RNA-Seq & Differential Expression

Introduction

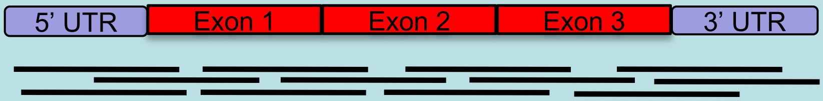
Understanding the genome is not simply about understanding which genes are there. Understanding when each gene is used helps us to find out how organisms develop, and which genes are used in response to particular external stimuli. The first layer in understanding how the genome is used is the transcriptome. This is also the most accessible because like the genome the transcriptome is made of nucleic acids and can (indirectly) be sequenced using the same technology. 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 in the cell.

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 pathogen uses its genome differently when exposed to stress, such as excessive heat or a drug. What happens if a gene gets silenced (this exercise), and how does that change the transcription profile? Alternatively we could ask what genes make human livers different from human kidneys.

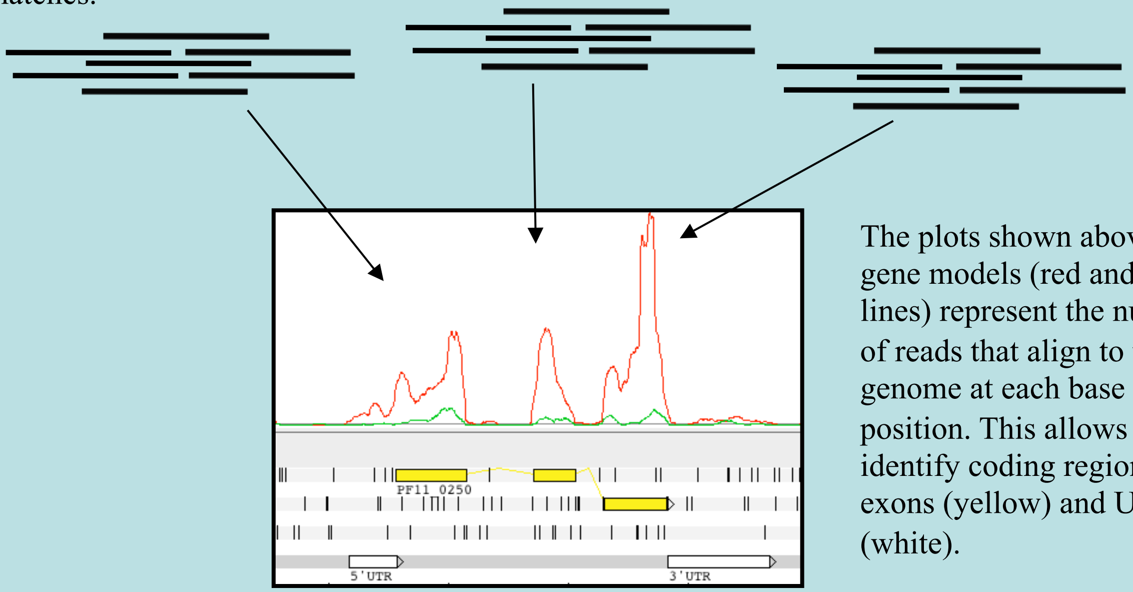
New technologies allow us to sequence RNA from single cells, understanding the heterogeneity of complex beings, however bulk RNA sequencing is still relevant for many applications, so let’s focus on that first in this exercise about malaria.

This is a graphical representation of how RNA-Seq works. Imagine this transcript is present in a 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. Some years later a group used RNA-seq to identify differentially expressed genes, showing that parasites from vector transmitted infections are less virulent than serially blood passaged parasites from the laboratory (Spence et al. 2013 – a possible group task). In www.PlasmoDB.org there are more than 40 transcriptomes for *Plasmodium*, and many more for the other parasites. Let's start to have a look at one RNA-Seq dataset:

Exercise

In this exercise you will need to determine the function of a gene that was knocked out. Your task is to find out the function of this gene, off you go!

A. Mapping with HiSat2- faster, less memory

We have two conditions, wildtype (WT) and knock out (KO). First, we are going to focus on the WT and map the reads against the reference genome of *Plasmodium berghei.* The tool we are using, HiSat2 is rather fast, but we did decrease the amount of reads to 30% for speed and space issues, compared to the original data.

First, change to the directory `~/Module_4_RNA-Seq (using cd)` and check if you are in the correct directory (pwd).

For the mapping, first an index of the reference (`PbANKA_v3.fasta`) must be constructed with hisat2-build. On the command syntax is: ***(Remember the names in italic are variables that you have to set!)***

```
$ hisat2-build reference_file index_name
```

For the index name you can use for example **`PbANKA_v3`**.

This will generate the index need for the mapping. You can ignore most of the output. Hisat2 is an improved version of Tophat2, and is several times faster. To start the command you should type:

```
$ hisat2 -p 4 --max-intronlen 10000 -x index_name -1 reads_1 -2
reads_2 -S WT1.sam
```

The read file are called `WT1_1.fastq.gz WT1_2.fastq.gz` and the *index_name* you gave, see above.

The results of the mapping is in WT1.sam. The –p 4 option runs the mapping on four processors - why not use them if they are there?

Now you need to transform the WT1.sam file into a bam file and index it. Do you remember the command from yesterday? It was something with samtools, sort and index. Look in your Linux history or go back in the manual.

Check that your bam files exists by getting the mapping stats:

```
$ samtools flagstat WT1.bam
$ samtools index WT1.bam
```

If that worked and you index the bam file, we need to clean up, as those files are pretty huge. Now that we have the bam file, we don't need the sam file anymore. Let's have a look at the files

```
$ ls -lrS
```

As in the mapping exercise, is the sam file non empty? You can see the large file called WT1.sam. If the bam file was created correclty, you can delete it with

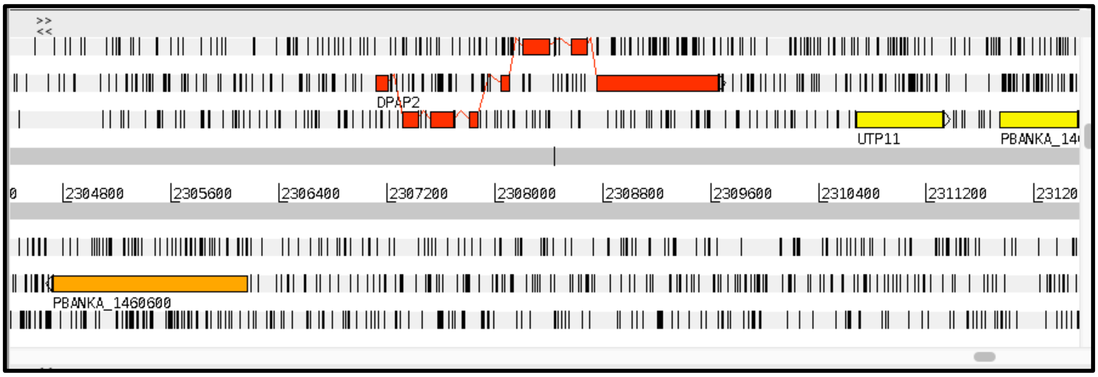
```
$ rm WT1.sam
```

B. Viewing the mapped reads in Artemis

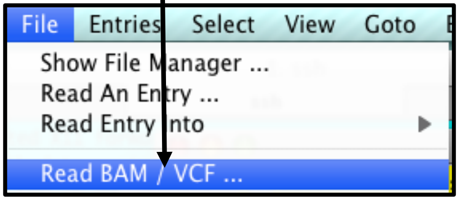
We will now examine the read mapping in Artemis using the BAM view feature. Be sure to be in the same directory as before. Open Artemis and load `PbANKA_14_v3.embl`. This file contains the sequence of chromosome 14 and the annotation. Here is also the gene that was knocked out.

```
$ art PbANKA_14_v3.embl &        ### to open Artemis
```

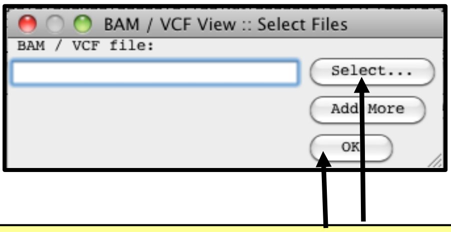
First go to the position 2308000 (Goto -> navigator).



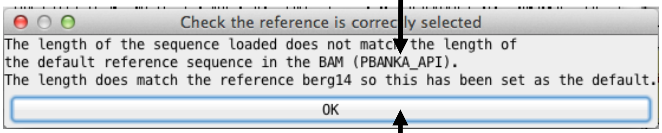
1. Now click on File -> "Reads BAM/ VCF…"



2. Select here the bam file you just generated (**prefixXXX.bam or WT1.bam**) and then press ok.



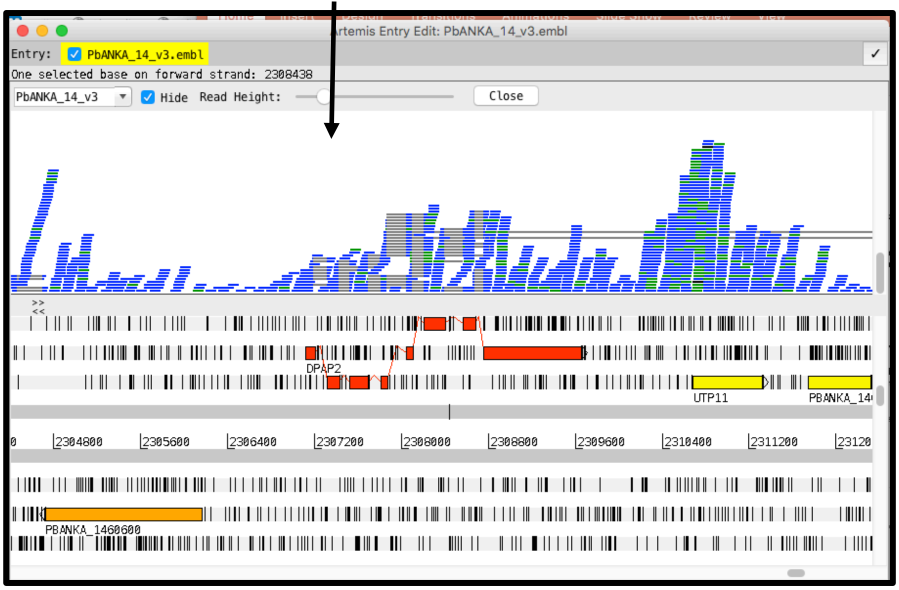
3. Confirm that the correct chromosome is chosen.



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?



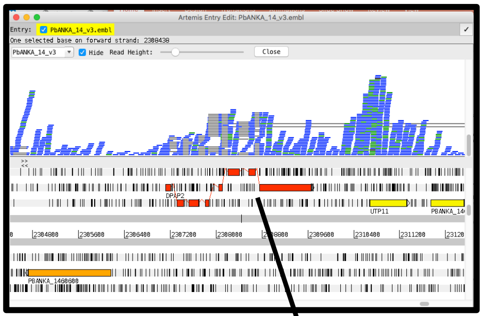
Artemis Entry Edit: PbANKA_14_v3.embl

Entry: ☑ PbANKA_14_v3.embl

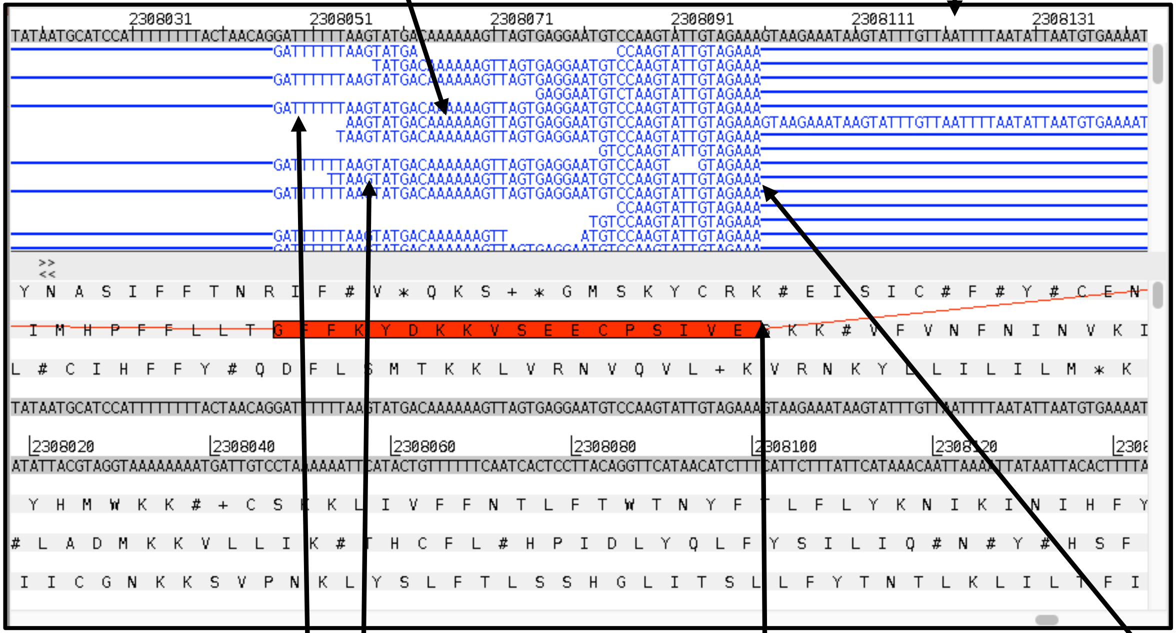
One selected base on forward strand: 2308438

PbANKA_14_v3 ☑ Hide Read Height:

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

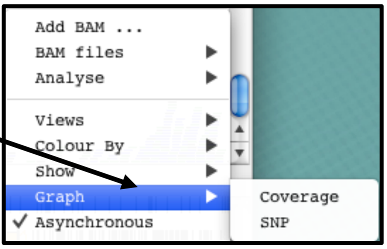

2308031
2308051
2308071
2308091
2308111
2308131
TATAATGCATCCATTTTTTTTACTAACAGGATTTTTTAAGTATGACAAAAAAGTTAGTGAGGAATGTCCAAGTATTGTAGAAAGTAAGAAATAAGTATTTGTTAATTTTAATATTAATGTGAAAAT
GATTTTTTAAGTATGA
CCAAGTATTGTAGAAA
TATGACAAAAAAGTTAGTGAGGAATGTCCAAGTATTGTAGAAA
GAGGAATGTCTAAGTATTGTAGAAA
AAGTATGACAAAAAAGTTAGTGAGGAATGTCCAAGTATTGTAGAAAGTAAGAAATAAGTATTTGTTAATTTTAATATTAATGTGAAAAT
TAAGTATGACAAAAAAGTTAGTGAGGAATGTCCAAGTATTGTAGAAA
G F F K Y D K K V S E E C P S I V E
2308020
2308040
2308060
2308080
2308100
2308120
ATATTACGTAGGTAAAAAAAATGATTGTCCTAAAAATTCATACTGTTTTTTCAATCACTCCTTACAGGTTCATAACATCTTTCATTCTTTATTCATAAACAATTAAAATTATAATTACACTTTTA

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?

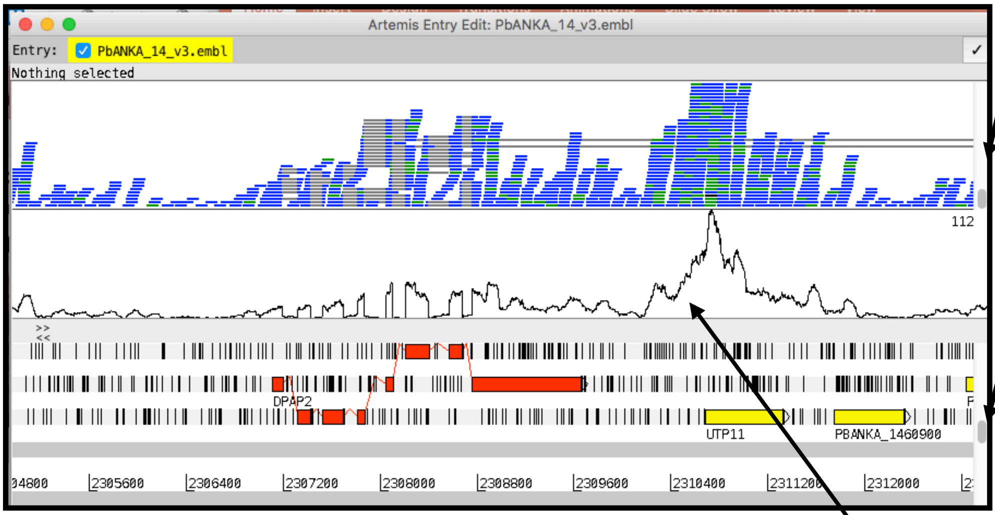
This is the so-called one-base pair resolution of RNA-Seq!

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



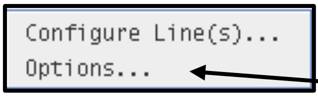
C. Interpreting the mapping

Zoom out until you have the same view as below:

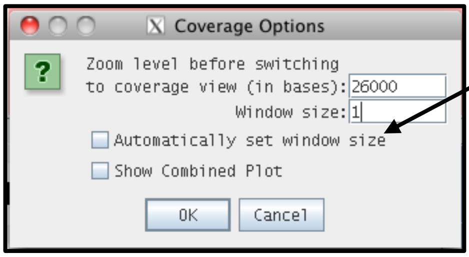


You can move the reads up and down, on the right scroll bar.

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…”)



Coverage Options

Zoom level before switching to coverage view (in bases): 26000

Window size: 1

□ Automatically set window size

□ Show Combined Plot

OK Cancel

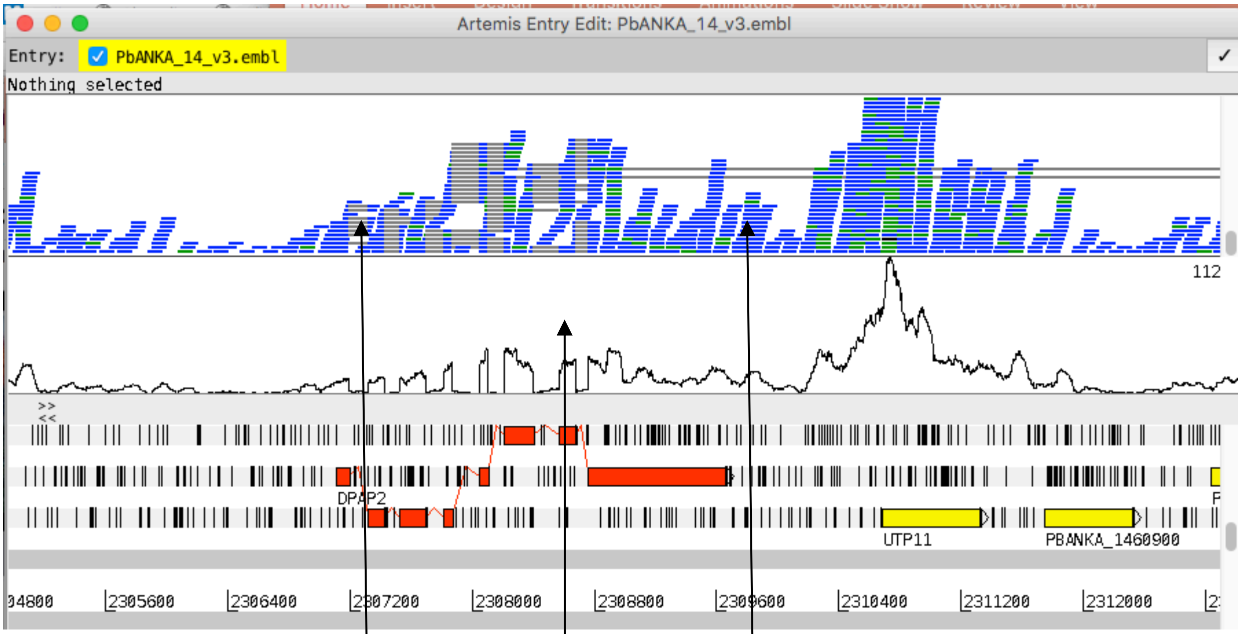
Please discuss following aspects with your neighbour:

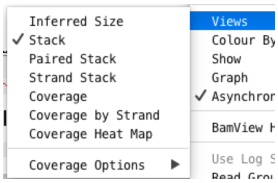
The coverage represents the amount of reads mapped over each position. Why are reads mapped where no exons are? Can you distinguish transcription start and stop sites of the genes?

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

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

Actually, these data are strand specific! This means that from the reads you can determine the orientation of the transcript. This is useful to determine the start and end of genes - that might also help to answer the questions from the page before, like where genes start, when they are head to head.



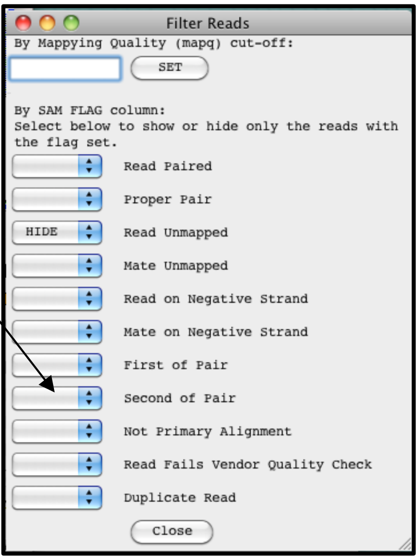


1. Right click, ->View -> strand stack

2. Right click, ->Filter Reads…

3. **Show** the second pair

4. Right click, on plot, -> Options -> plot by strand!



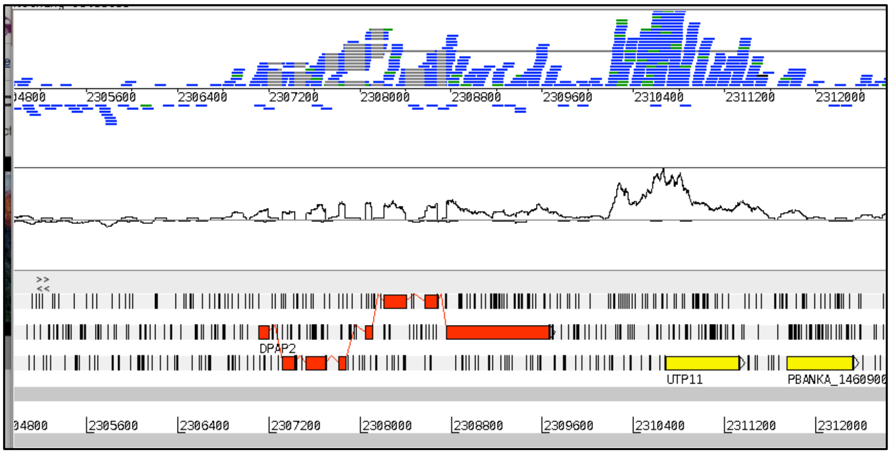
Filter Reads

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

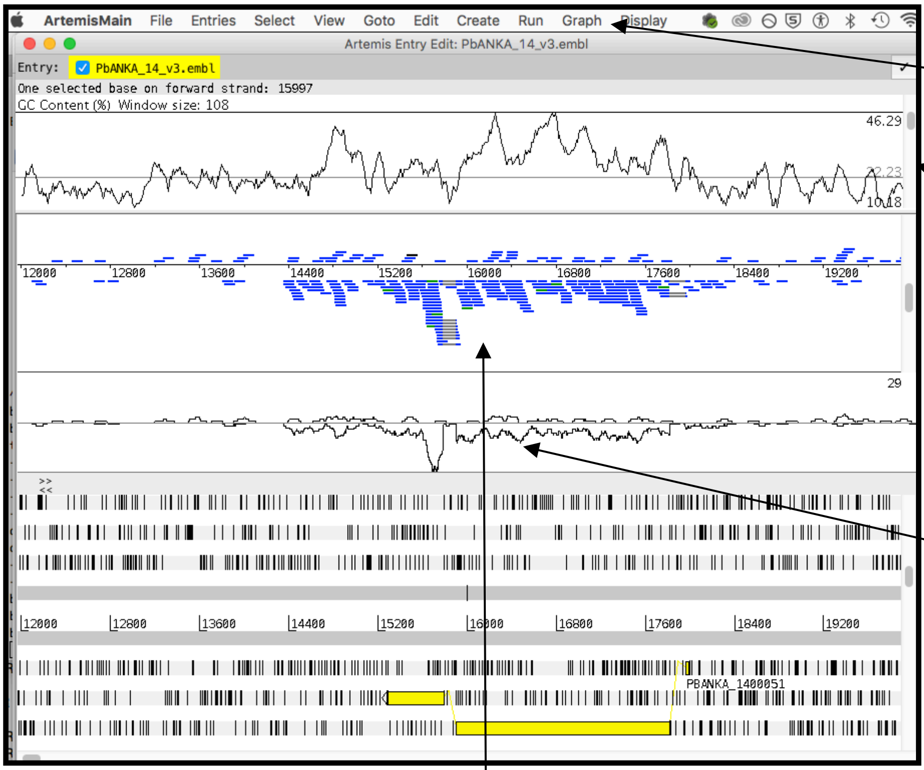
Close



NICE!

D. Uniqueness and GC content

Go to the position 16000 (Goto -> navigator).


ArtemisMain File Entries Select View Goto Edit Create Run Graph Display
Artemis Entry Edit: PbANKA_14_v3.embl
46.29
22.23
10.18
12000
12800
13600
14400
15200
16000
16800
17600
18400
19200
29
PBANKA_1400051

Entry: ☑ PbANKA_14_v3.embl

One selected base on forward strand: 15997

GC Content (%) Window size: 108

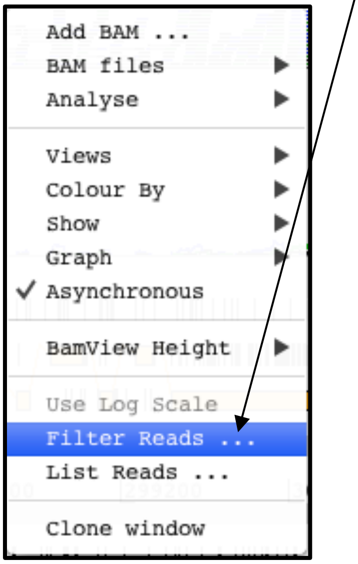
1. Enable the GC content, Graph -> GC Content.

2. Change the window size

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

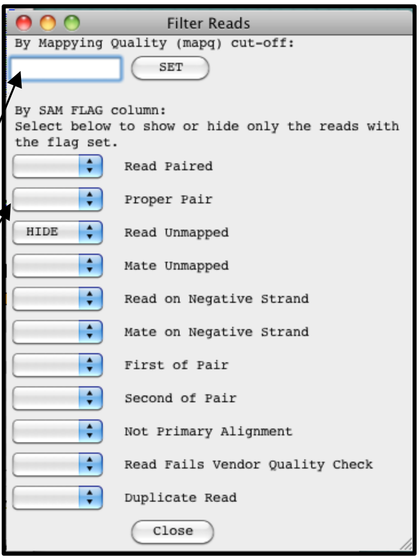
4. You can filter reads by mapping quality and if they are mapped as proper mate pairs.

5. Right click, then Filter Reads…



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

What does that mean for the expression values of those genes?


Filter Reads

By Mappying Quality (mapq) cut-off:

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

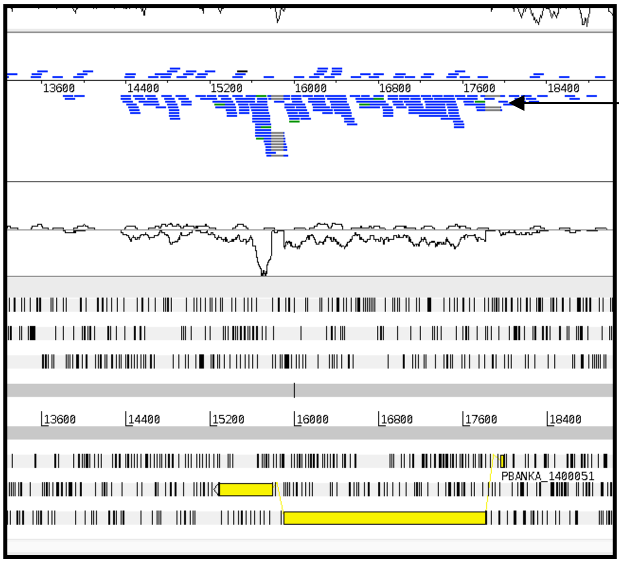
	Read Paired
	Proper Pair
HIDE	Read Unmapped
	Mate Unmapped
	Read on Negative Strand
	Mate on Negative Strand
	First of Pair
	Second of Pair
	Not Primary Alignment
	Read Fails Vendor Quality Check
	Duplicate Read

Variation in coverage can occur for many reasons, one is GC content. Importantly, reads can also 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 5.

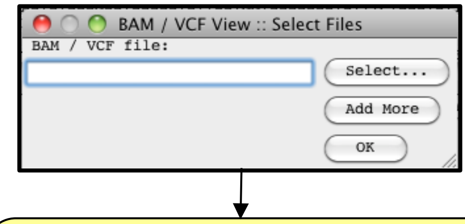
E. Including the mutant data set

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

The reads of the KO parasite have already been aligned and are in directory called bams.



Right click here, select add BAM

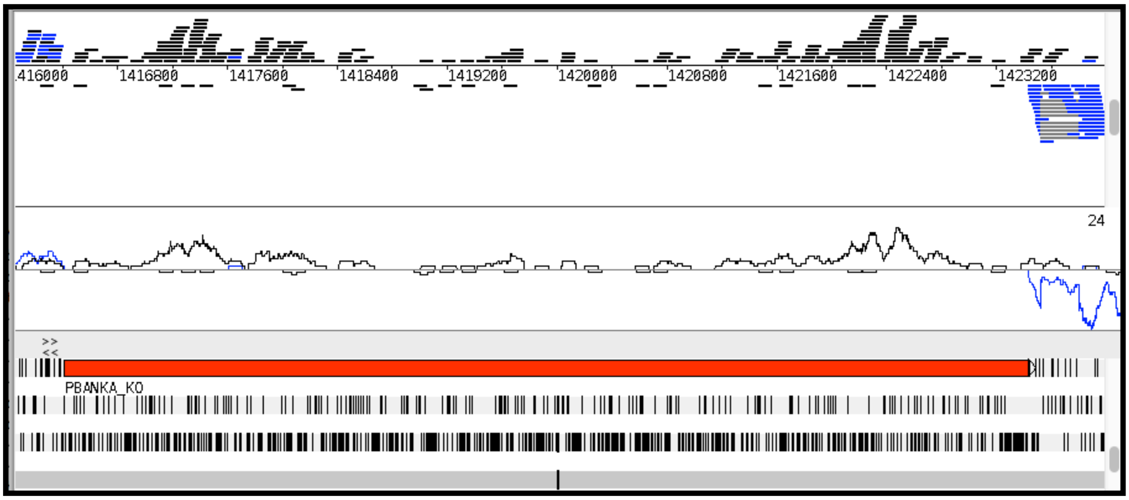


Include the file KO1.bam from the bams directory.

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 colour. You can colour the read by the coverage plot (right click BAMview -> colour by -> Coverage plot colours.

First have a look at the knock out gene (PBANKA_KO). 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 seems to have worked. Interestingly, expression is qquite low in the WT. Do you think this is still an important gene?



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.

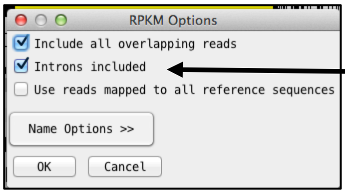
Is it enough to look at raw coverage, or would you need some kind of normalization?

F. Normalization - RPKM

You have seen that the coverage varies between the two samples and it is difficult to compare. 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

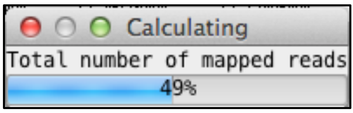
2. Right click on the BAM view -> Analysis -> RPKM values of selected features…



RPKM Options

☑ Include all overlapping reads
☑ Introns included
☐ Use reads mapped to all reference sequences

3. Unselect "Intron included"



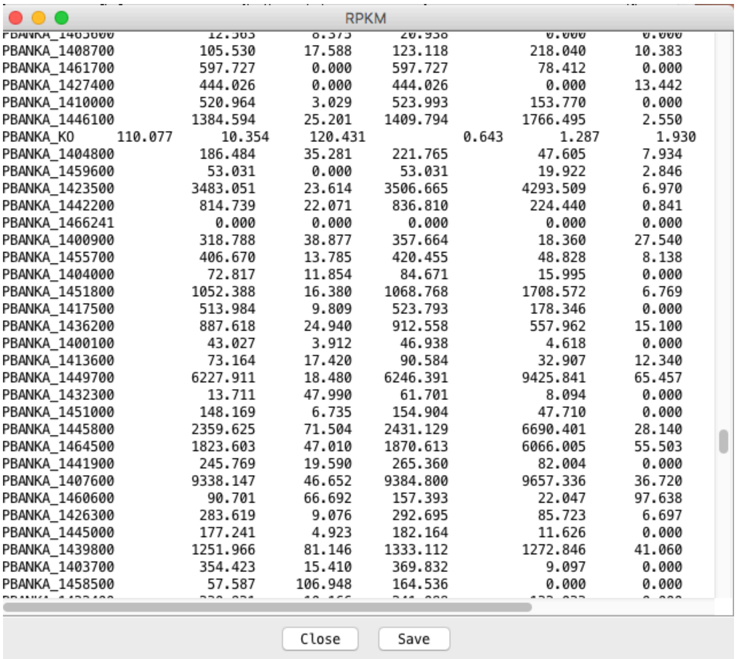
Calculating

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

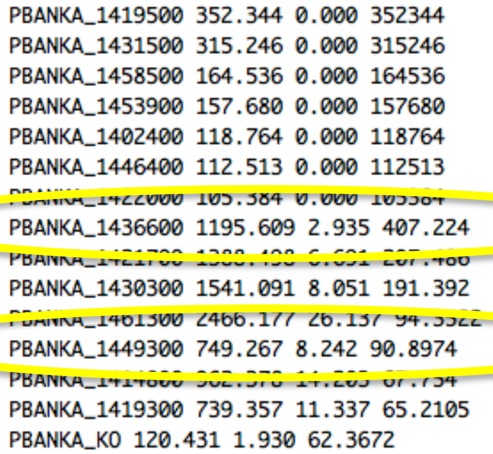
```
PBANKA_1408700         105.530     17.588     123.118          218.040     10.383
PBANKA_1461700         597.727      0.000     597.727           78.412      0.000
PBANKA_1427400         444.026      0.000     444.026            0.000     13.442
PBANKA_1410000         520.964      3.029     523.993          153.770      0.000
PBANKA_1446100        1384.594     25.201    1409.794         1766.495      2.550
PBANKA_KO      110.077       10.354     120.431          0.643         1.287       1.930
PBANKA_1404800         186.484     35.281     221.765           47.605      7.934
PBANKA_1459600          53.031      0.000      53.031           19.922      2.846
PBANKA_1423500        3483.051     23.614    3506.665         4293.509      6.970
PBANKA_1442200         814.739     22.071     836.810          224.440      0.841
PBANKA_1466241           0.000      0.000       0.000            0.000      0.000
PBANKA_1400900         318.788     38.877     357.664           18.360     27.540
PBANKA_1455700         406.670     13.785     420.455           48.828      8.138
PBANKA_1404000          72.817     11.854      84.671           15.995      0.000
PBANKA_1451800        1052.388     16.380    1068.768         1708.572      6.769
PBANKA_1417500         513.984      9.809     523.793          178.346      0.000
PBANKA_1436200         887.618     24.940     912.558          557.962     15.100
PBANKA_1400100          43.027      3.912      46.938            4.618      0.000
PBANKA_1413600          73.164     17.420      90.584           32.907     12.340
PBANKA_1449700        6227.911     18.480    6246.391         9425.841     65.457
PBANKA_1432300          13.711     47.990      61.701            8.094      0.000
PBANKA_1451000         148.169      6.735     154.904           47.710      0.000
PBANKA_1445800        2359.625     71.504    2431.129         6690.401     28.140
PBANKA_1464500        1823.603     47.010    1870.613         6066.005     55.503
PBANKA_1441900         245.769     19.590     265.360           82.004      0.000
PBANKA_1407600        9338.147     46.652    9384.800         9657.336     36.720
PBANKA_1460600          90.701     66.692     157.393           22.047     97.638
PBANKA_1426300         283.619      9.076     292.695           85.723      6.697
PBANKA_1445000         177.241      4.923     182.164           11.626      0.000
PBANKA_1439800        1251.966     81.146    1333.112         1272.846     41.060
PBANKA_1403700         354.423     15.410     369.832            9.097      0.000
PBANKA_1458500          57.587    106.948     164.536            0.000      0.000
```

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 '$4>100{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. We just want genes that are expressed ($4>100).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). More of awk can be found in the Linux exercise.

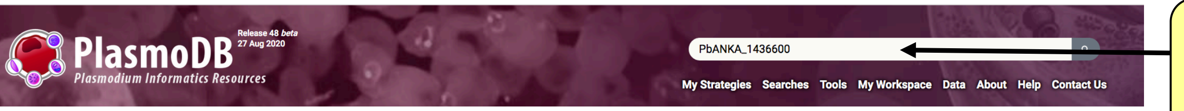
What happened if you try tail instead of head?



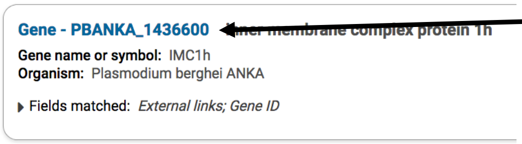
```
PBANKA_1419500 352.344 0.000 352344
PBANKA_1431500 315.246 0.000 315246
PBANKA_1458500 164.536 0.000 164536
PBANKA_1453900 157.680 0.000 157680
PBANKA_1402400 118.764 0.000 118764
PBANKA_1446400 112.513 0.000 112513
PBANKA_1422000 105.384 0.000 105384
PBANKA_1436600 1195.609 2.935 407.224
PBANKA_1421700 1388.498 6.691 207.486
PBANKA_1430300 1541.091 8.051 191.392
PBANKA_1461300 2466.177 26.137 94.3522
PBANKA_1449300 749.267 8.242 90.8974
PBANKA_1414800 962.378 14.203 67.754
PBANKA_1419300 739.357 11.337 65.2105
PBANKA_KO 120.431 1.930 62.3672
```

If your values are different - maybe you filtered the reads differently, and that is no problem at all!

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

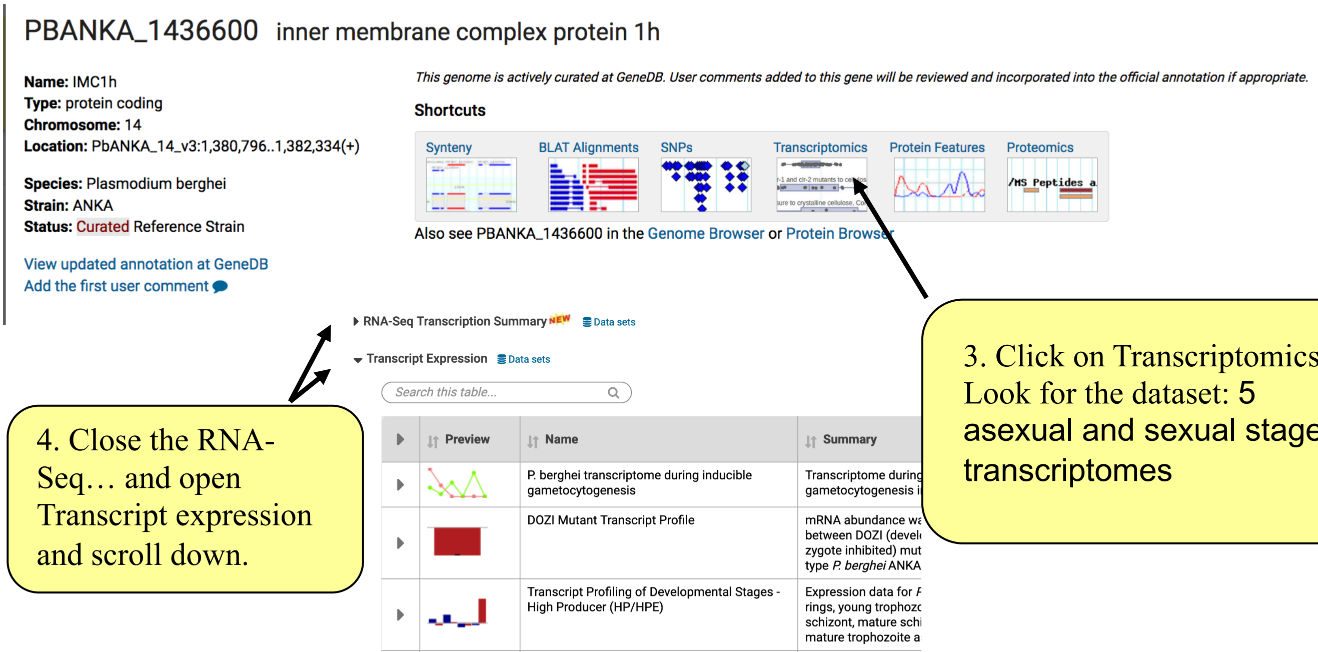


1. Type the gene IDs in here.



Gene - PBANKA_1436600 inner membrane complex protein 1h
Gene name or symbol: IMC1h
Organism: Plasmodium berghei ANKA
Fields matched: *External links; Gene ID*

2. Select the gene



PBANKA_1436600 inner membrane complex protein 1h

Name: IMC1h
Type: protein coding
Chromosome: 14
Location: PbANKA_14_v3:1,380,796..1,382,334(+)

Species: Plasmodium berghei
Strain: ANKA
Status: Curated Reference Strain

View updated annotation at GeneDB
Add the first user comment

This genome is actively curated at GeneDB. User comments added to this gene will be reviewed and incorporated into the official annotation if appropriate.

Shortcuts

Also see PBANKA_1436600 in the Genome Browser or Protein Browser

RNA-Seq Transcription Summary NEW Data sets

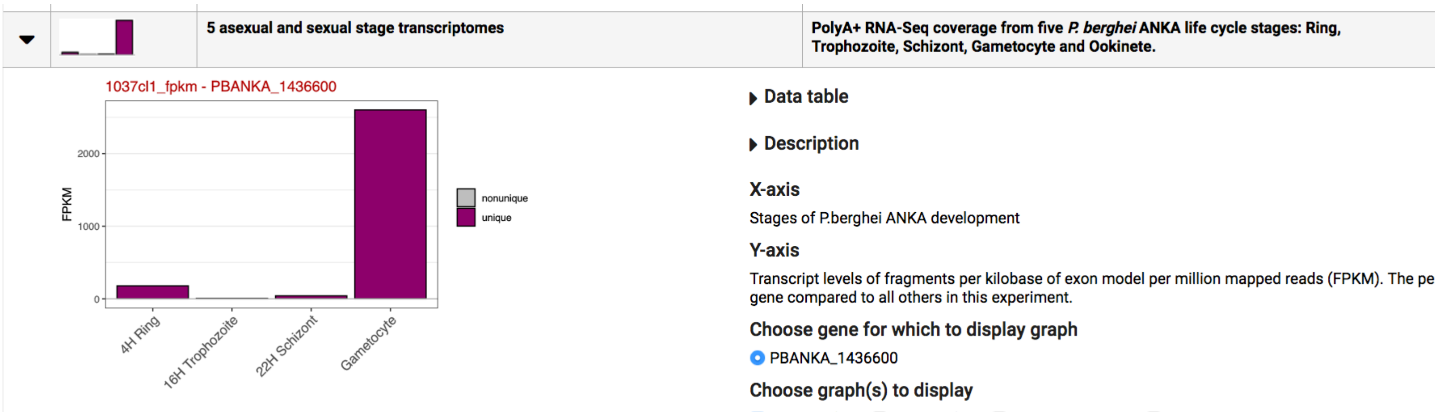
Transcript Expression Data sets

Search this table...

	Preview	Name	Summary
▸		P. berghei transcriptome during inducible gametocytogenesis	Transcriptome during gametocytogenesis i
▸		DOZI Mutant Transcript Profile	mRNA abundance wa between DOZI (devel zygote inhibited) mut type *P. berghei* ANKA
▸		Transcript Profiling of Developmental Stages - High Producer (HP/HPE)	Expression data for *F* rings, young trophozo schizont, mature schi mature trophozoite a

3. Click on Transcriptomics. Look for the dataset: 5 asexual and sexual stage transcriptomes

4. Close the RNA-Seq… and open Transcript expression and scroll down.


5 asexual and sexual stage transcriptomes

PolyA+ RNA-Seq coverage from five *P. berghei* ANKA life cycle stages: Ring, Trophozoite, Schizont, Gametocyte and Ookinete.

▸ Data table

▸ Description

X-axis

Stages of P.berghei ANKA development

Y-axis

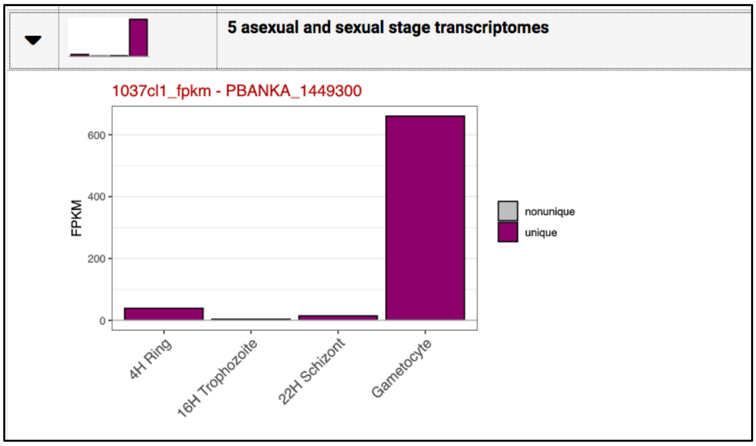
Transcript levels of fragments per kilobase of exon model per million mapped reads (FPKM). The pe gene compared to all others in this experiment.

Choose gene for which to display graph

● PBANKA_1436600

Choose graph(s) to display

Doing the same with the following gene (PBANKA_1449300), that has the annotation “CPW-WPC family protein, putative“, returns a similar pattern.


5 asexual and sexual stage transcriptomes

When are those genes mostly expressed? Could you formulate a hypothesis what kind of genes the knocked out gene might control? As a hint to non –parasitologist, Ring, Trophozoite and Schizont are blood stage, that the parasite uses to multiply.

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 replicates of the mutant to get more statistical power.

G. Using *Kalliso* and *Sleuth* to identify differentially expressed genes

Kallisto is a read mapper, but instead of mapping against the genome it is designed to map against the transcriptome, i.e. the spliced gene sequences inferred from the genome annotation. Rather than telling you the exact alignment for each read it aims to quantify the expression level of each transcript. It is very fast because it uses pseudo-alignment rather than true read alignment.

Kallisto needs an index of the **transcript sequences** (`Pb.CDS.fasta`). This is important if you want to use it later. The input is the mRNA sequences and not the genome like for STAR or HiSat2.

```
$ kallisto index -i Pb.transcript transcript_sequences
```

Quantify the expression levels of your transcripts for the WT1 sample. The read file are again **WT1_1.fastq.gz WT1_2.fastq.gz.**

```
$ kallisto quant -t 3 --rf-stranded -i index_name -o WT1 -b
100 read_1 read_2
```

The results are contained in the file WT1/abundance.tsv. The other 5 samples can be mapped with for following loop:

```
$ for x in WT2 WT3 KO1 KO2 KO3 ; do kallisto quant -t 3 --rf-stranded -i
Pb.transcript -o $x -b 100 $x\_1.fastq.gz $x\_2.fastq.gz; done
```

Sleuth uses the output from *Kallisto* to determine differentially expressed genes. It is written in the R statistical programming language, as is almost all RNA-seq analysis software. Helpfully however it produces a web page that allows interactive graphical analysis of the data. However, I would recommend learning R for anyone doing a significant amount of RNA-seq analysis. It is nowhere near as hard to get started with as full-blown programming languages such as Perl or Python!

We have provided a series of R commands which will get *Sleuth* running. These are in the file *sleuth.R*. Have a look with cat:

```
$ cat sleuth.R
```

It is not as hard as it seems, I copied most of this from the manual! To run this R script, you will have to open R:

```
$ R
> source("sleuth.R")
> sleuth_live(so)
```

This should open a browser…. When you are finished, press „crtl-c" in R and type q() to quit R

For some of you, firefox won't open use a different port, like 42429 or 42439

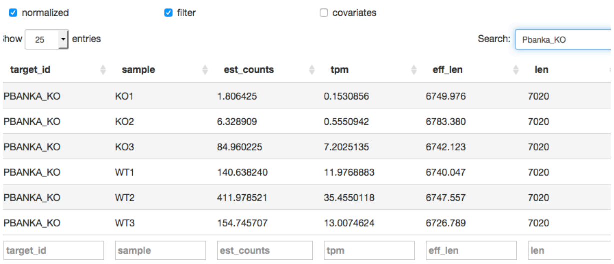
```
$ sleuth_live(so, options = list(port = 42429))
```

H. Using *Sleuth* to quality check the data

Sleuth provides several tabs which we can use to determine whether the data is of good quality and whether we should trust the results we get.

In the web page which has been launched click on Summaries->processed data.

Even though we have used the same number of reads for each sample, there are large differences in the number of reads mapping for each one. Why might this be? Is it a problem?

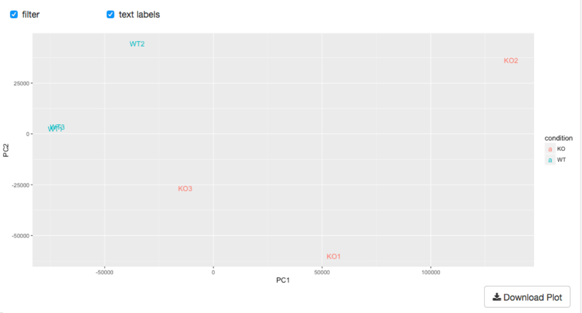


☑ normalized ☑ filter □ covariates

Show 25 entries

Search: Pbanka_KO

target_id	sample	est_counts	tpm	eff_len	len
PBANKA_KO	KO1	1.806425	0.1530856	6749.976	7020
PBANKA_KO	KO2	6.328909	0.5550942	6783.380	7020
PBANKA_KO	KO3	84.960225	7.2025135	6742.123	7020
PBANKA_KO	WT1	140.638240	11.9768883	6740.047	7020
PBANKA_KO	WT2	411.978521	35.4550118	6747.557	7020
PBANKA_KO	WT3	154.745707	13.0074624	6726.789	7020
target_id	sample	est_counts	tpm	eff_len	len



Click on map->PCA.

The Principal Components Analysis plot shows the relationship between the samples in two dimensions (PC1 and PC2). In this case almost all the variation between the samples is captured by just Principal Component 1. The WT samples are well separated from the KO samples, and they are close to each other. This meaning that the replicates are similar to each other something one want from a replicate. But it seems that one KO is a bit distant.

In some cases we identify outliers, e.g. samples which do not agree with other replicates and these can be excluded. If we don't have many replicates, it is hard to detect outliers and our power to detect differentially expressed genes is reduced.

BTW, for sleuth we needed to generate two more files! `hiseq_info.txt` and `Pb.CDS.fasta`, if you need to generate them, one is the name of the input files and the other product of the genes.

I. Interpreting the results

In the R script we printed out a file of results describing the differentially expressed genes in our dataset. This is called “kallisto.results”.

The file contains several columns, of which the most important are:

Column 1: target_id (gene id)
Column 2: pval (p value)
Column 3: qval (p value corrected for multiple hypothesis testing)
Column 4: b (fold change)
Column 12: description (some more useful description of the gene than its id)

Go back to Linux. With a little of magic we can get the list of differentially expressed genes with only the columns of interest as above. The following command will get those genes which have an adjusted p value less than 0.01 and a positive fold change. These genes are more highly expressed in SBP samples.

```
$ cut -f1,2,4,5 kallisto.results | awk -F "\t"  '$3 < 0.01 && $4  > 0' | head
```

These genes are more highly expressed in WT samples:

```
$ cut -f1,2,4,5 kallisto.results | awk -F "\t"  '$3 < 0.01 && $4  < 0' | head
```

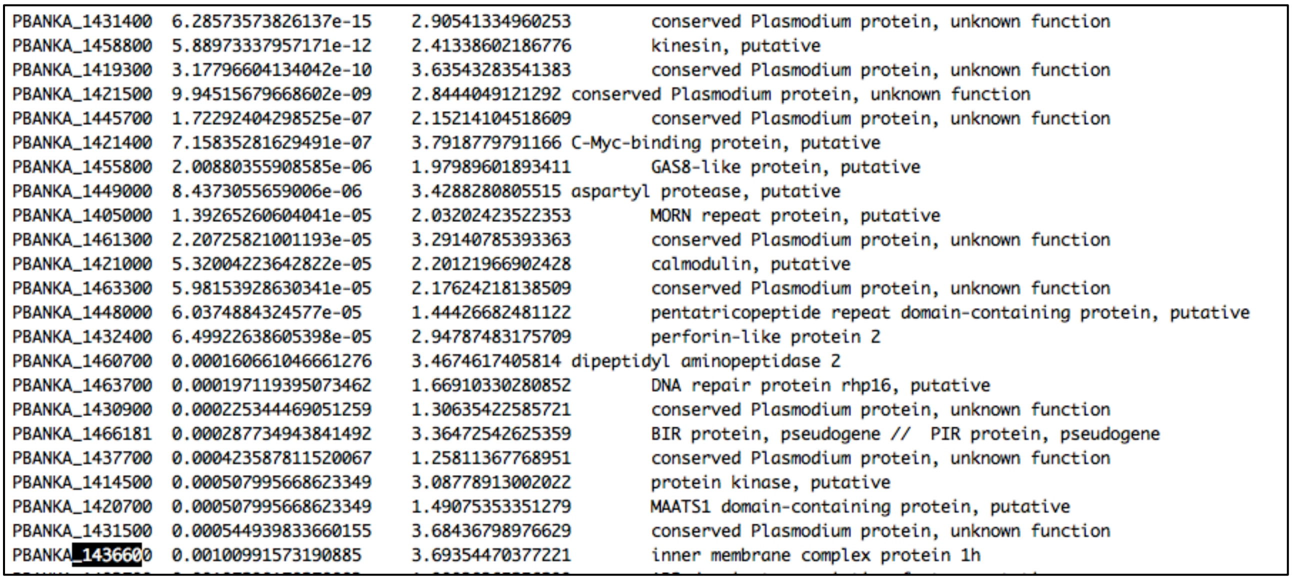
How many genes are more highly expressed in each condition? (use `| wc -l` command!)

Do you notice any genes that come up in the analysis that seem interesting?

Now let's compare this list to the one we made earlier using Artemis. What are the differences? Is the list similar to your first list of differentially expressed genes? Remember, with Artemis we only looked one replicate. Does adding more make a difference? Which results would you trust more (this or the ratio in the list before)?

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

gene_id   q-value   fold expression   product



```
PBANKA_1431400  6.28573573826137e-15    2.90541334960253        conserved Plasmodium protein, unknown function
PBANKA_1458800  5.88973337957171e-12    2.41338602186776        kinesin, putative
PBANKA_1419300  3.17796604134042e-10    3.63543283541383        conserved Plasmodium protein, unknown function
PBANKA_1421500  9.94515679668602e-09    2.8444049121292 conserved Plasmodium protein, unknown function
PBANKA_1445700  1.72292404298525e-07    2.15214104518609        conserved Plasmodium protein, unknown function
PBANKA_1421400  7.15835281629491e-07    3.7918779791166 C-Myc-binding protein, putative
PBANKA_1455800  2.00880355908585e-06    1.97989601893411        GAS8-like protein, putative
PBANKA_1449000  8.4373055659006e-06     3.4288280805515 aspartyl protease, putative
PBANKA_1405000  1.39265260604041e-05    2.03202423522353        MORN repeat protein, putative
PBANKA_1461300  2.20725821001193e-05    3.29140785393363        conserved Plasmodium protein, unknown function
PBANKA_1421000  5.32004223642822e-05    2.20121966902428        calmodulin, putative
PBANKA_1463300  5.98153928630341e-05    2.17624218138509        conserved Plasmodium protein, unknown function
PBANKA_1448000  6.0374884324577e-05     1.44426682481122        pentatricopeptide repeat domain-containing protein, putative
PBANKA_1432400  6.49922638605398e-05    2.94787483175709        perforin-like protein 2
PBANKA_1460700  0.000160661046661276    3.4674617405814 dipeptidyl aminopeptidase 2
PBANKA_1463700  0.000197119395073462    1.66910330280852        DNA repair protein rhp16, putative
PBANKA_1430900  0.000225344469051259    1.30635422585721        conserved Plasmodium protein, unknown function
PBANKA_1466181  0.000287734943841492    3.36472542625359        BIR protein, pseudogene //  PIR protein, pseudogene
PBANKA_1437700  0.000423587811520067    1.25811367768951        conserved Plasmodium protein, unknown function
PBANKA_1414500  0.000507995668623349    3.08778913002022        protein kinase, putative
PBANKA_1420700  0.000507995668623349    1.49075353351279        MAATS1 domain-containing protein, putative
PBANKA_1431500  0.000544939833660155    3.68436798976629        conserved Plasmodium protein, unknown function
PBANKA_1436600  0.00100991573190885     3.69354470377221        inner membrane complex protein 1h
```

The list is a bit different as we are looking now at the complete genome. But if you would do a `| grep _14` to look only at chromosome 14, you would see that there are not too many differences.

This does not necessarily help us to find the function of the gene we knocked out… let's write out the differential expressed genes, with a log fold change of 1, and do a GO enrichment in PlasmoDB.

```
$ cut -f1,2,4,5 kallisto.results | awk -F "\t"  '$3 < 0.01 &&
$4  > 1' | cut -f 1 > UpRegulated.txt
```

```
$ cut -f1,2,4,5 kallisto.results | awk -F "\t"  '$3 < 0.01 &&
$4  < 1’ | cut -f 1 > DownRegulated.txt
```

GO enrichment

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.

However, the association of GO terms to genes depends 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).

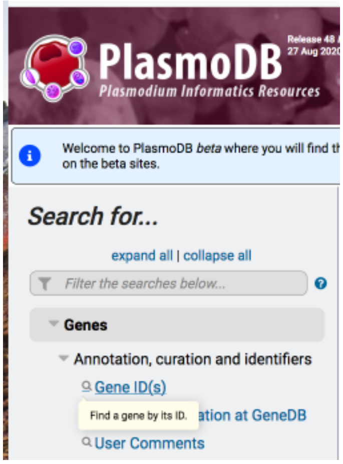
Change the directory and have a look at the files:

```
$ cd GO
$ ls
```

If you have finished the exercise from above, you have generated the Up and DownRegulated.txt file. Copy the file to this directory with:

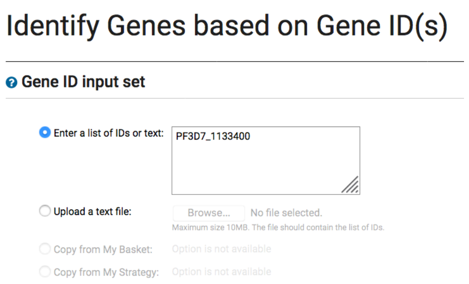
```
$ cp ../UpRegulated.txt .
```

There are methods like David to do GO enrichment for human. For parasites we would encourage to also visit VEuPathDB. We can show that if you want.


PlasmoDB
Plasmodium Informatics Resources
Release 48
27 Aug 2020

Welcome to PlasmoDB *beta* where you will find th
on the beta sites.

Search for...
expand all | collapse all
Filter the searches below...
Genes
Annotation, curation and identifiers
Gene ID(s)
Find a gene by its ID.
ation at GeneDB
User Comments



Identify Genes based on Gene ID(s)

Gene ID input set

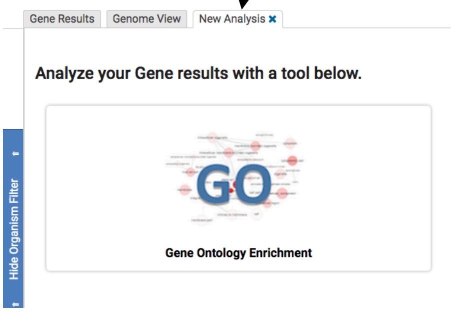
● Enter a list of IDs or text: PF3D7_1133400

○ Upload a text file: Browse... No file selected.
Maximum size 10MB. The file should contain the list of IDs.

○ Copy from My Basket: Option is not available

○ Copy from My Strategy: Option is not available

New analysis



Analyze your Gene results with a tool below.

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 UpRegulated.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 the down regulated genes? 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!*

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 replicated data. 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. By determining how much variability there is in the sample preparation and sequencing reactions we can better assess how highly 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 high levels, while deeper sequencing will improve power to detect differential expression for genes which are expressed at low 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 say 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 (~5500 genes for *Plasmodium*), 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.

Alternative software

If you have a good quality genome and genome annotation such as for model organisms e.g. human, mouse, *Plasmodium*, I would recommend mapping to the transcriptome for determining transcript abundance. This is even more relevant if you have variant transcripts per gene as you need a tool which will do its best to determine which transcript is really expressed. As well as Kallisto (Bray et al. 2016; PMID: 27043002), there is eXpress (Roberts & Pachter, 2012; PMID: 23160280) which will do this.

Alternatively you can map to the genome and then call abundance of genes, essentially ignoring variant transcripts. This is more appropriate where you are less confident about the genome annotation and/or you don't have variant transcripts because your organism rarely makes them or they are simply not annotated. Tophat2 (Kim et al., 2013; PMID: 23618408), HISAT2 (Pertea et al. 2016; PMID: 27560171), STAR (Dobin et al., 2013; PMID: 23104886) and GSNAP (Wu & Nacu, 2010; PMID: 20147302) are all splice-aware RNA-seq read mappers appropriate for this task. You then need to use a tool which counts the reads overlapping each gene model. HTSeq (Anders et al., 2015; PMID: 25260700) is a popular tool for this purpose.

There are a variety of programs for detecting differentially expressed genes from tables of RNA-seq read counts. DESeq2 (Love et al., 2014; PMID: 25516281), EdgeR (Robinson et al., 2010; PMID: 19910308) and BaySeq (Hardcastle & Kelly, 2010; PMID: 20698981) are good examples.

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 could annotate your genes with functional terms from GO using for instance Blast2GO (Conesa et al., 2005; PMID: 16081474) and then use TopGO (Alexa et al., 2005; PMID: 16606683) to determine whether any particular sorts of genes occur more than expected in your differentially expressed genes.

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.