Introduction to RNA-Seq Analysis

Research Computing Services
Northwestern University

Outline

In this workshop, you'll learn how to analyze RNA-seq gene expression data to identify the most expressed transcripts/genes using bash/R environment on Quest.

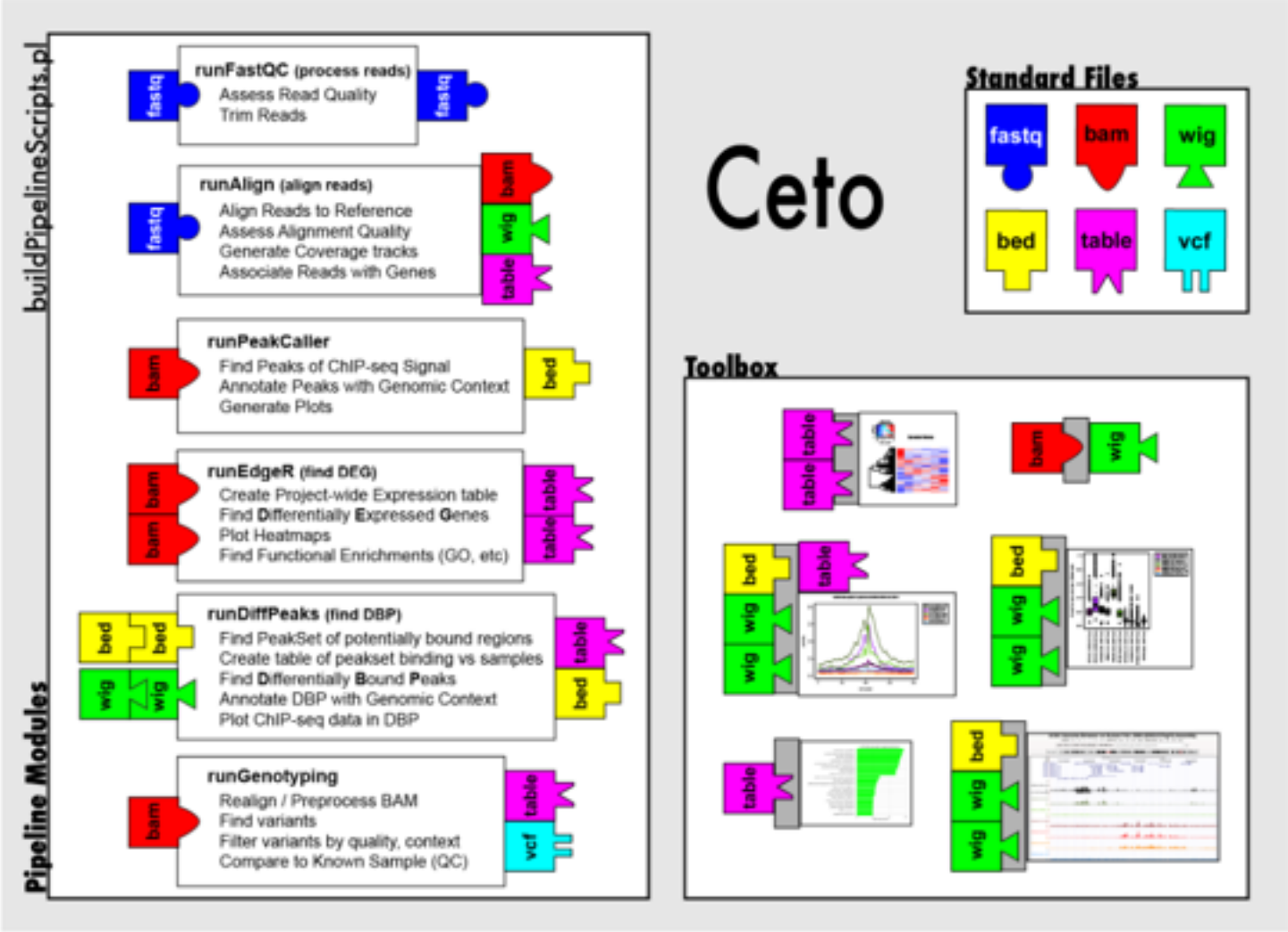
Things you'll learn in this workshop:

- Quality control of raw RNA-seq data
- Reads alignment to a reference using *Stringtie*
- File formats conversion
- Differential gene expression analysis
- Visualization of your results

CETO: a Northwestern RNAseq pipeline

https://github.com/ebartom/NGSbartom

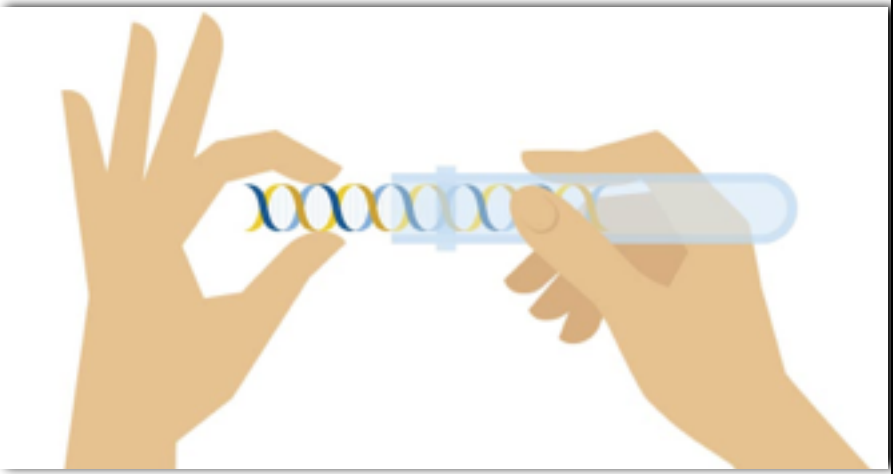
Ceto is a modular system for analyzing next generation sequence (NGS) data.



Introduction to RNA-seq

There are several sequencing analysis available based on the goal of your experiments:

- **RNA-seq** : co-expression networks, differentially expressed genes
- **Chip-Seq | ATAC-seq | DNase-seq** : Gene regulation dynamics, chromatin modeling
- **Whole-genome Seq** : Rare variant analysis



Introduction to RNA-seq

RNA-seq could answer the following experimental questions:

- Measure expression variation within or between species
- Transcriptome characterization
- Identify splicing sites
- Discover novel transcript
- Differential expression studies of a gene in different conditions, etc.

Introduction to RNA-seq

Before you start the analysis, you have to consider the following experiment designs:

- ✓ What resources do you have already?
  - ✓ reference genome, curated genes, etc
- ✓ Do you need biological replications? (usually yes)
- ✓ Do you need technical replications? (mostly not)
- ✓ Do you need controls?
- ✓ Do you need deep sequencing coverage?

Types of RNA-seq reads

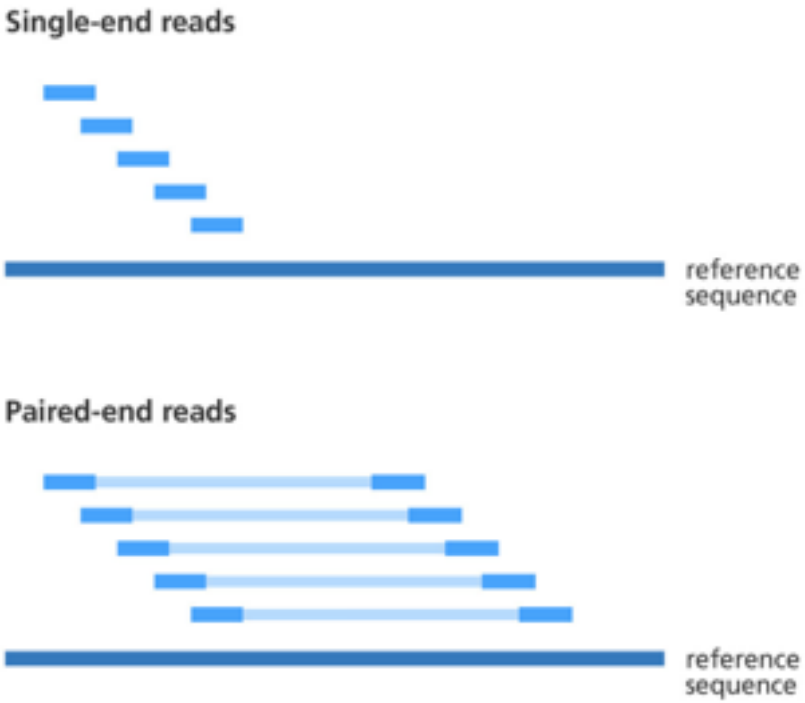
	Notes
Single-end reads	Fast run Less Expensive
Paired-end reads	More data for each fragment/alignment/assembly Good for detecting isoform/structural variation

Sanger sequencing

- fasta format
- 1 header followed by any number of sequences lines

NGS sequencing

- fastq format (Repeated 4 lines)
- Two fastQ files per sample in paired-end sequencing (+ strand, - strand)
- forward/reverse reads have almost same headers



RNA-seq Analysis Workflow

Step1: Quality Control

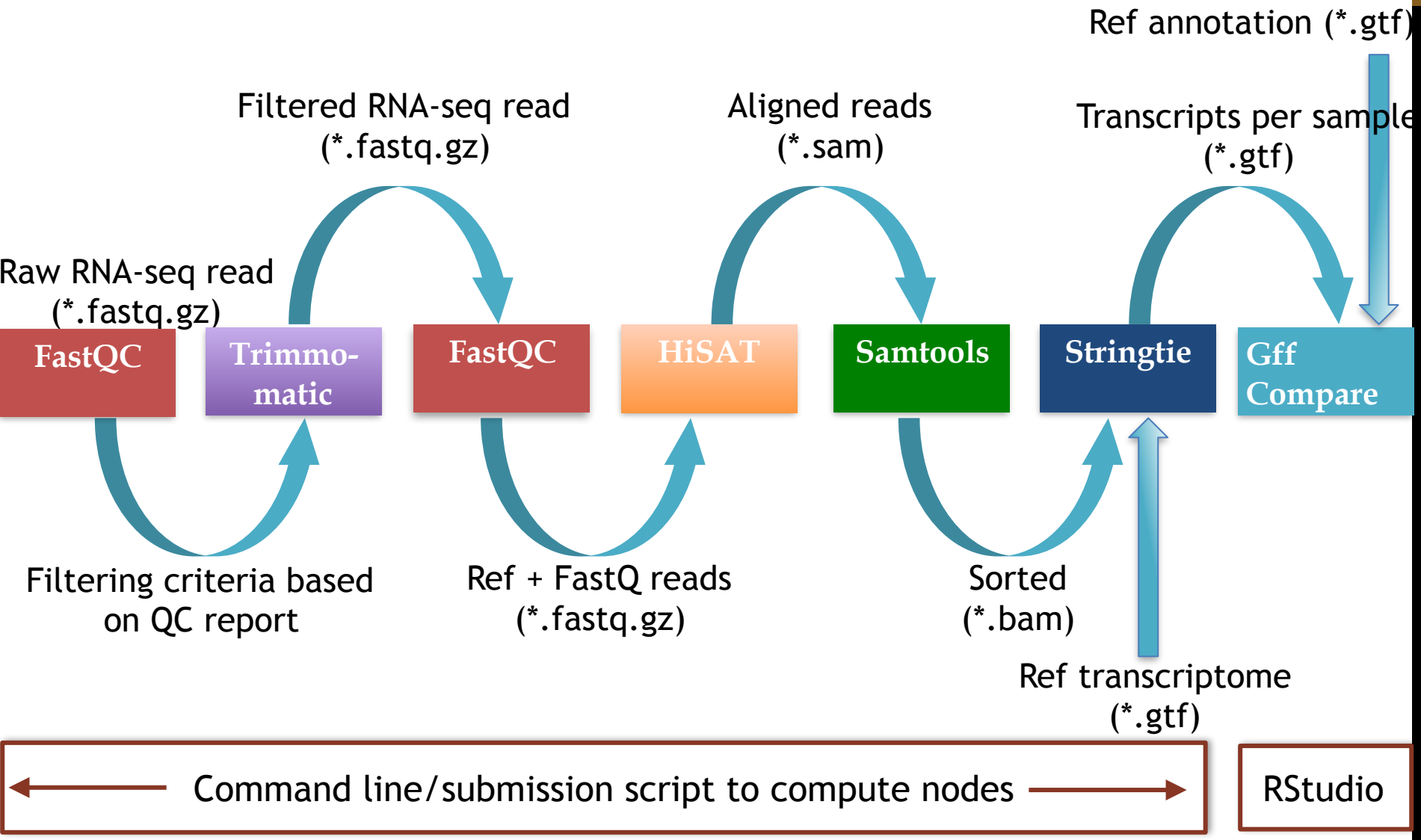
Step2: Data Prep

Step3: Map Reads to Genome/Transcriptome

Step4: Assemble Transcriptome

Step5: Identify Differential Expressed Genes

RNA-seq Software Workflow on Quest



Hands-on code with RNAseq

Please go to Github Repo and download the execution script to follow the rest steps in this workshop.

Github Repo for today's workshop:
github.com/nuitrcs/rnaseq_workshop

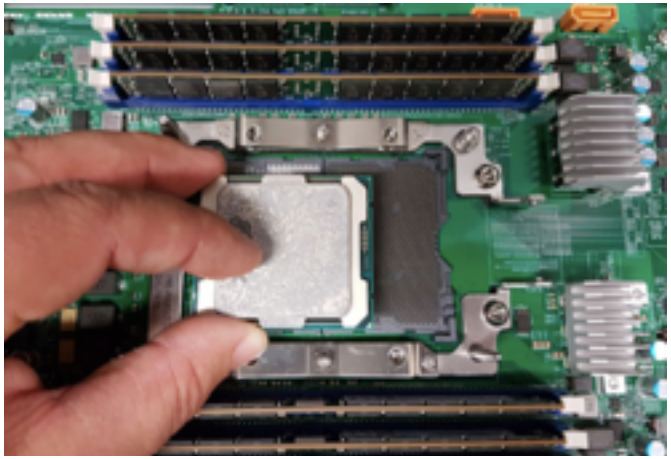
Script we are following for today's workshop:
https://github.com/nuitrcs/rnqseq_workshop/blob/master/RNA_Quest_script_workshop.md

Also available at this bit.ly address:
https://bit.ly/2DCX6O2

Quest: High Performance Computing Cluster

"node": a computer

800



"core": a processor

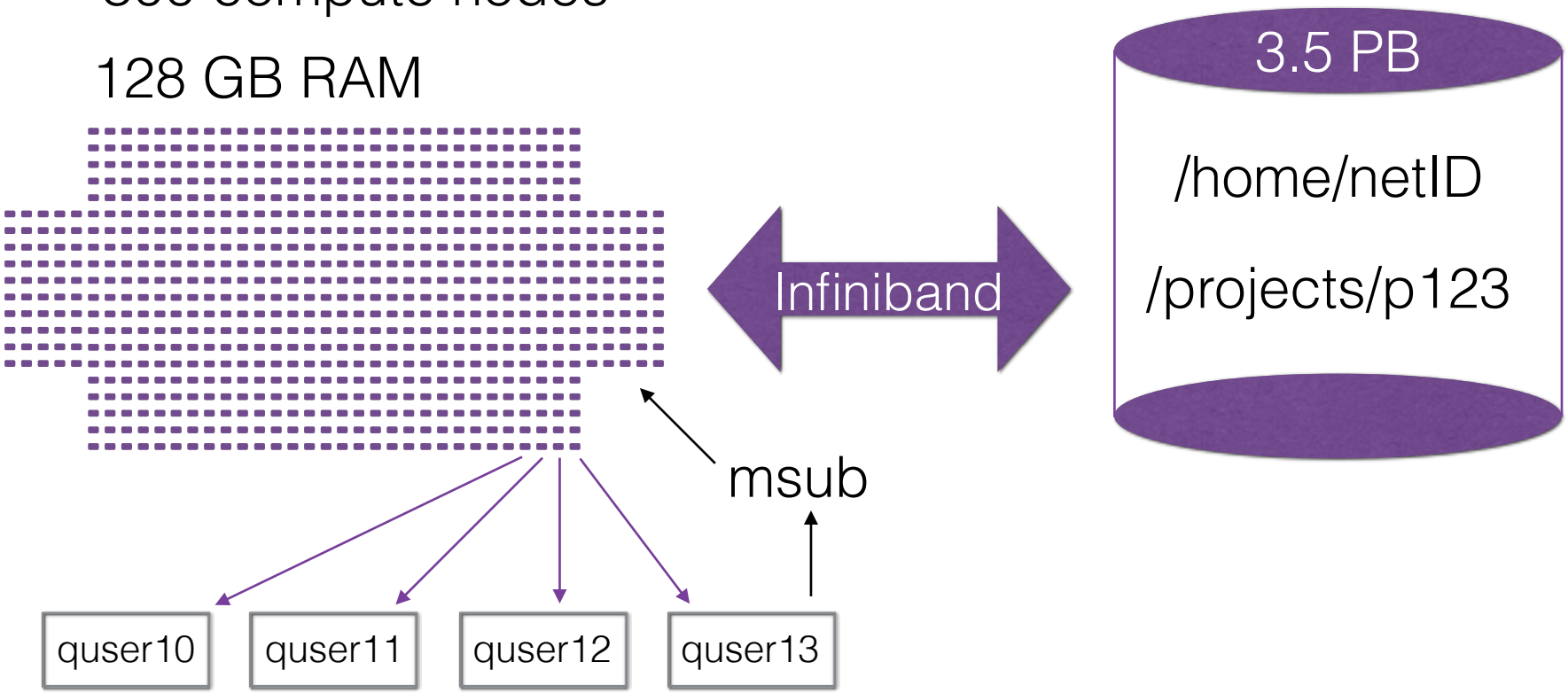
19,200

Quest: High Performance Compute Cluster

Allocation: compute hours, storage space, group

800 compute nodes

128 GB RAM



login nodes

ssh netID@quest.northwestern.edu

Hands-on code with RNAseq

In this protocol, we will run an example differential expression analysis with chromosome X data of Homo sapiens.

All necessary data you need are available in the following directory: ***/projects/genomicsshare/RNAseq_workshop/***

- ***'samples'*** directory contains paired-end RNA-seq reads for 6 samples, 3 male and 3 female subjects from YRI (Yoruba from Ibadan, Nigeria) population.
- ***'indexes'*** directory contains the indexes for chromosome X for HISAT2 alignment.
- '***genome'*** directory contains the sequence of human chromosome X (GrCH38 build 81)
- ***'genes'*** directory contains human gene annotations for GrCH38 from RefSeq database.
- *'mergelist.txt'* and *'phenodata.csv'* are exemplary scripts that you might want to write yourself in a text editor.
- Since it is paired-end reads, each sample has two files: all sequence is in compressed 'fastq' format

Ref: Pertea et al, Nature Protocol 2016

Get Started: Login & Copy Workshop Folder

Login to Quest

```
$ ssh YOUR_NETID@quest.it.northwestern.edu
```

Copy the workshop folder to your directory:

```
$ cd ~
$ cp -r /projects/genomicsshare/RNAseq_workshop .
```

Workshop Script: Bash environment - Setup

Get Started: Load Modules

Load the necessary modules on Quest: fastqc, trimmomatic, samtools, HISAT2

```
$ module load fastqc/0.11.5
$ module load fastx_toolkit/0.0.14
$ module load hisat2/2.0.4
$ module load samtools/1.6
$ module load stringtie/1.3.4
```

Most of the required software is already installed on ***Quest***.

Load the necessary packages in R

```
$ module load R
$ R
> library("devtools")
> source("http://www.bioconductor.org/biocLite.R")
>biocLite(c("alyssafrazee/RSkittleBrewer","ballgown",
                        "genefilter","dplyr","devtools"))
```

Workshop Script: Load the necessary modules

Step 1: Analyze raw reads' quality with ***FastQC***

- We can confirm average quality per read, consistency, GC content (PCR bias), adapter/k-mer content, excessive duplicated reads, etc with ***FastQC***.

- HTML ouput generated by ***FastQC*** helps visual inspection of overall read quality. Not all yellow and red highlights are problematic, so look through the reports with a grain of salt.

Step 1: Analyze raw reads' quality with ***FastQC***

```
$ fastqc --outdir ./qualitycheck/ ./samples/
*_chrX_*.fastq.gz
```

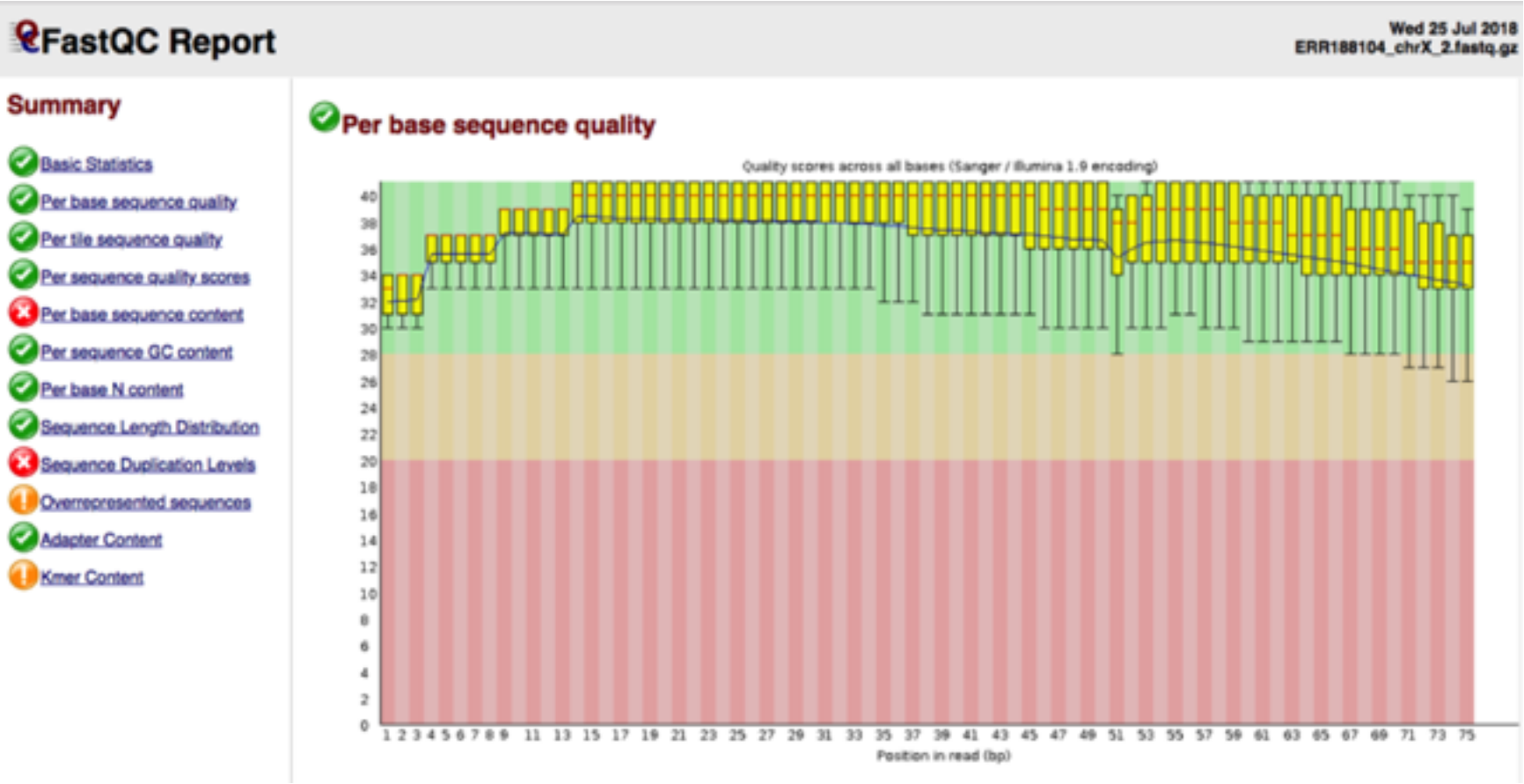
How many quality report (.html) files were created?
Download them and take a look in your choice of browser.
What is your average quality score per read?
- Look @Per sequence quality scores graph

How % of mean GC content per sequence?
- Look @Per sequence GC Content

Workshop Script: Step1. Analyze raw reads' quality with FastQC

Step 1: Analyze raw reads' quality with ***FastQC***

Inspect your FastQC report



✓ Overall very good quality!

Step 2: Filtering raw reads with **Trimmomatic**

- Even if the data looks fine, it is always a good idea to filter out low/poor quality reads.

- Before filtering, you need to specify the sequencing method of your reads since the software commands are different - was this data single-ended or paired-ended?
  - Our data are paired-ended, so we use the 'PE' option with ***Trimmomatic***.

Step 2: Filtering raw reads with **Trimmomatic**

```
$ java -jar trimmomatic-0.33.jar PE -phred33 ./
samples/ERR188273_chrX_1.fastq.gz ./samples/
ERR188273_chrX_2.fastq.gz ./
ERR188273_chrX_1_paired_filtered.fastq.gz ./
ERR188273_chrX_1_unpaired_filtered.fastq.gz ./
ERR188273_chrX_2_paired_filtered.fastq.gz ./
ERR188273_chrX_2_unpaired_filtered.fastq.gz
LEADING:3 TRAILING:3 SLIDINGWINDOW:70:20 MINLEN:30
```

This performs the following:

- Remove leading low quality or N bases (below quality 3) (LEADING:15)
- Remove trailing low quality or N bases (below quality 3) (TRAILING:15)
- Scan the read with a 70-base wide sliding window, cutting when the average quality per base drops below 20 (SLIDINGWINDOW:70:20)
- Drop reads below the 30 bases long (MINLEN:30)
- Convert quality score to Phred-33

Workshop Script: Step2. Filtering raw reads with Trimmomatic

Step 3: Re-analyze the quality of filtered reads with **FastQC**

```
$ fastqc --outdir ./qualitycheck *_chrX_*.fastq.gz
```

- We have to confirm the read quality after filtering.
- You can determine this easily by re-running FastQC on the output fastq files.
- *We didn't remove many, this step could be optional.*

[Q] How many reads are left?

Count the number of lines in a fastq file, which has 4 lines per entry:

`$ wc -l output.fastq`

Divide this number by 4 to get the total number of reads in the fastq file.

[Q] What % of raw reads passed the quality filter?

Step 4: Alignment of RNAseq reads to the genome with ***HISAT***

We will map the reads for each sample to the reference genome:

```
$ hisat2 -p 1 --dta -x ./indexes/chrX_tran -1 ./
ERR188044_chrX_1_paired_filtered.fastq.gz -2 ./
ERR188044_chrX_2_paired_filtered.fastq.gz -S
ERR188044_chrX.sam
```

- Use one alignment thread to launch (-p 1)
- Reports alignments tailored for transcript assemblers (--dta)
- (-x) input reference, (-S) output name
- **Repeat for the rest of 6 files**

Step4. Alignment of RNA-seq reads to the genome with HISAT2

Step 4: Alignment of RNAseq reads to the genome with ***HISAT***

[Q] What % of alignment reads on genome?
- Find the last line in the result screen!

Parameter for QC: proportion of mapped read on the reference to confirm sequencing accuracy and contaminated DNA

- If RNA-seq reads are mapped to *human genome*
  - 70~90% + a few multi-mapping reads
- If RNA-seq reads are mapped to *transcriptome*
  - less mapping % + more multi-mapping reads by sharing same exon among isoforms

If the result screen says that some reads aligned discordantly, it means some occurrences of infusion or translocation.

Step 5: Sort and convert the SAM file to BAM with **samtools**

Samtools (v 2.1.0) sorts and converts the SAM file to BAM.

- Both SAM/BAM formats represent alignments.
- BAM is more compressed format.
- Unmapped reads may also be in the BAM file.
- Reads that map to multiple location will show up multiple times.

```
$ samtools sort -@ 1 -o ERR188044_chrX.bam ERR188044_chrX.sam
$ samtools sort -@ 1 -o ERR188104_chrX.bam ERR188104_chrX.sam
$ samtools sort -@ 1 -o ERR188234_chrX.bam ERR188234_chrX.sam
$ samtools sort -@ 1 -o ERR188273_chrX.bam ERR188273_chrX.sam
$ samtools sort -@ 1 -o ERR188454_chrX.bam ERR188454_chrX.sam
$ samtools sort -@ 1 -o ERR204916_chrX.bam ERR204916_chrX.sam
```

Step 5. Sort and convert the SAM file to BAM with samtools

Step 6: Assemble & quantify expressed genes and transcripts with **StringTie**

(a) Stringtie assembles transcripts for each sample:

```
$ stringtie -p 2 -G ./genes/chrX.gtf -o
ERR188044_chrX.gtf -l ERR188044
ERR188044_chrX.bam        # repeat for 6 files
```

(b) Stringtie merges transcripts from all samples:

```
$ stringtie --merge -p 2 -G ./genes/chrX.gtf
-o stringtie_merged.gtf ./mergelist.txt
```

(c) Stringtie estimates transcript abundances and create table counts for Ballgown:

```
$ stringtie -e -B -p 2 -G
stringtie_merged.gtf -o ./ballgown/
ERR188044/ERR188044_chrX.gtf
ERR188044_chrX.bam        # repeat for 6 files
```

Differential Expression analysis

In this step, we will quantify differential expression of transcript/genes among different conditions:

Submitting this job to the compute nodes on Quest

```
$ more RNAseq_workshop_submit.sh
```

May need to edit the file:

```
#!/bin/bash
#MSUB -A b1042
#MSUB -q genomics
#MSUB -l walltime=08:00:00
#MSUB -l nodes=1:ppn=6
#MSUB -N RNAseq_workshop
```

Notes the loops in the script - improves efficiency for re-use

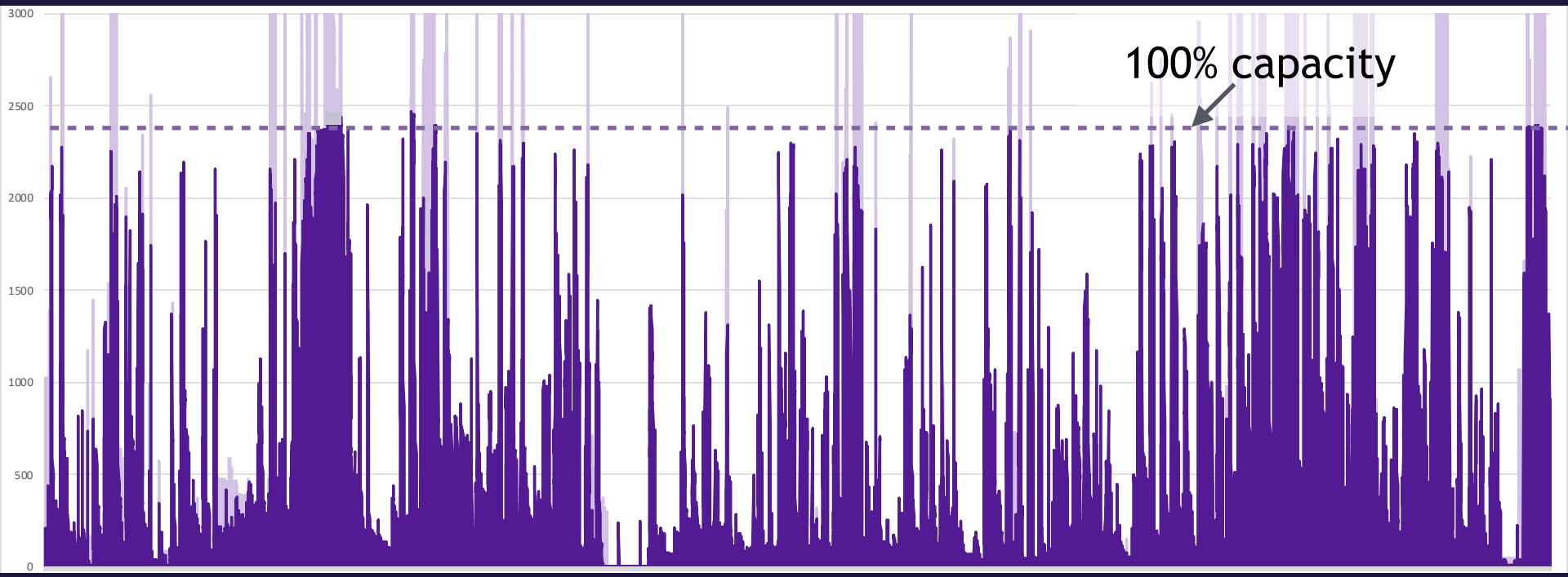
```
$ msub RNAseq_workshop_submit.sh
```

Submit a job to run the pipeline on all samples

text editor: vi/vim

- **On command line:**
  “vi RNAseq_workshop_submit.sh”
- **You land in command mode**
- **To enter insert mode: “i” where the cursor is; “o” inserts one line below that**
- **Navigate with arrow keys**
- **To exit insert mode: “esc”**
- **To save: “:w”**
- **To exit: “:q”**
- **To exit without saving: “q!”**

Utilization on Quest



Jan Feb Mar Apr May Jun

Number of processors engaged

Number of processors requested in all waiting jobs

Analysis and Visualization: RStudio on QuestAnalytics

https://rstudio.questanalytics.northwestern.edu/auth-sign-in

```
> setwd("/home/<YOUR_NETID>/RNAseq_workshop/")
```

Analysis and Visualization performed in RStudio

Step 7: Run differential expression analysis with **Ballgown**

```
>library("devtools")
>source("http://www.bioconductor.org/
biocLite.R")
>biocLite(c("alyssafrazee/
RSkittleBrewer","ballgown",
“genefilter","dplyr","devtools"))

>library("ballgown")
>library("RSkittleBrewer")
>library("genefilter")
>library("dplyr")
>library("devtools")
```

7-a. RStudio environment setup (takes ~ 10 mins)

Step 7: Run differential expression analysis with **Ballgown**

(b) Load phenotype data

- In your future experiment, create your own phenotype data specifying the sample conditions you would like to compare.
- Each sample information presents on each row of the file

```
> pheno_data = read.csv("phenodata.csv")
> head(pheno_data)
```

Take a look of the phenotype data!
[Q] How many different groups do we have?
[Q] How many samples in each group?

7-b. Load phenotype data into RStudio

Step 7: Run differential expression analysis with ***Ballgown***

(c) Read in the expression data that were calculated by Stringtie in previous step 6-(c)

- IDs of the input files should be matched with the phenotype info.

```
> chrX <- ballgown(dataDir="ballgown",
samplePattern="ERR", pData=pheno_data)
> str(chrX)
```

(d) Filter to remove low-abundance genes

- We can apply a variance filter for gene expression analysis. (remove low-expressed transcript with a variance across samples less than 1)

```
> chrX_filtered <- subset(chrX,
"rowVars(texpr(chrX)) >1", genomesubset=TRUE)
> str(chrX_filtered)
```

7-c. Read in expression data, 7-d.Remove low-abundance genes

Step 8: Identify transcripts/genes that show statistically significant differences with ***Ballgown***

Measurement criteria for RNA-seq quantification:

1. RPKM **(reads per kilobase of exon model per million reads)** adjusts for feature length and library size by sample normalization
2. FPKM **(fragment per kilobase of exon model per million mapped reads)** adjusts sample normalization of transcript expression (= similar to RPK)
   - RPKM = FPKM (in Single-end sequencing)
   - FPKM can be translated to TPM
3. TPM **(transcript per million)** is used for measuring RNA-seq gene expression by adjusting transcript differences with overall read # in library: useful in comparing inter-sample comparison with different origins/compositions

✓ Gene length is not important for inter-sample gene expression comparison, but important in ranking intra-sample gene expression

Step 8: Identify transcripts/genes that show statistically significant differences with ***Ballgown***

(a) Identify transcripts that show statistically significant differences between groups:

Look for transcripts that are differentially expressed between sexes, while correcting for any expression differences due to condition variable.

```
> results_transcripts <- stattest(chrX_filtered, feature="transcript", covariate="sex", adjustvars=c("condition"), getFC=TRUE, meas="FPKM")
```

Add gene names and gene IDs to the results:

```
> results_transcripts <- data.frame(geneNames=ballgown::geneNames(chrX_filtered), geneIDs=ballgown::geneIDs(chrX_filtered), results_transcripts)

> head(results_transcripts)
```

8-a. Identify transcripts

Step 8: Identify transcripts/genes that show statistically significant differences with ***Ballgown***

(b) Identify genes that show statistically significant differences between groups

```
> results_genes <- stattest(chrX_filtered,
feature="gene", covariate="sex",
adjustvars=c("condition"), getFC=TRUE,
meas="FPKM")

> head(results_genes)
```

Step 9. What is the most 'differentially' expressed *transcript/genes* between sexes?

Sort the results from the smallest-largest p-value

```
> results_transcripts <-
results_transcripts[order(results_transcripts$pval),]
> results_genes <- results_genes[order(results_genes
$pval),]
```

What are the top transcript/gene expressed differently between sexes?

```
> head(results_transcripts)
> head(results_genes)
```

Write/Save the analysis results:

```
> write.csv(results_transcripts,
file="DifferentialExpressionAnalysis_transcript_results
.csv", row.names=FALSE)
> write.csv(results_genes,
file="DifferentialExpressionAnalysis_gene_results.csv",
row.names=FALSE)
> save.image() # your workspace will be saved as '.RData'
in current working directory
```

Visualization

There are two ways on Quest to interactively visualize your outcomes depend on the size of your analysis:

- ✓ **For small-moderate size analysis:**
  - Rstudio-Quest analytics node on your choice of browser
- ✓ **For large size analysis that might require over 4GB of RAM or more than 4+ cores:**
  - Request an interactive session on Quest node
  - msub -I -l nodes=1:ppn=4 -l walltime=01:00:00 -q genomics -A b1042

Step11. Visualization

(a) Plot for distribution of gene abundances across samples:

We will compare the FPKM measurements for the transcripts colored by 'sex' variable in phenotype file.

```
> fpkm <- texpr(chrX, meas='FPKM')
> fpkm <- log2(fpkm +1)
> boxplot(fpkm, col=as.numeric(pheno_data
$sex), las=2,ylab='log2(FPKM+1)')
```

Step11. Visualization

(b) Plot for individual expression of a certain transcript between groups:

Setup palette with your favorite colors

```
> coloring <- c('darkgreen', 'skyblue',
'hotpink', 'orange', 'lightyellow')
> palette(coloring)
```

Choose your transcript of interest

By looking head(results_transcripts), decide the transcript/gene of your interest! If I randomly choose to draw the 13th most transcript in the dataset (gene name "XIST") because I like the gene name:

```
> which(ballgown::geneNames(chrX)=="XIST")
```

Step11. Visualization

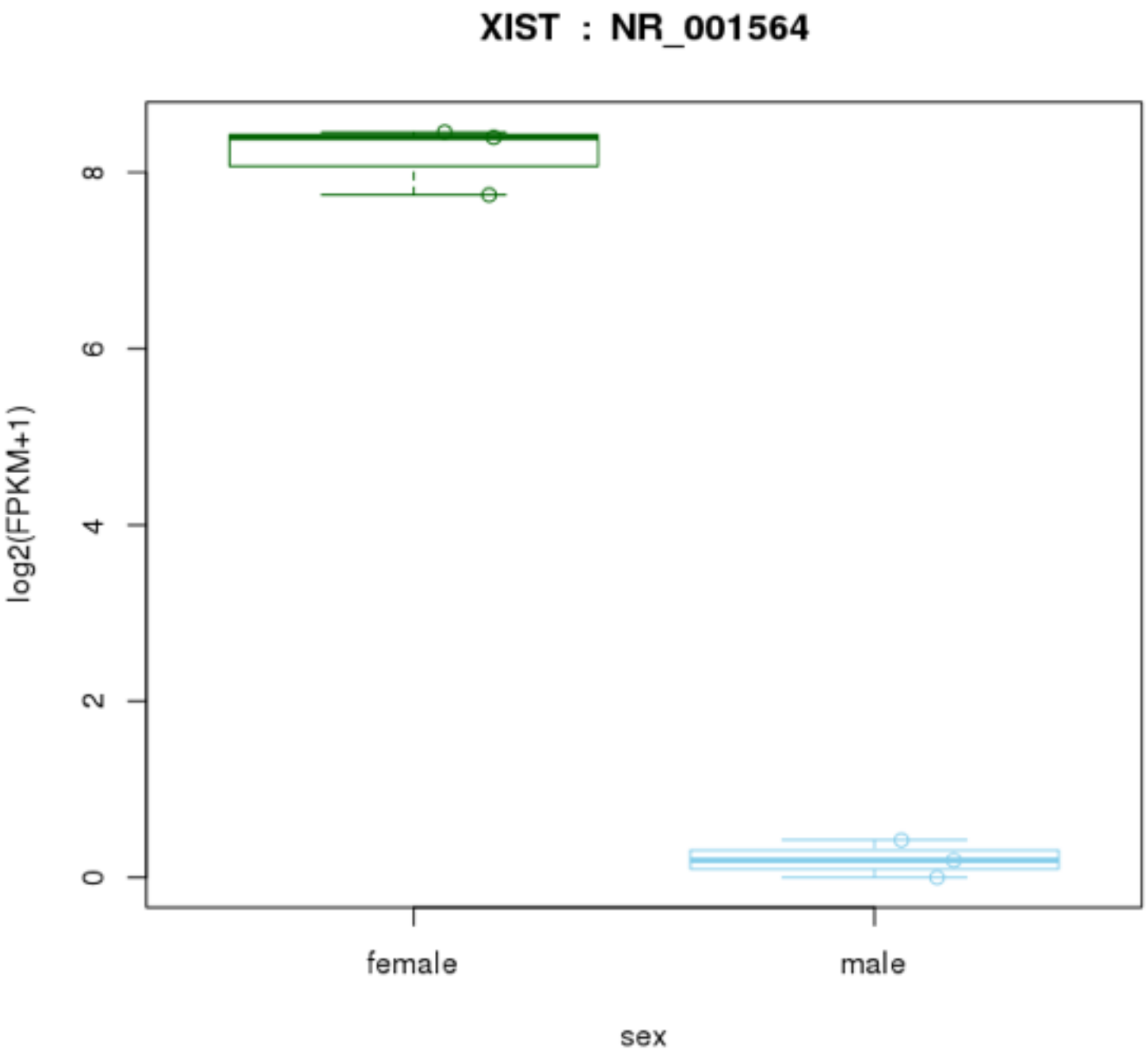
Find the row number of the interested gene in dataset # 1484 here

ballgown::transcriptNames(chrX)[1484]

get the transcript name in the gene

```
> plot(fpkm[1484,] ~ pheno_data$sex, border=c(1,2),
main=paste(ballgown::geneNames(chrX)[1484], ' :
',ballgown::transcriptNames(chrX)[1484]), pch=19,
xlab="sex", ylab='log2(FPKM+1)')
> points(fpkm[1484,] ~ jitter(as.numeric(pheno_data
$sex)), col=as.numeric(pheno_data$sex))
```

✓ The output plot will show the name of the transcript (NR_001564) and the name of the gene (XIST) that contains it.


XIST : NR_001564
log2(FPKM+1)
0
2
4
6
8
female
male
sex

Step11. Visualization

(c/d) Plot the average expression levels for all transcripts of a gene within different groups:

Using plotMeans() function, specify which gene to plot and which variable to group by.

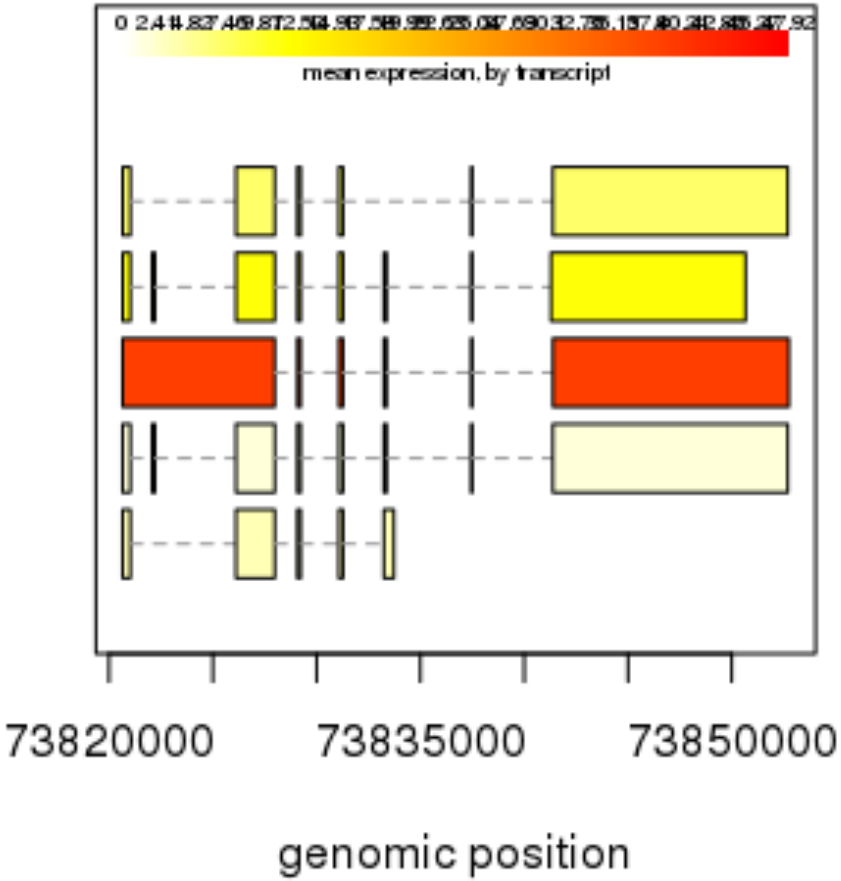
```
> geneIDs(chrX)[1484]
```

MSTRG.495

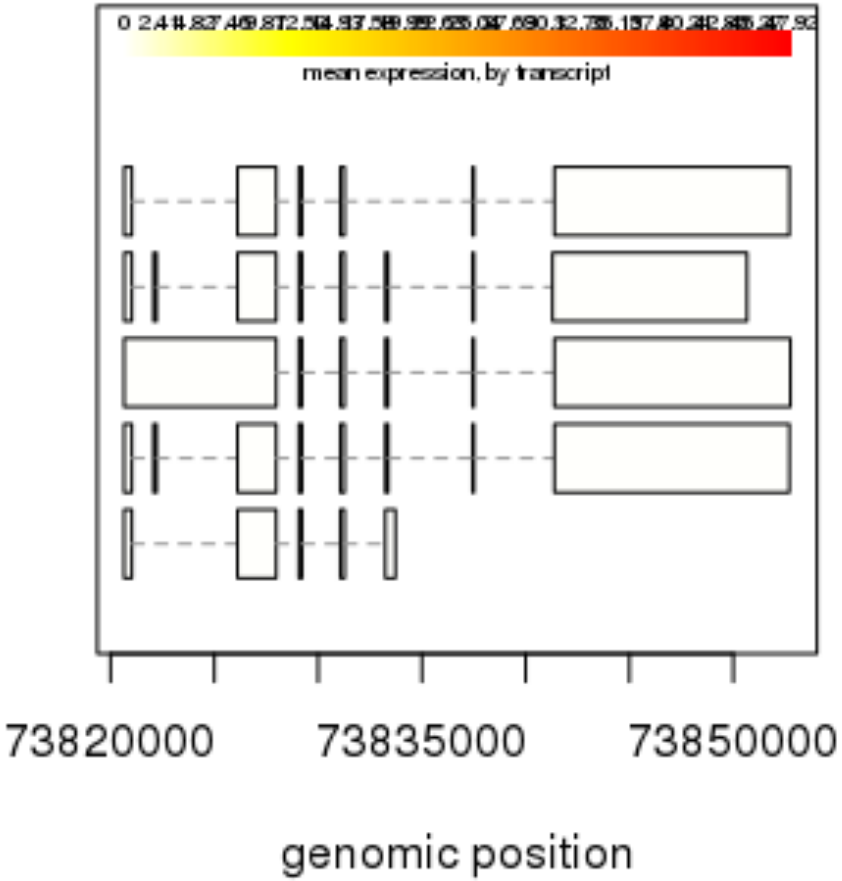
Same plot, Different command:

```
> plotMeans('MSTRG.495', chrX_filtered, groupvar="sex", legend=TRUE)
> plotMeans(ballgown::geneIDs(chrX)[1484], chrX, groupvar="sex", legend=TRUE)
```

MSTRG.495: female



MSTRG.495: male



Step11. Visualization

[Q] Can you tell the exclusive expression of XIST in females?

*(c.f.) In females, the **XIST gene** is expressed exclusively from the inactive X chromosome, and it is essential for the initiation and spread of X inactivation, which is an early developmental process that transcriptionally silences one of the pair of X chromosomes!*

Explore the results!

You can choose to use either **IGV or UCSC Genome browser** for visualizing your outcome.


QUESTIONS?

Step 10. Choose your environment for *Visualization*

Since our file sizes are small enough, Go to Rstudio-Quest Analytics node on your browser:

[https://rstudio.questanalytics.northwestern.edu/auth-sign-in]

You have to re-install the required R packages for differential data analysis described in previous steps:

```
> setwd("/YourWorkingDirectory/")
> load(".RData")
```