Data File Formats and Relevant Tools in Next-generation Sequencing

Yi Xianfu
xfyin@sibs.ac.cn

Institute of Health Sciences
Shanghai Institute for Biological Science
Chinese Academy of Science

November 3, 2011

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
3. PART III: Relevant tools in NGS

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS
   - FASTX-Toolkit, SolexaQA, BioPerl
   - BEDTools
   - VCFtools, BCFtools
   - SAMtools, BAMtools

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS
   - FASTX-Toolkit, SolexaQA, BioPerl
   - BEDTools
   - VCFtools, BCFtools
   - SAMtools, BAMtools

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS
   - FASTX-Toolkit, SolexaQA, BioPerl
   - BEDTools
   - VCFtools, BCFtools
   - SAMtools, BAMtools

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS
   - FASTX-Toolkit, SolexaQA, BioPerl
   - BEDTools
   - VCFtools, BCFtools
   - SAMtools, BAMtools

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS
   - FASTX-Toolkit, SolexaQA, BioPerl
   - BEDTools
   - VCFtools, BCFtools
   - SAMtools, BAMtools

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS
   - FASTX-Toolkit, SolexaQA, BioPerl
   - BEDTools
   - VCFtools, BCFtools
   - SAMtools, BAMtools

Overview

1. PART I: Knowledge for better understand
   - IUB/IUPAC code
   - Coordinate system
2. PART II: Data file formats in NGS
   - FASTA, FASTQ
   - BED
   - GFF/GTF
   - VCF/BCF
   - SAM/BAM
3. PART III: Relevant tools in NGS
   - FASTX-Toolkit, SolexaQA, BioPerl
   - BEDTools
   - VCFtools, BCFtools
   - SAMtools, BAMtools

Part I

Knowledge for better understand

1. IUB/IUPAC code
2. Coordinate system

Outline

Outline

IUPAC | Nucleic acid

Code	Meaning	Code	Meaning
A	Adenine	Y	Pyrimidine (C, T, or U)
C	Cytosine	K	T, U, or G (keto)
G	Guanine	W	T, U, or A (weak)
T	Thymine	B	C, T, U, or G (not A)
U	Uracil	D	A, T, U, or G (not C)
R	Purine (A or G)	H	A, T, U, or C (not G)
S	C or G (strong)	V	A, C, or G (not T, not U)
M	C or A (amino)	N	Any base (A, C, G, T, or U)
X	masked	-	gap of indeterminate length

IUPAC | Amino acid codes

1	3	Meaning	1	3	Meaning
A	Ala	Alanine	B	Asx	Aspartic acid or Asparagine
C	Cys	Cysteine	D	Asp	Aspartic acid
E	Glu	Glutamic acid	F	Phe	Phenylalanine
G	Gly	Glycine	H	His	Histidin
I	Ile	Isoleucine	K	Lys	Lysine
L	Leu	Leucine	M	Met	Methionine
N	Asn	Asparagine	P	Pro	Proline
Q	Gln	Glutamine	R	Arg	Arginine
S	Ser	Serine	T	Thr	Threonine
U	Sec	Selenocysteine	V	Val	Valine
W	Trp	Tryptophan	X	Xaa	Any amino acid
Y	Tyr	Tyrosine	Z	Glx	Glutamine or Glutamic acid
*		translation stop	-		gap of indeterminate length
O	Pyl	Pyrrolysine			

IUPAC | Reference

1. IUPAC code table
2. Sequence representation
3. http://www.ncbi.nlm.nih.gov/BLAST/blastcgihelp.shtml
4. FASTA format description
5. What is FASTA format?

Outline

Coordinate | 1-based

Definition

A coordinate system where the first base of a sequence is one. In this coordinate system, a region is specified by a closed interval. For example, the region between the 3rd and the 7th bases inclusive is `[3,7]`.

Formats using the 1-based coordinate system:

- GFF
- VCF
- SAM
- Wiggle

Coordinate | 1-based

Definition

A coordinate system where the first base of a sequence is one. In this coordinate system, a region is specified by a closed interval. For example, the region between the 3rd and the 7th bases inclusive is `[3,7]`.

Formats using the 1-based coordinate system:

- GFF
- VCF
- SAM
- Wiggle

Coordinate | 0-based

Definition

A coordinate system where the first base of a sequence is zero. In this coordinate system, a region is specified by a half-closed-half-open interval. For example, the region between the 3rd and the 7th bases inclusive is `[2,7)`.

Formats using the 0-based coordinate system:

- BED
- BAM
- PSL

Coordinate | 0-based

Definition

A coordinate system where the first base of a sequence is zero. In this coordinate system, a region is specified by a half-closed-half-open interval. For example, the region between the 3rd and the 7th bases inclusive is `[2,7)`.

Formats using the 0-based coordinate system:

- BED
- BAM
- PSL

Coordinate | Reference

1. 基因组的坐标系统：0-based 与 1-based
2. The SAM Format Specification
3. Database/browser start coordinates differ by 1 base
4. What does zero-based, half-open mean?
5. Coordinate Transforms
6. What are the advantages/disadvantages of one-based vs. zero-based genome coordinate systems
7. On genome coordinate systems and transposable element annotation
8. dbSNP 0-based (zero based) vs. 1-based Coordinate Representation
9. Reformatting / adjustments to the data

Part II

Data file formats in NGS

Outline

Outline

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:

FASTA | Format

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:
  - One line starting with a ">" sign, followed by a sequence identification code.

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:
  - One line starting with a ">" sign, followed by a sequence identification code.
  - It is optionally be followed by a textual description of the sequence.
  - One or more lines containing the sequence itself.
  - Note: no line may be longer than 80 characters in length.

FASTA | Format

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:
  - One line starting with a ">" sign, followed by a sequence identification code.
    It is optionally be followed by a textual description of the sequence.
  - One or more lines containing the sequence itself.

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:
  - One line starting with a ">" sign, followed by a sequence identification code.
    It is optionally be followed by a textual description of the sequence.
  - One or more lines containing the sequence itself.
    It is recommended that all lines of text be shorter than 80 characters in length.

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:
  - One line starting with a ">" sign, followed by a sequence identification code.
    It is optionally be followed by a textual description of the sequence.
  - One or more lines containing the sequence itself.
    It is recommended that all lines of text be shorter than 80 characters in length.

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:
  - One line starting with a ">" sign, followed by a sequence identification code.
    It is optionally be followed by a textual description of the sequence.
  - One or more lines containing the sequence itself.
    It is recommended that all lines of text be shorter than 80 characters in length.

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:
  - One line starting with a ">" sign, followed by a sequence identification code.
    It is optionally be followed by a textual description of the sequence.
  - One or more lines containing the sequence itself.
    It is recommended that all lines of text be shorter than 80 characters in length.

Definition

FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which nucleotides or amino acids are represented using single-letter codes. The format also allows for sequence names and comments to precede the sequences.

- A file in FASTA format may comprise more than one sequence.
- A sequence in FASTA format consists of:
  - One line starting with a ">" sign, followed by a sequence identification code.
    It is optionally be followed by a textual description of the sequence.
  - One or more lines containing the sequence itself.
    It is recommended that all lines of text be shorter than 80 characters in length.

FASTA | Format (contd.)

Sequences are expected to be represented in the standard IUB/IUPAC amino acid and nucleic acid codes, with these exceptions:

- lower-case letters are accepted and are mapped into upper-case;
- a single hyphen or dash can be used to represent a gap of indeterminate length;
- in amino acid sequences, U and * are acceptable letters;
- any numerical digits in the query sequence should either be removed or replaced by appropriate letter codes (e.g., N for unknown nucleic acid residue or X for unknown amino acid residue).

FASTA | Format (contd.)

Sequences are expected to be represented in the standard IUB/IUPAC amino acid and nucleic acid codes, with these exceptions:

- lower-case letters are accepted and are mapped into upper-case;
- a single hyphen or dash can be used to represent a gap of indeterminate length;
- in amino acid sequences, U and * are acceptable letters;
- any numerical digits in the query sequence should either be removed or replaced by appropriate letter codes (e.g., N for unknown nucleic acid residue or X for unknown amino acid residue).

FASTA | Format (contd.)

Sequences are expected to be represented in the standard IUB/IUPAC amino acid and nucleic acid codes, with these exceptions:

- lower-case letters are accepted and are mapped into upper-case;
- a single hyphen or dash can be used to represent a gap of indeterminate length;
- in amino acid sequences, U and * are acceptable letters;
- any numerical digits in the query sequence should either be removed or replaced by appropriate letter codes (e.g., N for unknown nucleic acid residue or X for unknown amino acid residue).

FASTA | Format (contd.)

Sequences are expected to be represented in the standard IUB/IUPAC amino acid and nucleic acid codes, with these exceptions:

- lower-case letters are accepted and are mapped into upper-case;
- a single hyphen or dash can be used to represent a gap of indeterminate length;
- in amino acid sequences, U and * are acceptable letters;
- any numerical digits in the query sequence should either be removed or replaced by appropriate letter codes (e.g., N for unknown nucleic acid residue or X for unknown amino acid residue).

FASTA | Example

```
1 >gi|5524211|gb|AAD44166.1| cytochrome b
2 LCLYTHIGRNIYYGSYLYSETWNTGIMLLLITMATAFMGYVLPWGQMSFWGATVITNLFS
3 AIPYIGTNLVEWIWGGFSVDKATLNRFFAFHFILPFTMVALAGVHLTFLHETGSNNPLGL
4 TSDSDKIPFHPYYTIKDFLGLLILILLLLLLALLSPDMLGDPDNHMPADPLNTPLHIKPE
5 WYFLFAYAILRSVPNKLGGVLALFLSIVILGLMPFLHTSKHRSMMLRPLSQALFWTLTMD
6 LLTLTWIGSQPVEYPYTIIGQMASILYFSIILAFLPIAGXIENY
```

FASTA | Reference

1. FASTA 格式
2. Fasta 格式的详细说明
3. FASTA format (From Wikipedia)
4. http://www.ncbi.nlm.nih.gov/BLAST/blastcgihelp.shtml
5. FASTA format description
6. What is FASTA format?
7. http://www.bioinformatics.nl/tools/crab_fasta.html

Outline

Definition

FASTQ format is a text-based format for storing both a biological sequence (usually nucleotide sequence) and its corresponding quality scores. Both the sequence letter and quality score are encoded with a single ASCII character for brevity. It was originally developed at the Wellcome Trust Sanger Institute to bundle a FASTA sequence and its quality data, but has recently become the de facto standard for storing the output of high throughput sequencing instruments such as the Illumina Genome Analyzer.

There is no standard file extension for a FASTQ file, but .fq, .fastq, and .txt are commonly used.

FASTQ | Format

Definition

FASTQ format is a text-based format for storing both a biological sequence (usually nucleotide sequence) and its corresponding quality scores. Both the sequence letter and quality score are encoded with a single ASCII character for brevity. It was originally developed at the Wellcome Trust Sanger Institute to bundle a FASTA sequence and its quality data, but has recently become the de facto standard for storing the output of high throughput sequencing instruments such as the Illumina Genome Analyzer.

There is no standard file extension for a FASTQ file, but .fq, .fastq, and .txt are commonly used.

FASTQ | Format (contd.)

A FASTQ file normally uses four lines per sequence:

1. Line 1: begins with a '@' character and is followed by a sequence identifier and an optional description (like a FASTA title line)
2. Line 2: the raw sequence letters
3. Line 3: begins with a '+' character and is optionally followed by the same sequence identifier (and any description) again
4. Line 4: encodes the quality values for the sequence in Line 2, and must contain the same number of symbols as letters in the sequence

FASTQ | Format (contd.)

A FASTQ file normally uses four lines per sequence:

1. Line 1: begins with a '@' character and is followed by a sequence identifier and an optional description (like a FASTA title line)
2. Line 2: the raw sequence letters
3. Line 3: begins with a '+' character and is optionally followed by the same sequence identifier (and any description) again
4. Line 4: encodes the quality values for the sequence in Line 2, and must contain the same number of symbols as letters in the sequence

FASTQ | Format (contd.)

A FASTQ file normally uses four lines per sequence:

1. Line 1: begins with a '@' character and is followed by a sequence identifier and an optional description (like a FASTA title line)
2. Line 2: the raw sequence letters
3. Line 3: begins with a '+' character and is optionally followed by the same sequence identifier (and any description) again
4. Line 4: encodes the quality values for the sequence in Line 2, and must contain the same number of symbols as letters in the sequence

FASTQ | Format (contd.)

A FASTQ file normally uses four lines per sequence:

1. Line 1: begins with a '@' character and is followed by a sequence identifier and an optional description (like a FASTA title line)
2. Line 2: the raw sequence letters
3. Line 3: begins with a '+' character and is optionally followed by the same sequence identifier (and any description) again
4. Line 4: encodes the quality values for the sequence in Line 2, and must contain the same number of symbols as letters in the sequence

FASTQ | Format (contd.)

A FASTQ file normally uses four lines per sequence:

1. Line 1: begins with a '@' character and is followed by a sequence identifier and an optional description (like a FASTA title line)
2. Line 2: the raw sequence letters
3. Line 3: begins with a '+' character and is optionally followed by the same sequence identifier (and any description) again
4. Line 4: encodes the quality values for the sequence in Line 2, and must contain the same number of symbols as letters in the sequence

FASTQ | Example

```
1 @SEQ_ID
2 GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTG
3 +
4 !''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF
```

- Illumina

```
@HWUSI-EAS100R:6:73:941:1973#0/1
```

- CASAVA 1.8

```
@EAS139:136:FC706VJ:2:2104:15343:197393 1:Y:18:ATCACG
```

- NCBI Sequence Read Archive

```
@SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
```

The NCBI have converted this FASTQ data from the original Solexa/Illumina encoding to the Sanger standard

FASTQ | Identifiers

- Illumina

```
1 @HWUSI-EAS100R:6:73:941:1973#0/1
```

- CASAVA 1.8

```
1 @EAS139:136:FC706VJ:2:2104:15343:197393 1:Y:18:ATCACG
```

- NCBI Sequence Read Archive

```
1 @SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
```

The NCBI have converted this FASTQ data from the original Solexa/Illumina encoding to the Sanger standard.

FASTQ | Identifiers

- Illumina

```
1 @HWUSI-EAS100R:6:73:941:1973#0/1
```

- CASAVA 1.8

```
1 @EAS139:136:FC706VJ:2:2104:15343:197393 1:Y:18:ATCACG
```

- NCBI Sequence Read Archive

```
1 @SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
```

The NCBI have converted this FASTQ data from the original Solexa/Illumina encoding to the Sanger standard.

FASTQ | Identifiers

- Illumina

```
1 @HWUSI-EAS100R:6:73:941:1973#0/1
```

- CASAVA 1.8

```
1 @EAS139:136:FC706VJ:2:2104:15343:197393 1:Y:18:ATCACG
```

- NCBI Sequence Read Archive

```
1 @SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
```

The NCBI have converted this FASTQ data from the original Solexa/Illumina encoding to the Sanger standard.

FASTQ | Identifiers

- Illumina

```
1 @HWUSI-EAS100R:6:73:941:1973#0/1
```

- CASAVA 1.8

```
1 @EAS139:136:FC706VJ:2:2104:15343:197393 1:Y:18:ATCACG
```

- NCBI Sequence Read Archive

```
1 @SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
```

The NCBI have converted this FASTQ data from the original Solexa/Illumina encoding to the Sanger standard.

FASTQ | Variations | Quality

A quality value Q is an integer mapping of p (i.e., the probability that the corresponding base call is incorrect).

Two different equations have been in use:

1. The standard Sanger variant (Phred quality score)

$$Q = -10 \log_{10} p \tag{1}$$

2. Solexa prior to v1.3

$$Q = -10 \log_{10} \frac{p}{1-p} \tag{2}$$

For raw reads, the range of scores will depend on the technology and the base caller used, but will typically be up to 40. For aligned sequences and consensuses higher scores are common.

FASTQ | Variations | Quality

A quality value Q is an integer mapping of p (i.e., the probability that the corresponding base call is incorrect).
Two different equations have been in use:

1. The standard Sanger variant (Phred quality score)

$$Q = -10 \log_{10} p \tag{1}$$

2. Solexa prior to v1.3

$$Q = -10 \log_{10} \frac{p}{1-p} \tag{2}$$

For raw reads, the range of scores will depend on the technology and the base caller used, but will typically be up to 40. For aligned sequences and consensuses higher scores are common.

A quality value Q is an integer mapping of p (i.e., the probability that the corresponding base call is incorrect).
Two different equations have been in use:

1. The standard Sanger variant (Phred quality score)

$$Q = -10 \log_{10} p \tag{1}$$

2. Solexa prior to v1.3

$$Q = -10 \log_{10} \frac{p}{1-p} \tag{2}$$

For raw reads, the range of scores will depend on the technology and the base caller used, but will typically be up to 40. For aligned sequences and consensuses higher scores are common.

A quality value Q is an integer mapping of p (i.e., the probability that the corresponding base call is incorrect).
Two different equations have been in use:

1. The standard Sanger variant (Phred quality score)

$$Q = -10 \log_{10} p \tag{1}$$

2. Solexa prior to v1.3

$$Q = -10 \log_{10} \frac{p}{1-p} \tag{2}$$

For raw reads, the range of scores will depend on the technology and the base caller used, but will typically be up to 40. For aligned sequences and consensuses higher scores are common.

FASTQ | Variations | Quality

A quality value Q is an integer mapping of p (i.e., the probability that the corresponding base call is incorrect).
Two different equations have been in use:

1. The standard Sanger variant (Phred quality score)

$$Q = -10 \log_{10} p \tag{1}$$

2. Solexa prior to v1.3

$$Q = -10 \log_{10} \frac{p}{1-p} \tag{2}$$

For raw reads, the range of scores will depend on the technology and the base caller used, but will typically be up to 40. For aligned sequences and consensuses higher scores are common.

A quality value Q is an integer mapping of p (i.e., the probability that the corresponding base call is incorrect).
Two different equations have been in use:

1. The standard Sanger variant (Phred quality score)

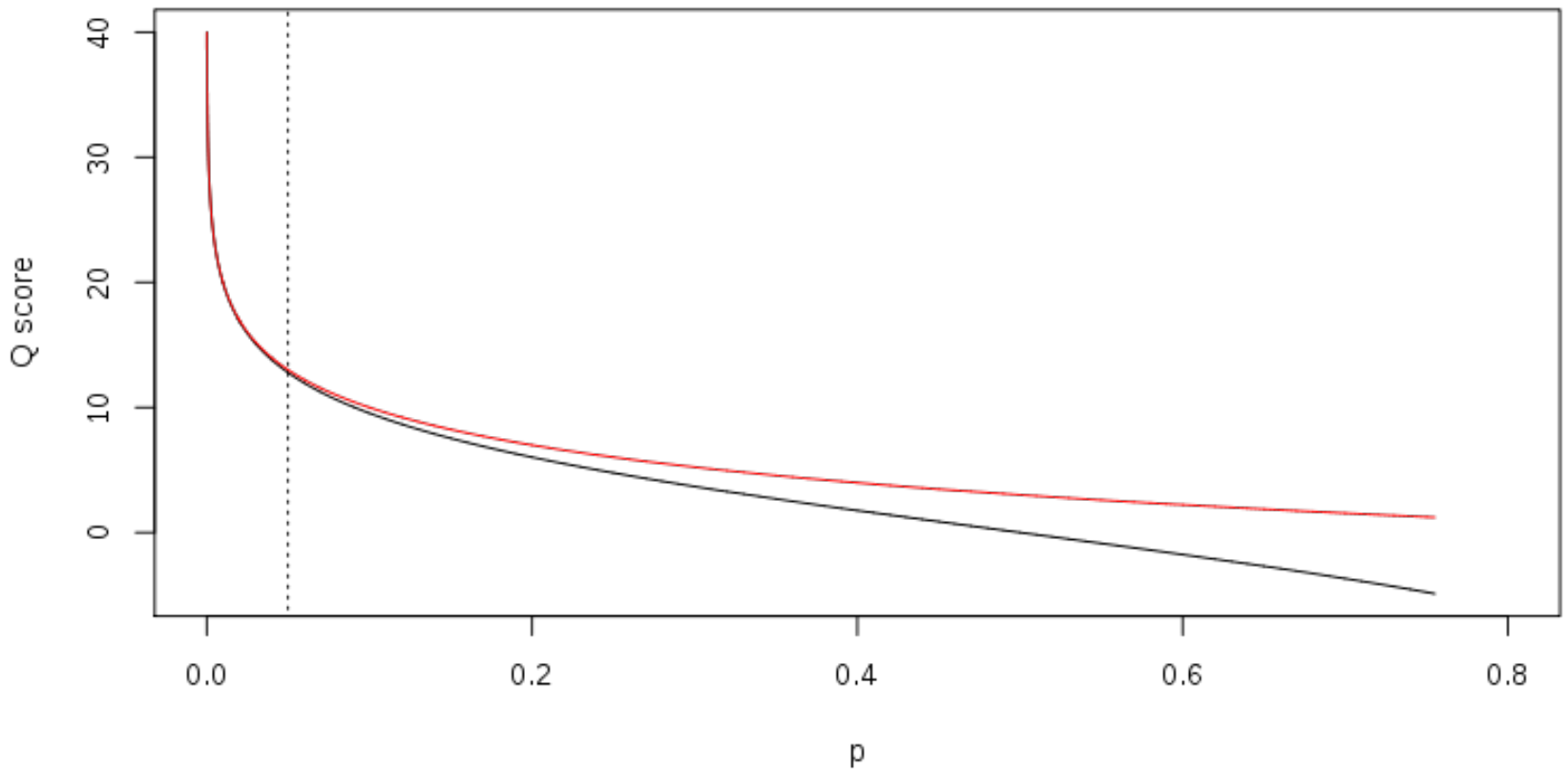
$$Q = -10 \log_{10} p \tag{1}$$

2. Solexa prior to v1.3

$$Q = -10 \log_{10} \frac{p}{1-p} \tag{2}$$

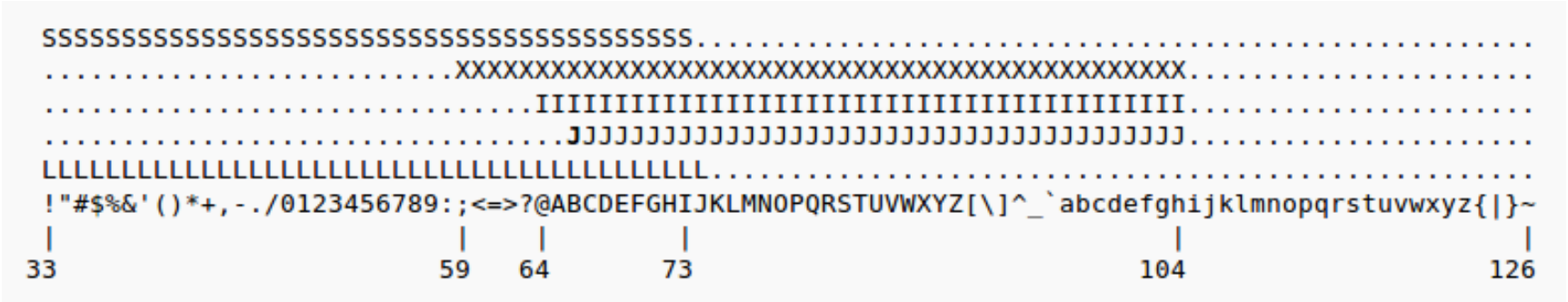
For raw reads, the range of scores will depend on the technology and the base caller used, but will typically be up to 40. For aligned sequences and consensuses higher scores are common.

FASTQ | Variations | Quality (contd.)



Relationship between Q and p using the Sanger (red) and Solexa (black) equations. The vertical dotted line indicates p = 0.05, or equivalently, Q ≈ 13.

FASTQ | Variations | Encoding



Letter	Meaning	Encoding	Raw reads typically
S	Sanger	Phred+33	(0, 40)
X	Solexa	Solexa+64	(-5, 40)
I	Illumina 1.3+	Phred+64	(0, 40)
J	Illumina 1.5+	Phred+64	(3, 40)‡
L	Illumina 1.8+	Phred+33	(0, 41)

‡ with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)

FASTQ | Reference

1. FASTQ 格式
2. FASTQ 格式中的测序质量
3. Fastq 格式的详细说明
4. FASTQ format (From Wikipedia)
5. FASTQ Format Specification
6. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants
7. fastq 质量值转化

Outline

BED | Format

Definition

Browser Extensible Data (BED) format provides a flexible way to define the data lines that are displayed in an annotation track. It defines one genomic region (a "BED record") per line. BED lines have three required fields and nine additional optional fields. The number of fields per line must be consistent throughout any single set of data in an annotation track. The order of the optional fields is binding.

Coordinate

The coordinates in a BED record are both 0-based, meaning the first base on a chromosome is numbered 0. A BED interval is also half-opened half-closed. The genome browser region "chr1:1-1000" would be described in a BED record as "chr1 0 1000" with the start coordinate being one smaller and the end coordinate being the same, describing the half-closed half-open interval `[0,1000)` of length 1000bp starting at base 0.

BED | Format

Definition

Browser Extensible Data (BED) format provides a flexible way to define the data lines that are displayed in an annotation track. It defines one genomic region (a "BED record") per line. BED lines have three required fields and nine additional optional fields. The number of fields per line must be consistent throughout any single set of data in an annotation track. The order of the optional fields is binding.

Coordinate

The coordinates in a BED record are both 0-based, meaning the first base on a chromosome is numbered 0. A BED interval is also half-opened half-closed. The genome browser region "chr1:1-1000" would be described in a BED record as "chr1 0 1000" with the start coordinate being one smaller and the end coordinate being the same, describing the half-closed half-open interval `[0,1000)` of length 1000bp starting at base 0.

BED | format (contd.)

3 required

1. chrom
2. chromStart
3. chromEnd

9 optional

4. name
5. score
6. strand
7. thickStart
8. thickEnd
9. itemRgb
10. blockCount
11. blockSizes
12. blockStarts

BED | format (contd.)

3 required

1. chrom
2. chromStart
3. chromEnd

9 optional

4. name
5. score
6. strand
7. thickStart
8. thickEnd
9. itemRgb
10. blockCount
11. blockSizes
12. blockStarts

BED | Example

```
1 chr22 1000 5000 cloneA 960 + 1000 5000 0 2 567,488, 0,3512
2 chr22 2000 6000 cloneB 900 - 2000 6000 0 2 433,399, 0,3601
```

BED | Reference

1. BED format
2. BED File Format - Definition and supported options
3. What is BED format?
4. Genomic regions: BED File Format - Description

Outline

GFF | Version

- GFF Version 2 has a number of deficiencies, notably that it can only represent two-level feature hierarchies and thus cannot handle the three-level hierarchy of gene $\rightarrow$ transcript $\rightarrow$ exon.
- GFF3 addresses this and other deficiencies. For example, it supports arbitrarily many hierarchical levels, and gives specific meanings to certain tags in the attributes field.
- The Gene transfer format (GTF) is a refinement of GFF Version 2 and is sometimes referred to as GFF2.5.

GFF | Version

- GFF Version 2 has a number of deficiencies, notably that it can only represent two-level feature hierarchies and thus cannot handle the three-level hierarchy of gene $\rightarrow$ transcript $\rightarrow$ exon.
- GFF3 addresses this and other deficiencies. For example, it supports arbitrarily many hierarchical levels, and gives specific meanings to certain tags in the attributes field.
- The Gene transfer format (GTF) is a refinement of GFF Version 2 and is sometimes referred to as GFF2.5.

GFF | Version

- GFF Version 2 has a number of deficiencies, notably that it can only represent two-level feature hierarchies and thus cannot handle the three-level hierarchy of gene $\rightarrow$ transcript $\rightarrow$ exon.
- GFF3 addresses this and other deficiencies. For example, it supports arbitrarily many hierarchical levels, and gives specific meanings to certain tags in the attributes field.
- The Gene transfer format (GTF) is a refinement of GFF Version 2 and is sometimes referred to as GFF2.5.

GFF3 | Format

Definition

GFF is a standard file format for storing genomic features in a text file. GFF stands for Generic Feature Format. GFF files are plain text, 9 column, tab-delimited files. The filename extension associated with such files is .GFF.

Definition

GFF3 format is a flat tab-delimited file. The first line of the file is a comment that identifies the file format and version. This is followed by a series of data lines, each one of which corresponds to an annotation. The ##gff-version 3 line is required and must be the first line of the file. It introduces the annotation section of the file.
Blank lines and lines beginning with a single # are ignored.

GFF3 | Format

Definition

GFF is a standard file format for storing genomic features in a text file. GFF stands for Generic Feature Format. GFF files are plain text, 9 column, tab-delimited files. The filename extension associated with such files is .GFF.

Definition

GFF3 format is a flat tab-delimited file. The first line of the file is a comment that identifies the file format and version. This is followed by a series of data lines, each one of which corresponds to an annotation. The ##gff-version 3 line is required and must be the first line of the file. It introduces the annotation section of the file.
Blank lines and lines beginning with a single # are ignored.

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required
6. score: a floating point number; "." (a period) for no score
7. strand: "+" for positive strand; "-" for minus strand; "." for not stranded
8. ?: for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required
6. score: a floating point number; "." (a period) for no score
7. strand: + for positive strand, - for minus strand, . for not stranded, ? for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required
6. score: a floating point number; "." (a period) for no score
7. strand: + for positive strand, - for minus strand, . for not stranded, ? for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required
6. score: a floating point number; "." (a period) for no score
7. strand: + for positive strand, - for minus strand, . for not stranded, ? for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required

Start & End

Start is always less than or equal (for zero-length features, such as insertion sites) to end.

6. score: a floating point number; "." (a period) for no score
7. strand: + for positive strand, - for minus strand, . for not stranded, ? for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required

Start & End

Start is always less than or equal (for zero-length features, such as insertion sites) to end.

6. score: a floating point number; "." (a period) for no score
7. strand: + for positive strand, - for minus strand, . for not stranded, ? for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required

Start & End

Start is always less than or equal (for zero-length features, such as insertion sites) to end.

6. score: a floating point number; "." (a period) for no score
7. strand: + for positive strand, - for minus strand, . for not strand
   ? for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required

Start & End

Start is always less than or equal (for zero-length features, such as insertion sites) to end.

6. score: a floating point number; "." (a period) for no score
7. strand: + for positive strand, - for minus strand, . for not strand, ? for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

The 9 columns of the annotation section are:

1. seqid: required; `[a-zA-Z0-9.:^*$@!+_?-|]`
2. source: not necessary; "." (a period) for no source
3. type: required
4. start: required
5. end: required

Start & End

Start is always less than or equal (for zero-length features, such as insertion sites) to end.

6. score: a floating point number; "." (a period) for no score
7. strand: + for positive strand, - for minus strand, . for not stranded ? for features whose strandedness is relevant, but unknown

GFF3 | Format (contd.)

8. phase: required for all CDS features; one of the integers 0, 1, or 2; "." (a period) for no phase

More

- 0, 1, or 2, indicating the number of bases that should be removed from the beginning of this feature to reach the first base of the next codon.
- For forward strand features, phase is counted from the start field.
- For reverse strand features, phase is counted from the end field.

9. attributes: not required; format tag=value; multiple tag=value pairs are separated by semicolons; separate the values of the same tag with ","; spaces are allowed; tag are case sensitive, tags that begin with an uppercase letter are reserved

GFF3 | Format (contd.)

8. phase: required for all CDS features; one of the integers 0, 1, or 2; "." (a period) for no phase

More

- 0, 1, or 2, indicating the number of bases that should be removed from the beginning of this feature to reach the first base of the next codon.
- For forward strand features, phase is counted from the start field.
- For reverse strand features, phase is counted from the end field.

9. attributes: not required; format tag=value; multiple tag=value pairs are separated by semicolons; separate the values of the same tag with ","; spaces are allowed; tag are case sensitive, tags that begin with an uppercase letter are reserved

GFF3 | Format (contd.)

8. phase: required for all CDS features; one of the integers 0, 1, or 2; "." (a period) for no phase

More

- 0, 1, or 2, indicating the number of bases that should be removed from the beginning of this feature to reach the first base of the next codon.
- For forward strand features, phase is counted from the start field.
- For reverse strand features, phase is counted from the end field.

9. attributes: not required; format tag=value; multiple tag=value pairs are separated by semicolons; separate the values of the same tag with "," ; spaces are allowed; tag are case sensitive, tags that begin with an uppercase letter are reserved

GFF3 | Format (contd.)

8. phase: required for all CDS features; one of the integers 0, 1, or 2; "." (a period) for no phase

More

- 0, 1, or 2, indicating the number of bases that should be removed from the beginning of this feature to reach the first base of the next codon.
- For forward strand features, phase is counted from the start field.
- For reverse strand features, phase is counted from the end field.

9. attributes: not required; format tag=value; multiple tag=value pairs are separated by semicolons; separate the values of the same tag with "," ; spaces are allowed; tag are case sensitive, tags that begin with an uppercase letter are reserved

GFF3 | Format (contd.)

8. phase: required for all CDS features; one of the integers 0, 1, or 2; "." (a period) for no phase

More

- 0, 1, or 2, indicating the number of bases that should be removed from the beginning of this feature to reach the first base of the next codon.
- For forward strand features, phase is counted from the start field.
- For reverse strand features, phase is counted from the end field.

9. attributes: not required; format tag=value; multiple tag=value pairs are separated by semicolons; separate the values of the same tag with "," ; spaces are allowed; tag are case sensitive, tags that begin with an uppercase letter are reserved

GFF3 | Format (contd.)

8. phase: required for all CDS features; one of the integers 0, 1, or 2; "." (a period) for no phase

More

- 0, 1, or 2, indicating the number of bases that should be removed from the beginning of this feature to reach the first base of the next codon.
- For forward strand features, phase is counted from the start field.
- For reverse strand features, phase is counted from the end field.

9. attributes: not required; format tag=value; multiple tag=value pairs are separated by semicolons; separate the values of the same tag with "," ; spaces are allowed; tag are case sensitive, tags that begin with an uppercase letter are reserved

GFF3 | Format (contd.)

Predefined tags

- ID: unique within the scope of the GFF file
- Name: no requirement be unique within the file
- Alias: no requirement be unique within the file
- Parent: can only be used to indicate a partof relationship
- Target: value format "target_id start end [strand]"
- Gap: for alignment is not collinear
- Derives_from: needed for polycistronic genes
- Note: free text note
- Dbxref: database cross reference
- Ontology_term: cross reference to an ontology term
- Is_circular: whether a feature is circula

GFF3 | Example

A gene named "EDEN" which has three alternatively-spliced mRNA transcripts:

```
ctg123 example gene             1050 9000 . + . ID=EDEN;Name=EDEN;Note=protein kinase

ctg123 example mRNA             1050 9000 . + . ID=EDEN.1;Parent=EDEN;Name=EDEN.1;Index=1
ctg123 example five_prime_UTR   1050 1200 . + . Parent=EDEN.1
ctg123 example CDS              1201 1500 . + 0 Parent=EDEN.1
ctg123 example CDS              3000 3902 . + 0 Parent=EDEN.1
ctg123 example CDS              5000 5500 . + 0 Parent=EDEN.1
ctg123 example CDS              7000 7608 . + 0 Parent=EDEN.1
ctg123 example three_prime_UTR  7609 9000 . + . Parent=EDEN.1

ctg123 example mRNA             1050 9000 . + . ID=EDEN.2;Parent=EDEN;Name=EDEN.2;Index=1
ctg123 example five_prime_UTR   1050 1200 . + . Parent=EDEN.2
ctg123 example CDS              1201 1500 . + 0 Parent=EDEN.2
ctg123 example CDS              5000 5500 . + 0 Parent=EDEN.2
ctg123 example CDS              7000 7608 . + 0 Parent=EDEN.2
ctg123 example three_prime_UTR  7609 9000 . + . Parent=EDEN.2

ctg123 example mRNA             1300 9000 . + . ID=EDEN.3;Parent=EDEN;Name=EDEN.3;Index=1
ctg123 example five_prime_UTR   1300 1500 . + . Parent=EDEN.3
ctg123 example five_prime_UTR   3000 3300 . + . Parent=EDEN.3
ctg123 example CDS              3301 3902 . + 0 Parent=EDEN.3
ctg123 example CDS              5000 5500 . + 1 Parent=EDEN.3
ctg123 example CDS              7000 7600 . + 1 Parent=EDEN.3
ctg123 example three_prime_UTR  7601 9000 . + . Parent=EDEN.3
```

GFF | Reference

1. General feature format
2. GFF
3. GffFormat
4. GFF files
5. GFF 格式说明
6. GENERIC FEATURE FORMAT VERSION 3
7. GFF
8. GFF: an Exchange Format for Feature Description
9. GFF2
10. GFF format
11. GTF format
12. GTF2.2: A Gene Annotation Format

Outline

VCF | Format

Definition

Variant Call Format (VCF) is a text file format (most likely stored in a compressed manner). It contains meta-information lines, a header line, and then data lines each containing information about a position in the genome.
There is an option whether to contain genotype information on samples for each position or not.

Definition

BCF, or the binary variant call format, is the binary version of VCF. It keeps the same information in VCF, while much more efficient to process especially for many samples. The relationship between BCF and VCF is similar to that between BAM and SAM.

VCF | Format

Definition

Variant Call Format (VCF) is a text file format (most likely stored in a compressed manner). It contains meta-information lines, a header line, and then data lines each containing information about a position in the genome.
There is an option whether to contain genotype information on samples for each position or not.

Definition

BCF, or the binary variant call format, is the binary version of VCF. It keeps the same information in VCF, while much more efficient to process especially for many samples. The relationship between BCF and VCF is similar to that between BAM and SAM.

VCF | Format | Meta-information lines

```
1  ##fileformat=VCFv4.1
2  ##INFO=<ID=ID,Number=number,Type=type,Description="description">
3  ##FILTER=<ID=ID,Description="description">
4  ##FORMAT=<ID=ID,Number=number,Type=type,Description="description
       ">
5  ##ALT=<ID=type,Description=description>
6  ##assembly=url
7  ##contig=<ID=ctg1,URL=ftp://somewhere.org/assembly.fa,...>
8  ##SAMPLE=<ID=S_ID,Genomes=G1_ID;G2_ID; ...;GK_ID,Mixture=N1;N2;
       ...;NK,Description=S1;S2; ...; SK >
9  ##PEDIGREE=<Name_0=G0-ID,Name_1=G1-ID,...,Name_N=GN-ID>
10 ##pedigreeDB=<url>
```

VCF | Format | The header line

Mandatory

tab-delimited 8, fixed:

1. #CHROM
2. POS
3. ID
4. REF
5. ALT
6. QUAL
7. FILTER
8. INFO

Optional

If genotype data is present:

9. FORMAT
10. sample ID1
11. sample ID2
12. ...
13. sample IDn

VCF | Format | The header line

Mandatory

tab-delimited 8, fixed:

1. #CHROM
2. POS
3. ID
4. REF
5. ALT
6. QUAL
7. FILTER
8. INFO

Optional

If genotype data is present:

9. FORMAT
10. sample ID1
11. sample ID2
12. …
13. sample IDn

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Format | Data lines

8 fixed fields per record; tab-delimited; . for missing values

1. CHROM: chromosome
2. POS: position, the 1st base having position 1
3. ID: semi-colon separated list of unique identifiers where available
4. REF: reference base(s)
5. ALT: comma separated list of alternate non-reference alleles called on at least one of the samples
6. QUAL: phred-scaled quality score for the assertion made in ALT
7. FILTER: "PASS" for pass, a semicolon-separated list of codes for fail
8. INFO: additional information

VCF | Example

```
##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 | Reference

1. VCF (Variant Call Format) version 4.1
2. VCF (Variant Call Format) version 4.0
3. Encoding Structural Variants in VCF (Variant Call Format) version 4.0
4. VCF format
5. Variant Call Format
6. VCF FORMAT

Outline

SAM | Format

Definition

SAM stands for Sequence Alignment/Map format. It is a TAB-delimited text format consisting of a header section, which is optional, and an alignment section. If present, the header must be prior to the alignments. Header lines start with "@", while alignment lines do not. Each alignment line has 11 mandatory fields for essential alignment information such as mapping position, and variable number of optional fields for flexible or aligner specific information.

Definition

Binary Alignment/Map (BAM) is the binary representation of SAM and keeps exactly the same information as SAM.

SAM | Format

Definition

SAM stands for Sequence Alignment/Map format. It is a TAB-delimited text format consisting of a header section, which is optional, and an alignment section. If present, the header must be prior to the alignments. Header lines start with "@", while alignment lines do not. Each alignment line has 11 mandatory fields for essential alignment information such as mapping position, and variable number of optional fields for flexible or aligner specific information.

Definition

Binary Alignment/Map (BAM) is the binary representation of SAM and keeps exactly the same information as SAM.

SAM | Alignment section | Mandatory fields

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,255}`	Query template NAME
2	FLAG	Int	$[0, 2^{16}-1]$	bitwise FLAG
3	RNAME	String	`\*|[!-()+-<>-~][!-~]*`	Reference sequence NAME
4	POS	Int	$[0, 2^{29}-1]$	1-based leftmost mapping POSition
5	MAPQ	Int	$[0, 2^{8}-1]$	MAPping Quality
6	CIGAR	String	`\*|([0-9]+[MIDNSHPX=])+`	CIGAR string
7	RNEXT	String	`\*|=|[!-()+-<>-~][!-~]*`	Ref. name of the mate/next segment
8	PNEXT	Int	$[0, 2^{29}-1]$	Position of the mate/next segment
9	TLEN	Int	$[-2^{29}+1, 2^{29}-1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

SAM | Alignment | Mandatory | FLAG

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

SAM | Alignment | Mandatory | CIGAR

Op	BAM	Description
M	0	alignment match (can be a sequence match or mismatch)
I	1	insertion to the reference
D	2	deletion from the reference
N	3	skipped region from the reference
S	4	soft clipping (clipped sequences present in SEQ)
H	5	hard clipping (clipped sequences NOT present in SEQ)
P	6	padding (silent deletion from padded reference)
=	7	sequence match
X	8	sequence mismatch

SAM | Alignment | Mandatory | CIGAR (contd.)

For example:

```
RefPos:     1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
Reference:  C  C  A  T  A  C  T  G  A  A  C  T  G  A  C  T  A  A  C
Read: ACTAGAATGGCT
```

Aligning these two:

```
RefPos:     1  2  3  4  5  6  7     8  9 10 11 12 13 14 15 16 17 18 19
Reference:  C  C  A  T  A  C  T     G  A  A  C  T  G  A  C  T  A  A  C
Read:                  A  C  T  A  G  A  A     T  G  G  C  T
```

With the alignment above, you get:

```
POS: 5
CIGAR: 3M1I3M1D5M
```

SAM | Example

```
1 @HD VN:1.0 SO:coordinate
2 @SQ SN:1 LN:249250621 AS:NCBI37 UR:file:/data/local/ref/GATK/
      human_g1k_v37.fasta M5:1b22b98cdeb4a9304cb5d48026a85128
3 @SQ SN:2 LN:243199373 AS:NCBI37 UR:file:/data/local/ref/GATK/
      human_g1k_v37.fasta M5:a0d9851da00400dec1098a9255ac712e
4 @RG ID:UM0098:1 PL:ILLUMINA PU:HWUSI-EAS1707-615LHAAXX-L001 LB
      :80 DT:2010-05-05T20:00:00-0400 SM:SD37743 CN:UMCORE
5 @RG ID:UM0098:2 PL:ILLUMINA PU:HWUSI-EAS1707-615LHAAXX-L002 LB
      :80 DT:2010-05-05T20:00:00-0400 SM:SD37743 CN:UMCORE
6 @PG ID:bwa VN:0.5.4
7 @PG ID:GATK TableRecalibration VN:1.0.3471 CL:Covariates=[
      ReadGroupCovariate, QualityScoreCovariate, CycleCovariate
      , DinucCovariate, TileCovariate], default_read_group=null
      , default_platform=null, force_read_group=null,
      force_platform=null, solid_recal_mode=SET_Q_ZERO,
      window_size_nqs=5, homopolymer_nback=7,
      exception_if_no_tile=false, ignore_nocall_colorspace=
      false, pQ=5, maxQ=40, smoothing=1
8 1:497:R:-272+13M17D24M 113 1 497 37 37M 15 100338662 0
      CGGGTCTGACCTGAGGAGAACTGTGCTCCGCCTTCAG
      0;==-==9;>>>>>=>>>>>>>>>>>=>>>>>>>>>> XT:A:U NM:i:0 SM:i
      :37 AM:i:0 X0:i:1 X1:i:0 XM:i:0 XO:i:0 XG:i:0 MD:Z:37
```

SAM | Reference

1. SAM Spec v1.4
2. The Sequence Alignment/Map format and SAMtools
3. SAM FORMAT
4. SAM
5. SAM

Outline

Pileup | Format

Definition

Pileup format describes the base-pair information at each chromosomal position. This format facilitates SNP/indel calling and brief alignment viewing by eyes. The pileup format has several variants.

Pileup | Format (contd.)

Columns

1. Chromosome
2. 1-based coordinate
3. Reference base
4. The number of reads covering the site
5. Read bases
6. Base qualities

Read bases

- `.` : match, forward strand
- `,` : match, reverse strand
- ACGTN : mismatch, forward strand
- acgtn : mismatch, reverse strand
- \+[0-9]+[ACGTNacgtn]+ : insertion
- -[0-9]+[ACGTNacgtn]+ : deletion
- ^ : start of a read segment
- $: end of a read segment
- ASCII of the character following ^ minus 33 gives the mapping quality
- > or < : reference skip

Pileup | Format (contd.)

Columns

1. Chromosome
2. 1-based coordinate
3. Reference base
4. The number of reads covering the site
5. Read bases
6. Base qualities

Read bases

- `.` : match, forward strand
- `,` : match, reverse strand
- ACGTN : mismatch, forward strand
- acgtn : mismatch, reverse strand
- `\+[0-9]+[ACGTNacgtn]+` : insertion
- `-[0-9]+[ACGTNacgtn]+` : deletion
- `^` : start of a read segment
- `$` : end of a read segment
- ASCII of the character following `^` minus 33 gives the mapping quality
- `>` or `<` : reference skip

Pileup | Example

```
1  seq1 272 T 24  ,.$.....,.,.,.,.,.,..^+.
                  <<<+;<<<<<<<<<<=<;<;7<&
2  seq1 273 T 23  ,.....,.,.,.,.,.,.,..A <<<;<<<<<<<<<3<=<<<;<<+
3  seq1 274 T 23  ,.$.....,.,.,.,.,.,.,...
                  7<7;<;<<<<<<<<<=<;<;<<6
4  seq1 275 A 23  ,$.....,.,.,.,.,.,.,...^l.
                  <+;9*<<<<<<<<<=<<:;<<<<
5  seq1 276 G 22  ...T,,.,.,.,.,.,.,.,....  33;+<<7=7<<7<&<<1;<<6<
6  seq1 277 T 22  ....,.,.,.C.,.,.,.,..G.  +7<;<<<<<<<&<=<<:;<<&<
7  seq1 278 G 23  .....,.,.,.,.,.,.,.,....^k.
                  %38*<<;<7<<7<=<<<;<<<<<
8  seq1 279 C 23  A..T,,.,.,.,.,.,.,.,.,.... ;75&<<<<<<<<<=<<<9<<:<<
9
10 seq2 156 A 11  .$......+2AG.+2AG.+2AGGG    <975;:<<<<<
11
12 seq3 200 A 20 ,,,,,,.,.-4CACC.-4CACC.....,,,.^~.
                  ==<<<<<<<<<<<::<;2<<
```

Pileup | Reference

1. Pileup Format
2. samtools
3. I do not understand the columns in the pileup output.

Outline

Others | 2bit

Definition

A .2bit file stores multiple DNA sequences (up to 4 Gb total) in a compact randomly-accessible format. The file contains masking information as well as the DNA itself.

Reference

1. .2bit format
2. UCSC 2Bit File Format
3. 2bit file format

Others | SRA

About

The Sequence Read Archive (SRA) was created and engineered at the National Center for Biotechnology Information (NCBI), National Library of Medicine, National Institutes of Health, Department of Health and Human Services.
The SRA is formerly known as the "Short Read Archive", but in recognition of the long reads now delivered by next generation platforms, the name was changed to "Sequence Read Archive".
SRA is the native archive format for the INSDC SRAs.

Reference

1. Sequence Read Archive
2. SRA Handbook
3. SRA File Formats Guide

Part III

Relevant tools in NGS

11 FASTA
12 FASTQ
13 FASTX
14 BED
15 GFF
16 VCF
17 SAM
18 BAM
19 Others

Outline

Outline

Tools

1. SeqTools (Javascript, online)
2. seqTools (Perl, CLI)
3. exonerate

```
Usage:   seqTools.pl <command> [<arguments>]

Command: format   Format the FASTA record(s).
         revcom   Reverse and complement the FASTA record(s).
         length   Get the length of FASTA record(s).
         content  Calculate the GC content of FASTA record(s).
         search   Find subseq in the FASTA record(s).

Author:  Yixf, yixf1986@gmail.com
Version: 20110601
Notes:   1. A similar webtool developed by lh3: http://lh3lh3.users.sourceforge.net/fasta.shtml
         2. When you search, you can use RegExp supported by Perl.
         3. The search function use case-sensitive mode.
```

FASTA | fetchGenomeSeq.pl

```
Program:    fetchGenomeSequence.pl (v20110514)
Author:     Yixf (xfyin@sibs.ac.cn)
Summary:    Fetch subsequence(s) from a specified genome according to coordinate(
s).

Usage:    fetchGenomeSequence.pl [OPTIONS]

Options:
        -g, --genome    Specify a genome.[default: hg19]
        -c, --coordinate    The coordinate of a region in "chr1:10-20" format.
        -i, --input    The input file containing many coordinates.
        -o, --output    The output file to save FASTA result(s).
        -h, --help    Print this help message.

Notes:
        1. You can use -c (coordinate for single mode) OR -i (input file for bat
ch mode), but not both at the same time.
        2. Unless you use -o, the results will be printed to the STDOUT[terminal
].
        3. If you use -i, please pay attention to:
                1) The delimiter in the file MUST be TAB.
                2) The file must have 3 columns at least.
                3) The first 3 columns MUST be "Chromosome,Start,Stop".
                4) Columns after the third column will be ignored.
                5) An example: "chr1\t10\t20".
        4. The coordinate uses 1-based system. (For more information, please ref
er to: http://yixf.name/2011/03/26/基因组的坐标系统：0-based与1-based/)
```

Applications for stand-alone use from UCSC

- faCount
- faOneRecord
- faPolyASizes
- faRandomize
- faSize
- faSomeRecords
- faToTwoBit
- fetchChromSize
- liftOver
- twoBitInfo
- twoBitToFa
- …

Outline

FASTQ | Toolkit

1. FASTX-Toolkit: FASTQ/A short-reads pre-processing tools
2. SolexaQA: At-a-glance quality assessment of Illumina second-generation sequencing data
3. Picard: A set of tools (in Java) for working with next generation sequencing data in the BAM format.
4. FastQC: A quality control tool for high throughput sequence data
5. PRINSEQ: A tool that generates summary statistics of sequence and quality data and that is used to filter, reformat and trim next-generation sequence data

FASTQ | SolexaQA

Tools/Softwares need you to know the FASTQ variants.

Which variant my FASTQ file belongs to?

I don't know …Check and guess by myself?

Don't worry! SolexaQA can detect it automatically!

```
$SolexaQA.pl SRX003920.fastq
Automatic format detection: Sanger FASTQ format
...
$SolexaQA.pl illumina.fq
Automatic format detection: Illumina FASTQ format, Illumina
      pipeline 1.3+
...
```

FASTQ | SolexaQA

Tools/Softwares need you to know the FASTQ variants.

Which variant my FASTQ file belongs to?

I don't know …Check and guess by myself?

Don't worry! SolexaQA can detect it automatically!

```
$SolexaQA.pl SRX003920.fastq
Automatic format detection: Sanger FASTQ format
...
$SolexaQA.pl illumina.fq
Automatic format detection: Illumina FASTQ format, Illumina
      pipeline 1.3+
...
```

FASTQ | SolexaQA

Tools/Softwares need you to know the FASTQ variants.

Which variant my FASTQ file belongs to?

I don't know …Check and guess by myself?

Don't worry! SolexaQA can detect it automatically!

```
$SolexaQA.pl SRX003920.fastq
Automatic format detection: Sanger FASTQ format
...
$SolexaQA.pl illumina.fq
Automatic format detection: Illumina FASTQ format, Illumina
      pipeline 1.3+
...
```

FASTQ | SolexaQA

Tools/Softwares need you to know the FASTQ variants.

Which variant my FASTQ file belongs to?

I don't know …Check and guess by myself?

Don't worry! SolexaQA can detect it automatically!

```
$SolexaQA.pl SRX003920.fastq
Automatic format detection: Sanger FASTQ format
...
$SolexaQA.pl illumina.fq
Automatic format detection: Illumina FASTQ format, Illumina
      pipeline 1.3+
...
```

FASTQ | SolexaQA

Tools/Softwares need you to know the FASTQ variants.

Which variant my FASTQ file belongs to?

I don't know …Check and guess by myself?

Don't worry! SolexaQA can detect it automatically!

```
1 $SolexaQA.pl SRX003920.fastq
2 Automatic format detection: Sanger FASTQ format
3 ...
4 $SolexaQA.pl illumina.fq
5 Automatic format detection: Illumina FASTQ format, Illumina
        pipeline 1.3+
6 ...
```

Bio::SeqIO::fastq (CPAN)

```
1  use Bio::SeqIO::fastq;
2
3  my $in = Bio::SeqIO->new(
4   -format  => 'fastq',
5   -variant => 'illumina',
6   -file => 'in.fq'
7  );
8
9  my $out = Bio::SeqIO->new(
10  -format  => 'fastq',
11  -variant => 'sanger',
12  -file    => '>out.fq'
13 );
14
15 while ( my $seq = $in->next_seq ) {
16  $out->write_seq($seq);
17 }
```

Outline

FASTX | FASTQ ⇒ FASTA

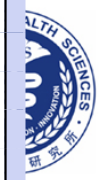
1. FASTX-Toolkit: FASTQ/A short-reads pre-processing tools
2. Bio::SeqIO::fastq (CPAN)

```
1  use Bio::SeqIO::fastq;
2
3  my $in = Bio::SeqIO->new(
4  -format => 'fastq',
5  -file => 'in.fq'
6  );
7
8  my $out = Bio::SeqIO->new(
9  -format => 'fasta',
10 -file   => '>out.fa'
11 );
12
13 while ( my $seq = $in->next_seq ) {
14 $out->write_seq($seq);
15 }
```

FASTX | FASTA + QUAL ⇒ FASTQ

Bio::Seq::Quality (CPAN); Merging sequence and quality to FASTQ

```
1  use Bio::SeqIO;
2  use Bio::Seq::Quality;
3
4  die "pass a fasta and a fasta-quality file\n" unless @ARGV;
5
6  my ( $seq_infile, $qual_infile ) =
7   ( scalar @ARGV == 1 ) ? ( $ARGV[0], "$ARGV[0].qual" ) : @ARGV;
8  my $in_seq_obj = Bio::SeqIO->new(
9     -file   => $seq_infile,
10    -format => 'fasta',
11 );
12 my $in_qual_obj = Bio::SeqIO->new(
13    -file   => $qual_infile,
14    -format => 'qual',
15 );
16 my $out_fastq_obj = Bio::SeqIO->new( -format => 'fastq' );
17
18 while (1) {
19    my $seq_obj = $in_seq_obj->next_seq || last;
20    my $qual_obj = $in_qual_obj->next_seq;
21    die "foo!\n" unless $seq_obj->id eq $qual_obj->id;
22    my $bsq_obj = Bio::Seq::Quality->new(
23        -id   => $seq_obj->id,
24        -seq  => $seq_obj->seq,
25        -qual => $qual_obj->qual,
26    );
27    $out_fastq_obj->write_fastq($bsq_obj);
28 }
```



Outline

Summary

The BEDTools utilities allow one to address common genomics tasks such as finding feature overlaps and computing coverage. The utilities are largely based on four widely-used file formats: BED, GFF/GTF, VCF, and SAM/BAM.

Reference

1. bedtools (Homepage)
2. BEDTools-User-Manual.v4.pdf
3. BEDTools: a flexible suite of utilities for comparing genomic features

Outline

GFF | GffTools

Alert

The tools are now rather outdated.

Reference

1. GFF tools
2. gfftools
3. Josep Abril's GFF programs (IMIM, Spain)

Outline

Summary

VCFtools - 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 methods for working with VCF files: validating, merging, comparing and calculate some basic population genetic statistics.

Reference

1. VCFtools
2. The variant call format and VCFtools

Summary

bcftools - Utilities for the Binary Call Format (BCF) and VCF

Reference

1. BCFTOOLS COMMANDS AND OPTIONS
2. Calling SNPs/INDELs with SAMtools/BCFtools

Outline

Summary

SAM Tools provide various utilities for manipulating alignments in the SAM format, including sorting, merging, indexing and generating alignments in a per-position format.

Reference

1. SAMtools
2. SAM tools
3. The Sequence Alignment/Map format and SAMtools
4. Bio-SamTools (Perl)
5. Pysam (Python)
6. Samtools-Ruby (Ruby)

Outline

BAM | BAMtools

Summary

BamTools provides both a programmer's API and an end-user's toolkit for handling BAM files.

Reference

1. bamtools
2. BamTools: a C++ API and toolkit for analyzing and managing BAM files

Outline

Tools | Others

1. Applications for stand-alone use from UCSC
2. SRA Toolkit
3. 进制转换的小程序 radixConvert.pl
4. Code collection

Tools | Version

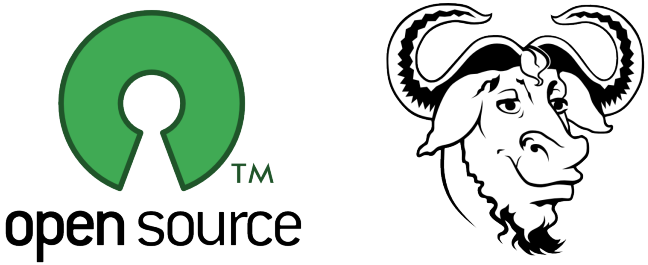
```
$exonerate -v
exonerate from exonerate version 2.2.0
Using glib version 2.4.7
Built on Oct 17 2008
Branch: unnamed branch

$fastq_to_fasta -h
Part of FASTX Toolkit 0.0.13 by A. Gordon (gordon@cshl.edu)

$intersectBed
[bedtools]
Program: intersectBed (v2.13.3)
Author:  Aaron Quinlan (aaronquinlan@gmail.com)
Summary: Report overlaps between two feature files.

$vcftools
VCFtools (v0.1.7)
© Adam Auton 2009

$bcftools
Program: bcftools (Tools for data in the VCF/BCF formats)
Version: 0.1.17-dev (r973:277)

$samtools
Program: samtools (Tools for alignments in the SAM format)
Version: 0.1.18 (r982:295)

$bamtools -v
bamtools 1.0.2
Part of BamTools API and toolkit
Primary authors: Derek Barnett, Erik Garrison, Michael Stromberg
(c) 2009-2011 Marth Lab, Biology Dept., Boston College

$fastq-dump -V
[SRA Toolkit]
fastq-dump : 2.1.7
```

Many hands make light work.

Please check if there is a update for the software before you use it!


INSTITUTE OF HEALTH SCIENCES
IHS
RESEARCH EDUCATION INNOVATION

Powered by



TEX LATEX XETEX Beamer

Thanks for your attention!

Any questions?