A short introduction to bioinformatics and variant calling

Adam Orr @AdamJOrr

5/17/19

Outline

- Sequencing methods and their advantages
- FASTQ Format
- Read Mapping / Alignment
- BAM Format
- Variant Calling
- VCF Format
- Annotation Formats
- Assembly?

Sequencing

The Driving Principle

We cannot sequence an entire genome or even an entire chromosome accurately in 1 attempt. To sequence we must shred the DNA and sequence the fragments.

There are 3 competing sequencing platform providers: Illumina, PacBio, and Oxford Nanopore
Each platform is based on a different technology that changes how the DNA samples are processed and how sequence information is extracted from them.

Illumina

- Cheapest and highest throughput
- Best single-base accuracy
- Reads are short (150 bp)

PacBio SMRT sequencing

- More expensive but longer reads (10KB)
- Decently accurate
- Use for long insertions/deletions

Oxford Nanopore: MinION

- Machine is much more affordable, throughput is lower.
- Lowest accuracy, but improving quickly
- Longest reads (100KB+)
- Use for long insertions/deletions

Sequencing Strategies

The less you sequence the less it costs

- Sequencing costs
- Data storage costs
- Computation costs

- Whole Genome Sequencing
- Whole Exome Sequencing
- Amplicon Sequencing

Generic Sequencing Experiment Workflow

- GrCH38 is the current reference

FASTQ Format

- 4 line format
- Most tools transparently work with `.gz` compressed files.
- Quality scores are phred-scaled and encoded as the ASCII value -33. (! = 0, J = 41)
- Best resource: wikipedia article on FASTQ format

```
1 @HJCMTCCXX160113:5:1101:1631:47668/1
2 GCCCAGCACAGAGGTGCCCAGGGTGCAGGCTGGCACTGGC
3 +
4 AAFAFF<<<7FFFKFFFFFAAFKAF7FKKKKK(,7A,<KA
```

FASTQC

FASTQC is a quality control program that:

- Checks for adaptor readthrough
- Checks there aren't any very obvious biases in the sequenced data
- Usually your data will be fine
- Ask your sequencing provider what QC they do
- Subsamples reads so it's quite fast

Read Mapping

- `bwa mem` for short reads (GPL3)
- `minimap` for long reads (MIT)
- Fairly fast

Algorithm

The reference genome is indexed first. The index is used to find plausible locations on the reference for each read, the matches are extended until the most likely location is found.

```
bwa index ref.fa
bwa mem reads.1.fq reads.2.fq > aligned.sam
```

SAM format

- Tabular format with header
- Binary compressed version called "**B**AM".
- Highly compressed version called "**CR**AM", keep reference handy.
- SAMtools is used to view/manipulate BAMs (MIT)
- Always sort your bams. MarkDuplicates is a good idea.
  - `samtools sort -Ob aln.sam > aln.sorted.bam`
  - `samtools markdup aln.sorted.bam aln.marked.bam`
- Resource: The SAM format specification

```
@HD VN:1.6 SO:coordinate
@SQ SN:ref LN:45
r001    99 ref  7 30 8M2I4M1D3M = 37   39 TTAGATAAAGGATACTG *
r002     0 ref  9 30 3S6M1P1I4M *  0    0 AAAAGATAAGGATA     *
r003     0 ref  9 30 5S6M       *  0    0 GCCTAAGCTAA        * SA:Z:ref,29,-,
r004     0 ref 16 30 6M14N5M    *  0    0 ATAGCTTCAGC        *
r003  2064 ref 29 17 6H5M       *  0    0 TAGGC              * SA:Z:ref,9,+,5
r001   147 ref 37 30 9M         =  7  -39 CAGCGGCAT          * NM:i:1
```

Variant Calling

Mapping is essentially a solved problem; variant calling and subsequent filtering is where things get interesting and start to take a long time.

State of the art: local reassembly in regions there may be variation and construct haplotypes.

Short variants (substitutions, indels <50bp):

- GATK ($$$)
- FreeBayes (MIT)
- Google DeepVariant (BSD)

Long insertions/deletions (>50bp, Structural Variants):

- Manta (Illumina; GPL3)
- pbsv (PacBio; GPL3)

VCF Format: Header

```
##fileformat=VCFv4.3
##fileDate=20090805
##source=myImputationProgramV3.1
##reference=file:///seq/references/1000GenomesPilot-NCBI36.fasta
##contig=<ID=20,length=62435964,assembly=B36,md5=
    f126cdf8a6e0c7f379d618ff66beb2da,species="Homo sapiens",taxonomy=x>
##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=A,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=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##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      NA00001        NA00002
          NA00003
```

VCF Format: Body

```
#CHROM POS ID        REF    ALT     QUAL FILTER INFO
      FORMAT   NA00001        NA00002        NA00003
20     14370   rs6054257 G      A       29   PASS  NS=3;DP=14;AF=0.5;DB;H2     GT:GQ:
      DP:HQ 0|0:48:1:51,51 1|0:48:8:51,51 1/1:43:5:.,.
20     17330   .         T      A       3    q10   NS=3;DP=11;AF=0.017         GT:GQ:
      DP:HQ 0|0:49:3:58,50 0|1:3:5:65,3 0/0:41:3
20     1110696 rs6040355 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 2/2:35:4
20     1230237 .         T      .       47   PASS  NS=3;DP=13;AA=T              GT:GQ:
      DP:HQ 0|0:54:7:56,60 0|0:48:4:51,51 0/0:61:2
20     1234567 microsat1 GTC    G,GTCT  50   PASS  NS=3;DP=9;AA=G               GT:GQ:
      DP 0/1:35:4       0/2:17:2       1/1:40:3
```

Variant Filtering

Tools tend to output many false-positives, so filtering is very important.

The variant caller you use will calculate many quality statistics, so exact filters will depend on the method you use and your use case.

- BCFTools (MIT or GPL3)
  - `bcftools filter -i 'DP > 10' variants.vcf > variants.filtered.vcf`
- Picard (MIT)
  - `java --jar picard.jar FilterVcf INPUT=input.vcf OUTPUT=out.vcf`
- VCFTools (GPL3)
  - `vcftools --vcf input.vcf --recode --recode-INFO-all`

Annotation Formats

GFF Format
BED Format