1 Answers

To run the commands alongside the answers you must first have run *all* of the tutorial commands for that section and generated the output files they reference.

1.1 Tutorial sections

- Introducing the tutorial dataset
- Mapping RNA-Seq reads to the genome using HISAT2
- Visualising transcriptomes with IGV
- Transcript quantification with Kallisto
- Identifying differentially expressed genes with Sleuth
- Interpreting the results
- Normalisation

Let's go to our `data` directory.

```
[ ]:  cd data
```

1.2 Introducing the tutorial dataset

1.2.1 Q1: Why is there more than one FASTQ file per sample?

There are **2** FASTQ files for each sample e.g. MT1_**1**.fastq and MT1_**2**.fastq. This is because this was **paired-end** sequence data.

```
[ ]:  ls MT1*.fastq
```

```
[ ]:  ls MT1*.fastq | wc -l
```

With Illumina paired-end sequencing, you will have a fragment (hopefully longer than your reads) which is sequenced at both ends. This means that there will be a "left" read (sometimes known as *r1*) and its corresonding "right" read (sometimes known as *r2*). These are indicated by the */1* and */2* at the end of the read name in the ****_1.fastq and _2****.fastq files respectively.

So, the left read header _@HS18_08296:8:1101:1352:48181#4_ in **MT1_1.fastq** has the */1* suffix:

@HS18_08296:8:1101:1352:48181#4/1

And, the corresponding right read header in **MT1_2.fastq** has the */2* suffix:

@HS18_08296:8:1101:1352:48181#4/2

1.2.2 Q2: How many reads are there for MT1?

There are **2.5 million** reads for MT1.

Ideally, you should count the reads in both files. This is because sometimes we have singletons (reads without a mate) after preprocessing steps such as trimming.

Our reads look like:

```
@HS18_08296:8:1101:1352:48181#4/2
ATCCGCCNANTTTNNNNATATAATTANNNGNAANNAANNNNAATNACANNNATTNNNNTAGNANNNGNNAGTNNACAAGGNTNNNNNNNAAAGNI
+
9AABE8A!D!DFA!!!!EE@CFCD@B!!!D!F6!!EE!!!!EE4!F5E!!!BEA!!!!B@6!A!!!D!!FD'!!C+D@@*!B!!!!!!!B>DE!
```

With the FASTQ format there are four lines per read. So, as long as our files are not truncated we can count the number of lines and divide them by four.

```
[ ]: wc -l MT1_1.fastq
```

So, we can then divide this by four to give us 1.25 million. We will get the same for our r2 reads:

```
[ ]: wc -l MT1_2.fastq
```

So, we have 1.25 million *read pairs* or 2.5 million (1.25 x 2) *reads* for our MT1 sample.

You may also have thought "aha, I can use the **@** which is at the start of the header"…

```
[ ]: grep -c '^@' MT1_1.fastq
```

```
[ ]: grep -c '^@' MT1_2.fastq
```

But, wait, there are 1,343,714 left reads and 1,250,000 right reads…can that be right…*no*.

Take a closer look at the quality scores on the fourth line…they also contain **@**. This is because the quality scores are Phred+33 encoded. For more information on quality score encoding see our Data Formats and QC tutorial or go here.

Instead, we can use the earlier information about the left and right read suffixes: **/1** and **/2**. This can be with two commands:

```
[ ]: grep -c '/1$' MT1_1.fastq
```

```
[ ]: grep -c '/2$' MT1_2.fastq
```

Or, with only one command:

```
[ ]: grep -c '/[12]$' MT1*.fastq
```

*Note: Don't forget to use the `-c` option for `grep` to count the occurences and `$` to make sure you're only looking for \1 or \2 at the **end of the line**.*

1.3 Mapping RNA-Seq reads to the genome using HISAT2

1.3.1 Map, convert (SAM to BAM), sort and index using the reads from the MT2 sample.

You can do this with several, individual steps:

```
hisat2 --max-intronlen 10000 -x PccAS_v3_hisat2.idx -1 MT2_1.fastq -2 MT2_2.
  ↪fastq -S MT2.sam
```

```
samtools view -b -o MT1.bam MT1.sam
```

```
samtools sort -o MT1_sorted.bam MT1.bam
```

```
samtools index MT1_sorted.bam
```

Or, you can do it in one step:

```
hisat2 --max-intronlen 10000 -x PccAS_v3_hisat2.idx -1 MT2_1.fastq -2 MT2_2.
  ↪fastq \
| samtools view -b - \
| samtools sort -o MT2_sorted.bam - \
&& samtools index MT2_sorted.bam
```

Or, you could write a for loop to run the command above for all five of your samples:

```
for r1 in *_1.fastq
do
   sample=${r1/_1.fastq/}
   echo "Processing sample: "$sample
   hisat2 --max-intronlen 10000 -x PccAS_v3_hisat2.idx -1 $sample"_1.fastq" -2
  ↪$sample"_2.fastq" \
   | samtools view -b - \
   | samtools sort -o $sample"_sorted.bam" - \
   && samtools index $sample"_sorted.bam"
done
```

For more information on how this loop works, have a look at our Unix tutorial and Running commands on multiple samples.

1.3.2 Q1: How many index files were generated when you ran `hisat2-build`?

There are **8** HISAT2 index files for our reference genome.

```
ls data/*.ht2
```

```
ls data/*.ht2 | wc -l
```

1.3.3   Q2: What was the *overall alignment rate* for each of the MT samples (MT1 and MT2) to the reference genome?

The *overall alignment rate* for MT1 was **94.15%** and **91.69%** for MT2.

```
1250000 reads; of these:
  1250000 (100.00%) were paired; of these:
    105798 (8.46%) aligned concordantly 0 times
    329468 (26.36%) aligned concordantly exactly 1 time
    814734 (65.18%) aligned concordantly >1 times
    ----
    105798 pairs aligned concordantly 0 times; of these:
      1797 (1.70%) aligned discordantly 1 time
    ----
    104001 pairs aligned 0 times concordantly or discordantly; of these:
      208002 mates make up the pairs; of these:
        146250 (70.31%) aligned 0 times
        19845 (9.54%) aligned exactly 1 time
        41907 (20.15%) aligned >1 times
94.15% overall alignment rate
```

```
1250000 reads; of these:
  1250000 (100.00%) were paired; of these:
    139557 (11.16%) aligned concordantly 0 times
    483583 (38.69%) aligned concordantly exactly 1 time
    626860 (50.15%) aligned concordantly >1 times
    ----
    139557 pairs aligned concordantly 0 times; of these:
      4965 (3.56%) aligned discordantly 1 time
    ----
    134592 pairs aligned 0 times concordantly or discordantly; of these:
      269184 mates make up the pairs; of these:
        207729 (77.17%) aligned 0 times
        28836 (10.71%) aligned exactly 1 time
        32619 (12.12%) aligned >1 times
91.69% overall alignment rate
```

Note: If a read pair is concordantly aligned it means both reads in the pair align with the same chromosome/scaffold/contig, the reads are aligned in a proper orientation (typically —> <—) and that the reads have an appropriate insert size.

1.3.4 Q3: How many MT1 and MT2 reads were not aligned to the reference genome?

146,250 reads (5.85%) and **207,729 reads (8.31%)** did not align to the reference genome for MT1 and MT2 respectively.

Here is a brief summary of what the HISAT2 summary tells us for our MT2 sample and how we can tell which of the summary lines gives us this information:

- We have **1,250,000 read pairs** or **2,500,000 reads** (*2 x 1,250,000 pairs*)

  ```
  1250000 reads; of these:
  ```

- All of our reads (100%) are paired - i.e. no reads without their mate

  ```
  1250000 (100.00%) were paired; of these:
  ```

- 1,110,443 pairs (88.84%) align concordantly one (38.69%) or more (50.15%) times

  ```
  483583 (38.69%) aligned concordantly exactly 1 time
  626860 (50.15%) aligned concordantly >1 times
  ```

- 139,557 pairs (11.16%) or 279,114 reads (2 x 139,557) did not align *concordantly* anywhere in the genome

  ```
  139557 (11.16%) aligned concordantly 0 times
  ```

- Of those 139,557 pairs, 4,965 pairs align *discordantly* (3.56% of the 139,557 pairs)

  ```
  139557 pairs aligned concordantly 0 times; of these:
   4965 (3.56%) aligned discordantly 1 time
  ```

- This leave us with 134,592 pairs (139,557 - 4,965) where both reads in the pair do not align to the genome (concordantly or discordantly)

  ```
  134592 pairs aligned 0 times concordantly or discordantly; of these:
  ```

- Of those 269,184 reads (2 x 134,592) we have 61,455 reads (22.83%) which align to the genome without their mate

  ```
  269184 mates make up the pairs; of these:
  ...
  28836 (10.71%) aligned exactly 1 time
  32619 (12.12%) aligned >1 times
  ```

- Leaving us with a **91.69% overall alignment rate**

```
91.69% overall alignment rate
```

- That means **207,729 (8.31%)** of the **2,500,000 reads** (or 77.17% of the unaligned pairs) do not align anywhere in the genome

```
207729 (77.17%) aligned 0 times
```

1.4   Visualising transcriptomes with IGV

1.4.1   Q1: How many CDS features are there in "PCHAS_1402500"?

There are **8** CDS features in PCHAS_1402500. You can get this in several ways:

Count the number of exons/CDS features in the gene annotation.
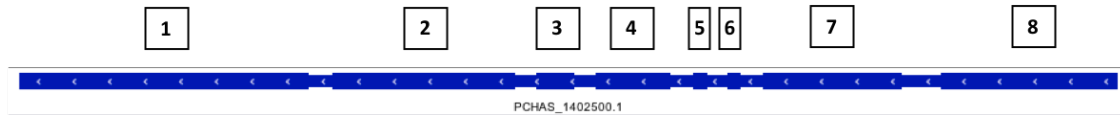
Count the number of CDS features in the GFF file.
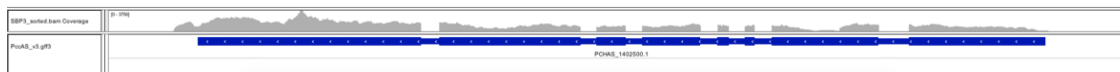
First, get all of the CDS features for PCHAS_1402500.

```
[ ]: grep -E "CDS.*PCHAS_1402500" PccAS_v3.gff3
```

Then count the number of the CDS features for PCHAS_1402500.

```
[ ]: grep -cE "CDS.*PCHAS_1402500" PccAS_v3.gff3
```

1.4.2   Q2: Does the RNA-seq mapping agree with the gene model in blue?

Yes. The peaks of the coverage tracks correspond to the annotated exon/CDS features.

1.4.3   Q3: Do you think this gene is differentially expressed and is looking at the coverage plots alone a reliable way to assess differential expression?

Possibly. But, you can't tell differential expression by the counts alone as there may be differences in the sequencing depths of the samples.

1.5 Transcript quantification with Kallisto

1.5.1 Use kallisto quant four more times, for the MT2 sample and the three SBP samples.

You can run the individual `kallisto quant` commands as you did for MT1 for each of the remaining samples:

```
[ ]: kallisto quant -i PccAS_v3_kallisto -o MT2 -b 100 MT2_1.fastq MT2_2.fastq
     kallisto quant -i PccAS_v3_kallisto -o SBP1 -b 100 SBP1_1.fastq SBP1_2.fastq
     kallisto quant -i PccAS_v3_kallisto -o SBP2 -b 100 SBP1_2.fastq SBP2_2.fastq
     kallisto quant -i PccAS_v3_kallisto -o SBP3 -b 100 SBP1_3.fastq SBP3_2.fastq
```

Or, you can write a `for` loop which will run `kallisto quant` on all of the samples:

```
[ ]: for r1 in *_1.fastq
     do
        echo $r1
        sample=${r1/_1.fastq/}
        echo "Quantifying transcripts for sample: "$sample
        kallisto quant -i PccAS_v3_kallisto -o $sample -b 100 $sample'_1.fastq'␣
        ↪$sample'_2.fastq'
     done
```

For more information on how this loop works, have a look at our Unix tutorial and Running commands on multiple samples.

1.5.2 Q1: What *k*-mer length was used to build the Kallisto index?

A *k*-mer length of **31** was used.

Look at the output from `kallisto index`:

`[build] k-mer length: 31`

Or, look for the `-k` or `--kmer-size` option in the `kallisto index` usage:

```
[ ]: kallisto index
```

1.5.3 Q2: How many transcript sequences are there in *PccAS_v3_transcripts.fa*?

There are **5177** transcript sequences.

Look at the output from `kallisto quant`:

`[index] number of targets: 5,177`

Or, look for **n_targets** in one of the *run_info.json* files:

```
[ ]: cat MT1/run_info.json
```

Or, you can run a grep on the transcript FASTA file and count the number of header lines:

```
[ ]: grep -c ">" PccAS_v3_transcripts.fa
```

1.5.4 Q3: What is the transcripts per million (TPM) value for PCHAS_1402500 in each of the samples?

Sample	Transcripts Per Million (TPM)
MT1	2342.23
MT2	1354.42
SBP1	2295.24
SBP2	3274.98
SBP3	2536.17

You can look at each of the individual abundance files:

```
[ ]: grep "^PCHAS_1402500" MT1/abundance.tsv
     grep "^PCHAS_1402500" MT2/abundance.tsv
     grep "^PCHAS_1402500" SBP1/abundance.tsv
     grep "^PCHAS_1402500" SBP2/abundance.tsv
     grep "^PCHAS_1402500" SBP3/abundance.tsv
```

Or you can use a recursive grep:

```
[ ]: grep -r "^PCHAS_1402500" .
```

Or you can use a loop:

```
[ ]: for r1 in *_1.fastq
     do
        sample=${r1/_1.fastq/}
        echo $sample
        grep PCHAS_1402500 $sample'/abundance.tsv'
     done
```

For more information on how this loop works, have a look at our Unix tutorial and Running commands on multiple samples.

1.5.5 Q4: Do you think PCHAS_1402500 is differentially expressed?

Probably not. We would need to run statistical tests to really be sure though.

1.6 Identifying differentially expressed genes with Sleuth

1.6.1 Q1: Is our gene from earlier, PCHAS_1402500, significantly differentially expressed?

No.

Look at the transcript table.

And the transcript view.
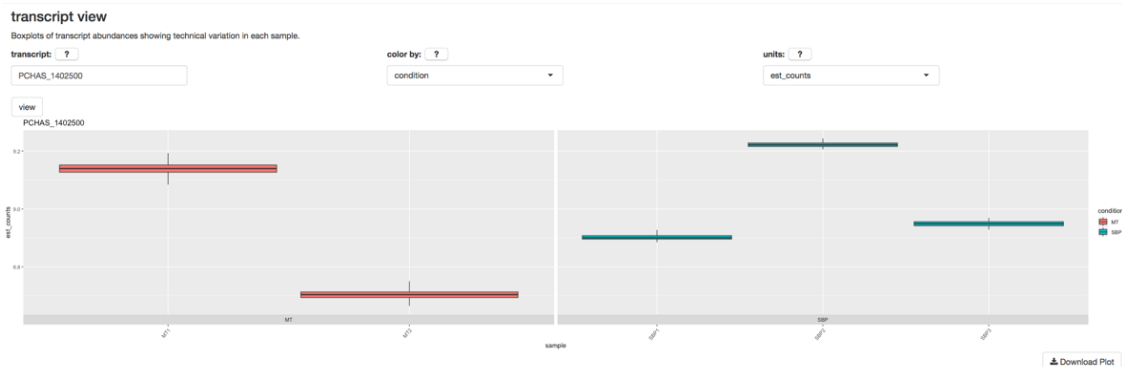
Although this gene looked like it was differentially expressed from the plots in IGV, our test did not show it to be so (q-value > 0.05). This might be because some samples tended to have more reads, so based on raw read counts, genes generally look up-regulated in the SBP samples.

Alternatively, the reliability of only two biological replicates and the strength of the difference between the conditions was not sufficient to be statistically convincing. In the second case, increasing the number of biological replicates would give us more confidence about whether there really was a difference.

In this case, it was the lower number of reads mapping to MT samples that mislead us in the IGV view. Luckily, careful normalisation and appropriate use of statistics saved the day!

```
grep PCHAS_1402500 kallisto.results | cut -f1,4
```

1.7 Interpreting the results

1.7.1 Q1: How many genes are more highly expressed in the SBP samples?

127. We can use `awk` to filter our Kallisto/sleuth results and `wc -l` to count the number of lines returned.

```
[ ]: awk -F"\t" '$4 < 0.01 && $5 > 0' kallisto.results | wc -l
```

1.7.2 Q2: How many genes are more highly expressed in the MT samples?

169. We can use `awk` to filter our Kallisto/sleuth results and `wc -l` to count the number of lines returned.

```
[ ]: awk -F"\t" '$4 < 0.01 && $5 < 0' kallisto.results | wc -l
```

1.7.3 Q3: Do you notice any particular genes that come up in the analysis?

Genes from the ***cir*** family are upregulated in the MT samples.

To get this we must first find out which genes (or gene descriptions) are seen most often in the genes which are more highly expressed in our SBP samples.

```
[ ]: awk -F"\t" '$4 < 0.01 && $5 > 0 {print $2}' kallisto.results | sort | uniq -c |␣
     ↪sort -nr
```

Then, we summarise the genes more highly expressed in our MT samples.

```
[ ]: awk -F"\t" '$4 < 0.01 && $5 < 0 {print $2}' kallisto.results | sort | uniq -c |␣
     ↪sort -nr
```

Perhaps the CIR proteins are interesting. There are only 2 CIR proteins upregulated in the SBP samples and 25 CIR in the MT samples.

The ***cir*** family is a large, malaria-specific gene family which had previously been proposed to be involved in immune evasion (Lawton et al., 2012). Here, however, we see many of these genes upregulated in a form of the parasite which seems to cause the immune system to better control the parasite. This suggests that these genes interact with the immune system in a subtler way, preventing the immune system from damaging the host.

1.8 Normalisation

1.8.1 How we got the information to help the questions

To answer the questions you needed the following for each sample:

- Number of reads assigned to PCHAS_1402500
- Length of exons in PCHAS_1402500 (bp)
- Total number of reads mapping
- Total RPK

First, take a quick look at the first five lines of the `abundance.tsv` for MT1.

```
[ ]: head -5 MT1/abundance.tsv
```

There are five columns which give us information about the transcript abundances for our MT1 sample.

Column	Description
target_id	Unique transcript identifier
length	Number of bases found in exons.
eff_length	*Effective length*. Uses fragment length distribution to determine the effective number of positions that can be sampled on each transcript.
est_counts	*Estimated counts*. This may not always be an integer as reads which map to multiple transcripts are fractionally assigned to each of the corresponding transcripts.
tpm	*Transcripts per million*. Normalised value accounting for length and sequence depth bias.

First, look for your gene of interest, **PCHAS_1402500**. Run this as a loop to `grep` the information for all five samples.

```
[ ]: for r1 in *_1.fastq
do
   sample=${r1/_1.fastq/}
   echo $sample
   grep PCHAS_1402500 $sample'/abundance.tsv'
done
```

Now you have the length (eff_length) and counts (est_counts) for PCHAS_1402500 for each of your samples. Next, you need to get the total number of reads mapped to each of your samples. You can use a loop to do this.

In the loop below `samtools flagstat` gives you the number of mapped paired reads (reads with

itself and mate mapped) and those where one read mapped but its mate didn't (singletons). If then uses `grep` to get the relevant lines and `awk` to add the mapped paired and singleton read totals together.

```
[ ]: for r1 in *_1.fastq
do
   sample=${r1/_1.fastq/}

   total=` samtools flagstat $sample'_sorted.bam' | \
           grep 'singletons\|with itself and mate mapped' | \
           awk 'BEGIN{ count=0} \
                {count+=$1} \
                END{print count}'`

   echo -e "$sample\t$total"
done
```

Finally, to calculate the TPM values, you need the total RPK for each of your samples. Again we use a loop. Notice the use of `NR>2` in the `awk` command which tells it to skip the two header lines at the start of the file. You will also notice that we divide the eff_length by 1,000 so that it's in kilobases.

```
[ ]: for r1 in *_1.fastq
do
   sample=${r1/_1.fastq/}

   awk -F"\t" -v sample="$sample" \
       'BEGIN{total_rpk=0;} \
       NR>2 \
       { \
         rpk=$4/($3/1000); \
         total_rpk+=rpk \
       } \
       END{print sample"\t"total_rpk}' $sample'/abundance.tsv'
done
```

1.8.2   Q1: Using the `abundance.tsv` files generated by Kallisto and the information above, calculate the RPKM for PCHAS_1402500 in each of our five samples.

Sample	Per million scaling factor	Reads per million (RPM)	Per kilobase scaling factor	RPKM
MT1	2.353750	1079.528	3.697	**292**
MT2	2.292271	1479.581	3.709	**398**
SBP1	2.329235	6270.170	3.699	**1695**
SBP2	2.187718	7908.652	3.696	**2140**
SBP3	2.163979	6767.949	3.699	**1830**

1.8.3 Q2: Using the `abundance.tsv` files generated by Kallisto and the information above, calculate the TPM for PCHAS_1402500 in each of our five samples.

Sample	Per kilobase scaling factor	Reads per kilobase (RPK)	TPM
MT1	3.697	687.30	**2342**
MT2	3.709	914.50	**1354**
SBP1	3.699	3947.87	**2295**
SBP2	3.696	4681.81	**3275**
SBP3	3.699	3959.78	**2536**

1.8.4 Q3: Do these match the TPM values from Kallisto?

Yes.

Well, almost. They may be a couple out because we rounded up to make the calculations easier.

1.8.5 Q4: Do you think PCHAS_1402500 is differentially expressed between the MT and SBP samples?

Probably not.

If we were to look at only the counts and RPKM values then it appears there is an 8 fold difference between the MT and SBP samples. However, when we look at the TPM values, they are much closer and so differential expression is less likely.