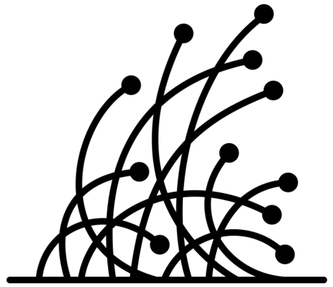


Workshop 1: *Candida auris*

Mapping to reference and calling variants

Dr Johanna Rhodes

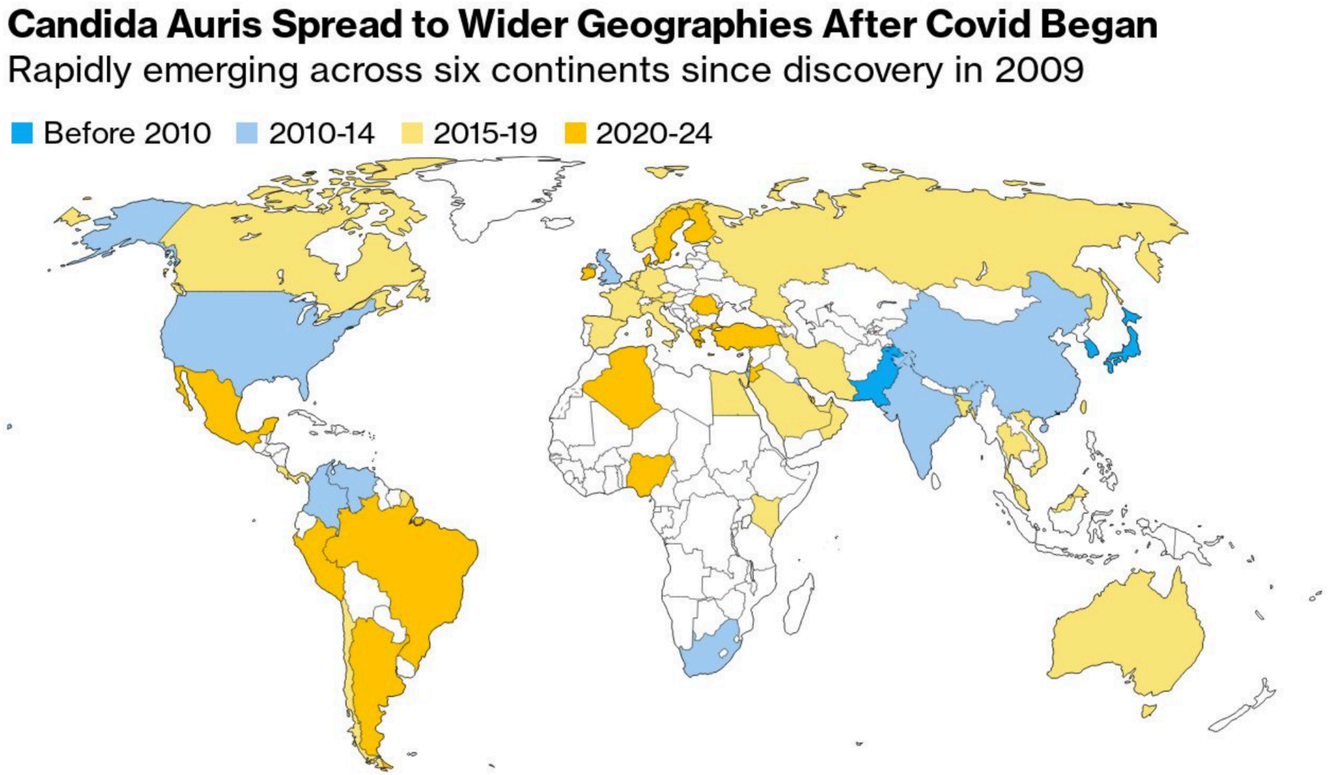
Learning outcomes

- **In practice**
  - Command line arguments
  - Troubleshooting errors

Background

***Candida auris* - WHO critical priority fungal pathogen**

- First described in 2009 after isolation from ear canal of patient
- Apparent simultaneous emergence of four distinct clades
  - Two further clades identified
  - Distinct antifungal resistance patterns
- Increasingly pan-drug resistant
- Invasive infections mortality 50-80%



Bloomberg 5th Nov 2024

Data analysis

Short read (Illumina) data

- Quality check data
- Prepare your reference - B8441_v3 (GCA_002759435.3)
  - Acquire from RefSeq
- Map sequence reads to reference genome
- Call variants and filter to obtain high-confidence SNPs

Quality check data

FastQC

- https://www.bioinformatics.babraham.ac.uk/projects/fastqc/
- Or use MultiQC on Galaxy (https://usegalaxy.org/)
  - Need to make an account

Prepare reference genome
Indexing

- After downloading reference genome FASTA, index for BWA, Samtools and GATK
- Index using BWA
  - `bwa index reference.fa`
  - Type ‘ls -lh’ to see the files created
- Index using Samtools
  - `samtools faidx reference.fa`
  - Creates a file with extension ‘.fai’, which contains one record per line for each of the costings (contiguous DNA segments) in the fasta file
- Index for GATK using Picard
  - Creates a ‘dictionary’ (.dict file) using Picard in order to call SNPs using GATK. This .dict file describes the contents of your reference fasta file
  - `picard CreateSequenceDictionary -R reference.fa -O reference.dict`

Mapping reads to reference genome

Like a jigsaw puzzle

```
bwa mem –M reference.fa R1.fastq.gz R2.fastq.gz > isolate_name.sam
```

Basic alignment!

- convert using samtools: `samtools view -bS isolate_name.sam > isolate_name.bam`
- Improve the alignment by co-ordinate sorting: `samtools sort isolate_name.bam -o isolate_name.sorted.bam`
- Remember to index the bam file…
  - `samtools index isolate_name.sorted.bam`

Improving the alignment
Fix the read groups

- We use ‘AddOrReplaceReadGroups’ from Picard to ‘fix’ the BAM file.
    - Downstream analyses (e.g. GATK) require the BAM file to contain read group information:
        - Read group identifier - this is the sequencer flow cell, lane number and name.
        - Read group platform - e.g. Illumina
        - Read group library - this is needed for marking duplicates to determine which read groups contain molecular duplicates
        - Read group platform unit - this is the run ‘barcode’ or the adaptor sequence
- The command may look a bit like this:

```
picard AddOrReplaceReadGroups -I isolate_name.sorted.bam -O isolate_name.fixed.sorted.bam --SORT_ORDER coordinate
--RGID K00166 --RGLB dnaseq --RGPL illumina --RGSM ’WGS’ --CREATE_INDEX TRUE --RGPU unknown --VALIDATION_STRINGENCY
SILENT
```

Improving the alignment

Marking duplicates

- Use Picard to mark any duplicated reads due to sequencing errors to prevent them being included in variant calling

```
picard MarkDuplicates -I isolate_name.fixed.sorted.bam -O
isolate_name.sorted.marked.bam --CREATE_INDEX TRUE --METRICS_FILE
picard_info.txt --REMOVE_DUPLICATES false --ASSUME_SORTED true --
VALIDATION_STRINGENCY SILENT
```

- This is still a rough ‘global’ alignment
- The rest of the pipeline is entirely GATK for improving the alignment further and variant calling

Improving the alignment

Base Quality Score Recalibration - commands

- First we need to provide a rough guide of variants (SNPs and indels) for BQSR. HaplotypeCaller (GATK) calls variants, and by default assumes a diploid organism so we need to specify ploidy is 1 (haploid)

```
gatk HaplotypeCaller –R reference.fa –I isolate_name.sorted.marked.bam –ploidy 1 –O
isolate_name.raw_variants.vcf
```

- This file contains both SNPs and indels, and can be used for BQSR:

```
gatk BaseRecalibrator –R reference.fa –I isolate_name.sorted.marked.bam –-known-sites
isolate_name.raw_variants.vcf –O isolate_name.recal_data.table
gatk ApplyBQSR –R reference.fa –I isolate_name.sorted.marked.bam –-bqsr-recal-file
isolate_name.recal_data.table –O isolate_name.recal_reads.bam
```

- Do this again (be careful with file names!) to acquire final BAM file with your final alignment

Quality check the alignment

- Coverage:

```
picard CollectWgsMetrics -R reference.fa -I isolate_name.post_recal_reads.bam
-O isolate_name.metrics.txt
```

```
samtools depth isolate_name.post_recal_reads.bam > isolate_name.coverage.txt
```

- Mapping statistics:

```
samtools flagstat isolate_name.post_recal_reads.bam
```

Calling high-quality variants

Commands using GATK

```
gatk HaplotypeCaller –R reference.fa –I isolate_name.post_recal_reads.bam –O
isolate_name.raw_variants_recal.vcf -ERC GVCF –-pcr-indel-model NONE –ploidy 1
–stand-call-conf 30 –mbq 20 –A QualByDepth

gatk GenotypeGVCFs –R reference.fa -V isolate_name.raw_variants_recal.vcf -O
isolate_name.genotyped_variants_recal.vcf
```

Filtering variants

Commands in GATK

- First, we separate the vcf into SNPs and indels (you can merge them back together after filtering if preferred)

```
gatk SelectVariants –R reference.fa –V
isolate_name.genotyped_variants_recal.vcf –O isolate_name.raw_snps_recal.vcf --
select-type-to-include SNP -select ‘vc.getGenotype(“WGS”).getAD().1*1.0 /
vc.getGenotype(“WGS”).getDP() > 0.90’
```

```
gatk SelectVariants –R reference.fa –V isolate_name.raw_variants_recal.vcf –O
isolate_name.raw_indels_recal.vcf --select-type-to-include INDEL
```

Filtering variants

Commands in GATK

- To gain a list of high confidence SNPs we filter on mapping quality, depth of coverage, Fisher strand bias and balance of alleles.
- Any SNP that fulfils any one of these criteria is labelled as ‘LowConf’. It is not removed
- We also include an additional filter on genotype quality
  - Note that we sort of did a filtering step earlier when selecting SNPs, where we filtered out SNPs that weren’t present in at least 90% of mapped reads.

```
gatk VariantFiltration -R reference.fa -V isolate_name.raw_snps_recal.vcf -O
isolate_name.filtered_snps_final.vcf -filter "QD < 2.0" --filter-name "LowConf" -filter
"FS > 60.0" --filter-name "LowConf" -filter "MQ < 40.0" --filter-name "LowConf" -filter
"MQRankSum < -12.5" --filter-name "LowConf" -filter "ReadPosRankSum < -8.0" --filter-
name "LowConf" -filter "SOR > 4.0" --filter-name "LowConf" -filter "DP < 5" --filter-
name "LowConf" -G-filter "GQ < 50" -G-filter-name "FILTER_GQ-50"
```

Filtering variants

Additional quality check

- Using the two files, ‘isolate_name.genotyped_variants_recal.vcf’ and ‘isolate_name.filtered_snps_final.vcf’, you want to count the number of lines in each, excluding the header
- `grep -v “#” isolate_name.genotyped_variants_recal.vcf | wc -l`
- `grep -v “#” isolate_name.filtered_snps_final.vcf | grep -v “LowConf” | grep -v “FILTER_GQ-50” | wc -l`
- The difference between these two numbers will give you an idea of quality
  - High % filtered out indicates something weird is going on - cross-check with coverage and mapping statistics
    - Hybrid/diploid
    - Not the same species as reference
    - Poor quality sequencing
  - Usually range can be 1-10% filtered out