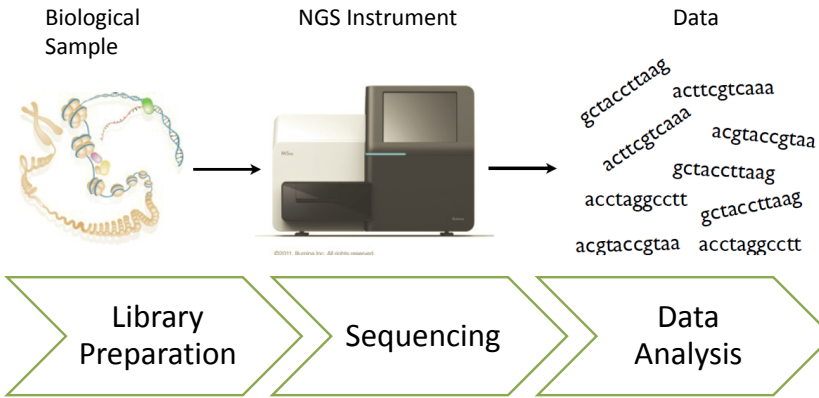
Introduction to Bioinformatics

Alexandros C. Dimopoulos
alexdem@di.uoa.gr

Master of Science
"Data Science and Information Technologies"
Department of Informatics and Telecommunications
National and Kapodistrian University of Athens

2024-25

NGS Overview



NGS “Hardware” I

Roche GS-FLX

Life Technologies SOLID

Life Technologies Ion Proton

Illumina HiSeq

Life Technologies Ion Torrent

Illumina MiSeq

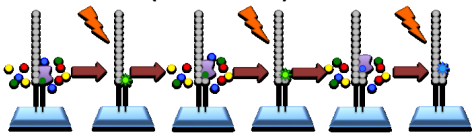
NGS “Hardware” II

Pacbio Sequel

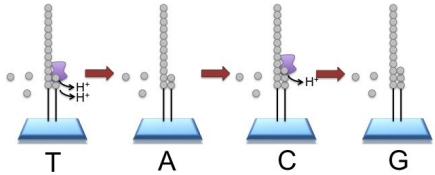
Oxford Nanopore

NGS Technologies I

- fluorescence-based (Illumina)



- hydrogen ion /pH-mediated based (Life)

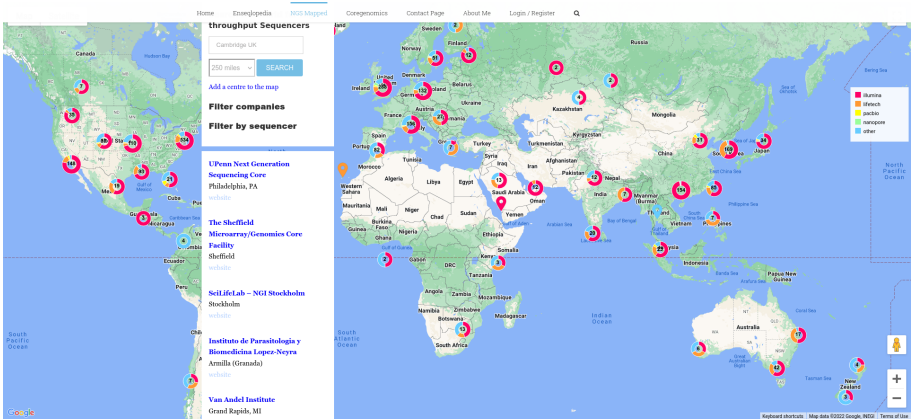


NGS Technologies II

	Run Time	Read Length	Quality	Total nucleotides sequenced	Cost /MB
454 Pyrosequencing	24h	700 bp	Q20-Q30	1 GB	$10
Illumina Miseq	27h	2x300bp	> Q30	15 GB	$0.15
Illumina Hiseq 2500	1 - 10days	2x250bp	>Q30	3000 GB	$0.05
Ion torrent	2h	400bp	>Q20	50MB-1GB	$1
Pacific Biosciences	30m - 4h	10kb - >40kb	>Q50 consensus >Q10 single	500 - 1000MB /SMRT cell	$0.13 - $0.60

http://www.hindawi.com/journals/bmri/2012/251364

Growing demand

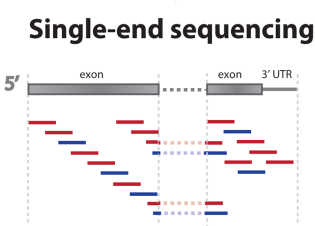


http://enseqlopedia.com/ngs-mapped/

Sequencing options

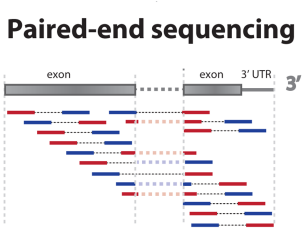
Single-end

- Cheaper
- Suitable for more general purpose analyzes, e.g. DE



Paired-end

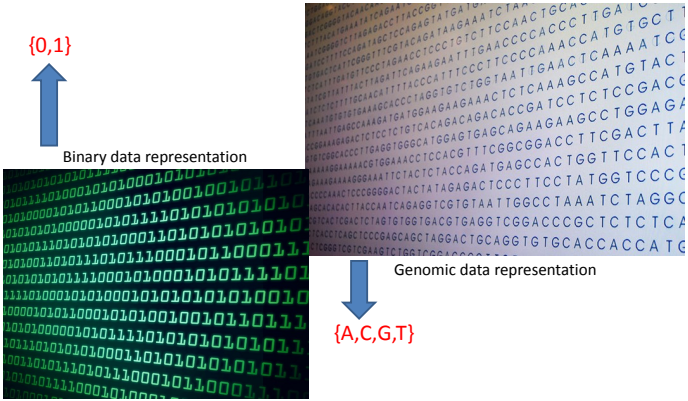
- More information regarding the length and the position of a read
- Useful for spliced junctions, indels etc.



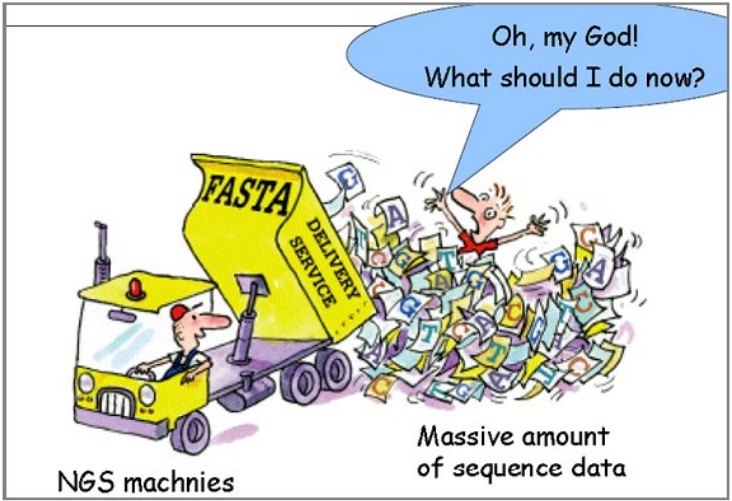
Zhernakova, Daria V., et al. "DeepSAGE reveals genetic variants associated with alternative polyadenylation and expression of coding and non-coding transcripts."

(2013): e1003594.

Bioinformatics is a necessity I



Bioinformatics is a necessity II



FASTQ file format

FASTQ

Text-based format for storing biological sequences

- Raw unaligned reads (nucleotides)
- Corresponding quality scores

```
@HWI-ST661:319:D28MYACXX:6:1101:1170:2180 1:N:0:GTGGCC
NAGTGGTTTATGCCTGTAATCCCAGCATTTTGGGAGACGAAGTTGAGAN
+
#1:ADDFFHGHHHIJGHHIIJJJIIIEHIJJJIHEHIGHIJJHHJGHC#
```

Phred quality score

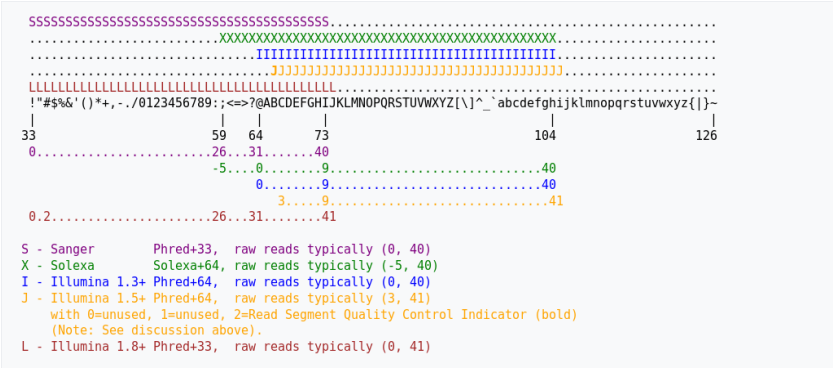
$$Q = -10 \log_{10} P$$

P: base-calling error probability

Q: Phred quality score

Phred Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10,000	99.99%
50	1 in 100,000	99.999%
60	1 in 1,000,000	99.9999%

Phred quality score encoding



https://en.wikipedia.org/wiki/FASTQ_format

FASTA file format I

FASTA

Text-based format for representing either nucleotide sequences or peptides encoded as a character

- Starts with the character “>” followed by an (alpharithmetic) identification code
- One or more lines contain the sequence

```
>1 dna:chromosome chromosome:GRCh37:1:1:249250621:1
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNTAACCCTAACCCTAACCCTA
ACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTA
ACCCTAACCCTAACCCTAACCCTAACCCAACCCTAACCCTAACCCTAACCCTAACCCTAA
CCCTAACCCCTAACCCTAACCCTAACCCTAACCCTAACCTAACCCTAACCCTAACCCTAA
CCCTAACCCTAACCCTAACCCTAACCCTAACCCCTAACCCTAACCCTAAACCCTAAACCC
```

FASTA file format II

- During the alignment process the reads of a fastq file are mapped on a reference genome stored in fasta format
- There are various available genomes, e.g. :
  - Human: hg16 (2003), hg17 (2004), hg18 (2006), hg19 (NCBI)/GRCh37 (Ensembl) (2009), hg38/GRCh38 (2013)
  - Mouse: mm7 (2005), mm8 (2006), mm9 (2007), mm10 (2011)
  - D. melanogaster: dm1 (2003), dm2 (2004), dm3 (2006), dm6 (2014)
  - . . .

SAM/BAM format I

SAM - Sequence Alignment Map

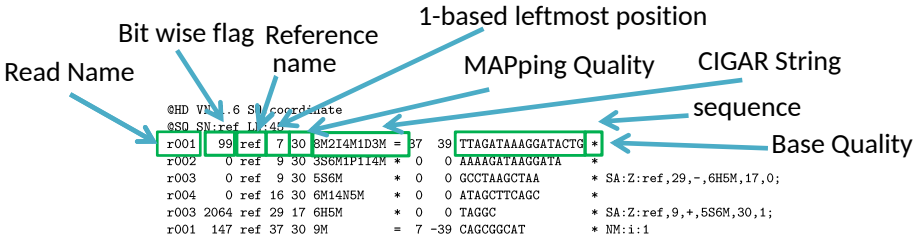
SAM format stores aligned reads and is independent of the sequencing technology used

- SAM: textbased
- BAM: binary

SAM/BAM format II

```
@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,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCAGC       *
r003 2064 ref 29 17 6H5M       *  0   0 TAGGC             * SA:Z:ref,9,+,5S6M,30,1;
r001  147 ref 37 30 9M         =  7 -39 CAGCGGCAT         * NM:i:1
```

SAM/BAM format III



BED format

BED - **B**rowser **E**xtensible **D**ata

Per line

- 3 mandatory fields
  - chrom - The chromosome name, e.g. chr3, chrY, chr2_random
  - chromStart - The start position of a feature in the chromosome - **0-based numbering**.
  - chromEnd - The end position of a feature in the chromosome. The end position **in not** included in the feature. E.g. the first 100 bases of a chromosome are defined as chromStart=0, chromEnd=100, and are the bases 0-99
- 9 optional fields
  - name, score, strand, thickStart, thickEnd, itemRgb, blockCount, blockSizes, blockStarts

VCF format I

VCF files - **V**ariant **C**all **F**ormat

- For encoding structural variations
- Widely used by the 1000 Genomes Project
- Only the variants are stored along with the reference genome

VCF format II

```
##fileformat=VCFv4.1
##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
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
```

VCF format III

```
#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
```

- AF: allele frequency for each ALT allele
- DP: combined depth across samples
- DB: dbSNP membership
- H2: membership in hapmap2
- NS: Number of samples with data

index files

Index creation, for faster data access since it enables non sequential search

- fasta $\rightarrow$ fai
- bam $\rightarrow$ bai
- vcf $\rightarrow$ idx
- vcf.gz $\rightarrow$ tbi

samtools I

samtools

Samtools is a suite of programs for interacting with high-throughput sequencing data. It consists of three separate repositories:
Samtools Reading/writing/editing/indexing/viewing SAM/BAM/CRAM format
BCFtools Reading/writing BCF2/VCF/gVCF files and calling/filtering/summarising SNP and short indel sequence variants
HTSlib A C library for reading/writing high-throughput sequencing data

`http://www.htslib.org/`

samtools II

```
$ samtools

Program: samtools (Tools for alignments in the SAM format)
Version: 1.3 (using htslib 1.3)

Usage:   samtools <command> [options]

Commands:
  -- Indexing
     dict           create a sequence dictionary file
     faidx          index/extract FASTA
     index          index alignment
```

samtools III

```
-- Editing
   calmd          recalculate MD/NM tags and '=' bases
   fixmate        fix mate information
   reheader       replace BAM header
   rmdup          remove PCR duplicates
   targetcut      cut fosmid regions (for fosmid pool only)
   addreplacerg   adds or replaces RG tags

-- File operations
   collate        shuffle and group alignments by name
   cat            concatenate BAMs
   merge          merge sorted alignments
   mpileup        multi-way pileup
```

samtools IV

```
    sort            sort alignment file
    split           splits a file by read group
    quickcheck      quickly check if SAM/BAM/CRAM file appears intact
    fastq           converts a BAM to a FASTQ
    fasta           converts a BAM to a FASTA

-- Statistics
    bedcov          read depth per BED region
    depth           compute the depth
    flagstat        simple stats
    idxstats        BAM index stats
    phase           phase heterozygotes
    stats           generate stats (former bamcheck)
```

samtools V

```
-- Viewing
   flags          explain BAM flags
   tview          text alignment viewer
   view           SAM<->BAM<->CRAM conversion
   depad          convert padded BAM to unpadded BAM
```

samtools VI

- **sort**: after the alignment process, the final sam file contains the reads in a random order. For random access and also for the conversion to bam, the sam file needs to be sorted

```
samtools sort sample.bam sample.sorted -@8
```

- **index**: for the faster data access of the bam file

```
samtools index sample.sorted.bam
```

- **view**: for sam to bam conversion and vice versa. For filtering a sam/bam file based on conditions

```
samtools view sample.bam
samtools view -f 0x2 sample.sorted.bam
samtools view -F 0x2 sample.sorted.bam
```

Bedtools I

bedtools: a powerful toolset for genome arithmetic

Collectively, the bedtools utilities are a swiss-army knife of tools for a wide-range of genomics analysis tasks. The most widely-used tools enable genome arithmetic: that is, set theory on the genome. For example, bedtools allows one to intersect, merge, count, complement, and shuffle genomic intervals from multiple files in widely-used genomic file formats such as BAM, BED, GFF/GTF, VCF. While each individual tool is designed to do a relatively simple task (e.g., intersect two interval files), quite sophisticated analyses can be conducted by combining multiple bedtools operations on the UNIX command line

`http://bedtools.readthedocs.io/en/latest/`

Bedtools II

```
$ bedtools
bedtools: flexible tools for genome arithmetic and DNA sequence analysis.
usage:    bedtools <subcommand> [options]

The bedtools sub-commands include:

[ Genome arithmetic ]
    intersect     Find overlapping intervals in various ways.
    window        Find overlapping intervals within a window around
                  an interval.
...
```

Bedtools III

bedtools intersect

By far, the most common question asked of two sets of genomic features is whether or not any of the features in the two sets “overlap” with one another. This is known as feature intersection. bedtools intersect allows one to screen for overlaps between two sets of genomic features. Moreover, it allows one to have fine control as to how the intersections are reported. bedtools intersect works with both BED/GFF/VCF and BAM files as input.

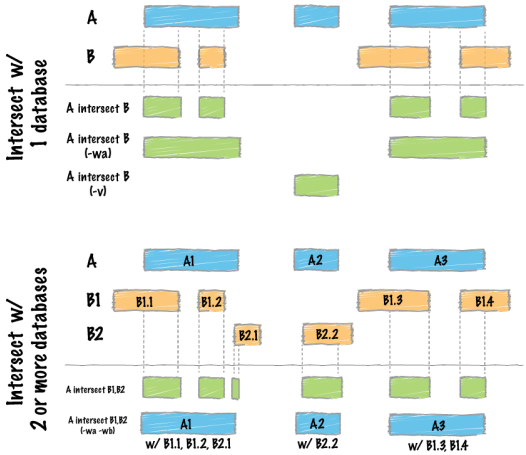
Bedtools IV

```
$ bedtools intersect
Tool:    bedtools intersect (aka intersectBed)
Version: v2.21.0
Summary: Report overlaps between two feature files.

Usage:   bedtools intersect [OPTIONS] -a <bed/gff/vcf> -b <bed/gff/vcf>

        Note: -b may be followed with multiple databases and/or
        wildcard (*) character(s).
...
```

Bedtools V


Intersect w/ 1 database
A
B
A intersect B
A intersect B (-wa)
A intersect B (-v)
Intersect w/ 2 or more databases
A
A1
A2
A3
B1
B1.1
B1.2
B1.3
B1.4
B2
B2.1
B2.2
A intersect B1,B2
A intersect B1,B2 (-wa -wb)
A1
A2
A3
w/ B1.1, B1.2, B2.1
w/ B2.2
w/ B1.3, B1.4

Bedtools VI

```
$ cat A.bed
chr1  10  20
chr1  30  40

$ cat B.bed
chr1  15   20

$ bedtools intersect -a A.bed -b B.bed
chr1  15   20
```

Bedtools VII

bedtools merge

bedtools merge combines overlapping or “book-ended” features in an interval file into a single feature which spans all of the combined features.

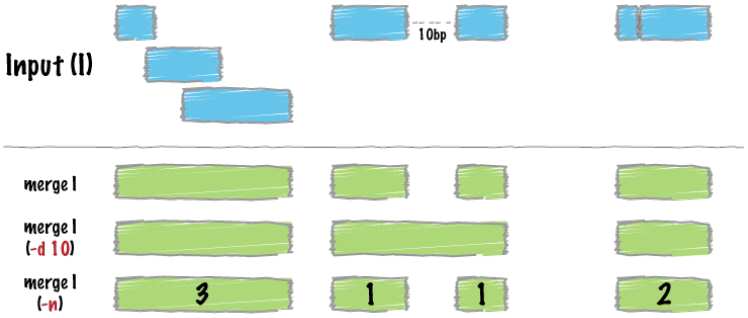
Bedtools VIII

```
$ bedtools merge
Tool:    bedtools merge (aka mergeBed)
Version: v2.21.0
Summary: Merges overlapping BED/GFF/VCF entries into a single interval.

Usage:   bedtools merge [OPTIONS] -i <bed/gff/vcf>

...
```

Bedtools IX



Bedtools X

```
$ cat A.bed
chr1  100  200
chr1  180  250
chr1  250  500
chr1  501  1000

$ bedtools merge -i A.bed
chr1  100  500
chr1  501  1000
```

Bedtools XI

bedtools genomecov

bedtools genomecov computes histograms (default), per-base reports (-d) and BEDGRAPH (-bg) summaries of feature coverage (e.g., aligned sequences) for a given genome.

Bedtools XII

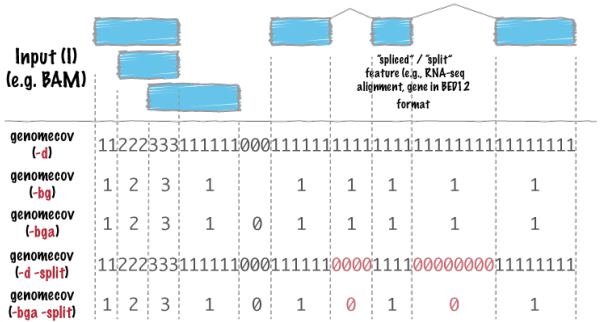
```
$ bedtools genomecov

Tool:    bedtools genomecov (aka genomeCoverageBed)
Version: v2.21.0
Summary: Compute the coverage of a feature file among a genome.

Usage: bedtools genomecov [OPTIONS] -i <bed/gff/vcf> -g <genome>

...
```

Bedtools XIII



Bedtools XIV

```
$ cat A.bed
chr1  10  20
chr1  20  30
chr2  0   500

$ cat my.genome
chr1  1000
chr2  500

$ bedtools genomecov -i A.bed -g my.genome
chr1   0  980  1000  0.98
chr1   1  20   1000  0.02
chr2   1  500  500   1
```

Bedtools XV

```
genome 0   980   1500   0.653333
genome 1   520   1500   0.346667
```

VCFtools I

Welcome to VCFtools

VCFtools is a program package designed for working with VCF files, such as those generated by the 1000 Genomes Project. The aim of VCFtools is to provide easily accessible methods for working with complex genetic variation data in the form of VCF files.

`https://vcftools.github.io/index.html`

VCFtools II

```
$ vcftools

VCFtools (0.1.15)
© Adam Auton and Anthony Marcketta 2009

Process Variant Call Format files

For a list of options, please go to:
    https://vcftools.github.io/man_latest.html
Alternatively, a man page is available, type:
    man vcftools
Questions, comments, and suggestions should be emailed to:
    vcftools-help@lists.sourceforge.net
```

VCFtools III

vcf-isec

Creates intersections and complements of two or more VCF files. Given multiple VCF files, it can output the list of positions which are shared by at least N files, at most N files, exactly N files, etc. The first example below outputs positions shared by at least two files and the second outputs positions present in the files A but absent from files B and C.

```
vcf-isec -n +2 A.vcf.gz B.vcf.gz | bgzip -c > out.vcf.gz
vcf-isec -c A.vcf.gz B.vcf.gz C.vcf.gz | bgzip -c > out.vcf.gz
```

VCFtools IV

vcf-merge

Merges two or more VCF files into one so that, for example, if two source files had one column each, on output will be printed a file with two columns. See also vcf-concat for concatenating VCFs split by chromosome.

```
$ vcf-merge A.vcf.gz B.vcf.gz C.vcf.gz | bgzip -c > out.vcf.gz
```

VCFtools V

vcf-concat

Concatenates VCF files (for example split by chromosome). Note that the input and output VCFs will have the same number of columns, the script does not merge VCFs by position (see also vcf-merge).

```
$ vcf-concat A.vcf.gz B.vcf.gz C.vcf.gz | gzip -c > out.vcf.gz
```

VCFtools VI

vcf-compare

Compares positions in two or more VCF files and outputs the numbers of positions contained in one but not the other files; two but not the other files, etc, which comes handy when generating Venn diagrams. The script also computes numbers such as nonreference discordance rates (including multiallelic sites), compares actual sequence (useful when comparing indels), etc.

```
$ vcf-compare  A.vcf.gz B.vcf.gz C.vcf.gz
```

bcftools

bcftools

BCFtools is a set of utilities that manipulate variant calls in the Variant Call Format (VCF) and its binary counterpart BCF. All commands work transparently with both VCFs and BCFs, both uncompressed and BGZF-compressed.

Most commands accept VCF, bgzipped VCF and BCF with filetype detected automatically even when streaming from a pipe. Indexed VCF and BCF will work in all situations. Un-indexed VCF and BCF and streams will work in most, but not all situations. In general, whenever multiple VCFs are read simultaneously, they must be indexed and therefore also compressed.

BCFtools is designed to work on a stream. It regards an input file "-" as the standard input (stdin) and outputs to the standard output (stdout). Several commands can thus be combined with Unix pipes.

https://samtools.github.io/bcftools/bcftools.html

Familiarizing with CLI tools

Lab Exercise 5 - Familiarizing with CLI tools

https://eclass.uoa.gr/modules/work/index.php?course=DI425&id=54428
or
https://eclass.uoa.gr/modules/document/file.php/DI425/2024-25/exercises/ITBI2024-2025-exercise5-ACD14112024.zip

Due for 28/11/24

Questions?