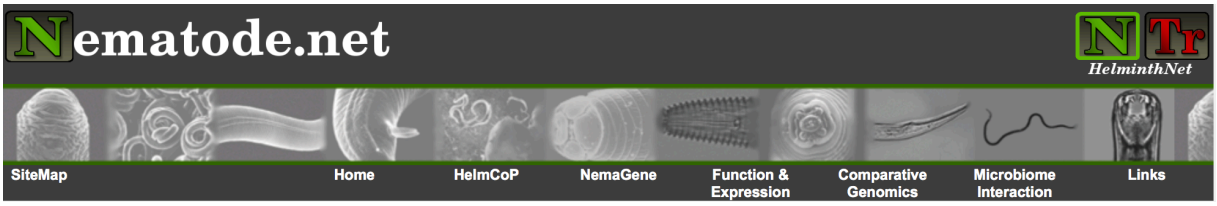
Washington University in St. Louis
School of Medicine
McDonnell Genome Institute



Bioinformatics Workshop for Helminth Genomics (2015)

Section 2: Transcriptome

Sponsors:

The Elizabeth H. and James S. McDonnell III Genome Institute, Washington University, Campus Box 8501
4444 Forest Park Blvd., Saint Louis, MO 63108 **Phone:** 314.286.1800 **Fax:** 314.286.1810 **Web:** genome.wustl.edu

Table of contents – Curriculum

Section 2: Transcriptome

Section 2: Transcriptome
Module 0: RNA isolation to sequence production

1) Experimental design
2) Library construction & sequencing

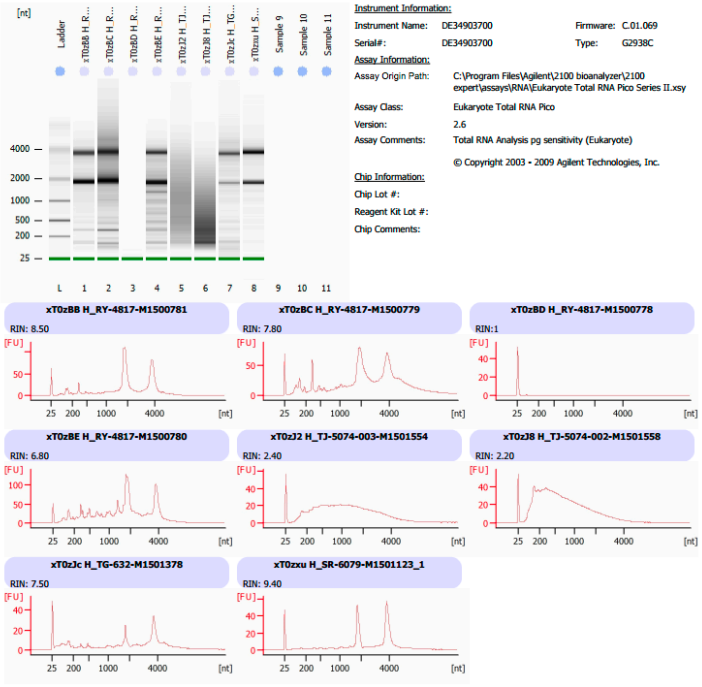
Experimental design

- What's the purpose?
  - Gene discovery
  - Differential expression
- More reads = more confidence
  - Depth
    - Depends on genome size, coding features, etc.
    - More for discovery of novel features, low expression genes
  - Replicates
    - Biological, not technical
    - More is better for differential expression, 3 per condition
- Collect appropriate meta-data when you collect your RNA
  - Strain/isolate/batch
  - Sex, age, patency
  - Treatments

Resource: https://genome.ucsc.edu/ENCODE/protocols/dataStandards/ENCODE_RNAseq_Standards_V1.0.pdf, http://gkno2.tumblr.com/post/24629975632/thinking-about-rna-seq-experimental-design-for

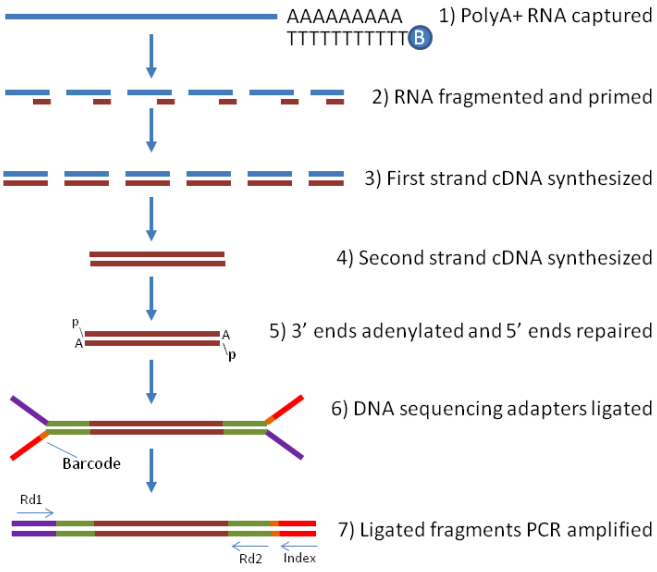
Quality control of RNA sample

- Nanodrop quantitation
  - Standard equipment
  - Peaks at particular absorbance range can signal contamination
  - Can't distinguish between DNA, RNA, free nucleotides
- Qubit fluorometric quantitation
  - Use separate kits to measure RNA, DNA and protein individually
- Agilent bioanalyzer to assess integrity
  - RNA integrity number (RIN)

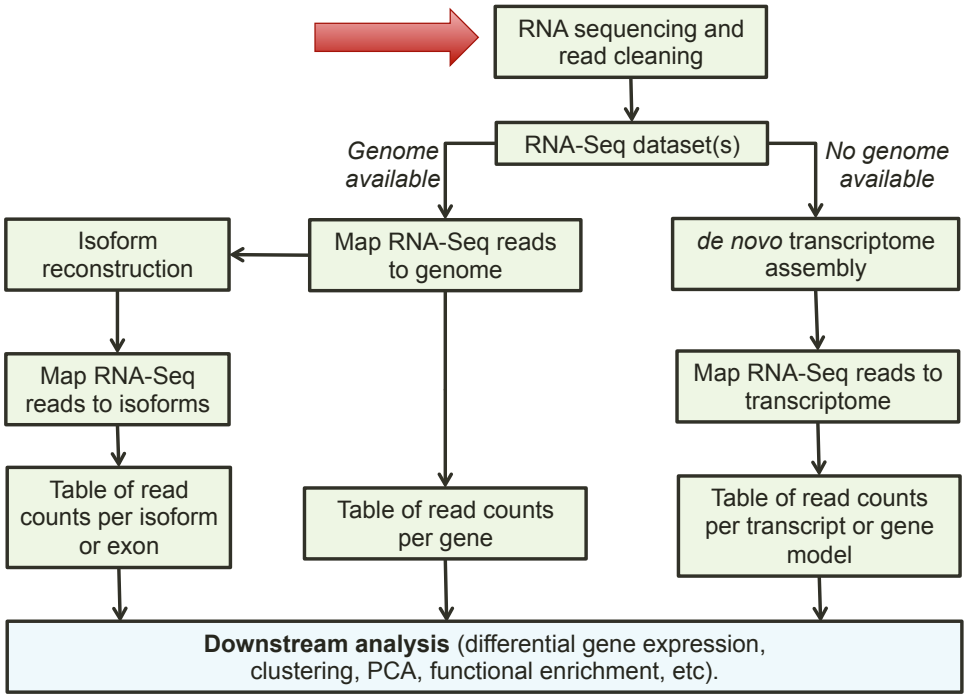


Production of Illumina RNAseq data

- Assess quality & concentration
- DNAse treatment
- Poly(A) selection
- Fragmentation
- cDNA synthesis
  - oligo(dT) & random hexamers
- Library preparation
- Sequencing



RNA-seq analysis overview



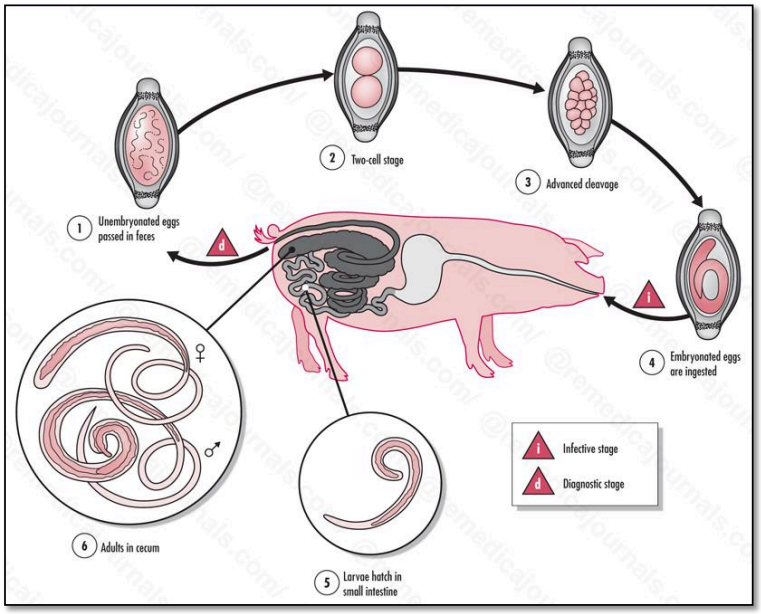
Read pre-processing and filtering: a very stringent protocol

1) Adapter removal
2) Quality trimming & filtering
3) Contaminant filtering

Resource: http://www.nature.com/nprot/journal/v8/n8/pdf/nprot.2013.084.pdf, specifically Box 1

Our "test" dataset

Life cycle of *Trichuris suis*



- **Larval**
  - 10 days post inoculation (dpi), L2
  - 16 dpi, L3
  - 17 dpi, L3
  - 21 dpi, L4
- **Adult**
  - 42 dpi, L5
  - Adult rep1
  - Adult rep2

Our "test" dataset

300-500bp fragment

	L2_10d	L3_16d	L3_17d	L4_21d	L5_42d	L5_r163	L5_r179	Total
Total raw pairs	43,592,929	54,459,409	47,371,505	58,231,629	55,800,467	32,809,672	41,902,924	**334,168,535**
Downsampled raw pairs	4,435,622	5,511,063	4,817,349	5,891,002	5,644,329	3,337,590	4,258,806	**33,895,761**

- Counting reads in a bam file

  ```
  samtools view -b -c input.bam
  ```

  - Divide by 2 to get pairs!
- Downsampling:

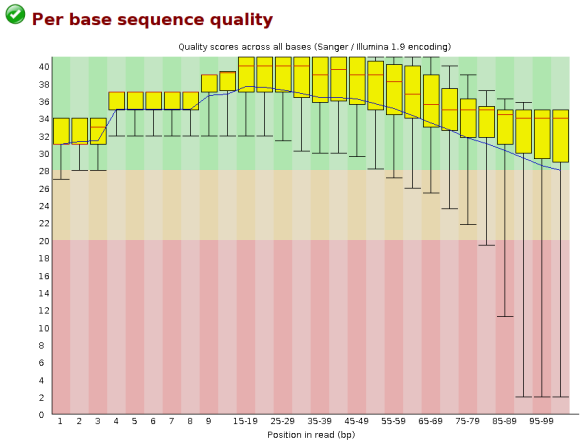
  ```
  samtools view -b -s XX.XX -o output.bam input.bam
  ```

  - -b: input is bam format
  - -s: random down-sampling, integer before the decimal is seed for random number generator, after the decimal is the % reads to maintain
  - -o: output file name
- Convert bam → fastq as before

Resource: http://www.htslib.org/doc/samtools.html

Adapter detection

- Use fastqc to identify any adapter sequences that may need to be clipped

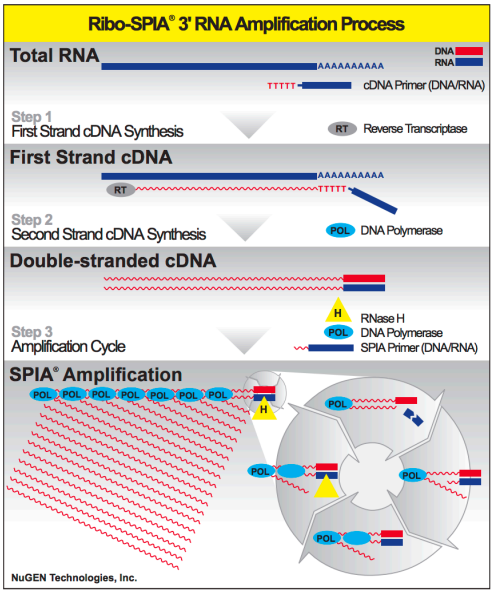


[WARN] Overrepresented sequences

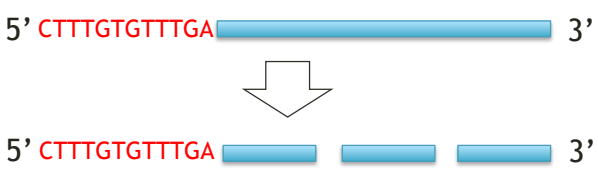
Sequence	Count	Percentage	Possible Source
CTTTGTGTTTGATTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT	116516	0.13364093979553426	No Hit
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	90699	0.10402948606642146	No Hit

Resource: http://www.bioinformatics.babraham.ac.uk/projects/fastqc/

NuGEN Ovation RNAseq System V2



- Single Primer Isothermal Amplification protocol used in cDNA synthesis
  - SPIA adapters linked to primers
- Fragmentation following cDNA synthesis, so most reads won't have SPIA



Resource: http://www.nugen.com/sites/default/files/M01114_v4.1%20-%20User%20Guide,%20Ovation%20RNA%20Amplification%20System%20V2.pdf

Adapter detection

- Checking for adapters in your file:

```
grep –B 1 –A 2 –colour "^CTTGTGTTTGA" L2_10d.1.raw.fastq
```

- To count sequences with an adapter

```
grep –c  "^CTTGTGTTTGA" L2_10d.1.raw.fastq
```

Resource: http://linuxcommand.org/man_pages/grep1.html

Adapter removal

- Tips:
  - Trimmomatic doesn't work well for short adapter sequences
  - clipping multiple adapters in one pass may not work well

- Other options for adapter trimming:
  - Flexbar: http://sourceforge.net/p/flexbar/wiki/Manual/
    - Adapter detection & removal
    - Barcode detection, removal and read binning
    - Filtering reads with uncalled bases
    - Quality trimming and filtering
    - Length trimming / filtering
  - Cutadapt: https://pypi.python.org/pypi/cutadapt/
  - FASTX-Toolkit: http://hannonlab.cshl.edu/fastx_toolkit/seq
  - Seq_crumbs toolkit: https://bioinf.comav.upv.es/seq_crumbs/

Removing SPIA adapters with Flexbar

- Command

```
flexbar --adapters Adapter.fasta --adapter-trim-end LEFT --min-read-length 60 --reads L2_10d.1.raw.fastq --reads2 L2_10d.2.raw.fastq --target L2_10d --format=sanger --adapter-min-overlap 7
```

- Result:
  - Clip adapters
  - Filter reads with uncalled bases
  - Remove any reads <60bp

```
2. smcnulty@blade14-2-10: /gscmnt/gc2529/mitrevalab/smcnulty/INFOR...
Adapter removal statistics
==========================
Adapter:            Overlap removal:    Full length:
SPIA                1426360             1062690
SPIARC              808949              635324

Min, max, mean and median adapter overlap: 7 / 13 / 11 / 12


Output file statistics
======================
Read file:              L2_10d.spia_1.fastq
  written reads         3991748
  skipped short reads   239258
Read file 2:            L2_10d.spia_2.fastq
  written reads         3991748
  too short reads       249235
Single read file:       L2_10d.spia_1_single.fastq
  written reads         194531
Single read file 2:     L2_10d.spia_2_single.fastq
  written reads         184554


Filtering statistics
====================
Processed reads                   8871244
  skipped due to uncalled bases     20170   (12773 uncalled in 10085 pairs)
  short prior adapter removal           0
  finally skipped short reads      488493
Discarded reads overall            508663
Remaining reads                   8362581   (94% of input reads)


Flexbar completed adapter removal.
```

Resource: http://sourceforge.net/p/flexbar/wiki/Manual/

Quality trimming & filtering with Trimmomatic

- Command:
  - `java -jar ~/bin/trimmomatic-0.33.jar PE -phred33 L2_10d.spia_1.fastq L2_10d.spia_2.fastq L2_10d.1.fb-tm.fastq L2_10d.1.junk.fastq L2_10d.2.fb-tm.fastq L2_10d.2.junk.fastq ILLUMINACLIP:Adapters.fasta:2:30:10 SLIDINGWINDOW:5:20 LEADING:20 TRAILING:20 MINLEN:60`

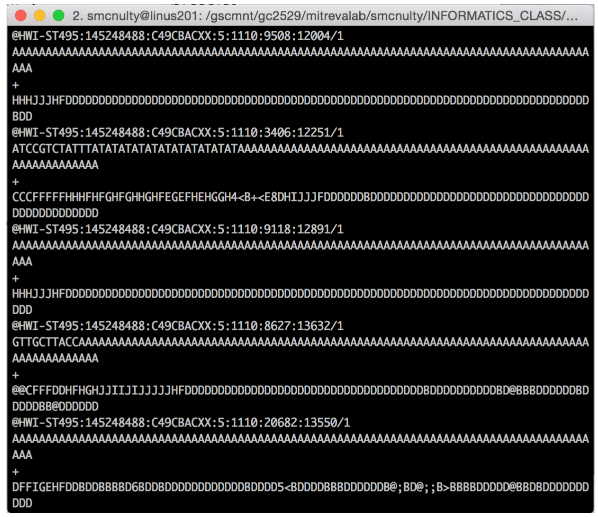
- Result
  - Clipping any remaining Illumina sequencing adapters
  - Clipping any bases from the end of the reads with quality score <20
  - Sliding window quality trim
  - Removing any reads that are <60bp after clipping and trimming

- Program prints basic statistics to standard output

Resource: http://www.usadellab.org/cms/?page=trimmomatic

Complexity filtering with seq-crumbs

- Seq-crumbs interleave fastq files
  - `interleave_pairs -o L2_10d.int.fb-tm.fastq L2_10d.1.fb-tm.fastq L2_10d.1.fb-tm.fastq`
- Filter low complexity reads
  - `filter_by_complexity -o L2_10d.int.fb-tm-sc.fastq --paired_reads --fail_drag_pair L2_10d.int.fb-tm.fastq`
- Seq-crumbs de-interleave fastq files
  - `deinterleave_pairs -o L2_10d.1.fb-tm-sc.fastq L2_10d.2.fb-tm-sc.fastq L2_10d.int.fb-tm-sc.fastq`

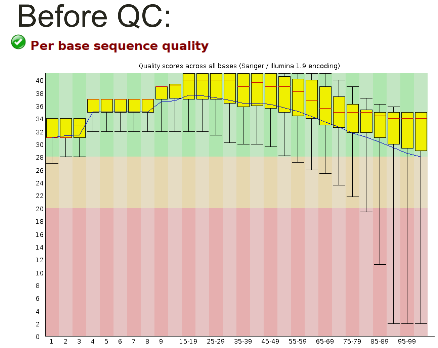


Resource: https://bioinf.comav.upv.es/seq_crumbs/available_crumbs.html

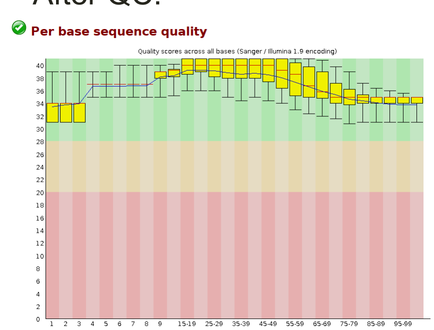
Quality control, reviewed

- Quality trimming/filtering
  - Adapter removal
  - Quality trimming
  - Length filtering
  - Complexity filtering
- Result: confidence in sequence presented

Before QC:



After QC:

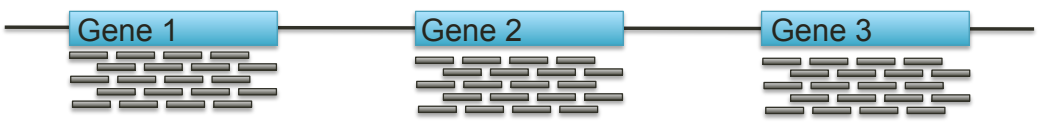


Contaminant filtering

- Do I need to do contaminant filtering?
- Questions to consider:
  - Where did my worm live?
    - Is the host's genome available?
    - If not, what's the next best thing?
  - Is my worm easily isolated from its host?
  - What does my worm/host eat?
  - Is my worm easily rinsed/ cleaned?
- What do you expect to see?

Contaminant filtering with Bowtie2

- Bowtie for mapping when splicing IS NOT a consideration
  - SILVA rRNA: http://www.arb-silva.de/
    - "SILVA provides comprehensive, quality checked and regularly updated datasets of aligned small (16s/18s, SSU) and large subunit (23s/28s, LSU) ribosomal RNA sequences for all three domains of life"
  - Bacteria
    - GenBank bacterial database
    - Custom database (human microbiome project)

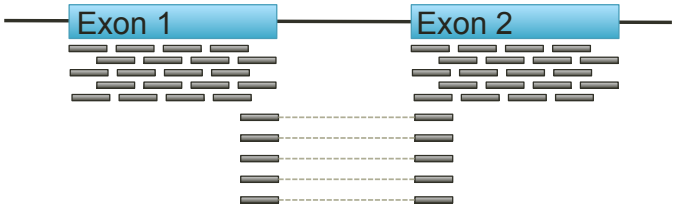


Resource: http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml

Contaminant filtering with Tophat2

- Tophat for mapping when splicing IS a consideration
  - Bowtie aligns reads that fall neatly within exons
  - Tophat splits reads across introns/gaps

- Databases
  - Human
  - Host
    - Intermediate
    - Definitive



- Sources
  - Genbank / Refseq
  - Ensembl.org

Resource: https://ccb.jhu.edu/software/tophat/manual.shtml

Remove contaminant reads

- Index database
  - `bowtie2-build Pig.fasta Pig.fasta`
- Map with bowtie
  - `bowtie2 -x Pig.fasta -1 L2_10d.1.fb-tm-sc.fastq -2 L2_10d.1.fb-tm-sc.fastq -S MapPig.sam`
- Map with tophat
  - `tophat2 -o L2_10d Pig.fasta L2_10d.1.fb-tm-sc.fastq L2_10d.1.fb-tm-sc.fastq`
- Counting mapped reads
  - For BAM file: `samtools view -c -F 4 accepted_hits.bam`
  - For SAM file: `samtools view -c -S -F 4 MapPig.sam`

- Remove contaminant reads and their mates as before

- Result:
  - High quality base calls
  - Confidence in the source of the reads

Resource: https://broadinstitute.github.io/picard/explain-flags.html (explanation of sam flags)

Results of quality control

- Count the number of reads maintained at each step!
  - `find . -name "*1.clean.fastq" | xargs wc -l`
  - Divide line count by 4 to get fastq entries

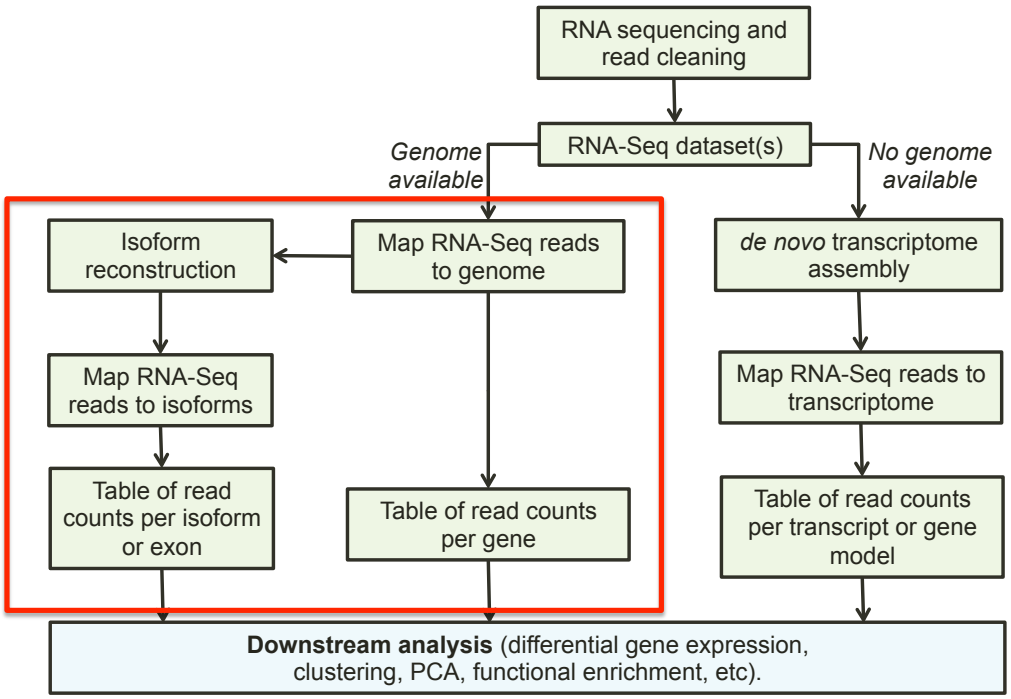
Downsampled read set:

	L2_10d	L3_16d	L3_17d	L4_21d	L5_42d	L5_r163	L5_r179	Total
Raw pairs	4,435,622	5,511,063	4,817,349	5,891,002	5,644,329	3,337,590	4,258,806	**33,895,761**
Flexbar	3,991,748	4,878,344	4,298,728	5,270,820	5,009,942	2,530,803	3,826,835	**29,807,220**
Trimmomatic	3,110,420	4,007,562	3,385,936	4,226,000	4,165,397	2,220,509	3,021,273	**24,137,097**
SeqCrumbs	3,093,078	3,917,497	3,373,150	4,183,440	4,113,913	2,219,777	3,011,416	**23,912,271**
Contaminants	2,696,239	3,643,862	3,350,928	3,927,395	3,926,103	2,211,368	2,993,460	**22,749,355**
% maintained	**60.80%**	**66.10%**	**69.60%**	**66.70%**	**69.60%**	**66.30%**	**70.30%**	**67.10%**

Full read set:

	L2_10d	L3_16d	L3_17d	L4_21d	L5_42d	L5_r163	L5_r179	Total
Raw pairs	43,592,929	54,459,409	47,371,505	58,231,629	55,800,467	32,809,672	41,902,924	**334,168,535**
Flexbar	39,229,484	48,195,339	42,272,646	52,090,873	49,524,734	24,877,392	37,657,504	**293,847,972**
Trimmomatic	30,586,411	40,437,016	33,302,203	42,655,938	41,935,364	21,862,295	29,745,662	**240,524,889**
SeqCrumbs	30,416,334	39,426,836	33,176,521	42,179,989	41,354,287	21,854,889	29,648,071	**238,056,927**
Contaminants	26,501,312	36,740,860	32,956,606	39,675,217	39,508,530	21,780,296	29,469,388	**226,632,209**
% maintained	**60.79%**	**67.46%**	**69.57%**	**68.13%**	**70.80%**	**66.38%**	**70.33%**	**67.82%**

RNA-seq analysis overview



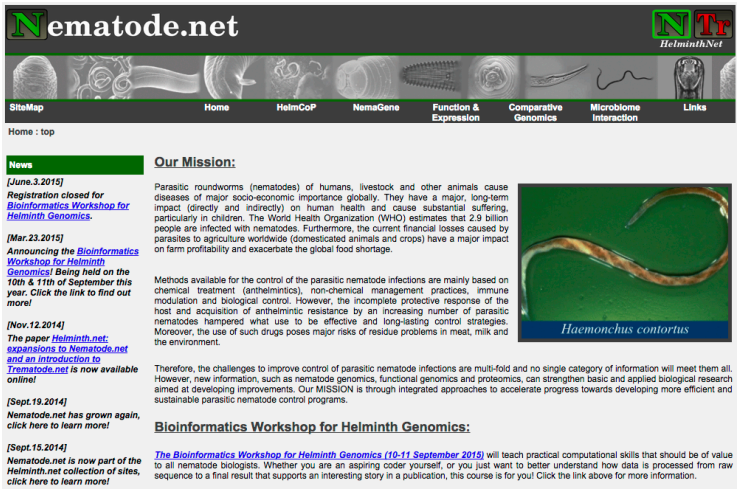
Section 2: Transcriptome
Module 1: Genome based RNA-seq analyses

1) Splice-aware alignment and verification
2) Genome-assisted transcript assembly
3) Counting reads in features for differential expression analyses

Resource: http://www.nature.com/nprot/journal/v8/n9/pdf/nprot.2013.099.pdf

Where to find a reference genome

- Sources:
  - Genbank/Refseq
  - Nematode.net
  - Wormbase.org
- Requirements:
  - Assembly fasta
  - GFF3
  - Functional annotation or protein/cds fasta



GFF3 format

```
3. ec2-user@ip-172-31-38-111:~/WORKSHOP_RESOURCES/Section_2/module_1 (ssh)
[ec2-user@ip-172-31-38-111 module_1]$ head -n 25 D918.gff3
##gff-version 3
T_suis-1.0_Cont72	Final_set	gene	74794	75765	.	-	.	ID=D918_GENE0001:gene;Name=D918_09719
T_suis-1.0_Cont72	Final_set	mRNA	74794	75765	.	-	.	ID=D918_GENE0001.1:mRNA;Parent=D918_GENE0001:gene;Name=D918_09719
T_suis-1.0_Cont72	Final_set	exon	74794	75765	.	-	.	ID=D918_GENE0001.1:exon;Parent=D918_GENE0001.1:mRNA
T_suis-1.0_Cont72	Final_set	CDS	74794	75765	.	-	0	ID=D918_GENE0001.1:CDS;Parent=D918_GENE0001.1:mRNA
T_suis-1.0_Cont40	Final_set	gene	395444	417867	.	+	.	ID=D918_GENE0002:gene;Name=D918_08404
T_suis-1.0_Cont40	Final_set	mRNA	395444	417867	.	+	.	ID=D918_GENE0002.1:mRNA;Parent=D918_GENE0002:gene;Name=D918_08404
T_suis-1.0_Cont40	Final_set	exon	395444	395926	.	+	.	ID=D918_GENE0002.1:exon;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	exon	396038	396514	.	+	.	ID=D918_GENE0002.1:exon;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	exon	396937	397314	.	+	.	ID=D918_GENE0002.1:exon;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	exon	397730	397910	.	+	.	ID=D918_GENE0002.1:exon;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	exon	398750	399097	.	+	.	ID=D918_GENE0002.1:exon;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	exon	417155	417867	.	+	.	ID=D918_GENE0002.1:exon;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	CDS	395444	395926	.	+	0	ID=D918_GENE0002.1:CDS;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	CDS	396038	396514	.	+	0	ID=D918_GENE0002.1:CDS;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	CDS	396937	397314	.	+	0	ID=D918_GENE0002.1:CDS;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	CDS	397730	397910	.	+	0	ID=D918_GENE0002.1:CDS;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	CDS	398750	399097	.	+	2	ID=D918_GENE0002.1:CDS;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont40	Final_set	CDS	417155	417867	.	+	2	ID=D918_GENE0002.1:CDS;Parent=D918_GENE0002.1:mRNA
T_suis-1.0_Cont17	Final_set	gene	633392	637389	.	-	.	ID=D918_GENE0003:gene;Name=D918_05776
T_suis-1.0_Cont17	Final_set	mRNA	633392	637389	.	-	.	ID=D918_GENE0003.1:mRNA;Parent=D918_GENE0003:gene;Name=D918_05776
T_suis-1.0_Cont17	Final_set	exon	633392	633550	.	-	.	ID=D918_GENE0003.1:exon;Parent=D918_GENE0003.1:mRNA
T_suis-1.0_Cont17	Final_set	exon	633607	633785	.	-	.	ID=D918_GENE0003.1:exon;Parent=D918_GENE0003.1:mRNA
T_suis-1.0_Cont17	Final_set	exon	633840	634033	.	-	.	ID=D918_GENE0003.1:exon;Parent=D918_GENE0003.1:mRNA
```

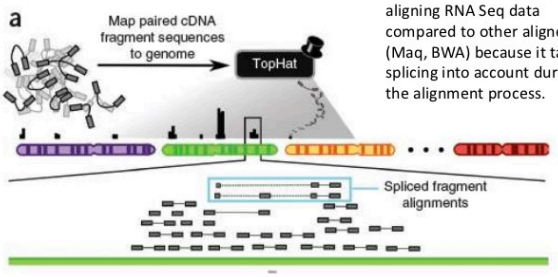
- Column 1: contig or scaffold
  - Must match the assembly fasta!
- Column 3: feature
  - CDS, coding_exon
- Column 9: mRNAs/genes the feature belongs to

Resource: http://www.usadellab.org/cms/?page=trimmomatic

Aligning reads with Tophat2

- Commands:

```
bowtie2-build
D918.fa D918.fa

tophat2 -o L2_10d -G
D918.gff3
D918.fa ../module_0/
L2_10d.
1.clean.fastq ../
module_0/L2_10d.
2.clean.fastq
```



TopHat is a good tool for aligning RNA Seq data compared to other aligners (Maq, BWA) because it takes splicing into account during the alignment process.

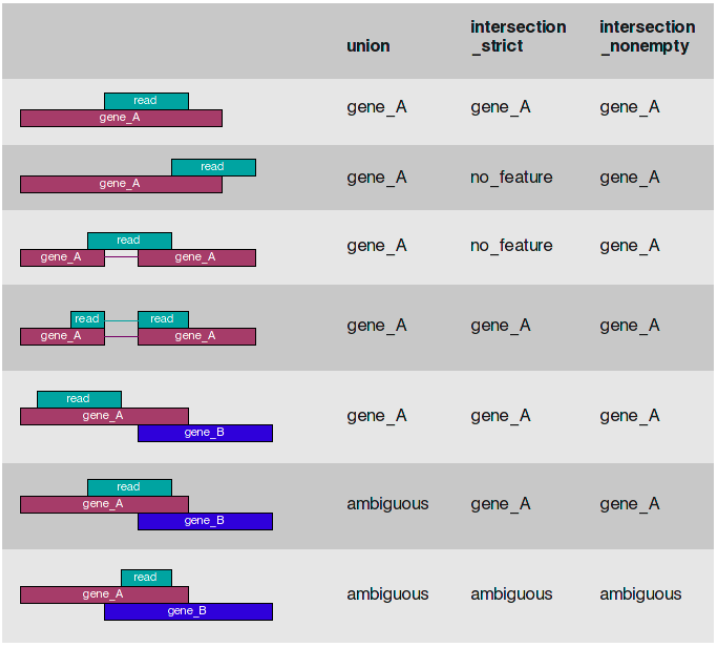
Figure from: Trapnell et al. (2010). Nature Biotechnology 28:511-515.

- -G option:
  - "If this option is provided, TopHat will first extract the transcript sequences and use Bowtie to align reads to this virtual transcriptome. Only the reads that do not fully map to the transcriptome will then be mapped on the genome. The reads that did map on the transcriptome will be converted to genomic mappings (spliced as needed) and merged with the novel mappings and junctions in the final tophat output"

Resource: https://ccb.jhu.edu/software/tophat/manual.shtml

Counting reads within features with htseq-count

- Command:
  - `htseq-count -f bam -r pos -t CDS -i Parent accepted_hits.bam D918.gff3 > L2_10d.htseq.txt`
- Arguments
  - -f: format
    - sam or bam
  - -r: order
    - name or pos
  - -t: feature type
    - coding_exon
    - exon
    - CDS
  - -i: feature ID
    - Parent



Resource: http://www-huber.embl.de/users/anders/HTSeq/doc/count.html

htseq-count output

	L2_10d	L3_16d	L3_17d	L4_21d	L5_42d	L5_r163	L5_r179
D918_00003	34	36	28	42	112	163	297
D918_00007	0	3	0	0	97	5	25
D918_00013	273	584	251	372	417	144	232
D918_00014	24	62	39	90	337	381	517
D918_00015	345	615	488	404	638	298	415
D918_00016	1801	1672	3838	1870	2614	1923	3446
D918_00017	3091	3833	4334	4376	3333	2011	2954
D918_00018	706	1680	1252	2430	2285	737	1040
D918_00019	3912	3062	1400	3638	3894	1643	1994
D918_00020	928	2060	2012	1971	3821	6971	3676
//							
alignment_not_unique	221176	839400	567739	890856	1011380	465826	512410
ambiguous	268632	549686	367069	470060	639345	330250	336040
no_feature	2888141	5856583	3677885	4318280	5470650	2710622	3702874
not_aligned	0	0	0	0	0	0	0
too_low_aQual	0	0	0	0	0	0	0

- All values should be integers
- 60-80% mapping rate is considered good
  - Sum counts for all genes and divide by cleaned read pairs

Resource: http://www-huber.embl.de/users/anders/HTSeq/doc/count.html

Cufflinks: genome-assisted transcript assembly

- Assembly transcripts for each sample separately using Cufflinks

```
cufflinks -o CuffOUTPUT
accepted_hits.bam
```

- Create a file that lists the assembly file for each sample

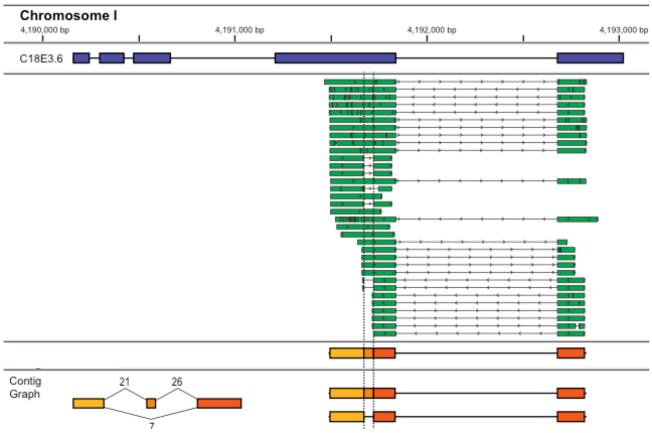
```
find . -name
"transcripts.gtf" >
assemblies.txt
```

- Run cuffmerge to create a single merged transcriptome annotation

```
cuffmerge -g genome.gtf
-s genome.fasta
assemblies.txt
```

  - Creates an output called merged.gtf

Resource: http://www.nature.com/nprot/journal/v7/n3/pdf/nprot.2012.016.pdf

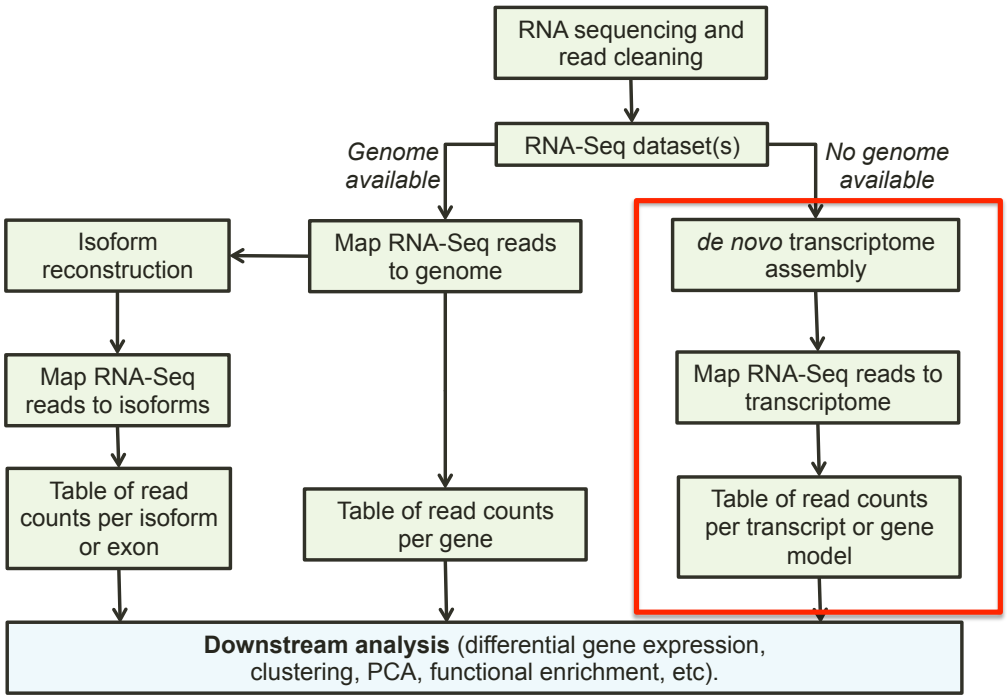


- Use gffread to print a fasta file of our transcripts

```
gffread merged.gtf -g genome.fasta
-w Transcripts.fa
```

  - Options:
    - -U: discard single-exon transcripts
    - -M: collapse matching transcripts
    - -K: collapse shorter, fully contained transcripts

RNA-seq analysis overview



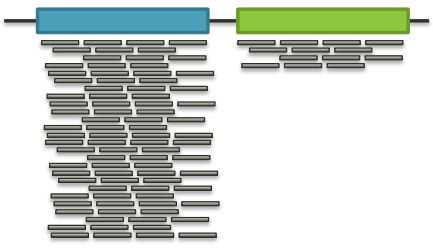
Section 2: Transcriptome
Module 2*: De novo* transcript assembly

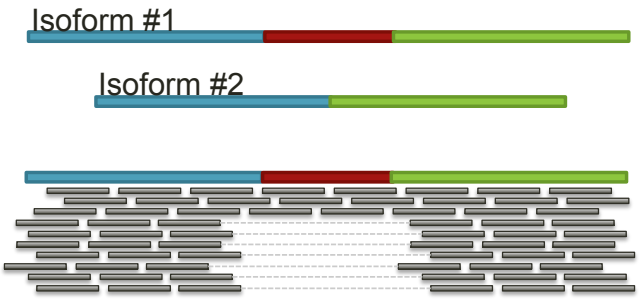
1) Digital read normalization
2) *De novo* transcript assembly
3) Post-assembly filtering
4) Mapping raw reads to the assembly

Problems with *de novo* transcript assembly

	L2_10d	L3_16d	L3_17d	L4_21d	L5_42d	L5_r163	L5_r179	Total
clean read pairs	26,501,312	36,740,860	32,956,606	39,675,217	39,508,530	21,780,296	29,469,388	**226,632,209**

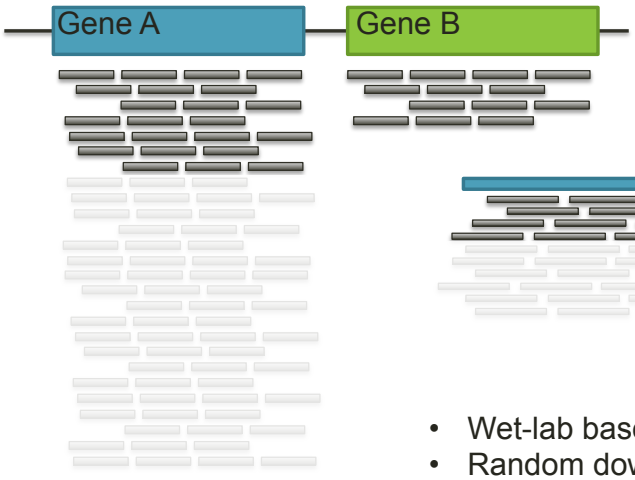
- Lots and lots of "puzzle pieces"
- Varying transcript abundance
- Alternative splicing
- Differential gene expression



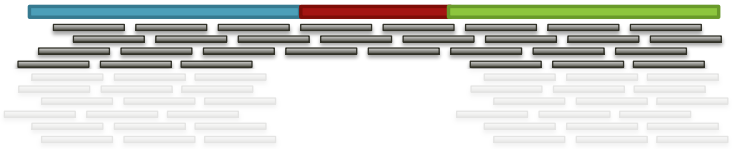


Resource: http://arxiv.org/pdf/1203.4802v2.pdf

Data reduction methods



What do you do when there's too much information?



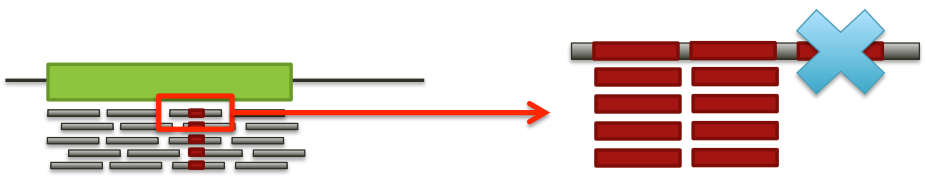
- Wet-lab based cDNA normalization techniques
- Random down sampling
- Digital read normalization

Resource: http://arxiv.org/pdf/1203.4802v2.pdf

Digital read normalization

- Solution: "a computational algorithm that systematizes coverage in shotgun sequencing data sets, thereby decreasing sampling variation, discarding redundant data, and removing the majority of errors"

- Method:
  - K-mer abundance correlates well with mapping-based estimates of read coverage
  - K-mers tend to have similar abundances within a read since they originate from the same DNA/RNA molecule



  - Estimate k-mer abundance (i.e., read coverage) to make the following determination

```
for read in dataset:
    if estimated_coverage(read) < C:
        accept(read)
    else:
        discard(read)
```

Resource: http://arxiv.org/pdf/1203.4802v2.pdf

Normalization software

- Khmer: http://khmer.readthedocs.org/en/v1.4.1/
  - Detailed protocol: http://khmer-protocols.readthedocs.org/en/v0.8.2/mrnaseq/2-diginorm.html
    - Decide which reads need to be maintained
    - Trim off low abundance parts of high coverage reads (i.e., errors)
    - Re-pair reads

- Trinity implementation:
  - https://trinityrnaseq.github.io/trinity_insilico_normalization.html

- For an explanation of the difference, see this blog post:
  - http://ivory.idyll.org/blog/trinity-in-silico-normalize.html

De novo transcript assembly with Trinity

- Trinity approach
  - Inchworm: assembles reads into unique sequences of transcripts, often generating full-length transcripts for a dominant isoform, and reporting unique portions of alternatively spliced transcripts
  - Chrysalis: clusters inchworm contigs into complete de Bruijn graphs for each cluster
  - Butterfly: processes the individual graphs to report full-length transcripts for alternatively spliced isoforms

- Trinity command:

```
Trinity --seqType fq --max_memory XXG --left AllLeft.fastq
--right AllRight.fastq --normalize_reads -output TRINITY
```

- Time and memory:
  - Approximately 1G of RAM per million read pairs
  - Approximately 0.5-1h per million read pairs

Resource: http://trinityrnaseq.github.io/

Trinity output

- Trinity will create a Trinity.fasta output file in the specified output directory
- Trinity groups transcripts into clusters based on shared sequence content. These clusters are loosely referred to as “genes” or “unigenes”. This information is coded in the trinity accession.

```
1. ec2-user@ip-172-31-38-111:~/WORKSHOP_RESOURCES/Section_2/mod...
[ec2-user@ip-172-31-38-111 TRINITY]$ head -n 25 Trinity.fasta
>TR1|c0_g1_i1 len=262 path=[240:0-261] [-1, 240, -2]
TAATCTGTTTTCGAAATGGTTTCTTCCTTTTTTCGTGGGTACCTACCAAGCAAAATGGAC
TGCACTTATCTTGCATTTCAAGCCATTCTAGAGCCTTATCGTCCGAAGAGCATATAGCTG
CTTAATAAGCGTTAATACTTCTCGGTCAGATGTTCCTGTTTGGTTCCCTTTATGCCTCTG
CTAAGCATTCAATAGTTTCATCATCGCTCTTTATTGCACGGCCTTCGTTTCCGCGAGCTG
AGCGGATGTCCTTGTCATACTG
>TR2|c0_g1_i1 len=255 path=[233:0-254] [-1, 233, -2]
GGAAGCTTTAGGGGAAATAAATTTCGCTCGATTTTGCTCTACGCGTTATCCAACGAAGCG
TAGCATTTAGTTTGGGCATAAGTAAACATGCGAATCGAAATCTTTGCAGAAATGCTTTTT
GTGCATCTTACTGTTGTCCGCTAGCGCTGCATTAAAATCAAGTAACCTTGACAAGTTACT
TTGATTATGCTTGAATAATTTTTCTTTCGACTTAAACGTATATTATTAGTGTTGGCTGGT
CATTTAGCCTTTGAA
>TR3|c0_g1_i1 len=418 path=[791:0-417] [-1, 791, -2]
GGTAACGCTTTGGGAACCCTTTTTCTTAATAAAGACTTTTGGTCCATCGTTTCAACGACG
CTACTTTATCTCTGTTGAAAAGTGAACAAAGATAAGATGGCGTCGCTCAAAAGGTGAAGC
TGTTGTTATCAGACAATCGATAATCCAATAAAAATGTTGATAGATTTTAAAAAGATACGT
ATGTGCGAGATAATTAAAATTTGCATAAAGTTACAAAGCAATCCCTCAGTGCTTCCTCTC
TGCCTTGCTGCAGCCTACGTGATTCACTTGCTAAGCCCTAAACCAATCAATGATGGAAGG
AAGCGCTTATTGTACCTTTGTCCTGACGTTTTTAGTCAGTTCGGAACTGTCCTTCCCTAT
ATCGCGTAGATTCAATATGAATAGTAGATTTGAAAGGTACGTCAATTTGATTTGCATA
```

http://trinityrnaseq.github.io/#trinity_output

Assembly statistics

- Command:

```
perl ~/bin/
trinityrnaseq-2.0.6/util/
TrinityStats.pl
Trinity.fasta
```

- In a perfect assembly, “unigenes” = expressed genes
- Why are there so many genes/ transcripts?
  - Fragmentation
  - Low-confidence transcripts

“Test” assembly:

```
################################
## Counts of transcripts, etc.
################################
Total trinity 'genes':  48361
Total trinity transcripts:      74070
Percent GC: 45.34

########################################
Stats based on ALL transcript contigs:
########################################

        Contig N10: 2736
        Contig N20: 2035
        Contig N30: 1646
        Contig N40: 1351
        Contig N50: 1102

        Median contig length: 557
        Average contig: 797.01
        Total assembled bases: 59034791


#################################################
## Stats based on ONLY LONGEST ISOFORM per 'GENE':
#################################################

        Contig N10: 2265
        Contig N20: 1676
        Contig N30: 1302
        Contig N40: 1036
        Contig N50: 829

        Median contig length: 451
        Average contig: 648.84
        Total assembled bases: 31378557
```

Resource: http://trinityrnaseq.github.io/#trinity_output

Assembly filtering

- Align reads and estimate abundance

```
perl ~/bin/trinityrnaseq-2.0.6/
util/
align_and_estimate_abundance.pl --
transcripts Trinity.fasta --seqType
fq --left ../AllLeft.fastq --
right ../AllRight.fastq --
est_method RSEM --output_dir RSEM
--aln_method bowtie2 --
prep_reference
```

- Filter lowly supported transcripts

```
perl ~/bin/trinityrnaseq-2.0.6/
util/filter_fasta_by_rsem_values.pl
--rsem_output=RSEM.isoforms.results
--fasta=../Trinity.fasta --
output=Trinity.filtered.fasta --
tpm_cutoff=1.0 --isopct_cutoff=1.00
```

Paragonimus kellicotti assembly:

	Unfiltered	Filtered
# unigenes	153,461	59,050
# transcripts	251,721	91,029
Ave transcript size	460 bp	563 bp
Alternative splicing	24.8% of unigenes, ave 3.6, max 85	24.4% of unigenes, ave 3.2, max 20
% pairs mapped	68.3%	66.3%

Resource: http://trinityrnaseq.github.io/analysis/abundance_estimation.html

Feature counting for differential expression

- Prepare reference

```
perl ~/bin/trinityrnaseq-2.0.6/util/
align_and_estimate_abundance.pl --transcripts
Trinity.filtered.fasta --est_method RSEM --aln_method bowtie2
--prep_reference
```

- Align reads and estimate abundance

```
perl ~/bin/trinityrnaseq-2.0.6/util/
align_and_estimate_abundance.pl --transcripts
Trinity.filtered.fasta --seqType fq --est_method RSEM --
aln_method bowtie2 --left ../../../module_0/L2_10d.
1.clean.fastq --right ../../../module_0/L2_10d.2.clean.fastq
--output_dir L2_10d
```

- Join the abundance values for each sample into matrix for DESeq2

```
perl ~/bin/trinityrnaseq-2.0.6/util/
abundance_estimates_to_matrix.pl --est_method RSEM L2_10d/
RSEM.genes.results L3_16d/RSEM.genes.results …
```

Resource: http://trinityrnaseq.github.io/analysis/diff_expression_analysis.html

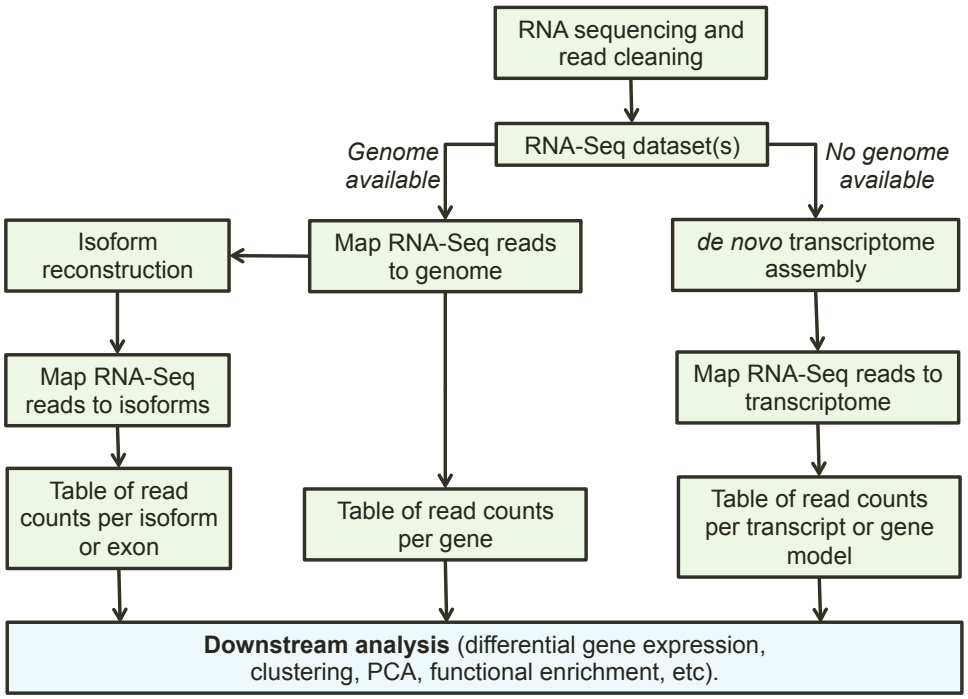
Feature counting for differential expression

***Cooperia punctata* count table**

	HIGH.genes.results	LOW.genes.results	UntreatedA.genes.results	UntreatedB.genes.results
comp197262_c2	53.02	51.97	24	107
comp196358_c0	90	125	104	91
comp194909_c0	3	2	0	79.07
comp189445_c0	15	5	7	15
comp199614_c0	19	23	24.67	18.89
comp191897_c2	16	20	26	3
comp196155_c1	223	283	119	467
comp196537_c0	74.2	98	38.67	200.96
comp194722_c1	11	6	1	33
comp200992_c1	9.24	21.98	27	11
comp189025_c0	57993.94	35917.49	21809.97	76141.69
comp195426_c0	32	74.17	52.45	100.2
comp197998_c0	27	8	12	13
comp201556_c2	22	19	22	25

Resource: http://trinityrnaseq.github.io/analysis/diff_expression_analysis.html

RNA-seq analysis overview



Section 2: Transcriptome
Module 3: Expression and differential expression

Introduction - Expression and differential expression

- For this module, we will be off of the server and working directly on your laptops.

- We will use data files that you downloaded using scp yesterday, which should be saved in ~/Desktop/WORKSHOP_RESOURCES/Section_2/module_3/. Please check that you have downloaded files and folders to this directory.

- Raw data was produced in the previous modules.

- You should already have both RStudio and MS Excel installed on your laptops, as requested before the class started.

Differential gene expression software

- Calling differentially expressed genes is a complicated statistical problem.
- “Dispersion” of a gene or a sample is used to estimate baseline (within-replicate) variability, and is essential for accurate statistical measurement. Genes with high inter-replicate variability should not be considered “differential”.
- Some measure of dispersion is calculated by all widely-accepted differential callers, but they all calculate it in slightly different ways.
- Three software packages are primarily used: **DESeq**, **EdgeR**, and **CuffDiff**. Others include SAMseq, baySeq, NOIseq, and EBSeq.
- **DESeq** and **EdgeR** are the two most commonly used differential gene expression calculation packages. These produce similar overall results in terms of final gene lists.

<u>How to choose a differential expression caller</u>

- The primary practical difference between **DESeq** and **EdgeR** is sensitivity (i.e. the number of genes called differential).
- If you are interested in transcript / isoform data, then use **CuffDiff**. CuffDiff tends to be very stringent (fewer differentially expressed genes than DESeq or EdgeR).
- **SAMseq** can be useful for cross-sample differential expression calling, but should not be used for two-sample comparisons.
- Having a larger set of differentially expressed genes is not necessarily better!
- More differentially expressed genes = more false positives, and a larger set of genes to summarize for biological interpretation.

http://bib.oxfordjournals.org/content/early/2013/12/02/bib.bbt086.long

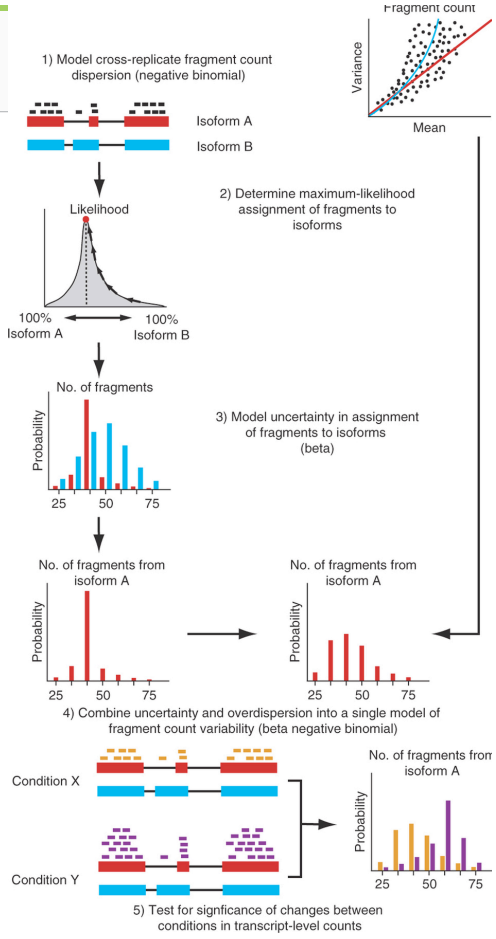
CuffDiff

- CuffDiff considers read counts per exon, and can identify significant changes in exon use and isoform abundance for the same gene.

- This is useful (a) for model organisms where there is known functional significance for specific exons/isoforms or for (b) for studies of a subset of specific genes of interest.

- At a genome-wide level, quantifying differential exon usage complicates downstream analysis without providing practically useful data.

- For example, it is difficult to perform genome-wide functional enrichment testing on differentially expressed isoforms, since multiple isoforms from the same gene can contribute to enrichment scores.



http://www.nature.com/nbt/journal/v31/n1/fig_tab/nbt.2450_F2.html

Replicate considerations

- At least triplicate is preferred for accurate analysis.

- Some samples may be lost due to very high variability from other replicates or low quality RNA, so duplicate is risky (single-replicate produces unreliable statistics).

- Collecting the replicates by repeating an experiment at a later time almost never works for helminth studies.

- Both DESeq and EdgeR *can* be executed with single replicates, but use different statistical models.

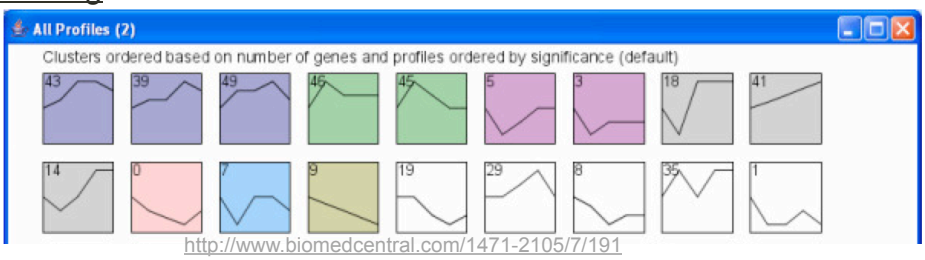
- Another program called **GFOLD** is designed specifically for single-replicate samples, but these comparisons with any software are not confident without additional validation (e.g. qPCR of identified genes).

- Track metadata carefully whenever possible. E.g., the number of worms collected, whether there is a possibility of having mixed samples (male and female, L3 and L4, etc), time of sampling, etc. This may help to explain within-replicate variability in some cases.

Gene clustering

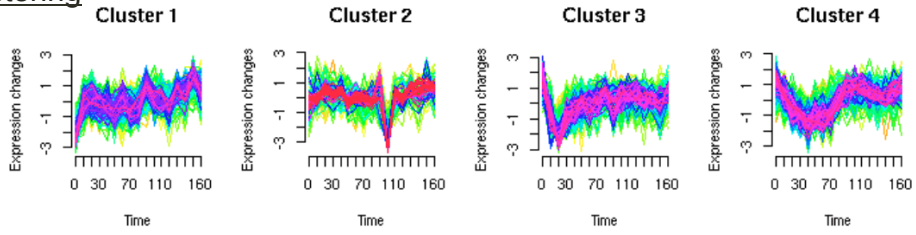
- Another analysis approach is to cluster samples based on their overall expression patterns across all available RNA-Seq datasets.
- While this is useful for grouping and classifying genes, the clusters only consider the pattern and do not consider whether the genes are statistically differentially expressed.
- One tool called Short Time Series Expression Miner (STEM) clustering will also identify over-represented patterns, representing clusters of probable biological significance.

STEM Clustering



http://www.biomedcentral.com/1471-2105/7/191

Mfuzz Clustering



http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2139991/

Differential gene expression measurement

Experimental design considerations: **What are the samples you want to compare? What approach will you use to compare them?**

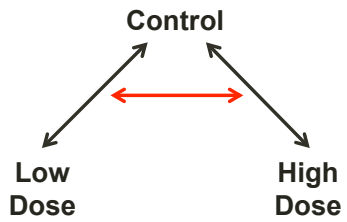
Example 1: Treatment(s) vs Control

1A. Simple treatment / control pair:

Control ⟷ **Treatment**

- Which genes are high in treatment (upregulated) or lower in treatment (downregulated)?

1B. Control vs multiple treatments
(e.g. high and low doses of a drug treatment)

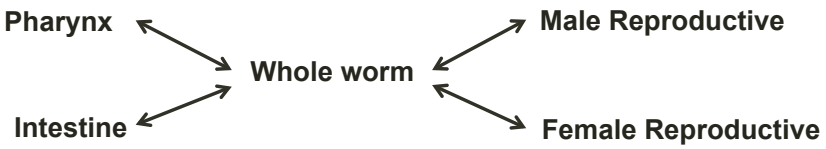


- Which genes are upregulated or downregulated by both treatments, and which ones are only differentially regulated by high-dose treatment but not low?

Differential gene expression measurement

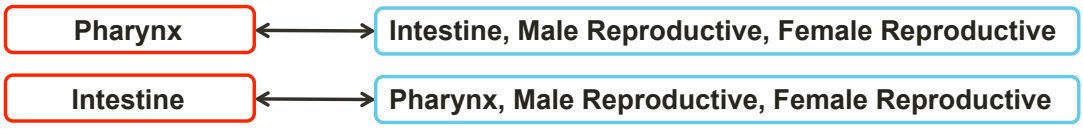
Example 2: Tissue-based (unordered, multiple samples)
e.g. Whole-worm, intestine, pharynx, and male and female reproductive tissue.

2A. Each compared to whole-worm:



- What are the tissue-specific overexpressed genes relative to the whole-worm sample?

2B. Each compared to all other tissues:



- What are the tissue-specific overexpressed genes relative to the other sampled tissues?

2C. Cross-sample combinatorial comparisons

- Some cross-sample differential expression callers (e.g. SAMSeq) can identify combinations of samples with upregulation (e.g. upregulated in both pharynx and intestine relative to other tissues).

http://statweb.stanford.edu/~tibs/SAM/

Differential gene expression measurement

Example 3: Stage-based (time series) data
(e.g. L2, L3, L4, L5 larvae)

3A Pairwise : L2 ⟷ L3 ⟷ L4 ⟷ L5

- Which genes are upregulated in one stage vs its surrounding stage(s)?

3B Grouped : L2 L3 ⟷ L4 L5

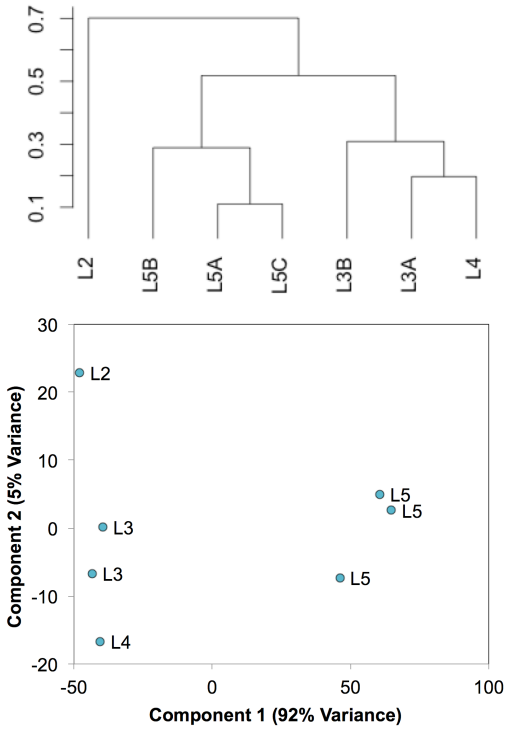
- Which genes are upregulated in early stages relative to late stages?
- Stages are treated as pseudo-replicates for each other.

3C Individual : L2 ⟷ L3 L4 L5

L5 ⟷ L2 L3 L4

Etc.

- Which genes are upregulated in one stage relative to all others?



Using RStudio

- R is a free software environment for statistical computing and graphics.
- RStudio is a set of integrated tools to make R much easier to use.

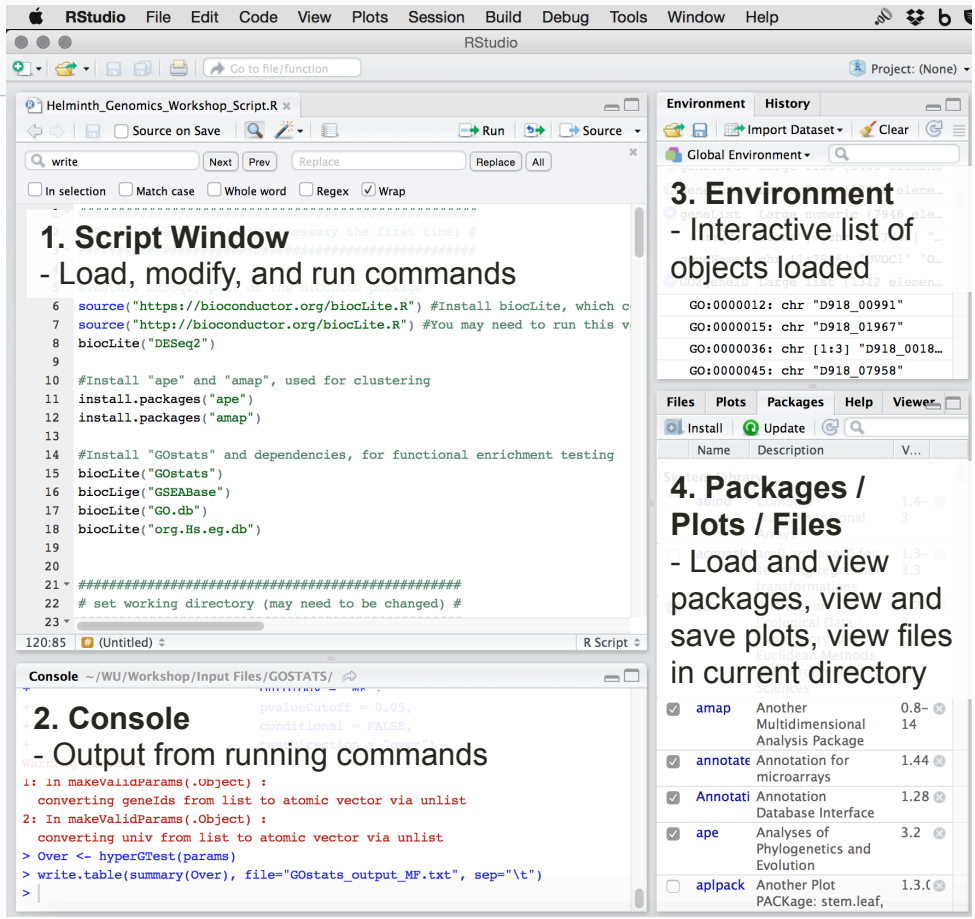
- “Packages” of existing software can be downloaded, installed, and loaded easily.
- Many bioinformatics tools (especially for statistics analysis) are available exclusively in R.
- You can typically work with R by modifying existing scripts, most of which can be downloaded from manuals or other internet resources.

- In this module, we will learn how to use R studio to:
  - Install libraries, set the working directory and input files
  - Run DESeq2 for differential gene expression analysis
  - Run PCA and hierarchical clustering
  - Run GOSTATS for enrichment of differentially expressed genes

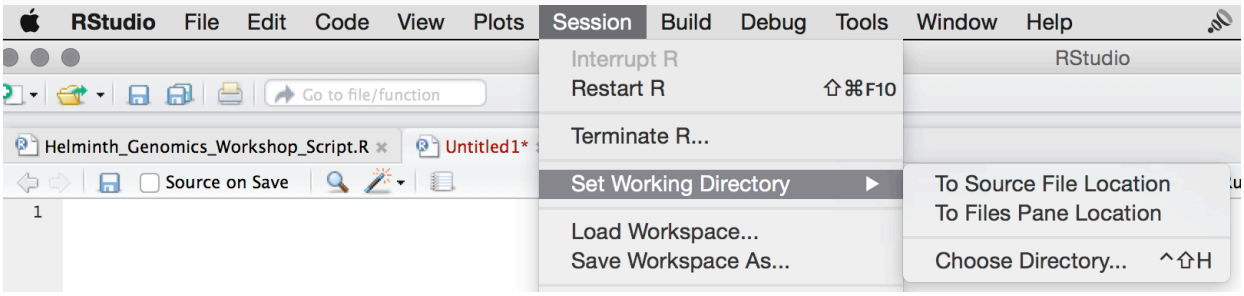
Using RStudio

- The RStudio interface is split into four windows.

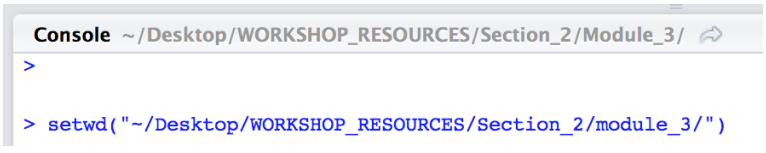
- If you only download R, then you will only have the console to work with.



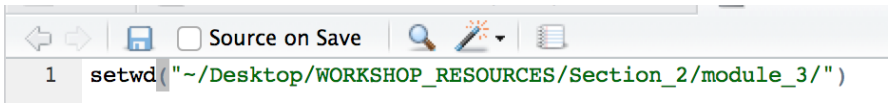
An example of interacting with RStudio



- From the menu, select “choose directory” as shown above, to set the working directory where files will be loaded from and saved to. Set to ‘~/Desktop/WORKSHOP_RESOURCES/Section_2/module_3/’ for this course.



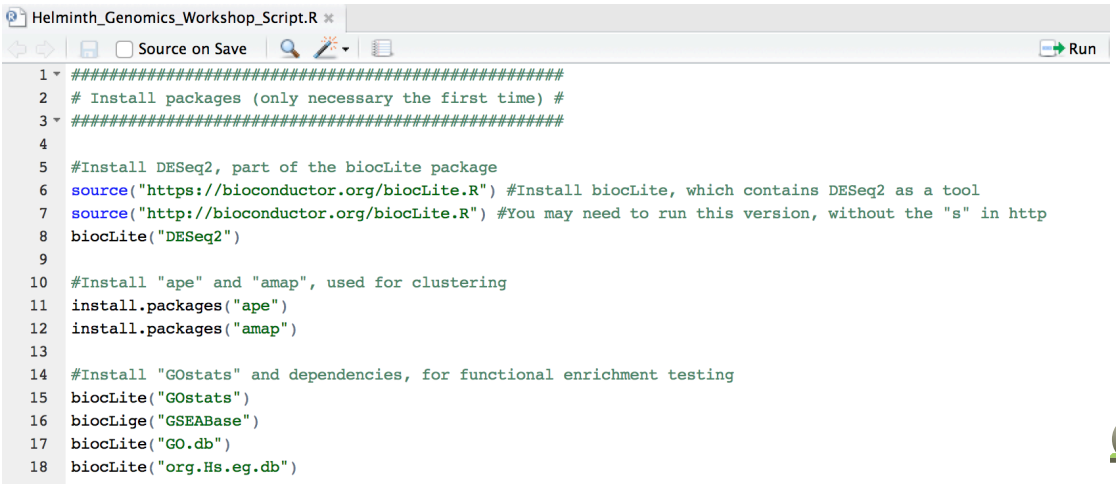
- When you do this, you will see the “setwd” R command ran in the console. This can then be copied and pasted in the script window.



- If you were to save this script in the future, you could now highlight and run this command in order to set the working directory more easily.

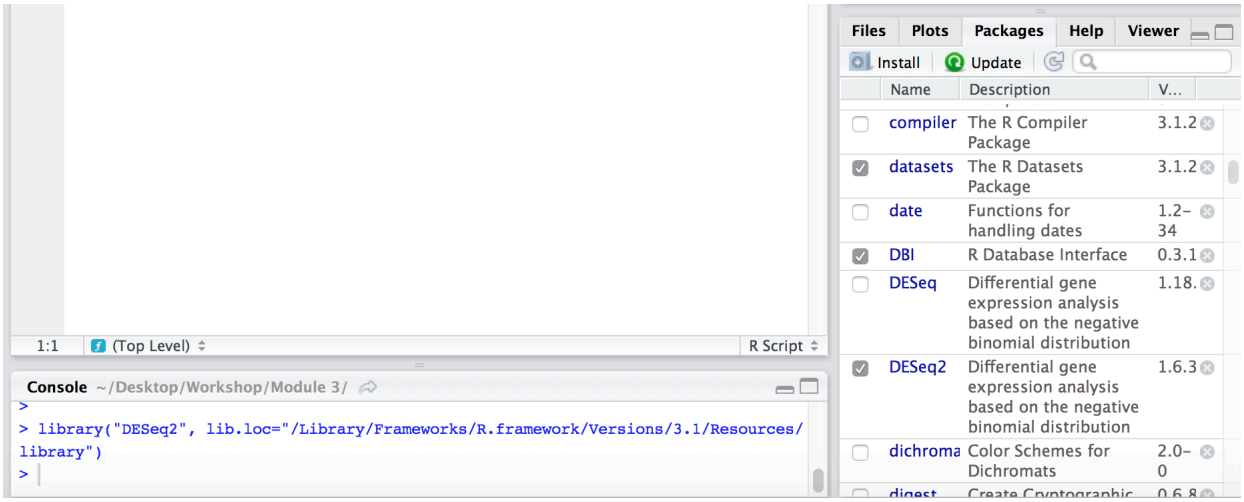
Installing R packages

- Now open the “Helminth_Genomics_Workshop_Script.R” file. This contains all of the commands we will need for the workshop.
- Any information following a **#** sign is a comment to clarify what the code is for.
- First, we will install packages. Packages are either installed directly using “install.packages()”, or they are loaded through bioconductor (“biocLite”).
- Highlight the code shown and click “run” to install all of the necessary packages.
- The manuals for different R packages will include the line necessary to install them.
- Installations only need to be performed one time on each computer, but the packages need to be loaded every time R is restarted.



Loading R packages

- After you install packages, they will show up in the “Packages” list in your RStudio sidebar. To “load” the packages in the future, you can simply check them off. When you do, you will the package loading code in the console window.
- This code can also be pasted into scripts. Note that the full path is not necessary (e.g., in the screenshot below, you can just use **library(“DESeq2”)** instead, which will make your script compatible on other people’s computers.
- Packages can also be searched and installed from this menu, but it is typically easier to paste the install code from a guide.



Preparing and loading input files: DESeq analysis

- Almost all differential expression callers require raw reads as input.
- We generated read counts per sample from HTSeq output in the previous module.

- Open “tsuis_rnaseq_htseq_countstable.txt” from the DESeq directory (in MS Excel)
- This file contains unprocessed HTSeq count output (from the previous module) for *T. suis* collected from different stages. All downstream work will be performed on this dataset.
- Note that this is saved as a **tab-delimited text file**. This will be the standard output from most linux programs. If you save in Excel, you will need to specify this format in the “Save as” menu.

	A	B	C	D	E	F	G	H
1	Gene	TSAC-10_day	TSAC-16_day	TSAC-17_day	TSAC-21_day	TSAC-42_day	TSAC-Adult1-	TSAC-adult_w
2	D918_00003	34	36	28	42	112	163	297
3	D918_00007	0	3	0	0	97	5	25
4	D918_00013	273	584	251	372	417	144	232
5	D918_00014	24	62	39	90	337	381	517
6	D918_00015	345	615	488	404	638	298	415

- DESeq requires the genes to be listed in the first columns, the samples labeled in the first row, and read counts in the matrix. This is standard to many of the other differential callers (including EdgeR)

Loading input files

- After setting the working directory and loading DESeq, we load the input reads file.
- In R, “objects” are defined using an ‘arrow’ **<-**
- We will call the object for the HTSeq counts table “COUNTS”
- It is important to understand the input command because (a) it is often omitted when you download scripts (they assume you know how to do this) and (b) having the input formatted or loaded incorrectly is a very common reason that scripts don’t work when they are launched. Pay close attention to manuals describing input data.

```
##########
# DESEQ2 #
##########

#Set working directory
setwd("~/Desktop/Workshop/Module 3/DESeq/")

#Load library
library("DESeq2") #DESeq

#Read INPUT READS table (1 row of headers with sample names and 1 column with gene names; Saved as tab-delimit
COUNTS <- as.matrix(read.table(file="tsuis_rnaseq_htseq_countstable.txt", sep="\t", header=TRUE, row.names=1))
```

- Object name
- Load as a matrix object (not always necessary)
- Most typical command for loading data
- Filename in working directory
- Separator for the text file; Can also be comma or space, but \t (tab) is the most common.
- Set to “FALSE” if there are no headers
- Omit if there are no row names

Loading input files

- For DESeq, you will also need to prepare a metadata file describing your samples.
- This input file is formatted as shown below. Column names can be customized, but the first column must contain sample names corresponding to the counts table.

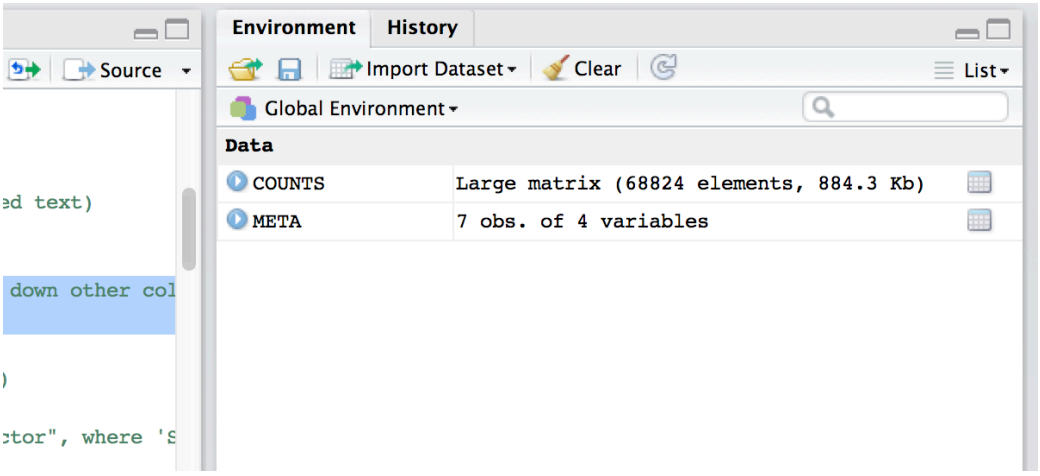
	A	B	C	D	E
1	Sample ID	Age	Stage	Comparison1	Comparison2
2	TSAC-10_day_larvae-R182	10	L2	Early	L2
3	TSAC-16_day_larvae-R171	16	L3	Early	Early
4	TSAC-17_day_larvae-R181	17	L3	Early	Early
5	TSAC-21_day_larvae-R165	21	L4	Early	Early
6	TSAC-42_day_larvae-R166	42	L5	Late	Late
7	TSAC-Adult1-r163	Adult	L5	Late	Late
8	TSAC-adult_worms-R179	Adult	L5	Late	Late

- The samples that you want to compare should be grouped in one of the columns. Here, we will focus on “Comparison1”, which is early larval stages vs late stages.
- You will need to construct this metadata file yourself prior to running R. We will look at creating tables in Excel later in this module.
- Unlike the read counts table, this input command is not loaded “as.matrix”, but is just a table:

```
#Read META DATA table (Sample names corresponding to input reads file down first column, c
META <- read.table(file ="tsuis_rnaseq_metadata.txt", sep="\t", header=TRUE, row.names=1)
```

Managing data

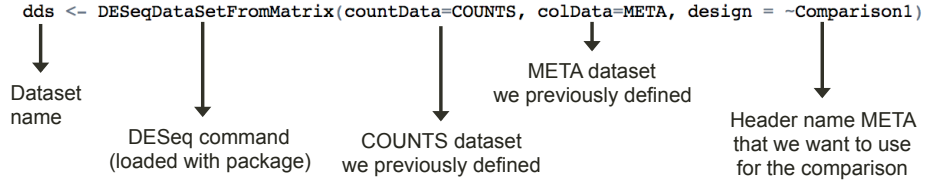
- In RStudio, loaded objects show up in the environment window.
- If you click on the table icon to the right of the object, you can view the object (in the script window) to ensure that files have loaded properly.
- Checking to see if intermediate objects are empty (“NULL”) is a good way to troubleshoot where problems are starting.



Running DESeq

- First, we will make “dds”, the DESeq DataSet object

```
#Make DESeq object ("design" refers to the column header in the meta data defining your comparison of interest)
dds <- DESeqDataSetFromMatrix(countData=COUNTS, colData=META, design = ~Comparison1)
```



- In some cases, there are secondary factors to consider. For example, samples may have been collected in two batches, introducing potential variance independent of the comparison.
- This data can be specified in the metadata file, and considered by DESeq using the following syntax:

```
dds <- DESeqDataSetFromMatrix(countData=COUNTS, colData=META, design = ~SecondaryFactor + Comparison1)
```

- This is also useful in cases of paired samples (e.g., the same individuals before and after treatment). DESeq and EdgeR can both utilize secondary factors, but CuffDiff and other software cannot.

Running DESeq and saving results

- The following line runs the core DESeq code:

```
#Core DESeq code
dds <- DESeq(dds)
```

- Then, this summarizes the results, and writes the summary to a file:

```
#Results summary
res<-results(dds)
summary(res)
sink(file="Comparison2_Early_vs_Late_tsuis_deseq2_results_summary.txt") #Define output file
summary(res)
sink(NULL)
```

The results are also shownin the console:

```
out of 9816 with nonzero total read count
adjusted p-value < 0.1
LFC > 0 (up)      : 1988, 20%
LFC < 0 (down)    : 1525, 16%
outliers [1]      : 299, 3%
low counts [2]    : 0, 0%
(mean count < 0.1)
```

- This shows that at an adjusted p-value of 0.1, ~36% of genes are differentially expressed.
- We will parse the output manually later, with a different p value cutoff.

Running DESeq and saving results

- Next, we prepare the output data:

```
#Output results from target comparison (enter header name from metadata f
outputtable <- results(dds, contrast=c("Comparison1", "Early", "Late"))
```

- We will save this object to a file in the next command
- Interpret dds object as readable results
- Define comparisons
- Header name from the META file
- First comparison group name under the header
- Second comparison group name under the header

- Finally, the write.table command is used to export the results to a file in the working directory. We'll look at the results later, during the Excel tutorial.

```
write.table(outputtable, file="Comparison1_Early_vs_Late_tsuis_deseq2_output.txt", sep="\t")
```

- Object name
- Output filename (try to be descriptive)
- Tab delimited

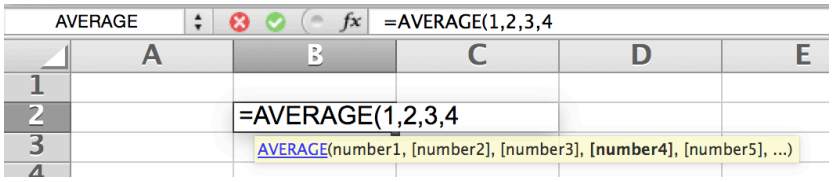
Introduction to Microsoft Excel

- Excel is a spreadsheet program which is useful for organizing and visualizing data, calculating statistics, and performing analyses.
- Today we will learn a variety of approaches for using Excel to work with whole-genome data, with a focus on maintaining data integrity and organizing data in the most accessible way possible.
- We will go from several raw data files (generated in previous modules) to a complete database with functional annotation data, expression levels, differential expression data, and more.
- Open "Module 3 Table Completed.xlsx" in the 'Excel' folder to view a copy of the completed database, before we create it.

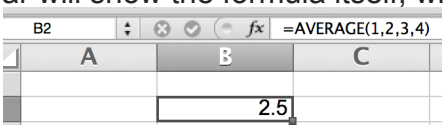
Gene	InterProScan data (Sept 11 2015)		HTSeq output (tsuis_rnaseq_htseq_countstable.txt, Sept 11 2015)							Gene Lengths (bp)	
	InterPro domains	Gene Ontology Terms	Stage	L2	L3	L3	L4	L5	L5	L5	
			Age (days)	10	16	17	21	42	Adult	Adult	
			Sample Na	TSAC-1(	TSAC-1(	TSAC-1	TSAC-2	TSAC-4	TSAC-A	TSAC-a	
D918_00007	-	-	0	3	0	0	97	5	25	369	
D918_00013	IPR018468:Double-strar	-	273	584	251	372	417	144	232	1230	
D918_00014	-	-	24	62	39	90	337	381	517	1059	
D918_00015	-	-	345	615	488	404	638	298	415	1341	
D918_00016	IPR018972:Something ε	GO:0005634:Cellular C	1801	1672	3838	1870	2614	1923	3446	1410	
D918_00017	IPR000793:ATPase, F1/	GO:0046034:Biological	3091	3833	4334	4376	3333	2011	2954	1860	
D918_00018	IPR001841:Zinc finger, f	GO:0005515:Molecular	706	1680	1252	2430	2285	737	1040	660	
D918_00019	-	-	3912	3062	1400	3638	3894	1643	1994	1806	
D918_00020	IPR008974:TRAF-like:1	GO:0005515:Molecular	928	2060	2012	1971	3821	6971	3676	2682	
D918_00021	IPR011989:Armadillo-lik	GO:0005515:Molecular	772	1395	1202	1287	1159	852	883	1983	
D918_00022	IPR004947:Deoxyribonu	GO:0004531:Molecular	32	422	533	4792	25899	9485	12312	1065	
D918_00023	-	-	0	16	25	45	278	1213	315	195	
D918_00024	IPR021869:Ribonucleas	GO:0004531:Molecular	72	565	679	3744	15520	3983	9318	1344	
D918_00025	IPR006990:Tweety:8.7e	-	523	872	989	1024	922	1377	673	1059	
D918_00026	-	-	960	2019	847	1410	1032	352	363	348	
D918_00027	IPR017441:Protein kina	GO:0005524:Molecular	416	383	435	220	450	427	338	741	
D918_00028	IPR000719:Protein kina	GO:0004674:Molecular	2406	3518	1893	2507	4375	1455	1547	1188	

Introduction to MS Excel: Formulas

- The spreadsheet is laid out in a coordinate system of “cells” with lettered columns and numbered rows. Numbers or string can be entered into any cell just by typing and pressing enter.
- Navigate the spreadsheet using either your cursor or by using the arrows on your keyboard. Multiple cells can be highlighted with the keyboard by holding shift and scrolling with the arrows.
- Formulas can be entered in any cell by entering an “=“ sign.
- All formulas follow a specific format of the “=“ sign, the formula name, an open bracket, variables, and a closed bracket.
- As you type a formula, a yellow box will pop up to tell you what variables can be entered. Here, I am calculating the average of a series of numbers, in cell B2. The yellow box indicates that I should enter the numbers with commas in between:

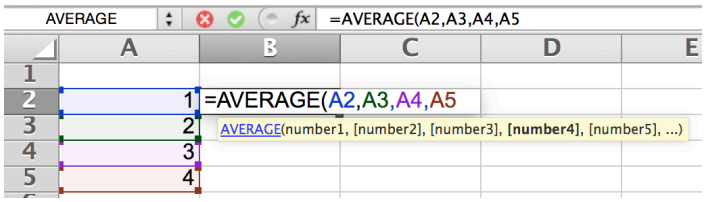


- After you close the bracket and press enter, the cell value will show the *result* of the formula, but the formula bar will show the formula itself, when cell B2 is selected:

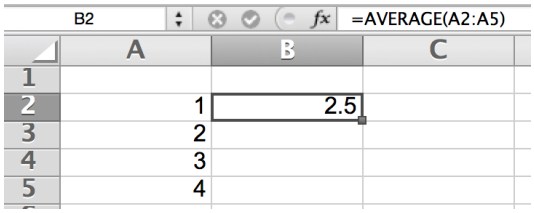


Formulas in MS Excel

- Formulas can also be calculated on references to cells containing numbers. This is the same formula, but the numbers have been replaced with references to cells containing numbers:



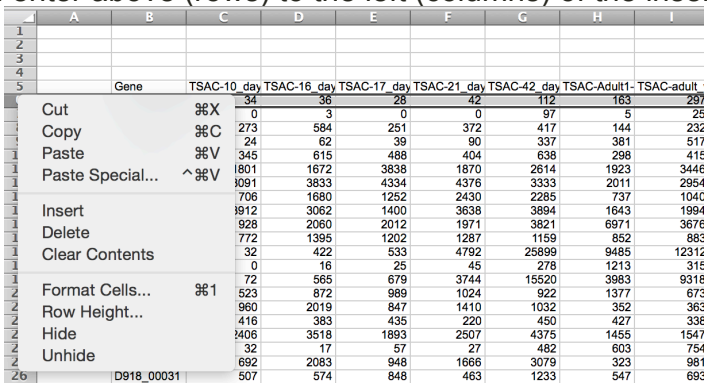
- Rather than list all of the cells, *cell ranges* can be used. This follows the format of the first cell, a colon, and then the last cell:



- Ranges can span columns and rows (e.g., take the average of a large table).
- Cell references do not need to be typed in manually. You can select the range with your mouse, or you can use the keyboard to select it, after typing the formula and opening the bracket.
- A full list of Excel formulas can be found here: http://www.techonthenet.com/excel/formulas/

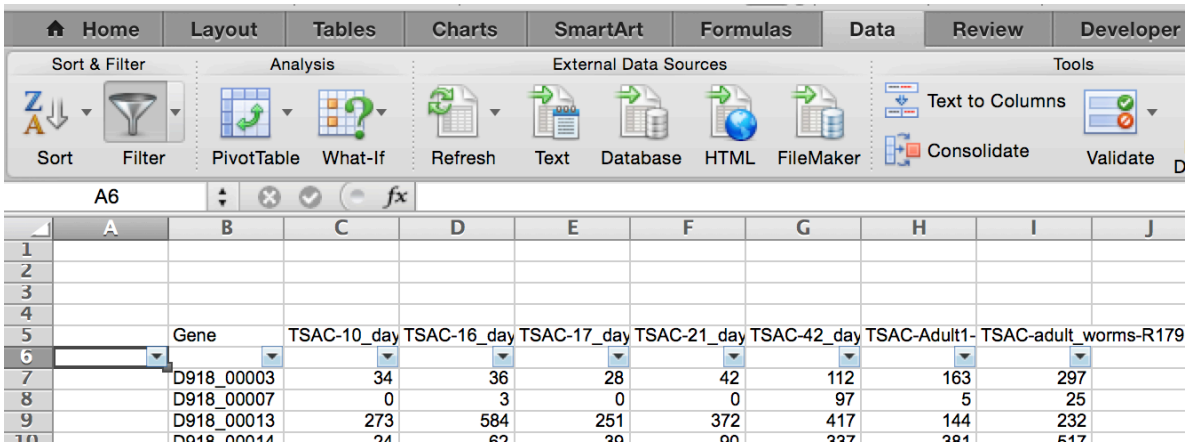
Working with large datasets

- Open ~/Desktop/WORKSHOP_RESOURCES/Section_2/module_3/Excel/ tsuis_rnaseq_htseq_countstable.txt, in Excel.
- This is a large table, with 9,833 rows and 8 columns, but we are going to add more columns as we build the database.
- If you hold down the “command” key on a Mac (⌘) or the “CTRL” key on Windows, and then scroll with your keyboard arrows, the selection will skip to the end of the table. This becomes essential for highlighting all of the cells in a column in a large table, since scrolling with the mouse can take several minutes.
- The first thing we will do is insert four empty rows above the dataset and one below the headers, in order to make room to add more detailed descriptions.
- To do this, right click on the number on the left-hand border, and choose “insert”. New columns or rows will enter above (rows) to the left (columns) of the insertion point.



Sorting data in Excel

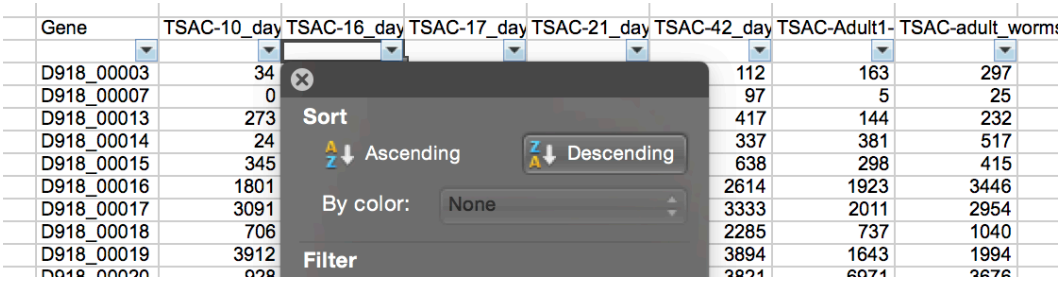
- The most important thing when working with these spreadsheets is to never sort the data incorrectly. Not only will all of the results be wrong, but it will be very difficult to tell that something went wrong.
- For this reason, you should never use “Data -> Sort” to sort your data. Instead, always use the “filter” feature.
- In this example, I am highlighting (selecting) the empty row below my headers and then clicking the funnel icon that says “Filter” below it (under the “Data” tab of the ribbon).



- Once this has been clicked, small grey arrows will appear in the row that was highlighted.

Sorting data in Excel

- When you click on these “sorting arrows”, you can choose to sort a column of your choice, either ascending or descending. All of the data that is underneath an arrow will sort with that data, every time. If you were to sort manually, it is up to you to select the entire dataset every time, so this is the safe option to ensure data integrity.



- Since we are going to add more data, we want the arrows to extend very far to the right of the spreadsheet, so that new data will also sort. Excel will only let you add the arrows to columns spanning any actual content, so scroll far to the right with the keyboard and add a space with the spacebar to a cell in row 6 (for example, in cell EA6). Then, hold shift and command/CTRL, and press left to scroll all the way back, highlighting all of the cells along the way. With the entire row selected, press the filter button in the “Data” tab of the ribbon.
- Now, as we add data to the table, all of it will be sortable and will stay organized.
- I do not recommend ever actually using the “Filter” functionality, since this hides rows from view.

Formatting headers

- Descriptive, organized headers are essential for keeping your data organized, communicating your data to others, and for keeping track of where results came from.

	Stage	L2	L3	L3	L4	L5	L5	L5
	Age (days)	10	16	17	21	42	Adult	Adult
Gene	Sample Name	TSAC-10_day	TSAC-16_day	TSAC-17_day	TSAC-21_day	TSAC-42_day	TSAC-Adult1-	TSAC-adult_worr
D918_00003		34	36	28	42	112	163	297
D918_00007		0	3	0	0	97	5	25
D918_00013		273	584	251	372	417	144	232

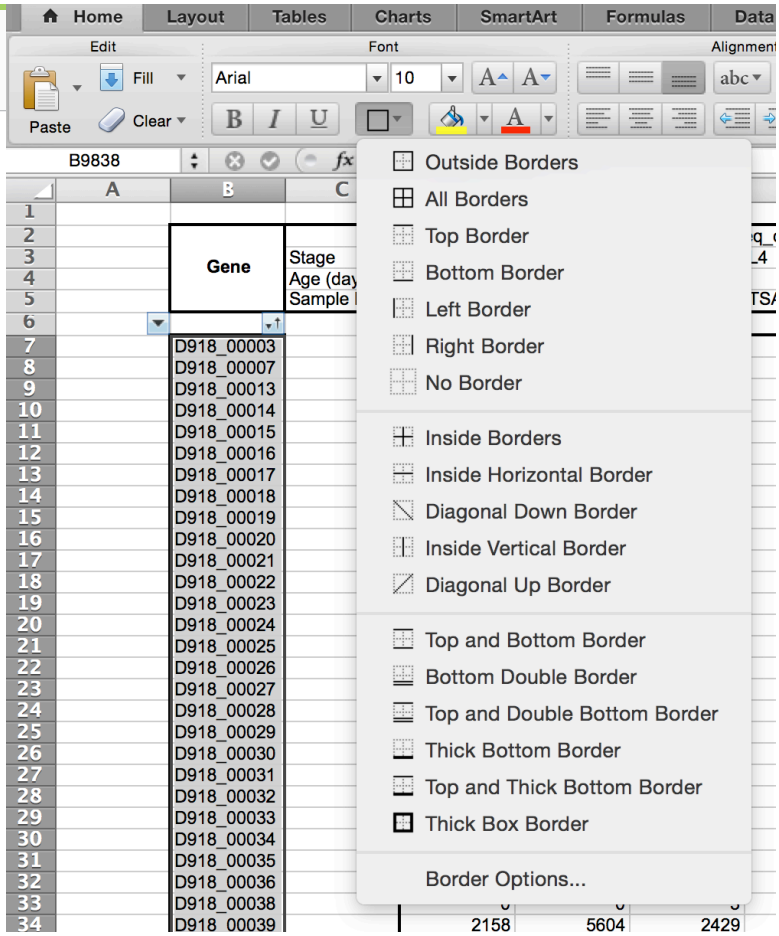
- Start by inserting a column before the read data, and adding row labels for the metadata. Always retain the original sample names from the raw data so that data can be compared in the future.
- Next, in cell C2, type "HTSeq output (tsuis_rnaseq_htseq_countstable.txt, Sept 11 2015)”, because this is a complete, descriptive header for this entire set of columns. Then highlight cells C2:J2, and click “Merge” under the “Home” tab of the ribbon:

	HTSeq output (tsuis_rnaseq_htseq_countstable.txt, Sept 11 2015)							
	Stage	L2	L3	L3	L4	L5	L5	L5
	Age (days)	10	16	17	21	42	Adult	Adult
Gene	Sample Name	TSAC-10_day	TSAC-16_day	TSAC-17_day	TSAC-21_day	TSAC-42_day	TSAC-Adult1-	TSAC-adult_worn
D918_00003		34	36	28	42	112	163	297
D918_00007		0	3	0	0	97	5	25
D918_00013		273	584	251	372	417	144	232

- This groups all of the columns together, while still allowing them to have separate descriptions. Each set of data with more than one column should be formatted this way to keep it as organized as possible.

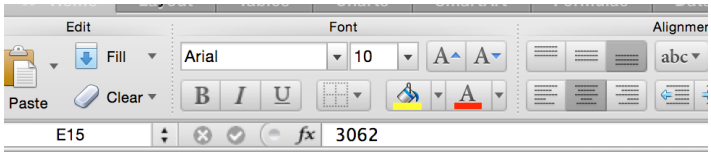
Formatting headers

- Use borders to box off the headers and the different sections of data. To do this, highlight a cell range, then click the borders box in the “home” section of the ribbon.
- For database tables, “Thick Box Borders” make it easier to read. For any table that is to be printed or published, the thinner “outside borders” look better.
- Reminder: Use Command/CTRL + shift and the arrow keys to highlight all of the data to the very bottom, to add borders to the entire data block.



Formatting headers

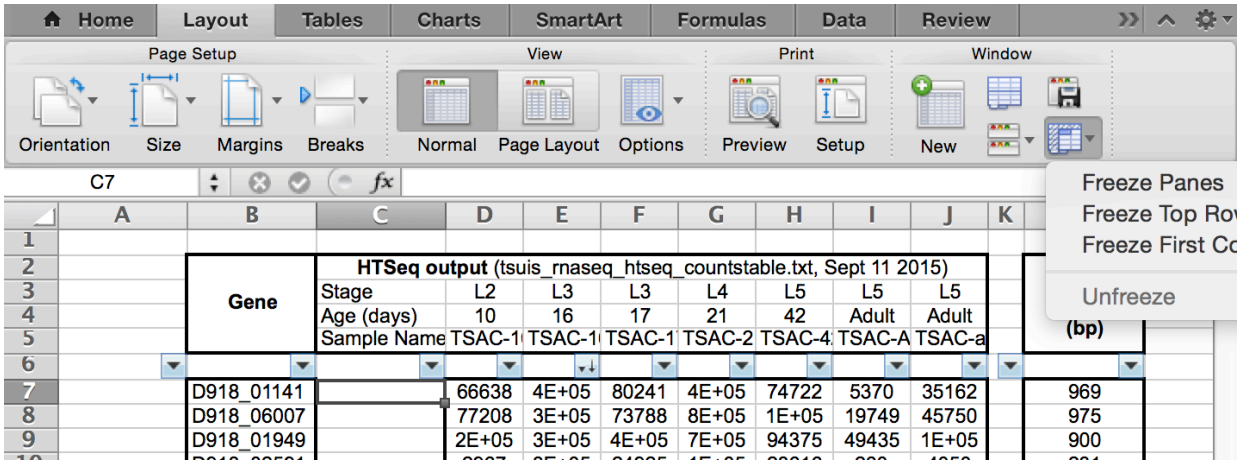
- Finally, highlight your data, and use the font settings in the ribbon to make it more readable.
- Choose Arial size 10 font, and center the data whenever it’s not in a long string format.
- Major headings can be bolded.
- Adjust the column widths by dragging from the edges of the column letters on the outside of the sheet, so that they only use as much space as needed.



Gene	HTSeq output (tsuis_rnaseq_htseq_countstable.txt, Sept 11 2015)						
Stage	L2	L3	L3	L4	L5	L5	L5
Age (days)	10	16	17	21	42	Adult	Adult
Sample Name	TSAC-1	TSAC-1	TSAC-1	TSAC-2	TSAC-4	TSAC-A	TSAC-a
D918_00003	34	36	28	42	112	163	297
D918_00007	0	3	0	0	97	5	25
D918_00013	273	584	251	372	417	144	232
D918_00014	24	62	39	90	337	381	517
D918_00015	345	615	488	404	638	298	415
D918_00016	1801	1672	3838	1870	2614	1923	3446
D918_00017	3091	3833	4334	4376	3333	2011	2954

Freezing panes

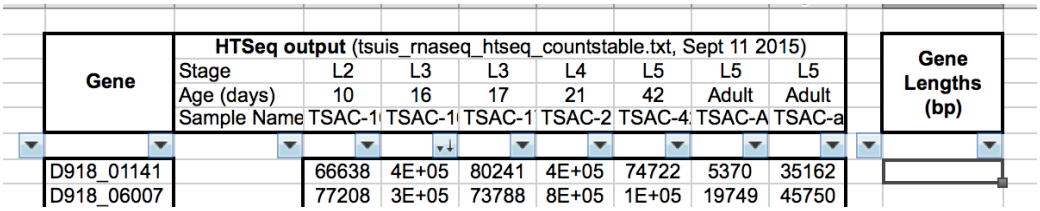
- Under “Layout”, and then “Freeze Panes”, you can choose to ‘freeze’ all of the rows above and all of the columns to the left of the currently selected cell.
- Doing this will lock the headers and gene names in place, so that when you scroll through the table, you will always be able to see this critical data.



Adding additional data: Gene Lengths

- We will use the gene lengths to calculate FPKM values from the raw counts table.
- First, open up “gene lengths.txt” from the Excel folder, select the entire table, and copy it to the clipboard.
- Now, go back to your main file and make a new “sheet” in Excel by clicking the + sign on beside the tabs at the bottom. Paste the data into this second sheet, so that it doesn’t paste mis-aligned into the main table.

- Add a header to your main table for where the new data will go.
- The “wrap text” font feature is helpful when the header name is long but the data will not be wide.



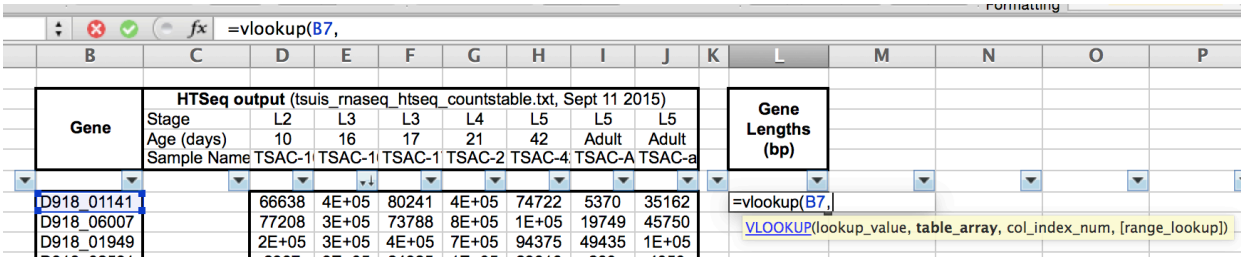
Why don’t we just sort the two tables by gene name and then copy and paste the data?

- Because even if the same *number* of genes is present, we can’t necessarily trust that every gene is present or entered in the same way.
- For example, in an updated genome draft, one gene can be removed and one new gene can be added. The genes at the start and ends of the table will match, but there will be mismatches for every gene in between these two. Any mistakes in a gene name will cause you reach false conclusions about your entire dataset.

Looking up data in Excel with =VLOOKUP

=VLOOKUP is one of the most useful formulas in Excel, and allows for looking up matching values in a Vertical reference list.
The syntax is:
= VLOOKUP ([Value to lookup], [Table containing the value in the first column], [column number to return], FALSE)

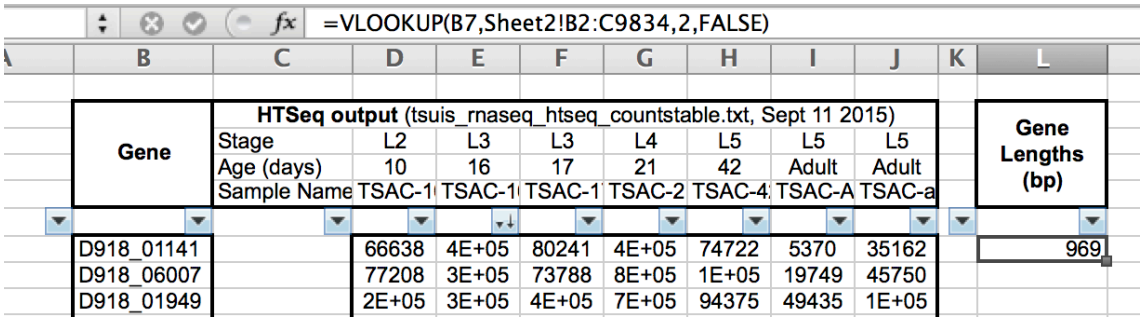
- In this case, we want to look up the gene length corresponding to each gene name in the main table. We will start with the first gene, which is in cell B7 in this example:



- Type "=VLOOKUP(B7," and then click to the second tab in your file containing the gene lengths. Highlight this entire table using Command/CTRL+Shift and the arrow keys, and then type a second comma. If you make a mistake doing this, just press escape and start over. Then, click back to your main table, and finish the formula with "2" and "FALSE" as the last two entries.

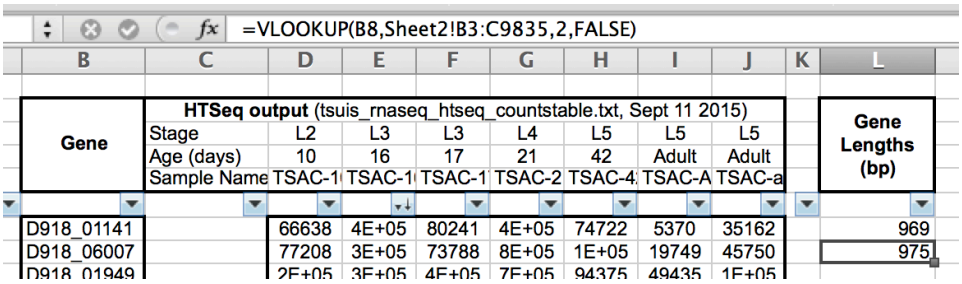
Looking up data in Excel with =VLOOKUP

- This formula now identifies the gene length of the first gene (in cell B7) by referencing the table in Sheet 2, cells B2:C9834, by matching the gene name in the first column and returning the value in the second column. The last value of "FALSE" is necessary because "TRUE" will allow approximate matches. This should always be false in all cases for any scientific work.

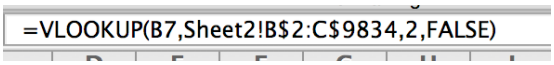


Copying and pasting formulas in Excel

- Copy and paste the VLOOKUP formula to the cell below it, to look up the value of the second gene. You can right click or use the menus to do this, but I recommend getting used to Command/CTRL+C and Command/CTRL+V to do this.
- Note that in Excel, if you copy and paste a formula down one row, all of the cell references in the formula also move by one row (also with columns). Here, we are now looking up cell B8, to get the value for the second gene instead of the first.
- While this is useful, we have to be careful, because the cell references for the **lookup table** of gene lengths (in sheet 2) has also moved down (from B2:C9834 to B3:C9835).

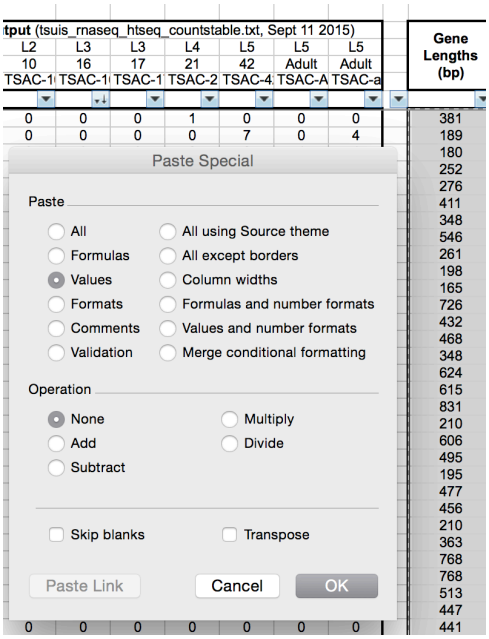


- In order to fix this, we can use $ signs to “lock” the row references in place for the lookup table.
- Any column letter or row number with a $ in front of it will not change when the formula is copied and pasted.
- Return to the first formula cell and change the reference to B$2:C$9834, and paste that down.



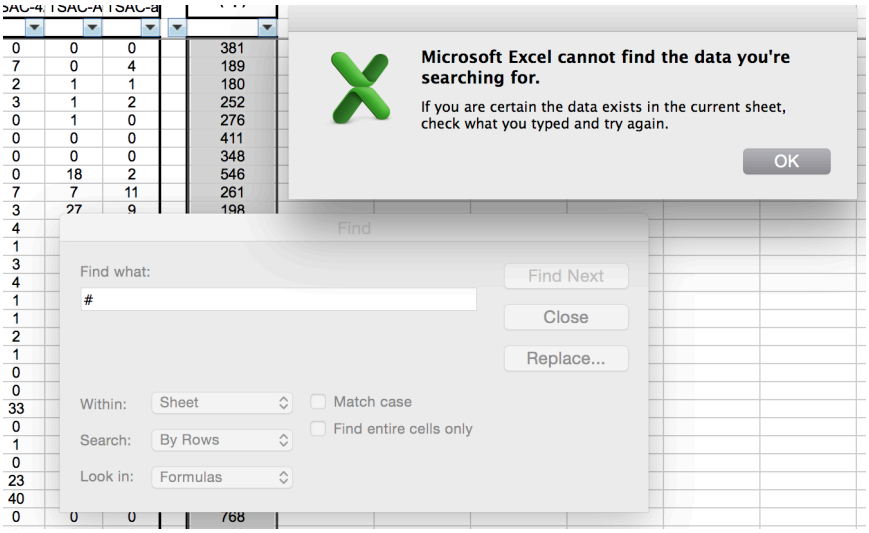
Filling and ‘clearing’ formulas

- We need to paste the formula down the entire column.
- Copy the formula, then scroll to the bottom of the table by command/CTRL+down on one of the gene count columns.
- Starting at the bottom of the ‘gene lengths’ column, hold shift and command/CTRL and press up, to highlight the entire column. Then, paste with command/CTRL+V.
- Now we have aligned all of the gene lengths.
- The formulas are still “active” and will re-calculate every time the table is sorted or the file is saved. Enough of these active formulas will cause the spreadsheet to slow down or crash eventually.
- We will therefore “clear” the formulas, leaving their values behind.
- To do this, highlight the entire column and copy (command/CTRL+C), and then within the copied cells, right click and choose “paste special”.
- In the “Paste special” dialog, choose “values” and then click “ok”.



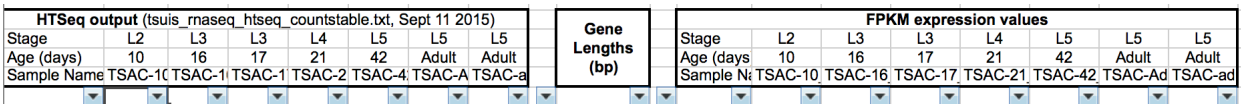
Checking for formula errors

- Formulas in Excel can return errors. In the case of =VLOOKUP, if there is no lookup value in the reference table, it will return '#N/A', indicating that there is no match in the lookup table.
- All errors start with a # sign, so they can be searched easily.
- After clearing the formulas (previous slide), highlight the column and press command/ CTRL+F to search.
- If there is no match in this search, then all of the genes were matched up and there is no problem.

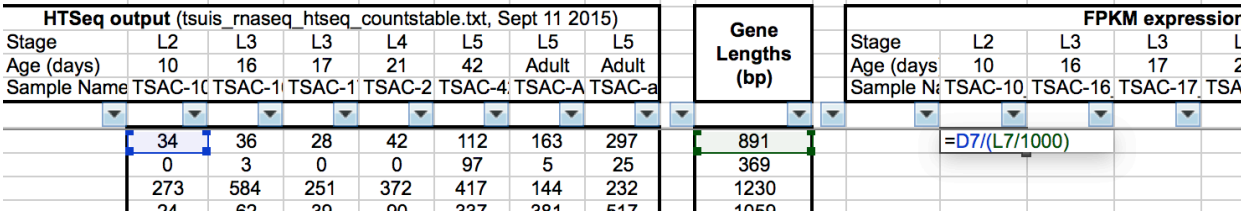


Calculating FPKM values

- We can now calculate FPKM expression values from the raw read counts. Start by copying and pasting the read count headers to the right of the gene lengths, and change the title of the new header set:



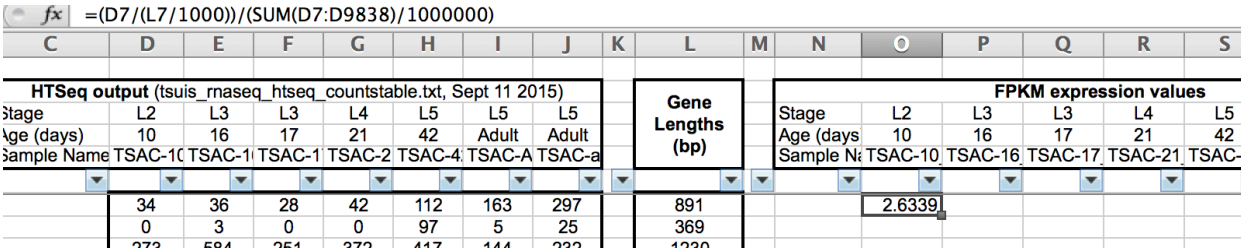
- FPKM = Fragments (counts from HTSeq) Per Kilobase (gene length / 1000) per Million of reads mapped (the total read count in the sample's column in the HTSeq data).
- This gene expression measure is used because it is normalized both for the gene length and the library size, making the values directly comparable across the entire dataset, and between different experiments.
- We can calculate all of this in a single formula. Start by dividing by the count by the gene length as shown below:



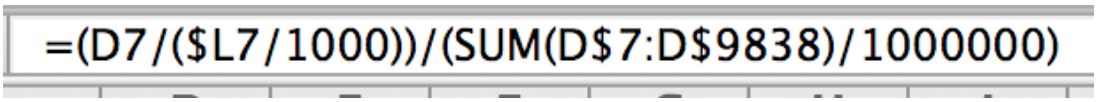
- Using parentheses organizes the formula to ensure that the order of operations is correct (i.e., we are not dividing D7 by L7 first, and then dividing by 1000).

Calculating FPKM values

- Now all of this needs to be divided by (the library size / 1,000,000). So put the entire existing formula in parentheses, and then divide by (the sum of the sample's column / a million):

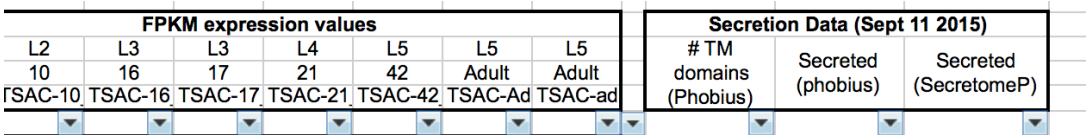


- Verify this value to ensure that the formula is typed correctly (D918_00003 in L2 = 2.6339).
- We need to lock several things in place in order to copy and paste for the entire table. First, the reference to L7 (the gene length) needs to move down, but not left-to-right, so put a $ sign in front of the **L** but not the **7**.
- Second, the "sum" range needs to be locked to the rows but not the columns. So change that to **D$7:D$9838**, so that the columns move with the formula.
- The final formula should look like this:

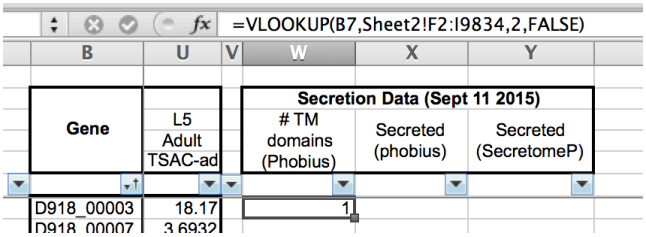


Aligning additional data

- Copy and paste this formula for the entire FPKM table, and then clear the formulas and check for errors as shown previously.
- This normalized data will later be used as input for hierarchical clustering (in R), but for now we will continue building the database.
- Open "secretion data.txt" in the "Excel" directory, and paste into the second sheet of your database file as before.
- This data is output from two different programs (Phobius and SecretomeP)
- Create headers for the data in your main table:



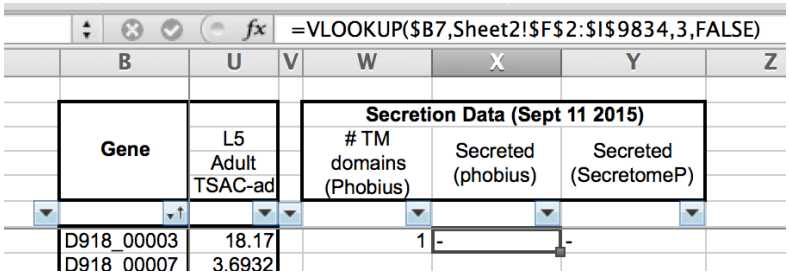
- Set up the =VLOOKUP formula for the first row and column:



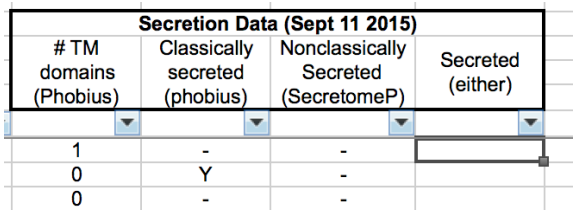
- This data needs to be pasted both down and to the right. Using $ signs, lock the column of the gene name, and the entire table: =VLOOKUP($B7,Sheet2!$F$2:$I$9834,2,FALSE)

Aligning additional data

- When pasting to the right, we also need to change the “2” to a “3” in the formula, to return the value of the third column in the lookup table instead of the second.
- Also change this value to a “4” in the last column. Then, copy all three values and paste down for the entire table, clear formulas, and check for errors.

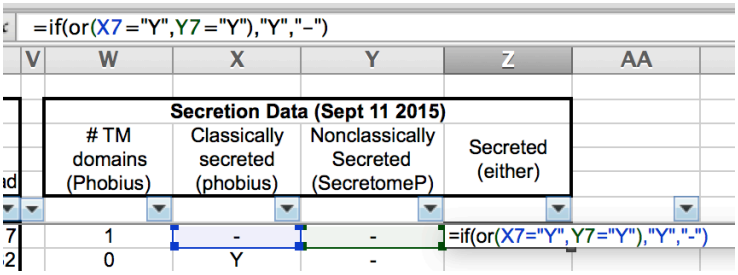


- Now we will add an additional column, to indicate if each gene is secreted **either** by classical or nonclassical secretion. This should be a “Y” if either of the other two columns are a “Y”. We will use an =IF statement to perform this.

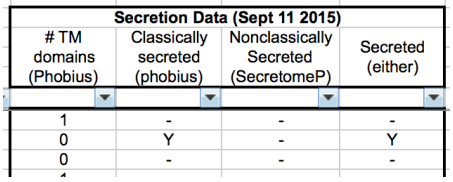


=IF formula

=IF is a very useful Excel formula for parsing data. The syntax is:
=IF([A logical test returning true or false, usually =, <, >, or =>, <=], [value if true], [value if false])
- So for example, try entering =IF(1=2,”Yes”,”No”).
- This will return “No” in the cell, because the ‘logical test’ is false. If you change this to 1=1, then it will return “Yes”.
- Here, we need to check whether either of the cells beside the new column are “Y”. In order to accomplish this we will use OR() in the logical test:

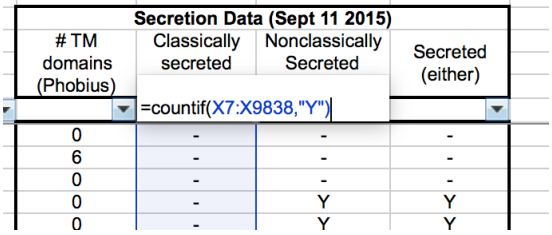


- Copy and paste this formula, clear values, and check for errors before moving on.



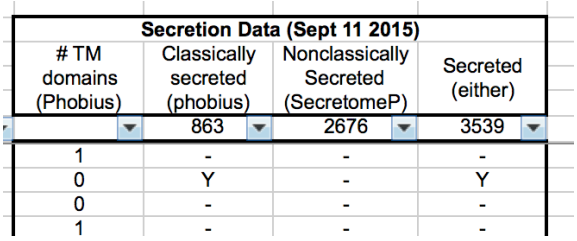
=COUNTIF formula

- In the empty 'sorting' row below your secretion header, use the =COUNTIF formula to count how many genes are secreted according to each criteria.
=COUNTIF([range of cells to count], [criteria for counting])



Secretion Data (Sept 11 2015)			
# TM domains (Phobius)	Classically secreted	Nonclassically Secreted	Secreted (either)
	=countif(X7:X9838,"Y")		
0	-	-	-
6	-	-	-
0	-	-	-
0	-	Y	Y
0	-	Y	Y

- Here, we are counting how many "Y" values there are in the column. Paste this to the right to count for each criteria:

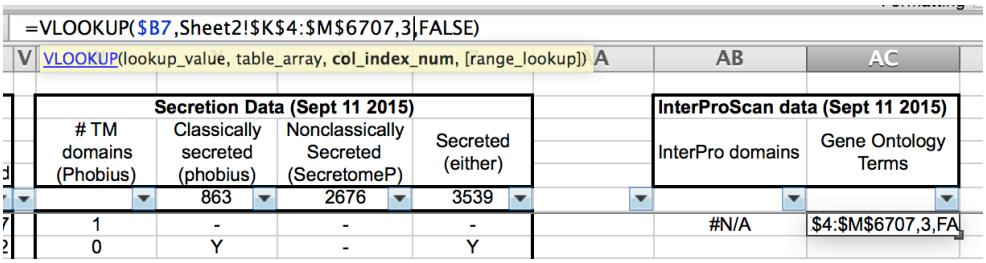


Secretion Data (Sept 11 2015)			
# TM domains (Phobius)	Classically secreted (phobius)	Nonclassically Secreted (SecretomeP)	Secreted (either)
	863	2676	3539
1	-	-	-
0	Y	-	Y
0	-	-	-
1	-	-	-

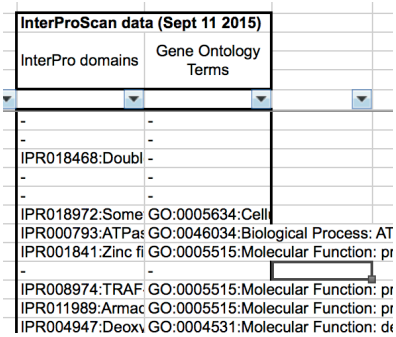
- This is an easy way to summarize your data. You can also check if values are greater than zero (">0"), if values are larger than the value in another cell, etc.
- =COUNTIFS (with an S) can check multiple criteria in multiple columns.

Annotation data (lookup with missing values)

- Open "interproscan_annotations_per_gene.txt" from the "Excel" file, and copy and paste into the second sheet as before.
- Prepare the headers and use =VLOOKUP as before:



- This time there is an #N/A value because the lookup table does not contain unannotated genes. Paste the formulas through, and then clear the formulas.
- Now, replace the #N/A values with "-", to clean up the table.
- When long strings "hang" over into the next cell, add an empty space in the column to the right, to cover it up:



InterProScan data (Sept 11 2015)	
InterPro domains	Gene Ontology Terms
-	-
-	-
IPR018468:Doubl	-
-	-
-	-
IPR018972:Some	GO:0005634:Cell
IPR000793:ATPas	GO:0046034:Biological Process: AT
IPR001841:Zinc fi	GO:0005515:Molecular Function: pr
-	-
IPR008974:TRAF	GO:0005515:Molecular Function: pr
IPR011989:Armac	GO:0005515:Molecular Function: pr
IPR004947:Deoxy	GO:0004531:Molecular Function: de

Parsing DESeq results

- Now we will add the DESeq results we calculated in RStudio.
- Open the “Comparison1_Early_vs_Late_tsuis_deseq2_output.txt” file in the DESeq folder, and paste it into the second sheet of the dataset as before.
- First, note that the headers are all shifted to the left by 1 column. Cut and paste those to the right to fix this. This problem commonly occurs with R output (row.names has no header entry), so always be sure to check for an empty final column.

	baseMean	log2FoldChan	lfcSE	stat	pvalue	padj	
D918_00003	102.244975	-2.2852271	0.54219132	-4.2147983	2.50E-05	0.00019566	
D918_00007	13.3896063	-4.7819266	1.08457881	-4.4090172	1.04E-05	8.68E-05	
D918_00013	310.784483	0.74493719	0.37667336	1.9776742	0.04796547	0.12076386	

	baseMean	log2FoldChan	lfcSE	stat	pvalue	padj
D918_00003	102.244975	-2.2852271	0.54219132	-4.2147983	2.50E-05	0.00019566
D918_00007	13.3896063	-4.7819266	1.08457881	-4.4090172	1.04E-05	8.68E-05
D918_00013	310.784483	0.74493719	0.37667336	1.9776742	0.04796547	0.12076386

From the DESeq manual:

The interpretation of the columns of *data.frame* is as follows.

`id`	feature identifier
`baseMean`	mean normalised counts, averaged over all samples from both conditions
`baseMeanA`	mean normalised counts from condition A
`baseMeanB`	mean normalised counts from condition B
`foldChange`	fold change from condition A to B
`log2FoldChange`	the logarithm (to basis 2) of the fold change
`pval`	p value for the statistical significance of this change
`padj`	p value adjusted for multiple testing with the Benjamini-Hochberg procedure (see the R function `p.adjust`), which controls false discovery rate (FDR)

Parsing DESeq results

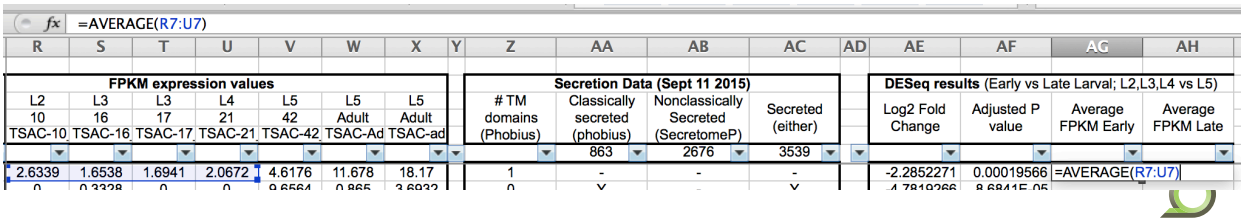
- We are only interested in the Log2 Fold Change and Adjusted P value, so delete the other columns by right-clicking the column letters on the border and deleting them:

	log2FoldChan	padj
D918_00003	-2.2852271	0.00019566
D918_00007	-4.7819266	8.68E-05
D918_00013	0.74493719	0.12076386

- Set up these headers in the main sheet, and perform the VLOOKUP for these values, then add two headers, for the average FPKM values from the two sample groups:

DESeq results (Early vs Late Larval; L2,L3,L4 vs L5)			
Log2 Fold Change	Adjusted P value	Average FPKM Early	Average FPKM Late
-2.2852271	0.00019566		
-4.7819266	8.6841E-05		
0.74493719	0.12076386		

- Use =AVERAGE to calculate the average value of the sample groups, then paste the formulas down and clear the formulas.



Parsing DESeq results

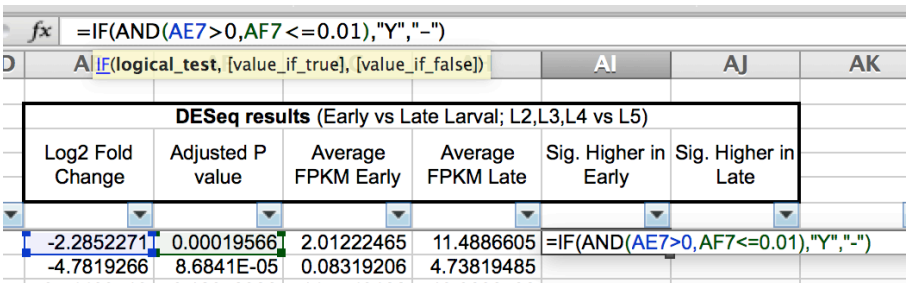
- We want to know whether each gene is significantly differentially expressed in either early larval or late larval stages. Start by setting up additional headers:

DESeq results (Early vs Late Larval; L2,L3,L4 vs L5)					
Log2 Fold Change	Adjusted P value	Average FPKM Early	Average FPKM Late	Sig. Higher in Early	Sig. Higher in Early
-2.2852271	0.00019566	2.01222465	11.4886605		
-4.7819266	8.6841E-05	0.08319206	4.73819485		
0.74493719	0.12076386	14.7543182	10.0696598		
-2.7988517	7.828E-07	2.41819608	20.4226954		

- We can see that a negative fold change corresponds to a gene that is higher in the late stages than the early stages (and vice versa for a positive value).
- Therefore, in order to call a gene significantly higher in the early stages: (a) the fold change value needs to be greater than zero, and (b) the P value needs to be less than a threshold value of your choice.
- DESeq recommends a maximum threshold P value of 0.1, but we will parse more conservatively, at 0.01 instead.
- For a very high-confidence small gene set, a threshold of 10^{-5} could be used.
- Generally, 0.05, 0.01, or 10^{-5} are used for publications.
- Fold change thresholds should **not** be used for RNA-Seq data. There is justification for it with microarrays, but the high sensitivity of RNA-Seq data (and high abundance of zero values) invalidates its use for statistical cutoffs.

Parsing DESeq results

- For the first column, use an =IF statement with an “AND” function to check whether both (a) the Fold change value is greater than zero and (b) the P value is less than or equal to 0.01:

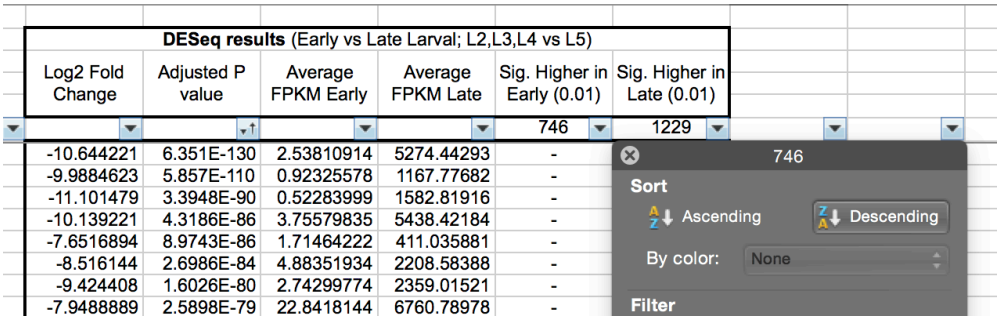


- Repeat for the second column, but check if the fold change is *less than* zero for it. Then paste the two columns down, clear the formulas, and check for errors.
- Paste the =COUNTIF formula from the secretion columns to count the differentially expressed genes. Note that this doesn’t match the RStudio summary because we are using a different threshold; At a 0.1 threshold, the counts do match.

DESeq results (Early vs Late Larval; L2,L3,L4 vs L5)					
Log2 Fold Change	Adjusted P value	Average FPKM Early	Average FPKM Late	Sig. Higher in Early (0.01)	Sig. Higher in Late (0.01)
				746	1229
-2.2852271	0.00019566	2.01222465	11.4886605	-	Y
-4.7819266	8.6841E-05	0.08319206	4.73819485	-	Y
0.74493719	0.12076386	14.7543182	10.0696598	-	-

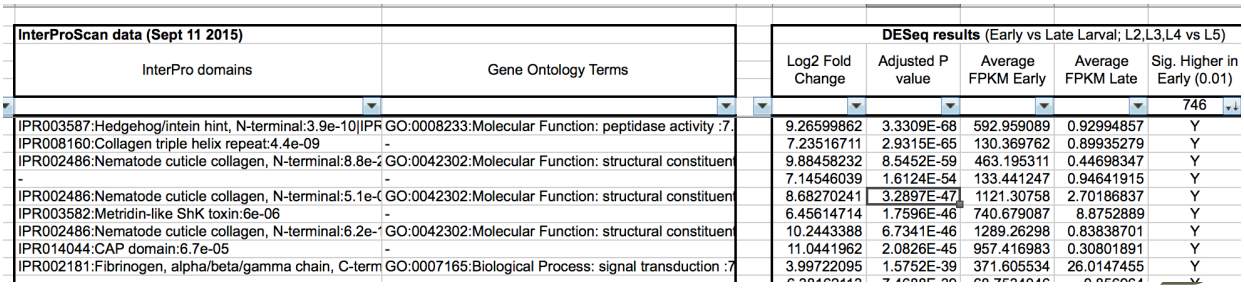
Analyzing data

- Look at the most significantly differentially expressed genes by sorting by P value (A->Z), and then by one of the two categories (Z -> A):



DESeq results (Early vs Late Larval; L2,L3,L4 vs L5)					
Log2 Fold Change	Adjusted P value	Average FPKM Early	Average FPKM Late	Sig. Higher in Early (0.01)	Sig. Higher in Late (0.01)
				746	1229
-10.644221	6.351E-130	2.53810914	5274.44293	-	
-9.9884623	5.857E-110	0.92325578	1167.77682	-	
-11.101479	3.3948E-90	0.52283999	1582.81916	-	
-10.139221	4.3186E-86	3.75579835	5438.42184	-	
-7.6516894	8.9743E-86	1.71464222	411.035881	-	
-8.516144	2.6986E-84	4.88351934	2208.58388	-	
-9.424408	1.6026E-80	2.74299774	2359.01521	-	
-7.9488889	2.5898E-79	22.8418144	6760.78978	-	

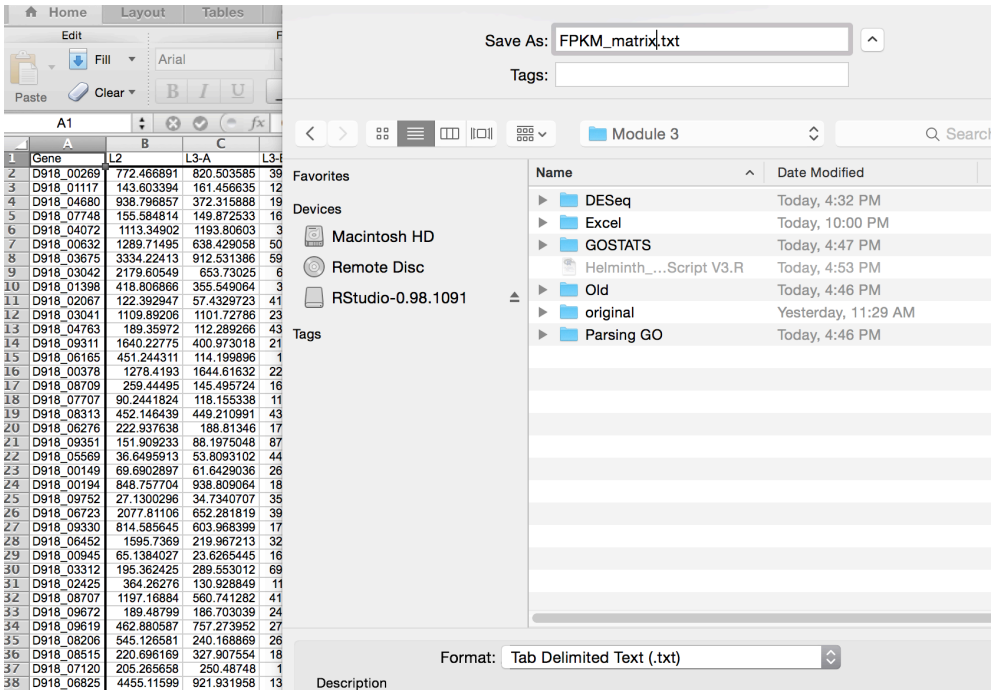
- Scroll to the left to see the the InterProScan annotation data, which gives information on the functions of these most significant genes:



InterProScan data (Sept 11 2015)		DESeq results (Early vs Late Larval; L2,L3,L4 vs L5)				
InterPro domains	Gene Ontology Terms	Log2 Fold Change	Adjusted P value	Average FPKM Early	Average FPKM Late	Sig. Higher in Early (0.01)
						746
IPR003587:Hedgehog/intein hint, N-terminal:3.9e-10|IPR	GO:0008233:Molecular Function: peptidase activity :7.	9.26599862	3.3309E-68	592.959089	0.92994857	Y
IPR008160:Collagen triple helix repeat:4.4e-09	-	7.23516711	2.9315E-65	130.369762	0.89935279	Y
IPR002486:Nematode cuticle collagen, N-terminal:8.8e-2	GO:0042302:Molecular Function: structural constituent	9.88458232	8.5452E-59	463.195311	0.44698347	Y
-	-	7.14546039	1.6124E-54	133.441247	0.94641915	Y
IPR002486:Nematode cuticle collagen, N-terminal:5.1e-(	GO:0042302:Molecular Function: structural constituent	8.68270241	3.2897E-47	1121.30758	2.70186837	Y
IPR003582:Metridin-like ShK toxin:6e-06	-	6.45614714	1.7596E-46	740.679087	8.8752889	Y
IPR002486:Nematode cuticle collagen, N-terminal:6.2e-1	GO:0042302:Molecular Function: structural constituent	10.2443388	6.7341E-46	1289.26298	0.83838701	Y
IPR014044:CAP domain:6.7e-05	-	11.0441962	2.0826E-45	957.416983	0.30801891	Y
IPR002181:Fibrinogen, alpha/beta/gamma chain, C-term	GO:0007165:Biological Process: signal transduction :7	3.99722095	1.5752E-39	371.605534	26.0147455	Y

Saving data for clustering and functional enrichment testing

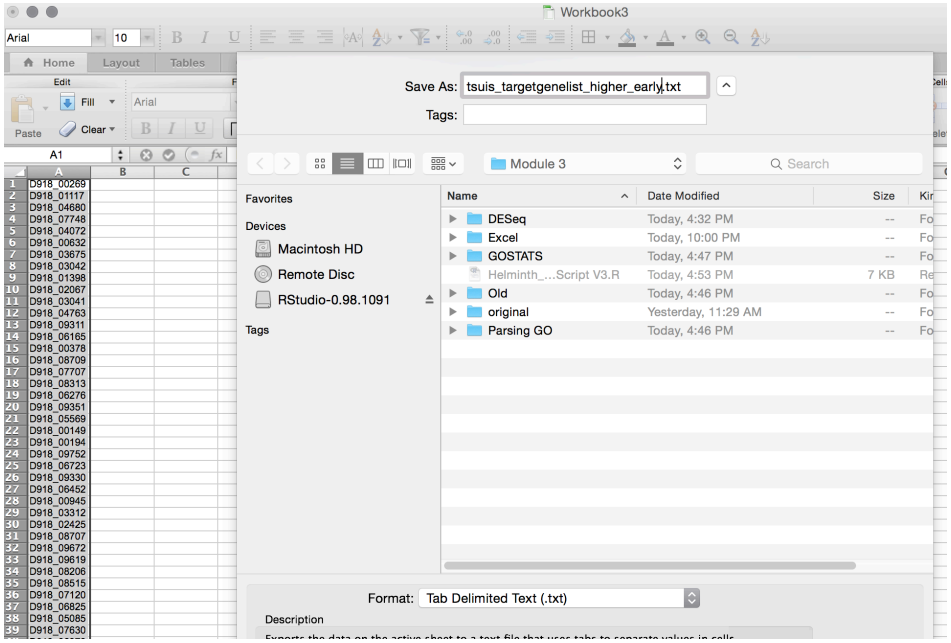
- For clustering, copy and paste the gene names and the FPKM values for each sample into a new spreadsheet, then save as a tab-delimited text file. Renaming the long sample names to shorter IDs will make the final cluster look nicer:



Gene	L2	L3-A	L3-B
D918_00269	772.466891	820.503585	39
D918_01117	143.603394	161.456635	12
D918_04680	938.796857	372.315888	19
D918_07748	155.584814	149.872533	16
D918_04072	1113.34902	1193.80603	3
D918_00632	1289.71495	638.429058	50
D918_03675	3334.22413	912.531386	59
D918_03042	2179.60549	653.73025	6
D918_01398	418.806866	355.549064	3
D918_02067	122.392947	57.4329723	41
D918_03041	1109.89206	1101.72786	23
D918_04763	189.35972	112.289266	43
D918_09311	1640.22775	400.973018	21
D918_06165	451.244311	114.199896	1
D918_00378	1278.4193	1644.61632	22
D918_08709	259.44495	145.495724	16
D918_07707	90.2441824	118.155338	11
D918_08313	452.146439	449.210991	43
D918_06276	222.937638	188.81346	17
D918_09351	151.909233	88.1975048	87
D918_05569	36.6495913	53.8093102	44
D918_00149	69.6902897	61.6429036	26
D918_00194	848.757704	938.809064	18
D918_09752	27.1300296	34.7340707	35
D918_06723	2077.81106	652.281819	39
D918_09330	814.585645	603.968399	17
D918_06452	1595.7369	219.967213	32
D918_00945	65.1384027	23.6265445	16
D918_03312	195.362425	289.553012	69
D918_02425	364.26276	130.928849	11
D918_08707	1197.16884	560.741282	41
D918_09672	189.48799	186.703039	24
D918_09619	462.880587	757.273952	27
D918_08206	545.126581	240.168869	26
D918_08515	220.696169	327.907554	18
D918_07120	205.265658	250.48748	1
D918_06825	4455.11599	921.931958	13

Saving data for clustering and functional enrichment testing

- For functional enrichment, we will need a “target” gene list of differentially expressed genes. In the interest of time, we will just save the “higher in early” gene list. Sort the spreadsheet by that column, then copy and paste all of the genes with “Y” values into a new file, then save as a tab delimited text with no headers:



PCA from DESeq results

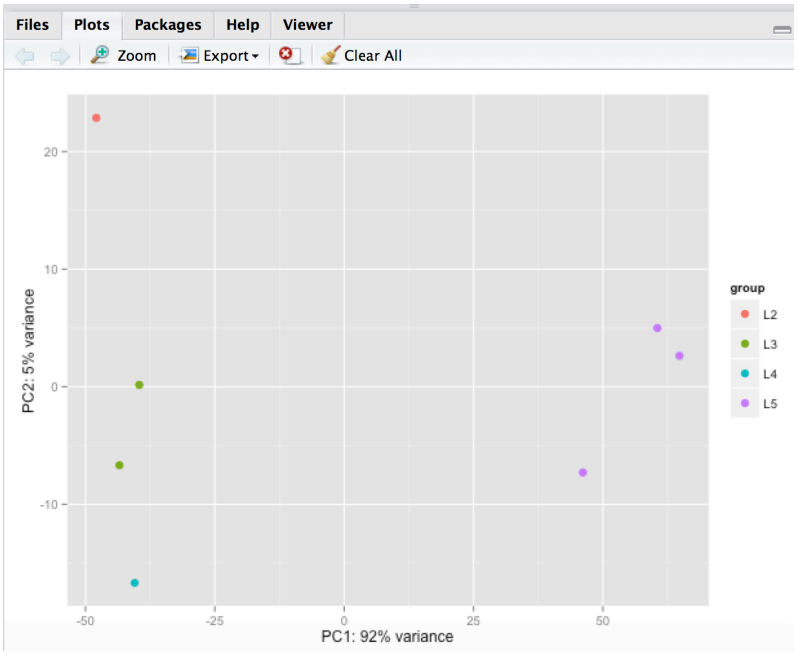
- Principal component analysis (PCA) is one approach for visualizing how expression patterns vary across samples.
- Go back to R and find the PCA code section.
- DESeq has a built-in tool for running PCA that utilizes the dds object created earlier.

```
#######
# PCA #
#######

#Log transform deseq object data
rld <- rlogTransformation(dds, blind=TRUE)

#Perform and plot PCA based on data from top X expressed genes (default 500)
plotPCA (rld, intgroup=c("Stage"), ntop=500)
```

- These commands log transform the data, and then plot the PCA.
- Note that “intgroup” can be any column of the metadata file. Here we use “stage” to give more detail on each sample, as opposed to just the two categories in “Comparison1”.
- “ntop” defines the number of genes to use to calculate the PCA. Using too many low-information genes may add noise to the clustering. The default is 500, but the results are generally not sensitive to changing the number.

PCA from DESeq results

- After running these commands, the PCA plot will show up in the bottom-right panel.
- Clicking “Export” will allow you save this file. If you save as a PDF, you can edit the plot directly in a vector-based image editing program (Adobe Illustrator, or “Inkscape”, which is free).
- We will also export the plot co-ordinates so that the data can be replotted in Excel later.



PCA from DESeq results

- The following code will save the PCA coordinates into a file so that the data can be graphed in other programs, and outputs the variance of each component, including those not shown on the plot.

```
#Output PCA coordinates
PCAcoordinates<-plotPCA (rld, intgroup=c("Stage"), ntop=500,returnData = TRUE)
write.table(PCAcoordinates, file="tsuis_PCA_coordinates", sep="\t")

#Output variance per component
rlogMat <- assay(rld)
rv = apply(rlogMat, 1, var)
select = order(rv, decreasing=TRUE)[seq_len(min(500, length(rv)))]
pca = prcomp(t(rlogMat[select,]))
sink(file="tsuis_PCA_variances_per_component.txt") #Define output file
summary(pca)
sink(NULL)
```

Hierarchical clustering in RStudio

- PCA was calculated directly from the DEseq dataset, but we will use FPKM values for hierarchical clustering.
- Run this code to load libraries and prepare the input files:

```
#############
# CLUSTERING #
#############

setwd("~/Desktop/Workshop/Module 3/")

#Load libraries
library("ape") #for clustering
library("amap") #for clustering

#Input file - Usually FPKM values per gene, genes down the r
x <- read.delim("FPKM_matrix.txt",header=TRUE, row.names=1)
#Transpose matrix (clustering takes genes in columns, sample
x <-t(x)
```

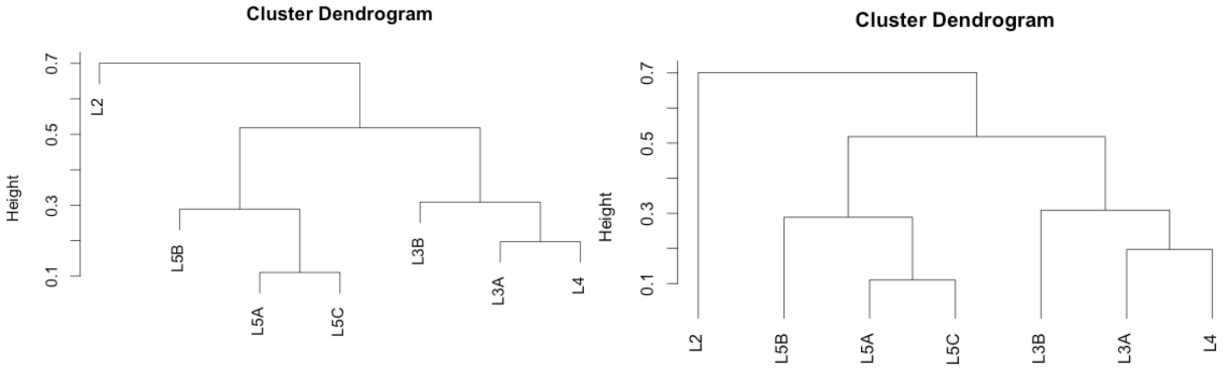
- If there is an error, check that the file names match.
- Next, we create a distance matrix. The statistic specified here determines the clustering algorithm. Pearson or Spearman correlation is typically used for RNA-Seq data, and "average" linkage is typically best for drawing the clusters:

```
#Create distance matrix, can use different clustering methods here instead (pearson, sp
dist.mat<-Dist(x,method="pearson", diag = FALSE, upper = FALSE)
#Cluster distance matrix, can use different clustering methods (average, complete, sing
cluster=hclust(dist.mat, method = "average", members=NULL)
```

Hierarchical clustering in RStudio

- The script includes two approaches for viewing the clustering:

```
#Plot the cluster results with actual distances
plot(cluster)
#Plot the cluster with equal distances for each sample
plot(cluster,hang=-1)
```



- You can export one or both of these as PDF for future reference.
- Finally, the script exports a newick-format file for input into other clustering programs (e.g. FigTree or ITOL):

```
#Optional: Convert cluster plot to newick cluster file format for input to other s
my_tree <- as.phylo(cluster)
write.tree(phy=my_tree, file="tsuis_rnaseq_clustering_pearson_average.newick")
```

Functional enrichment using GOSTATS in RStudio

- Run the following to prepare the GO database:

```
################################
# Functional Enrichment (GOstats) #
################################

setwd("~/Desktop/Workshop/Module 3/GOSTATS/")

#Load necessary libraries
library("GOstats")
library("GSEABase")
library("org.Hs.eg.db")

#Input gene to GO file (tab delimited, three columns: go_ID, evidence [always "IEA"], gene_ID)
genetogo=read.table("GO_to_geneID.txt", sep="\t",header=TRUE)

#Process input GO file

goframeData = data.frame(genetogo)
goFrame=GOFrame(goframeData,organism="Trichuris suis")
goAllFrame=GOAllFrame(goFrame)
gsc <- GeneSetCollection(goAllFrame, setType = GOCollection())
frame = toTable(org.Hs.egGO)
```

- “Go_to_geneID.txt” is a pairwise GO and Gene list, generated from InterProScan output in a different module.
- Producing this file is the difficult part about running enrichment on a custom genome. Most tools (including GOSTATS) are designed to be easy to use primarily for model organisms.

Functional enrichment using GOSTATS in RStudio

- Here we will input the complete (background) *T. suis* gene set, and our shorter target gene set that we saved from Excel, based on the DESeq output:

```
#Input full gene list and test gene list (no header, just single-column gene name lists)
universe=read.table("tsuis_full_gene_list.txt", sep="\t",header=FALSE)
testgenes=read.table("../tsuis_targetgenelist_higher_early.txt", sep="\t",header=FALSE)
```

- The remaining code runs the enrichment test and produces output. It is ran three times, one for Biological Process (BP), one for Molecular Function (MF) and one for Cellular Component (CC) Gene Ontology terms. Run all of this code to produce the three output files:

```
#BP
params <- GSEAGOHyperGParams(name="My Custom GSEA based annot Params",
geneSetCollection=gsc,
geneIds = testgenes,
universeGeneIds = universe,
ontology = "BP",
pvalueCutoff = 1,
conditional = FALSE,
testDirection = "over")
Over <- hyperGTest(params)
write.table(summary(Over), file="GOstats_output_BP.txt", sep="\t")
```

Manual False Discovery Rate (FDR) correction

- Open the “GOSTATS_output_MF.txt” file in the GOSTATS folder (Using Excel).
- As with DESeq output, shift the headers to the right by 1 column:

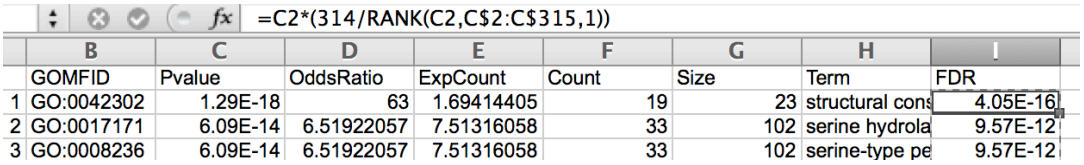
	GOMFID	Pvalue	OddsRatio	ExpCount	Count	Size	Term
1	GO:0042302	1.29E-18	63	1.69414405	19	23	structural constituent of cuticle
2	GO:0017171	6.09E-14	6.51922057	7.51316058	33	102	serine hydrolase activity
3	GO:0008236	6.09E-14	6.51922057	7.51316058	33	102	serine-type peptidase activity
4	GO:0004252	1.46E-13	6.78484848	6.85023465	31	93	serine-type endopeptidase activity
5	GO:0008233	4.85E-13	3.62266637	20.2560702	56	275	peptidase activity
6	GO:0070011	1.66E-11	3.41863045	19.5194858	52	265	peptidase activity, acting on L-ami
7	GO:0005198	2.16E-11	3.90359153	14.3633952	43	195	structural molecule activity
8	GO:0004175	2.39E-11	3.75228763	15.5419302	45	211	endopeptidase activity
9	GO:0003735	5.37E-06	3.36456279	8.32340339	23	113	structural constituent of ribosome

- The list is sorted by P value, with the most significant terms at the top. However, these P values are not population-corrected, and this must be done manually for GOSTATS.
- We need to do correction because there are multiple tests being performed. A 5% chance of being false is not acceptable when performing hundreds of tests.
- Generally, FDR correction is preferred for multiple-testing because it is a reasonable balance of stringency. The most stringent approach is Bonferroni correction (multiplying P values by the number of tests).

- For FDR, the most significant P value is multiplied by the number of tests. The second-most significant P value is multiplied by the number of tests divided by two. The third-most significant P value is multiple by the number of tests divided by three, etc.

Manual False Discovery Rate (FDR) correction

- This output file contains 314 tests. So the P values need to recalculated according to:
  P value * (314 / [rank of P value])
- We can accomplish this using the =RANK formula in Excel:
=RANK([value], [range of all values], [0 = Largest first, 1 = Smallest First])



	B	C	D	E	F	G	H	I
	GOMFID	Pvalue	OddsRatio	ExpCount	Count	Size	Term	FDR
1	GO:0042302	1.29E-18	63	1.69414405	19	23	structural cons	4.05E-16
2	GO:0017171	6.09E-14	6.51922057	7.51316058	33	102	serine hydrola	9.57E-12
3	GO:0008236	6.09E-14	6.51922057	7.51316058	33	102	serine-type pe	9.57E-12

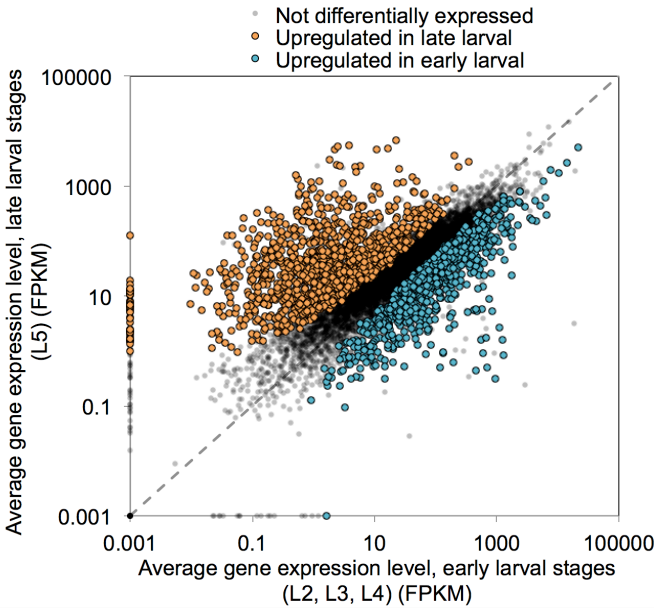
- The formula shown will calculate FDR-corrected P values in column I. The threshold value (0.01) will be applied on these FDR values.
- Some additional formatting will clean up the table and make it ready for publication:

Table 1: Molecular Function Gene Ontology terms significantly enriched among genes upregulated in early larval stages compared to late stages

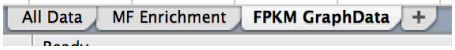
GO ID	Term Description	Gene Counts			FDR-corrected P
		Expected	Observed	Total	
GO:0042302	structural constituent of cuticle	1.7	19	23	4.05E-16
GO:0017171	serine hydrolase activity	7.5	33	102	9.57E-12
GO:0008236	serine-type peptidase activity	7.5	33	102	9.57E-12
GO:0004252	serine-type endopeptidase activity	6.9	31	93	1.15E-11
GO:0008233	peptidase activity	20.3	56	275	3.04E-11
GO:0070011	peptidase activity, acting on L-amino acid peptides	19.5	52	265	8.71E-10
GO:0005198	structural molecule activity	14.4	43	195	9.70E-10
GO:0004175	endopeptidase activity	15.5	45	211	9.37E-10
GO:0003735	structural constituent of ribosome	8.3	23	113	1.88E-04
GO:0061134	peptidase regulator activity	6.7	17	91	8.71E-03
GO:0030414	peptidase inhibitor activity	6.7	17	91	8.71E-03

Graphing in Excel

- Excel is a very useful program for graphing data, since graphs are easily customizable and interactive.
- We will go through the steps required to create a publication-quality scatterplot image of the previously-generated differential gene expression data.
- Note that within excel, graphs are called “charts”. Also note that Excel, particularly on Macs, can sometimes be prone to crashing when working with graphs. Be sure to save frequently.

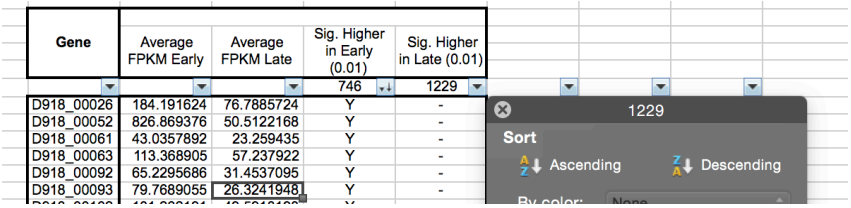


- The points in a graph on it will stay linked to the data you enter. So, if data in the sheet is re-sorted or changed, then the graph will automatically update. For this reason, we will start by moving the data to be graphed onto a new separate sheet, where it won’t be changed later:



Graphing in Excel

- Copy and paste gene names, and all of the DESeq data from the main data sheet into the new graph data sheet.
- Delete the fold change and P value columns by selecting the entire columns (by clicking the letters on the border of the spreadsheet) and then right clicking and “delete”. This data is not required to construct the graph.
- Add the sorting arrows, then sort the sheet by ‘higher in early’ and then ‘higher in late’, so that the three categories of differential expression are in blocks in the table:



Gene	Average FPKM Early	Average FPKM Late	Sig. Higher in Early (0.01)	Sig. Higher in Late (0.01)
			746	1229
D918_00026	184.191624	76.7885724	Y	-
D918_00052	826.869376	50.5122168	Y	-
D918_00061	43.0357892	23.259435	Y	-
D918_00063	113.368905	57.237922	Y	-
D918_00092	65.2295686	31.4537095	Y	-
D918_00093	79.7689055	26.3241948	Y	-

- Cut and paste this table into three sections: Higher in early, higher in late, and not differentially expressed. This isn’t strictly necessary to construct the graph, but it is helpful for organization. Copy and paste the headers to organize the data:

Higher Late				
Gene	Average FPKM Early	Average FPKM Late	Sig. Higher in Early (0.01)	Sig. Higher in Late (0.01)
D918_00003	2.01222465	11.4886605	-	Y
D918_00007	0.08319206	4.73819485	-	Y
D918_00014	2.41819608	20.4226954	-	Y
D918_00023	5.09743301	179.170881	-	Y
D918_00029	1.34108405	24.2946085	-	Y
D918_00034	1.56275633	15.8094969	-	Y
D918_00038	0.14974576	7.33730798	-	Y
D918_00040	0.98383638	186.467957	-	Y

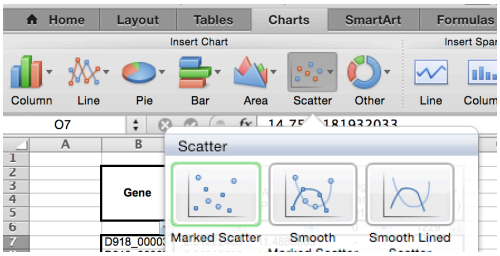
Higher Early				
Gene	Average FPKM Early	Average FPKM Late	Sig. Higher in Early (0.01)	Sig. Higher in Late (0.01)
D918_00026	184.191624	76.7885724	Y	-
D918_00052	826.869376	50.5122168	Y	-
D918_00061	43.0357892	23.259435	Y	-
D918_00063	113.368905	57.237922	Y	-
D918_00092	65.2295686	31.4537095	Y	-
D918_00093	79.7689055	26.3241948	Y	-
D918_00102	101.262191	49.5918198	Y	-
D918_00113	27.1748317	4.12973106	Y	-

Not. Diff Expressed				
Gene	Average FPKM Early	Average FPKM Late	Sig. Higher in Early (0.01)	Sig. Higher in Late (0.01)
D918_00013	14.7543182	10.0696598	-	-
D918_00015	17.3395579	16.177347	-	-
D918_00016	85.3995195	96.128585	-	-
D918_00017	106.959572	73.8053354	-	-
D918_00018	110.435852	94.7858156	-	-
D918_00019	87.258974	65.8212612	-	-
D918_00020	31.9975026	97.6557965	-	-
D918_00021	29.2006658	24.3899591	-	-

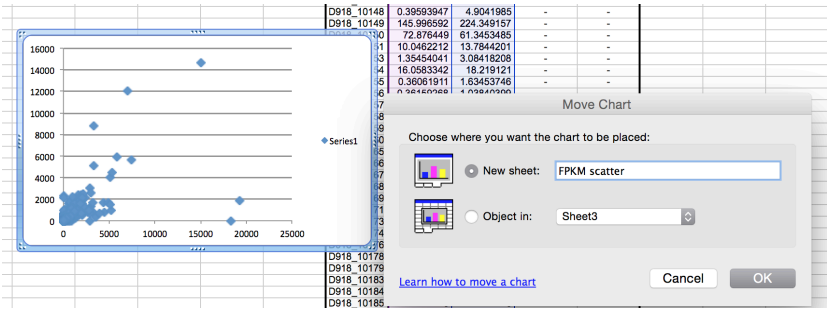
Graphing in Excel

- We will start by graphing the “not differentially expressed” genes as an X-Y scatterplot.
- Use Shift + command/CTRL to highlight the FPKM data down this entire column. Then, under “charts”, choose “scatter” and then “Marked scatter” (with no lines):

Gene	NOT DIFFERENTIAL			
	Average FPKM Early	Average FPKM Late	Sig. Higher in Early (0.01)	Sig. Higher in Late (0.01)
D918_00013	14.7543182	10.0696598	-	-
D918_00015	17.3395579	16.177347	-	-
D918_00016	85.3995195	96.128585	-	-
D918_00017	106.959572	73.8053354	-	-
D918_00018	110.435852	94.7858156	-	-
D918_00019	87.258974	65.8212612	-	-
D918_00020	31.9975026	97.6557965	-	-
D918_00021	29.2008658	24.3899591	-	-

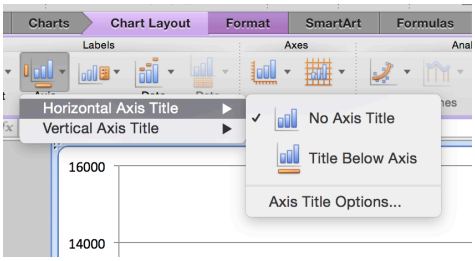


- When you do this, Excel will generate a simple plot of the data, as an object on the sheet. Right click the empty white space on the plot, select “Move Chart”, and then specify a “new sheet” instead, so that it puts the chart on its own sheet:

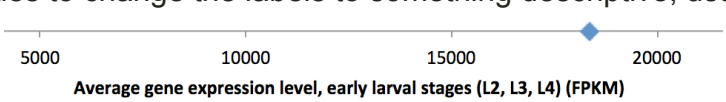


Graphing in Excel

- The default chart is not formatted nicely, and may vary by version of Excel.
- Note that the order of the following formatting steps doesn’t matter.
- First, we will add axes labels. Under “chart layout”, select “axis titles”, and then click to add a title below the X axis and a rotated title on the Y axis:



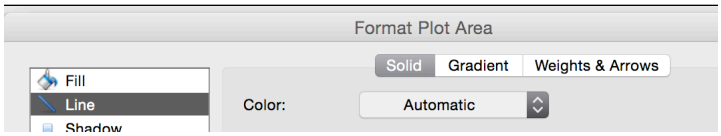
- Click on the axes titles to change the labels to something descriptive, usually with units in parentheses:



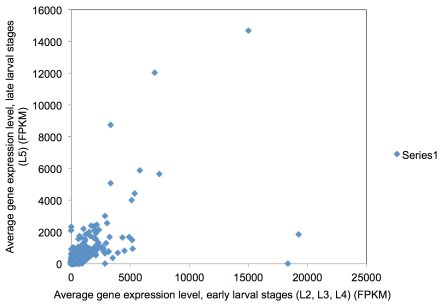
- Next, click on empty white space in the corner of the sheet, to select the entire graph. This will allow you to set a global font without adjusting each component manually. Arial font is always acceptable for publication, so choose it, and choose size 16 font. This large font size is necessary because graphs are rarely printed as a full page, but instead are often shrunk into a single panel.

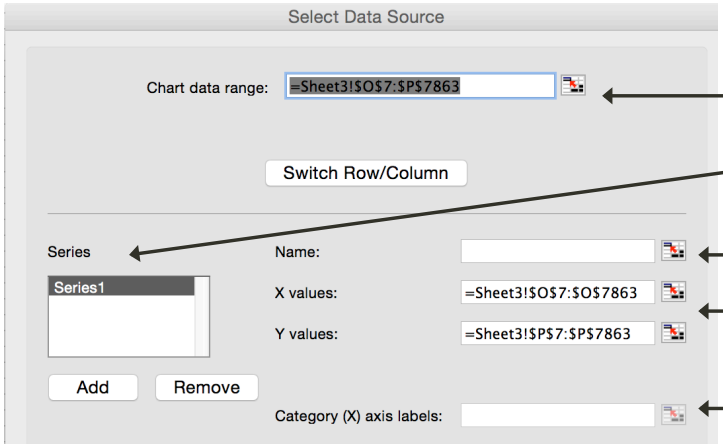
Graphing in Excel

- Remove horizontal gridlines by clicking on one of them and pressing the “delete” key (backspace on windows). Double-click on the plot area and under “line”, choose black for the color instead of “automatic”. This will put a border around the plot.



- We will now start to add the other two data series to the graph.
- Right click on the plot area and then click “Select data”.





Ignore this.

A list of the different series of data on the graph. Each series can be formatted independently.

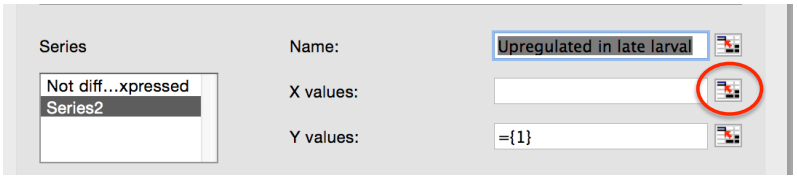
The name of the selected series. If blank, it will default to numbering.

Range of X values and range of Y values for the selected series.

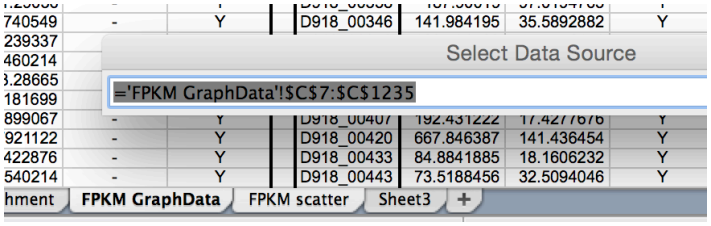
This only matters for graphs with categories (not numbers) on the x axis.

Graphing in Excel

- First, rename the existing series to “Not differentially expressed” (this is the data we started the graph with).
- Click “add” to add a second series. Title the series (“Upregulated in late larval”), and then click the red arrow beside the “X values” to select the x axis values for this series.



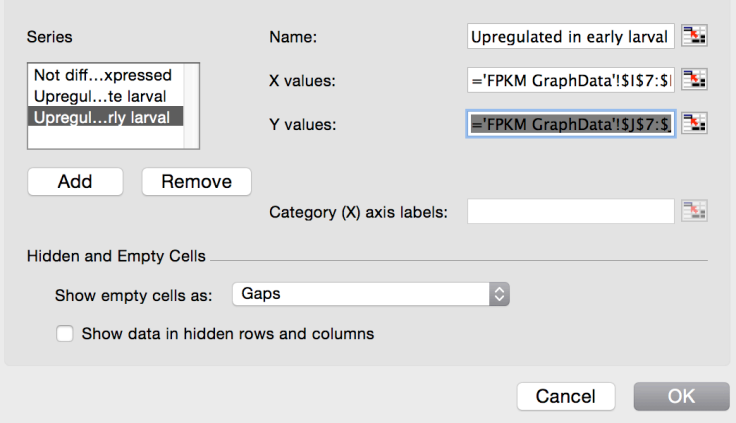
- Click back to the ‘FPKM GraphData’ tab, and highlight the X values (early larval) from the “Upregulated in late larval” columns you previously set up:



- On windows, you can click on the first cell, and shift + CTRL down to select the entire column. This doesn’t always work in the Mac version (a bug), so you may need to either select with the mouse, or type in the range manually.

Graphing in Excel

- Once the data is selected, press enter or press the red arrow to return to the main data selection menu. Repeat this process to select the Y values, and then add another series for the “Upregulated in early larval” data, and add those x and y values.
- When all of this is finished, click “ok” to return to the graph.
- Note that if an error pops up when entering data, it is probably because you clicked in multiple places, and it is expecting a single range of values. If this happens, delete everything in the white box, and then click the red arrow again.

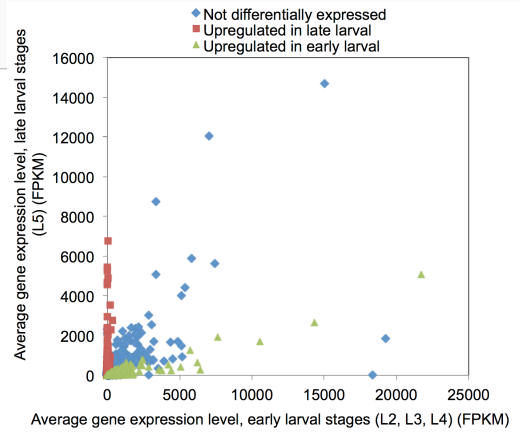


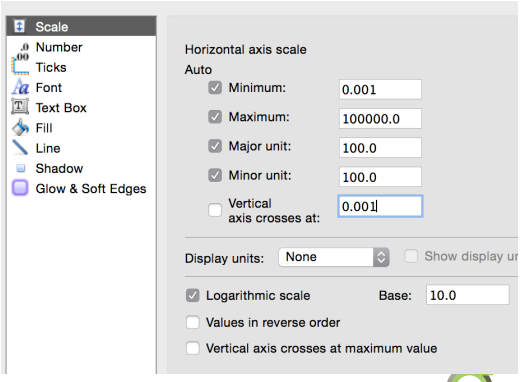
- Click OK to finish the data entry.

Graphing in Excel

- Resize and reposition the legend and the graph to reduce empty white space.
- We will format the axes so that they display log values instead of natural values. Start by double-clicking on any of the numbers on the x axis.
- In the “scale” menu, check “Logarithmic scale”. You will get a warning that zero values cannot be displayed, which we will address shortly.
- Set the “vertical axis crosses at” value to 0.001, so that the axes intersect on the corner.
- Repeat both of these steps for the y-axis, except for the y axis, also set the “major unit” to 100, so that it matches the X axis.

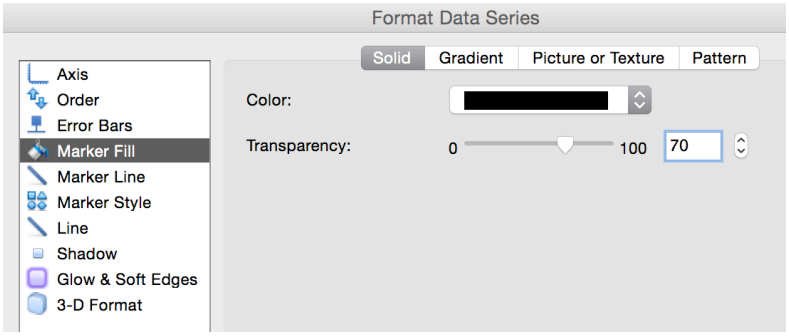
- Although we do not need it for this graph, note that this menu is where you can manually set the minimum and maximum values for the plot.





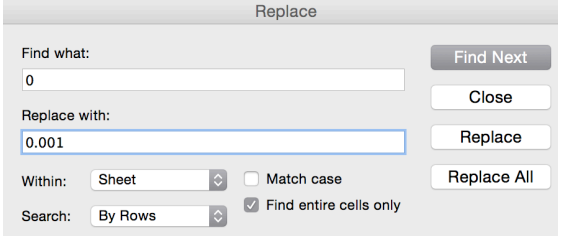
Graphing in Excel

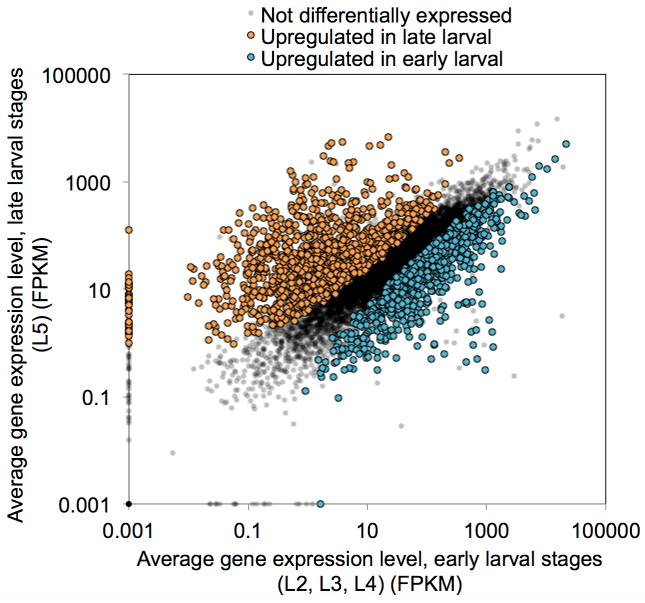
- Next, we will format the data series points. Start by double clicking on one of the “not differentially expressed” points. Note that if you single-click, and then double-click, you will be formatting a single point and not the entire series. Ensure that the popup window says “format data series” and not “format data point”.
- Go to “Marker style” and choose a circle, then set it to size 4. We make these points small because we want the differentially expressed genes to stand out.
- Now choose “Marker line” and choose “no line”. This is for the border around each point which we don’t want for this series.
- Go to “marker fill”, and set to black with 70% transparency. This will make the points translucent, making it easier to tell where they overlap. Click ok to finish formatting.
- Repeat for the two upregulated gene sets, except choose a size 5 circle, a black marker line, and a solid fill with no transparency (orange and blue).



Graphing in Excel

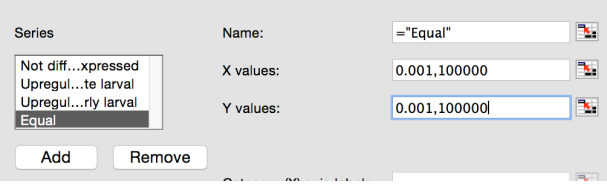
- Now, we will fix the zero values. Rather than not including points with zero expression, we want them to show up along the axis. We will do this by changing all zero values in the graph data to 0.001.
- Go back to the FPKM GraphData tab, and press “command/CTRL + F” to bring up the “Find” dialog. From here, click “replace”, and check off “find entire cells only”. Use this to replace all zero-value cells with 0.001. The graph will auto-update since the cell references are still linked.
- Now, the points plotted along the axes are zero-value, and not 0.001 as indicated. This can either be mentioned in the figure caption, or the 0.001 values can later be covered up in imaging software and replaced with 0 on the plot.



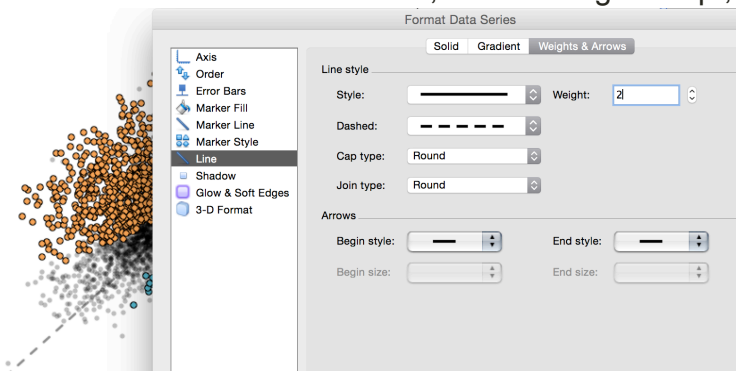


Graphing in Excel

- Finally, we will add a diagonal line to define where the x and y values are equal. To do this, go back to the “select data” menu (right click the empty space on the graph).
- Now add another series called “Equal”. Manually type in the values 0.001,100000 to both the x and y axis values, then click OK.



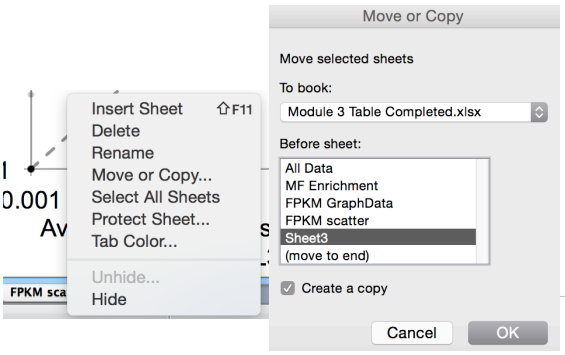
- Two points will show up in the corner. Double click one of them, then set the Marker style to “no marker”, the “line” color to dark grey, and then click to the “weights & arrows” dialog under the “line” menu. In that menu, set the weight to 2pt, and choose a dashed line:

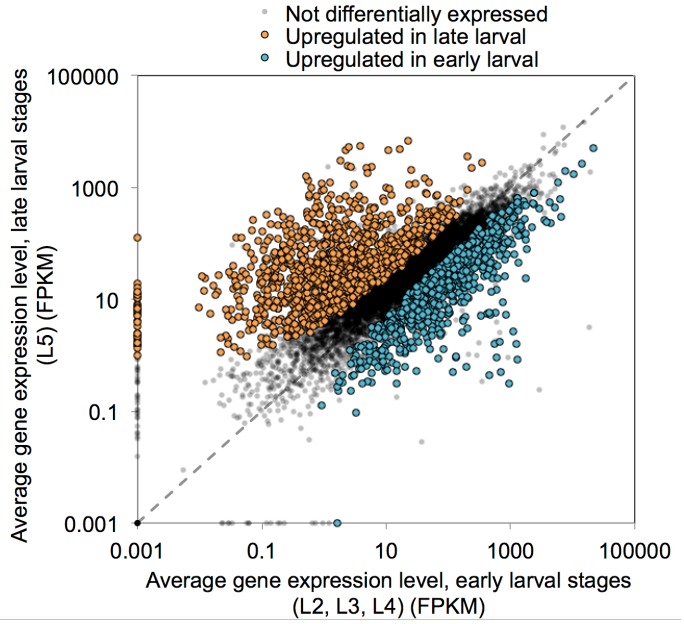


Graphing in Excel

- If “equal” shows up in the legend, click it and delete it.
- At this point, the graph is complete. This can be saved as a PDF file in the “save as” menu, and imported as a vector-format image into other software.

- You can make a copy of the graph by right clicking the sheet tab at the bottom, and choosing “Move or Copy...”, and then specifying to create a copy. This way, if you make a second scatterplot, you can just change the series data, and keep all of the formatting.





Helpful resources for Section 2

- List of RNA-seq bioinformatics tools:
  - https://en.wikipedia.org/wiki/List_of_RNA-Seq_bioinformatics_tools
- khmer website and blog
  - http://khmer-protocols.readthedocs.org/en/v0.8.2/mrnaseq/index.html
  - http://ivory.idyll.org/blog/category/science.html
- DESeq2
  - https://www.bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.pdf
  - http://www.bioconductor.org/help/workflows/rnaseqGene/
- GOstats
  - https://bioconductor.org/packages/release/bioc/vignettes/GOstats/inst/doc/GOstatsHyperG.pdf