Aligning DNA sequences on compressed collections of genomes

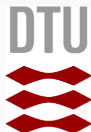
Part 5. Practical session: alignment and SNP calling

The CODATA-RDA Research Data Science Applied workshop on Bioinformatics
ICTP, Trieste - Italy
July 24-28, 2017

Nicola Prezza

Technical University of Denmark
DTU Compute
DK-2800 Kgs. Lyngby
Denmark

Slides adapted from
"Fastq and SAM formats Visualize at single base level", Cristian Del Fabbro



Outline

Fastq

SAM format

View alignment at single base level

SNP calling

Fastq

Raw data

The RAW data we get as input is a list of DNA *reads*

Each read comes with its name and *quality* (i.e. how sure we are that each base called by the sequencer is correct)

fastq format: 4 lines for each read (see next slide)

Raw Data

```
@HISEQ1:83:B06F9ABXX:1:1101:13:21 1:N:0:ACTTGA
CCGGTGTAAGCTTAGGCCTTTGACATGTGAACGATAAGGTCAACG
+
CCCFFFFFHHHHHJJJIJJJJIIJJJJIJJJJJJJJJJHIJJJI
```

version	conversion
Illumina $>=$ 1.8 Sanger	`ASCII(BQ + 33)`

Raw Data

Decimal - Binary - Octal - Hex – ASCII
Conversion Chart

Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII
0	00000000	000	00	NUL	32	00100000	040	20	SP	64	01000000	100	40	@	96	01100000	140	60	`
1	00000001	001	01	SOH	33	00100001	041	21	!	65	01000001	101	41	A	97	01100001	141	61	a
2	00000010	002	02	STX	34	00100010	042	22	"	66	01000010	102	42	B	98	01100010	142	62	b
3	00000011	003	03	ETX	35	00100011	043	23	#	67	01000011	103	43	C	99	01100011	143	63	c
4	00000100	004	04	EOT	36	00100100	044	24	$	68	01000100	104	44	D	100	01100100	144	64	d
5	00000101	005	05	ENQ	37	00100101	045	25	%	69	01000101	105	45	E	101	01100101	145	65	e
6	00000110	006	06	ACK	38	00100110	046	26	&	70	01000110	106	46	F	102	01100110	146	66	f
7	00000111	007	07	BEL	39	00100111	047	27	'	71	01000111	107	47	G	103	01100111	147	67	g
8	00001000	010	08	BS	40	00101000	050	28	(	72	01001000	110	48	H	104	01101000	150	68	h
9	00001001	011	09	HT	41	00101001	051	29	)	73	01001001	111	49	I	105	01101001	151	69	j
10	00001010	012	0A	LF	42	00101010	052	2A	*	74	01001010	112	4A	J	106	01101010	152	6A	j
11	00001011	013	0B	VT	43	00101011	053	2B	+	75	01001011	113	4B	K	107	01101011	153	6B	k
12	00001100	014	0C	FF	44	00101100	054	2C	,	76	01001100	114	4C	L	108	01101100	154	6C	l
13	00001101	015	0D	CR	45	00101101	055	2D	-	77	01001101	115	4D	M	109	01101101	155	6D	m
14	00001110	016	0E	SO	46	00101110	056	2E	.	78	01001110	116	4E	N	110	01101110	156	6E	n
15	00001111	017	0F	SI	47	00101111	057	2F	/	79	01001111	117	4F	O	111	01101111	157	6F	o
16	00010000	020	10	DLE	48	00110000	060	30	0	80	01010000	120	50	P	112	01110000	160	70	p
17	00010001	021	11	DC1	49	00110001	061	31	1	81	01010001	121	51	Q	113	01110001	161	71	q
18	00010010	022	12	DC2	50	00110010	062	32	2	82	01010010	122	52	R	114	01110010	162	72	r
19	00010011	023	13	DC3	51	00110011	063	33	3	83	01010011	123	53	S	115	01110011	163	73	s
20	00010100	024	14	DC4	52	00110100	064	34	4	84	01010100	124	54	T	116	01110100	164	74	t
21	00010101	025	15	NAK	53	00110101	065	35	5	85	01010101	125	55	U	117	01110101	165	75	u
22	00010110	026	16	SYN	54	00110110	066	36	6	86	01010110	126	56	V	118	01110110	166	76	v
23	00010111	027	17	ETB	55	00110111	067	37	7	87	01010111	127	57	W	119	01110111	167	77	w
24	00011000	030	18	CAN	56	00111000	070	38	8	88	01011000	130	58	X	120	01111000	170	78	x
25	00011001	031	19	EM	57	00111001	071	39	9	89	01011001	131	59	Y	121	01111001	171	79	y
26	00011010	032	1A	SUB	58	00111010	072	3A	:	90	01011010	132	5A	Z	122	01111010	172	7A	z
27	00011011	033	1B	ESC	59	00111011	073	3B	;	91	01011011	133	5B	[	123	01111011	173	7B	{
28	00011100	034	1C	FS	60	00111100	074	3C	<	92	01011100	134	5C	\	124	01111100	174	7C	|
29	00011101	035	1D	GS	61	00111101	075	3D	=	93	01011101	135	5D	]	125	01111101	175	7D	}
30	00011110	036	1E	RS	62	00111110	076	3E	>	94	01011110	136	5E	^	126	01111110	176	7E	~
31	00011111	037	1F	US	63	00111111	077	3F	?	95	01011111	137	5F	_	127	01111111	177	7F	DEL

This work is licensed under the Creative Commons Attribution-ShareAlike License. To view a copy of this license, visit http://creativecommons.org/licenses/by-sa/3.0/

ASCII Conversion Chart.doc Copyright © 2008 Donald Weiman 12 August 2008

Phred values

Phred Quality Score	Probability of Incorrect Based Call	Base Call Accuracy	Q-score
10	1 in 10	90%	Q10
20	1 in 100	99%	Q20
30	1 in 1000	99.9%	Q30
40	1 in 10000	99.99%	Q40

$$\text{error probability} = \frac{1}{10^{\frac{Q}{10}}}$$

Example

In the previous slide, a base associated with quality 'J' has Phred quality score ASCII(J)-33 = 74 - 33 = 41. The probability that this base is incorrect is $1/(10^{41/10}) \approx 0.00008$

Exercise

Create a valid fastq file `/scratch/2M_low_quality.fastq` containing all sequences from `/scratch/2M.fastq` that have a sub-sequence of at least 20 bases with Phred score 0. How many sequences do you obtain?

Hint

Convert Phred 0 to ASCII, use `grep` to search, filter the result to remove extra symbols added by `grep`.

SAM format

The SAM and BAM formats

A DNA aligner takes as input an indexed genome and a fastq file and produces a SAM or BAM file

A SAM file contains, for every aligned read, the information relative to the alignment. SAM is a text format: you can visualize and read it.

The SAM and BAM formats

BAM is the binary version of SAM. In a BAM file, information is ”packed” and cannot be directly visualized. As a result, BAM files are much smaller than SAM.

Using **samtools** we can (among other things), convert SAM $\leftrightarrow$ BAM

Inside the SAM/BAM file

```
@SQ     SN:Chr1 LN:500000
@SQ     SN:Chr2 LN:500000
@SQ     SN:Chr3 LN:500000
@SQ     SN:Chr4 LN:500000
@PG     ID:bwa  PN:bwa  VN:0.6.1-r104

ILLUMINA-BA4A85_0078:6:10:15480:18085#0 73      Chr1    4       25      100M    =       4       0
GGCGAGACTACCAGTTCTTAGATTCGTCAAGATTGGTCTTAATCAGTTTCCACTCTACACCTCAA
ATTGTCCACATGGTCGGGTGTCCAGAGTGCCCCAA
ffffffffdfffefcfffdfffcffcff^ff^fffd^ceece^eefedfdfffeefdf`fdae\edaabbee^dc__`YaBBBBBBBBBBBBBBBB
BBBB    XT:A:U  NM:i:5  SM:i:25 AM:i:0  X0:i:1  X1:i:0  XM:i:5  XO:i:0  XG:i:0
MD:Z:15G24C35A0A16T5

ILLUMINA-BA4A85_0078:6:10:15480:18085#0 133     Chr1    4       0       *       =       4       0
GGCGAGACTACCAGTTCTTAGATTCGTCAAGATTGGTCTTAATCAGTTTCCACTCTACACCTCAA
ATTGTCCACATGGTCGGGTGTCCAGAGTGCCCCAA
ffffffffdfffefcfffdfffcffcff^ff^fffd^ceece^eefedfdfffeefdf`fdae\edaabbee^dc__`YaBBBBBBBBBBBBBBBB
BBBB
```

Columns

Col	Field	Type	Brief description
1	QNAME	String	Query template NAME
2	FLAG	Int	bitwise FLAG
3	RNAME	String	Reference sequence NAME
4	POS	Int	1-based leftmost mapping POSition
5	MAPQ	Int	MAPping Quality
6	CIGAR	String	CIGAR string
7	RNEXT	String	Ref. name of the mate/next segment
8	PNEXT	Int	Position of the mate/next segment
9	TLEN	Int	observed Template LENgth
10	SEQ	String	segment SEQuence
11	QUAL	String	ASCII of Phred-scaled base QUALity+33

Flags

Bit	Description
0x1	template having multiple segments in sequencing
0x2	each segment properly aligned according to the aligner
0x4	segment unmapped
0x8	next segment in the template unmapped
0x10	SEQ being reverse complemented
0x20	SEQ of the next segment in the template being reversed
0x40	the first segment in the template
0x80	the last segment in the template
0x100	secondary alignment
0x200	not passing quality controls
0x400	PCR or optical duplicate

Flags

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

Flag: 163 Explain

Switch to mate

Explanation:

- [x] read paired
- [x] read mapped in proper pair
- [ ] read unmapped
- [ ] mate unmapped
- [ ] read reverse strand
- [x] mate reverse strand
- [ ] first in pair
- [x] second in pair
- [ ] not primary alignment
- [ ] read fails platform/vendor quality checks
- [ ] read is PCR or optical duplicate
- [ ] supplementary alignment

Other info

After the quality string, there are other info. In particular, the field NM:i tells us the number of mismatches of the alignment

bwa mem

BWA

BWA (Burrows-Wheeler aligner) is one of the most accurate and fast DNA aligners. We will use the algorithm BWA-MEM (the newest and more optimized for reads of length ≥ 70)

Index construction

```
bwa index genome.fa
```

Alignment

single reads:

```
bwa mem genome.fa reads.fastq > out.sam
```

paired-end reads:

```
bwa mem genome.fa reads_1.fastq reads_2.fastq > out.sam
```

bwa mem

Exercise

Align the paired-end reads `2M_1.fastq` and `2M_2.fastq` on the genome `hg38_reduced.fa`, and save the alignment in a file `alignment.sam` in folder `alignment`

Exercise

Count the number of alignments with 1, 2, ..., 9 mismatches

View alignment at single base level

The result of an alignment can be visualized using graphical tools such as **tablet**

Before using tablet, we must convert the SAM file to BAM and the BAM file must be sorted and indexed.

Indexing is needed to speed-up the retrieval of alignments overlapping a specific genome position

SAM to BAM conversion

To convert SAM to BAM, we use **samtools**:

```
samtools view -b -S alignment.sam > alignment.bam
```

Flag -S means that input is SAM. Flag -b means that output must be BAM.

Samtools and Tablet

Sorting and indexing bam files

`samtools sort input.bam out_sorted` (creates file out_sorted.bam)

`samtools index out_sorted.bam`

Visualize

- Just type "`tablet`" in the terminal and a interactive program starts.
- Open assembly → select the sorted bam and fasta files
- Selecting color schemes → variants we can visualize errors and SNPs

SNP calling

SNP calling

We can call SNPs using samtools/bcftools:

```
samtools mpileup -uD -f genome.fasta alignment_sorted.bam
| bcftools view -vc - > calls.vcf
```

samtools part

- -u tells it to output into an uncompressed bcf file (rather than compressed)
- -D tells it to keep read depth for each sample
- -f tells it that the next argument is going to be the reference genome file

bcftools part

- -v tells to output vcf file (ASCII, readable) rather than bcf (binary)
- -c tells to do SNP calling
- the last ”-” means that input comes from stdin (i.e. the pipe)

VCF format

```
##fileformat=VCFv4.0
##fileDate=20100707
##source=VCFtools
##reference=NCBI36
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality (phred score)">
##FORMAT=<ID=GL,Number=3,Type=Float,Description="Likelihoods for RR,RA,AA genotypes (R=ref,A=alt)">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##ALT=<ID=DEL,Description="Deletion">
##INFO=<ID=SVTYPE,Number=1,Type=String,Description="Type of structural variant">
##INFO=<ID=END,Number=1,Type=Integer,Description="End position of the variant">
#CHROM POS ID     REF  ALT     QUAL FILTER  INFO                FORMAT     SAMPLE1   SAMPLE2
1        1 .      ACG  A,AT    .    PASS    .                   GT:DP      1/2:13    0/0:29
1        2 rs1    C    T,CT         PASS    H2;AA=T             GT:GQ      0|1:100   2/2:70
1        5 .      A    G       .    PASS    .                   GT:GQ      1|0:77    1/1:95
1      100 .      T    <DEL>   .    PASS    SVTYPE=DEL;END=300  GT:GQ:DP   1/1:12:3  0/0:20
```

VCF header

Body

Mandatory header lines

Optional header lines (meta-data about the annotations in the VCF body)

Reference alleles (GT=0)

Alternate alleles (GT>0 is an index to the ALT column)

Phased data (G and C above are on the same chromosome)

Deletion

SNP

Large SV

Insertion

Other event

SNP calling

Exercise

Call the SNPs resulting from the alignment of `2M_1.fastq` and `2M_2.fastq` on the genome `hg38_reduced.fa`, and save the output in a file `calls.vcf` in folder `calls`

Exercise

Use `tablet` to manually verify and visualize some SNP positions predicted in the previous exercise (use function ”jump to base”)