Complete Genomics Analysis Tools (cgatools) 1.0.0 Methods

Table of Contents

Version 1.0.0

Introduction

The Complete Genomics Analysis Tools (cgatools) is an open source project to provide tools for downstream analysis of Complete Genomics data. This document describes the motivation and design decisions for cgatools.

Reference Tools

Cgatools generally needs access to the reference sequence. At times, an API to access the reference sequence at a random location in the genome is required or desired. The most common reference sequence format, FASTA, is not ideal for processing tasks that require a randomly accessible reference because the entire sequence must be read into memory at the start of the program, and this memory cannot be shared amongst processes.

Cgatools uses its own file format, the Compact Randomly Accessible Reference (CRR) file format, to represent a reference sequence. The CRR file format stores two bits per base of reference, plus lookup tables to resolve regions of the reference that are represented by ambiguous IUPAC codes. CRR files are memory mapped, so that processes can share a reference, and the overall memory requirement due to the reference for all processes is less than 1GB. The CRR file format does not preserve case (FASTA reference files downloaded from UCSC use case to denote the region's repeat status.), and considers all the bases described in the reference FASTA sequence as upper case.

The FASTA sequences for the human reference, NCBI build 36, can be downloaded from UCSC http://hgdownload.cse.ucsc.edu/goldenPath/hg18/bigZips/chromFa.zip (899MB). That is the reference currently used by Complete Genomics variant files and mapping files. Once downloaded, they can be converted into a single reference CRR file using the fasta2crr command. Once you have successfully created a human NCBI build 36 CRR file for use with Complete Genomics data, the listcrr command will return the following output:

Version 1.0.0

ChromosomeId	Chromosome	Length	Circular	Md5
0	chr1	247249719	false	9ebc6df9496613f373e73396d5b3b6b6
1	chr2	242951149	false	b12c7373e3882120332983be99aeb18d
2	chr3	199501827	false	0e48ed7f305877f66e6fd4addbae2b9a
3	chr4	191273063	false	cf37020337904229dca8401907b626c2
4	chr5	180857866	false	031c851664e31b2c17337fd6f9004858
5	chr6	170899992	false	bfe8005c536131276d448ead33f1b583
6	chr7	158821424	false	74239c5ceee3b28f0038123d958114cb
7	chr8	146274826	false	1eb00fe1ce26ce6701d2cd75c35b5ccb
8	chr9	140273252	false	ea244473e525dde0393d353ef94f974b
9	chr10	135374737	false	4ca41bf2d7d33578d2cd7ee9411e1533
10	chr11	134452384	false	425ba5eb6c95b60bafbf2874493a56c3
11	chr12	132349534	false	d17d70060c56b4578fa570117bf19716
12	chr13	114142980	false	c4f3084a20380a373bbbdb9ae30da587
13	chr14	106368585	false	c1ff5d44683831e9c7c1db23f93fbb45
14	chr15	100338915	false	5cd9622c459fe0a276b27f6ac06116d8
15	chr16	88827254	false	3e81884229e8dc6b7f258169ec8da246
16	chr17	78774742	false	2a5c95ed99c5298bb107f313c7044588
17	chr18	76117153	false	3d11df432bcdc1407835d5ef2ce62634
18	chr19	63811651	false	2f1a59077cfad51df907ac25723bff28
19	chr20	62435964	false	f126cdf8a6e0c7f379d618ff66beb2da
20	chr21	46944323	false	f1b74b7f9f4cdbaeb6832ee86cb426c6
21	chr22	49691432	false	2041e6a0c914b48dd537922cca63acb8
22	chrX	154913754	false	d7e626c80ad172a4d7c95aadb94d9040
23	chrY	57772954	false	62f69d0e82a12af74bad85e2e4a8bd91
24	chrM	16571	true	d2ed829b8a1628d16cbeee88e88e39eb

Fasta2ccr

This tool converts FASTA sequence files to the CRR file format. See the Reference Tools section above for further description on the CRR file format.

Crr2fasta

This tool converts CRR sequence files to the FASTA file format. See the Reference Tools section above for further description on the CRR file format.

Listcrr

This command lists the chromosomes, contigs, or regions of ambiguous sequence within the reference, depending on the parameters. The contigs described by listcrr are defined to be the contiguous sequence bases separated by at least `min-contig-gap-length` no-call bases, where `min-`

`contig-gap-length` defaults to 50. The default contigs correspond to the notion of contig employed in the Complete Genomics data, such as reference scores.

Decodecrr

This command quickly retrieves the sequence for a given range.

Genome Comparison Tools

A Note on Conventions

To call low certainty regions or "no-call" regions, Complete Genomics augments the alphabet {A, C, G, T} with two additional characters: "N" and "?". The "N" character corresponds to a one-base sequence that may be any of { A, C, G, T }. The "?" character corresponds to zero or more bases of unknown sequence.

Introduction

Genome comparison is the problem of identifying genomic sequence that is identical, compatible (perhaps with no-calls), or incompatible, with sequence from another genome. Within the space of genome comparison problems, there are three common tasks:

1. Is a genome identical, compatible, or incompatible with the reference genome at a given location?
2. Is a genome identical, compatible, or incompatible with a known common sequence?
3. Is a genome identical, compatible, or incompatible with a particular genome at a given location within the reference genome?

The particular way a genome is described by re-sequencing technologies goes a long way towards solving genome comparison problems 1 and 2: genomes are represented as a set of differences (or variants) against the reference genome. The Complete Genomics variant file format differs from most other common variant file formats in that in addition to describing the variants, it also distinguishes regions of the genome that are called as reference from those that are no-called. As we will see later, this distinction is essential in solving many comparison problems.

Let's see how the Complete Genomics variant file would describe the following situation, where chr1 is a diploid chromosome and chr2 is a haploid chromosome:

```
chr1 reference:      CATGACCCGCAAA-TCTGAAACTATCTGGCCCTTGGCAGGGG--A
chr1 haplotype 1:    ?ATGACCTGCAAAATCTGAAACT--CTGGCCCTTGGCAGGGGGGA
chr1 haplotype 2:    ?ATGACCCGCAAAATCTGAAACTATCTGGCTNTTGGCAGGGT--A

chr2 reference:      TGATATTTTTCATCAACATTACAGGCA
chr2:                TGATATTTTTNATCAACACGACAGGCA
```

Here is the corresponding variant file:

>locus	ploidy	allele	chromosome	begin	end	varType	reference	alleleSeq	totalScore	hapLink	xRef
1	2	all	chr1	0	1	no-call	=	?			
2	2	all	chr1	1	7	ref	=	=			
3	2	1	chr1	7	8	snp	C	T	87	1	dbsnp:123
3	2	2	chr1	7	8	ref	C	C	58	2	dbsnp:123
4	2	all	chr1	8	13	ref	=	=			
5	2	1	chr1	13	13	ins		A	36		
5	2	2	chr1	13	13	ins		A	42		
6	2	all	chr1	13	22	ref	=	=			
7	2	1	chr1	22	24	del	AT		47	1	
7	2	2	chr1	22	24	ref	AT	AT	55	2	
8	2	all	chr1	24	29	ref	=	=			
9	2	1	chr1	29	31	ref	CC	CC	57	1	
9	2	2	chr1	29	31	no-call-ri	CC	TN	65	2	
10	2	all	chr1	31	40	ref	=	=			
11	2	1	chr1	40	41	ref	G	G	101	1	
11	2	1	chr1	41	41	ins		GG	120	1	
11	2	2	chr1	40	41	snp	G	T	479	2	
12	2	all	chr1	41	42	ref	=	=			
13	1	all	chr2	0	10	ref	=	=			
14	1	1	chr2	10	11	no-call-rc	C	N	47		
15	1	all	chr2	11	18	ref	=	=			
16	1	1	chr2	18	20	sub	TT	CG	102		
17	1	all	chr2	20	27	ref	=	=			

The genome is first aligned to the reference, and then split into loci. Each locus may describe multiple alleles (if ploidy > 1), and for each allele at each locus, there may be one or more lines (or “calls”) to describe the sequence. The variant file describes 0-based offsets within the reference chromosome.

In the above variant file, locus 3 describes a heterozygous SNP (one-base polymorphism on one allele, reference on the other allele). Locus 5 describes a homozygous insertion (in which the confidence is slightly higher for allele 2 than allele 1). The allele column is used to distinguish the alleles of calls within a locus. For example, the “ref” and “ins” calls of locus 11 are on the same haplotype, whereas the “snp” call is on the opposite haplotype. To declare that two calls of different loci are on the same haplotype, the hapLink field is used. Calls known to be on the same haplotype have the same hapLink value; calls with different hapLink values may or may not be on the same haplotype (the phasing is no-called). For a

detailed reference of the Complete Genomics variant file format, see the Date File Format document provided with your genome.

Problems not solved by variant file format

One problem you may have noticed is that the problem of aligning a genome to the reference is not necessarily well-defined. For example, the homozygous insertion at locus 5 could have also been described by the same homozygous insertion three bases to the left. Or the substitution at locus 16 could have been described as two SNPs. Comparing two genomes that describe the same sequence in different ways can be tricky.

We could make canonicalization rules such as "always use the rightmost insertion for any insertion that has multiple possible representations" or "always decompose an allele consisting of a SNP, two reference bases, then another SNP, into separate calls." Indeed, Complete Genomics has rules like these that are generally followed. But there are at least three remaining problems in solving the genome comparison problems described above:

- Known variants are not always described in their canonical form. For example, entries rs34330821 and rs34544546 in the dbSNP database of known variants describe equivalent insertions that are 18 bases apart. This may seem superficial, in that dbSNP entries that are not described in their canonical form can be canonicalized. But if our canonical form uses less decomposition than the dbSNP submission, this may not be possible; if a dbSNP submission has been decomposed, the submission has lost information about nearby variants that exist on the same haplotype.
- Canonical forms of near-identical sequences are not necessarily near-identical. For example, suppose we have a genome that is equivalent to a SNP and an insert against the reference, as described in canonicalization 1 below:

  ```
  Reference:          TG A TGTGAATTGGTG -------------------------- AGT
  Canonicalization 1: TG C TGTGAATTGGTG TAGTGTGAATGAGTGTGTGAATTGGTG AGT

  Reference:          TG A-------------------------- TGTGAATTGGTGAGT
  Canonicalization 2: TG CTGTGAATTGGTGTAGTGTGAATGAGTG TGTGAATTGGTGAGT
  ```

  The insert in canonicalization 1 might be the simplest way to describe the genome if the SNP did not exist. But one could argue that the single substitution in canonicalization 2 is the simplest canonicalization of the genome, given that the SNP does exist. (This would be the case for a canonicalization which favors fewer calls.) It is not obvious by visual inspection that the insert from canonicalization 1 and the substitution of canonicalization 2 differ by only a SNP.

- No-calls may not be canonicalized like insertions or deletions, such that an insert may be compatible with another genome only when viewing a larger sequence of the genome. For example, suppose we have the following reference and the following genome:

Reference: CGAAAAAAA-TTTTCG
Genome: CGAAAAAAAATTTTCG

Now suppose the genome reconstruction process discovers that an insertion has occurred, but it does not know if the first base in the run of A's is really an A, or perhaps was a C. In this case, we are forced to align the no-call at the beginning as follows:

Reference: CG-AAAAAAATTTTCG
Genome: CGNAAAAAAATTTTCG

Length no-calls ("?") may further complicate the situation so that the alignment is unclear. For example, suppose in the same example above, in addition to not knowing if the first base of the run is an A or a C, we also don't know the length of the run of A's at all. Suppose also that we know that the run of T's has increased in length from four to five. There could be at least two reasonable alignments of the result, corresponding to a called insert or a called SNP:

Reference: CGAAAAAAATTTT-CG
Alignment 1: CG?AAAAAATTTTTCG

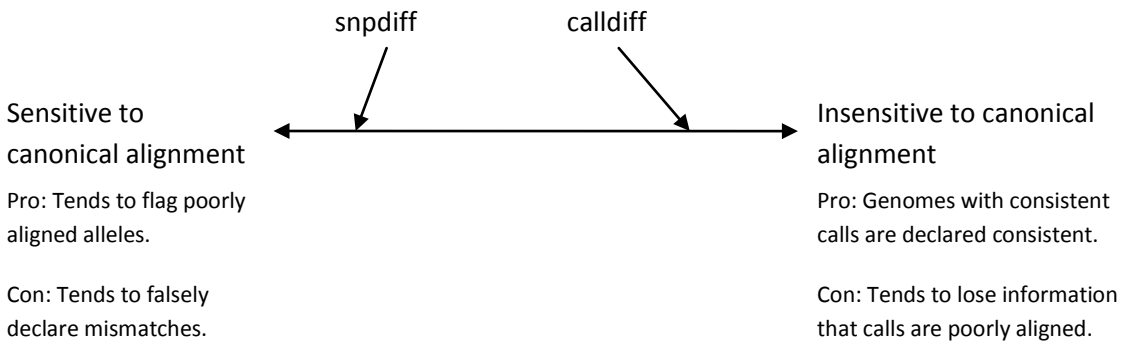
Reference: CG-AAAAAAATTTTCG
Alignment 2: CG?AAAAAATTTTTCG

Genome comparison with cgatools

There is a wide spectrum of useful genome comparison methods, which range in their sensitivity to the canonical alignment of called sequence. Algorithms that are very sensitive to canonical alignment tend to declare sequences inconsistent when in fact they are consistent. Algorithms that are less sensitive to canonical alignment tend to be less discriminating in terms of the quality of the alignment of called sequence.

Cgatools includes two genome comparison utilities, which provide varying degrees of sensitivity to inconsistent canonical alignments (Figure 1). The snpdiff tool can be used to compare the results of a SNP caller to a Complete Genomics variant file. It is quite sensitive to the canonical alignment of called sequence. The calldiff tool can be used to compare two variant files. It is less sensitive to the canonical alignment of called sequence.

Figure 1: Snpdiff and calldiff vary in their sensitivity to canonical alignment. Both algorithms have their merits.



Snpdiff

The snpdiff tool compares SNP calls to a Complete Genomics variant file. It is particularly useful for comparing Complete Genomics variant file to SNP calls provided by an alternative sequencing or genotyping platform that only produces SNP calls. The input SNP calls must be in a tab-delimited file with columns like the following:

Chromosome	Offset0Based	GenotypesStrand	Genotypes
chr13	17919222	+	CC
chr13	17919650	+	AT
chr13	17920392	+	NN
chr13	17921548	+	TT

Here, the “Genotypes” column specifies the base call for each allele. The output produced for this input may be something along these lines:

Chromosome	Offset0Based	GenotypesStrand	Genotypes	Reference	Variants	DiscordantAlleles	NoCallAlleles
chr13	17919222	+	CC	T	CC	0	0
chr13	17919650	+	AT	A	AA	1	0
chr13	17920392	+	NN	G	GN	0	1
chr13	17921548	+	TT	T	.-	0	0

The result for each allele described in the “Variants” column above are any base call (A, C, G, or T), a no-call (N), a deletion (-), or a larger variation that is not consistent with a SNP at all (.). In order to compare the SNP calls to the calls in the variant file, snpdiff first determines the variant file’s calls at the given position. The algorithm that is used is sensitive to the canonical alignment, and it is aggressive in terms of making a base call at positions where the call does not have a varType of “snp” or “ref”. That being

said, it is tested to be largely concordant with SNP calls made by several alternative technologies. A discordance found by snpdiff is likely to be a true discrepancy between the calls made by the SNP caller and the variant file. The algorithm employed by snpdiff is as follows, for each allele:

- Find the call in the variant file that overlaps the position in question. Use this call alone to determine the base call for the position in question.
- Walk the alleleSeq column of the call from the right and left until reaching the position in question. For each direction, any of the following outcomes may be reached:
    - WALK_OK – The position in question was reached.
    - WALK_EOS – The end of alleleSeq was reached before getting to the position in question.
    - WALK_INCOMPATIBLE – A base call incompatible with the reference base was found at some position before reaching the position in question.
    - WALK_LENGTH_NOCALL – A length no-call ("?") is discovered before reaching the position in question.
- Combine the results of the walk from the right and left to determine the result. The results are combined by the following rules:
    - If the walk from the left and right both end up at the position of interest (WALK_OK):
        - If the base calls discovered by the two walks are in conflict, declare a larger variation (".").
        - If the base calls discovered by the two walks are consistent, and at least one is called, use the base call.
        - If both walks end up with a no-call ("N"), the result is no-call.
    - If only one of the walks ends up at the position of interest (WALK_OK), use the base discovered by that walk.
    - If neither walk ends up at the position of interest, then:
        - If either walk ends up as WALK_LENGTH_NOCALL, mark the position as no-call ("N").
        - If either walk ends up as WALK_EOS, mark the position as deleted ("-").
        - Otherwise, mark the position as a larger variant (".").

Below are some examples (Table 1). The reference base we wish to determine a call for is highlighted in red:

reference	alleleSeq	Walk L->R	Walk R->L	Outcome
A	C	WALK_OK: C	WALK_OK: C	C
ACGTACGT	ACGTACGT	WALK_OK: T	WALK_OK: T	T
G	CC	WALK_OK: C	WALK_OK: C	C
G	CG	WALK_OK: C	WALK_OK: G	.
ACGT	AGGN	WALK_OK: G	WALK_OK: G	G
ACGT	AGG?	WALK_OK: G	WALK_LENGTH_NOCALL	G
ACGT	?GG?	WALK_LENGTH_NOCALL	WALK_LENGTH_NOCALL	N
ACGT	CGGT	WALK_INCOMPATIBLE	WALK_OK: G	G
ACGT	CGGG	WALK_INCOMPATIBLE	WALK_INCOMPATIBLE	.
CACACAC	CAC	WALK_EOS	WALK_EOS	-

Table 1 Example results from snpdiff

Calldiff

The calldiff tool compares two variant files. Its purpose is to determine where the two genomes differ, and how. For example, it can be used to help find potential somatic mutations in a tumor-normal comparison, or to find where two assemblies of the same genome differ. Calldiff is less sensitive than snpdiff to the canonical alignment. To achieve this, it first gathers variants into superloci, which may account for several nearby variants. It compares the genomes for each superlocus, then refines the comparison result to get call-level and locus-level detail.

If the superloci are too small, superlocus comparison tends to be overly sensitive to canonical alignment. But if superloci are too large, superlocus comparison tends to allow any sequence from one genome to match in a gap of unknown sequence in the other genome. As an example of a superlocus that is too large (Figure 2 , suppose we had the following sequence from a haploid chromosome of two genomes:

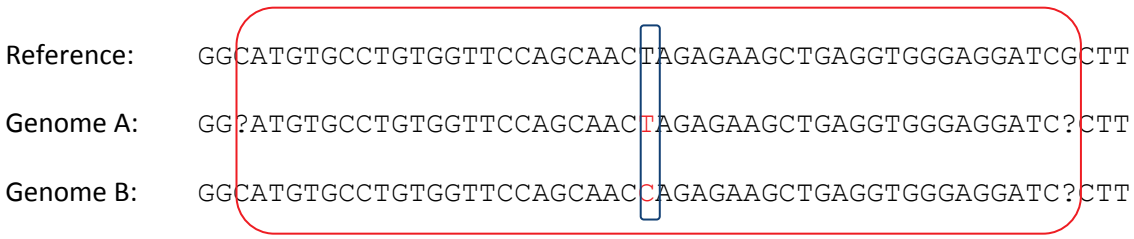


Figure 2: Example of a superlocus that is too large (red). Genomes A and B are consistent when considering the red superlocus as a whole, because the called sequence between the "?" characters in Genome A may be aligned to the "?" character of Genome B.

When considering the red superlocus in Figure 2, and when interpreting the meaning of the calls literally, we can see that all the called bases between the "?" characters in Genome A may be aligned to the "?" character of Genome B, and the genomes are consistent. But when considering the blue box to be the superlocus, we see that the genomes are inconsistent. In different contexts, one superlocus or the other may be preferable, but generally for most comparisons, we would want a comparison algorithm in this case to state the inconsistency between the genomes. In order to achieve this, a comparison algorithm must either be very precise about how to compare superloci (i.e., when using the red superlocus, determine that there is enough high complexity and uncommon sequence between the "?" characters in Genome A that the SNP in the middle must be aligned as called) or very precise about how to define a superlocus (i.e., always use the blue superlocus in this situation). Calldiff achieves its specificity by being precise about its superlocus definition.

To determine the superloci, calldiff begins by labeling each reference region containing a variant in either variant file as a superlocus. The superloci are then extended according to the following criteria:

1. **Circular prefix/suffix matching.** For every call whose alleleSeq does not contain "N" or "?", do prefix matching to the right along the reference and suffix matching to the left along the reference of both the alleleSeq and the reference sequence, such that the superlocus extension does not exceed P bases (the P limit is necessary to limit the superlocus size for pathological situations). For example, if the call is for an insertion of "ACGT" and the reference sequence directly to the right is "ACGA", three prefix bases of the alleleSeq can be matched to the reference sequence directly to the right, indicating that an equivalent insertion exists at each position in that range. So the superlocus must be extended to account for any variants within three bases to the right of the variant. Additionally, in the example above, if the sequence directly to the right of the call was ACGTACGA, then the entire insertion of four bases can be prefix matched, and continuing along the reference, the next three bases also match the prefix of the insertion. (This is circular prefix matching.) So the superlocus must be extended to the right by seven bases.
2. **Fixed base count.** Always extend superloci to the right and left by N bases, where N is a command-line configurable parameter.
3. **Fixed count of distinct 3-mers.** Always extend by M distinct reference 3-mers to the right and left, where M is a command-line configurable parameter. In regions of low reference sequence complexity, this results in longer superloci. In regions of high reference sequence complexity, this results in shorter superloci.

Once the superloci have been fully extended, overlapping and abutting superloci are combined into a single superlocus.

Once superloci have been found, all possible phasings consistent with the hapLinks in the calls are used to produce hypotheses about what the genome sequence is, for each variant file. Then each permutation of each hypothesis (one permutation for haploid, two for diploid, and six for triploid) is

compared to each hypothesis of the other variant file according to a literal interpretation of their sequence. In other words, any number of bases may align against length no-calls ("?"). The best comparison is produced, such that the number of discordant haplotypes is minimized. The alleles of the best comparison are then segmented to get call-level comparison results. The call-level comparison results are defined to be no worse than the result for the allele as a whole; if a segment comparison results in a worse comparison result than the allele as a whole, the allele's comparison result is used in its place. The call-level comparison results are then used to classify the comparison of each locus as a whole.

The results of all calldiff are, for each allele, a comparison classification (Table 2). The classifications are as follows:

Classification	Description
ref-identical	The alleles of the two variant files are identical, and they are consistent with the reference.
alt-identical	The alleles of the two variant files are identical, and at least one is inconsistent with the reference.
ref-consistent	The alleles of the two variant files are consistent, and they are consistent with the reference.
alt-consistent	The alleles of the two variant files are consistent, and at least one is inconsistent with the reference.
onlyA	The alleles of the two variant files are inconsistent, and only file A is inconsistent with the reference.
onlyB	The alleles of the two variant files are inconsistent, and only file B is inconsistent with the reference.
mismatch	The alleles of the two variant files are inconsistent, and they are both inconsistent with the reference.
phase-mismatch	The two variant files would be consistent if the hapLink field had been empty, but they are inconsistent.
ploidy-mismatch	The superlocus did not have uniform ploidy.

Table 2: Example results from calldiff

For non-haploid superloci, the comparison results for the alleles are joined by a semi-colon. For example, for a diploid hypothesis where variant file A calls reference and variant file B calls a het SNP, you might have a comparison result that looks like "ref-identical;onlyB".

For example, suppose we use calldiff to compare a normal genome (file A) and a tumor genome (file B) from the same individual. We can find purported somatic mutations by looking for "ref-identical;onlyB". We can find purported loss of heterozygosity (LOH) by looking for "ref-identical;onlyA" or "alt-identical;onlyB". We might expect fewer superloci classified as "alt-identical;onlyA", as the likely reason for this is assembly error – overcall in the normal genome.

Format Conversion Tools

The primary goal of the Complete Genomics export formats is to represent the data in a concise and simple way. As such, they are not always the best formats for doing certain kinds of data processing. Moreover, some users have existing programs that expect inputs in various other data formats. As a result, cgatools aims to provide data conversion capabilities.

Map2sam

The map2sam tool converts Complete Genomics exported reads and initial reference mappings to the SAM format. For pipelines that require eventually converting to the BAM format, the output of map2sam can be sent to standard output, which can be processed by SAM Tools. For example, this command pipeline creates an indexed, reference-sorted BAM file:

```
cgatools map2sam --reads=/path/to/reads.tsv.bz2 \
                 --mappings=/path/to/mappings.tsv.bz2 \
                 --library=/path/to/lib_DNB.tsv | \
    samtools view -uS - | \
    samtools sort - result && samtools index result.bam
```

Complete Genomics reads are initially mapped to the reference genome using a fast algorithm, and these initial mappings are later both expanded and refined by a form of local *de novo* assembly applied to putatively variant regions of the genome.

IMPORTANT: The map2sam tool converts the initial reference mappings, and not the additional mappings to variants discovered during the assembly process.

The following additional limitations apply to map2sam output:

- The converted mappings are reference mappings only. The mappings used as evidence to make indel calls are not included.
- SAM does not have strong support for overlapping sub-reads (e.g., the negatively sized intra-read gaps), which are present in Complete Genomics data. To represent overlapping reads, the strongest base call is put in the SAM mapping record, and the alternative base calls are represented using the GS/GQ/GC tags of the mapping record.
- The SAM validator provided by the Picard project (picard.sourceforge.net) does not allow specifying a primary mapping for reads that do not have consistent mates. As a result, only reads which have consistent mate pair mappings have a mapping marked as the primary mapping record (mapping record with the “not primary” FLAG, 0x0100, set to 0).
- The NM tag (edit distance to reference sequence) is not currently produced by map2sam.
- The R2 and Q2 tags (mate sequence and quality scores) can be generated optionally.

Representation of the Complete Genomics data in the SAM output

The detailed descriptions of the map2sam output below assume some familiarity with Complete Genomics data and terminology. We recommend you consult the Complete Genomics Data File Formats document and FAQs if you are mostly familiar with other Next-Gen platforms. These documents can be obtained from support@completegenomics.com. Additionally, the Complete Genomics assembly process and some of its implications are described in the Complete Genomics technology whitepaper (available from www.completegenomics.com) and in more detail the *Science* paper (Drmanac et al. *Science*, Jan 2010, which can be accessed at www.drmanac.com). We recommend you consult the Complete Genomics FAQ documents available from support@completegenomics.com in considering how to best use these data.

This description is based on the SAM Format Specification "Sequence Alignment/Map (SAM) Format", Version 0.1.2-draft (August 20, 2009) available at http://samtools.sourceforge.net/SAM1.pdf.

Header Fields:

Section	Tag	Value	Description	Example
@HD	VN	0.1.2	Version of SAM spec.	
	SO	"DnbId sorted"	Sort Order. Note: A DNB is a clone.	
@SQ	SN	<ChromosomeName>	Sequence Name. Included for all chromosomes in the reference genome	SN:chr1
	LN	<ChromosomeLength>	Same as above	LN:247249719
	UR	<ReferenceFilePath>	Path to the input reference file	UR: reference.crr
	AS	<ASM ID>	CG Data Analysis Pipeline Run ID	AS:GS19240-ASM
@RG	ID	<LaneId>	Slide and Lane ID	ID:GS08081-FS3-L02
	SM	<SampleId>	Sample Id	SM:GS00028-DNA-C01
	LB	<LibraryId>	Library ID of the library	LB:GS00433-CLS
	PU	<LaneId>	Slide and Lane ID	PU:GS08081-FS3-L02
	CN	"Complete Genomics"	Name of sequencing center producing the read	
	DT	<ExportDate>	The CG data analysis timestamp stored in the CG reads file	DT:2010-01-21
	PL	"Complete Genomics"	Platform/technology used to produce the read	
@PG	ID	"cgatools"	Program name	
	VN	<version>	Program version	VN:0.5.0
	CL	<command line>	Command line	Complete string

Mapping Record Fields:

Field	Value	SAM Definition	map2sam Usage
QNAME	<SlideId>- <LaneId>: <DnbOffest>	Query Name	The QNAME value is constructed from the full lane Id (SlideId+LaneId) and 0-based DNB offset from the beginning of the Reads file provided as input. For example: GS08081-FS3-L02-3:244
FLAG	0x0001	the read is paired in sequencing	The flag is always set for CGI data. The current CGI technology always produces paired reads.
	0x0002	the read is mapped in a proper pair	There are two possible behaviors: 1) The flag is set only if the other HalfDNB is mapped consistently i.e. within the valid range of the mate gap and on the same strand as the current HalfDnb; 2) The case is similar to the previous one but also includes the situation when the other HalfDNB is mapped in a random place but both HalfDNBs have unique mappings within the whole genome. In that case the mate HalfDNB will be treated as a mate even though it is not consistent.
	0x0004	the query sequence itself is unmapped	the flag is set when there are absolutely no mappings found for this HalfDNB
	0x0008	the mate is unmapped	the flag is set only when there are no mappings found for this HalfDNB's mate
	0x0010	strand of the query	0 for forward; 1 for reverse strand
	0x0020	strand of the mate	0 for forward; 1 for reverse strand
	0x0040	the read is the first read in a pair	The flag is set if the current HalfDNB is from the 5' end of the original cloned insert.
	0x0080	the read is the second read in a pair	The flag is set if the current HalfDNB is from the 3' end of the original cloned insert.

	0x0100	the alignment is not primary	The flag is set if there is a better mapping of the same HalfDNB having higher value of MAPQ (see the full description below) or the other HalfDNB is not mapped
	0x0200	the read fails platform/vendor quality checks	always set to 0
	0x0400	the read is either a PCR duplicate or an optical duplicate	always set to 0
RNAME	<ChromosomeId> or "*"	Reference sequence NAME	Can be "*" if this HalfDNB doesn't have mappings
POS	<Current Mapping Position> or 0	1-based leftmost POSition/coordinate of the clipped sequence	The position reported in a Mappings file record from a CGI export package offset by 1 (CGI export format reports mapping positions 0-based). 0 is reported if there are no mappings found for this HalfDNB
MAPQ	<CG_Mapping weight>	MAPping Quality (phred-scaled probability that the mapping position of this read is incorrect)	The probabilities are reported in different ranges for consistent pair reads (Flag 0x0002, case 1) and for non-paired mappings. It's not recommended to directly compare values of consistent and inconsistent mappings.
CIGAR	<SigarString> and GS/GQ/GC flags	extended CIGAR string	Currently, CGI initial mappings files do not allow insertions or deletions. Therefore, only M and N operations are used in the CIGAR field. The negative gaps are represented using GS/GQ/GC flags. The CIGAR sequence will represent the positive gaps using N and ignore the negative gaps.
MRNM	"=" or <ChromosomeId> or "*"	Mate Reference sequence NaMe; "=" if the same as <RNAME>	Reports "*" if there is no consistent mate found.
MPOS	<MatePosition>	1-based leftmost Mate	Reports 0 if there is no consistent mate

	or 0	POSition of the clipped sequence	found.
ISIZE	<DistanceToMate > or 0	inferred Insert SIZE	The distance between the consistent mate start position and the start position of the current HalfDNB mapping. The value is 0 if the mates are mapped to different chromosomes.
SEQ	<Sequence>	query SEQuence; "=" for a match to the reference; n/N/. for ambiguity	The regions of overlapping bases in the negative gaps contain the bases with higher scores.
QUAL	<QualityScores>	query QUALity; ASCII-33 gives the Phred base quality	The values are copied from the corresponding record of the CGI Reads file
TAG	GS/GQ/GC R2/Q2	Tags	GS/GQ/GC flags are used to represent CGI-specific negative wobble gaps in HalfDNBs. See SAM Format Specification (Format of optional fields, Note 6) for the description of the flags. The other tags that are optionally provided are R2,Q2.

Rules to set the "not primary" flag (0x0100):

The flag "not primary" is set for a HalfDNB mapping in the following cases:

- There is another mapping of the same HalfDNB having a higher MAPQ value.
- The mapping of a HalfDNB doesn't have a consistent mate pair mapping, and there are mappings found for the mate HalfDNB.
- The mapping's best mate has a best mate which is not the current mapping.

Combining mapping records in SAM:

1. The best mapping pair (the best score) of a DNB is reported with the "non-primary" flag set to 0. Both mappings should refer to each other as the best mates.
2. All the other mappings of that DNB are reported in non defined order and have the "non-primary" flag set to 1.
3. If both HalfDNBs are mapped uniquely but not consistently, they are not reported as primary ("non-primary" flag is set to 0) even though they are not consistent mates.

4. If only one HalfDNB is mapped, the best mapping of that HalfDNB is reported followed by a "non-mapped" mapping record of the other HalfDNB. The alignment position of the other HalfDNB is set to the same values as the mapped HalfDNB and the "non-primary" flag of the records is set to 0. The not mapped record will be marked as "not mapped" by appropriate flags.
5. All the other mappings of the mapped read from number 4 above are reported one record per mapping having "non-primary" flag set to 1.
6. All the not mapped reads are reported in pairs aligned to the 0 position. The alignment position is important to keep the records together while sorting and merging BAM files.

Evidence2sam (Beta version)

The evidence2sam tool converts Complete Genomics evidence mappings to the SAM format. The current implementation is in beta form. For pipelines that require eventually converting to the BAM format, the output of evidence2sam can be sent to standard output, which can be processed by SAM Tools. For example, this command pipeline creates an indexed, reference-sorted BAM file:

```
cgatools evidence2sam \
        --beta \
        --evidence-dnbs=/path/to/evidenceDnbs-chrN-XXX.tsv.bz2 \
        --reference=/path/to/build36.crr | \
    samtools view -uS - | \
    samtools sort - result && samtools index result.bam
```

Complete Genomics evidence mappings are the mappings that were used to call variations found by the CGI genome assembly process. The assembly process uses a local *de novo* method to find likely alleles for a variation interval (small region of the genome, typically less than 200 bases), then an optimization process to refine the allele choices. The evidence mappings are DNB alignments that indicate support for the best hypothesis found during the assembly process. The evidence2sam tool can be used to convert these mappings to SAM for visualization in a genome browser like IGV. The details of the Complete Genomics data representation in the SAM output are covered in the map2sam tool description in this document.

In two situations, a DNB may have multiple mapping records present in the evidence DNB mappings provided by Complete Genomics. First, if the best hypothesis is heterozygous and contains two non-reference alleles, support is also given for the reference allele. In this case, if a DNB supports two of the three alleles equally well (or similarly well) and not the third allele, then the evidence DNB mappings contain a record showing alignment of the DNB to each of the two alleles it supports. Second, if there are two regions of the genome with similar sequence such that DNBs align well to either sequence, those DNBs may be used as evidence for alleles in both regions. A post-processing step of the CGI assembly process finds such regions and no-calls them.

Because most tools that visualize SAM do not have rich features to specify an allele a DNB maps against, visualization of the duplicate mapping records present in the evidence can be confusing. For this reason, the evidence2sam tool has an option to de-duplicate the mappings present in the evidence, for both forms of duplication described above, for duplicate DNB mappings that are nearby on the reference. Specifically, the evidence2sam tool de-duplicates using the following algorithm, for each variation interval:

1. Update the read-ahead buffer to ensure it contains all evidence mapping records up to 1 Kb to the right of the position of the rightmost evidence mapping record for this interval.
2. Moving from position 0 to the end of the chromosome, processing each mapping record of the current variation interval as follows:
    a. Collect all the mappings of the same DNB that belong to the current interval or mappings from different intervals that overlap the corresponding arm/both arms of the selected DNB.
    b. Run one-DNB de-duplication. This operation deletes all the collected DNB mappings from the buffer except the “best” one.
    c. Write the “best” mapping into the SAM output stream.
    d. Remove the “best” mapping from the buffer and proceed to the next mapping in the current interval.
    e. If the last mapping in the current interval has been processed, remove the mappings processed for the current interval from the read-ahead buffer.

During de-duplication, the following rules are used to determine the best mapping for a DNB:

1. If several mapping records belong to the same variation interval, leave only the record that has maximum mapping quality.
2. If several mapping records belong to adjacent variation intervals (same side and strand), leave only the record that has maximum mapping quality.
3. If there are only two mapping records in the set and their different arms support different intervals, construct a composite mapping inheriting MAPQ, position in the reference, and reference alignment from a corresponding mapping record.
4. If there is still more then one mapping record in the input set, select the mapping with highest MAPQ and remove the other mappings.

The following additional limitations apply to evidence2sam output:

- The converted evidence support mappings are the mappings that belong only to the regions where variations were called.
- SAM does not have strong support for overlapping sub-reads (e.g., the negatively sized intra-read gaps), which are present in Complete Genomics data. To represent overlapping reads, the

strongest base call is put in the SAM mapping record, and the alternative base calls are represented using the GS/GQ/GC tags of the mapping record.

- When the option to de-duplicate mapping records is not used, evidence2sam reports all mappings as non-primary mappings.
- The NM tag (edit distance to reference sequence) is not currently produced by map2sam.
- The R2 and Q2 tags (mate sequence and quality scores) can be generated optionally.