VARIANT CALLING

A commonly used pipeline and how to use parallelization to make the most of it

Merete Molton Worren

Abdulrahman Azab

Variant calling

- The goal is to find differences between a reference and your data
  - Disease/normal
  - Different strains of bacteria
- Differences can include SNVs, Indels, Copy number variations
- This lecture focuses on SNVs

SNV calling

- Several ways of doing things
- Different experiments may require different analysis steps
- In todays lecture you will learn a very basic SNV calling pipeline. The focus will be on parallellisation.

A couple of resources

- GATK Broad Institute
  https://www.broadinstitute.org/gatk/index.php

- Teaching material from INF9121 fall 2014
  https://wiki.uio.no/projects/clsi/index.php/INF-BIOX121_H14_course_material#Variant_calling

The data

- Mouse, exome sequencing, Illumina
- Sequence Read Archive (SRA)
- One with a mendelian disease
- One reference strain
- Data was part of a large experiment

http://www.ncbi.nlm.nih.gov/pubmed/25917818

Workflow

- Quality control with FastQC
- Quality filtering with Prinseq
- Mapping/Alignment with BWA-MEM
- Preprocessing SAM file
  - Mark duplicates, Realign indels, Base calibration
- Variant calling
  - Call and filter

Quality control
FastQC

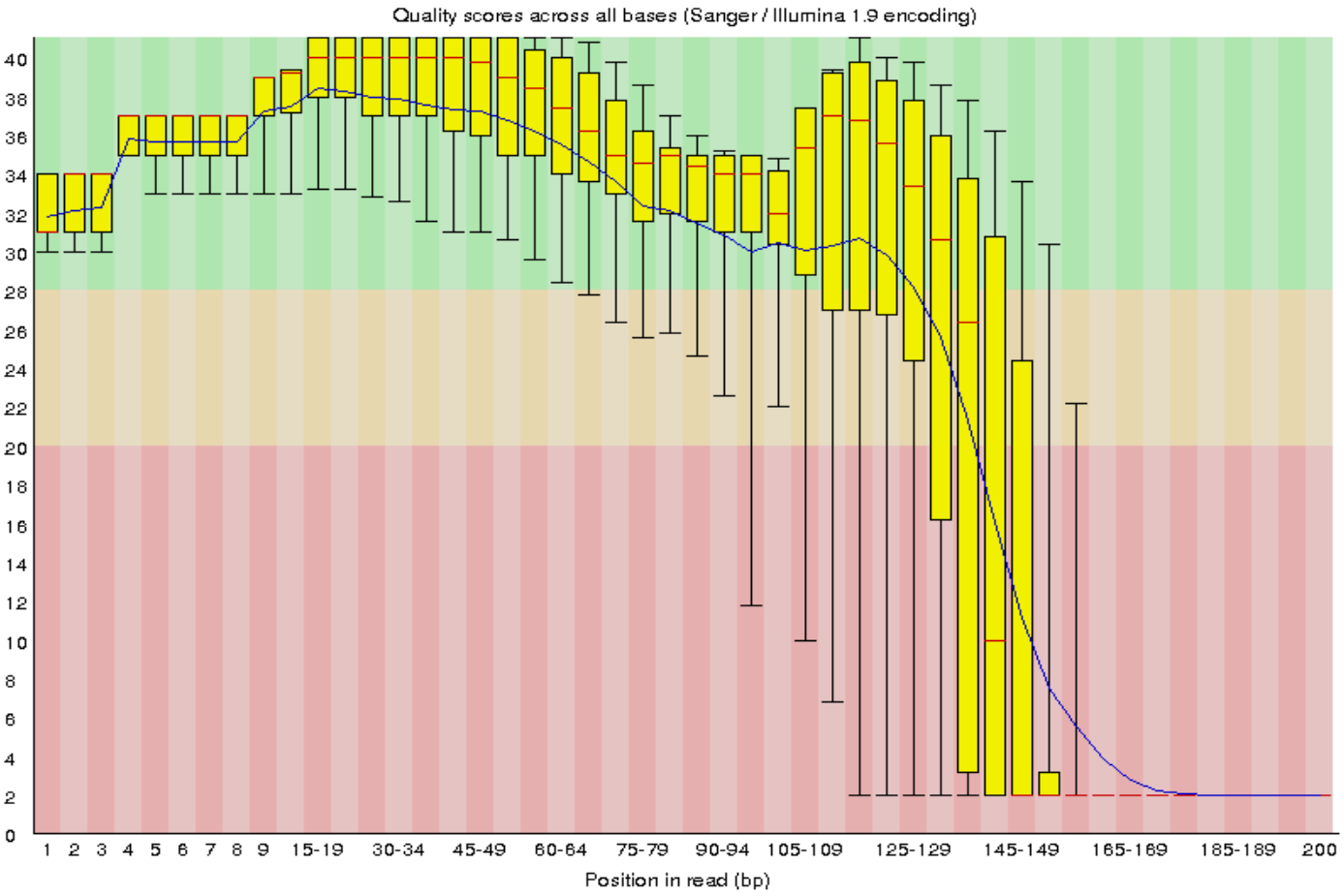
- Fastq files straight from the sequencer often have bad quality and/or contamination
- We need to know, and we need to fix it before doing variant calling
- Running FastQC will give us an overview of what our reads look like

How does FastQC work?

- Series of analysis modules
- For all these modules, you get results from all the reads.
- A link to the FastQC documentation can be found at their website

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

FastQC per base quality plot



Parallelizing FastQC

- No option for parallelizing with only one file
- If you work with several files from different samples, you should run them in parallel to save time
- From the FastQC manual:

-t --threads Specifies the number of files which can be processed simultaneously. Each thread will be allocated 250MB of memory so you shouldn't run more threads than your available memory will cope with, and not more than 6 threads on a 32 bit machine

Time for practical exercises!

All the exercises under

1 Get ready and 2 Quality control

Discussion

- Why does it not help to add more threads than the number of files?
- Why is the real time the only one that changes when we add more threads?
- Can you think of a way to parallelize FastQC for only one file?

Quality filtering
Prinseq lite

- The bad quality reads will have to be filtered before starting variant calling
- Since bad quality is common at the end of reads, for a lot of reads removing the end is sufficient
- Prinseq is one of several programs that can be used to do this.

What is Prinseq lite?

- Standalone Perl script
- Does not require any non-core perl modules
- A number of options for filtering, trimming and statistics
- Website: http://prinseq.sourceforge.net/manual.html
- The manual includes a description of all the options

Parallelizing Prinseq

- No option for running in parallel
- Still possible to run the same command several times in parallel for different files

Time for practical exercises!

All exercises under

3 Quality filtering

Discussion

- Why did we need the line number in this exercise to be dividable by 4
- What's your opinion on the overhead in this parallelization exercise?
- Are there unused possibilities for parallelizing Prinseq lite?
- Would it be useful?

Alignment/Mapping

- Mapping – where in the genome does the read come from?
- Alignment – what is the exact placement of each base in the read?
- A number of different mappers/aligners are available
- We will use BWA-MEM

Burrows -Wheeler Alignment Tool

- Indexing based on Burrows-Wheeler transform
- Several different alignment algorithms
- We will use the mem algorithm (maximal exact matches)

http://bio-bwa.sourceforge.net/bwa.shtml

How does BWA-MEM work?

- Indexing (Burrows-Wheeler)
- Seeding (MEM)
- Reseeding
- Chaining
- Chain filtering
- Seed extension (affine-gap Smith-Waterman)

Parallellizing BWA-MEM

- Index – no command line option for parallelizing
- MEM – option for threads

Time for practical exercises!

All exercises under

4 Alignment

Discussion

- Is BWA-MEM a mapper or an aligner?
- How do you think the BWA-MEM algorithm is parallelized?

Preprocessing

- The Sam files we get after aligning may not be of optimal quality
- Duplicate reads
- Misalignments caused by indels
- Suboptimal quality scores
- We want to ensure the best possible variant calling by trying to remove these sources of error

Preprocessing

- GATK and Picard have several tools to aid us
- Requires some work to get it done
  - 3 tasks
  - 7 steps

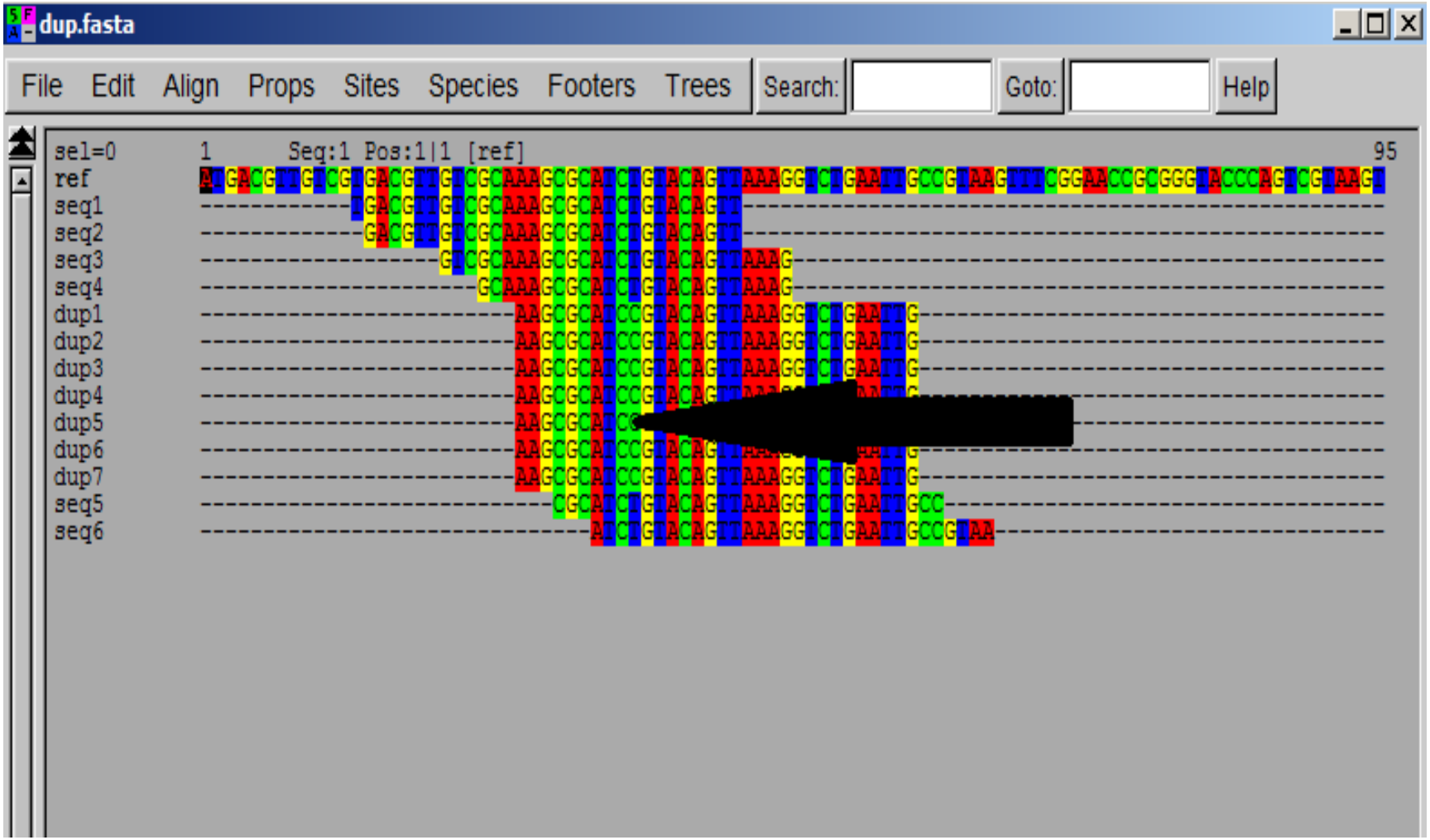
Preprocessing

Mark duplicates

- Duplicates stem from the exact same DNA sequence
- Non-independent reads violate assumptions made when calling variants
- We use Picard to mark duplicates
- Picard is a (java) toolkit with several tools for manipulating HTS data

https://broadinstitute.github.io/picard/command-line-overview.html

Duplication



Preprocessing

What does Mark duplicates do?

- Locates duplicates and tags them by default. It is also possible to have them removed.
- Uses start coordinates to determine if reads are duplicates, for paired-end reads info from Cigar is also used
- The input file has to be coordinate sorted
- No option for parallelizing

Parallelizing Mark duplicates

- While there is no threads option for Picard, fortunately it is still possible to run several instances of the command in parallel
- We do this for each file

Preprocessing
Realigning indels

- The reason for doing this step is to avoid calling SNVs due to misalignment near an indel

- We use GATK RealignerTargetCreator and GATK IndelRealigner for this

Preprocessing
Realigning indels

- We use GATK RealignerTargetCreator and GATK IndelRealigner for this
- GATK is a (java) program including a lot of methods. Options that can be used for all of them includes
  - --num-threads and
  - --num_cpu_threads_per_data_threads

Realigning indels

- Realigns reads locally
- Minimizes the mismatching bases for all the reads taken together
- RealignerTargetCreator identifies small areas that may contain misaligned reads
- IndelRealigner will realign the reads in these areas

Preprocessing

Base Recalibration

- The base quality scores are used when calling variants
- The scores (assigned by the sequencer) are subject to systematic errors
- Base recalibration will model these errors and adjust the scores for more accuracy
- We use GATK BaseRecalibrator and GATK PrintReads for this

Preprocessing
Base Recalibration

- BaseRecalibrator : Does the modeling and produces a report
- PrintReads : Will take in the report from BaseRecalibrator (with --BQSR argument) and recalibrates the bases

Time for practical exercises!

All exercises under

5 Preprocessing for GATK

Discussion

- If you were to implement a new MarkDuplicates with an option to parallelize, what would you do?

- How would you parallelize a sorting algorithm?

Variant calling

- The data is now ready, so we can do the main task, variant calling
- We do this with the GATK HaplotypeCaller
- It can find both SNPs and indels
- When a region is showing signs of variation, the HaplotypeCaller re-assembles the reads in that region
- More accurate in difficult regions

Parallelizing the HaplotypeCaller

- The HaplotypeCaller can be run in multi-threaded mode using the options

  --num_threads

  --num_cpu_threads_per_data_threads (-nct)

Variant filtering

- Variant quality score recalibration (VQSR)
  - Uses machine learning
  - Requires know variants
- Hard filtering
  - General filtering based on chosen thresholds
- We will use hard filtering, with GATK VariantFiltration

Comparing subject to control

- Overall goal is to find SNPs that are present in the subject with the mendelian disease, but not there in the control
- We will make our own script to do this
- The script should use parallelization

Time for practical exercises!

All exercises under

6 Variant calling and

7 Compare subject to control