Introduction to NGS Data Analysis

High Performance
Research Computing
DIVISION OF RESEARCH

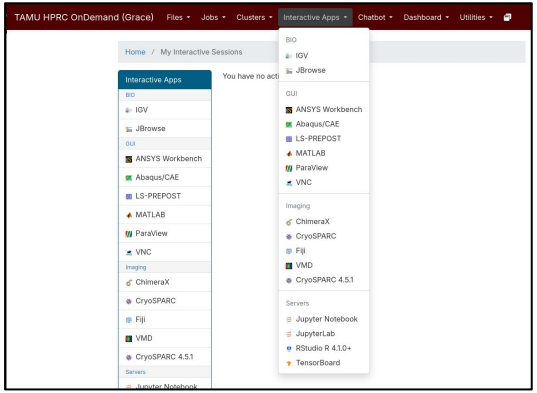
Fall 2025

Introduction to NGS

- HPRC Cluster Utilities
- NGS Technology
- NGS Tools on Grace
- Run Template Job Scripts
  - Quality Control (QC)
  - trim sequence reads
  - align trimmed reads to reference genome
  - sequence variant calling
- Biocontainers

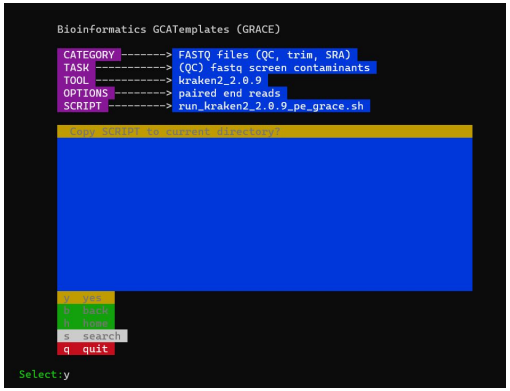
Options for Running Bioinformatics Tools

HPRC Portal



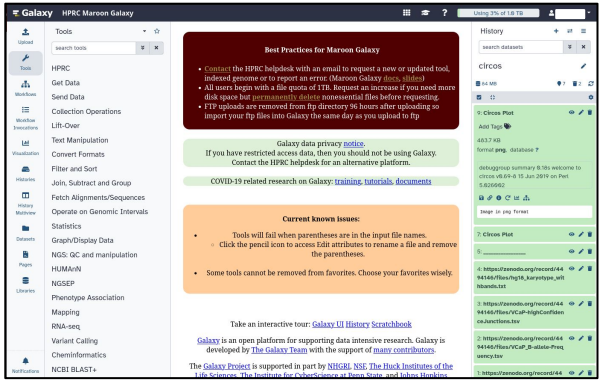
- portal-grace.hprc.tamu.edu
- Access is web-browser-based
- All HPRC software tools are available either as a GUI or via Unix command line
- Can access Unix command line
- Best for GUI apps
  - RStudio
  - IGV

Unix command line



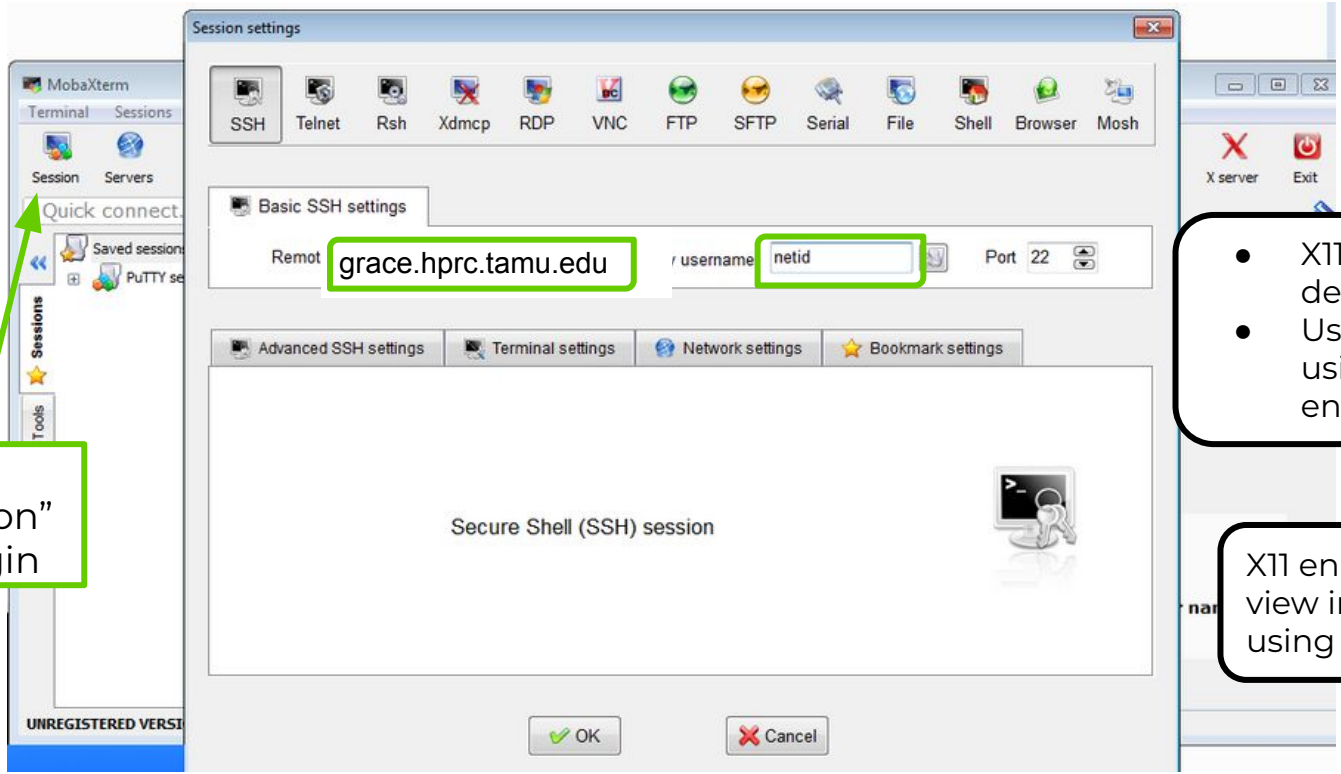
- Need to learn Unix and Slurm
- Bioinformatics template scripts available
- GUI software is not very responsive interactively
- Need SSH client on your Windows computer such as MobaXterm or use the HPRC portal

HPRC Maroon Galaxy



- Access is web-browser-based
- First apply for an HPRC account then request a Galaxy account
- Can request HPRC to add tools from the usegalaxy.org toolshed or create a custom tool

Using SSH - MobaXterm (on Windows) to Connect to Grace



click "Session" to begin

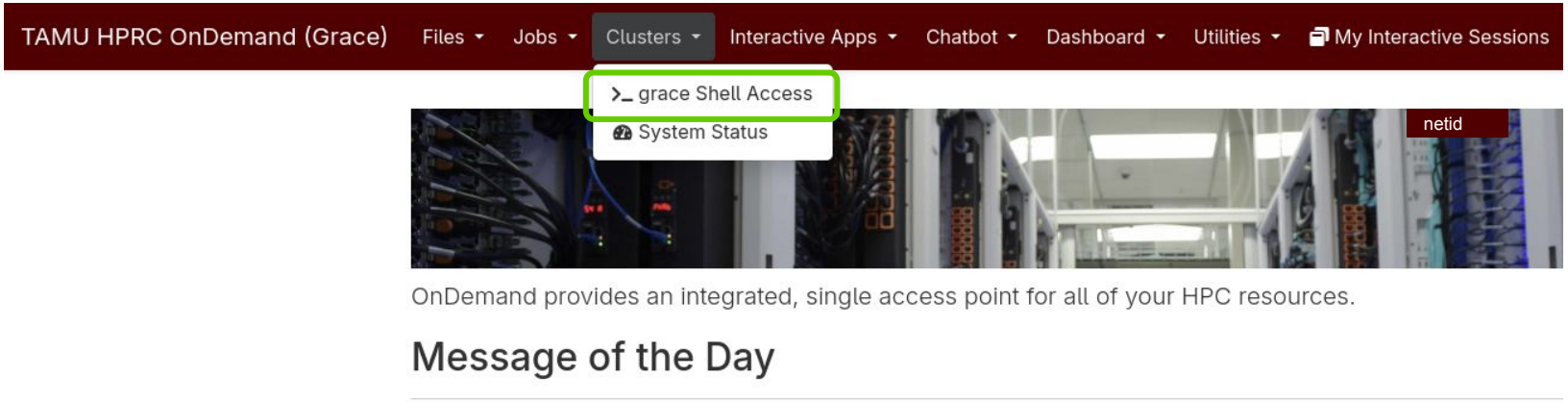
- X11 is enabled by default in MobaXterm
- Use "ssh -X" when using the terminal to enable X11

X11 enables you to view images when using the terminal

https://hprc.tamu.edu/kb/Helpful-Pages/#mobaxterm

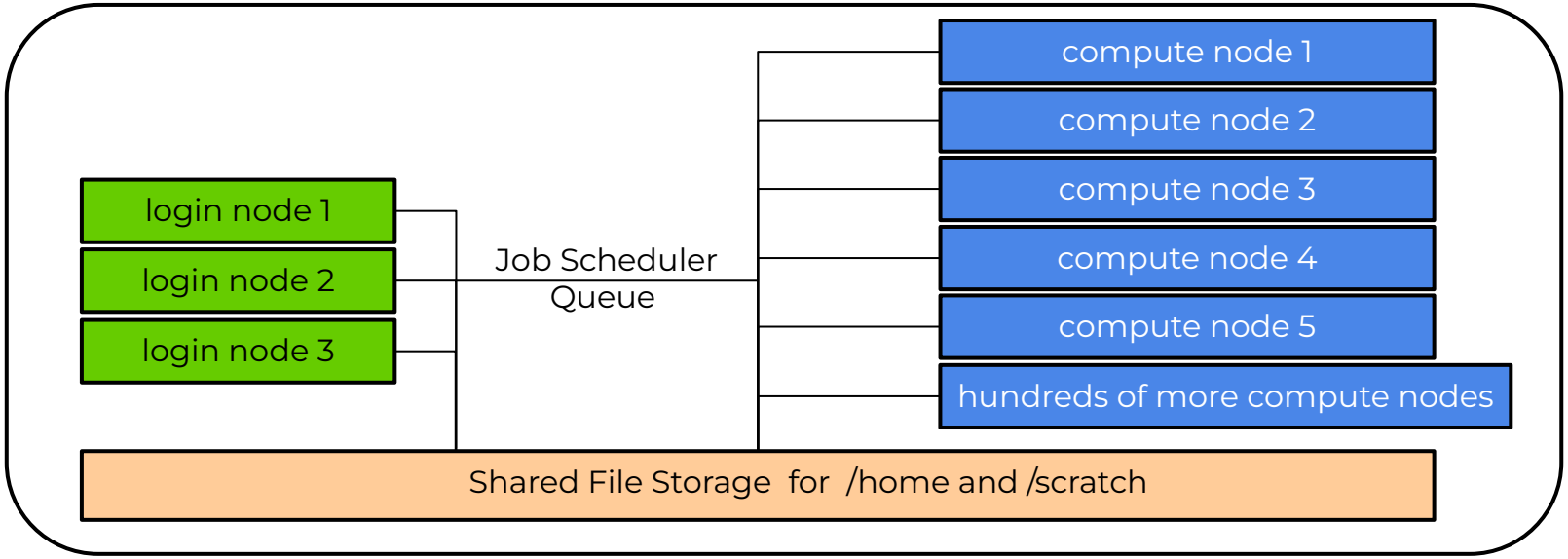
Connect using the HPRC portal

portal-grace.hprc.tamu.edu



- There are no SUs charged for using the Shell Access since an actual job is not launched.
- Interactive Apps launch jobs and charge SUs.

HPC Diagram



login nodes are for:

- file manipulation and job script preparation
- software installation and testing
- short tasks (< 60 minutes and max 8 cores)
  - also be aware of amount of memory utilized
- using the login node does not charge SUs

compute nodes are for:

- computational jobs which can use up to 48 cores and/or up to 360GB memory per Grace compute node.
- all jobs running > 60 minutes
- jobs running on compute nodes charge SUs

HPRC Cluster Utilities

A number of cluster utilities are available to help you query resources at the command line such as available nodes, GPUs, cores, memory, template job scripts and shared conda and Python environments.

myjob	**gpuavail**
maxconfig	**cpuavail**
gcatemplates	**envsavail**
jobstats	**venvavail**
toolchains	**maintenance**

use the **-h** or **--help** flag with any utility to see available options

Show Your Job Details using myjob

```
[userid@grace ~]$ myjob 11760808

              Job ID: 11760808
             Cluster: grace
          User/Group: userid/userid
               State: COMPLETED (exit code 0)
           Partition: cpu
          Node Count: 1
            NodeList: c098
      Cores per node: 24
        CPU Utilized: 01:01:54
      CPU Efficiency: 0.57% of 7-12:11:12 core-walltime
         Submit time: 2024-10-21 15:20:05
          Start time: 2024-10-21 15:20:26
            End time: 2024-10-21 22:50:54
 Job Wall-clock time: 07:30:28
     Memory Utilized: 37.08 GB
   Memory Efficiency: 20.60% of 180.00 GB
            Job Name: parafold-cpu
Job Submit Directory: /scratch/user/userid/af_demo
         Submit Line: sbatch run_parafold_2.0_alphafold_2.3.2_a100_monomer_ptm_grace.sh
```

use the -h flag to view usage

```
myjob -h
```

PENDING Job due to a Scheduled Maintenance

```
[userid@grace ~]$ myjob 1320633

              Job ID: 1320633
             Cluster: faster
          User/Group: userid/userid
             Account: 123456789101
               State: PENDING
              Reason: ReqNodeNotAvail, Reserved for maintenance
         Submit time: 2024-10-14 10:09:59
           Partition: cpu
          Node Count: 1
            NodeList: None assigned
               Cores: 1
                Note: Efficiency not available for jobs in the PENDING state.
            Job Name: picard
Job Submit Directory: /scratch/user/userid/myproject/picard
         Submit Line: sbatch run_bwa_samtools_pilon_faster.sh
                Note: job is PENDING due to runtime overlapping with maintenance window.
                Note: maintenance will begin in  22 hours, and 49 minutes.
```

PENDING Job due to Partition Time Limit

```
[userid@grace ~]$ myjob 11805612

              Job ID: 11805612
             Cluster: grace
          User/Group: userid/userid
             Account: 132787216126
               State: PENDING
              Reason: PartitionTimeLimit
         Submit time: 2024-10-23 13:14:58
           Partition: bigmem
          Node Count: 1
            NodeList: None assigned
               Cores: 1
                Note: Efficiency not available for jobs in the PENDING state.
            Job Name: my_job
Job Submit Directory: /scratch/user/userid/myproject/canu
         Submit Line: sbatch run_canu_bigmem_grace.sh
                Note: The bigmem Partition has a maximum of 2-00:00:00 but the job script has 7-00:00:00
                Note: Use the 'scancel 11805612' command to cancel this job and reconfigure your bigmem
                        job script to use a maximum of 2-00:00:00
                Note: Use the 'sinfo' command to see all partition time limits.
```

Viewing Maximum Available Resources

The **maxconfig** command will show the recommended Slurm parameters for the maximum available resources (cores, memory, time) per node for a specified accelerator or partition (default Grace partition: long)

```
[username@grace ~]$ maxconfig

  Grace partitions:   short  medium  long  xlong  vnc  gpu  bigmem  special  gpu-a40
  Grace GPUs in gpu partition:  a100:2  a40:3  rtx:2  t4:4

  Showing max parameters (cores, mem, time) for partition long

  CPU-billing * hours * nodes =   SUs
            48 *   168 *      1 = 8,064

#!/bin/bash
#SBATCH --job-name=my_job
#SBATCH --time=7-00:00:00
#SBATCH --nodes=1              # max 64 nodes for partition long
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=48
#SBATCH --mem=360G
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
```

Viewing Maximum Available Resources

See the recommended Slurm parameters for requesting 1 x A100 GPU with ½ the total CPUs and memory since there are 2 x A100s per node.

```
[username@grace ~]$ maxconfig -g a100 -G 1

  Grace partitions:   short  medium  long  xlong  vnc  gpu  bigmem  special  gpu-a40
  Grace GPUs in gpu partition:  a100:2  a40:3  rtx:2  t4:4

  Showing 1/2 of total cores and memory for using 1 x a100 GPU

  (CPU-billing + (GPU-billing * GPU-count)) * hours * nodes =   SUs
  (         25 + (      72 *        1)) * 96 *  1 = 9,312

#!/bin/bash
#SBATCH --job-name=my_job
#SBATCH --time=4-00:00:00
#SBATCH --nodes=1       # max 32 nodes for partition gpu
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=24
#SBATCH --mem=180G
#SBATCH --gres=gpu:a100:1
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
```

https://hprc.tamu.edu/kb/Software/useful-tools/maxconfig

Checking GPU Configuration & Availability on ACES

- Use the command line (shell) to see the current GPU configuration and availability.
- The GPU configuration can change since ACES is a composable resource cluster.
- If there are no GPUs in the AVAILABILITY output, it means that a GPU job that you submit may take a while to start.
- AlphaFold does not support running on PVC GPUs.

```
[username@grace ~]$ gpuavail

    CONFIGURATION
NODE            NODE
TYPE            COUNT
---------------------
gpu:a100:2      100
gpu:a40:3       15
gpu:rtx:2       9
gpu:t4:4        8


                    AVAILABILITY
NODE     GPU        GPU     GPUs    CPUs    GB MEM   PARTITION
NAME     TYPE       COUNT   AVAIL   AVAIL   AVAIL
-------------------------------------------------------------
g018     a100       2       2       48      360      vnc,gpu,special
g056     a100       2       2       48      360      vnc,gpu,special
g017     a100       2       2       48      360      vnc,gpu,special
g021     a100       2       2       48      360      vnc,gpu,special
g043     a100       2       2       48      360      vnc,gpu,special
```

https://hprc.tamu.edu/kb/Software/useful-tools/gpuavail

Check non-GPU node Availability

Use the **cpuavail** command to see non-GPU nodes readily available for jobs.

```
[username@grace ~]$ cpuavail

     CONFIGURATION                          AVAILABILITY
  NODE               NODE                NODE          CPUs GB MEM
  TYPE               COUNT               NAME          AVAIL     AVAIL
  ---------------------                  ----------------------------
  CPU-only           808                 c001          16        330
  GPU                132                 c005          16        107
                                         c006          16        104
                                         c007          16        104
                                         c008          16        104
                                         c009          16        104
                                         c010          16        104
                                         c011          16        104
                                         c013          16        104
                                         c014          16        104
                                         c015          16        104
                                         c016          16        104
```

https://hprc.tamu.edu/kb/Software/useful-tools/cpuavail

Cluster maintenance

- You can use the maintenance command to see if there is a scheduled cluster maintenance.

```
[username@grace ~]$ maintenance

 The scheduled 11 hour Grace maintenance will start in:

   3 days    16 hours    41 minutes

 Scheduled jobs will not start if they overlap with this maintenance window.
```

A 7-day job submitted at the time of the above message will remain queued and will not start until after the maintenance is complete.

Add Useful alias Commands in your ~/.bashrc

```
# list files newest to oldest
alias lhl='ls -lsth'
alias lh='ls -lsth | head'
alias lhh='ls -lsth | head -20'
alias lhhh='ls -lsth | head -30'


# show your running jobs and usage
alias q='squeue -u $USER'
alias p='pestat -u $USER'
```

Next Generation Sequencing Technologies

Illumina Sequencing Technology

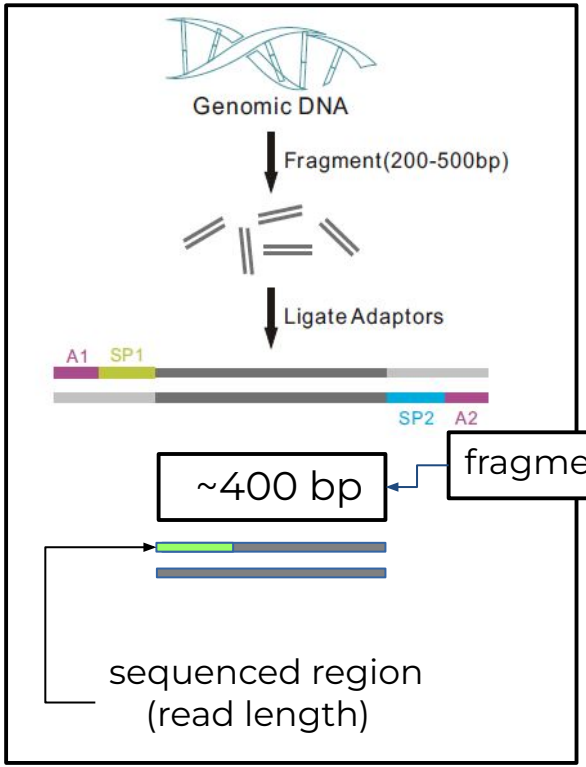
images are not to scale

http://www.illumina.com/systems/sequencing-platforms.html

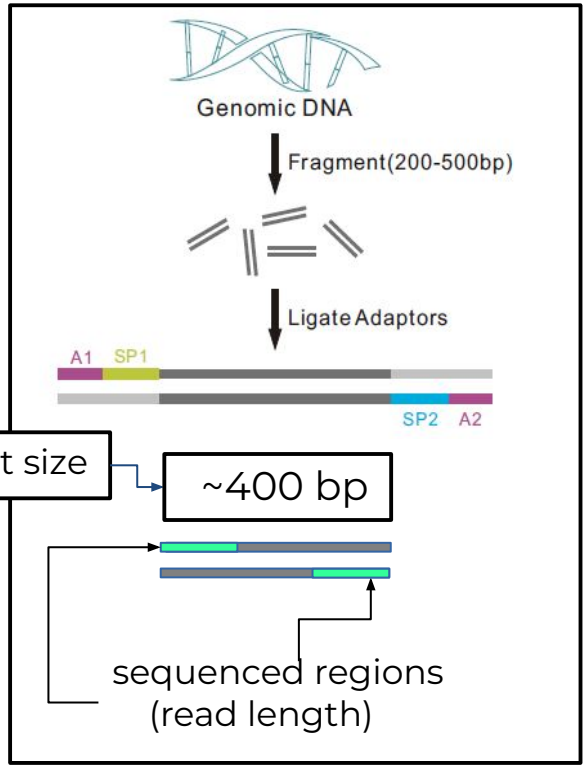
Illumina Sequencing Libraries

illumina.com

single end

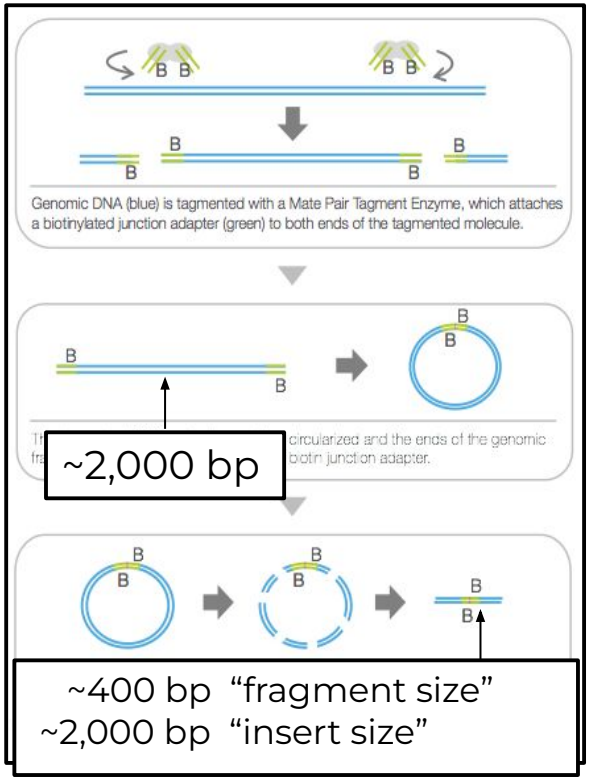


paired ends

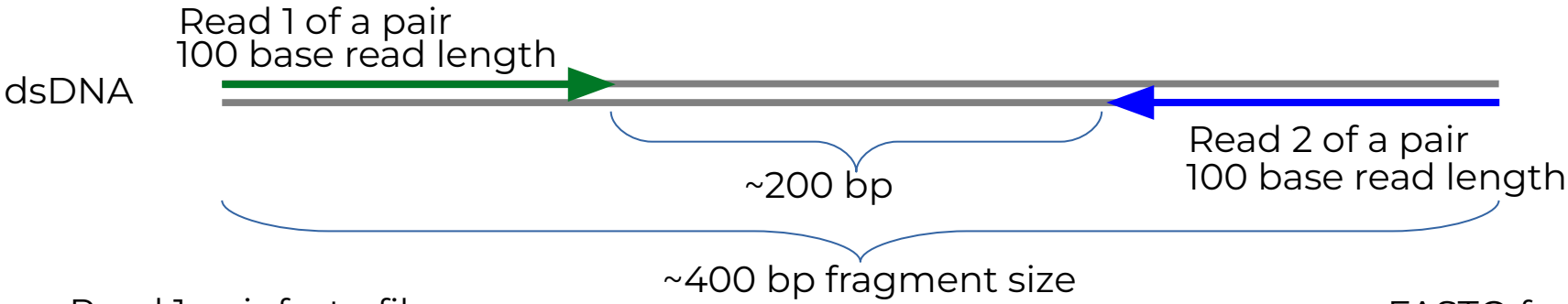


fragment size

mate pairs



Paired End Reads



Read 1 pair fastq file

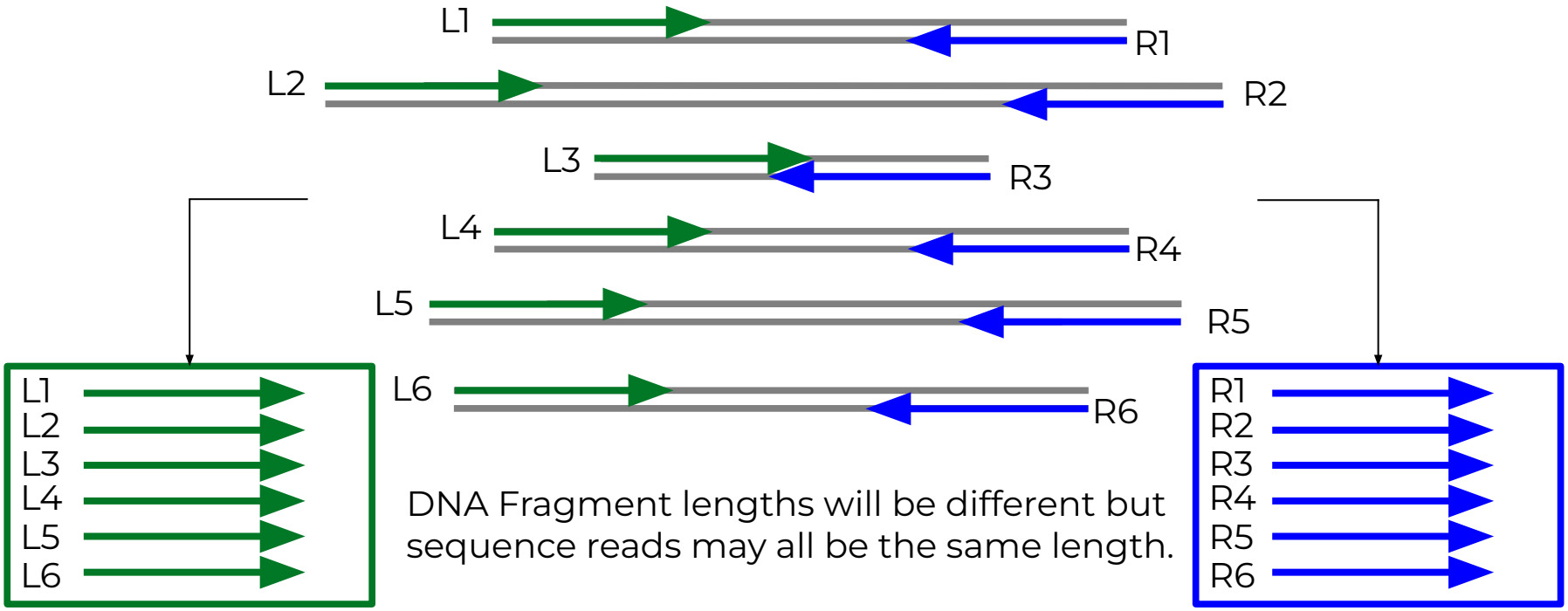
FASTQ format

```
@M00861:1:000000000-A36BE:1:1101:14650:1529 1:N:0:8
TTCTTAAAAATACCATAAAAGGCTTAAACTTGCCATTTACGACGGATTAATTCCAACTCTTTTCGGCTATCTTCATCTTTTAAGGTTAAATGACTCATAACGG
+
FFFHBFFHHIIIIIIHFHHCGEFGHHIHHHIHD/?DGGHHH@DEB,5EGHGHHIIHIF?FGGHHCCBFDGHFHDGHGFFFGDFHH?DFHDFFHHFHFHFFHHH
```

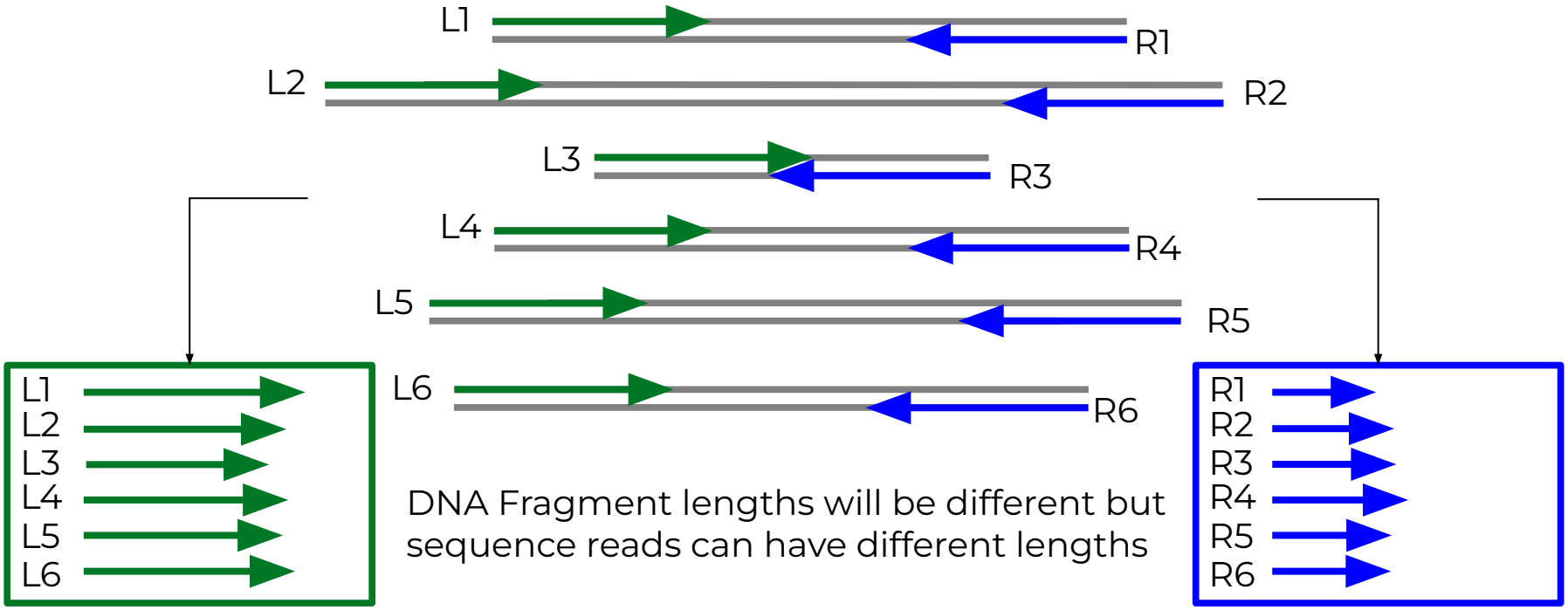
Read 2 pair fastq file

```
@M00861:1:000000000-A36BE:1:1101:14650:1529 2:N:0:8
ACTAAAAATCAATTTTATCAATTTCAAGCTCTACCTTATTTACTCATTATTTTAGTGATGGCCACTTTAATAAAAATATTGGTAGCATATTTTGCAATAGCGG
+
BFFHIHHHFHHDGHIHHIHHHGHHHHHFHHDFFHIHIIIHIHDFHHHIHIIIH=AAFHIIIHFGFHHHHHGGHHIHHFGFFFEGGHHHDGHHH/CGHIFFHHH
```

Maintain Read Pair Order



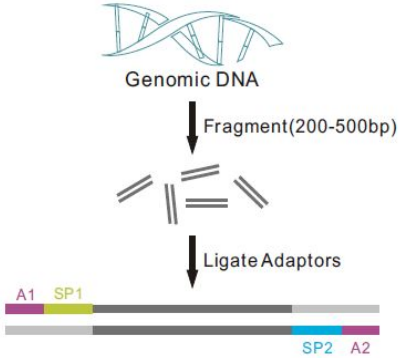
MiSeq Can Perform Initial QC Trimming



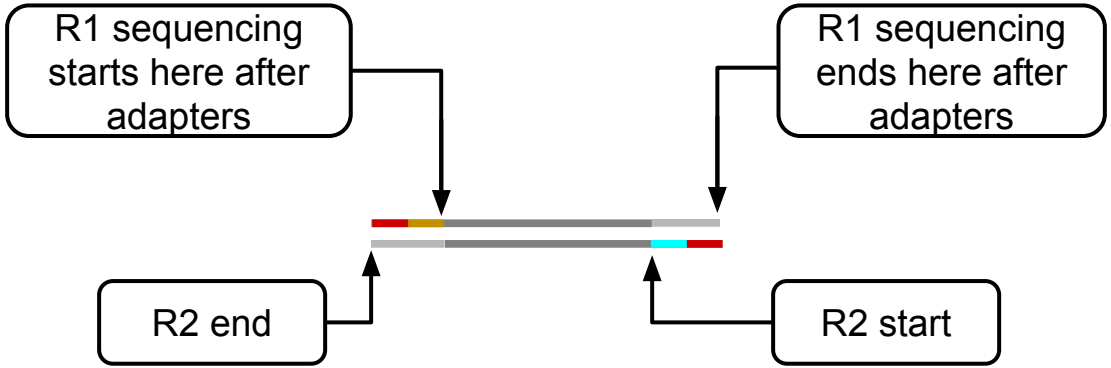
Left Read 1 paired end fastq file

Right Read 2 paired end fastq file

Adapters in Sequence Reads

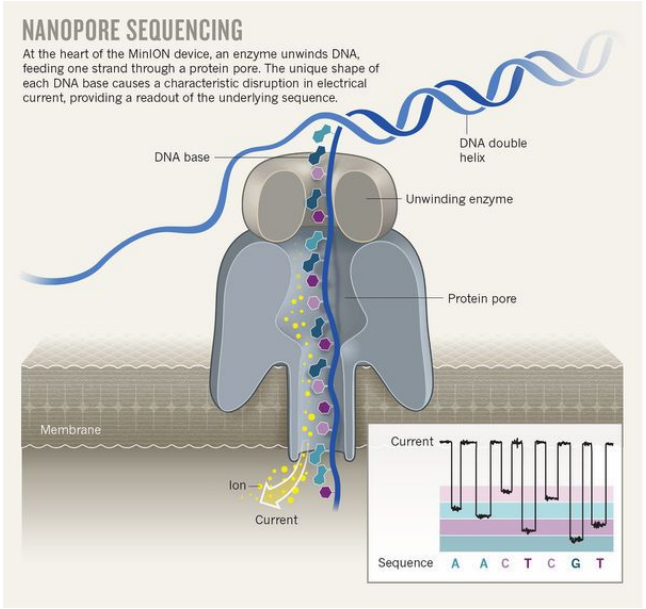


~250 bp fragment size and 300 bp read length



Oxford Nanopore Long Read Sequencing

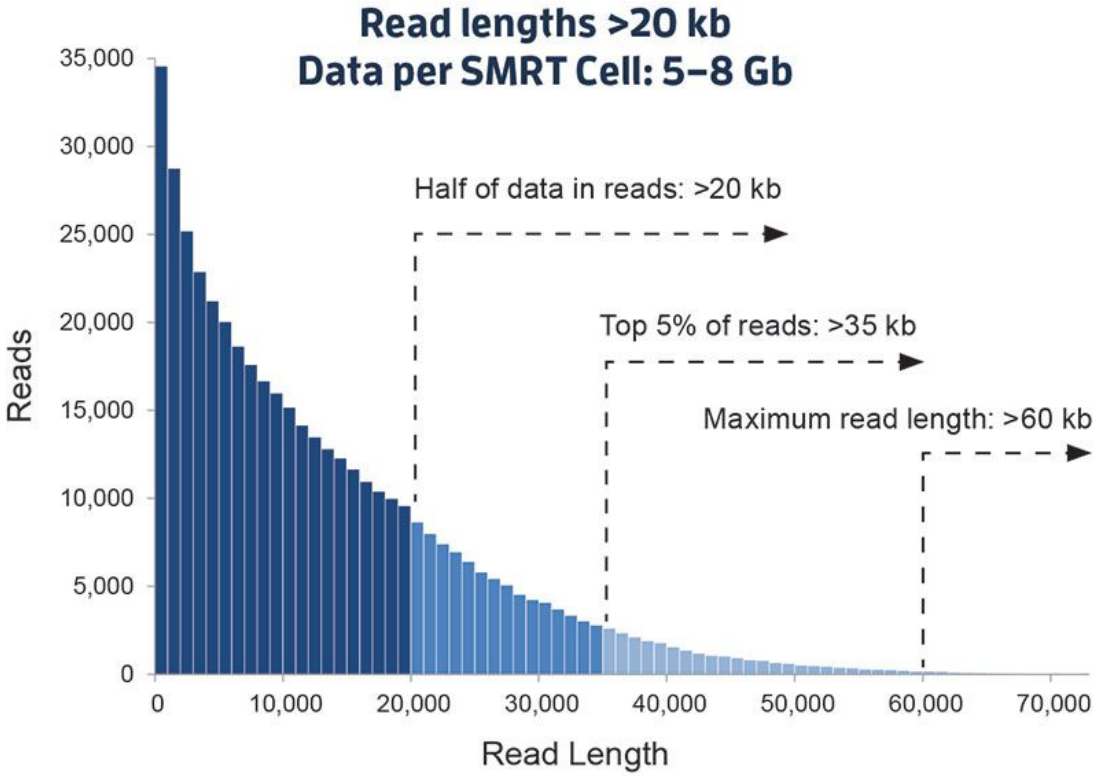
Oxford Nanopore Technologies (ONT)



http://blogs.nature.com
TechBlog: The nanopore toolbox
16 Oct 2017 | 12:00 GMT | Posted by Jeffrey Perkel

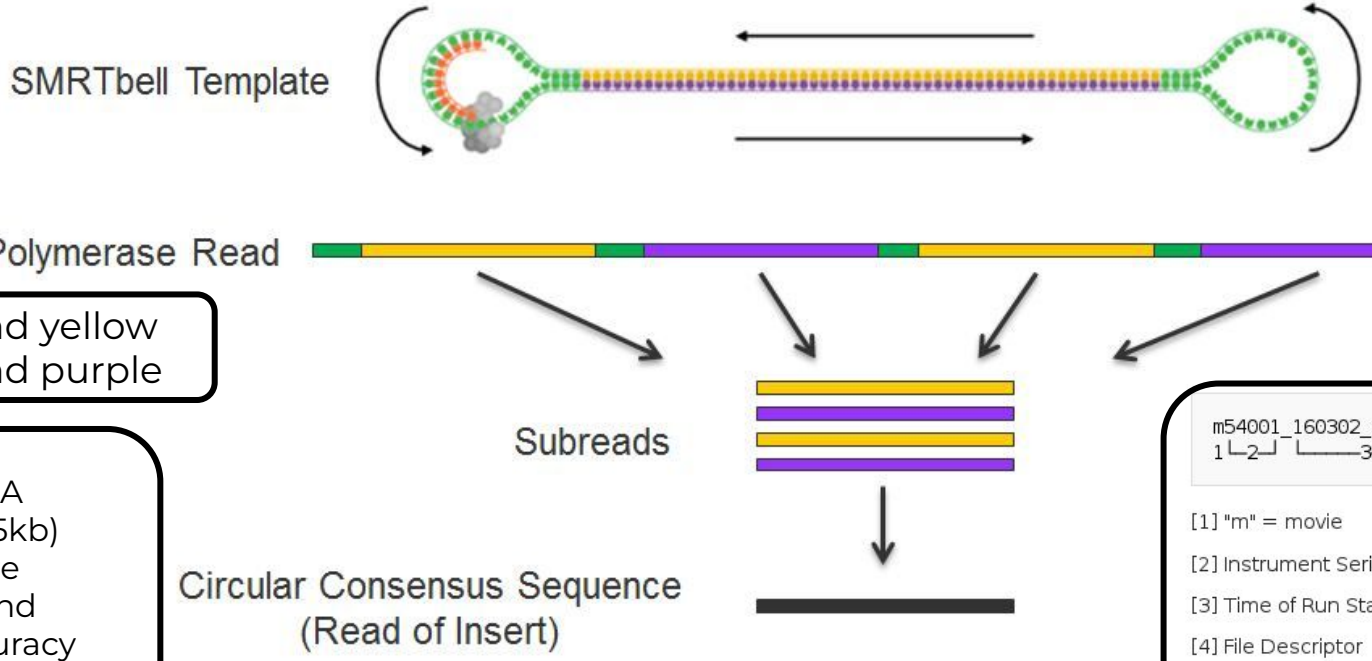
PacBio Long Read Sequencing

Sequel Sequence



pacb.com

PacBio Long Read Sequencing



\+ Strand yellow
\- Strand purple

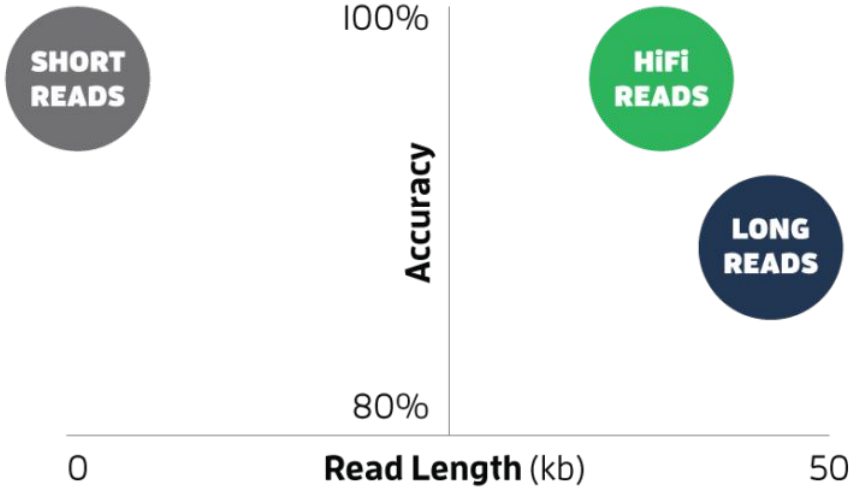
Shorter DNA fragment (5kb) equals more subreads and higher accuracy than longer (60kb)

```
m54001_160302_121501.subreads.bam
1└2┘ └─────3──────┘└──4──┘ └5┘
```

[1] "m" = movie
[2] Instrument Serial Number
[3] Time of Run Start (`yymmdd_hhmmss`)
[4] File Descriptor
[5] File Extension

pacb.com

PacBio HiFi Reads



https://www.pacb.com/blog/hifi-webinar

- HiFi reads are not subreads since they are already error corrected.
- HiFi utilize ssingle-molecule consensus rather than multiple-molecule consensus.

NGS Tools on Grace

Where to Find NGS Tools

- Search TAMU HPRC Documentation.
  - https://hprc.tamu.edu/kb/Software/Bioinformatics
- Type any the following Unix commands to see which tools are already installed on Grace (replace toolname with sw name).
  - `module spider toolname` (not case sensitive, but reads the entire output)
  - `module key assembly` (some modules may be missed because this searches tool descriptions)
- If you are unable to find a tool that you want installed on Grace, send an email with the URL link to: help@hprc.tamu.edu
- Useful websites: long-read-tools.org

PacBio Sequencing Tools

- Sequence Alignments
  - Minimap2
- Genome Assembly
  - Canu: PacBio long read assembler
    - grid mode not supported on HPRC clusters
  - Unicycler: bacterial genomes
  - wtdbg2: 10x faster than Canu
    - assembly is very close but not as complete as Canu
  - Flye: PacBio and ONT reads; metagenome also available
- Improve draft assemblies
  - Purge_Haplotigs
  - Circlator

https://hprc.tamu.edu/kb/Software/Bioinformatics/PacBio_tools

Tool Suites

- SMRT-Link (PacBio)
  - contains command line versions of additional SMRT Analysis tools
  - PacBio's open-source SMRT Analysis software suite is designed for use with Single Molecule, Real-Time (SMRT) Sequencing data.

bam2fasta	bam2fastq	bamsieve	bax2bam	blasr	ccs	cleric	cromwell
daligne	r	daligner_p	datander	dataset	dazcon	DB2Falcon	DB2fasta
DBdump	DBdust	DBrm	DBshow	DBsplit	DBstats	dexta	falconc
falconcpp	fasta2DB	fuse	gcpp	HPC.daligner	HPC.REPmask		HPC.TANmask
ipdSummary	ipython	ipython2	isoseq3	juliet	julietflow	LA4Falcon	LA4Ice
laa	laagc	LAmerge	LAsort	lima	minimap2	motifMaker	pbalign
pbcromwell	pbdagcon	pbindex	pbmm2	pbservice	pbsv	pbvalidate	ra
REPmask	samtools	sawriter	summarizeModifications		TANmask	undexta	womtool

```
module spider SMRT-Link
```

Genome Hybrid Assemblers (Long-reads + Illumina reads)

- SPAdes
  - subreads as input files
  - no need to correct subreads with short reads prior to assembly
  - uses long reads for gap closure and repeat resolution
- MaSuRCA
  - All long-reads must be in a single fasta file
- Unicycler
  - assembly pipeline for bacterial genomes
  - circularises replicons without the need for a separate tool like Circulator

Grace Software Toolchains and Modules

- search for module names using module spider

```
module spider samtools
```

- use module spider with module name for details on how to load module

```
module spider SAMtools/1.18
```

- read output to see which other module(s) to load first

```
module load GCC/12.3.0 SAMtools/1.18
```

show loaded modules

```
module list
```

- see what other modules are compatible with the loaded module(s)

```
module avail bwa
```

- load additional compatible modules

```
module load BWA/0.7.17
```

- list then unload all loaded modules

```
module list
```

```
module purge
```

Use $TMPDIR Whenever Possible

- Use the $TMPDIR if the application you are running can utilize a temporary directory for writing temporary files which are deleted when the job ends.
- A temp directory (**$TMPDIR**) is automatically assigned for each job which uses the local disk on the compute node–not the /scratch shared file system.
  - Especially useful when a computational tool writes tens of thousands of temporary files which are deleted when the job is finished and are not needed for the final results
  - Useful because files on **$TMPDIR** will not count against your file quota
  - Don't use **$TMPDIR** if your software uses temporary files for restarting where it left off if it should stop before completion.
  - Will significantly speed up an mpiBLAST job

```
java -Xmx350g -jar $EBROOTPICARD/FastqToSam.jar TMP_DIR=$TMPDIR \
FASTQ=$pe1_1 FASTQ2=$pe1_2 OUTPUT=$outfile SAMPLE_NAME=$sample_name \
SORT_ORDER=$sort_order MAX_RECORDS_IN_RAM='null'
```

Quality Control (QC)

QC Evaluation

- Use FastQC to visualize quality scores
  - Displays quality score distribution of a subset of ~200,000 reads
    - Input is a fastq file or files
    - Can disable grouping (binning) of sequence regions
  - Will alert you of poor read characteristics
  - Can be run as a GUI or a command line interface

```
module load FastQC/0.11.9-Java-11
```

- FastQC will process using one CPU core per file
  - If there are 10 fastq files to analyze and 4 cores are used, 4 files will start processing and 6 will wait in a fastqc queue
  - If there is only one fastq file to process then using 10 cores does not speed up the process

Digital Normalization for Assembly

- Reduce memory requirements by reducing the number of redundant sequence reads if you have a very high sequencing coverage (> 200x)
- Used for genome and transcriptome assembly, not variant calling or quantitative analysis (ChIP-seq, RNA-seq for expression profiling)
- Trinity 2.4.0+ automatically normalizes reads to a depth of 50x using a modified version of seqtk
- The **`bbnorm.sh`** script in BBMap can normalize reads
  - use reformat.sh to subsample

```
module spider BBMap
```

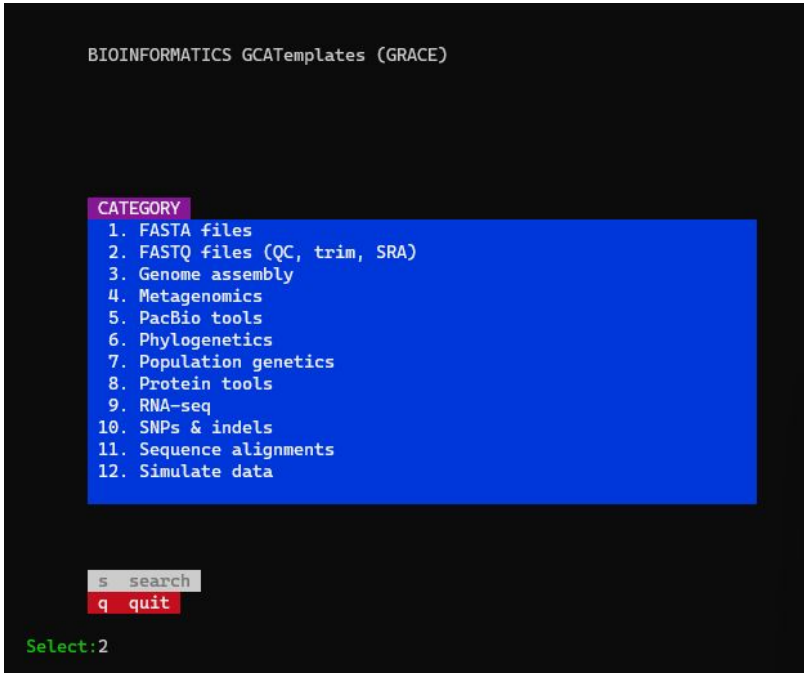
Template Job Scripts

GCATemplates

Genomic Computational Analysis Templates

```
gcatemplates
```

- GCATemplates is a collection of template job scripts for each of the HPRC clusters.
- The template scripts are configured with example input files and can be run without changes to get an idea of how the tool and job scheduling work.
- Update the template script as needed for larger datasets, different option values or additional options.



https://hprc.tamu.edu/kb/Software/useful-tools/GCATemplates

FastQC Template Job Script

```
mkdir $SCRATCH/ngs_class
```

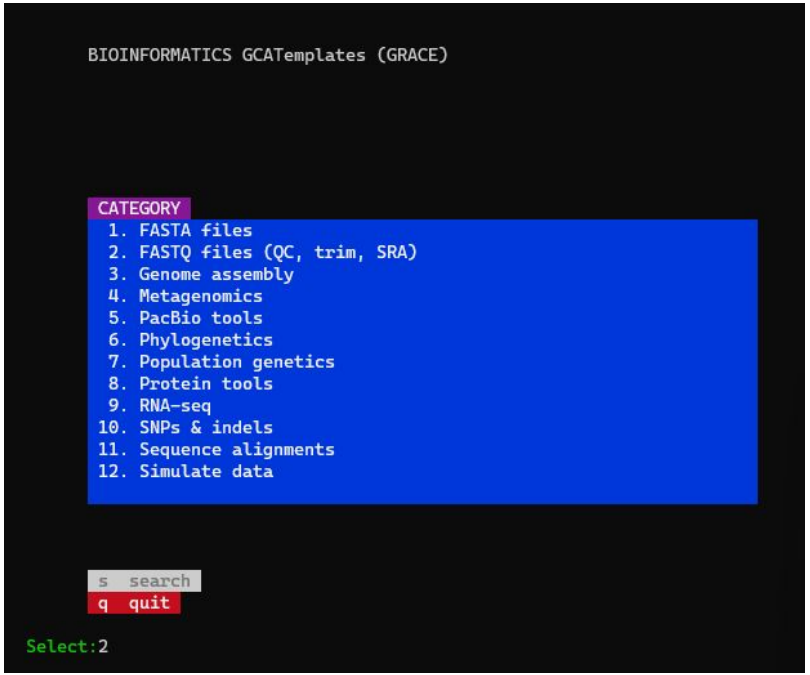
```
cd $SCRATCH/ngs_class
```

```
gcatemplates
```

For practice, we will copy a template file.

- Enter 2, then find the template that contains **fastqc**
  - or use the search function to find **fastqc**
- Final step will save a template job script file to your current working directory

Genomic Computational Analysis Templates



Sample GCATemplate Job Script (Grace)

```
#!/bin/bash
#SBATCH --job-name=fastqc             # job name
#SBATCH --time=01:00:00               # max job run time dd-hh:mm:ss
#SBATCH --ntasks-per-node=1           # tasks (commands) per compute node
#SBATCH --cpus-per-task=2             # CPUs (threads) per command
#SBATCH --mem=4G                      # total memory per node
#SBATCH --output=stdout.%j            # save stdout to file (%j is jobid)
#SBATCH --error=stderr.%j             # save stderr to file (%j is jobid)

module purge
module load FastQC/0.12.1-Java-11

<<README
        - FASTQC manual: http://www.bioinformatics.babraham.ac.uk/projects/fastqc/Help
README
################################### VARIABLES ###################################
# TODO Edit these variables as needed:
########## INPUTS ##########
pe1_1='/scratch/data/bio/GCATemplates/data/miseq/c_dubliniensis/DR34_R1.fastq.gz'
pe1_2='/scratch/data/bio/GCATemplates/data/miseq/c_dubliniensis/DR34_R2.fastq.gz'

######## PARAMETERS ########
threads=$SLURM_CPUS_PER_TASK

########## OUTPUTS #########
output_dir='./'

################################### COMMANDS ####################################
fastqc -t $threads -o $output_dir $pe1_1 $pe1_2

<<CITATION
        - Acknowledge TAMU HPRC: https://hprc.tamu.edu/research/citations.html
        - FastQC: http://www.bioinformatics.babraham.ac.uk/projects/fastqc
CITATION
```

Sample GCATemplate Job Script (Grace)

```
#!/bin/bash
#SBATCH --job-name=fastqc              # job name
#SBATCH --time=01:00:00                # max job run time dd-hh:mm:ss
#SBATCH --ntasks-per-node=1            # tasks (commands) per compute node
#SBATCH --cpus-per-task=2              # CPUs (threads) per command
#SBATCH --mem=4G                       # total memory per node
#SBATCH --output=stdout.%j             # save stdout to file (%j is jobid)
#SBATCH --error=stderr.%j              # save stderr to file (%j is jobid)

module purge
module load FastQC/0.12.1-Java-11

<<README
        - FASTQC manual: http://www.bioinformatics.babraham.ac.uk/projects/fastqc/Help
README
################################### VARIABLES ###################################
# TODO Edit these variables as needed:
########## INPUTS ##########
pe1_1='/scratch/data/bio/GCATemplates/data/miseq/c_dubliniensis/DR34_R1.fastq.gz'
pe1_2='/scratch/data/bio/GCATemplates/data/miseq/c_dubliniensis/DR34_R2.fastq.gz'

######## PARAMETERS ########
threads=$SLURM_CPUS_PER_TASK

########## OUTPUTS #########
output_dir='./'

################################### COMMANDS ####################################
fastqc -t $threads -o $output_dir $pe1_1 $pe1_2

<<CITATION
        - Acknowledge TAMU HPRC: https://hprc.tamu.edu/research/citations.html
        - FastQC: http://www.bioinformatics.babraham.ac.uk/projects/fastqc
CITATION
```

These parameters are read by the job scheduler

Load the required module(s) first

This is a section of comments

This is a single line comment and not run as part of the script

This is the command to run the application

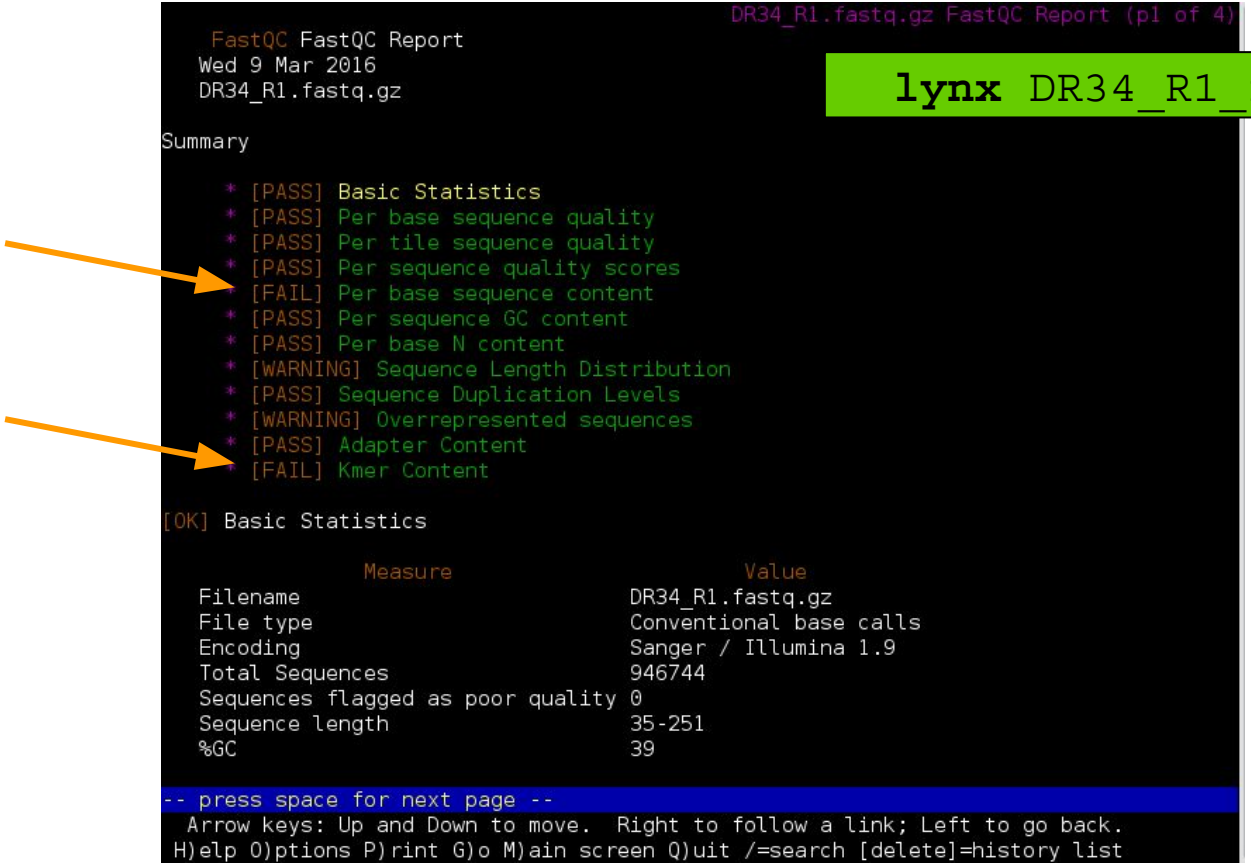
FastQC Exercise

- Use the GCATemplate for FastQC to submit a job evaluating the two sequence files
  - `cat run_fastqc_0.12.1_grace.sh`
  - `sbatch run_fastqc_0.12.1_grace.sh`
  - `squeue --me`
  - `myjob JOBID`
- After your fastqc job is complete, unzip the results file and you can view the results files with the **`lynx`** and **`eog`** Unix commands
  - use the Files app to view images

```
unzip DR34_R1_fastqc.zip
```

FastQC Report Using lynx

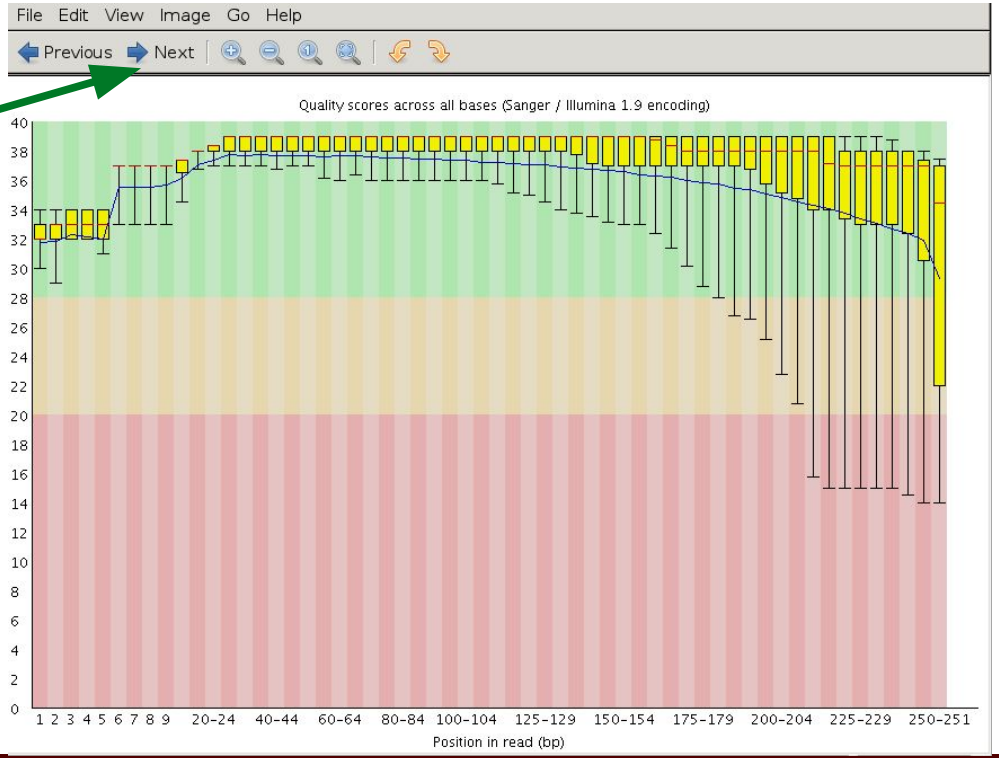
```
lynx DR34_R1_fastqc.html
```



FastQC Output Image Quality Distribution

```
eog DR34_R1_fastqc/Images/per_base_quality.png
```

click for the next image in the same directory, or use the left/right arrow keys

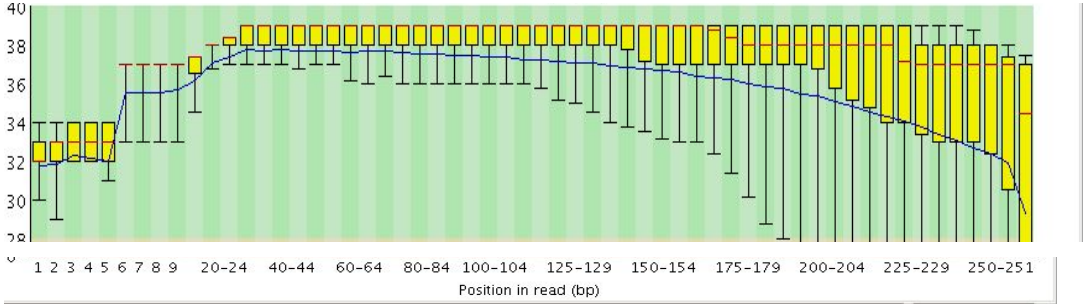


Prior to QC trimming

FastQC Output Image Quality Distribution

FASTQ format

```
@ERR504787.2.1 M00368:15:000000000-A0HKH:1:5:21261:10968-1 length=100
GATCGGAAGAGCACACGTCTGAACTCCAGTCACGATCAGATCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAAAAACAAAAAACATAATGCCGTAAA
+ERR504787.2.1 M00368:15:000000000-A0HKH:1:5:21261:10968-1 length=100
=:=4AD=B8A:+<A::1<:AE<C3*?F<B???<?:8:6?B*9BD;/638.=-'-.@7=).=A:6?DDDCBBBB9555&&)+((+2&&+(((((((()&&&+
@ERR504787.3.1 M00368:15:000000000-A0HKH:1:3:12724:25677-1 length=100
GATGTTTTGTTACTGATTGGAACCATGATTGGTGCTTTACTTGGTTTCTTCCTATTTAACCACAAGCCTGCCAAAGTATTTATGGGAGATGTAGGTAGTT
+ERR504787.3.1 M00368:15:000000000-A0HKH:1:3:12724:25677-1 length=100
BCCFDEFFHHHHHJJJJIJJJJJJJJJJIJJJFHIJJJIJJJJJIJJJJJJJIJJJIJIIJJJJJFHJJJJIJJHH=CHHFFFFFFEDDEDEEEDCCDCE
@ERR504787.5.1 M00368:15:000000000-A0HKH:1:2:16161:12630-1 length=100
TATTTTAAGTGACCAAGGAATGACTCCCCAATCATGGCTGTATCAACTCCAAAATTTTCTGCAACAGTCGCTGAAATATCTGCAAAATGCCCTTGTGGAA
+ERR504787.5.1 M00368:15:000000000-A0HKH:1:2:16161:12630-1 length=100
CCCFFFFFHHHHHJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJIJJJHIJHHGFFFFFFFEEEEEEDDDDDDDDDCDDDD
```

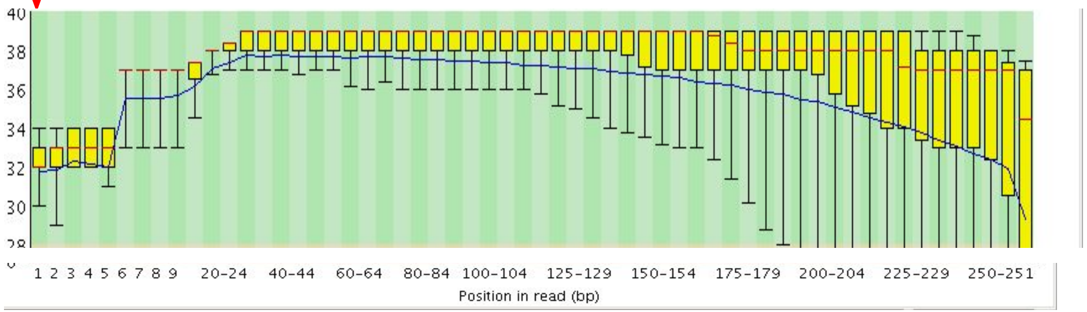


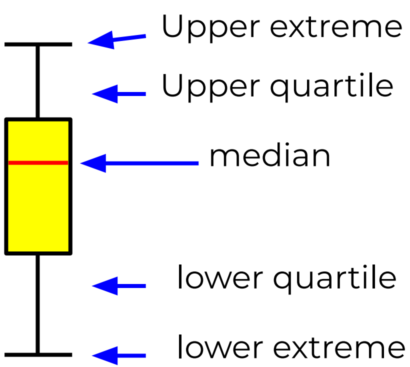
FastQC Output Image Quality Distribution

FASTQ format

```
@ERR504787.2.1 M00368:15:000000000-A0HKH:1:5:21261:10968-1 length=100
GATCGGAAGAGCACACGTCTGAACTCCAGTCACGATCAGATCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAAAAACAAAAAACATAATGCCGTAAA
+ERR504787.2.1 M00368:15:000000000-A0HKH:1:5:21261:10968-1 length=100
=:=4AD=B8A:+<A::1<:AE<C3*?F<B???<?:8:6?B*9BD;/638.=-'-.@7=).=A:6?DDDCBBBB9555&&)+((+2&&+((((((()&&&+
@ERR504787.3.1 M00368:15:000000000-A0HKH:1:3:12724:25677-1 length=100
GATGTTTTGTTACTGATTGGAACCATGATTGGTGCTTTACTTGGTTTCTTCCTATTTAACCACAAGCCTGCCAAAGTATTTATGGGAGATGTAGGTAGTT
+ERR504787.3.1 M00368:15:000000000-A0HKH:1:3:12724:25677-1 length=100
BCCFDEFFHHHHHJJJJIJJJJJJJJJJJIJJJFHIJJJIJJJJJIJJJJJJJIJJJIJIIJJJJJFHJJJJIJJHH=CHHFFFFFFFEDDEDEEEDCCDCE
@ERR504787.5.1 M00368:15:000000000-A0HKH:1:2:16161:12630-1 length=100
TATTTTAAGTGACCAAGGAATGACTCCCCAATCATGGCTGTATCAACTCCAAAATTTTCTGCAACAGTCGCTGAAATATCTGCAAAATGCCCTTGTGGAA
+ERR504787.5.1 M00368:15:000000000-A0HKH:1:2:16161:12630-1 length=100
CCCFFFFFHHHHHJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJIJJJHIJHHGFFFFFFFEEEEEEDDDDDDDDDCDDDD
```

Average quality score distribution at position one



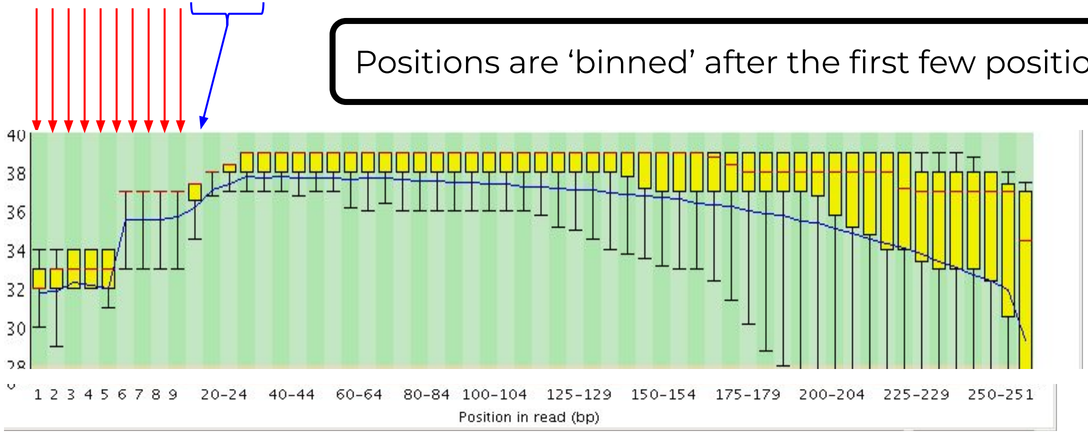


FastQC Output Image Quality Distribution

FASTQ format

```
@ERR504787.2.1 M00368:15:000000000-A0HKH:1:5:21261:10968-1 length=100
GATCGGAAGAGCACACGTCTGAACTCCAGTCACGATCAGATCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAAAACAAAAAACATAATGCCGTAAA
+ERR504787.2.1 M00368:15:000000000-A0HKH:1:5:21261:10968-1 length=100
=:=4AD=B8A:+<A::1<:AE<C3*?F<B???<?:8:6?B*9BD;/638.=-'-.@7=).=A:6?DDDCBBBB9555&&)+((+2&&+(((((((()&&&+
@ERR504787.3.1 M00368:15:000000000-A0HKH:1:3:12724:25677-1 length=100
GATGTTTTGTTACTGATTGGAACCATGATTGGTGCTTTACTTGGTTTCTTCCTATTTAACCACAAGCCTGCCAAAGTATTTATGGGAGATGTAGGTAGTT
+ERR504787.3.1 M00368:15:000000000-A0HKH:1:3:12724:25677-1 length=100
BCCFDEFFHHHHHJJJJIJJJJJJJJJJIJJJFHIJJJIJJJJJIJJJJJJJIJJJIJIIJJJJJFHJJJJIJJHH=CHHFFFFFFEDDEDEEEDCCDCE
@ERR504787.5.1 M00368:15:000000000-A0HKH:1:2:16161:12630-1 length=100
TATTTTAAGTGACCAAGGAATGACTCCCCAATCATGGCTGTATCAACTCCAAAATTTTCTGCAACAGTCGCTGAAATATCTGCAAAATGCCCTTGTGGAA
+ERR504787.5.1 M00368:15:000000000-A0HKH:1:2:16161:12630-1 length=100
CCCFFFFFHHHHHJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJIJJJHIJHHGFFFFFFEEEEEEDDDDDDDDDCDDDD
```

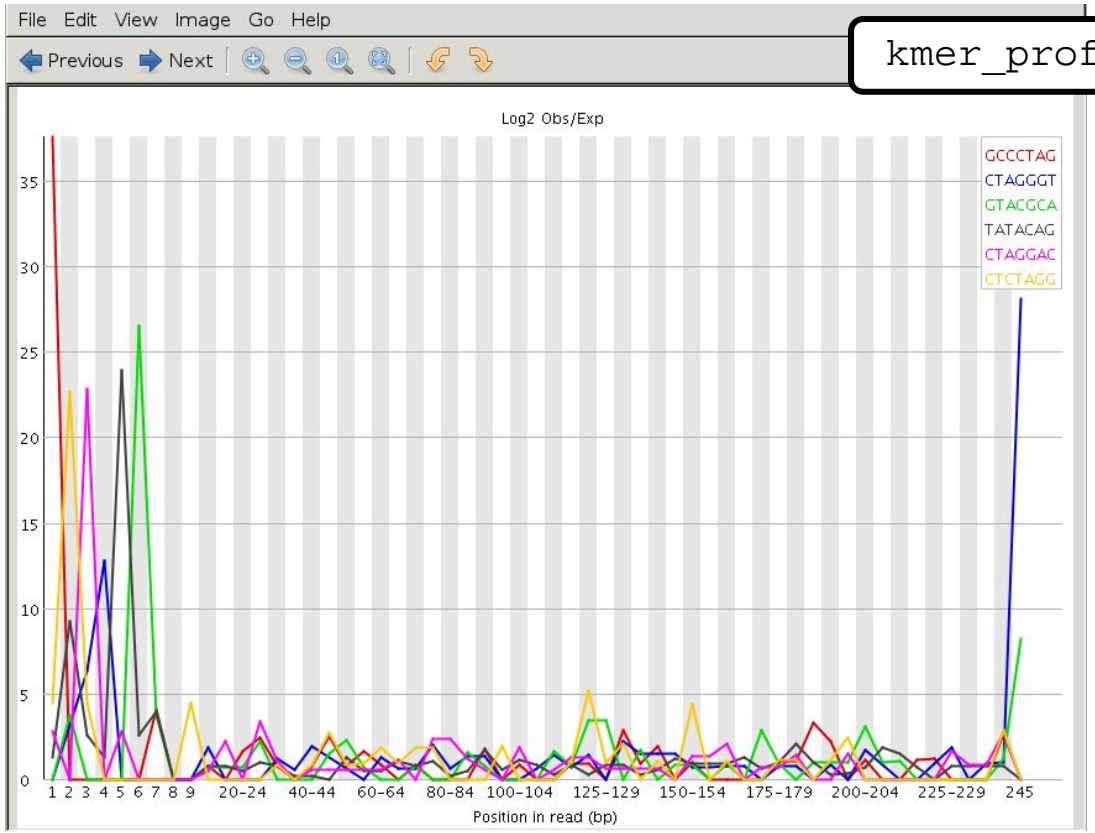
Positions are ‘binned’ after the first few positions



Illumina Transposon Insertion Site

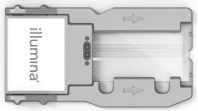
per_base_sequence_content.png

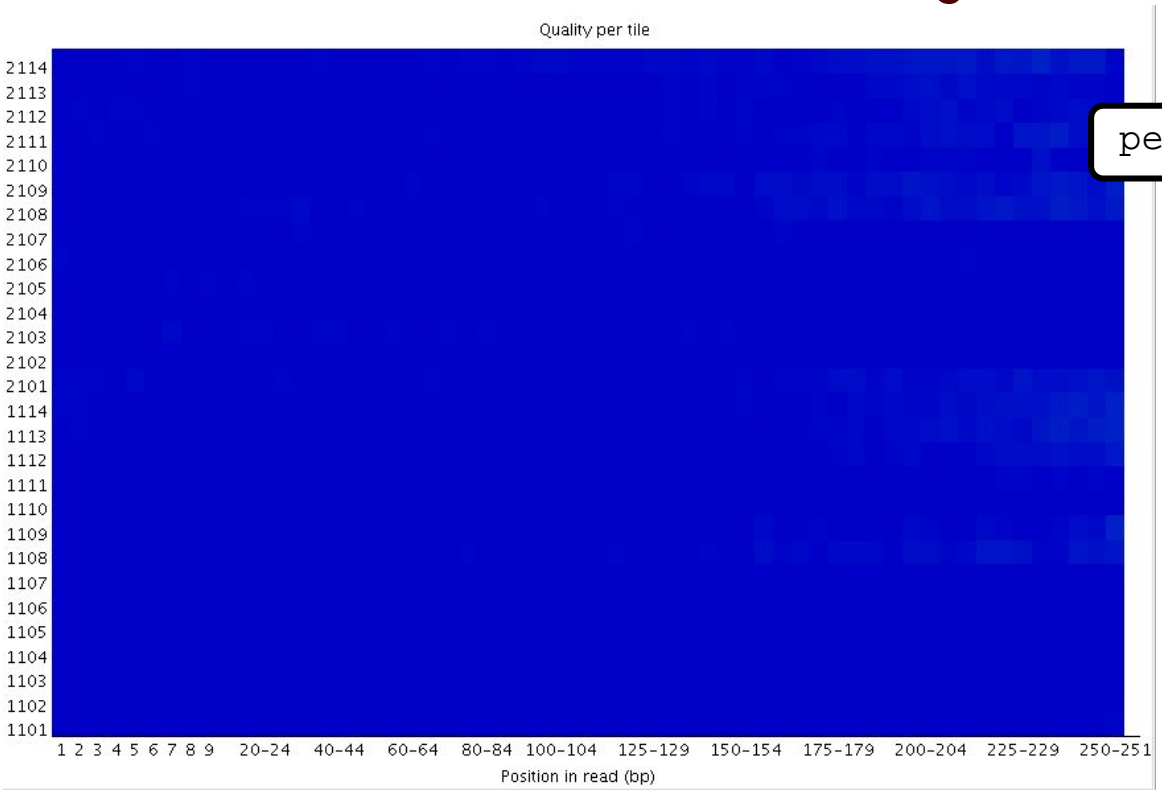
Illumina Transposon Insertion Site



FastQC Flowcell Quality Image

MiSeq flowcell





good quality — poor quality

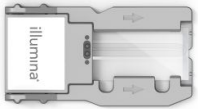
`per_tile_quality.png`

Flowcell quality mapping
Good per_tile quality

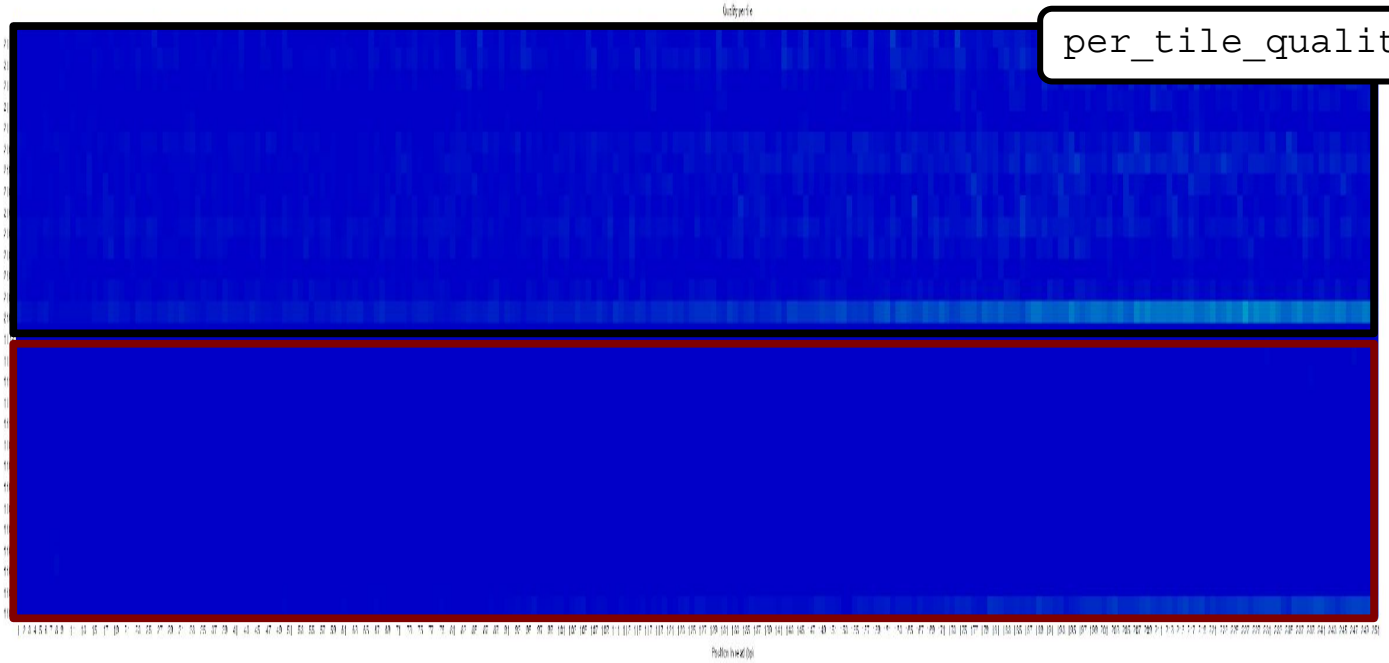
FastQC Flowcell Quality Image

bottom of flowcell

MiSeq flowcell



top of flowcell

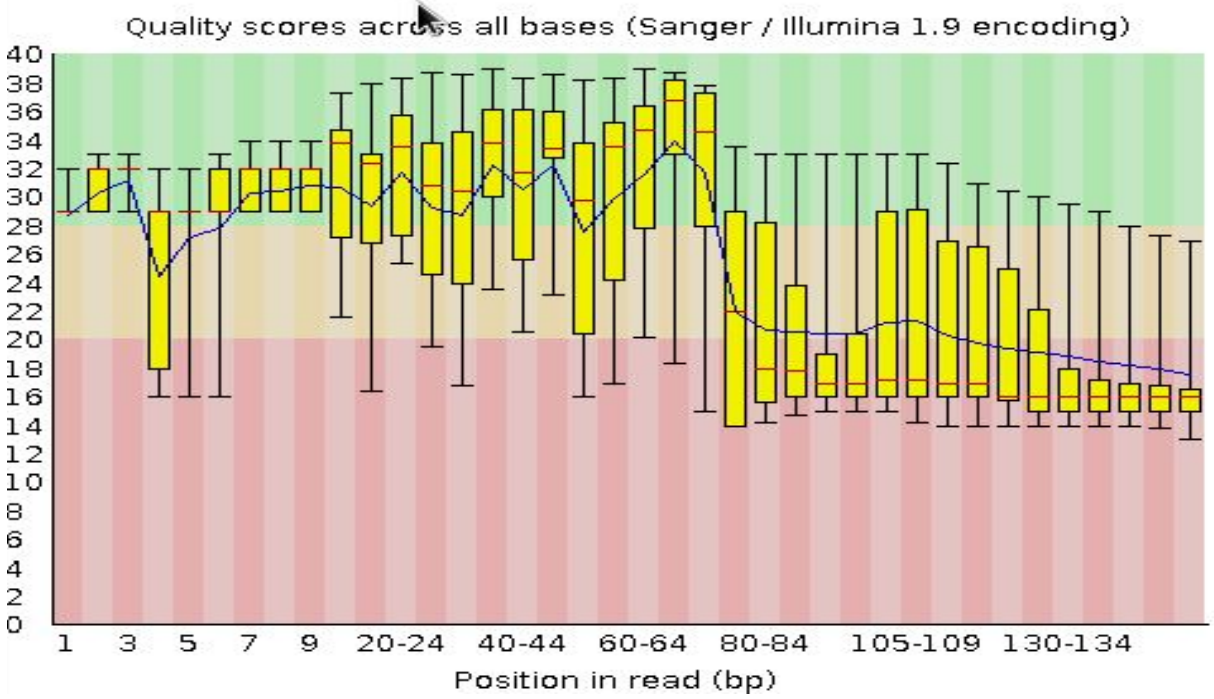


good quality — poor quality

Failed QC Examples

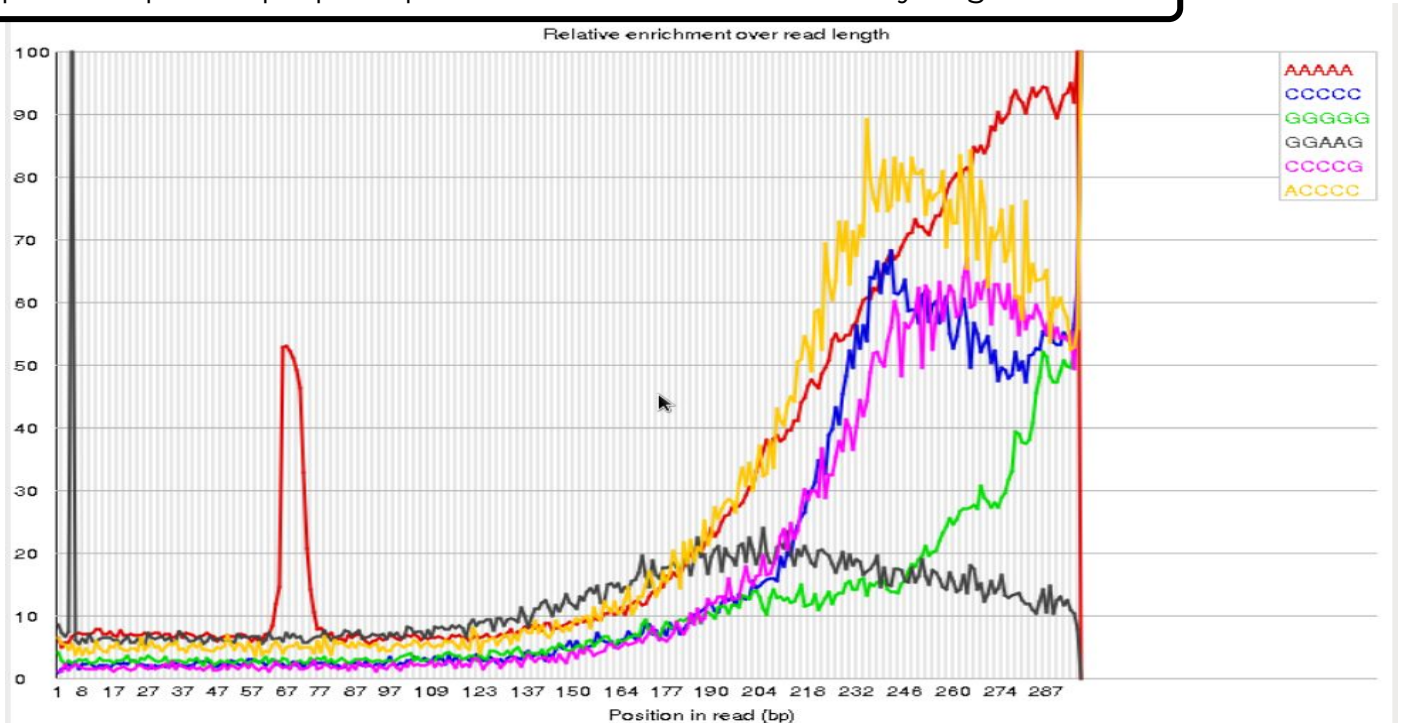
FastQC Output Image
Failed Per Base Sequence Quality

Example 1. Expired MiSeq mate-pair kit (9 months expired)



FastQC Output Image
Failed Kmer Content

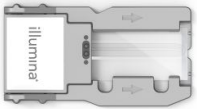
Example 2. Sequence prep adapters still on ends of DNA library fragments

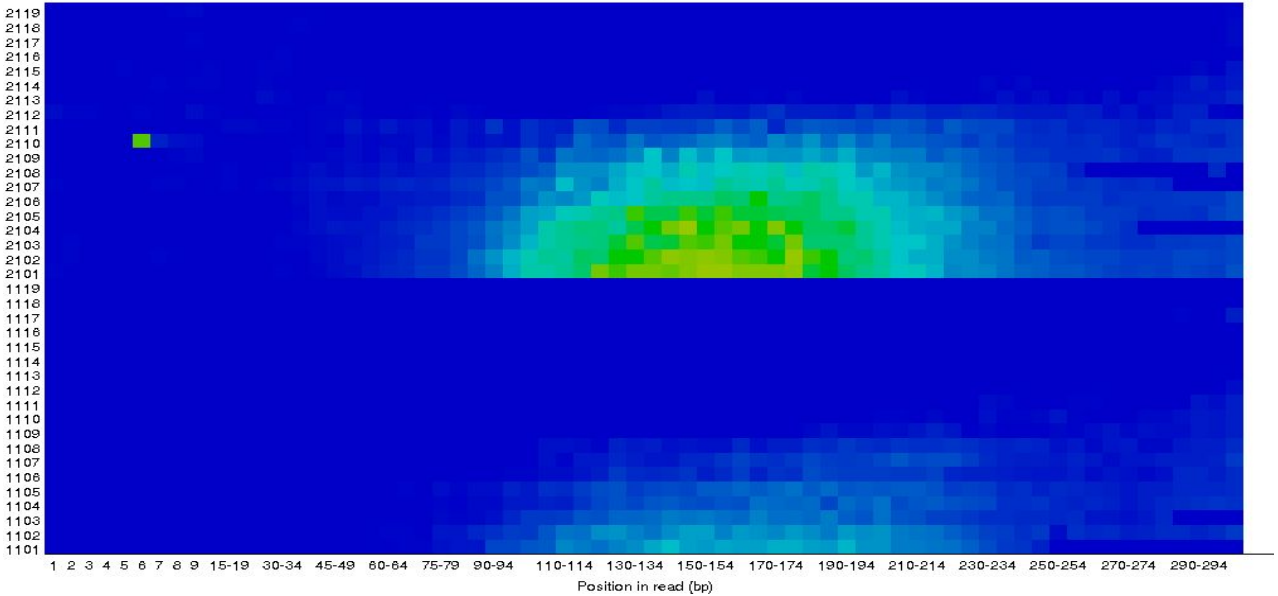


FastQC Output Image
Flowcell: Not Good per_tile Quality

Example 3. Faulty flowcell

MiSeq flowcell

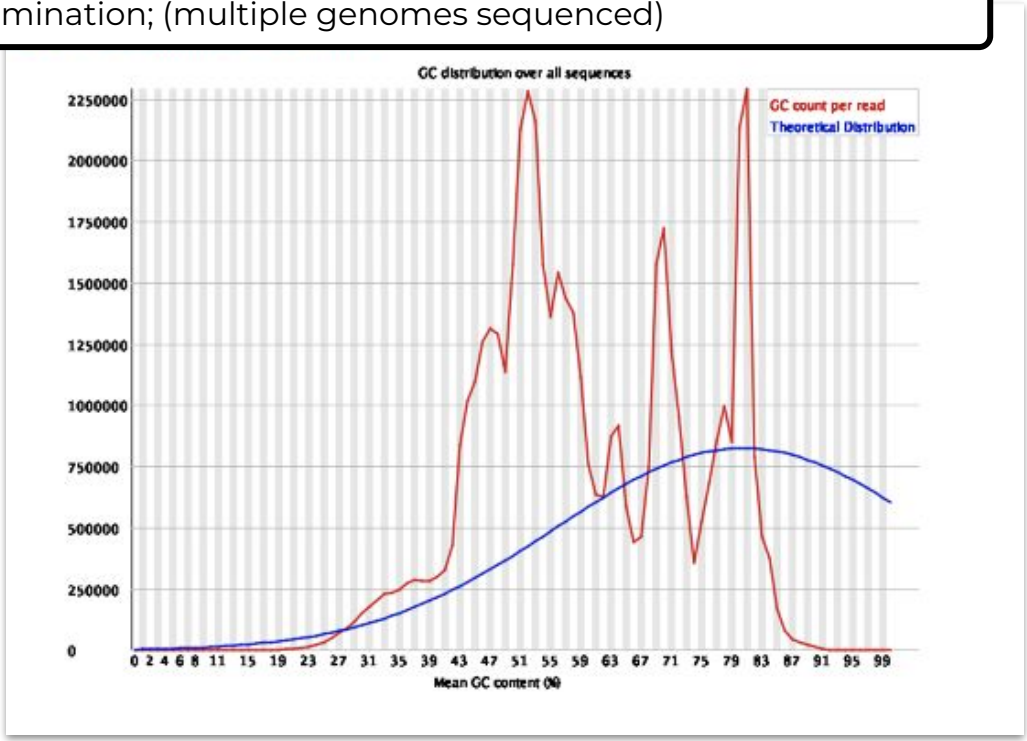




good quality — poor quality

FastQC Output Image
Failed Per Sequence GC Content

Example 4. Contamination; (multiple genomes sequenced)



QC Quality Trimming

- Sequence quality trimming tools

```
module spider Trimmomatic
module spider Trim_Galore
```

- Trimmomatic will maintain paired end read pairing after trimming
- Trim reads based on quality scores
  - Trim the same number of bases from each read or
  - Use a sliding window to calculate average quality at ends of sequences
- Decide if you want to discard reads with Ns
  - some assemblers replace Ns with As or a random base G, C, A or T
- Trim adapter sequences
  - Trimmomatic has a file of Illumina adapter sequences

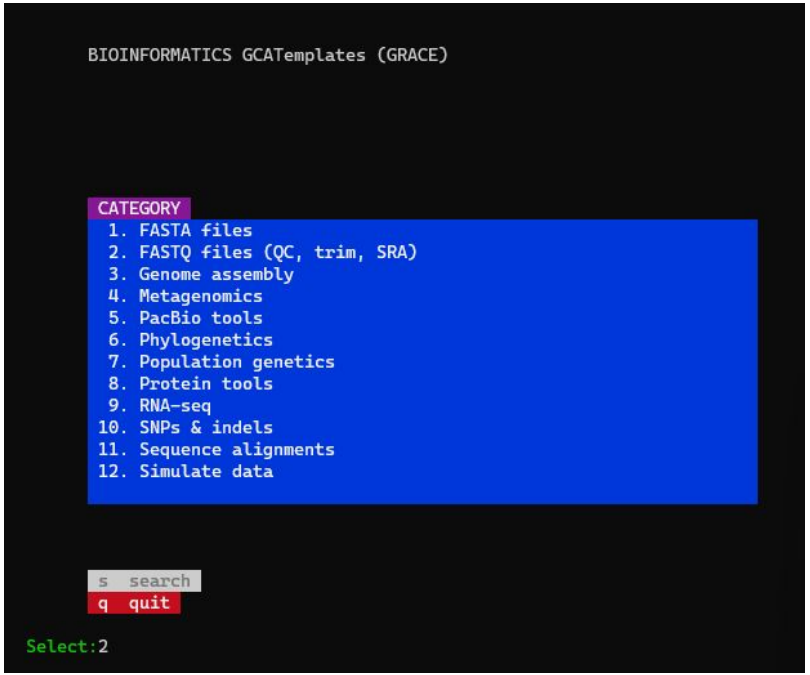
```
module load Trimmomatic/0.39-Java-11
ls $EBROOTTRIMMOMATIC/adapters/
```

Template Script for Trimming Sequence Reads based on Quality Scores

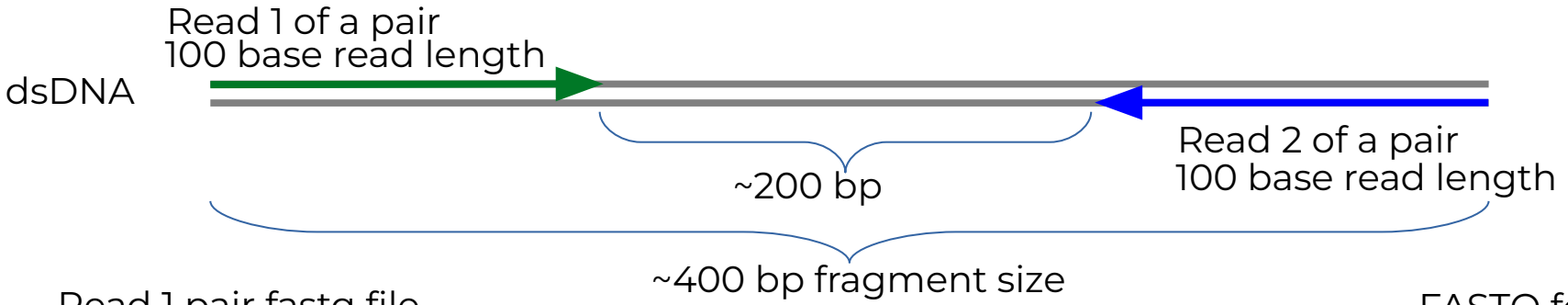
Genomic Computational Analysis Templates

```
gcatemplates
```

- Enter 2, then find the template that contains **trim_galore**
  - or use the search function to find **trim_galore**
- Final step will save a template job script file to your current working directory
- then submit the job script to the Slurm scheduler



Paired End Short Reads



Read 1 pair fastq file

FASTQ format

```
@M00861:1:000000000-A36BE:1:1101:14650:1529 1:N:0:8
TTCTTAAAAATACCATAAAAGGCTTAAACTTGCCATTTACGACGGATTAATTCCAACTCTTTTCGGCTATCTTCATCTTTTAAGGTTAAATGACTCATAACGG
+
FFFHBFFHHIIIIIIHFHHCGEFGHHIHHHIHD/?DGGHHH@DEB,5EGHGHHIIHIF?FGGHHCCBFDGHFHDGHGFFFGDFHH?DFHDFFHHFHFHFFHHH
```

Read 2 pair fastq file

```
@M00861:1:000000000-A36BE:1:1101:14650:1529 2:N:0:8
ACTAAAAATCAATTTTATCAATTTCAAGCTCTACCTTATTTACTCATTATTTTAGTGATGGCCACTTTAATAAAAATATTGGTAGCATATTTTGCAATAGCGG
+
BFFHIHHHFHHDGHIHHIHHHGHHHHHFHHDFFHIHIIIHIHDFHHHIHIIIH=AAFHIIIHFGFHHHHHGGHHIHHFGFFFEGGHHHDGHHH/CGHIFFHHH
```

Trimming PE Short Sequence Reads

File 1 from sequencer

100 bases

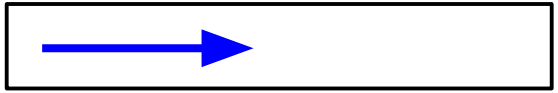
QC trim

100 bases

File 2 from sequencer

100 bases

QC trim



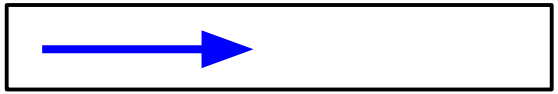
50 bases

minimun read length = 40

Resulting FASTQ Files with trimmed reads

Paired end 1 trimmed file

Paired end 2 trimmed file



Trimming PE Short Sequence Reads

File 1 from sequencer

100 bases

QC trim

100 bases

File 2 from sequencer

100 bases

QC trim

20 bases

minimun read length = 40

Resulting FASTQ Files with trimmed reads

Paired end 1 trimmed file

Paired end 2 trimmed file

Single end reads

Merge Overlapping Paired End Short Reads

fragment 1

dsDNA

---

fragment 2

dsDNA

Merge Overlapping Paired End Short Reads

fragment 1

dsDNA

---

fragment 2

dsDNA

Paired end read 1 (left)

Paired end read 2 (right)

Merge Overlapping Paired End Short Reads

fragment 1

dsDNA

fragment 2

dsDNA

Paired end read 1 (left)

Paired end read 2 (right)

Unpaired 'merged' read

Tools for merging overlapping reads:

```
module spider FLASH
```

```
module spider BBMap
```

```
module spider PEAR
```

Review Trimmed Files

- unzip the fastqc report for read file R1

```
ls trimmed_files
unzip trimmed_files/DR34_R1_val_1_fastqc.zip
```

- Compare the raw sequence FastQC and the trimmed FastQC reports for per_base_quality.png using the portal files app

Mapping Reads to a Reference Assembly

Mapping Short Reads to a Reference Assembly

- Align reads using bwa or bwa-mem2

```
module spider bwa
```

- Align reads using bwa-mem on GPUs

```
module spider Parabricks
```

- Align reads using bowtie or bowtie2

```
module spider Bowtie2
```

- genome index files for found here:

```
/scratch/data/bio/genome_indexes/
```

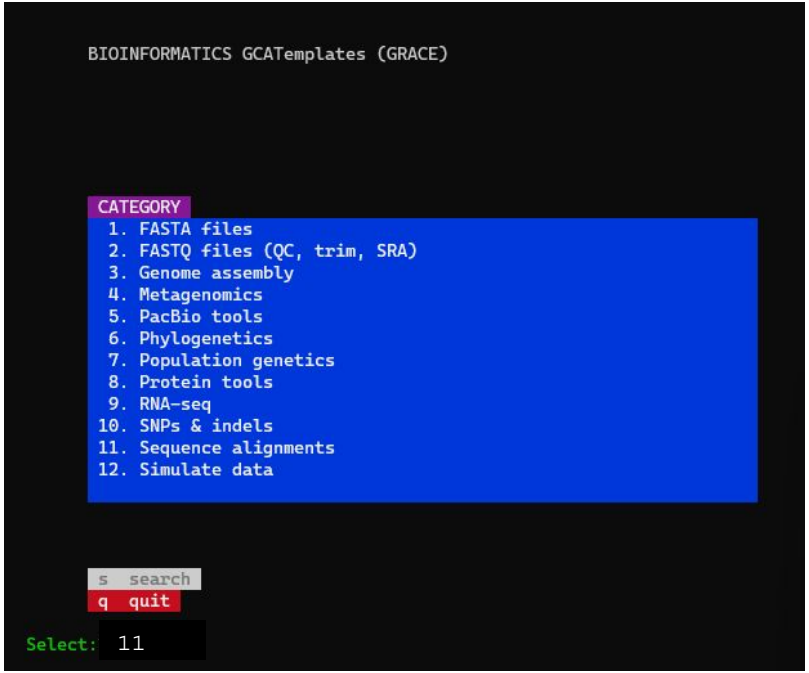
Send an email to help@hprc.tamu.edu if you need and index that is not found in the genome_indexes directory

Template for Sequence Alignment to a Reference Genome

Genomic Computational Analysis Templates

```
gcatemplates
```

- Ensure that your trim_galore job has completed
- Enter 11, then find the template that contains **bwa**
  - or use the search function to find **bwa**
- Final step will save a template job script file to your current working directory
- change the inputs to match your trim_galore files
- change the genome
- submit the job script to the Slurm scheduler



ref_genome=
/scratch/data/bio/genome_indexes/gmod_genomes/C_dubliniensis_CD36/bwa_0.7.12/C_dubliniensis_CD36_current_chromosomes.fasta

Create a symlink

Create a symlink which is like a shortcut pointing to a file in a different directory

The symbolic links are used to make the commands shorter for demonstration purposes only.

You do not need to make symbolic links in order to use **`samtools tview`**

```
ln -s /scratch/data/bio/training/genomes/c_dubliniensis.fa
```

Use the tab key when typing long paths and file names
Unix autocompletes directory and file names if they exist

```
ls -l c_dubliniensis.fa
```

Update bwa script with trim_galore output files

run_bwa_0.7.17_samtools_1.18_sort_pe_tmpdir_grace.sh

line #

```
25    pe1_R1='trimmed_files/DR34_R1_val_1.fq.gz'
26    pe1_R3='trimmed_files/DR34_R2_val_2.fq.gz'

28    ref_genome='c_dubliniensis.fa'


31    readgroup='dr34'
32    sample='DR34'
```

Visualize bam Alignment Files

SAMtools

- Sequence Alignment/Map format (sam)
  - view sam files using the UNIX command:
- Binary Alignment/Map format (bam)
  - Compressed (binary) sam files need samtools to view
    - `module load GCC/12.3.0 SAMtools/1.18`
- Create an index of the bam file using samtools
    - A samtools index is needed prior to viewing bam files in browsers

```
samtools index DR34_sorted.bam
```

```
DR34_sorted.bam.bai
```

Viewing sam/bam Files

Viewing the binary bam files using samtools

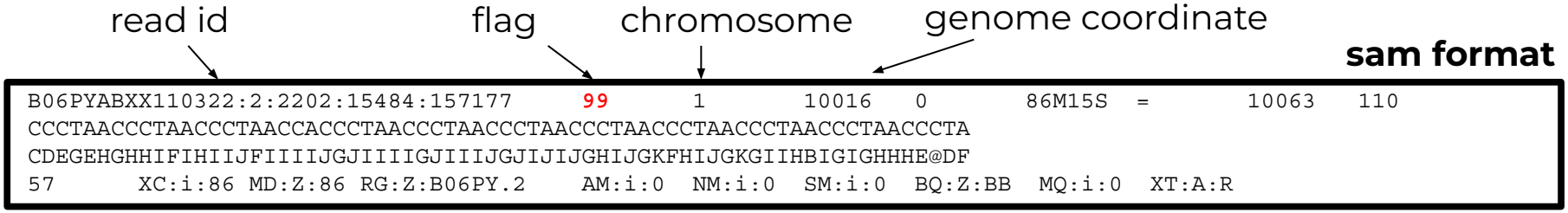
```
samtools view DR34_sorted.bam | more
```
view only alignments

```
samtools view -H DR34_sorted.bam
```
view only header

```
samtools view -h DR34_sorted.bam | more
```
view header + alignments

Sam Flags and Bits

- Flags describe alignments (the flag value is the sum of bits)



bits: **1** **2** 4 8 16 **32** **64** 128 256 512 1024 2048

1 + 2 + 32 + 64 = 99

- Filter bam alignments based on bit in flag (-f and/or -F)
  - Keep only reads that are 'mapped in proper pair'

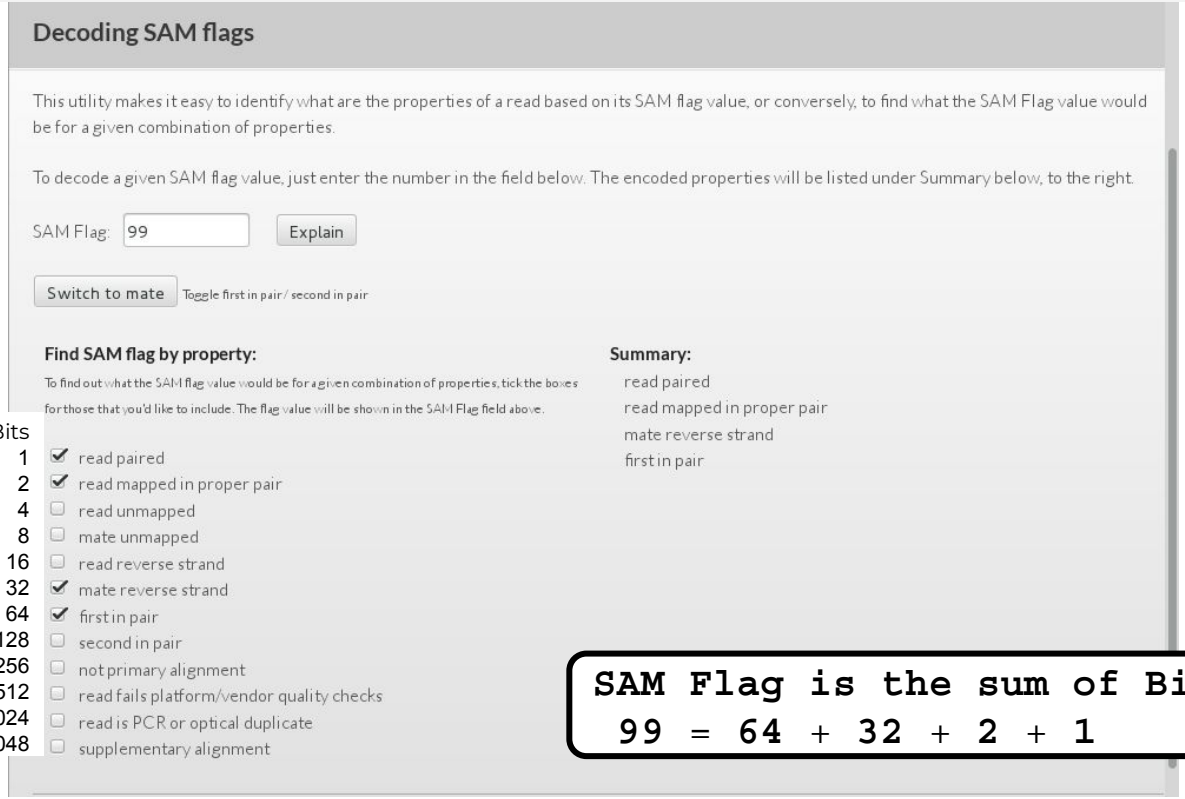
```
samtools view -h -b -f 2 DR34_sorted.bam > dr34_paired_reads.bam
```

  - Keep all except reads that are 'PCR or optical duplicate'

```
samtools view -h -b -F 1024 DR34_sorted.bam > dr34_dedup_reads.bam
```

Sam Flags and Bits

https://broadinstitute.github.io/picard/explain-flags.html



SAM Flag is the sum of Bits

99 = 64 + 32 + 2 + 1

Alignment Statistics

samtools flagstat DR34_sorted.bam

```
1894443 + 0 in total (QC-passed reads + QC-failed reads)
1889642 + 0 primary
4801 + 0 secondary
0 + 0 supplementary
0 + 0 duplicates
0 + 0 primary duplicates
1863297 + 0 mapped (98.36% : N/A)
1858496 + 0 primary mapped (98.35% : N/A)
1889642 + 0 paired in sequencing
944821 + 0 read1
944821 + 0 read2
1852490 + 0 properly paired (98.03% : N/A)
1856814 + 0 with itself and mate mapped
1682 + 0 singletons (0.09% : N/A)
2388 + 0 with mate mapped to a different chr
1342 + 0 with mate mapped to a different chr (mapQ>=5)
```

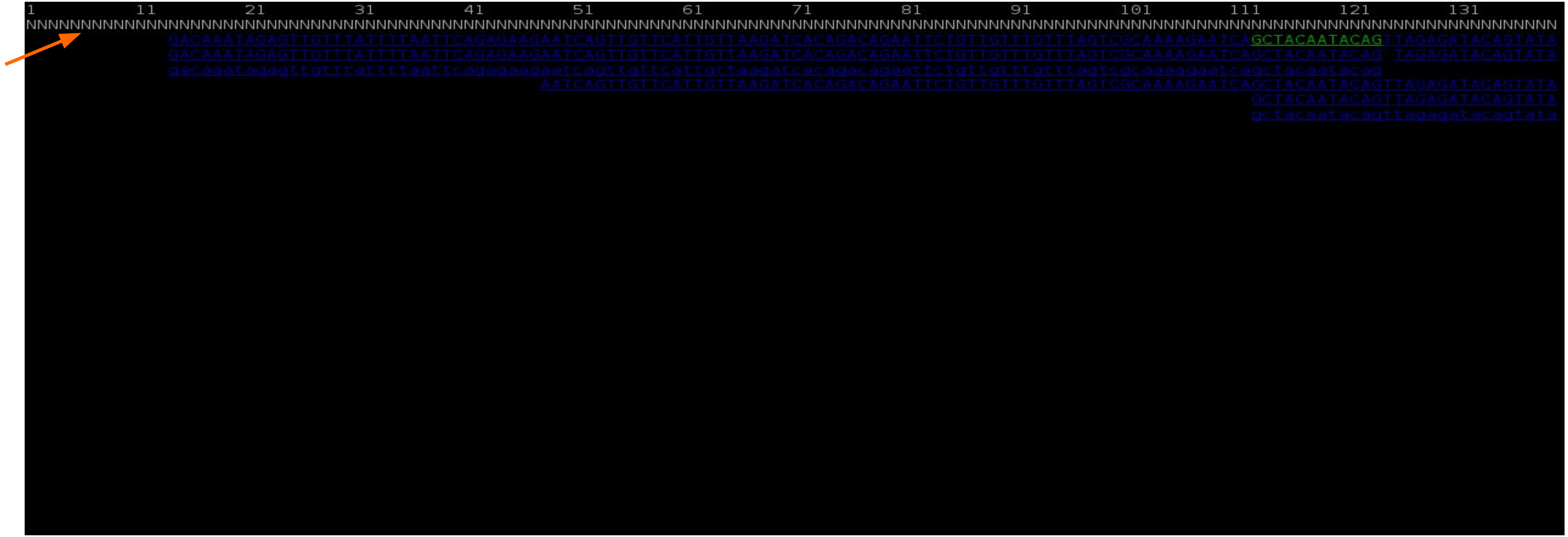
Both reads in the pair are mapped on the same chromosome and in FR or RF orientation

SAMtools Without a Reference Genome

Reference genome represented on top as NNNNNNNNN since no reference genome was

p

```
samtools tview DR34_sorted.bam
```



SAMtools with a Reference Genome

Reference genome sequence displayed on top when reference file is provided

```
samtools tview DR34_sorted.bam c_dubliniensis.fa
```

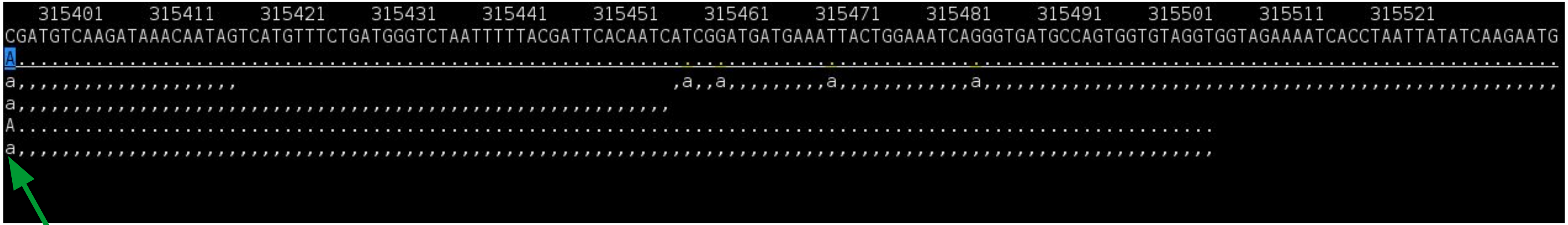
SAMtools with a Reference Genome

Type ? for help menu

```
samtools tview DR34_sorted.bam c_dubliniensis.fa
```

View at a Specific Coordinate

```
samtools tview DR34_sorted.bam c_dubliniensis.fa -p 1:315398
```



homozygous SNP at coordinate 1:315398

. = matches the forward strand
, = matches the reverse strand
A = does not match reference; read on forward strand
a = does not match reference; read on reverse strand
_ = secondary alignment

Sequence Variant Calling

Sequence Variant Calling

- Start with aligning reads to a reference
  - Parabricks aligns using bwa-mem and calls variants using GATK on GPUs
  - GATK does not require QC trimming
  - Mark PCR duplicates with Picard
- Differentiate between sequencing errors and SNPs
  - Calling SNPs may require a min read depth of 10x (higher for indels)
  - Calling variants may require 1/3 of reads to contain SNP
  - Strand bias may result as a consequence of the sequencing chemistry's response to certain DNA sequence motifs but it can be detected computationally
- BLAST reads with SNPs to identify variant calls due to misalignments especially with duplicated genes
- Variant Call Format (vcf) – standard format of variant calls
- Identify multiple-nucleotide polymorphism (MNP)
  - Two SNPs within a single codon

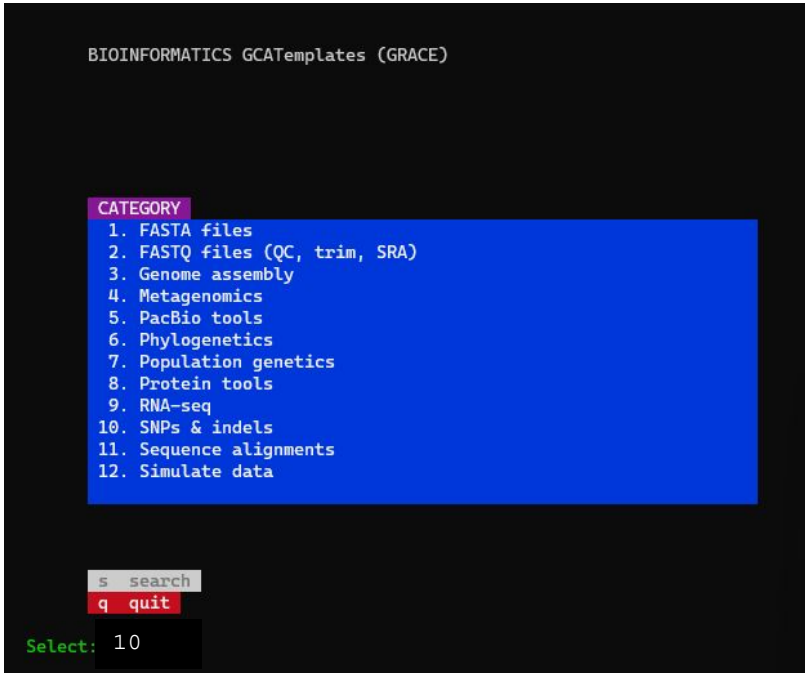
	codon	translation
Reference:	**TTT**	Phe
SNP 1:	**TTA**	Leu
SNP 2:	**TAT**	Tyr
SNP 1 + 2:	**TAA**	**STOP**

Template for Sequence Variant Calling

Genomic Computational Analysis Templates

```
gcatemplates
```

- Ensure that your **bwa** job has completed
- Enter 10, then find the template that contains **freebayes**
  - or use the search function to find **freebayes**
- Final step will save a template job script file to your current working directory
- change the inputs to match your **bam** file
- change the ploidy level to 2
- then submit the job script to the Slurm scheduler



reference_fasta=
/scratch/data/bio/genome_indexes/gmod_genomes/C_dubliniensis_CD36/bwa_0.7.12/C_dubliniensis_CD36_current_chromosomes.fasta

Marking PCR Duplicates

- PCR duplicates are artifacts resulting from a PCR amplification step during NGS library preparations.
- PCR duplicates should be removed/marked as to not bias the frequency of variants or gene expression levels
  - Use picard tools to mark duplicates
  - freebayes will ignore marked duplicates during variant calling

```
module spider picard
```

Variant Calling Tools

1. Parabricks
   a. processing alignments and variant calling on GPUs
2. GATK
   a. No need to QC trim reads–the GATK best practices pipeline will perform the necessary steps, including marking PCR duplicates
   b. You need a set of known variants for your species (dbSNP), or you can bootstrap your population to get variant frequency
   c. Used in conjunction with other tools such as samtools and picard
3. SAMtools and BCFtools
4. freebayes
5. VarScan

Sample vcf File Format

```
##fileformat=VCFv4.0
##fileDate=20110705
##reference=1000GenomesPilot-NCBI37
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=.,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
#CHROM POS     ID        REF  ALT     QUAL FILTER INFO                              FORMAT      Sample1         Sample2
2      4370    rs6057    G    A       29   .      NS=2;DP=13;AF=0.5;DB;H2           GT:GQ:DP:HQ 0|0:48:1:52,51  1|0:48:8:51,51
2      7330    .         T    A       3    q10    NS=5;DP=12;AF=0.017               GT:GQ:DP:HQ 0|0:46:3:58,50  0|1:3:5:65,3
2      110696  rs6055    A    G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB GT:GQ:DP:HQ 1|2:21:6:23,27  2|1:2:0:18,2
2      130237  .         T    .       47   .      NS=2;DP=16;AA=T                   GT:GQ:DP:HQ 0|0:54:7:56,60  0|0:48:4:56,51
2      134567  microsat1 GTCT G,GTACT 50   PASS   NS=2;DP=9;AA=G                   GT:GQ:DP    0/1:35:4        0/2:17:2
```

3 more columns not shown due to width of rows

vcf File Column Descriptions

```
##fileformat=VCFv4.0
##fileDate=20110705
##reference=1000GenomesPilot-NCBI37
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=.,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
#CHROM POS     ID        REF  ALT     QUAL FILTER INFO                              FORMAT      Sample1        Sample2
2      4370    rs6057    G    A       29   .      NS=2;DP=13;AF=0.5;DB;H2           GT:GQ:DP:HQ 0|0:48:1:52,51 1|0:48:8:51,51
2      7330    .         T    A       3    q10    NS=5;DP=12;AF=0.017               GT:GQ:DP:HQ 0|0:46:3:58,50 0|1:3:5:65,3
2      110696  rs6055    A    G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB GT:GQ:DP:HQ 1|2:21:6:23,27 2|1:2:0:18,2
2      130237  .         T    .       47   .      NS=2;DP=16;AA=T                   GT:GQ:DP:HQ 0|0:54:7:56,60 0|0:48:4:56,51
2      134567  microsat1 GTCT G,GTACT 50   PASS   NS=2;DP=9;AA=G                   GT:GQ:DP    0/1:35:4       0/2:17:2
```

variants that are phased are inherited together (paternal)

| indicates phased variants

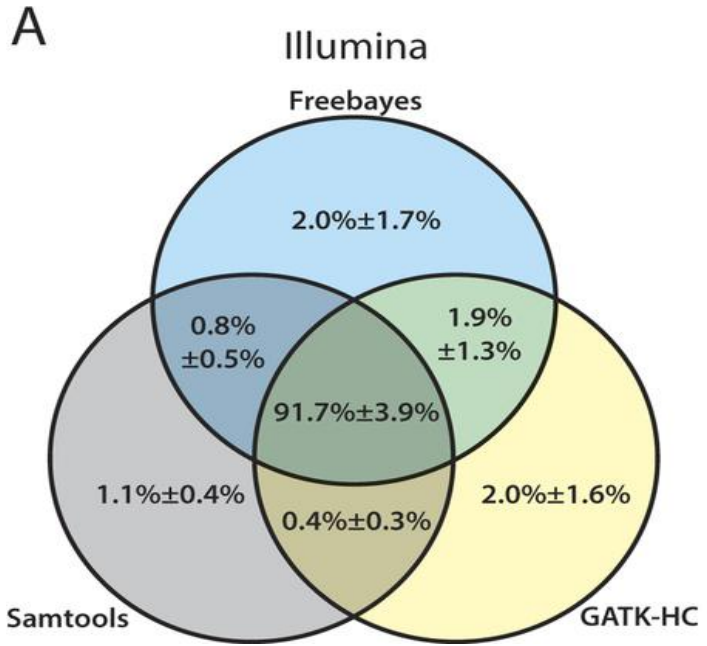
/ indicates non-phased variants

INFO section is for all samples combined

Sample1 haplotypes: **GTGT** and **GTTT**

Sample2 haplotypes: **ATTT** and **GAGT**

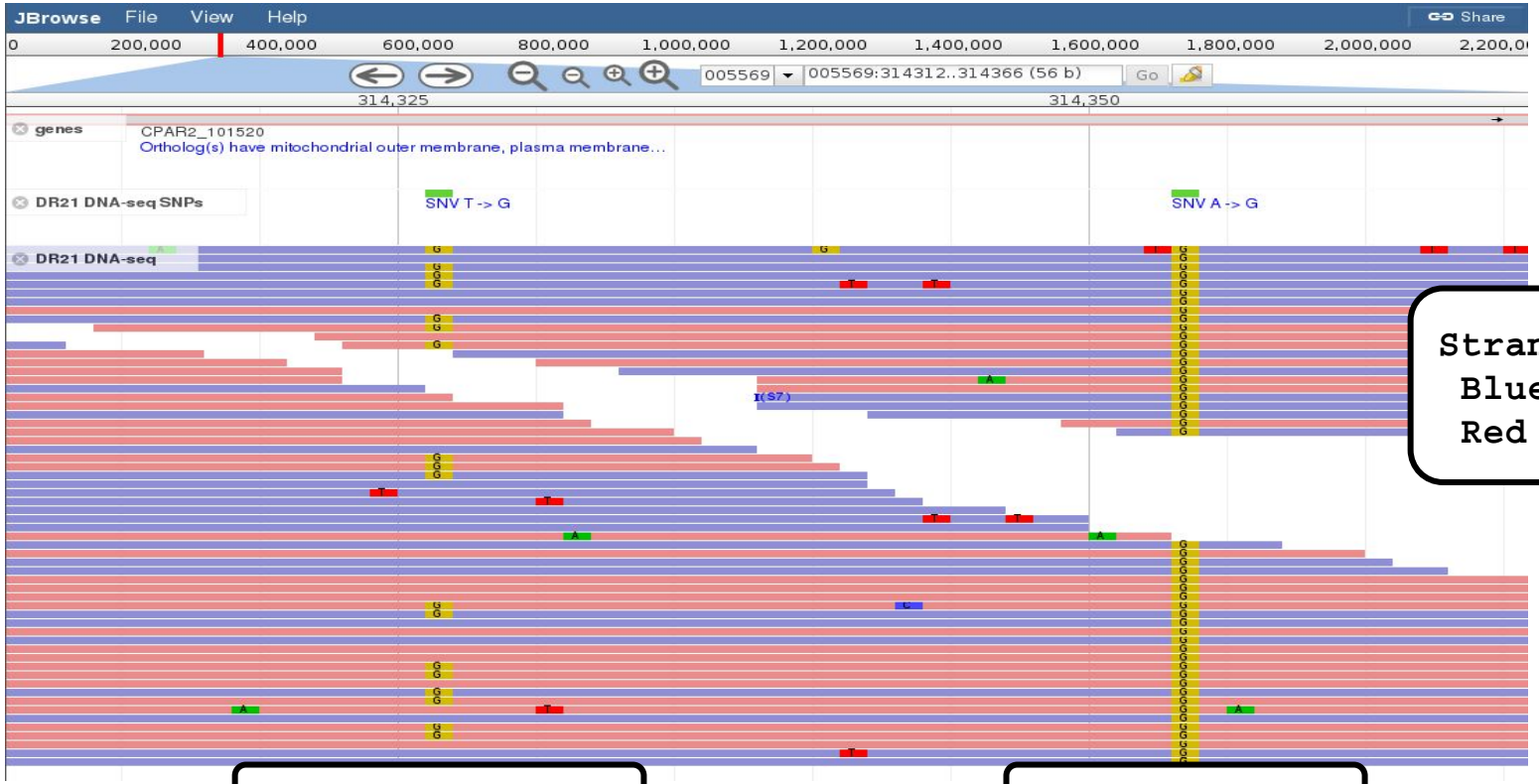
Summarizing Variant Calls from Different Tools



The mean percentage with standard deviation of confidence variant calls with equal to or higher than the quality score threshold of 20 are represented for (A) Illumina data sets

Huang et al 2015 doi:10.1038/srep17875

Viewing SNPs in a Diploid Organism



Annotate Variants

```
module spider snpEff
```

- A file of variant calls in vcf format is needed
- A reference sequence with gene annotations is needed
- snpEff annotates a vcf file
  - There are > 2,500 pre-built databases available and you can build your own if needed
  - Annotates MNP (multiple nucleotide polymorphism)
    - Codon change due to two SNPs: ACA → GGA

```
5       325795  .       AC      GG      23.8901 .
AB=0.428571;ABP=3.32051;AC=1;AF=0.5;AN=2;AO=3;CIGAR=2X;DP=7;DPB=7;DPRA=0;EPP=3.73412;
EPPR=3.0103;GTI=0;LEN=2;MEANALT=1;MQM=33;MQMR=48.5;NS=1;NUMALT=1;ODDS=5.49681;PAIRED=0;
PAIREDR=0.5;PAO=0;PQA=0;PQR=0;PRO=0;QA=114;QR=150;RO=4;RPL=3;RPP=9.52472;RPPR=3.0103;
RPR=0;RUN=1;SAF=2;SAP=3.73412;SAR=1;SRF=2;SRP=3.0103;SRR=2;TYPE=mnp;technology.ILLUMINA=1;
ANN=GG|missense_variant|MODERATE|CD36_51230|CD36_51230|transcript|CAX41505.1|
protein_coding|1/1|c.1657_1658delACinsGG|p.Thr553Gly|1657/1851|1657/1851|553/616||
GT:DP:RO:QR:AO:QA:GL    0/1:7:4:150:3:114:-6.7054,0,-11.1847
```

Consequence of Amino Acid Change

- Assess consequence of amino acid change based on sequence conservation across multiple species using the PROVEAN tool
- Variants with a score equal to or below -2.5 are considered “deleterious”

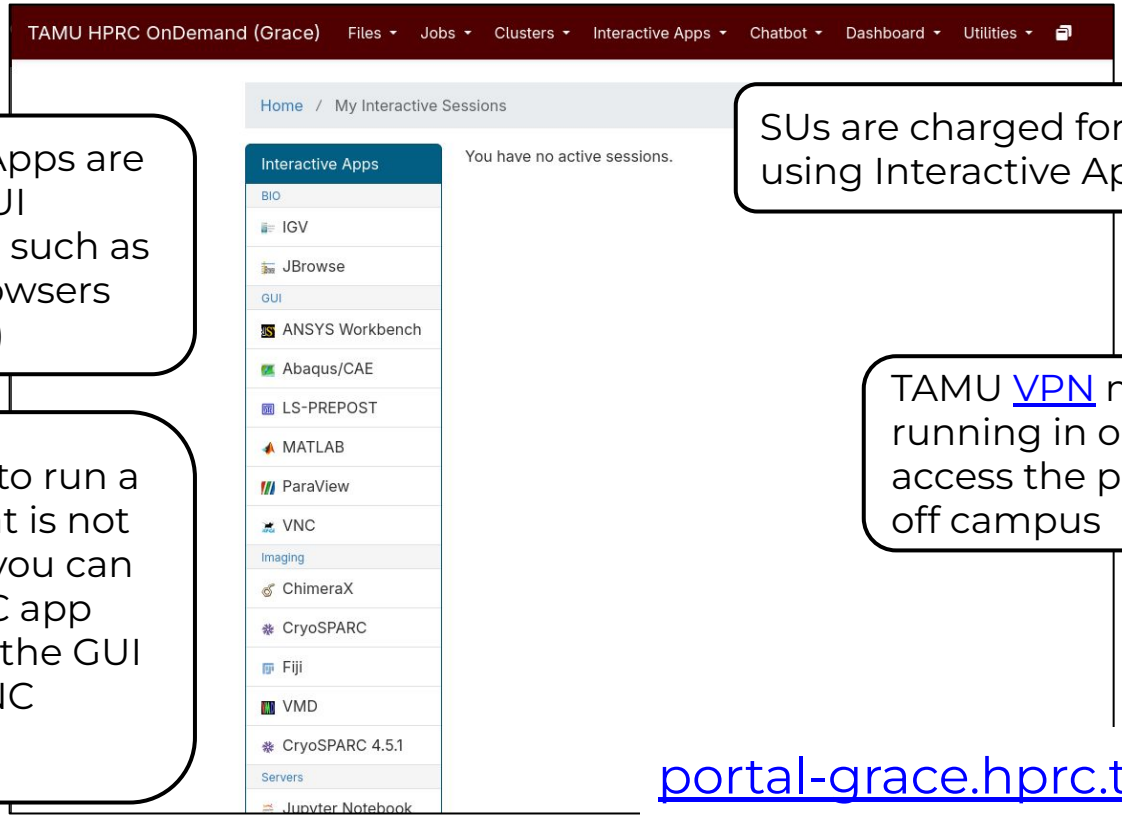
```
module spider PROVEAN
```

```
## PROVEAN v1.1 output ##
# Query sequence file:  CTRG_00013.fa
# Variation file:   CTRG_00013.var
# Protein database: /scratch/datasets/blast/nr
[16:01:13] searching related sequences...
[16:16:36] clustering subject sequences...
# Number of clusters:   30
# Number of supporting sequences used: 245
[16:18:39] computing delta alignment scores...
## PROVEAN scores ##
# VARIATION SCORE
A431S   -0.455
E411K   -3.051
E226Q   -1.564
```

Verify that enough supporting sequences were found

“deleterious”

HPRC Portal Interactive Apps



Interactive Apps are useful for GUI applications such as genome browsers (IGV, Mauve)

If you want to run a GUI app that is not on this list, you can use the VNC app and launch the GUI from the VNC terminal

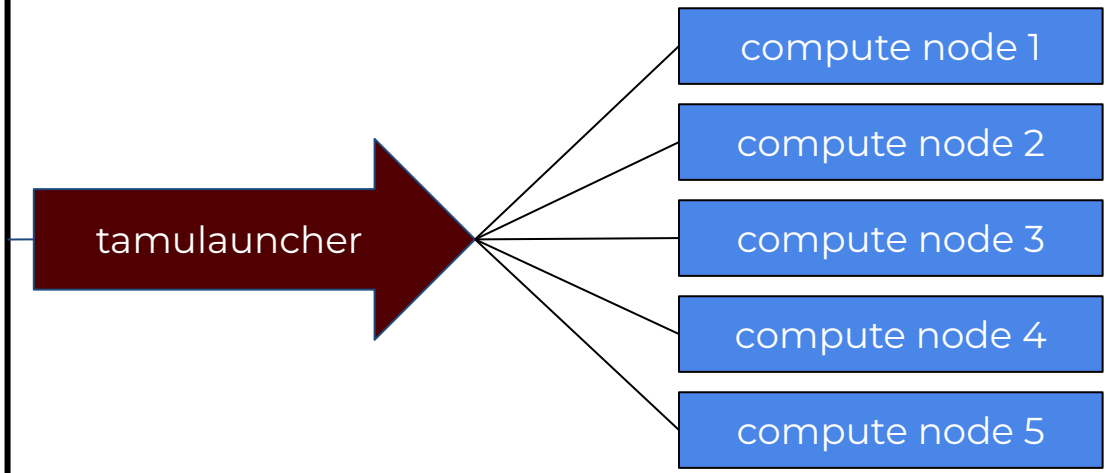
SUs are charged for using Interactive Apps

TAMU VPN must be running in order to access the portal from off campus

portal-grace.hprc.tamu.edu

TAMULauncher

```
blastn -query chunk0000.fa -db 'nt.bacteria'  -task megablast -out chunk0000.fa.out -outfmt 6
blastn -query chunk0001.fa -db 'nt.bacteria'  -task megablast -out chunk0001.fa.out -outfmt 6
blastn -query chunk0002.fa -db 'nt.bacteria'  -task megablast -out chunk0002.fa.out -outfmt 6
blastn -query chunk0003.fa -db 'nt.bacteria'  -task megablast -out chunk0003.fa.out -outfmt 6
blastn -query chunk0004.fa -db 'nt.bacteria'  -task megablast -out chunk0004.fa.out -outfmt 6
blastn -query chunk0005.fa -db 'nt.bacteria'  -task megablast -out chunk0005.fa.out -outfmt 6
blastn -query chunk0006.fa -db 'nt.bacteria'  -task megablast -out chunk0006.fa.out -outfmt 6
blastn -query chunk0007.fa -db 'nt.bacteria'  -task megablast -out chunk0007.fa.out -outfmt 6
blastn -query chunk0008.fa -db 'nt.bacteria'  -task megablast -out chunk0008.fa.out -outfmt 6
blastn -query chunk0009.fa -db 'nt.bacteria'  -task megablast -out chunk0009.fa.out -outfmt 6
blastn -query chunk0010.fa -db 'nt.bacteria'  -task megablast -out chunk0010.fa.out -outfmt 6
blastn -query chunk0011.fa -db 'nt.bacteria'  -task megablast -out chunk0011.fa.out -outfmt 6
blastn -query chunk0012.fa -db 'nt.bacteria'  -task megablast -out chunk0012.fa.out -outfmt 6
blastn -query chunk0013.fa -db 'nt.bacteria'  -task megablast -out chunk0013.fa.out -outfmt 6
blastn -query chunk0014.fa -db 'nt.bacteria'  -task megablast -out chunk0014.fa.out -outfmt 6
blastn -query chunk0015.fa -db 'nt.bacteria'  -task megablast -out chunk0015.fa.out -outfmt 6
blastn -query chunk0016.fa -db 'nt.bacteria'  -task megablast -out chunk0016.fa.out -outfmt 6
blastn -query chunk0017.fa -db 'nt.bacteria'  -task megablast -out chunk0017.fa.out -outfmt 6
blastn -query chunk0018.fa -db 'nt.bacteria'  -task megablast -out chunk0018.fa.out -outfmt 6
blastn -query chunk0019.fa -db 'nt.bacteria'  -task megablast -out chunk0019.fa.out -outfmt 6
blastn -query chunk0020.fa -db 'nt.bacteria'  -task megablast -out chunk0020.fa.out -outfmt 6
blastn -query chunk0021.fa -db 'nt.bacteria'  -task megablast -out chunk0021.fa.out -outfmt 6
blastn -query chunk0022.fa -db 'nt.bacteria'  -task megablast -out chunk0022.fa.out -outfmt 6
blastn -query chunk0023.fa -db 'nt.bacteria'  -task megablast -out chunk0023.fa.out -outfmt 6
```



- A convenient way to run a large number (>100) of single-core jobs
- Available for use from the Unix command line and Galaxy (BLAST+)
- Only need to submit one job instead of many small jobs
- Useful for submitting thousands of BLAST jobs

Biocontainers

What is a Biocontainer?

- A biocontainer is a container that has one or more bioinformatics software packages installed
- A container is a single image file that contains one or more installed software components
- Docker or Singularity is used to build and access a biocontainer
- A container does not contain the entire OS, just components to complement the host OS for running software installed in the container

Pros

- Software is already installed with a specific version together with all software dependencies
- The biocontainer is just one file compared to a software module or conda environment which can have thousands of files

Cons

- The latest software version of a bioinformatics tool may not have a biocontainer available
- Software dependencies within a container may not be the version you want to use

https://biocontainers-edu.readthedocs.io/en/latest/index.html

Using Biocontainers on HPRC Clusters

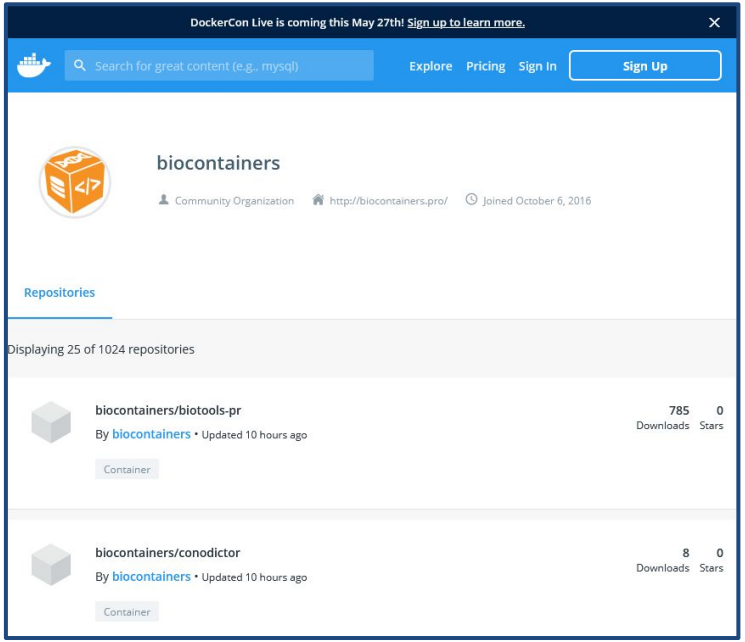
1. Docker is not available on HPRC clusters but Singularity is available
2. Docker images must be converted to a Singularity image
3. Singularity is only available on compute nodes–not login nodes
4. Compute nodes do not have internet access but you can enable a proxy configuration in order to download biocontainers
5. Use VNC portal app or srun to access command line on a compute node in order to convert Docker recipe to a Singularity image
6. Once you have your biocontainer converted to Singularity, you can run it from within a job script

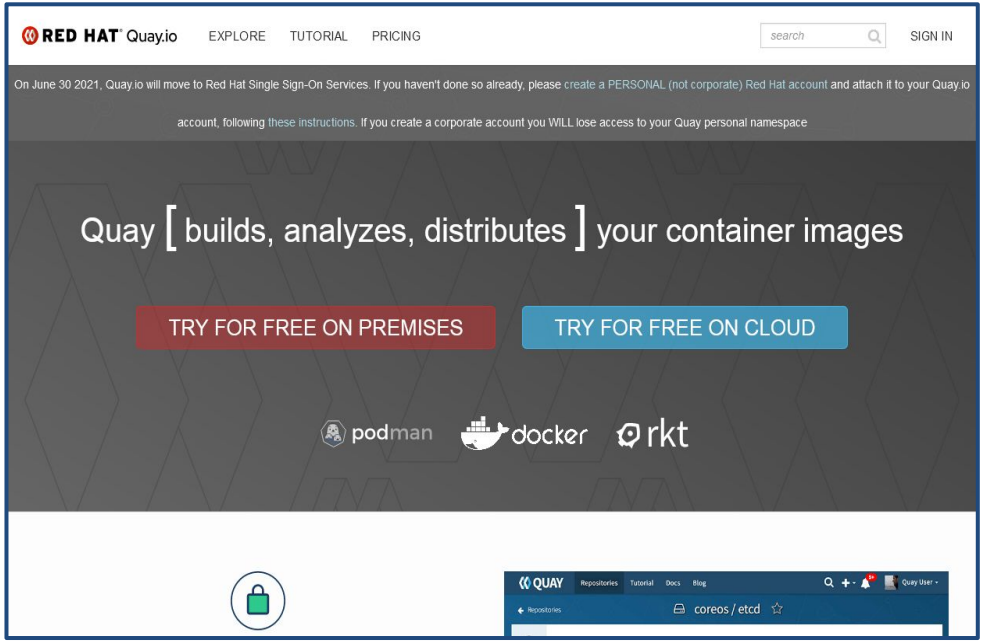
https://hprc.tamu.edu/kb/Software/Bioinformatics/Biocontainers

Finding Biocontainers

https://hub.docker.com/u/biocontainers

https://quay.io





Build the Singularity Biocontainer

1. Connect to a compute node using the HPRC portal VNC app, **srun** interactive job, or run as a job script

```
srun --time=01:00:00 --mem=14G --ntasks=1 --cpus-per-task=2 --pty bash
```

2. Run the following on the compute node command line to enable proxy for internet connection

```
module load WebProxy
```

3. Images can be large during the conversion from Docker to Singularity. Run the following:

```
export SREGISTRY_DATABASE=$TMPDIR
export SINGULARITY_CACHEDIR=$TMPDIR
```

4. The following will create a Singularity image from an available Docker container and name it blast_2.2.31.sif (takes about 4 minutes to download with 2 CPUs and convert the blast container)

```
singularity pull docker://biocontainers/blast:2.2.31
```

https://hprc.tamu.edu/kb/Software/Singularity/#singularity-pull-examples

Singularity Build Time

blast_2.2.31.sif	#SBATCH cpus	#SBATCH memory	time to build .sif file
build 1	2 cores	4 GB	3 min 45 sec
build 2	28 cores	54 GB	2 min

```
#!/bin/bash
#SBATCH --job-name=singularity   # job name
#SBATCH --time=01:00:00          # max job run time dd-hh:mm:ss
#SBATCH --ntasks-per-node=1      # tasks (commands) per compute node
#SBATCH --cpus-per-task=2        # CPUs (threads) per command
#SBATCH --mem=4G                 # total memory per node
#SBATCH --output=stdout.%j       # save stdout to file (%j is jobid)
#SBATCH --error=stderr.%j        # save stderr to file (%j is jobid)

module purge
module load WebProxy
export SREGISTRY_DATABASE=$TMPDIR
export SINGULARITY_CACHEDIR=$TMPDIR
singularity pull docker://biocontainers/blast:2.2.31
```

Test the Singularity Biocontainer

Enter the following in the same directory as the .sif file to see the help info for the blastp command

```
singularity exec blast_2.2.31.sif  blastp -h
```

Or run the following to launch a prompt which you can use to explore the commands available in the container. This is not used to update the image with new or additional software.

```
singularity run blast_2.2.31.sif
```

type **exit** to exit the **Singularity>** prompt

Example using blastp in a singularity command

These are just examples not available to run.

Example command to run on the compute node command line after either using **`srun`** or the VNC portal app to launch a job or by putting the command into a job script

singularity exec blast_2.2.31.sif blastp -query seqs.fa -db hg19 -out blastout.csv -num_threads 28

Or if your .sif file is in a different directory than the working directory

singularity exec /sw/hprc/sw/bio/containers/blast_2.2.31.sif blastp -query seqs.fa -db hg19 -out blastout.csv -num_threads 28

https://hprc.tamu.edu/kb/Software/Singularity/#interact-with-container

HIGH PERFORMANCE RESEARCH COMPUTING
TEXAS A&M UNIVERSITY

https://hprc.tamu.edu

HPRC Helpdesk:

help@hprc.tamu.edu
Phone: 979-845-0219

Take our short course survey!

https://u.tamu.edu/hprc_shortcourse_survey

Help us help you. Please include details in your request for support, such as, **Cluster** (ACES, FASTER, Grace, Launch), NetID (UserID), Job information (**JobID**(s), Location of your jobfile, input/output files, Application, Module(s) loaded, Error messages, etc), and Steps you have taken, so we can reproduce the problem.