicates, however I didn’t utilise this oppertunity as in RNA-Seq experiments usually you observe lowly expressed transcripts and therefore it is common to greatly over-sequence high expressed transcripts to include these low abundant transcripts in our analysis.

I then proceeded to “Ensembl” and downloaded the “mm9.gtf” file.

Created a stringtie bashscript:

```
#!/bin/bash

# Your job name
#$ -N Mayo4Sam

# The job should be placed into the queue 'all.q'
#$ -q all.q

# Running in the current directory
#$ -cwd

# Export some necessary environment variables
#$ -v PATH
#$ -v LD_LIBRARY_PATH
#$ -v PYTHONPATH
#$ -S /bin/bash

#Finally, put your command here
stringtie-1.3.4c/stringtie -p 8 -G mm9.gtf -o  SRR1658055_mm9.gft -l  SRR1658055
SSR1658055_mm9.bam
stringtie-1.3.4c/stringtie -p 8 -G mm9.gtf -o  SRR1658054_mm9.gtf -l  SRR1658054
SRR1658054_mm9.bam
```

stringtie-1.3.4c/stringtie -p 8 -G mm9.gtf -o SRR1658053_mm9.gtf -l SRR1658053 SRR1658053_mm9.bam
stringtie-1.3.4c/stringtie -p 8 -G mm9.gtf -o SRR1658052_mm9.gtf -l SRR1658052 SRR1658052_mm9.bam
stringtie-1.3.4c/stringtie -p 8 -G mm9.gtf -o SRR1658051_mm9.gtf -l SRR1658051 SRR1658051_mm9.bam
stringtie-1.3.4c/stringtie -p 8 -G mm9.gtf -o SRR1658050_mm9.gtf -l SRR1658050 SRR1658050_mm9.bam
stringtie-1.3.4c/stringtie -p 8 -G mm9.gtf -o SRR1658049_mm9.gtf -l SRR1658049 SRR1658049_mm9.bam

Description:
-p = This argument has been described in the *Hisat* protocol.
-G = This uses the reference annotation file in the ".gtf" format, in our case the "mm9.gtf", to guide the assembly process.
-o = output file name for the merged transcripts GTF
-l = label or name prefix for output transcripts

Commands:
chmod a+x stringtieAP.sh

module load samtools
./stringtieAP.sh
rm *.sam *unsorted.bam

nano mergelist.text
SRR1658055_mm9.gtf
SRR1658054_mm9.gtf
SRR1658053_mm9.gtf
SRR1658052_mm9.gtf
SRR1658051_mm9.gtf
SRR1658050_mm9.gtf
SRR1658049_mm9.gtf
SRR1658048_mm9.gtf
SRR1658047_mm9.gtf
SRR1658046_mm9.gtf
SRR1658045_mm9.gtf
SRR1658044_mm9.gtf
SRR1658043_mm9.gtf
SRR1658042_mm9.gtf
SRR1658041_mm9.gtf
SRR1658040_mm9.gtf
SRR1658039_mm9.gtf
SRR1658038_mm9.gtf

This reason for "merging" was described above during the S*tringtie* "Protocol description":
stringtie-1.3.4c/stringtie --merge -p 8 -G mm9.gtf -o stringtie_merged.gtf mergelist.txt

This examines how the transcripts compare with the reference annotation:
gffcompare-0.10.4.Linux_x86_64/gffcompare -r mm9.gtf -G -o merged stringtie_merged.gtf

Description:
The -r = denotes annotation file to use as reference. In our case this is the “mm9.gtf” file.
-G = compares all transcripts in the input mm9.gtf file, even those that might be redundant
-- again this indicates the output file

Stringtie has to be completed again for reasons discussed in the “Protocol Discription”.

Created a second stringtie bashscript:

```
stringtie-1.3.4c/stringtie -e -B -p 8 -G ballgown/SRR1658037/stringtie_merged.gtf -o
SRR1658037_mm9.gtf SRR1658037_mm9_rmd.bam
stringtie-1.3.4c/stringtie -e -B -p 8 -G ballgown/SRR1658038/stringtie_merged.gtf -o
SRR1658038_mm9.gtf SRR1658038_mm9_rmd.bam
stringtie-1.3.4c/stringtie -e -B -p 8 -G ballgown/SRR1658039/stringtie_merged.gtf -o
SRR1658039_mm9.gtf SRR1658039_mm9_rmd.bam
stringtie-1.3.4c/stringtie -e -B -p 8 -G ballgown/SRR1658040/stringtie_merged.gtf -o
SRR1658040_mm9.gtf SRR1658040_mm9_rmd.bam
stringtie-1.3.4c/stringtie -e -B -p 8 -G ballgown/SRR1658041/stringtie_merged.gtf -o
SRR1658041_mm9.gtf SRR1658041_mm9_rmd.bam
stringtie-1.3.4c/stringtie -e -B -p 8 -G ballgown/SRR1658042/stringtie_merged.gtf -o
SRR1658042_mm9.gtf SRR1658042_mm9_rmd.bam
stringtie-1.3.4c/stringtie -e -B -p 8 -G ballgown/SRR1658043/stringtie_merged.gtf -o
SRR1658043_mm9.gtf SRR16580343_mm9_rmd.bam
stringtie-1.3.4c/stringtie -e -B -p 8 -G ballgown/SRR1658044/stringtie_merged.gtf -o
SRR1658044_mm9.gtf SRR1658044_mm9_rmd.bam
```

qsub ./stringtie2AP.sh

Description:
-e = This option is recommended for this run in order to produce more accurate abundance estimations of the input transcripts.
All the other arguments have been described above.

This creates a “ballgown” directory.

tar czf compressed.tar.gz ballgown

nano pheno.data

ID,phenotype,status
SRR1658038,VEC,U
SRR1658039,VEC,S
SRR1658040,VEC,F
SRR1658041,CRE,U
SRR1658042,CRE,S
SRR1658043,CRE,F
SRR1658044,VEC,U
SRR1658045,VEC,S
SRR1658046,VEC,F
SRR1658047,CRE,U
SRR1658048,CRE,S
SRR1658049,CRE,F
SRR1658050,VEC,U
SRR1658051,VEC,S
SRR1658052,VEC,F
SRR1658053,CRE,U
SRR1658054,CRE,S
SRR1658055,CRE,F

I procedeed to copy the “ballgown” directory and “pheno.csv” to my local machine and ran ballgown using “**R studio**”.

cd /home/nextgen2015/users/17232658/
cp -r /data4/nextgen2015/users/17232658/RNA_Seq/compressed.tar.gz ./
cp /data4/nextgen2015/users/17232658/RNA_Seq/*pheno*./

Terminal 2:
scp -r nextgen2015@syd:/home/nextgen2015/users/17232658/compressed.tar.gz ./
scp nextgen2015@syd:/home/nextgen2015/users/17232658/*pheno* ./

Ballgown:
```{r}
library(ballgown)
library(genefilter)
library(dplyr)
```

This loads in the relavant libraries that we will be working with.

```{r}
pheno_data = read.csv("pheno.csv")
bg_mm9 = ballgown(dataDir = "/home/user9/Desktop/RNA-Seq/ballgown", samplePattern = "SRR", pData=pheno_data)
bg_mm9_filt = subset(bg_mm9,"rowVars(texpr(bg_mm9)) >1",genomesubset=TRUE)
```

This code reads in the “pheno.csv” file that contains information about your RNA-seq samples, containing “ID”, “phenotype” and “status” columns respectively as illustrated above. The second

line of code reads in the expression data that was calculated by StringTie and couples this with the phenotypic data just read in. The data directory has to be specified (“dataDir”) and a pattern observed in each sample (samplePattern = “SRR”). The third step filters out the low-abundance genes, with all transcripts H with a variance across samples less than one filtered out.

```{r}
results_transcripts = stattest(bg_mm9_filt, feature="transcript", covariate="phenotype",adjustvars = c("status"),getFC=TRUE, meas="FPKM")

results_genes = stattest(bg_mm9_filt, feature="gene", covariate = "phenotype", adjustvars = c("status"), getFC=TRUE, meas="FPKM")

results_transcripts = data.frame(geneNames=ballgown::geneNames(bg_mm9_filt), geneIDs=ballgown::geneIDs(bg_mm9_filt), results_transcripts)

results_transcripts = arrange(results_transcripts,pval)
results_genes = arrange(results_genes,pval)
```

The first line of code accounts for transcripts that are differentially expressed between the phenotypes (VEC & CRE), while correcting for any differences in expression due to the status (U S, F) variable. The second line identifies genes that show statistically significant differences between groups. The third code line adds the gene names and gene IDs to the results_transcripts data frame previously created. The final two line’s of code sort the transcript and gene dataframes by ordering them from the smallest p-value to the largest.

```{r}
write.csv(results_transcripts, "mm9_transcript_results.csv", row.names=FALSE)
subset(results_transcripts,results_transcripts$qval<0.05)

write.csv(results_genes, "mm9_gene_results.csv", row.names=FALSE)
subset(results_genes,results_genes$qval<0.05)

```
The first half of this code block writes the data generated earlier to a “.csv” file and identifies transcripts with a qvalue <0.05, and the second half performs the same operation with genes.

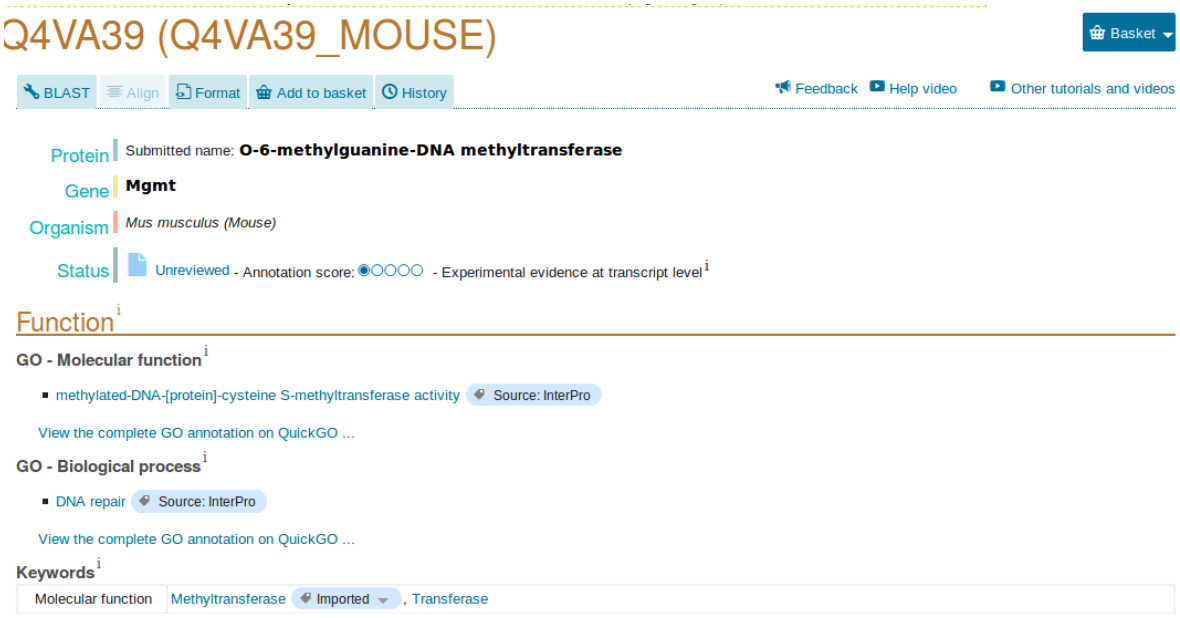
Output:

```
write.csv(results_transcripts, "mm9_transcript_results.csv", row.names=FALSE)
subset(results_transcripts,results_transcripts$qval<0.05)
```

```
##           geneNames     geneIDs    feature    id          fc         pval
## 1          Serpine2   MSTRG.411 transcript  1116  0.07458043 0.000000e+00
## 2              Perp  MSTRG.1088 transcript  3127  0.11154260 0.000000e+00
## 3                 .  MSTRG.2600 transcript  7831  4.40667914 0.000000e+00
## 4                 .  MSTRG.3786 transcript 11319  0.31593078 0.000000e+00
## 5     2210016F16Rik  MSTRG.4077 transcript 12133  0.12036769 0.000000e+00
## 6              Irx1  MSTRG.4197 transcript 12444  6.83492760 0.000000e+00
## 7              Gpc6  MSTRG.5013 transcript 14926  0.17654381 0.000000e+00
## 8            Man2a1  MSTRG.6899 transcript 21010  1.66059988 0.000000e+00
## 9            Zfp345  MSTRG.9110 transcript 27909  3.26681430 0.000000e+00
## 10         Gm13242 MSTRG.11158 transcript 34124  4.17155425 0.000000e+00
## 11           Igfbp7 MSTRG.11714 transcript 35705  0.01740597 0.000000e+00
## 12           Antxr2 MSTRG.11783 transcript 35950  1.84659395 0.000000e+00
## 13            Hspb8 MSTRG.11975 transcript 36532  2.15469996 0.000000e+00
## 14              Ptn MSTRG.12492 transcript 38087  0.16205587 0.000000e+00
## 15             Dkk3 MSTRG.14028 transcript 43391 10.41183141 0.000000e+00
## 16             Mgmt MSTRG.14259 transcript 44036 19.12326880 0.000000e+00
## 17                . MSTRG.16378 transcript 50726 17.79827806 0.000000e+00
```

```
##                qval
## 1     0.000000e+00
## 2     0.000000e+00
## 3     0.000000e+00
## 4     0.000000e+00
## 5     0.000000e+00
## 6     0.000000e+00
## 7     0.000000e+00
## 8     0.000000e+00
## 9     0.000000e+00
## 10    0.000000e+00
## 11    0.000000e+00
## 12    0.000000e+00
## 13    0.000000e+00
## 14    0.000000e+00
## 15    0.000000e+00
## 16    0.000000e+00
## 17    0.000000e+00
## 18    0.000000e+00
```

The above is the printed data for the "transcripts" with sorted smallest p-values correlating to the difference in transcript expression between the samples. As you can see from above, all 17 transcripts above have a "p-value" of 0, and therefore in a biological setting (alpha signifcance threshold= 0.05) would be comprehensively signifiant and consequently undergo further downstream analysis. All the columns are intuitive, bar perhaps the "fc" parameter which correlates to "fold change". The largest "fc" transcript within the above screengrab corresponds to the "mgmt" gene. The right image dispalays the "q-value" associated with each of these transcripts, with a "q-value" of "0" evident for each transcript displayed. The "Q-values" are the name given to the adjusted p-values found using an optimised false discovery rae approach, and can therefore be considered strong estimates for rightful p-values. I proceeded to "Uniprot" to elucidate the biological function of the mouse "mgmt" gene:

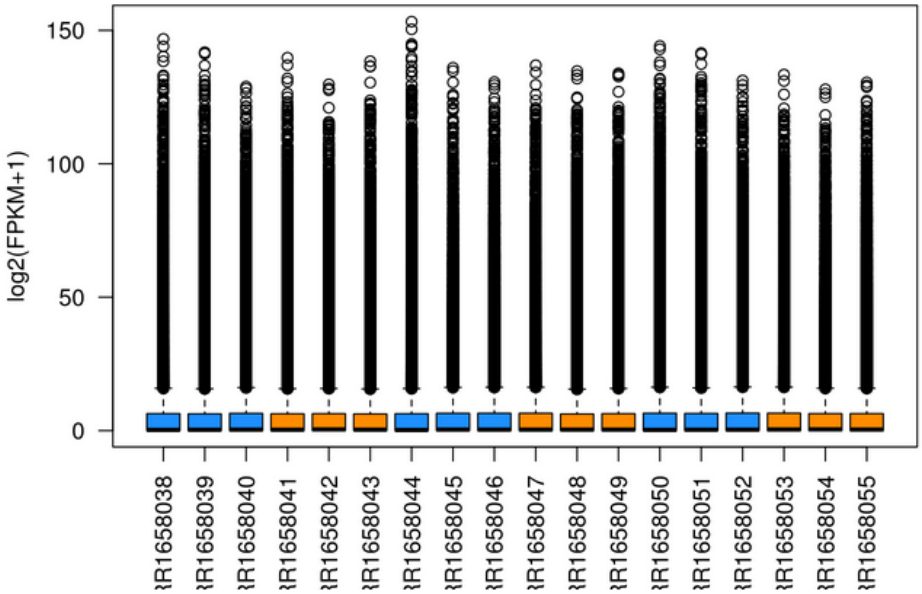


The "mgmt" transcript is deroved from the "O-6-methylguanine-DNA methyltransferase" gene and serves to methylate DNA and repair DNA (above). Therefore, this gene has paramount importace in terms of the genetic remodelling and epigenetic mechanisms that transform chromatin to "active" and "closed" states. This notion is consistent with the paper's findings with "*sprouty* gene deletion remodels histone modifications associated with active typical and super enhancers." Potentially this altering of epigentic state could adopt a resident tumour methylation profile as "epigenetic deregulation are root causes of tumorigenesis".

Data Visulaisation:

```{r}
tropical= c('darkorange', 'dodgerblue', 'hotpink', 'limegreen', 'yellow')

palette(tropical)

fpkm = texpr(bg_mm9,meas="FPKM")

boxplot(fpkm,col=as.numeric(pheno_data$phenotype),las=2,ylab='log2(FPKM+1)')
```

The initial line helps to make the plots look more colourful in the figures generated. Show the distribution of gene abundances (measured as FPKM values) across samples, colored by phenotype. The plot compares the FPKM measurements for the transcripts, after a log2 transformation which makes the data easier to visualise.



The “blue” samples are the empty vector or “control” group with the subgroups “U”, “S” & “F” respectively (left to right) as described earlier. The “orange” samples are the *Spr* knockout groups, with the subgroups “U”, “S” & “F” respectively. It is apparent that the gene abundances for the control group are larger than the “Cre” *Spry* deleted group. The relative abundances suggest a greater difference that can be attributed to just the knockout’s of the *Spry* genes alone. Therefore, this would suggest, that the *Spry* genes are perhaps key regulators of an array of key biological pathways, and without this key regulation, many genes are left dysregualted. This notion is consitent with *Spry2* biological function gene ontology in the *Uniprot* database, whcih suggests it’s vast role in a biological function regulation capacity:

GO - Biological process [i]

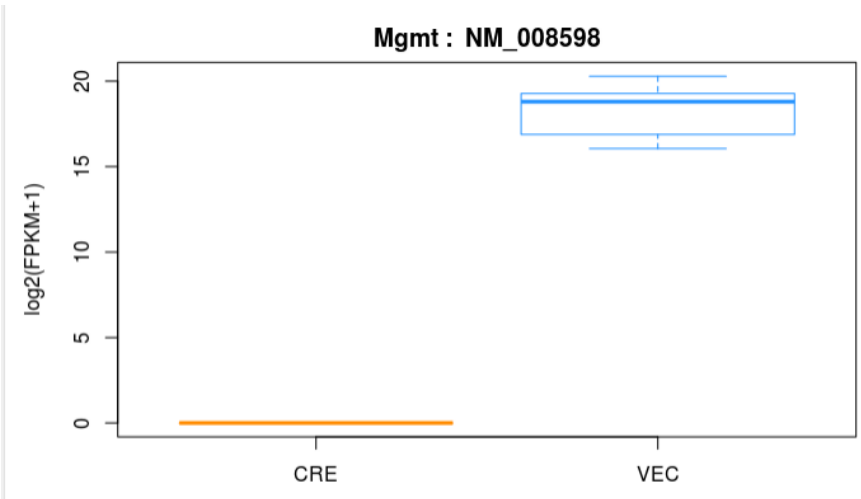
- branching morphogenesis of an epithelial tube — Source: MGI
- bud elongation involved in lung branching — Source: MGI
- cell fate commitment — Source: MGI
- cellular response to leukemia inhibitory factor — Source: MGI
- establishment of mitotic spindle orientation — Source: MGI
- inner ear morphogenesis — Source: MGI
- lung development — Source: MGI
- lung growth — Source: MGI
- lung morphogenesis — Source: MGI
- negative regulation of apoptotic process — Source: UniProtKB
- negative regulation of cell projection organization — Source: BHF-UCL
- negative regulation of cell proliferation — Source: MGI
- negative regulation of ERK1 and ERK2 cascade — Source: BHF-UCL
- negative regulation of fibroblast growth factor receptor signaling pathway — Source: MGI
- negative regulation of GTPase activity — Source: MGI
- negative regulation of MAP kinase activity — Source: MGI
- negative regulation of neurotrophin TRK receptor signaling pathway — Source: MGI
- negative regulation of peptidyl-threonine phosphorylation — Source: MGI
- negative regulation of Ras protein signal transduction — Source: MGI
- positive regulation of cell migration — Source: MGI
- positive regulation of ERK1 and ERK2 cascade — Source: MGI
- positive regulation of gene expression — Source: MGI
- positive regulation of peptidyl-serine phosphorylation — Source: MGI
- positive regulation of protein kinase B signaling — Source: MGI
- regulation of cell differentiation — Source: InterPro
- respiratory system development — Source: MGI
- sensory perception of sound — Source: MGI

`

Central to this gene's biological process ontology is "positive regulation" of gene expression, which is consistent with the above graph that deletion would lead to least transcriptional abundance.

```{r}
ballgown::transcriptNames(bg_mm9)[44036]

ballgown::geneNames(bg_mm9)[44036]

plot(fpkm[44036,] ~ pheno_data$phenotype, border=c(1,2),
main=paste(ballgown::geneNames(bg_mm9)[44036],': ', ballgown::transcriptNames(bg_mm9)
[44036],pch=19, xlab="Sex", ylab='log2(FPKM+1)')
```

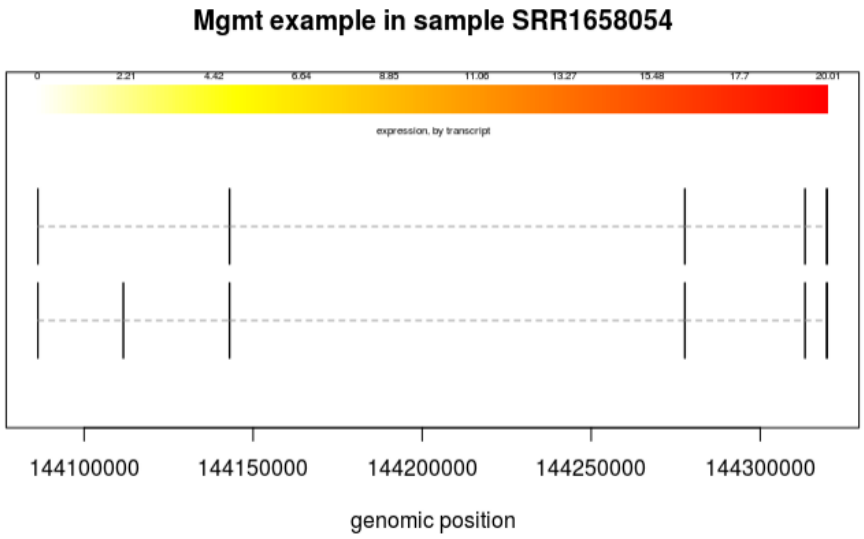
The above code produces a plot for one particular trancsript across both phenotype samples, which in the above cse is transcript 44,036. The first two commands display the name of the transcript (NM_008598) and the name of the gene that contains it (Mgmt). This gene was selected based on the large fold change (fc values) as obtained above.



It is evident from the above diagram that there are vast differences in the abundances in the *Mgmt* gene between the control (“VEC”) and knockout (“CRE”) phenotype groups. The “CRE” cohort have abundances approximating 0 log2 fold change, whereas the “VEC” cohort approximate a log2 fold change of 20. It is therefore apparnet, that in the absence of the *Spry* gene (*Spry* knockout), there is a dramatic reduction in *Mgmt* abundances. As was previously discussed, this gene’s product is paramount in terms of epigenetic maintenance and as it methylates regions of the genome. Therefore, when this gene is in low abundances, perhaps vast regions of the genome are left unmethylated,, leading to more accessible chromatin states, more transcription factor binding and subsequently enable cancer driver and dysregulatory genes to propogate more easily.

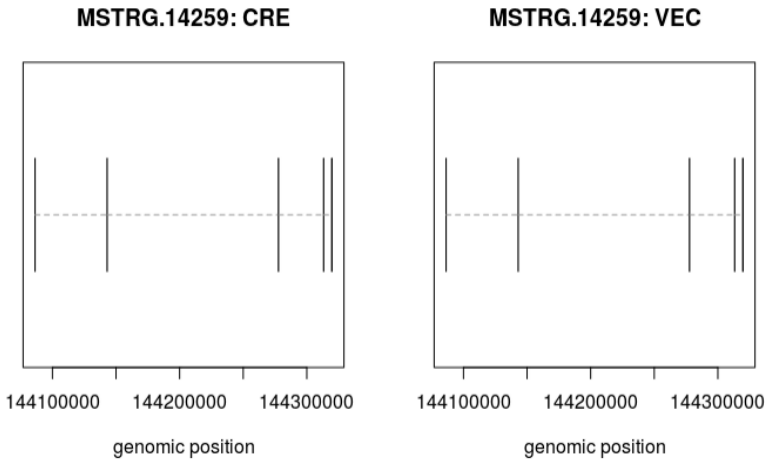
```{r}
plotTranscripts(ballgown::geneIDs(bg_mm9)[44036], bg_mm9, main=c('Gene example in sample SRR1658054), sample=c('SRR1658054'))

plotMeans('MSTRG.56', bg_mm9_filt,groupvar="phenotype",legend=FALSE)
```

The above code plots the structure and expression levels in a sample of all transcripts that share the same gene locus. In our case I choose the “44036”th transcript, as was used above that corresponded to the *mgmt* gene. The “SRR1658054” sample was selected as it corresponded with the lowest expression profile “peak” in the logfold plot as generated above.



Above is the plot formed regarding structure and expression levels of two distinct isoforms of the *mgmt* gene in sample SRR1658054. Expression levels are shown in varying shades of yellow. However, presumably due to the much lower abundance, this sample contains no varying shades of yellow and hence both isoforms appear to be unifirmly not expressed in this sample.

Two plots can be generated to compare the average expression levels for all transcripts of a gene across the two sample groups, in our case the control (“VEC”) and *Spr* deleted (“CRE”) groups. Again I analysed the *mgmt* gene:

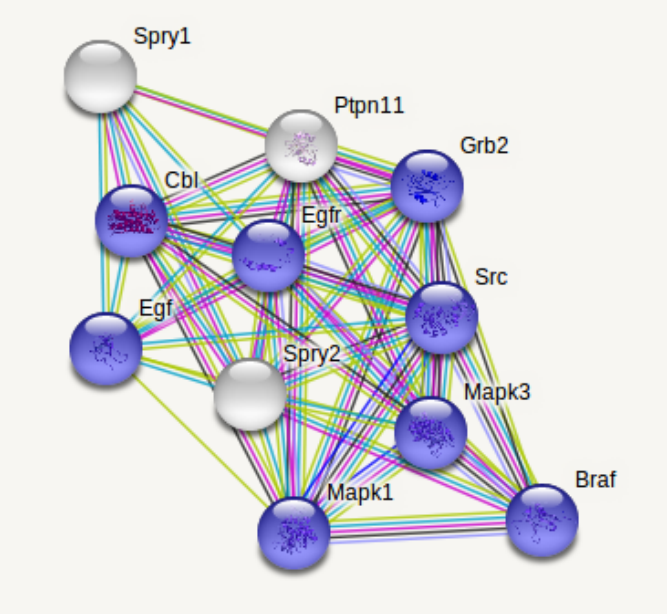


The above images are contradictory, as one would expect greater gene expression levels in the "VEC" group when compared to the "CRE" group for the *mgmt* gene as it's fold change values corresponds to approximately "20". Therefore, perhaps the *mgmt* expression levels are extremely small in the control groups however this could be the optimum abundance required to perform their normal function.

Conclusions:

Overall, there appears to n=be many differntially expressed gene's between the "control" and *Spry* knockout groups. This is evidenced by the transcript "csv" file, which unloaded all the differentially expressed transcripts in oreder of smallest p-value. Of the these transcripts that obtained minimal p-values and associated q-values (after false discovery rate correction), the looked further into the *Mus musculus* gene *mgmt* which had evidently a large fold change in gene expression levels. This gene is involved in DNA methylation and therefore paramount in the epigentic methylome of an individual. This significant gene is consistent with the papers that epigenetic deregulation results from defective receptor tyrosine signalling, and perhaps these samples should be investigated further to elucidate a strong causal link as to how "epigenetic deregulation are root causes of tumorigenesis". Potentially, dysregulated *Spry* gene's in cancer patients (i.e fuse an activtor domain using *CRISPR Cas9* technology) to improve *Spry* abundances to help control the disease. After all, the transcriptional profile is substantially altered in the *Spry* knockout, indicated by the transcriptional abundance plot above, whereby samples with *Spry* deletion have less gene abundances than their wild-type counterparts. This is consistent with the act that *Spry* is a major positive regulator of gene expression from the uniprot databse. In an attempt to look at *Spry*'s interactions with other proteins and help confirm a causal link with cancer I proceed to the *STRING* databse and searched for *Spry2* in the *Mus musculus* genome. This database returned a set of protein's the translated protein interacted with and I looked at the *KEGG* pathways and inform me of the pivotal processes this gene is involved in:

KEGG Pathways			
pathway ID	*pathway description*	*count in gene set*	*false discovery rate*
04012	ErbB signaling pathway	8	1.17e-15
05205	Proteoglycans in cancer	8	1.53e-12
05213	Endometrial cancer	6	4.15e-12
05223	Non-small cell lung cancer	6	4.46e-12
05214	Glioma	6	6.22e-12
			(more ...)



The above image is a table of all the *KEGG* interaction pathways that the gene is involved in. The second gives a visual representation of *Spry's* interaction partners. All the *KEGG* pathways present are invloved in cancer progression, and the "blue" circles above are all involved in the *ErbB* signalling pathway. The **ErbB** family of proteins contains four receptor trosine kinase proteins (with RTK pathways abberant in *Spry* deletion) with *HER2* (involved in breast cancer suceptibility) a member of this family. It is therefore evident that *Spry2* is a major gene regualtor, involved in a range of cancer pathways, and with tgreater research potentially unlocking therapeutic strategies to target this gene is cancer treatment sprotocols.