Essential File Formats in Genomics

Michael C. Zody, Ph.D.

Workshop on Genomics

Český Krumlov

January 7, 2025

Fasta

- Used for: sequences of nucleotides (DNA or RNA) *or* proteins
- Written by: many programs
- Read by: many programs
- Visualizing by eye: less (more), any text processor
- Header info: per sequence entry, minimal global standard
- Extension: .fasta, .fa, .fna, .fsa, and more
- Compression: no specific compression (gzip or bzip)
- Indexing: no standard indexing (but see .dict or .fai)

Looking at Fasta

- First:

  cd workshop_materials/file_types

  ls

- Look at the human genome file

  less human_g1k_v37.fa

  /> inside less will search for lines that start with >

  q inside less will exit less

- A clever trick to see all the headers:

  grep '>' human_g1k_v37.fa (Ctrl-C to stop) *OR*

  cat human_g1k_v37.fa | grep '>'

Other Fasta Files

- Look at some fasta for yeast:

  less yeast/GCF_000146045.2_R64_genomic.fna

  less yeast/S288C_reference_sequence_R64-5-1_20240529.fsa

IUPAC Codes (Uncertain Bases)

IUPAC nucleotide code	Base
A	Adenine
C	Cytosine
G	Guanine
T (or U)	Thymine (or Uracil)
R	A or G
Y	C or T
S	G or C
W	A or T
K	G or T
M	A or C
B	C or G or T
D	A or G or T
H	A or C or T
V	A or C or G
N	any base
. or -	gap

Compressing Fasta

- No data-specific compression format
- Typically use gzip
- Some programs will read gzipped fasta on the fly, others not
- Some versions of less will work on compressed data, others not (the Ubuntu installation on your VMs does not)
- Can use zcat file | less to view without gunzipping

Indexing Fasta

- No generalized indexing for fasta
- Samtools will make a .fai file that is used like an index
- GATK does something similar but calls it a .dict file
- These are used in SAM alignment format
- We can look at one of these

  less human_g1k_v37.fa.fai
- We can also look at the raw characters

  hexdump -c human_g1k_v37.fa.fai | less

Qual

- Used for: storing base quality for Sanger reads (archaic)
- Written by: phred (some others)
- Read by: phrap, others
- Visualizing by eye: less (more), any text processor
- Header info: per sequence entry, minimal global standard
- Extension: .qual (usually)
- Compression: no specific compression
- Indexing: no standard indexing

Quality Scores

- Also known as Q scores or phred scores
- First introduced in the ‘90s by Phil Green and Brent Ewing in phred
- Gives an estimate of the probability that a given base is wrong
- $Q = -10 \log_{10} (P(error))$
  - 10% (0.1) chance of error, Q = 10
  - 1%, Q = 20
- To convert Q to P(error), $P(e) = 10^{-Q/10}$ or 1 in $10^{Q/10}$
- Tells you nothing about what the base should be instead
- Q < 6 would be worse than random, typically used conventionally

Understanding Quality Scores

- Why quality scores?
- What does a quality score mean?
- Recalibrating quality scores
- Do we still need quality scores?

Fastq

- Used for: storing nucleotide sequences together with their quality
- Written by: vendor base calling software, some 3rd party programs
- Read by: aligners, assemblers
- Visualizing by eye: less (more), any text processor
- Header info: per sequence entry, minimal global standard
- Extension: .fq, .fastq
- Compression: no specific compression
- Indexing: no standard indexing

ASCII

32	SPC	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	DEL

Fastq encoding char = chr(Q + offset)

Q = ascii(char) - offset

Issues with Fastq

- No centrally maintained formal specification
- No header or versioning
- Designed for very short read data
  - Sequence and qual each fit on one line
  - Now used for much longer sequences and uncertain if these can span lines
  - Cannot strictly count on each sequence being in a block of 4 lines
- The start symbols for the read and qual headers can legitimately appear as the first character in a quality line using offset 33

SAM (Sequence/Alignment Map)

- Used for: storing aligned reads with their sequences and alignments
- Written by: aligners
- Read by: base callers, counting tools (RNA, ChIP, etc.), visualization
- Visualizing by eye: less (more), samtools, GATK, IGV (browsers)
- Header info: global, describes the reference sequence, read groups, read alignment and processing history
- Extension: .sam
- Compression: BAM and CRAM (to be discussed)
- Indexing: only indexed in binary format using samtools/GATK
- https://samtools.github.io/hts-specs/SAMv1.pdf

Look at a SAM File

- We can look at a SAM file here:

  less 3regs.final.sam

- Header
- Sequence lines
- Cigar string (see MD tag)
- Bit flags
- Optional fields

Bit Flags

- A bit flag is a way to densely store Boolean information (true/false)
- True/false (T/F) can be represented as 1 or 0 in one bit
- Computer integers consist of a series of bits representing powers of 2
- 10110101 = 1x128 + 1x32 + 1x16 + 1x4 + 1x1 = 181 (in decimal)
- We also say the first, third, fifth, sixth, and eighth bits are set true
- Example is the file permissions settings
  - read = 4, write = 2, exec = 1
  - chmod 754 would mean user can do all 3, group can read & exec, others read
- SAM/BAM format uses the bit flag to store Boolean information
- Endianess (big endian, little endian)

Look at a BAM File

- We cannot directly view a BAM file, because it is compressed
  - If you try, less will give you a warning

  less 3regs.final.bam
- We can look at a BAM file using samtools:

  samtools view -h 3regs.final.bam | less
- Headers and piping
- Can also use the index to skip directly to a spot in the file:

  samtools view 3regs.final.bam 6 | less
- Extension: .bam, index can be either .bai or .bam.bai

Look at a CRAM File

- We cannot directly view a CRAM file either
- We can look at a CRAM file using samtools, but we have to provide a reference genome

  samtools view -h -T human_g1k_v37.fa 3regs.final.cram | less

- Reference-based compression
- Extension: .cram, index .crai or .cram.crai
- Look at the sizes of the files:

  ll 3regs*am

VCF (Variant Call Format)

- Used for: storing variant calls on a reference genome
- Written by: variant calling and filtering programs
- Read by: analysis programs
- Visualizing by eye: less (more), bcftools
- Header info: global header that describes the data in the file
- Extension: .vcf, index .vcf.tbi or .vcf.idx
- Compression: no specific compression, gzip or bzip
- Indexing: generally indexed with tabix (in raw or compressed form)

Look at some VCFs

- We will start with the VCF matching the SAM we were looking at:

  less 3regs.final.vcf

- Header and info in header
- Command line
- Fields line
- Sample listings

Other VCFs

- We have the same sample from IGSR for 1000 Genomes 30x
  - It is gzipped, so we can use bcftools to view it

  bcftools view chr17.raw.subset.vcf.gz | less
- Note the more detailed header
- There is more information in the info field, sample info
- We can also use bcftools to look at a subset of the file
- We also have the phased version of these calls with SVs:

  bcftools view chr17.subset.vcf.gz | less
- And we can use VCF without any calls

  bcftools view dbsnp_138.b37.vcf.gz | less

Replacing VCF?

- The main issue with VCF is that it is BIG, especially for large cohorts
- Also, most variants in large cohorts are rare
- All of Us is moving to using VDS (Variant Data Set) instead
- VDS stores data sparsely, keeping data only for the alleles present in a given sample, and compressing long runs of reference into blocks
- However, currently VDS is only usable by Hail
- VDS can be “densified” into VCF (or Hail MT), but the assumption is people would only do that for a subset of the data at a time

GTF (Gene Transfer Format)

- Used for: storing gene annotations on a reference genome
- Written by: gene callers, database output
- Read by: analysis and visualization programs
- Visualizing by eye: less (more)
- Header info: global header that describes the data in the file
- Extension: .gtf
- Compression: no specific compression, gzip or bzip
- Indexing: none

GFF (General Feature Format)

- Used for: storing annotations on a reference genome
- Written by: feature identification programs, databases
- Read by: analysis and visualization programs
- Visualizing by eye: less (more)
- Header info: global header that describes the data in the file
- Extension: .gff, .gff3
- Compression: no specific compression, gzip or bzip
- Indexing: none

Looking at GTF and GFF

- We have both GTF and GFF of the same files for the S288C reference genome for S. cerevisiae:

  less yeast/genomic.gtf

  less yeast/genomic.gff

BED (Browser Extensible Data)

- Used for: storing generic features on a reference genome
- Written by: multitudes
- Read by: multitudes
- Visualizing by eye: less (more)
- Header info: formally none, optional comment lines
- Extension: .bed (legacy .bed*n* where *n* is the number of columns)
- Compression: no specific compression, gzip or bzip
- Indexing: none (tabix?)
- https://samtools.github.io/hts-specs/BEDv1.pdf

Looking at BED

- We have three annotation files in BED format, all in tracks:

  less tracks/PB_indels.bed

  less tracks/NA12878_lumpy_validated.deletions.sorted.bed

  less tracks/3regs.rm.bed

Coordinate Systems

- Chromosome coordinates are numbered from start to end
- Biologists typically use 1 based coordinates to count bases
- Computer scientists typically use 0 based coordinates to count arrays
- Many problems results from off by one or fence post errors when these coordinate systems do not agree between input and expected
- BED uses 0-based half-open coordinates
  - The first position is the start in 0-based coordinates
  - The second position is the position one past the end in 0-based coordinates

Whitespace, Line Feeds, and Genomics

- Even text files contain non-printing characters that format the text
- Whitespace includes spaces (“ “) and tabs (\t)
  - A tab creates one or more spaces depending on the typesetting
  - They look the same, but the computer treats them as separate characters
  - Some programs interpret them the same, but most do not
- End of line is marked by either newline (\n) or carriage return (\r)
  - Most viewers will accept either or both
  - Some programs care which you have
- Modern (Unicode) text contains many characters outside of ASCII
  - Dashes (– - —), directional quotes (“ ”), etc.
  - These will not necessarily convert to the correct ASCII character

Do Not Use Excel (and Be Wary of Word)

COMMENT Open Access

Gene name errors are widespread in the scientific literature

Mark Ziemann[1], Yotam Eren[1,2] and Assam El-Osta[1,3*]

Abstract

The spreadsheet software Microsoft Excel, when used with default settings, is known to convert gene names to dates and floating-point numbers. A programmatic scan of leading genomics journals reveals that approximately one-fifth of papers with supplementary Excel gene lists contain erroneous gene name conversions.

Keywords: Microsoft Excel, Gene symbol, Supplementary data

Abbreviations: GEO, Gene Expression Omnibus; JIF, journal impact factor

frequently reused. Our aim here is to raise awareness of the problem.

We downloaded and screened supplementary files from 18 journals published between 2005 and 2015 using a suite of shell scripts. Excel files (.xls and.xlsx suffixes) were converted to tabular separated files (tsv) with ssconvert (v1.12.9). Each sheet within the Excel file was converted to a separate tsv file. Each column of data in the tsv file was screened for the presence of gene symbols. If the first 20 rows of a column contained five or more gene symbols, then it was suspected to be a list of gene symbols, and then a regular expression (regex) search of the entire column was applied to identify gene symbol errors. Official gene symbols from Ensembl ver-

Visualizing Files with IGV

- Most of the files we have looked at can be examined visually
- IGV will load most of these file types
- Log in to your desktop on guacamole
- Launch IGV from the desktop (icon should be near the middle left)
- In the upper right of IGV, change the genome to hg19
- Under File, choose Load from File…, navigate to the file_types dir
  - Load 3regs.final.bam
  - Load 3regs.final.vcf
  - Load tracks/3regs.rm.bed
- To see reads, navigate to 1:246,155,000 (or 6:157,943,000 or 20:61,725,000)

MAF (Multiple Alignment Format)

- Used for: storing alignments between more than one large sequence
- Written by: multiple aligners, alignment post-processors
- Read by: analysis and visulation programs
- Visualizing by eye: less (more), browsers
- Header info: version and scoring
- Extension: .maf
- Compression: no specific compression, gzip or bzip
- Indexing: none

Look at a MAF

- We have a 7-way alignment from UCSC

  less hg38.7way.maf

Other Formats for Multiple Alignments

- Clustal
- Aligned Fasta
- Phylip format
- Chain and net

Raw Sequence Data Formats

- BCL
  - Illumina files with raw signal intensities
  - Since HiSeq X these have not been accessible to users
- HDF5 (Hierarchical Data Format version 5)
  - Generic scientific data storage format, extensible and customizable
  - Like a database organized like a file system inside a file
  - Designed for very large data stored together but randomly accessed
  - The ONT fast5 format is also a derivative of HDF5
- Pod5
  - New ONT format
  - Custom binary file designed to be streamed into efficiently