HTS data formats and Quality Control

petr.danecek@sanger.ac.uk

wellcome trust sanger institute

Data Formats

FASTQ

- Unaligned read sequences with base qualities

SAM/BAM

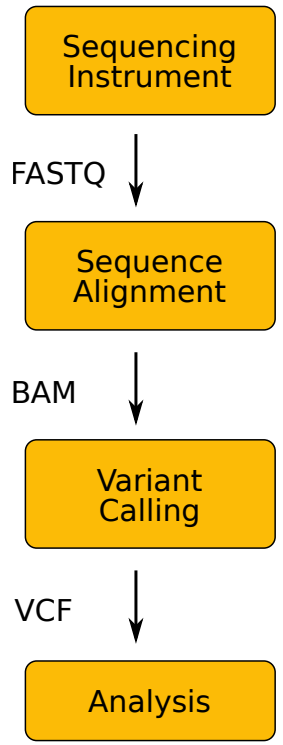
- Unaligned or aligned reads
- Text and binary formats

CRAM

- Better compression than BAM

VCF/BCF

- Flexible variant call format
- Arbitrary types of sequence variation
- SNPs, indels, structural variations



Specifications maintained by the Global Alliance for Genomics and Health

FASTA - reference genome

```
>1 dna:chromosome chromosome:GRCh37:1:1:249250621:1
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
TATTCAAAAAATTGAGAATTTCTGACCACTTAACAAACCCACAGAAAATCCACCCGAGTG
CACTGAGCACGCCAGAAATCAGGTGGCCTCAAAGAGCTGCTCCCACCTGAAGGAGACGCG
CTGCTGCTGCTGTCGTCCTGCCTGGCGCCTTGGCCTACAGGGGCCGCGGTTGAGGGTGGG
AGTGGGGGTGCACTGGCCAGCACCTCAGGAGCTGGGGGTGGTGGTGGGGGCGGTGGGGGT
GGTGTTAGTACCCCATCTTGTAGGTCTGAAACACAAAGTGTGGGGTGTCTAGGGAAGAAG
>2
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
AAAAGCATTTATGCTACAAATTACTATGGTAATTATGCTACAAATTTATGGTACCATAAA
TTACCATAGTAATTTGTAGCATAAATTTGTACTATGGTACAAATTACATGGGAGAGTGAA
GGTGGGTTAAAACATTCATATTAAAGAACTTCCACTCAGATTGCAAGAAAAGAGAGAGGA
ATGGAGATGGTAGCACAAGTCCCTACAATAAAAGTAGATGTTTTGAGATCAGTTCTATTT
```

FASTA - reference genome

```
>1 dna:chromosome chromosome:GRCh37:1:1:249250621:1
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
TATTCAAAAAATTGAGAATTTCTGACCACTTAACAAACCCACAGAAAATCCACCCGAGTG
CACTGAGCACGCCAGAAATCAGGTGGCCTCAAAGAGCTGCTCCCACCTGAAGGAGACGCG
CTGCTGCTGCTGTCGTCCTGCCTGGCGCCTTGGCCTACAGGGGCCGCGGTTGAGGGTGGG
AGTGGGGGTGCACTGGCCAGCACCTCAGGAGCTGGGGGTGGTGGTGGGGGCGGTGGGGGT
GGTGTTAGTACCCCATCTTGTAGGTCTGAAACACAAAGTGTGGGGTGTCTAGGGAAGAAG
>2
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
AAAAGCATTTATGCTACAAATTACTATGGTAATTATGCTACAAATTTATGGTACCATAAA
TTACCATAGTAATTTGTAGCATAAATTTGTACTATGGTACAAATTACATGGGAGAGTGAA
GGTGGGTTAAAACATTCATATTAAAGAACTTCCACTCAGATTGCAAGAAAAGAGAGAGGA
ATGGAGATGGTAGCACAAGTCCCTACAATAAAAGTAGATGTTTTGAGATCAGTTCTATTT
```

2003	NCBI Build 34	hg16
2004	NCBI Build 35	hg17
2006	NCBI Build 36.1	hg18
2009	GRCh37	hg19
2013	GRCh38	hg38

FASTQ

- Simple format for raw unaligned sequencing reads
- Extension to the FASTA file format
- Sequence and an associated per base quality score

Read 1

```
@ERR007731.739 IL16_2979:6:1:9:1684/1
CTTGACGACTTGAAAAATGACGAAATCACTAAAAAACGTGAAAAATGAGAAATG...
+
BBCBCBBBBBBBABBABBBBBBBABBBBBBBBBBBBBBABAAAABBBBB=@>BB...
```

← **Read name**

← **Sequence**

← **Base qualities**

Read 2

```
@ERR007731.740 IL16_2979:6:1:9:1419/1
AAAAAAAAAGATGTCATCAGCACATCAGAAAAGAAGGCAACTTTAAAACTTTTC...
+
BBABBBABABAABABABBABBBAAA>@B@BBAA@4AAA>.>BAA@779:AAA@A...
```

- Quality encoded in ASCII characters with decimal codes 33-126
  - ASCII code of "A" is 65, the corresponding quality is Q$= 65 - 33 = 32$
  - Phred quality score: $P = 10^{-Q/10}$

```
perl -e 'printf "%d\n",ord("A")-33;'
```

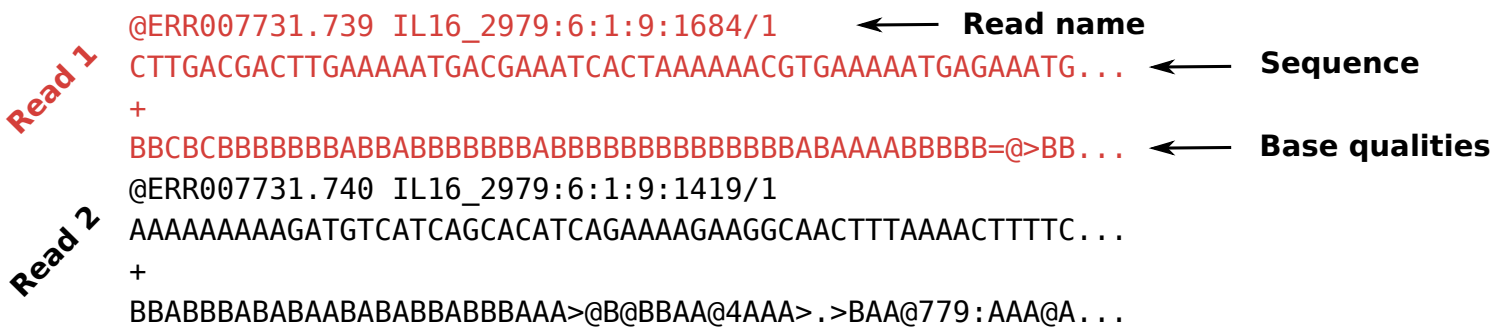
- Beware: multiple quality scores were in use!
  - Sanger, Solexa, Illumina 1.3+
- Paired-end sequencing produces two FASTQ files

FASTQ

Quality	Probability of error	Accuracy
10 (Q10)	1 in 10	90%
20 (Q20)	1 in 100	99%
30 (Q30)	1 in 1000	99.9%
40 (Q40)	1 in 10000	99.99%

FASTQ

- Simple format for raw unaligned sequencing reads
- Extension to the FASTA file format
- Sequence and an associated per base quality score



- Quality encoded in ASCII characters with decimal codes 33-126
  - ASCII code of "A" is 65, the corresponding quality is Q$= 65 - 33 = 32$
  - Phred quality score: $P = 10^{-Q/10}$

```
perl -e 'printf "%d\n",ord("A")-33;'
```

- Beware: multiple quality scores were in use!
  - Sanger, Solexa, Illumina 1.3+
- Paired-end sequencing produces two FASTQ files

ASCII Table

0	NUL '\0' (null character)	33	!	66	B	99	c
1	SOH (start of heading)	34	"	67	C	100	d
2	STX (start of text)	35	#	68	D	101	e
3	ETX (end of text)	36	$	69	E	102	f
4	EOT (end of transmission)	37	%	70	F	103	g
5	ENQ (enquiry)	38	&	71	G	104	h
6	ACK (acknowledge)	39	'	72	H	105	i
7	BEL '\a' (bell)	40	(	73	I	106	j
8	BS '\b' (backspace)	41	)	74	J	107	k
9	HT '\t' (horizontal tab)	42	*	75	K	108	l
10	LF '\n' (new line)	43	+	76	L	109	m
11	VT '\v' (vertical tab)	44	,	77	M	110	n
12	FF '\f' (form feed)	45	-	78	N	111	o
13	CR '\r' (carriage ret)	46	.	79	O	112	p
14	SO (shift out)	47	/	80	P	113	q
15	SI (shift in)	48	0	81	Q	114	r
16	DLE (data link escape)	49	1	82	R	115	s
17	DC1 (device control 1)	50	2	83	S	116	t
18	DC2 (device control 2)	51	3	84	T	117	u
19	DC3 (device control 3)	52	4	85	U	118	v
20	DC4 (device control 4)	53	5	86	V	119	w
21	NAK (negative ack.)	54	6	87	W	120	x
22	SYN (synchronous idle)	55	7	88	X	121	y
23	ETB (end of trans. blk)	56	8	89	Y	122	z
24	CAN (cancel)	57	9	90	Z	123	{
25	EM (end of medium)	58	:	91	[	124	|
26	SUB (substitute)	59	;	92	\	125	}
27	ESC (escape)	60	<	93	]	126	~
28	FS (file separator)	61	=	94	^	127	DEL
29	GS (group separator)	62	>	95	_		
30	RS (record separator)	63	?	96	`		
31	US (unit separator)	64	@	97	a		
32	SPACE	65	A	98	b		

ASCII Table

0	NUL '\0' (null character)	33	!	66	B	99	c
1	SOH (start of heading)	34	"	67	C	100	d
2	STX (start of text)	35	#	68	D	101	e
3	ETX (end of text)	36	$	69	E	102	f
4	EOT (end of transmission)	37	%	70	F	103	g
5	ENQ (enquiry)	38	&	71	G	104	h
6	ACK (acknowledge)	39	'	72	H	105	i
7	BEL '\a' (bell)	40	(	73	I	106	j
8	BS '\b' (backspace)	41	)	74	J	107	k
9	HT '\t' (horizontal tab)	42	*	75	K	108	l
10	LF '\n' (new line)	43	+	76	L	109	m
11	VT '\v' (vertical tab)	44	,	77	M	110	n
12	FF '\f' (form feed)	45	-	78	N	111	o
13	CR '\r' (carriage ret)	46	.	79	O	112	p
14	SO (shift out)	47	/	80	P	113	q
15	SI (shift in)	48	0	81	Q	114	r
16	DLE (data link escape)	49	1	82	R	115	s
17	DC1 (device control 1)	50	2	83	S	116	t
18	DC2 (device control 2)	51	3	84	T	117	u
19	DC3 (device control 3)	52	4	85	U	118	v
20	DC4 (device control 4)	53	5	86	V	119	w
21	NAK (negative ack.)	54	6	87	W	120	x
22	SYN (synchronous idle)	55	7	88	X	121	y
23	ETB (end of trans. blk)	56	8	89	Y	122	z
24	CAN (cancel)	57	9	90	Z	123	{
25	EM (end of medium)	58	:	91	[	124	|
26	SUB (substitute)	59	;	92	\	125	}
27	ESC (escape)	60	<	93	]	126	~
28	FS (file separator)	61	=	94	^	127	DEL
29	GS (group separator)	62	>	95	_		
30	RS (record separator)	63	?	96	`		
31	US (unit separator)	64	@	97	a		
32	SPACE	65	A	98	b		

FASTQ

- Simple format for raw unaligned sequencing reads
- Extension to the FASTA file format
- Sequence and an associated per base quality score

Read 1

```
@ERR007731.739 IL16_2979:6:1:9:1684/1
CTTGACGACTTGAAAAATGACGAAATCACTAAAAAACGTGAAAAATGAGAAATG...
+
BBCBCBBBBBBBABBABBBBBBBABBBBBBBBBBBBBBABAAAABBBBB=@>BB...
```

Read name ← (first line)

Sequence ← (second line)

Base qualities ← (fourth line)

Read 2

```
@ERR007731.740 IL16_2979:6:1:9:1419/1
AAAAAAAAAGATGTCATCAGCACATCAGAAAAGAAGGCAACTTTAAAACTTTTC...
+
BBABBBABABAABABABBABBBAAA>@B@BBAA@4AAA>.>BAA@779:AAA@A...
```

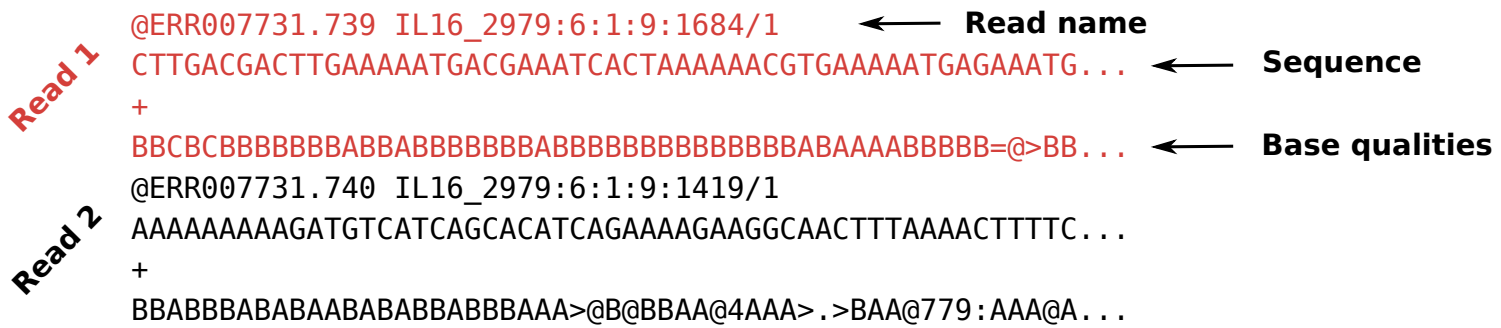
- Quality encoded in ASCII characters with decimal codes 33-126
  - ASCII code of "A" is 65, the corresponding quality is Q$= 65 - 33 = 32$
  - Phred quality score: $P = 10^{-Q/10}$

```
perl -e 'printf "%d\n",ord("A")-33;'
```

- Beware: multiple quality scores were in use!
  - Sanger, Solexa, Illumina 1.3+
- Paired-end sequencing produces two FASTQ files

FASTQ

- Simple format for raw unaligned sequencing reads
- Extension to the FASTA file format
- Sequence and an associated per base quality score



- Quality encoded in ASCII characters with decimal codes 33-126
  - ASCII code of "A" is 65, the corresponding quality is Q$= 65 - 33 = 32$
  - Phred quality score: $P = 10^{-Q/10}$

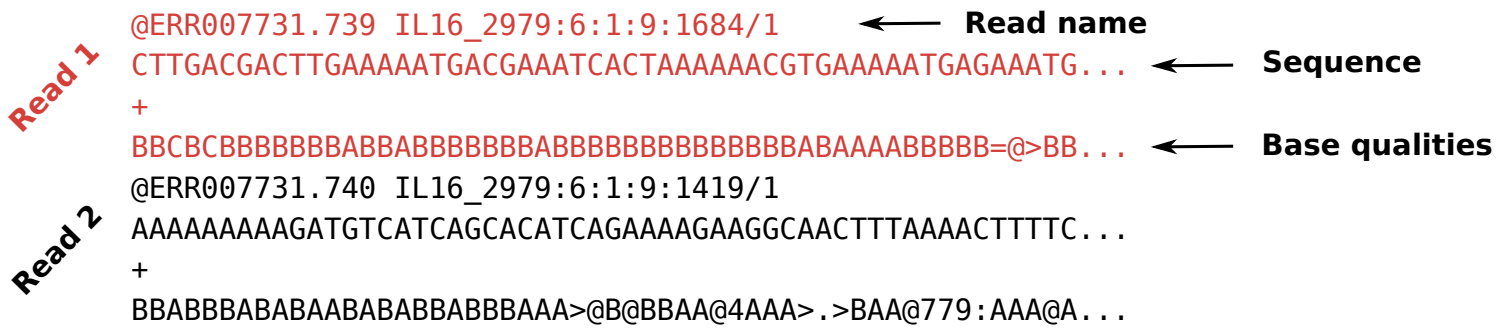
```
perl -e 'printf "%d\n",ord("A")-33;'
```

- Beware: multiple quality scores were in use!
  - Sanger, Solexa, Illumina 1.3+
- Paired-end sequencing produces two FASTQ files

Q: What is the probability of a sequencing error if the quality is "?" (ASCII code 63)

FASTQ

- Simple format for raw unaligned sequencing reads
- Extension to the FASTA file format
- Sequence and an associated per base quality score



- Quality encoded in ASCII characters with decimal codes 33-126
  - ASCII code of "A" is 65, the corresponding quality is Q$= 65 - 33 = 32$
  - Phred quality score: $P = 10^{-Q/10}$

```
perl -e 'printf "%d\n",ord("A")-33;'
```

- Beware: multiple quality scores were in use!
  - Sanger, Solexa, Illumina 1.3+
- Paired-end sequencing produces two FASTQ files

Q: What is the probability of a sequencing error if the quality is "?" (ASCII code 63)

A: Q=30, one error in 1000 bases

SAM / BAM

SAM (Sequence Alignment/Map) format

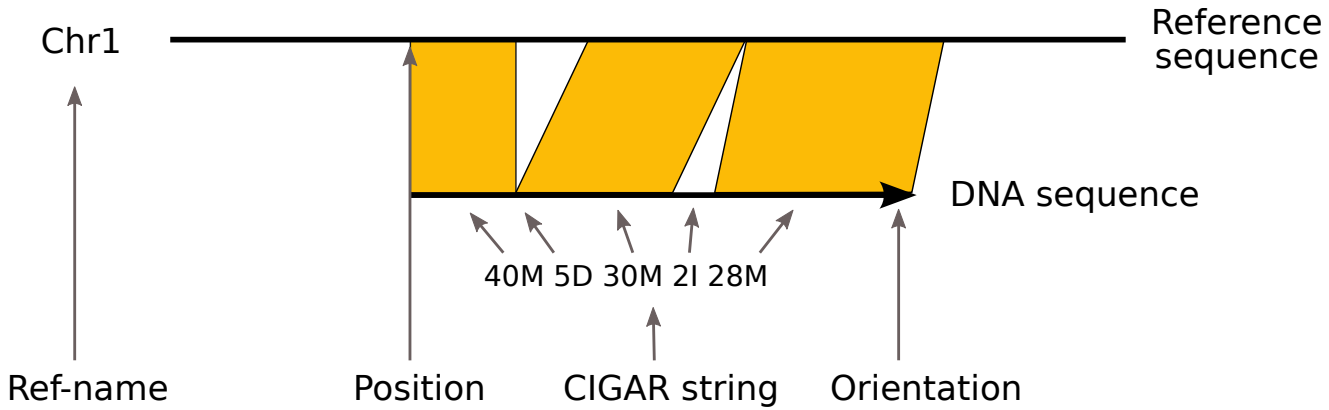
- Single unified format for storing read alignments to a reference genome
- Developed by the 1000 Genomes Project group in 2009

SAM / BAM

SAM (Sequence Alignment/Map) format

- Single unified format for storing read alignments to a reference genome
- Developed by the 1000 Genomes Project group in 2009

- One record (a single DNA fragment alignment) per line describing alignment between fragment and reference
- 11 fixed columns + optional key:type:value tuples

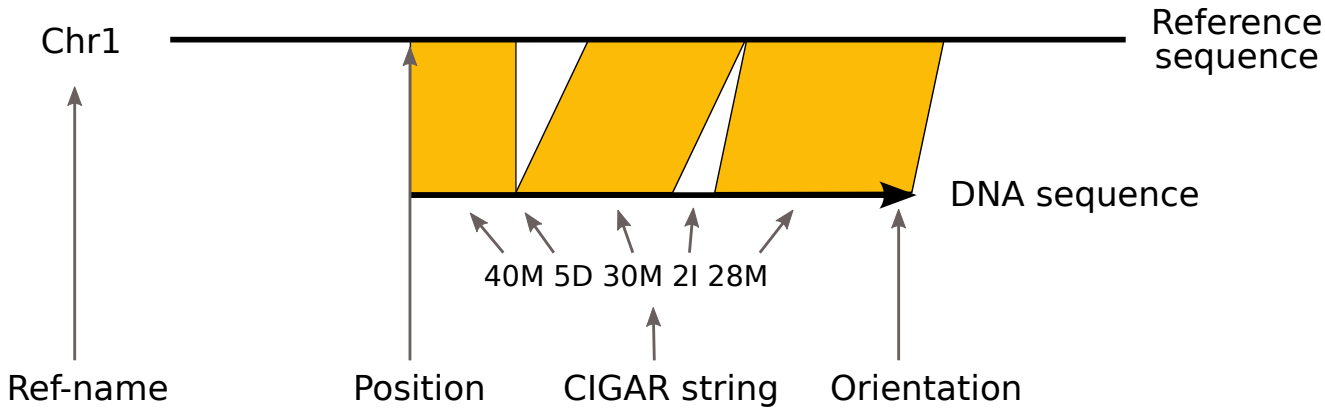


SAM / BAM

SAM (Sequence Alignment/Map) format

- Single unified format for storing read alignments to a reference genome
- Developed by the 1000 Genomes Project group in 2009

- One record (a single DNA fragment alignment) per line describing alignment between fragment and reference
- 11 fixed columns + optional key:type:value tuples



Note that BAM can contain

- unmapped reads
- multiple alignments of the same read
- supplementary (chimeric) reads

SAM

```
$ samtools view -h file.bam | less

@HD VN:1.0  GO:none SO:coordinate
@SQ SN:1    LN:249250621    UR:hs37d5.fa.gz AS:NCBI37   M5:1b22b98cdeb4a9304cb5d48026a85128 SP:Human
@SQ SN:2    LN:243199373    UR:hs37d5.fa.gz AS:NCBI37   M5:a0d9851da00400dec1098a9255ac712e SP:Human
@RG ID:1    PL:ILLUMINA PU:13350_1  LB:13350_1  SM:13350_1  CN:SC
@PG ID:bwa  PN:bwa  VN:0.7.10-r806    CL:bwa mem hs37d5.fa.gz 13350_1_1.fq 13350_1_1.fq
1:2203:10256:56986  97  1   9998    20   106M45S =  10335    0 \
    CCATAACCCTAACCCTAACCCTAACCATAGCCCTAACCCTAACCCTAACCCTAACCCT[...]CAAACCCACCCCCAAACCCAAAACCTCACCAC \
    FFFFFJJJJJJJJFJJJJFJAJJJJJ-JJAAAJFJJFFJJF<FJJFFJJJJFJJJJFF[...]<---F-----A7-J-<J-A--77AF---J7-- \
    MD:Z:1G24C2A76  PG:Z:MarkDuplicates RG:Z:1  NM:i:3  MQ:i:0  AS:i:94 XS:i:94
```

SAM fields

1	QNAME	Query NAME of the read or the read pair
2	FLAG	Bitwise FLAG (pairing, strand, mate strand, etc.)
3	RNAME	Reference sequence NAME
4	POS	1-Based leftmost POSition of clipped alignment
5	MAPQ	MAPping Quality (Phred-scaled)
6	CIGAR	Extended CIGAR string (operations: MIDNSHPX=)
7	MRNM	Mate Reference NaMe ('=' if same as RNAME)
8	MPOS	1-Based leftmost Mate POSition
9	ISIZE	Inferred Insert SIZE
10	SEQ	Query SEQuence on the same strand as the reference
11	QUAL	Query QUALity (ASCII-33=Phred base quality)
12-	OTHER	Optional fields

CIGAR string

Compact representation of sequence alignment

M	alignment match or mismatch
=	sequence match
X	sequence mismatch
I	insertion to the reference
D	deletion from the reference
S	soft clipping (clipped sequences present in SEQ)
H	hard clipping (clipped sequences NOT present in SEQ)
N	skipped region from the reference
P	padding (silent deletion from padded reference)

Examples:

```
Ref:   ACGTACGTACGTACGT
Read:  ACGT----ACGTACGA
Cigar: 4M 4D 8M

Ref:   ACGT----ACGTACGT
Read:  ACGTACGTACGTACGT
Cigar: 4M 4I 8M

Ref:   ACTCAGTG--GTATCGTTAATTC
Read:  ACGCA-TGCAGTTAGACGTACGT
Cigar: 5M 1D 2M 2I 2M 7S
```

CIGAR string

Compact representation of sequence alignment

M	alignment match or mismatch
=	sequence match
X	sequence mismatch
I	insertion to the reference
D	deletion from the reference
S	soft clipping (clipped sequences present in SEQ)
H	hard clipping (clipped sequences NOT present in SEQ)
N	skipped region from the reference
P	padding (silent deletion from padded reference)

Examples:

```
Ref:   ACGTACGTACGTACGT
Read:  ACGT----ACGTACGA
Cigar: 4M 4D 8M

Ref:   ACGT----ACGTACGT
Read:  ACGTACGTACGTACGT
Cigar: 4M 4I 8M

Ref:   ACTCAGTG--GTATCGTTAATTC
Read:  ACGCA-TGCAGTTAGACGTACGT
Cigar: 5M 1D 2M 2I 2M 7S

Ref:   TGTCGTCACGCATG---CAGTTTTTTTAAAA
Read:  ACGTACGAAGCATGCGGCAGTACGACGTTCG
Cigar: ???
```

CIGAR string

Compact representation of sequence alignment

M	alignment match or mismatch
=	sequence match
X	sequence mismatch
I	insertion to the reference
D	deletion from the reference
S	soft clipping (clipped sequences present in SEQ)
H	hard clipping (clipped sequences NOT present in SEQ)
N	skipped region from the reference
P	padding (silent deletion from padded reference)

Examples:

```
Ref:   ACGTACGTACGTACGT
Read:  ACGT----ACGTACGA
Cigar: 4M 4D 8M

Ref:   ACGT----ACGTACGT
Read:  ACGTACGTACGTACGT
Cigar: 4M 4I 8M

Ref:   ACTCAGTG--GTATCGTTAATTC
Read:  ACGCA-TGCAGTTAGACGTACGT
Cigar: 5M 1D 2M 2I 2M 7S

Ref:   TGTCGTCACGCATG---CAGTTTTTTTAAAA
Read:  ACGTACGAAGCATGCGGCAGTACGACGTTCG
Cigar: 7H 7M 3I 4M 7S
```

Flags

Hex	Dec	Flag	Description
0x1	1	PAIRED	paired-end (or multiple-segment) sequencing technology
0x2	2	PROPER_PAIR	each segment properly aligned according to the aligner
0x4	4	UNMAP	segment unmapped
0x8	8	MUNMAP	next segment in the template unmapped
0x10	16	REVERSE	SEQ is reverse complemented
0x20	32	MREVERSE	SEQ of the next segment in the template is reversed
0x40	64	READ1	the first segment in the template
0x80	128	READ2	the last segment in the template
0x100	256	SECONDARY	secondary alignment
0x200	512	QCFAIL	not passing quality controls
0x400	1024	DUP	PCR or optical duplicate
0x800	2048	SUPPLEMENTARY	supplementary alignment

Bit operations made easy

- python

  0x1 | 0x2 | 0x20 | 0x80 .. 163

  bin(163) .. 10100011

- samtools flags

  0xa3 163 PAIRED,PROPER_PAIR,MREVERSE,READ2

Flags

Hex	Dec	Flag	Description
0x1	1	PAIRED	paired-end (or multiple-segment) sequencing technology
0x2	2	PROPER_PAIR	each segment properly aligned according to the aligner
0x4	4	UNMAP	segment unmapped
0x8	8	MUNMAP	next segment in the template unmapped
0x10	16	REVERSE	SEQ is reverse complemented
0x20	32	MREVERSE	SEQ of the next segment in the template is reversed
0x40	64	READ1	the first segment in the template
0x80	128	READ2	the last segment in the template
0x100	256	SECONDARY	secondary alignment
0x200	512	QCFAIL	not passing quality controls
0x400	1024	DUP	PCR or optical duplicate
0x800	2048	SUPPLEMENTARY	supplementary alignment

Bit operations made easy

- python
  0x1 | 0x2 | 0x20 | 0x80 .. 163
  bin(163) .. 10100011
- samtools flags
  0xa3 163 PAIRED,PROPER_PAIR,MREVERSE,READ2

Q: What is the flag of the first read if both reads are mapped?

Flags

Hex	Dec	Flag	Description
0x1	1	PAIRED	paired-end (or multiple-segment) sequencing technology
0x2	2	PROPER_PAIR	each segment properly aligned according to the aligner
0x4	4	UNMAP	segment unmapped
0x8	8	MUNMAP	next segment in the template unmapped
0x10	16	REVERSE	SEQ is reverse complemented
0x20	32	MREVERSE	SEQ of the next segment in the template is reversed
0x40	64	READ1	the first segment in the template
0x80	128	READ2	the last segment in the template
0x100	256	SECONDARY	secondary alignment
0x200	512	QCFAIL	not passing quality controls
0x400	1024	DUP	PCR or optical duplicate
0x800	2048	SUPPLEMENTARY	supplementary alignment

Bit operations made easy

- python
  0x1 | 0x2 | 0x20 | 0x80 .. 163
  bin(163) .. 10100011
- samtools flags
  0xa3 163 PAIRED,PROPER_PAIR,MREVERSE,READ2

Q: What is the flag of the first read if both reads are mapped?

A: PAIRED,PROPER_PAIR,READ1 ..

Flags

Hex	Dec	Flag	Description
0x1	1	PAIRED	paired-end (or multiple-segment) sequencing technology
0x2	2	PROPER_PAIR	each segment properly aligned according to the aligner
0x4	4	UNMAP	segment unmapped
0x8	8	MUNMAP	next segment in the template unmapped
0x10	16	REVERSE	SEQ is reverse complemented
0x20	32	MREVERSE	SEQ of the next segment in the template is reversed
0x40	64	READ1	the first segment in the template
0x80	128	READ2	the last segment in the template
0x100	256	SECONDARY	secondary alignment
0x200	512	QCFAIL	not passing quality controls
0x400	1024	DUP	PCR or optical duplicate
0x800	2048	SUPPLEMENTARY	supplementary alignment

Bit operations made easy

- python
  0x1 | 0x2 | 0x20 | 0x80 .. 163
  bin(163) .. 10100011
- samtools flags
  0xa3 163 PAIRED,PROPER_PAIR,MREVERSE,READ2

Q: What is the flag of the first read if both reads are mapped?

A: PAIRED,PROPER_PAIR,READ1 .. 1+2+64=67

Flags

Hex	Dec	Flag	Description
0x1	1	PAIRED	paired-end (or multiple-segment) sequencing technology
0x2	2	PROPER_PAIR	each segment properly aligned according to the aligner
0x4	4	UNMAP	segment unmapped
0x8	8	MUNMAP	next segment in the template unmapped
0x10	16	REVERSE	SEQ is reverse complemented
0x20	32	MREVERSE	SEQ of the next segment in the template is reversed
0x40	64	READ1	the first segment in the template
0x80	128	READ2	the last segment in the template
0x100	256	SECONDARY	secondary alignment
0x200	512	QCFAIL	not passing quality controls
0x400	1024	DUP	PCR or optical duplicate
0x800	2048	SUPPLEMENTARY	supplementary alignment

Bit operations made easy

- python
  0x1 | 0x2 | 0x20 | 0x80 .. 163
  bin(163) .. 10100011
- samtools flags
  0xa3 163 PAIRED,PROPER_PAIR,MREVERSE,READ2

Q: What is the meaning of the flag 65?

Flags

Hex	Dec	Flag	Description
0x1	1	PAIRED	paired-end (or multiple-segment) sequencing technology
0x2	2	PROPER_PAIR	each segment properly aligned according to the aligner
0x4	4	UNMAP	segment unmapped
0x8	8	MUNMAP	next segment in the template unmapped
0x10	16	REVERSE	SEQ is reverse complemented
0x20	32	MREVERSE	SEQ of the next segment in the template is reversed
0x40	64	READ1	the first segment in the template
0x80	128	READ2	the last segment in the template
0x100	256	SECONDARY	secondary alignment
0x200	512	QCFAIL	not passing quality controls
0x400	1024	DUP	PCR or optical duplicate
0x800	2048	SUPPLEMENTARY	supplementary alignment

Bit operations made easy

- python
  0x1 | 0x2 | 0x20 | 0x80 .. 163
  bin(163) .. 10100011
- samtools flags
  0xa3 163 PAIRED,PROPER_PAIR,MREVERSE,READ2

Q: What is the meaning of the flag 65?

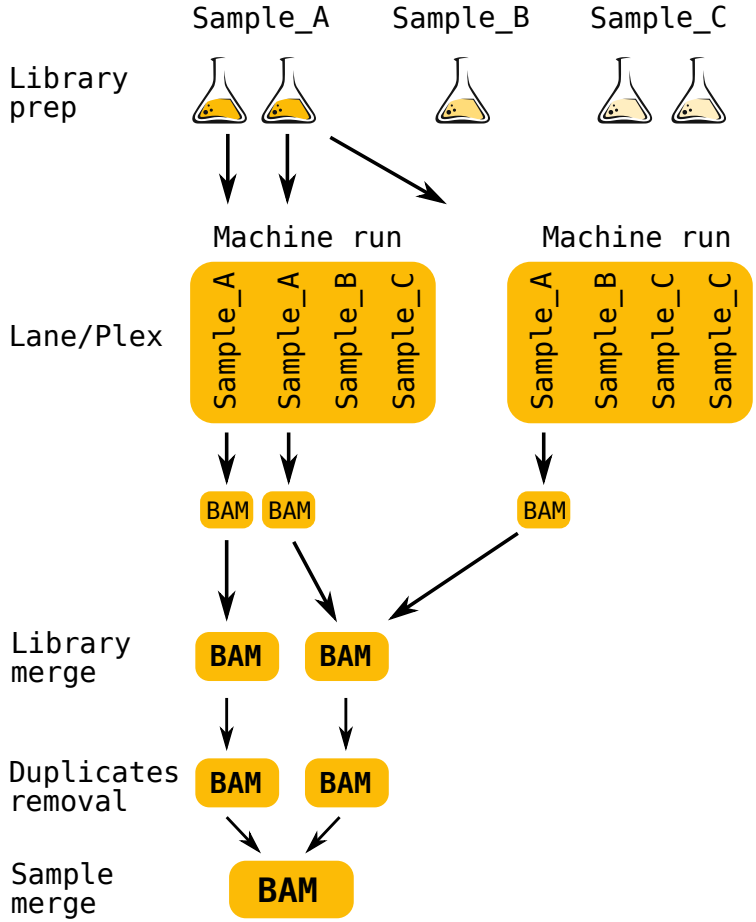
A: 65=1+64 .. PAIRED,READ1

Optional tags

Each lane has a unique RG tag that contains meta-data for the lane

RG tags

- ID: SRR/ERR number
- PL: Sequencing platform
- PU: Run name
- LB: Library name
- PI: Insert fragment size
- SM: Individual
- CN: Sequencing center



BAM

BAM (Binary Alignment/Map) format

- Binary version of SAM
- Developed for fast processing and random access
  - BGZF (Block GZIP) compression for indexing

Key features

- Can store alignments from most mappers
- Supports multiple sequencing technologies
- Supports indexing for quick retrieval/viewing
- Compact size (e.g. 112Gbp Illumina = 116GB disk space)
- Reads can be grouped into logical groups e.g. lanes, libraries, samples
- Widely supported by variant calling packages and viewers

SAM/BAM tools

Several tools and programming APIs for interacting with SAM/BAM files

Samtools - Wellcome Sanger Institute (http://www.htslib.org)

- convert between SAM, BAM, CRAM
- sort, index
- flagstat - summary of the mapping flags
- merge multiple BAM files
- rmdup - remove PCR duplicates from the library preparation

Picard tools - Broad Institute (https://www.broadinstitute.org/gatk/)

- MarkDuplicates, CollectAlignmentSummaryMetrics, CreateSequenceDictionary, SamToFastq, MeanQualityByCycle, FixMateInformation etc.

Others

- Bio-SamTool - Perl (http://search.cpan.org/~lds/Bio-SamTools/)
- Pysam - Python (https://github.com/pysam-developers/pysam)
- R - Bioconductor/Rsamtools

BAM Visualisation

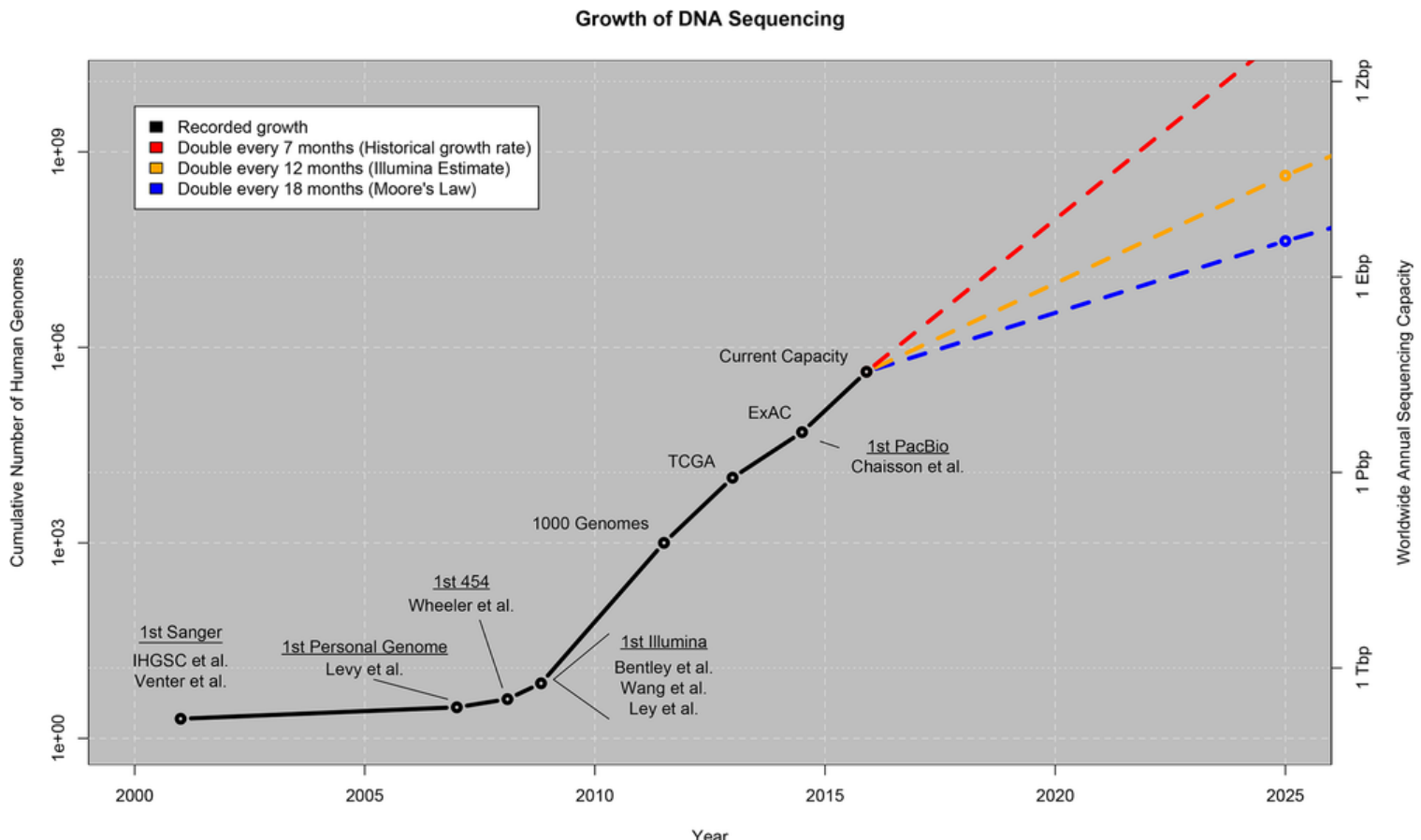
- IGV: http://www.broadinstitute.org/igv/
- BamView, LookSeq, Gap5, Tablet, Ensembl, UCSC, Bambino, Biodalliance...

Reference based Compression

BAM files are too large

- ~1.5-2 bytes per base pair

Increases in disk capacity are being far outstripped by sequencing technologies



Zachary D. Stephens, *et al*, Big Data: Astronomical or Genomical? DOI: 10.1371/journal.pbio.1002195

Reference based Compression

BAM files are too large

- ~1.5-2 bytes per base pair

Increases in disk capacity are being far outstripped by sequencing technologies
BAM stores all of the data

- Every read base
- Every base quality
- Using a single conventional compression technique for all types of data

```
Reference sequence: ACGTACGTACGTACGTACGTACGTACGTACGTAC
read 1:             ACGTACGTACGTACGTACGTGC
read 2:                TACGTACGCACGTACGTGCGTA
read 3:                  CGTACGCACGTACGTACGTACG
read 4:                    TACGTACGTACGTGCGTACGTA
read 5:                      CGCACGTACGTACGTACGTACG
read 6:                            TACGTGCGTACGTACGTAC
```

Reference based Compression

BAM files are too large

- ~1.5-2 bytes per base pair

Increases in disk capacity are being far outstripped by sequencing technologies
BAM stores all of the data

- Every read base
- Every base quality
- Using a single conventional compression technique for all types of data

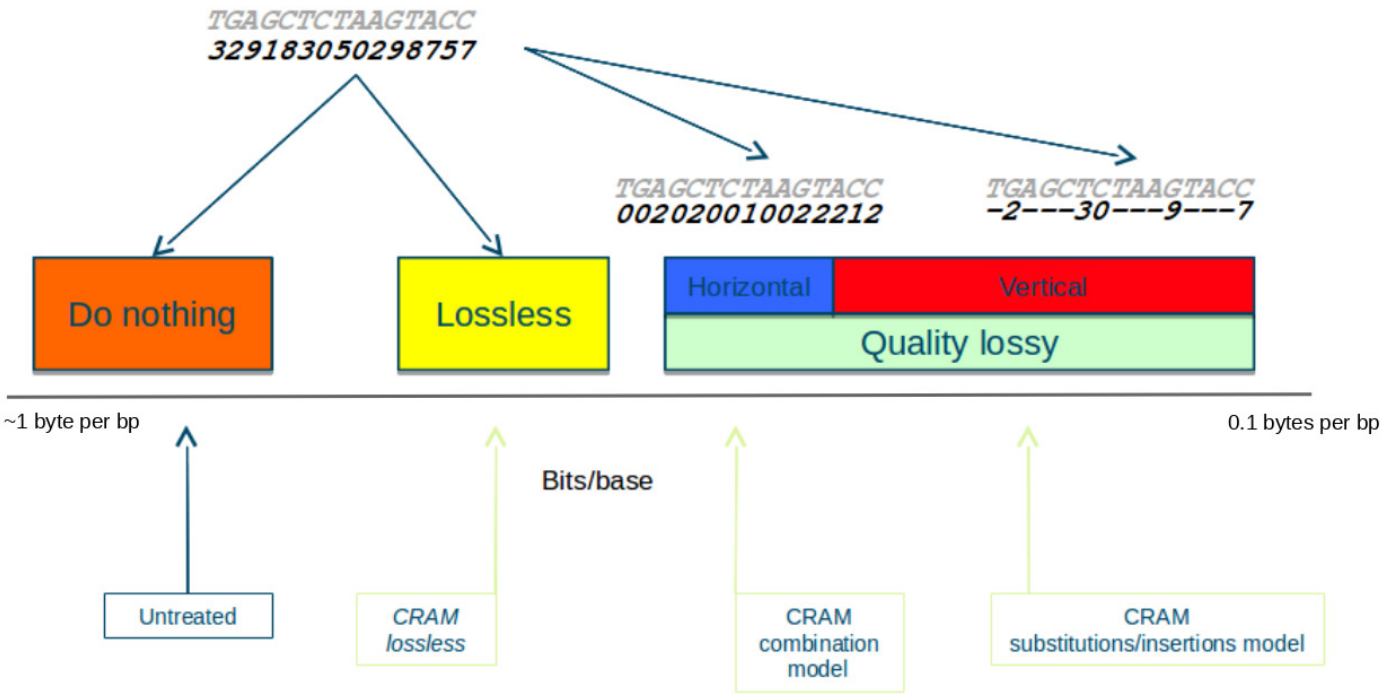
```
Reference sequence: ACGTACGTACGTACGTACGTACGTACGTAC
read 1:                    ....................G.
read 2:                       ........C............
read 3:                         ......C..............
read 4:                           ............G........
read 5:                             ..C..................
read 6:                                   .....G............
```

CRAM

Three important concepts

- Reference based compression
- Controlled loss of quality information
- Different compression methods to suit the type of data, e.g. base qualities vs. metadata vs. extra tags

In lossless mode: 60% of BAM size

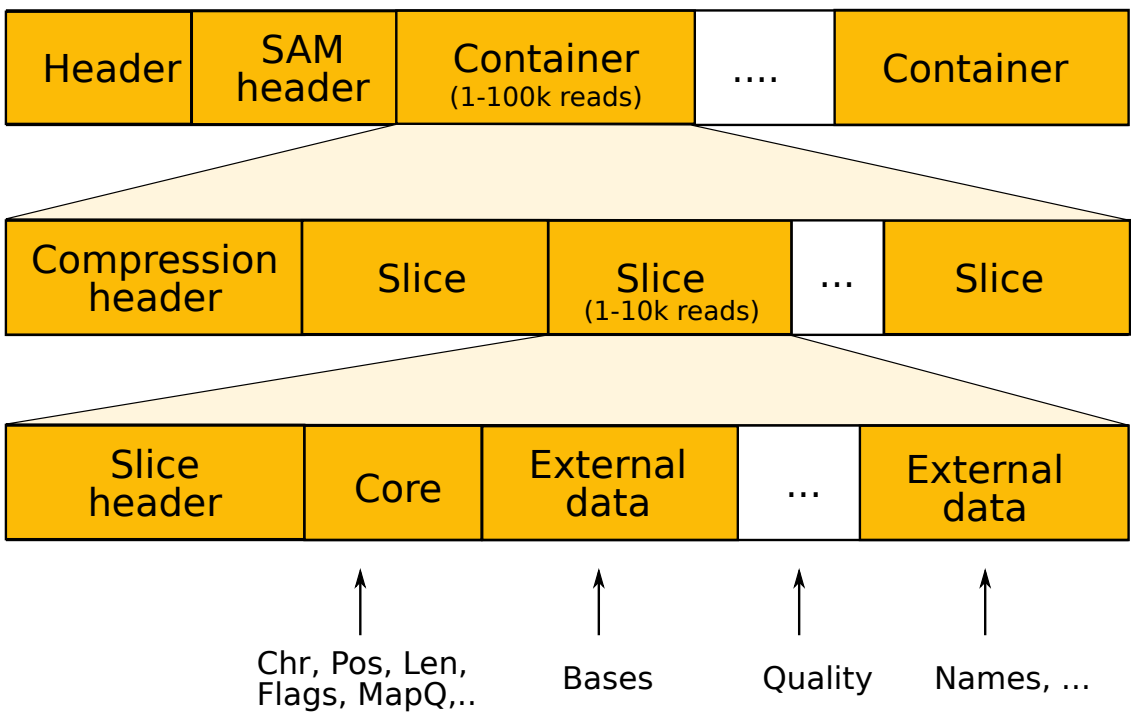


CRAM

Three important concepts

- Reference based compression
- Controlled loss of quality information
- Different compression methods to suit the type of data, e.g. base qualities vs. metadata vs. extra tags

In lossless mode: 60% of BAM size



CRAM

Three important concepts

- Reference based compression
- Controlled loss of quality information
- Different compression methods to suit the type of data, e.g. base qualities vs. metadata vs. extra tags

In lossless mode: 60% of BAM size

Support for CRAM

- added to Samtools/HTSlib in 2014, to GATK in 2015
- CRAM is now mature and used in production pipelines
  - all sequencing data by default in CRAM format
  - 40% disk space saving immediately