Week 15, Lecture 29

István Albert

Biochemistry and Molecular Biology
and Bioinformatics Consulting Center

Penn State
Homework due dates

• This is the last homework.

• All homework must be turned in by Sunday Dec 14th midnight.

• You can email the last homework if you cannot turn it in on Thursday.
What makes RNA-Seq analysis difficult

It requires a mix of many skills:

- Running tools → system administration
- Understanding algorithms → theoretical comp-sci
- Write scripts to process results → programming
- Differential expression analysis → statistics

RNA-Seq requires understanding each of these steps

Very few people (almost no one) masters all of them
Tuxedo Suite

Bowtie
Extremely fast, general purpose short read aligner

TopHat
Aligns RNA-Seq reads to the genome using Bowtie
Discovers splice sites

Cufflinks package
Cufflinks
Assembles transcripts

Cuffcompare
Compares transcript assemblies to annotation

Cuffmerge
Merges two or more transcript assemblies

Cuffdiff
Finds differentially expressed genes and transcripts
Detects differential splicing and promoter use

CummeRbund
Plots abundance and differential expression results from Cuffdiff
As described in:

*Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks*

Trapnell et al

*Nature Protocols, 2012*
RNA-Seq analysis

Alignment
  – Tophat, STAR, SubRead, RSEM

Feature Counting
  – Cuffdiff, htseq, featureCount, eXpress, corset,

**NOTE:** most read counters need data sorted by read name (!) not position. Most aligners produce data in this format by default. But remember that bam files sorted by read name cannot be visualized in IGV! You may need two bam files for each dataset.

Differential Expression Analysis (statistics)
  – Cuffdiff, DESeq, DESeq2, edgeR, EBSeq
Why are there so many methods?

We are working around the limitations of the instruments.

Instruments do not measure transcripts – they measure small fragments that originate from transcripts!

Analogy: we are attempting to reconstruct a large puzzle that has many identical and repetitive elements.
Different “schools of thought”

1. Count only reads that map to exons unique to a transcript $\rightarrow$ estimate transcript from that.

2. Build an internal model and find the simplest/most likely model that fits the data.
A novel transcriptome assembly is always far less reliable than annotated genome with multiple sources of evidence.

A different methodology is then required.
Personal recommendation

• Start with the Tuxedo suite since it is a really straightforward methodology. It will get you at least 80% there. Often it works really well especially for well-annotated organisms.

• Investigate the results and look for unexpected behaviours that your biological system might exhibit. Evaluate mapping at genes of interest. Look for changes relative to isoforms, look for the quantification of overlapping isoforms.

• Simple situations will be handled well by all tools – complex phenomena may be modeled radically differently.

• The statistical power of different methods can be different – these make different assumptions and compute different quantities.
Bad news first:

- Statisticians build very clunky tools
- Usually assume and build upon tacit assumptions
- Their minds work very differently than that of other people – hence “easy” means something else entirely.

Good news

- Once you figure out how to get the data in the right shape published pipelines work well
- The tasks that you need to do are formatting files and labeling and grouping them properly - no higher order statistical knowledge required
How to compare methods

We have a full run with TopHat (lecture 28)

Let’s analyze the same data with a different approaches.

Running the tool is the easy part → making sense of what file goes where and what the tool will output is the hard part
eXpress

• Quantify against a transcriptome

1. Download and install eXpress

2. Download or create a transcriptome

(see codebase)
Subread mapper and featureCounts

A pretty new method with surprisingly high performance.

Part of a pipeline that stretches into R/Bioconductor.

(see codebase)
htseq-count

HTSeq: Analysing high-throughput sequencing data with Python

HTSeq is a Python package that provides infrastructure to process data from high-throughput sequencing assays.

- Please see the chapter *A tour through HTSeq* first for an overview on the kind of analysis you can do with HTSeq and the des
- While the main purpose of HTSeq is to allow you to write your own analysis scripts, customized to your needs, there are also Python knowledge. See the *Scripts* section in the overview below for what is available.
- For downloads and installation instructions, see *Prerequisites and installation*.

Paper

HTSeq is described in the following publication (which is currently under review but already available as preprint):

Simon Anders, Paul Theodor Pyl, Wolfgang Huber
*HTSeq — A Python framework to work with high-throughput sequencing data*

Works well with DESeq as the same author wrote both.
Installation is a bit more complicated.
Due this week Sunday Dec 14\textsuperscript{th} (latest). Try for Thursday.

Compare the expression results of produced via cuffdiff to a different counting method.

• Pick a sample and compute the FPKM levels with another method.

• Compare the top 20 list of genes by count (the most highly expressed genes) between the two results.

• Show the lists and discuss what you see.