RNA-Seq: Analysis of the transcriptional landscape in a knock out parasite

A. Introduction

In this module we are going to learn about RNA sequencing ("RNA-Seq" - Mortazavi et al., 2008; Wang et al, 2009) using Illumina sequencing. The application today will be to compare a WT type *Plasmodium berghei* RNA-seq dataset with an RNA-seq data set from a mutant that had a transcription factor (Api AP2) gene knocked out. The goal of the exercise would be to determine the function of the gene that was knocked out.

During the exercise you will be introduced to the genome viewer "Artemis" and how to visualize RNA-Seq reads. Next we are going to compare the expression of the genes with the aim to find differentially expressed genes. Those will be analysed in the PlasmoDB database. Last we should discuss the role of biological replicates and which tools could be used to perform differential expression.

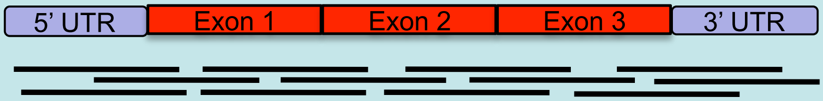
RNA-Seq

Transcriptome sequencing is a very useful addition to genome sequencing projects as it helps to identify genes and thus aids in genome annotation. In this sense it is similar to earlier transcriptome sequencing using capillary methods (EST sequencing), but provides much higher coverage of the transcriptome.

Sequence reads from RNA sequencing can be treated in much the same way as those from DNA sequencing. The exception is the occurrence of splicing, where intronic sequences are missing from RNA-seq reads. In this module we will use a similar approach used to map DNA sequencing data to map RNA sequencing data from *Plasmodium berghei*.

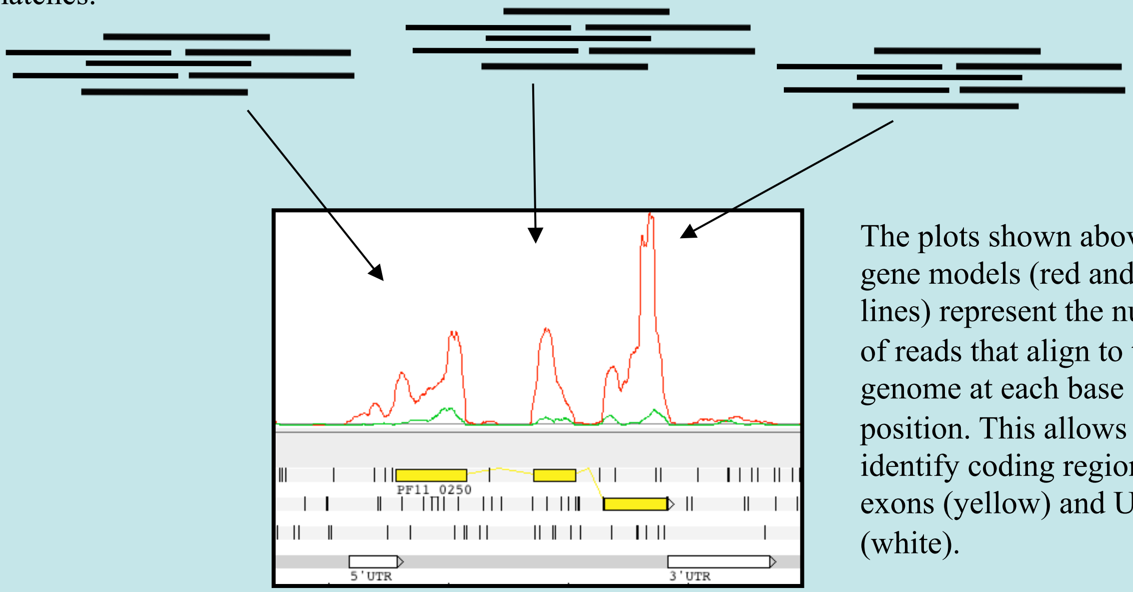
Due to the vast number of reads produced by next-gen sequencing technology, the transcriptome is also sequenced very deeply. Each gene is sequenced in proportion to its abundance and the large number of reads means that even low abundance genes are sequenced to some extent. This means that expression levels of genes can be compared. One can visualize the "pile up" of reads in a particular region by looking at coverage plots. The higher the plot, the more expressed a transcript is. For the purpose of the following exercises, remember that the sequences originate from transcriptome sample (mRNA) and therefore only contains information about the exons and UTRs.

In a more visual way … imagine this transcript is present in the sample



Reads belonging to the transcript are produced by the sequencing process.

When the reads come out as raw data, there is no information about where they belong on the reference genome. What is more, all reads from several different transcripts come out together. An alignment algorithm finds where they belong in the reference genome based on similarity matches.



The plots shown above the gene models (red and green lines) represent the number of reads that align to the genome at each base position. This allows us to identify coding regions: exons (yellow) and UTRs (white).

The first RNA-Seq study in *Plasmodium* parasites focused on *P. falciparum* (Otto et. al. 2010). The aim was to show the viability of the RNA-Seq protocol in comparison to microarrays and also to improve the genome annotation and find alternative splicing. Recently a group used RNA-seq to identify differentially expressed genes, showing that parasites from vector transmitted infections are less virulence than serially blood passaged in the laboratory (Spence et al. 2013). There will be a many more to come…

Exercise

All data you will need for this exercise are available online. So you could repeat (or finish) the exercises later at home.

In the appendix are all the commands used that you would need to replicate the analysis. This includes mapping and the differential expression. Alternatively you could also try webpages like http://pathogenportal.org.

A. Mapping with Tophat

First we will map RNA sequence reads from the WT parasite of *Plasmodium berghei* to the chromosome 14 sequence of the same strain.

In the directory of the module you can find the *Plasmodium* chromosome 14 reference sequence (berg14.fasta) as well as the two files of RNA sequence reads of the WT: `Pb_WT1.bam_1.fastq.gz` and `Pb_WT1.bam_2.fastq.gz`.

To work with the command line of Linux you will first need to open a terminal. Then go to the Module’ s directory:

```
$ cd ~/Module_6_RNA-Seq
```

For the mapping, first an index of the reference (here chromosome 14 of *P. berghei*) must be constructed with bowtie-build. On the command line, you should type:

```
$ bowtie2-build berg14.fa berg14.fa
```

This will generate the index need for bowtie. Most of the output you can ignore. Tophat first maps the un-spliced reads with bowtie, mapping the reads falling within exon boundaries. The non-mapping reads will be than split by Tophat. To start the command you should type:

```
$ ln -s berg14.fa berg14.fa.fa
$ tophat2 -o WT1 -I 2000 -r 150 -g 1 berg14.fa
Pb_WT1.bam.Chr14_1.fastq.gz Pb_WT1.bam.Chr14_2.fastq.gz
```

The mapping result will be written into the directory WT1/. If you have doubts about parameters of the program, type:

```
$ tophat2
```

What would the –g parameter do? Does it seem an important option?

Next you need to index the bam file

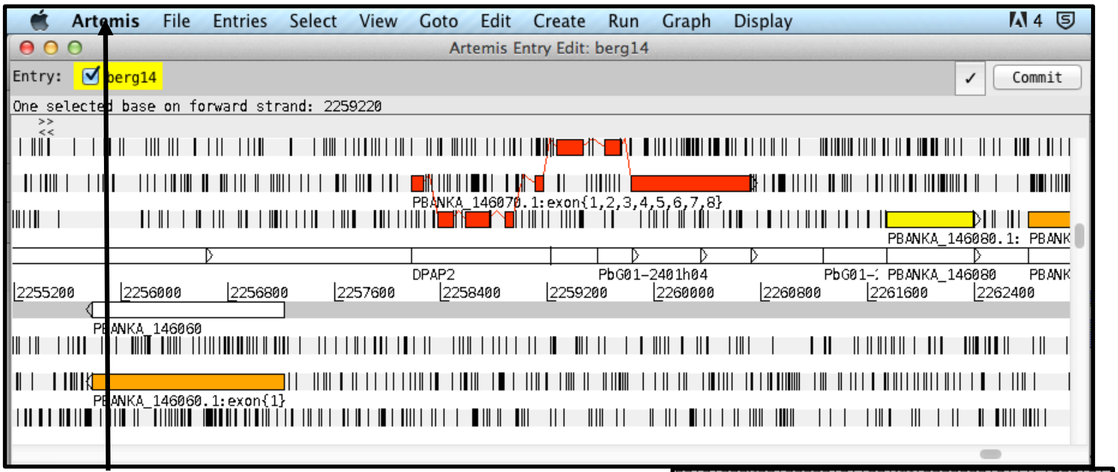
```
$ samtools index WT1/accepted_hits.bam
```

B. Viewing the mapped reads in Artemis

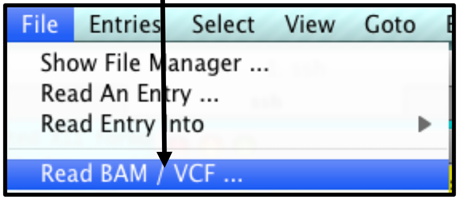
We will now examine the read mapping in Artemis using the BAM view feature.
Be sure to be in the same directory as before. Open Artemis and load `berg14.embl`. This contains exactly the same sequence as `berg14.embl`, but also has genome annotation so we can see the gene models.

```
$ art berg14.embl &        ### to open Artemis
```

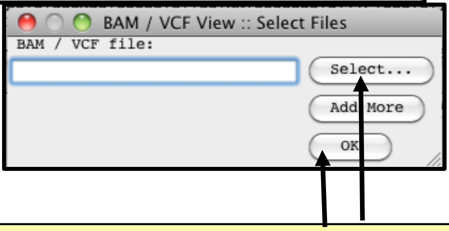
First go to the position 2259160 (Goto -> navigator).



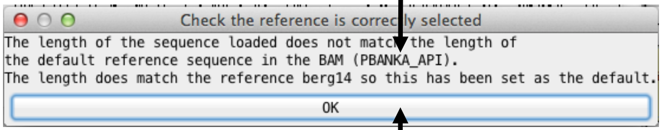
1. Now click on File -> “Reads BAM/ VCF…”



2. Select here the bam file from the WT1 directory you just genereated and then press ok.

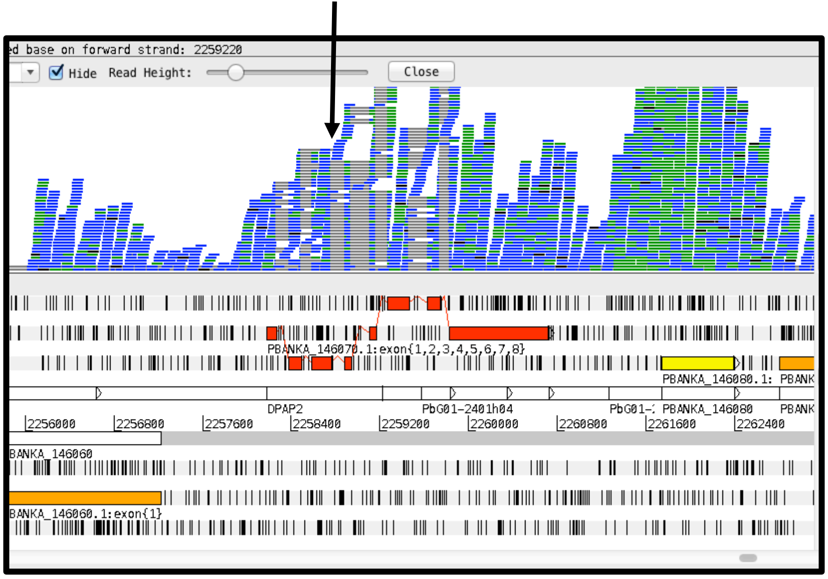


3. Confirm that the correct chromosome is chosen.


Check the reference is correctly selected

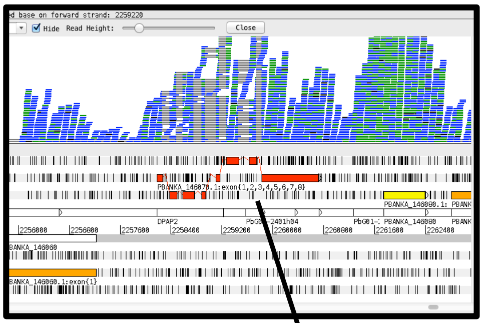
The length of the sequence loaded does not match the length of
the default reference sequence in the BAM (PBANKA_API).
The length does match the reference berg14 so this has been set as the default.

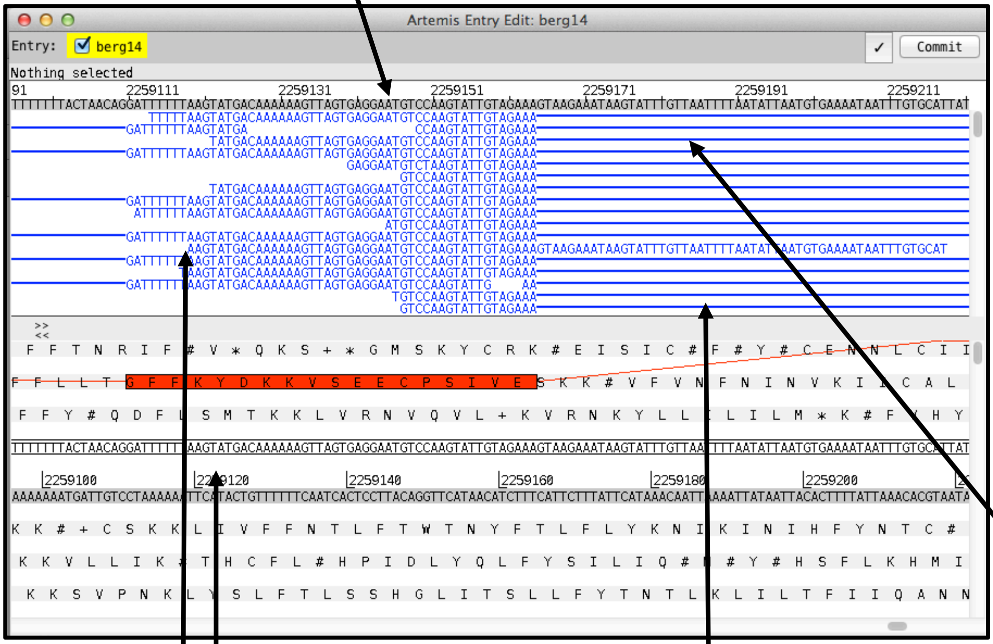
4. You should see following window… any idea what it means?



Congratulations, you have opened a Malaria chromosome with RNA-Seq mapping on it! The horizontal blue/green lines are sequencing reads, mapped against the reference. Let’s have a look how the reads are “mapped” against the reference.

1. Zoom in as much as you can



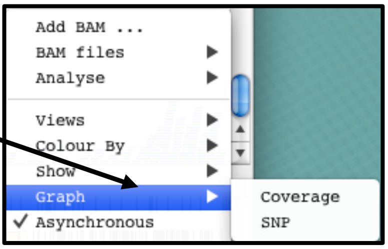


Each sequence represents a read. It is very similar to the genomic sequence at this regions, and therefore was mapped at this position. The abundance of reads represents the amount of mRNA of this gene.

Those reads are mapped over a splice site. The bar shows the intronic regions, which should be skipped in the reads. Can you see where the other parts of the reads are mapping?

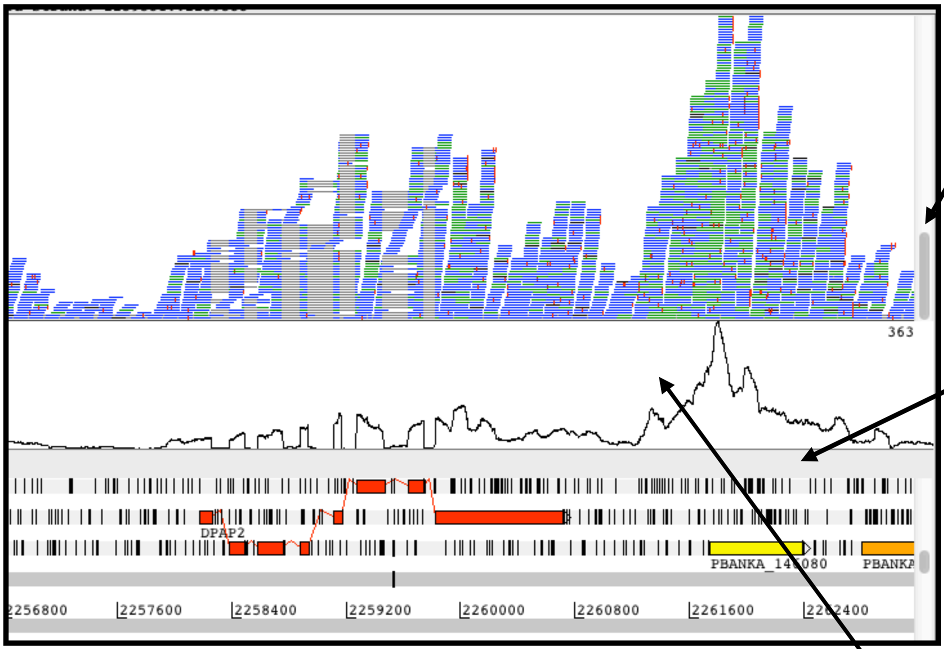
This is the so-called one-base pair resolution of RNA-Seq!

Right click in “BAM view,, select Graph -> Coverage. Then zoom out again



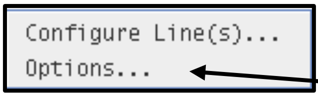
C. Interpreting the mapping

Zoom out until you have the same view as below:

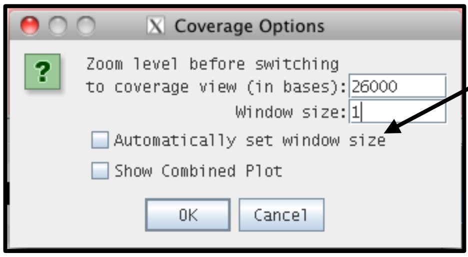


You can move the reads up and down, on the right scroll bar.

You can increase the size of the bam view, by dragging down with the mouse.



To better see the splice sites, do right click. Select “Options…”:
Set the window size to 1 (before unselect “Automatic…”)



Zoom level before switching to coverage view (in bases): 26000
Window size: 1
□ Automatically set window size
□ Show Combined Plot

Please discuss following aspects with your neighbour:

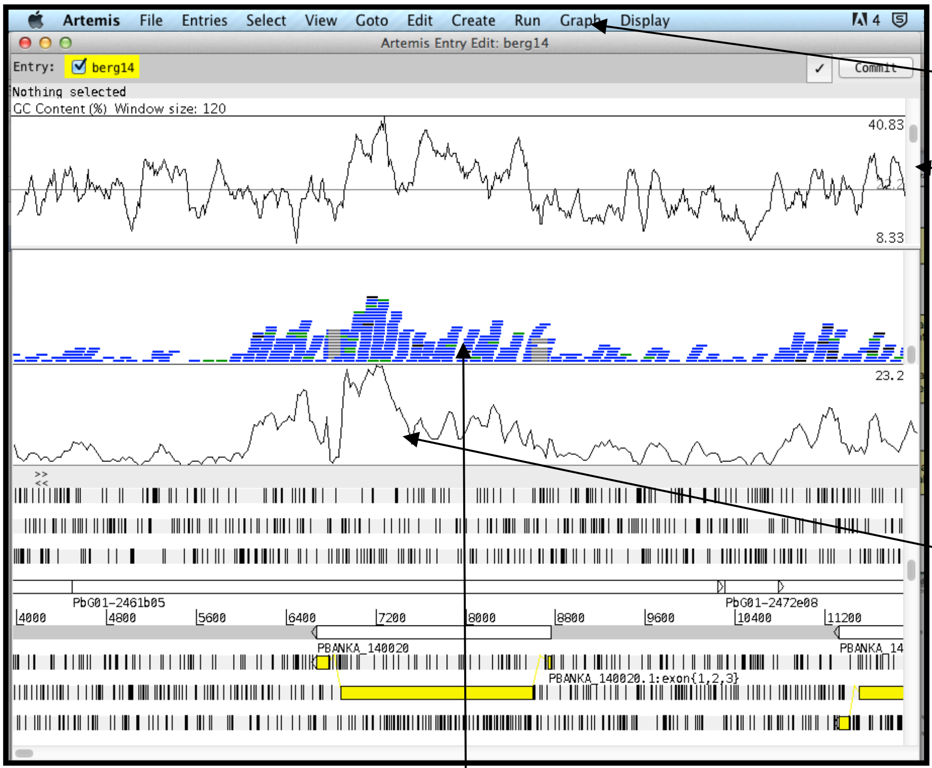
The coverage represents the amount of reads mapped over each position. Why are reads mapped where no exons are? Can you distinguish transcription starts and stops of genes?

Notice that different genes have different depths of coverage. What does this means?

Scroll through the genome and look at half a dozen genes, also some longer ones.
Why do some genes have less coverage? Have some genes no reads mapped to them? Is the coverage very even over the genes?

D. Uniqueness and GC content

Go to the position 8000 (Goto -> navigator).


Artemis File Entries Select View Goto Edit Create Run Graph Display
Artemis Entry Edit: berg14
Entry: berg14
Commit
Nothing selected
GC Content (%) Window size: 120
40.83
8.33
23.2
PbG01-2461b05
PbG01-2472e08
4000
4800
5600
6400
7200
8000
8800
9600
10400
11200
PBANKA_140020
PBANKA_14
PBANKA_140020.1:exon{1,2,3}

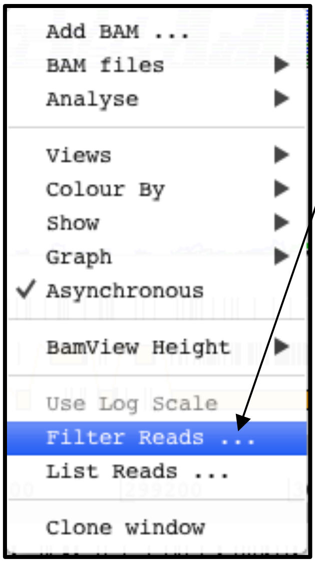
1. Enable the GC content, Graph -> GC Content.

2. Change the window size

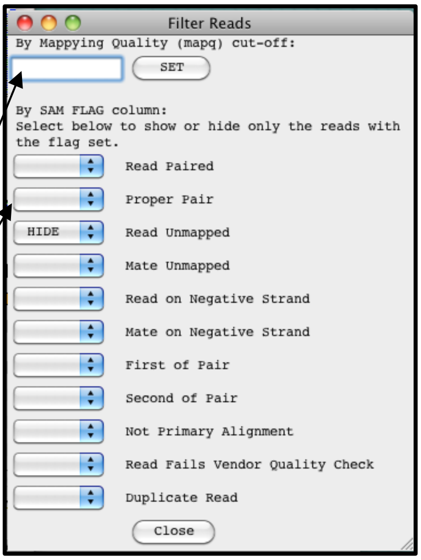
3. What are those peaks? Is there a correlation to the GC content?

4. You can filter reads by mapping quality and if they are mapped as proper mate pairs.

5. Right click, than Filter Reads…



6. Set the mapping quality to 10 and show proper pairs. What happens?


Filter Reads

By Mapping Quality (mapq) cut-off:

SET

By SAM FLAG column:
Select below to show or hide only the reads with the flag set.

	Read Paired
	Proper Pair
HIDE	Read Unmapped
	Mate Unmapped
	Read on Negative Strand
	Mate on Negative Strand
	First of Pair
	Second of Pair
	Not Primary Alignment
	Read Fails Vendor Quality Check
	Duplicate Read

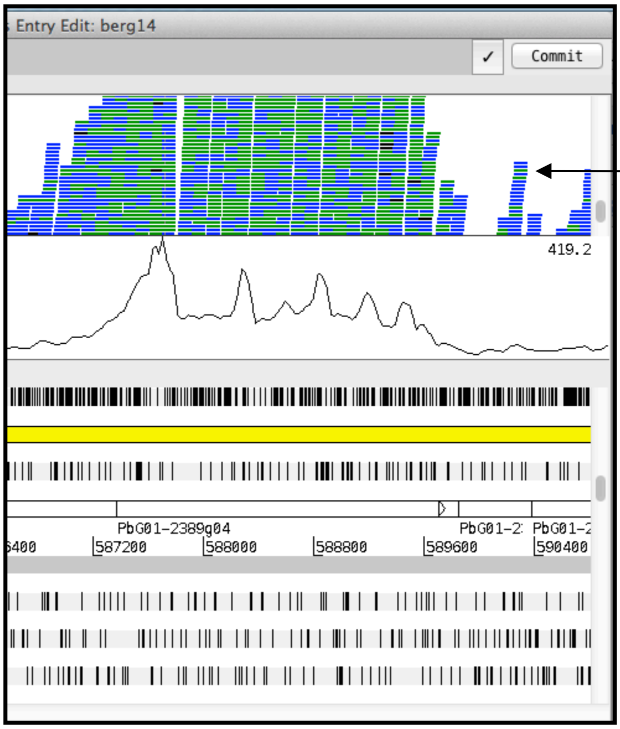
Close

Variation in coverage can have many reasons, one is GC content. Also important, reads can be placed more than once, when they are mapped repetitively. More conservative mapping is to just look at proper pairs, and ignore reads with a mapping quality score below 10.

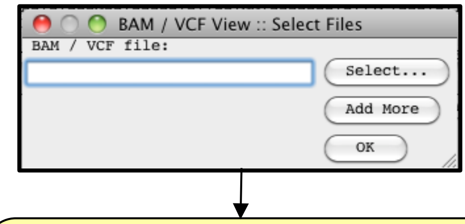
E. Including the mutant data set

Next we want to include the mutant (knock out) data set.

The reads of the KO parasite are in directory bam.



Right click here, select add BAM

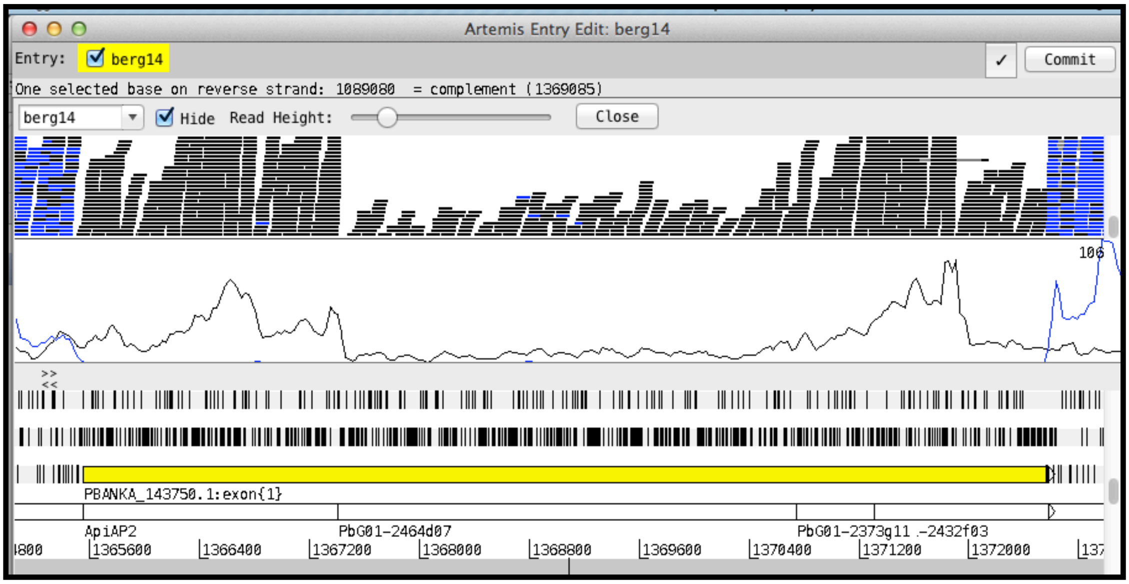


Include the file Pb_MUT1.bam.Chr14.bam from the bams directory.

In the BAM view of the reads, it might be difficult to distinguish the differences between the two different BAM files (data sets). But in the coverage plot, one can see the differences in coverage by the color. You can color the read by the coverage plot (right click BAMview -> color by -> Coverage plot colors.

First have a look at the knock out gene (PBANKA_143750). Is it really knocked out?

It seems quite convincing that this gene is not expressed at all in the mutant (blue coverage plot). So the knock out seem to have worked.



Artemis Entry Edit: berg14

Entry: ☑ berg14

One selected base on reverse strand: 1089080 = complement (1369085)

berg14 ☑ Hide Read Height:

Skim through the genome and compare the expression (coverage plots) between the two conditions. Again discuss the following questions with your neighbour or a tutor:

Which genes have extreme different coverage? Find a few and write the gene id numbers down.

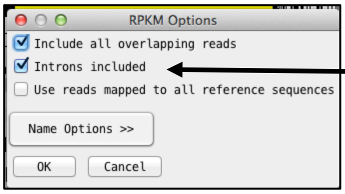
Is it enough to look at raw coverage, or would you need some kind of normalization?

F. Normalization - RPKM

One possibility of normalizing the data is to generate the RPKM for each gene. RPKM stands for **r**eads per **k**ilobase **p**er **m**illion mapped reads. It is a measure of how many reads map to a gene, normalized by the gene length and by the amount of mapped reads in the run.

1. Select all genes by: Click on Select -> All CDS Features

2. Right click on the BAM view -> Analysis -> RPKM values of selected features…



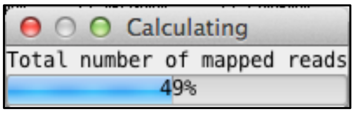
RPKM Options

☑ Include all overlapping reads

☑ Introns included

□ Use reads mapped to all reference sequences

3. Unselect "Intron included"



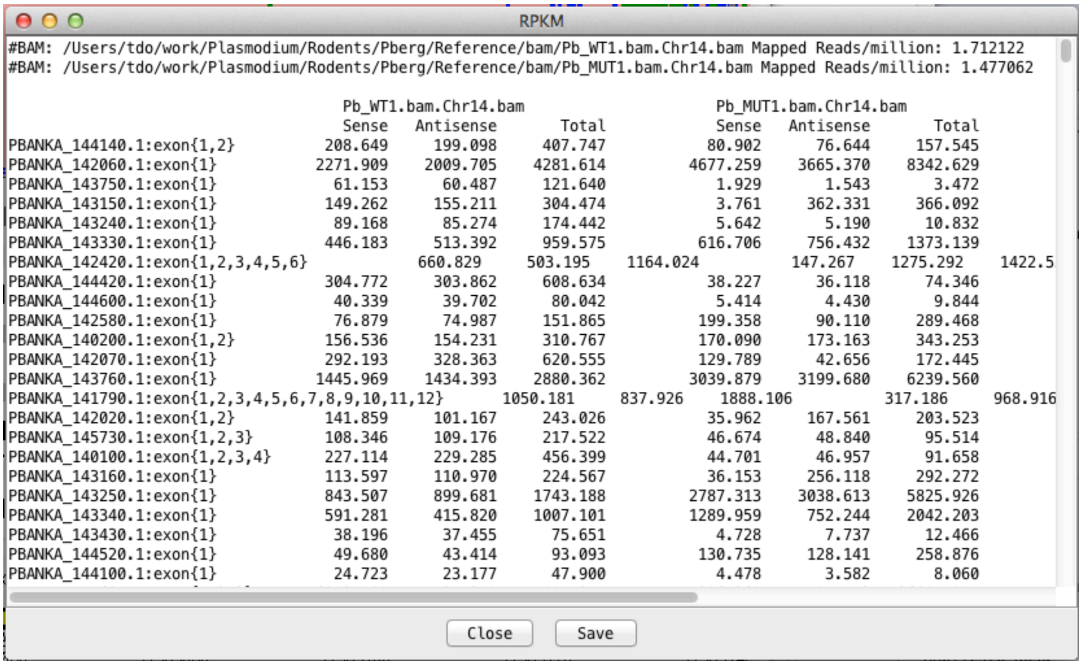
Calculating

Total number of mapped reads

4. Wait until the box says it is done.
Maybe take a break to do some stretching for your back… at home this will take longer. It is faster to use local copies of the BAM files!

5. The upcoming window will have RPKM values for each gene, for both the WT and the mutant. This will be split by strand of the DNA and a total score for both strands of DNA.

Save the file as Pb_RPKM.csv. This one you could load into LibreOffice (Excel), but here we are going to use a Linux "one-liner".



RPKM

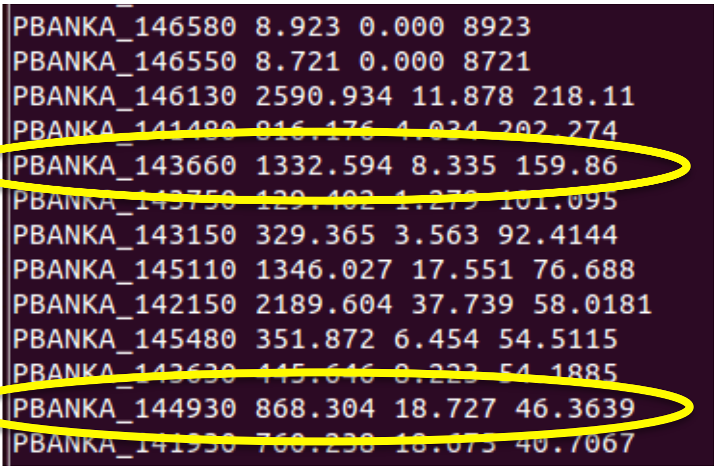
```
#BAM: /Users/tdo/work/Plasmodium/Rodents/Pberg/Reference/bam/Pb_WT1.bam.Chr14.bam Mapped Reads/million: 1.712122
#BAM: /Users/tdo/work/Plasmodium/Rodents/Pberg/Reference/bam/Pb_MUT1.bam.Chr14.bam Mapped Reads/million: 1.477062

                              Pb_WT1.bam.Chr14.bam                     Pb_MUT1.bam.Chr14.bam
                              Sense    Antisense      Total            Sense    Antisense      Total
PBANKA_144140.1:exon{1,2}       208.649    199.098    407.747           80.902     76.644    157.545
PBANKA_142060.1:exon{1}        2271.909   2009.705   4281.614         4677.259   3665.370   8342.629
PBANKA_143750.1:exon{1}          61.153     60.487    121.640            1.929      1.543      3.472
PBANKA_143150.1:exon{1}         149.262    155.211    304.474            3.761    362.331    366.092
PBANKA_143240.1:exon{1}          89.168     85.274    174.442            5.642      5.190     10.832
PBANKA_143330.1:exon{1}         446.183    513.392    959.575          616.706    756.432   1373.139
PBANKA_142420.1:exon{1,2,3,4,5,6}          660.829    503.195   1164.024          147.267   1275.292   1422.5
PBANKA_144420.1:exon{1}         304.772    303.862    608.634           38.227     36.118     74.346
PBANKA_144600.1:exon{1}          40.339     39.702     80.042            5.414      4.430      9.844
PBANKA_142580.1:exon{1}          76.879     74.987    151.865          199.358     90.110    289.468
PBANKA_140200.1:exon{1,2}       156.536    154.231    310.767          170.090    173.163    343.253
PBANKA_142070.1:exon{1}         292.193    328.363    620.555          129.789     42.656    172.445
PBANKA_143760.1:exon{1}        1445.969   1434.393   2880.362         3039.879   3199.680   6239.560
PBANKA_141790.1:exon{1,2,3,4,5,6,7,8,9,10,11,12}       1050.181     837.926    1888.106          317.186     968.916
PBANKA_142020.1:exon{1,2}       141.859    101.167    243.026           35.962    167.561    203.523
PBANKA_145730.1:exon{1,2,3}     108.346    109.176    217.522           46.674     48.840     95.514
PBANKA_140100.1:exon{1,2,3,4}   227.114    229.285    456.399           44.701     46.957     91.658
PBANKA_143160.1:exon{1}         113.597    110.970    224.567           36.153    256.118    292.272
PBANKA_143250.1:exon{1}         843.507    899.681   1743.188         2787.313   3038.613   5825.926
PBANKA_143340.1:exon{1}         591.281    415.820   1007.101         1289.959    752.244   2042.203
PBANKA_143430.1:exon{1}          38.196     37.455     75.651            4.728      7.737     12.466
PBANKA_144520.1:exon{1}          49.680     43.414     93.093          130.735    128.141    258.876
PBANKA_144100.1:exon{1}          24.723     23.177     47.900            4.478      3.582      8.060
```

Now we would like to know which genes have the biggest difference in terms of expression between them. One way is to generate the ratio of the RPKM of WT and KO and look at the most extreme values. This can be done very easily on the command line:

```
$ awk '{print $1,$4,$7,($4/($7+0.001)}' Pb_RPKM.csv | sort -rnk
4 | head -n 20
```

The `awk` commands can access columns in a file (like Excel) and do mathematical operations in this case the ratio. The output is piped into the `sort` program, that sort **n**umeric **r**everse and column 4 (k). And we are just interested in the top 20 lines (head -n 20).

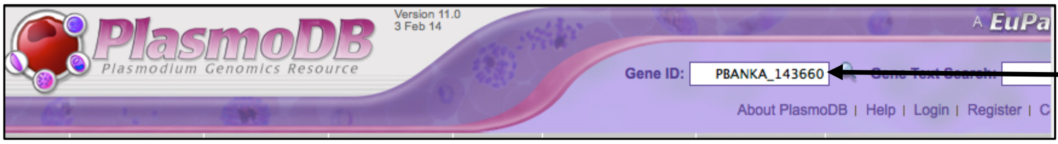
What happened if you try `tail` instead of `head`?



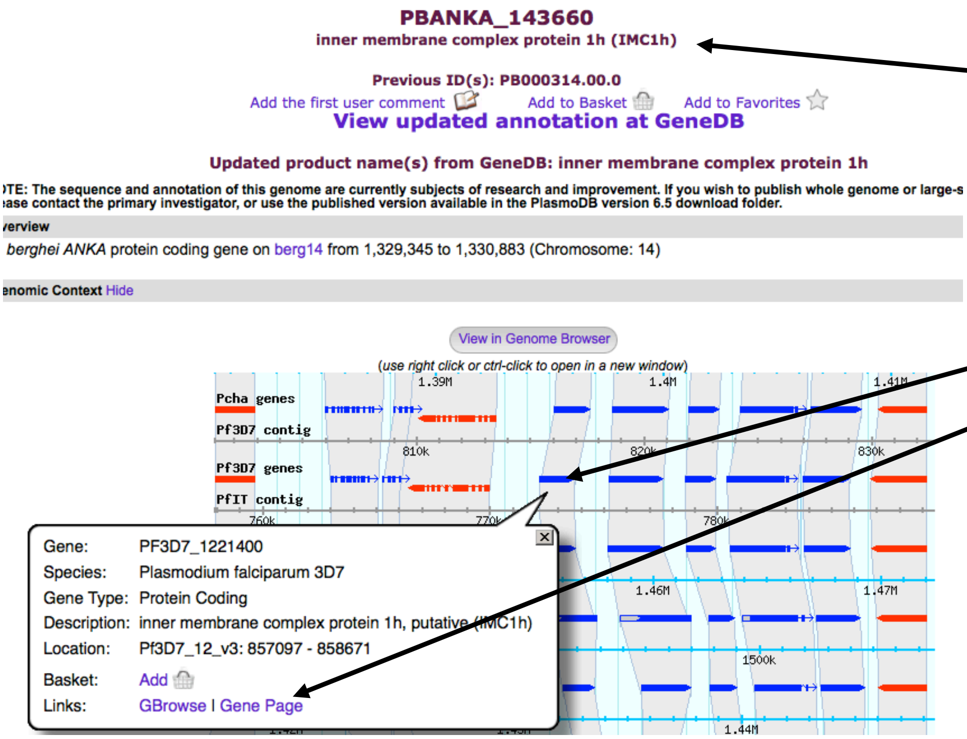
```
PBANKA_146580 8.923 0.000 8923
PBANKA_146550 8.721 0.000 8721
PBANKA_146130 2590.934 11.878 218.11
PBANKA_141480 816.176 4.034 202.274
PBANKA_143660 1332.594 8.335 159.86
PBANKA_143750 129.402 1.279 101.095
PBANKA_143150 329.365 3.563 92.4144
PBANKA_145110 1346.027 17.551 76.688
PBANKA_142150 2189.604 37.739 58.0181
PBANKA_145480 351.872 6.454 54.5115
PBANKA_143630 445.646 8.223 54.1885
PBANKA_144930 868.304 18.727 46.3639
PBANKA_141930 760.238 18.673 40.7067
```

If your values are different - maybe you filtered the reads differently.

Open at least the two marked genes in PlasmoDB (http://plasmodb.org) and enter the first (yellow) gene id.


PlasmoDB
Plasmodium Genomics Resource
Version 11.0
3 Feb 14
A EuPa
Gene ID:
PBANKA_143660
Gene Text Search:
About PlasmoDB | Help | Login | Register | C

1. Type the gene IDs in here.



PBANKA_143660
inner membrane complex protein 1h (IMC1h)

Previous ID(s): PB000314.00.0

Add the first user comment   Add to Basket   Add to Favorites
View updated annotation at GeneDB

Updated product name(s) from GeneDB: inner membrane complex protein 1h

)TE: The sequence and annotation of this genome are currently subjects of research and improvement. If you wish to publish whole genome or large-s
ease contact the primary investigator, or use the published version available in the PlasmoDB version 6.5 download folder.

verview

berghei ANKA protein coding gene on berg14 from 1,329,345 to 1,330,883 (Chromosome: 14)

enomic Context Hide

View in Genome Browser

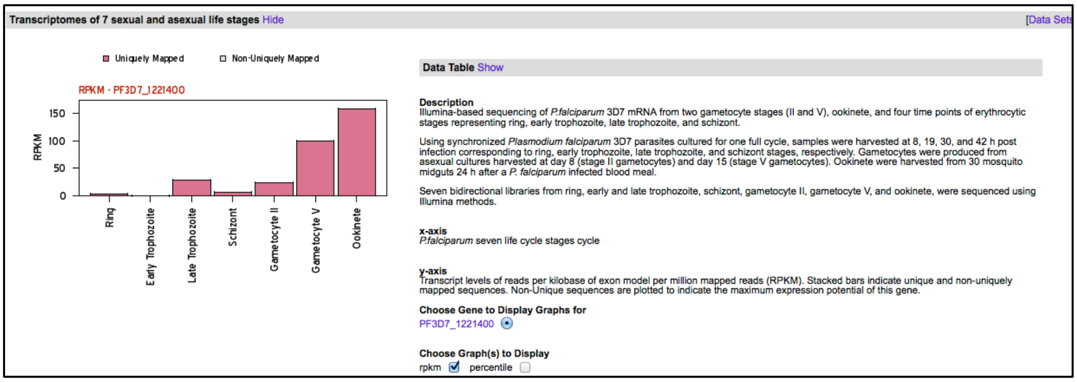
(use right click or ctrl-click to open in a new window)

Gene:	PF3D7_1221400
Species:	Plasmodium falciparum 3D7
Gene Type:	Protein Coding
Description:	inner membrane complex protein 1h, putative (IMC1h)
Location:	Pf3D7_12_v3: 857097 - 858671
Basket:	Add
Links:	GBrowse | Gene Page

2. Read the gene page. Does it tell you about the function of the down regulated gene?

3. The genome of *P. falciparum* 3D7 has a far richer annotation, so let's look at the orthologue.

Scroll down until you come to the transcriptome data for expression in the sexual stages .



Transcriptomes of 7 sexual and asexual life stages Hide [Data Sets

Data Table Show

Description
Illumina-based sequencing of *P.falciparum* 3D7 mRNA from two gametocyte stages (II and V), ookinete, and four time points of erythrocytic stages representing ring, early trophozoite, late trophozoite, and schizont.

Using synchronized *Plasmodium falciparum* 3D7 parasites cultured for one full cycle, samples were harvested at 8, 19, 30, and 42 h post infection corresponding to ring, early trophozoite, late trophozoite, and schizont stages, respectively. Gametocytes were produced from asexual cultures harvested at day 8 (stage II gametocytes) and day 15 (stage V gametocytes). Ookinete were harvested from 30 mosquito midguts 24 h after a *P. falciparum* infected blood meal.

Seven bidirectional libraries from ring, early and late trophozoite, schizont, gametocyte II, gametocyte V, and ookinete, were sequenced using Illumina methods.

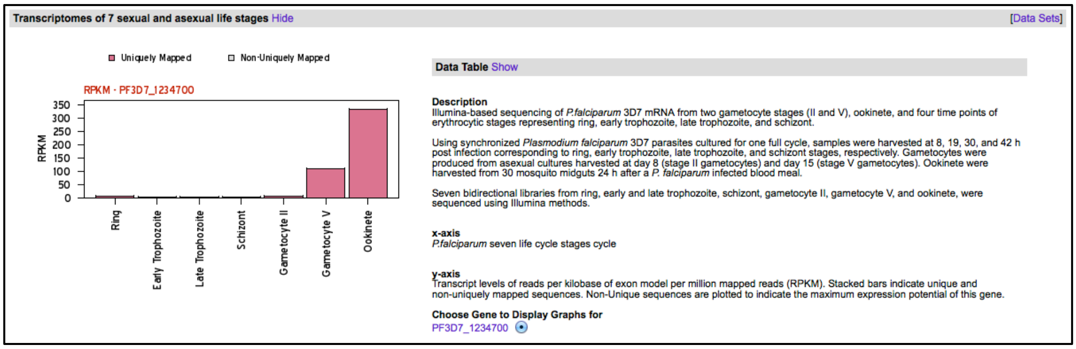
x-axis
P.falciparum seven life cycle stages cycle

y-axis
Transcript levels of reads per kilobase of exon model per million mapped reads (RPKM). Stacked bars indicate unique and non-uniquely mapped sequences. Non-Unique sequences are plotted to indicate the maximum expression potential of this gene.

Choose Gene to Display Graphs for
PF3D7_1221400 ●

Choose Graph(s) to Display
rpkm ☑ percentile □

Doing the same with the following gene (PBANKA_144930), that has the annotation “CPW-WPC family protein, putative“, returns a similar pattern.



Transcriptomes of 7 sexual and asexual life stages Hide [Data Sets]

Data Table Show

Description
Illumina-based sequencing of *P.falciparum* 3D7 mRNA from two gametocyte stages (II and V), ookinete, and four time points of erythrocytic stages representing ring, early trophozoite, late trophozoite, and schizont.

Using synchronized *Plasmodium falciparum* 3D7 parasites cultured for one full cycle, samples were harvested at 8, 19, 30, and 42 h post infection corresponding to ring, early trophozoite, late trophozoite, and schizont stages, respectively. Gametocytes were produced from asexual cultures harvested at day 8 (stage II gametocytes) and day 15 (stage V gametocytes). Ookinete were harvested from 30 mosquito midguts 24 h after a *P. falciparum* infected blood meal.

Seven bidirectional libraries from ring, early and late trophozoite, schizont, gametocyte II, gametocyte V, and ookinete, were sequenced using Illumina methods.

x-axis
P.falciparum seven life cycle stages cycle

y-axis
Transcript levels of reads per kilobase of exon model per million mapped reads (RPKM). Stacked bars indicate unique and non-uniquely mapped sequences. Non-Unique sequences are plotted to indicate the maximum expression potential of this gene.

Choose Gene to Display Graphs for
PF3D7_1234700 ●

When are those genes mostly expressed? Could you formulate a hypothesis what kind of genes the knocked out gene might control?

What genes would you expected to be up regulated in the mutant?

Conversely, how much can you trust those results? Could the variation be down to noise, or natural variation?

What extra data would be useful to help us to be more confident about our conclusions?

Differential Expression

Introduction

Understanding the genome is not simply about understanding which genes are there. Understanding when each gene is used helps us to find out how organisms develop and which genes are used in response to particular external stimuli. The first layer in understanding how the genome is used is the transcriptome. This is also the most accessible because like the genome the transcriptome is made of nucleic acids and can be sequenced relatively easily. Arguably the proteome is of greater relevance to understanding cellular biology however it is chemically heterogeneous making it much more difficult to assay.

Over the past decade or two microarray technology has been extensively applied to addressing the question of which genes are expressed when. Despite its success this technology is limited in that it requires prior knowledge of the gene sequences for an organism and has a limited dynamic range in detecting the level of expression, e.g. how many copies of a transcript are made. RNA sequencing technology using, for instance Illumina HiSeq machines, can sequence essentially all the genes which are transcribed and the results have a more linear relationship to the real number of transcripts generated.

The aim of differential expression analysis is to determine which genes are more or less expressed in different situations. We could ask, for instance, whether a bacterium uses its genome differently when exposed to stress, such as excess heat or a drug. Alternatively we could ask what genes make human livers different from human kidneys.

In this module we will try to gain more understanding of the genes differentially expressed between the wild type and knock out of our experiment. We are going to use three biological replicates of the WT and three biological replicats of the mutant to get more statistical power. Those were already mapped with tophat, as done before.

G. Finding differentially expressed genes with *cuffdiff*

Cuffdiff is a part of the cufflinks package which will enumerate the number of reads mapping to gene models in different RNAseq experiments and calculate those genes which have significantly different levels of expression.

Cufflinks requires a particular format of GFF file, which Artemis cannot output and so we introduce a Perl script to convert the EMBL file of chromosome 14 into the appropriate format. The role of Perl script as glue between different programs, converting one format to another, is very important in bioinformatics.

Convert the EMBL file into a GTF compatible with *cuffdiff*.

```
$ perl ./embl2gff.pl berg14.embl > berg14.gtf
```

Then use cuffdiff to determine which genes are differentially expressed:

```
$ cuffdiff -u -N berg14.gtf bams/Pb_WT1.bam.Chr14.bam,bams/
Pb_WT2.bam.Chr14.bam,bams/Pb_WT3.bam.Chr14.bam bams/
Pb_MUT1.bam.Chr14.bam,bams/Pb_MUT2.bam.Chr14.bam,bams/
Pb_MUT3.bam.Chr14.bam
```

Cuffdiff options for more accurate differential expression calculation:

•-u rescue method

- Where sequence is non-unique, spread the expression signal across identical regions based on their local expression level

•-N upper-quartile normalisation

- Rather than normalising the fragment counts for each gene by the total number of fragments sequenced, use the upper-quartile of fragments mapping to individual loci (more robust calls for less abundant genes)

Optional:
Run cuffdiff without the above options (-u, -N) and see how the results differ. How do your conclusions about differential expression of particular genes change?

Interpreting the results

Cuffdiff produces several files, but the one of interest to us is *gene_exp.diff*. This contains the statistics relating to the RNAseq read counts relating to each gene in the two timepoints. It is sorted by gene id, but it would be more useful to sort it by the significance of differential expression. Then the most clearly differentially expressed gene is at the top of the list.

Sort the results file by q-value (corrected p-value)

```
$ sort -k13 -g gene_exp.diff | more
```

Infact we can get the most useful result using the following command

```
$ sort -k13 -g gene_exp.diff | cut -f1,10,13,14 | grep yes
```

However we lose the headers and can't see which column is which so we can add in an extra command:

```
$ head -1 gene_exp.diff | cut -f1,10,13,14; sort -k13 -g
gene_exp.diff | cut -f1,10,13,14 | grep yes
```

How many genes are predicted to be differentially expressed?

How many are upregulated in the KO?

How many are downregulated?

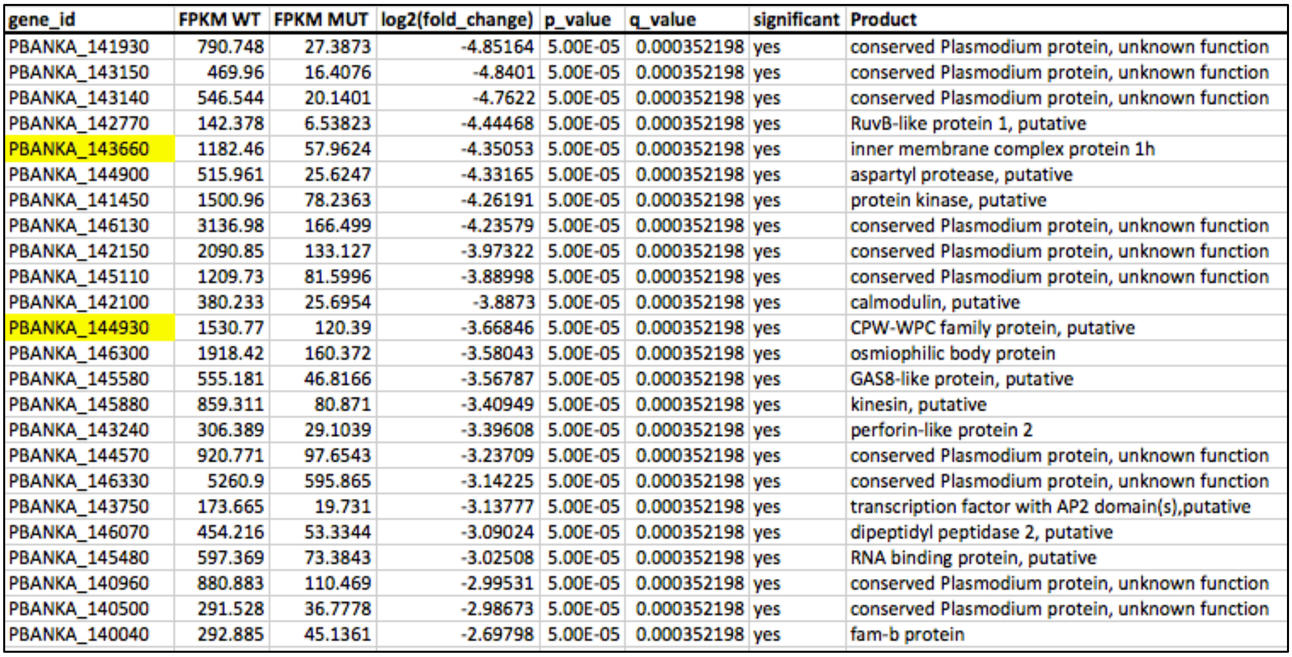
Now let’s compare this list to the one before. What are the differences? Is the list similar to your first list of differentially expressed genes?

Do you understand each column?

Which results would you trust more (this or the ratio in the Excel table)?

If time permits lookup more genes up in plasmodb…

What other datasets would help in the interpretation of the results?



gene_id	FPKM WT	FPKM MUT	log2(fold_change)	p_value	q_value	significant	Product
PBANKA_141930	790.748	27.3873	-4.85164	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_143150	469.96	16.4076	-4.8401	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_143140	546.544	20.1401	-4.7622	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_142770	142.378	6.53823	-4.44468	5.00E-05	0.000352198	yes	RuvB-like protein 1, putative
PBANKA_143660	1182.46	57.9624	-4.35053	5.00E-05	0.000352198	yes	inner membrane complex protein 1h
PBANKA_144900	515.961	25.6247	-4.33165	5.00E-05	0.000352198	yes	aspartyl protease, putative
PBANKA_141450	1500.96	78.2363	-4.26191	5.00E-05	0.000352198	yes	protein kinase, putative
PBANKA_146130	3136.98	166.499	-4.23579	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_142150	2090.85	133.127	-3.97322	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_145110	1209.73	81.5996	-3.88998	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_142100	380.233	25.6954	-3.8873	5.00E-05	0.000352198	yes	calmodulin, putative
PBANKA_144930	1530.77	120.39	-3.66846	5.00E-05	0.000352198	yes	CPW-WPC family protein, putative
PBANKA_146300	1918.42	160.372	-3.58043	5.00E-05	0.000352198	yes	osmiophilic body protein
PBANKA_145580	555.181	46.8166	-3.56787	5.00E-05	0.000352198	yes	GAS8-like protein, putative
PBANKA_145880	859.311	80.871	-3.40949	5.00E-05	0.000352198	yes	kinesin, putative
PBANKA_143240	306.389	29.1039	-3.39608	5.00E-05	0.000352198	yes	perforin-like protein 2
PBANKA_144570	920.771	97.6543	-3.23709	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_146330	5260.9	595.865	-3.14225	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_143750	173.665	19.731	-3.13777	5.00E-05	0.000352198	yes	transcription factor with AP2 domain(s),putative
PBANKA_146070	454.216	53.3344	-3.09024	5.00E-05	0.000352198	yes	dipeptidyl peptidase 2, putative
PBANKA_145480	597.369	73.3843	-3.02508	5.00E-05	0.000352198	yes	RNA binding protein, putative
PBANKA_140960	880.883	110.469	-2.99531	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_140500	291.528	36.7778	-2.98673	5.00E-05	0.000352198	yes	conserved Plasmodium protein, unknown function
PBANKA_140040	292.885	45.1361	-2.69798	5.00E-05	0.000352198	yes	fam-b protein

H. GO enrichment on the command line - OPTIONAL

Maybe some of you have already determined the function of the transcription factor. But this would have been done manually. A more automated method would be to do a GO enrichment. Basically, statistics are used to test if a function (or GO term) is enriched in the down or up regulated genes compared to all of the GO terms associated to the genes that are expressed.

Gene Ontology or GO, is a major bioinformatics initiative to unify the representation of gene and gene product attributes across all species, see http://en.wikipedia.org/wiki/Gene_ontology. GO terms are represented in directed acyclic graph, so functions can be further specified in a sub node. The GO enrichment test we will use takes the structure of this hierarchy into account.

But the association of GO terms to genes depend on the known functions and level of curation. For example, in *P. berghei*, less than half of the genes have GO terms associated!

In this exercise we will do a GO enrichment of the differentially expressed genes of the complete gene set (not just chromosome 14).

Change the directory and have a look at the files:

```
$ cd ~/Module_6_RNA-Seq/GO
$ ls
```

The file `full.gene_exp.diff` has the same format as the output of `cuffdiff` you produced. But it was generated for all of the genes in the genome, not just for chromosome 14.

The next command will get all the gene ids of genes that are

- differentially expressed (`grep yes`)
- down regulated in the mutant (`$10<0` - log fold change),
- have a FPKM of at least 40 (`$8>40` - FPKM of WT),
- are three times more expressed in the WT (`$8 > (3*$9)`).

To do the filtering, we are using the command `awk`. The "$" refers to the i-th column in the text file. As the first row contains the id's, it is returned with `cut -f 1` and then saved in a file, using the ">" command. (You could do that in excel, but it might take a bit more time…)

```
$ grep yes full.gene_exp.diff | awk '$10<0 && $8>40 && $8>(3*
$9)' | cut -f 1 > list.down.txt
$ head list.down.txt
$ head Pb.GOterms.txt
```

The two head commands give you an idea of the format of the two files we are going to use.

Though the enrichment test is done in R, using the bioconductor class topGO, we are going to call it directly from the command line. Maybe have a quick look at the code to see how the enrichment is done.

```
$ cat doGO.R
```

So next we are going to call the program, looking for the biological process (BP), see http://en.wikipedia.org/wiki/Gene_ontology.

```
$ R CMD BATCH "--args list.down.txt Pb.GOterms.txt BP “ doGO.R
```

This command tells R to run from the command line the program `doGO.R`. Three parameters are given:

1. Genes of interest - which you generated
2. GO database
3. The domain search: BP (biological process, e.g. cell cycle), MF (molecular function, e.g. kinase) or CC (cellular component, e.g. nucleus, cytoplasm)

The result is in file Result.txt

```
$ cat Result.txt
```

Google the first hit, “microtubule-based movement” including “malaria” as further search term. What paper pops out first? Does this help to understand which genes the knocked out transcription factor might regulate?

Can you repeat the analysis with with the other GO domains (CC and MF)?

Would you be able to repeat the analysis with up regulated genes in the mutant? Which processes are enriched. Are the results expected?

Would it make sense to change the criteria to generate the list of up and down regulated genes? If so, how and why?

Do not panic…

… if you don’t understand everything! This is a very advanced methodology. It involved bioinformatics, statistics and deep knowledge into the parasite. At the same time, the results depend on many parameters like, experiment setup, quality of your RNA-Seq data, parameter used in the different steps and the quality of the GO database.

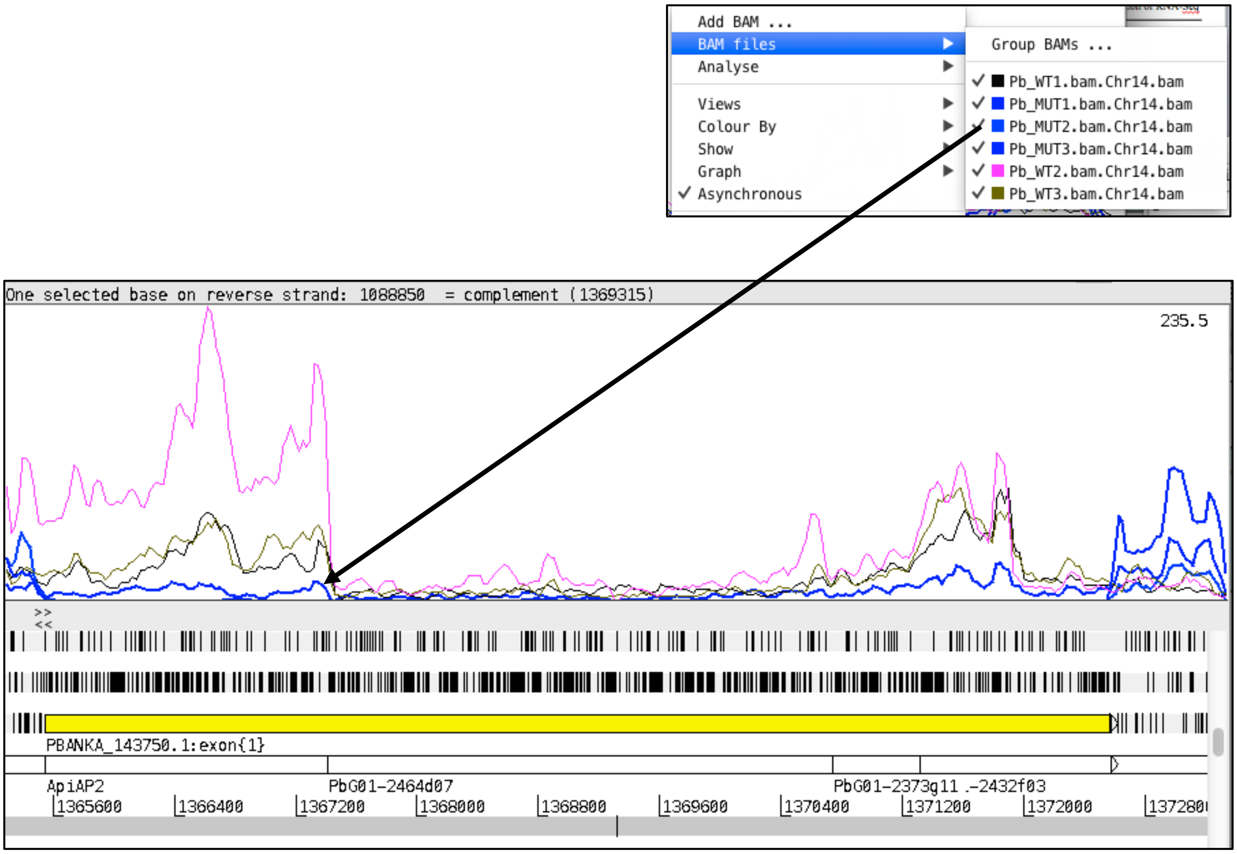
Important: In the end you got several enriched functions as result of your experiment that characterize the function of the knocked out gene! *Well done!*

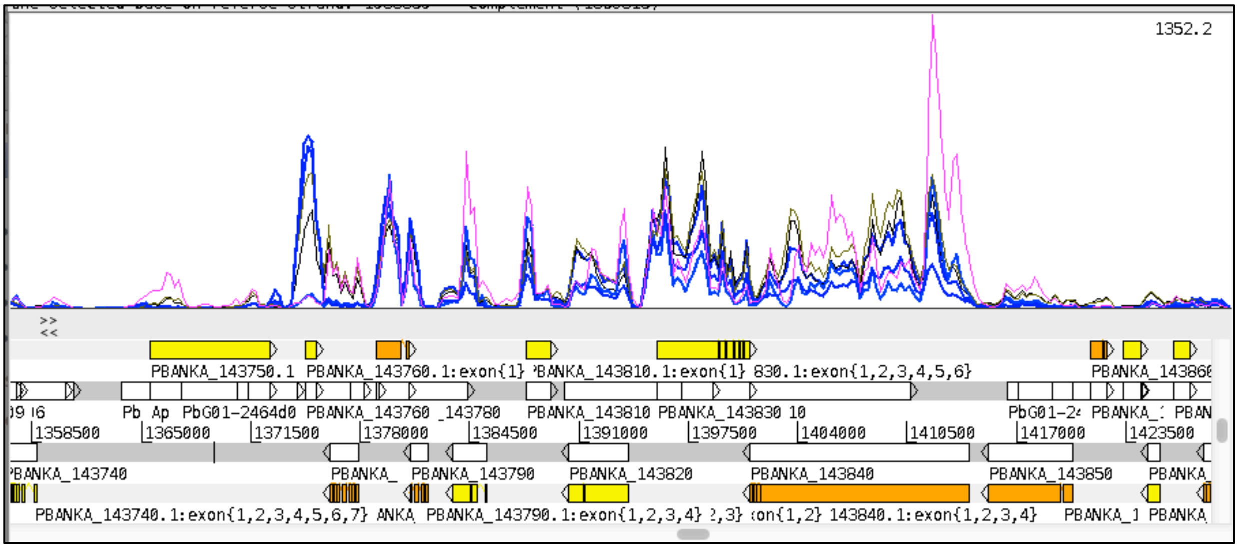
OPTIONAL: I. Including more data set

If time permits, include the further 4 data sets in Artemis (2 WT and 2 mutants, all on webpage), which we used in the differential expression. Skim through the genome and think about following questions:

How well do they correlate? Do the differential expression results make sense?

Is the Api AP2 knocked out in all mutant data sets? Would you need to redo the differentail expression?





Key aspects of differential expression analysis

Replicates and power

In order to accurately ascertain which genes are differentially expressed and by how much it is necessary to use replicates. As with all biological experiments doing it once is simply not enough. There is no simple way to decide how many replicates to do, it is usually a compromise of statistical power and cost. Although we have seen that statistically significant differences in gene expression can be ascertained without replicates, this is often not the case. By determining how much variability there is in the sample preparation and sequencing reactions we can better assess whether genes are really expressed and more accurately determine any differences. The key to this is performing biological rather than technical replicates. This means, for instance, growing up three batches of parasites, treating them all identically, extracting RNA from each and sequencing the three samples separately. Technical replicates, whereby the same sample is sequenced three times do not account for the variability that really exists in biological systems or the experimental error between batches of parasites and RNA extractions.

N.B. More replicates will help improve power for genes that are already detected at higher levels, while deeper sequencing will improve power to detect differential expression for genes which are expressed at lower levels.

P-values vs. Q-values

When asking whether a gene is differentially expressed we use statistical tests to assign a P-value. If a gene has a P-value of 0.05 we know that there is only a 5% chance that it is not really differentially expressed**.** However, if we are asking this question for every gene in the genome (~5,500 genes for *Plasmodium* parasites), then we would expect to see P-values less than 0.05 for many genes even though they are not really differentially expressed. Due to this statistical problem we must correct the P-values so that we are not tricked into accepting a large number of erroneous results. Q-values are P-values which have been corrected for what is known as **multiple hypothesis testing**. Therefore it is a Q-value of less than 0.05 that we should be looking for when asking whether a gene is differentially expressed.

What do I do with a gene list?

Differential expression analysis results is a list of genes which show differences between two conditions. It can be daunting trying to determine what the results mean. On one hand you may find that that there are no real differences in your experiment. Is this due to biological reality or noisy data? On the other hand you may find several thousands of genes are differentially expressed. What can you say about that?

Other than looking for genes you expect to be different or unchanged, one of the first things to do is look at Gene Ontology (GO) term enrichment. There are many different algorithms for this, but you should annotate your genes with functional terms from GO using for instance Blast2GO (Conesa et al., 2005) and then use perhaps TopGO (Alexa et al., 2005) to determine whether any particular sorts of genes occur more than expected in your differentially expressed genes.

Alternative software to cuffdiff

There are a variety of programs for detecting differential expression in RNA-Seq data: DESeq (Anders & Huber, 2010), EdgeR (Robinson et al., 2010) and BaySeq (Hardcastle & Kelly, 2010) are good examples.

References

Alexa A, Rahnenfuhrer J, Lengauer T. 2006. Improved scoring of functional groups from gene expression data by decorrelating GO graph structure. Bioinformatics 22(13): 1600-1607.

Anders S, Huber W. 2010. Differential expression analysis for sequence count data. Genome Biol 11(10): R106.

Conesa A, Gotz S, Garcia-Gomez JM, Terol J, Talon M, Robles M. 2005. Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research. Bioinformatics 21(18): 3674-3676.

Hardcastle TJ, Kelly KA. 2010. baySeq: empirical Bayesian methods for identifying differential expression in sequence count data. BMC Bioinformatics 11: 422.

Lawton J, Brugat T, Yan YX, Reid AJ, Bohme U, Otto TD, Pain A, Jackson A, Berriman M, Cunningham D et al. 2012. Characterization and gene expression analysis of the cir multi-gene family of Plasmodium chabaudi chabaudi (AS). BMC Genomics 13: 125.

Otto et al. (2010) Mol Microbiol Apr;76(1):12-24. New insights into the blood stage transcriptome of Plasmodium falciparum using RNA-Seq.

Robinson MD, McCarthy DJ, Smyth GK. 2010. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26(1): 139-140.

Spence PJ, Jarra W, Levy P, Reid AJ, Chappell L, Brugat T, Sanders M, Berriman M, Langhorne J. 2013. Vector transmission regulates immune control of Plasmodium virulence. Nature 498(7453): 228-231.

Trapnell C, Pachter L, Salzberg SL. 2009. TopHat: discovering splice junctions with RNA-Seq. Bioinformatics 25(9): 1105-1111.

Trapnell C, Williams BA, Pertea G, Mortazavi A, Kwan G, van Baren MJ, Salzberg SL, Wold BJ, Pachter L. 2010. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nature biotechnology 28(5): 511-515.