Overview of Sequence Data Formats

Hongen Zhang

Abstract

Next-generation sequencing experiment can generate billions of short reads for each sample and processing of the raw reads will add more information. Various file formats have been introduced/developed in order to store and manipulate this information. This chapter presents an overview of the file formats including FASTQ, FASTA, SAM/BAM, GFF/GTF, BED, and VCF that are commonly used in analysis of next-generation sequencing data.

Key words Sequencing data file format, Next-generation sequencing, Sequencing data, FASTQ, FASTA, SAM/BAM, GFF/GTF, BED, VCF

1 Introduction

Next-generation sequencing (NGS) refers to high-throughput technologies for large scale DNA sequencing (such as whole genome sequencing, whole-exome sequencing, RNA-seq, miRNA-seq, ChIP-seq, and DNA Methylation) and could be conducted with different platforms, for example, Illumina(Solexa), Roche 454, Ion Torrent, and SOLiD [1–5]. The main output of a next-generation sequencing experiment is short reads (short DNA sequences of <200 bases). In general, one NGS experiment can generate billions of short reads for each sample and stores the short reads in one or multiple files. While these raw sequencing data may contain some meta-data, genomic position information is not included in each short read. Therefore a series of procedures must be applied to parse and manipulate the raw reads in downstream analysis in order to uncovering interested genomic structures and variations [6–11]. In addition to short sequences, downstream analysis will add more information such as description and annotations, which may increase the file size from few gigabytes to hundreds gigabytes. To efficiently store, view, and manipulate such information, various file formats have been introduced/developed [12–16].

This chapter gives an overview of file formats commonly used in storage and analysis of next-generation sequencing data.

2 Results

2.1 FASTQ: Format of Raw Short Read Files

FASTQ format was originally invented by Jim Mullikin at the Wellcome Trust Sanger Institute to storing both nucleotide sequence and its corresponding Phred quality scores [17, 18] and currently it has became a common file format for sharing sequencing read data [13–16].

FASTQ files are plain text file with extension ".fq" or ".fastq" and could be viewed directly from command line on computers with Unix/Linux operating system. In FASTQ file, each sequence (short read of NGS) is defined by four lines of text:

The first line starts with a "@" character and is followed by a sequence identifier and an *optional* description.

The second line is the raw sequence letters: A, T, G, C, and N (unknown).

The third line begins with a "+" symbol and is optionally followed by the same sequence identifier (and any description) again. The "+" sign serves as a marker indicating the end of sequence.

The fourth line is the quality values for sequence in the second line, and must contain the same number of symbols as letters in the sequence.

Following is an example of a single sequence from Illumina sequence system:

```
@HWI-ST193:542:C2H0GACXX:8:1101:4404:2179 1:Y:0:ACACGA
ATGCNTTTTATAATCAAAAGCGAAGACCTAGCAGGAGGTTAAAAACCTTT
+
<<<<#2<@@5:9@44:@@?4(-8@(<9@<<658.==5=0<>??????9??
```

The first line is sequencer identifier and description items separated by ":" or " ". The descriptions may vary for different sequencers or processing pipelines. In this example, each item identifies as following:

HWI-ST193: sequencer name (should be unique, usually from manufacturer).

542: run id.

C2H0GACXX: flowcell id.

8: flowcell lane.

1101: tile number within the flowcell lane.

4404: *x*-coordinate of the cluster within the tile.

2179: *y*-coordinate of the cluster within the tile.

1: the member of a pair (1 for single read, and 2 for *paired-end and mate-pair reads*).

Y: *filter status* (Y if the read is filtered, N otherwise).

0: status of control bits (0 for none of the control bits are on, otherwise it is an even number).

ACACGA: index sequence.

The second line of above example is the sequence contents and the fifth base is unable to determine so an N is used. The third line has only a "+" sign and has no more information added.

The fourth line is the quality scores (Phred quality scores) for corresponding bases in the second line.

The Phred quality scores stand for sequencing quality for each base and the higher the Phred score the more accurate the base to be determined. Phred scores range from 0 to 93 and are encoded with following characters in the order from left to right:

```
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTU
VWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
```

The "!" represents the lowest Phred score and the "~" is for the highest one.

Phred score encoding varies for different sequencers and cannot always be directly compared among them until they have been converted to probabilities through a transformation defined by the sequencer manufacturer.

FASTQ format has no restriction on total number of single sequences in a FASTQ file but since the number of short reads from next-generation sequencing experiment is very huge, for convenience, each sample will use several FASTQ files to hold all reads. In most case, the FASTQ files are compressed with software to GNU zip format (with .gz file extension) to reduce file size.

Due to the high volume of sequence reads in a FASTQ file and no genomic position information for each reads, it is not practical to view or check a single reads manually. During the analysis, the quality analysis of short reads is usually performed with computer software. The most commonly used one is FASTQC developed by Simon Andrews at Babraham Institute [19] which takes FASTQ files as input and generates summary graphs and tables for a quick overview of the raw sequence read quality.

2.2 FASTA: Format of Reference Genome Files

FASTA format is also text based format and commonly uses file extension ".fa" or ".fasta". FASTA has been standard format for nucleotide sequence since the first generation sequencing [20, 21]. For next-generation sequencing, it is the standard format for reference genome sequences used by mapping/alignment software tools [22–25].

FASTA format is very simple. It starts with a header line followed by lines of sequence. The header line begins with a ">" sign for sequence identifier and may contain optional descriptive information. Sequence lines consist of characters representing nucleotide bases in the sequence and are usually no more than 80 characters per line but have no limitation for total number of lines. Same as sequence in FASTQ files, each nucleotide base is encoded as a single character (one of A, T, G, C, and N if undetermined, with case insensitive).

FASTA format can be used to display sequence of a single gene, a single chromosome, or multiple ones. Following is the header line and first two lines of human p53 gene (NC_000017.11) sequence in FASTA format (http://www.ncbi.nlm.nih.gov/nuccore/NC_000017.11?report=fasta&from=7668402&to=7687550&strand=true):

```
>gi|568815581:c7687550-7668402 Homo sapiens chromosome
17, GRCh38 Primary Assembly
GATGGGATTGGGGTTTTCCCCTCCCATGTGCTCAAGACTGGCGCTAAAAGT TTT
GAGCTTCTCAAAAG TC
TAGAGCCACCGTCCAGGGAGCAGGTAGCTGCTGGGCTCCGGGGACACTTTG CGT
TCGGGCTGGGAGCGTG
```

For a single chromosome, the sequence identifier can be only the chromosome name, for example, below is partial sequence of human chromosome 1 in FASTA format:

```
>chr1
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN NNNNNNNNNNNNNNNNNN
TAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTA
ACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAAC
CCTAACCCAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCC
```

A small FASTA file could be viewed with text editing tools and command line of UNIX/LINUX platforms. For big FASTA files such as a reference genome in FASTA format, Java based stand along software, Integrative Genomics Viewer (IGV) developed by Broad Institute in Boston, is a commonly used tool to navigate genome sequence [26, 27] and accompanying features aligned to it.

2.3 BAM/SAM: Format of Alignment (Mapping) Data Files

BAM/SAM file has import role in analysis of next-generation sequencing data because the raw sequence reads from sequencers have no genomic position information associated and they must be mapped/aligned to known reference genome. The mapping/alignment outputs are BAM/SAM files that will serve as inputs for various downstream analysis such as feature counts and variant calling.

SAM stands for *S*equence *A*lignment/*M*ap format, and is a generic alignment format for storing read alignments against reference sequences, supporting short and long reads (up to 128 Mbp) produced by different sequencing platforms [14]. A SAM file often uses file extension ".sam" and is a tab-delimited text file that contains sequence alignment data. BAM is the binary version of a SAM file and usually has a file extension ".bam". Both BAM file and SAM file contain same information and with SAMtools software, a BAM file can be easily converted to SAM file.

A BAM/SAM file has two sections: header section and alignment section. Details of BAM/SAM format is described on the SAMtools website (http://samtools.sourceforge.net) and here is a short description of header section and alignment section.

2.3.1 Header Section

The header section is optional for BAM/SAM file. If present, it provides generic information for BAM/SAM file. Header section can have multiple lines and each line must start with a "@" symbol. The information can be divided into four types: @HD, @SQ, @RG, and @PG and contents of each line are tab-delimited TAG:Value field(s). An optional comment line starting with @CO may exist at the end of header section.

@HD Line

The header line. If header section exists in a BAM/SAM file, the first line must start with @HD followed by required and optional field(s) including:

- VN: format version. This field is required.
- SO: sorting order of alignments. Valid values include: unknown (default), unsorted, queryname, and coordinate.
- GO: group order (full sorting is not imposed in a group). Valid values are: none, query, or reference.

@SQ Lines

@SQ line serves as reference sequence dictionary. A BAM/SAM file usually has multiple @SQ lines and the order of @SQ lines defines the alignment sorting order. @SQ line has following fields:

- SN: reference sequence name. This field is required and each @SQ line must have a unique SN tag.
- LN: reference sequence length with range of 1 to $2^{31} - 1$. This is a required field.
- AS: genome assembly identifier.
- M5: MD5 checksum of the sequence in the uppercase, excluding spaces but including pads (as "*"s).
- SP: species.
- UR: URI of the sequence. This value may start with one of the standard protocols, e.g., http: or ftp:. If it does not start with one of these protocols, it is assumed to be a file-system path.

@RG Lines

Read group information is listed in @RG line(s). Unordered multiple @RG lines are allowed. @RG line has also required and optional fields:

- ID: read group identifier. This is a required field and each @RG line must have a unique ID. The value of ID is used in the RG tags of alignment records. It must be unique among all read groups in header section. Read group IDs may be modified when merging SAM files in order to handle collisions.
- CN: name of sequencing center producing the read.
- DS: description.
- DT: the date the run was produced (using ISO8601 date or date/time format).
- FO: flow order.
- KS: the array of nucleotide bases that correspond to the key sequence of each read.
- LB: library.
- PG: programs used for processing the read group.
- PI: predicted median insert size.
- PL: platform/technology used to produce the reads. Valid values: CAPILLARY, LS454, ILLUMINA, SOLID, HELICOS, IONTORRENT, and PACBIO.
- PM: Platform model. Free-form text providing further details of the platform/technology used.
- PU: platform unit (e.g., flowcell-barcode lane for Illumina or slide for SOLiD). It must be a unique identifier.
- SM: sample name. Use pool name where a pool is being sequenced.

@PG Lines

These lines describe the program used to generate BAM/SAM file with following fields:

- ID: program record identifier. Each @PG line must have a unique ID. The value of ID is used in the alignment PG tag and PP tags of other @PG lines.
- PN: program name.
- CL: command line.
- PP: previous @PG-ID. It must match another @PG header's ID tag.
- DS: description of the program.
- VN: program version.

@CO Line

One-line text comment. Unordered multiple @CO lines are allowed.

Following is an example of a BAM file header viewed with SAMtools software:

```
@HD VN:1.4 GO:none SO:coordinate
@SQ SN:chr1 LN:249250621
```

```
@SQ SN:chr2 LN:243199373
@SQ SN:chr3 LN:198022430
@RG ID:1048 PL:Illumina LB:4949 SM:PAVECB_Tumor
@RG ID:1054 PL:Illumina LB:4949 SM:PAVECB_Tumor
@RG ID:26343 PL:Illumina LB:5144 SM:PAVECB_Tumor
@PG ID:GATK IndelRealigner VN:3.1-1-g07a4bf8
CL:knownAlleles=[(RodBinding name=knownAlleles
source=/data/CCRBioinfo/public/GATK/bundle/2.3/hg19/Mills_and_1000G_gold_
standard.indels.hg19.vcf)]
targetIntervals=bam/SAMPLE/DNA/SAMPLE.intervals
LODThresholdForCleaning=5.0  consensusDeterminationModel=USE_READS  entropy
Threshold=0.15
maxReadsInMemory=150000maxIsizeForMovement=3000
maxPositionalMoveAllowed=200maxConsensuses=30 maxReadsForConsensuses=120
maxReadsForRealignment=20000  noOriginalAlignmentTags=false  nWayOut=null
generate_nWayOut_md5s=false check_early=false noPGTag= false
keepPGTags=false indelsFileForDebugging=null
statisticsFileForDebugging=null SNPsFileForDebugging=null
```

This example shows the BAM file version is 1.4 and alignments are sorted by genomic coordinates. Reference sequences used are chromosome 1 ~ 3 and sequence length are listed for each chromosome. The platform is Illumina sequencer and sample name is PAVECB_Tumor. Software used to generate this BAM file is GATK IndelRealigner, version 3.1-1-g07a4bf8, and a copy of the command line text is included.

2.3.2 Alignment Section

Alignment section in SAM/BAM file contains sequences with genomic position and other descriptive information. In BAM/SAM format, each single sequence (short reads from FASTQ file) and its associated information are presented as one line text and each line consists of multiple tab-delimited text fields. There are 11 mandatory fields for each line and these fields are always in the same order. Mandatory fields provide:

- QNAME: query name, same as the sequence identifier in FASTQ file.
- FLAG: bitwise flag of two byte length to indicate the read property such as the segment mapped or unmapped, passed or not passed quality controls. An easy way to convert the flag to human readable meaning can be found at https://broadinstitute.github.io/picard/explain-flags.html.
- RNAME: reference sequence name, one from SN field defined in @SQ line of header section.
- POS: the leftmost position of the first matching base in reference sequence.

- MAPQ: mapping quality.
- CIGAR: a character string indicating the match status of bases in the short read. For example, a CIGAR string "3M1I3M1D5M" indicates that, in a short read with 13 base long, the first 3 bases match to reference sequence, the fourth base is an insertion (not found in reference sequence), following 3 bases match to reference sequence and 1 base is missing in the read comparing to reference sequence, and the last 5 bases match to reference sequence.
- RNEXT: reference name of next aligned read.
- PNEXT: position of the primary alignment of the next read.
- TLEN: signed observed template length.
- SEQ: sequence of the short read. Same as from FASTQ file.
- QUAL: quality scores for each base in the short read, same as the one in FASTQ file.

One or more optional fields may follow after the mandatory fields. Details can be found in SAM file documentation.

Below is a single line from BAM file viewed with SAMtools software:

```
HWI-ST1108:4:1214:9025:65227#1 1187 chr1 12111 0 101M =
12121 111 CATTGTTCATCTTCT
GGCCCCTGTTGTCTGCATGTAACTTAATACCACAACCAGGCATAGGGGAAA
GATTGGAGGAAAGATGAGTGACAGCATCAACTTCT
CCCFFFFFHHHHHJJJJJIJJJJJJJJJJJJJJJIJJJJJIJJJIJJJIJJJIJ
JJIJIJJJJFHJJIHHHHHFFFFDEEEEDDCDDDDDDDDDDDDDDDA     MC:
Z:101M RG:Z:26316 NM:i:1 MQ:i:0 AS:i:96 XS:i:96
```

In addition to the SAMtools, the software of Integrative Genomics Viewer [26, 27] is commonly used to view BAM/SAM file contents against a reference genome.

2.4 GFF/GTF and BED: Formats for Genomic Feature Annotation Files

Uncovering the genetic structures/variations and their biological meaning is one of final goals of next-generation sequencing data analysis. To meet this goal, next-generation sequencing data is often summarized at different annotation levels. Currently, the GFF/GTF and BED format are among of most commonly used data formats to provide genomic annotation information for next-generation sequencing data analysis.

The GFF stands for *G*eneral *F*eature *F*ormat. GFF format was initially proposed by Richard Durbin and David Haussler in Wellcome Trust Sanger Institute in England and its version 2 was introduced in 1998. The GTF (*G*eneral *T*ransfer *F*ormat) format is identical to GFF version 2 [28, 29]. GFF file uses file extension ".gff" and GTF files have extension ".gtf".

In GFF/GTF file, each feature is represented with one line text and each line has nine columns (fields) of data values, plus optional track definition lines. Data fields must be tab-separated and each field must contain a value. "empty" columns should be denoted with a ".". Data fields in each line must be in the same order as <seqname> <source> <feature> <start> <end> <score> <strand> <frame> [attributes] [comments] (<> denotes mandatory fields and [] is optional).

Following are field names and descriptions of values for relevant fields:

- seqname: The name of the sequence. Having an explicit sequence name allows a feature file to be prepared for a data set of multiple sequences. Normally the seqname will be the identifier of the sequence in an accompanying fasta format file. An alternative is that <seqname> is the identifier for a sequence in a public database, such as an EMBL/Genbank/DDBJ accession number. Which is the case, and which file or database to use, should be explained in accompanying information.
- source: name of the program that generated this feature, or the data source (database or project name), the source of this feature. This field will normally be used to indicate the program making the prediction, or if it comes from public database annotation, or is experimentally verified, etc.
- feature: feature type name, e.g., gene, exon, transcript.
- start: Start position of the feature, with sequence numbering starting at 1.
- end: End position of the feature, with sequence numbering starting at 1.
- score: A floating point value.
- strand: defined as + (forward) or − (reverse).
- frame: One of "0", "1", "2", or ".". "0" indicates that the specified region is in frame, i.e., that its first base corresponds to the first base of a codon. "1" indicates that there is one extra base, i.e., that the second base of the region corresponds to the first base of a codon, and "2" means that the third base of the region is the first base of a codon. If the strand is "−", then the first base of the region is value of <end>, because the corresponding coding region will run from <end> to <start> on the reverse strand. As with <strand>, if the frame is not relevant then set <frame> to ".". It has been pointed out that "phase" might be a better descriptor than "frame" for this field.
- attribute: A semicolon-separated list of tag-value pairs, providing additional information about each feature. From version 2 onwards, the attribute field must have an tag value structure following the syntax used within objects in a .ace file,

flattened onto one line by semicolon separators. Tags must be standard identifiers ([A-Za-z][A-Za-z0-9_]*). Free text values must be quoted with double quotes.

GFF/GTF files may contains comment lines at the beginning and all comment lines must start with "#". For example, the first line of a GFF file may be

```
##gff-version 2
```

which indicates the file uses version 2 of GFF format.

Since GFF/GTF files are text based they can be viewed from command line of Unix/Linux computers. Microsoft Excel and other text edit software can also be used to access small GFF/GTF files. Below is partial content of a GFF/GTF file.

```
#!genome-build GRCh38
#!genome-date 2013-12
#!genome-build-accession NCBI:GCA_000001405.15
#!genebuild-last-updated 2014-08
1 havana    gene   11869   14409   .   +   .   gene_id "ENSG00000223972"
1 havana    exon   11869   14409   .   +   .   gene_id "ENSG00000223972"
1   havana   exon   11869   12227   .   +   .   gene_id "ENSG00000223972"
1   havana   exon   12613   12721   .   +   .   gene_id "ENSG00000223972"
```

In this example, the first four lines are comments showing the genome version is GRCh38 generated in December of 2013 and accession number is NCBI:GCA_000001405.15. All gene information was updated in August, 2014. The following four lines display the information of Ensembl gene ENSG00000223972 in the order of chromosome name, data source name, start and end position, score, strand, frame information, and attribute (here gene id).

BED format is another flexible way to define the features of genome annotation [30, 31]. BED format is also tab-delimited text file and each line represents one feature. A basic BED file has only three required fields in each line, i.e., chrom, chromStart, and chromEnd to define chromosome name, start, and end position of the feature on the chromosome. A detailed BED file will have extra optional fields to describe how the feature will be displayed (specifically, in Web browser). The optional fields are:

- name: defines the name of the BED line. This label is displayed to the left of the BED line in a genome browser window when the track is open to full display mode or directly to the left of the item in pack mode.
- score: a score between 0 and 1000. If the track line *useScore* attribute is set to 1 for this annotation data set, the *score* value will determine the level of gray in which this feature is displayed (higher numbers = darker gray).
- strand: defined as + (forward) or − (reverse).

- thickStart: The starting position at which the feature is drawn thickly (for example, the start codon in gene displays). When there is no thick part, thickStart and thickEnd are usually set to the chromStart position.
- thickEnd: The ending position at which the feature is drawn thickly (for example, the stop codon in gene displays).
- itemRgb: An RGB value of the form R,G,B (e.g., 255,0,0). If the track line *itemRgb* attribute is set to "On", this RBG value will determine the display color of the data contained in this BED line.
- blockCount—The number of blocks (exons) in the BED line.
- blockSizes—A comma-separated list of the block sizes. The number of items in this list should correspond to blockCount.
- blockStarts—A comma-separated list of block starts. All of the blockStart positions should be calculated relative to chromStart. The number of items in this list should correspond to blockCount.

BED file can also contain track line definition to configure the display further, e.g., by grouping features into separate tracks. Track lines should be placed at the beginning of the list of features they are to affect. And they should consist of the word "track" followed by space-separated key=value pairs which include name, description, priority, and useScore of the track.

An example of BED file with track line definition looks like below:

```
track name="ItemRGBDemo" description="Item RGB demonstration" itemRgb="On"
chr7 127471196 127472363 Pos1 0 + 127471196 127472363 255,0,0
chr7 127472363 127473530 Pos2 0 + 127472363 127473530 255,0,0
chr7 127473530 127474697 Pos3 0 + 127473530 127474697 255,0,0
```

2.5 VCF: A Special File Format for Genomic Variations

VCF (*V*ariant *C*all *F*ormat) format was initially introduced by the 1000 Genomes Project to store the most prevalent types of genomic sequence variation, such as SNPs (single nucleotide polymorphism) and small INDELS (insertions/deletions), enriched by annotations [16, 32, 33]. VCF file is text file and contains meta-information (header) lines and data lines. Most VCF files contain both header lines and data lines. The header lines start with "##" and have a field in the format of "ID=value". The first header line is always the VCF format version followed by lines starting with ##INFO=, #FILTER=, and ##FORMAT, which define the name, length, value type, and description of each item in relevant fields (columns) of each data line.

Below is an example of VCF file header:

```
##fileformat=VCFv4.1
##source=VarScan2
##INFO=<ID=ADP,Number=1,Type=Integer,Description="Average per-sample depth
of bases with Phred score >= 15">
##INFO=<ID=WT,Number=1,Type=Integer,Description="Number of samples called
reference (wild-type)">
##INFO=<ID=HET,Number=1,Type=Integer,Description="Number of samples called
heterozygous-variant">
##INFO=<ID=HOM,Number=1,Type=Integer,Description="Number of samples called
homozygous-variant">
##INFO=<ID=NC,Number=1,Type=Integer,Description="Number of samples not
called">
##FILTER=<ID=str10, Description="Less than 10 % or more than 90 % of variant
supporting reads on one strand">
##FILTER=<ID=indelError,Description="Likely artifact due to indel reads at
this position">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=SDP,Number=1,Type=Integer,Description="Raw Read Depth as
reported by SAMtools">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Quality Read Depth of
bases with Phred score >= 15">
##FORMAT=<ID=RD,Number=1,Type=Integer,Description="Depth of reference-
supporting bases (reads1)">
##FORMAT=<ID=AD,Number=1,Type=Integer,Description="Depth of variant-
supporting bases (reads2)">
##FORMAT=<ID=FREQ,Number=1,Type=String,Description="Variant allele
frequency">
##FORMAT=<ID=PVAL,Number=1,Type=String,Description="P-value from Fisher's
Exact Test">
##FORMAT=<ID=RBQ,Number=1,Type=Integer,Description="Average quality of
reference-supporting bases (qual1)">
##FORMAT=<ID=ABQ,Number=1,Type=Integer,Description="Average quality of
variant-supporting bases (qual2)">
##FORMAT=<ID=RDF,Number=1,Type=Integer,Description="Depth of reference-
supporting bases on forward strand (reads1plus)">
##FORMAT=<ID=RDR,Number=1,Type=Integer,Description="Depth of reference-
supporting bases on reverse strand (reads1minus)">
##FORMAT=<ID=ADF,Number=1,Type=Integer,Description="Depth of variant-
supporting bases on forward strand (reads2plus)">
##FORMAT=<ID=ADR,Number=1,Type=Integer,Description="Depth of variant-
supporting bases on reverse strand (reads2minus)">
```

In this example, VCF file version is v4.1 and the file is generated with VarScan2 software. The ##INFO lines listed five IDs which will be used in the INFO column of data line to describe, for the variant in each line, the average read depth of bases with Phred score ≥ 15, how many of samples are wild type, heterozygous, homozygous, or no call. The two ##FILTER lines listed how a variant being filtered. The nine ##FORMAT lines show what contents will and how be arranged in the FORMAT column of each data line.

Data lines in VCF file are tab-delimited text lines and each line contains nine fixed fields (columns) followed by one or more sample column(s). The first nine fields are:

- CHROM: chromosome name.
- POS: the leftmost position of the variant in the sequence.
- ID: variant identifier such as SNP id. "." denotes unavailable.
- REF: reference base (for SNP) or sequence (for INDEL).
- ALT: alternate base or bases (the observed variant).
- QUAL: Phred-scaled quality score.
- FILTER: fiter status. PASS for passed all filters or a semicolon-separated list of code for filters that fail.
- INFO: additional information in <key>=<data> format.
- FORMAT: colon separated key and value.
- Sample field(s): one or more sample columns may exist in a VCF file. This field has the values for a sample arranged in the format defined in previous FORMAT field.

The first seven columns in each VCF data line are information of the variant itself, for example, the line below shows variant is on chromosome 1 at position of 880639 base, no ID (snp) since it is a deletion, reference bases are TC and C is missing in sample sequence. The call of this variant has no quality score but has passed the filtering.

```
#CHROM  POS  ID  REF  ALT  QUAL  FILTER
1  880639  .  TC  T  .  PASS
```

The INFO and FORMAT columns in VCF data line list information of the variant in sample(s). To understand the meaning of contents in these two columns, one must refer to the information given by header lines. For example, the contents of "ADP=10;WT=0;HET=1;HOM=0;NC=0" in INFO column indicate that average depth (ADP) for the variant in the samples is 10, one heterozygous (HET) is called and no other calls. The FORMAT column lists all tags (type of information) and their order for each sample. If a FORMAT column has following content: "GT:GQ:SDP:DP:RD:AD:FREQ:PVAL:RBQ:ABQ:RDF:RDR:ADF:ADR" and values in a sample column is as "0/1:22:10:10:4:6:60%:5.418E-3:36:38:3:1:0:6", the information listed for the sample will be genotype (0/1), genotype quality (22), raw read depth from SAMtools (10), quality read depth with Phred score>=15 (10), depth of reference-supporting bases (4), depth of variant-supporting bases (6), variant allele frequency (60 %), *P*-value from Fisher's Exact Test (0.005418), average quality of reference-supporting bases (36), average quality of variant-supporting bases (38), depth of reference-supporting bases on forward strand (3), depth of reference-supporting bases on reverse strand (1), depth of variant-supporting bases on forward strand (0), and depth of variant-supporting bases on reverse strand (6).

Detailed specification of VCF format can be found at https://github.com/samtools/hts-specs. VCFtools [16] is a software equipped with numerous functionalities to process VCF files. A small VCF file can also be viewed from command line of Unix/Linux computer or Microsoft Excel.

3 Conclusions

Next-generation sequencing technologies have brought new file formats and resurrected old formats. Understanding the basics of these formats is important to knowing what information they contain and how they can be used in various steps from raw data generation to interpretable biological results.

References

1. Shendure J, Ji H (2008) Next-generation DNA sequencing. Nat Biotechnol 26:1135–1145
2. Metzker ML (2010) Sequencing technologies—the next generation. Nat Rev Genet 11:31–46
3. Quail MA, Smith M, Coupland P et al (2012) A tale of three next generation sequencing platforms: comparison of Ion Torrent, Pacific Biosciences and Illumina MiSeq sequencers. BMC Genomics 13:341
4. Mardis ER (2008) Next-generation DNA sequencing methods. Annu Rev Genomics Hum Genet 9:387–402
5. Mardis ER (2013) Next-generation sequencing platforms. Annu Rev Anal Chem 6:287–303
6. Flicek P, Birney E (2009) Sense from sequence reads: methods for alignment and assembly. Nat Methods 6(Suppl 11):S6–S12
7. Medvedev P, Stanciu M, Brudno M (2009) Computational methods for discovering structural variation with next-generation sequencing. Nat Methods 6(Suppl 11):S13–S20
8. Pepke S, Wold B, Mortazavi A (2009) Computation for ChIP-seq and RNA-seq studies. Nat Methods 6(Suppl 11):S22–S32
9. van Dijk EL, Auger H, Jaszczyszyn Y et al (2014) Ten years of next-generation sequencing technology. Trends Genet 30:418–426
10. Voelkerding KV, Dames SA, Durtschi JD (2009) Next-generation sequencing: from basic research to diagnostics. Clin Chem 55:641–658
11. Pavlopoulos GA, Oulas A, Lacucci E et al (2013) Unraveling genomic variation from next generation sequencing data. BioData Min 6:13
12. Allcock RJN (2014) Production and analytic bioinformatics for next-generation DNA sequencing. In: Trent R (ed) Clinical bioinformatics, 2nd edn. Humana, New York, pp 17–30
13. Cock PJ, Fields CJ, Goto N et al (2010) The Sanger FASTQ file format for sequences with quality scores, and the solexa/illumina FASTQ variants. Nucleic Acids Res 38:1767–1771
14. Li H, Handsaker B, Wysoker A et al (2009) The sequence alignment/Map format and SAMtools. Bioinformatics 25:2078–2079
15. The SAM/BAM Format Specification Working Group (2014) Sequence alignment/map format specification. http://samtools.github.io/hts-specs/SAMv1.pdf

16. Danecek P, Auton A, Abecasis G et al (2011) The variant call format and VCFtools. Bioinformatics 27:2156–2158
17. Ewing B, Hillier L, Wendl MC et al (1998) Base-calling of automated sequencer traces using Phred. I. Accuracy assessment. Genome Res 8:175–185
18. Ewing B, Green P (1998) Base-calling of automated sequencer traces using Phred. II. Error probabilities. Genome Res 8:186–194
19. Andrews S (2010) FastQC: a quality control tool for high throughput sequence data., Available online at http://www.bioinformatics.babraham.ac.uk/projects/fastqc
20. Lipman D, Pearson W (1985) Rapid and sensitive protein similarity searches. Science 227:1435–1441
21. Pearson WR, Lipman DJ (1988) Improved tools for biological sequence comparison. Proc Natl Acad Sci 85:2444–2448
22. Wu TD, Watanabe CK (2005) GMAP: a genomic mapping and alignment program for mRNA and EST sequences. Bioinformatics 21:1859–1875
23. Langmead B, Trapnell C, Pop M et al (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol 10:R25
24. Li H, Durbin R (2010) Fast and accurate long-read alignment with Burrows-Wheeler transform. Bioinformatics 26:589–595
25. Dobin A, Davis CA, Schlesinger F et al (2013) STAR: ultrafast universal RNA-seq aligner. Bioinformatics 29:15–21
26. Robinson JT, Thorvaldsdóttir H, Winckler W et al (2011) Integrative Genomics Viewer. Nat Biotechnol 29:24–26
27. Thorvaldsdóttir H, Robinson JT, Mesirov JP (2013) Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. Brief Bioinform 14:178–192
28. Generic Feature Format (GFF). http://www.sanger.ac.uk/resources/software/gff/spec.html
29. GFF/GTF File Format—Definition and supported options. http://www.ensembl.org/info/website/upload/gff.html
30. BED File Format. Definition and supported options. http://useast.ensembl.org/info/website/upload/bed.html
31. BED format. http://genome.ucsc.edu/FAQ/FAQformat.html#format1
32. The 1000 Genomes Project Consortium (2010) A map of human genome variation from population-scale sequencing. Nature 467:1061–1073
33. McVean GA, Abecasis DM, Auton R et al (2012) An integrated map of genetic variation from 1,092 human genomes. Nature 491:56–65