Sentieon Documentation

Release 202503.02

Sentieon, Inc

Jan 27, 2026

Contents

1 Sentieon Quick Start

1.1 Introduction

This document serves as a guide to get first-time users introduced to the Sentieon® software. If you have any additional questions, please contact the technical support at Sentieon® Inc. at support@sentieon.com.

1.2 What do you need to get started?

To get started using Sentieon® software, you will need the following:

1. Hardware requirements: A Linux server with the following configuration:

- Linux running one of the following distributions or higher: RedHat/CentOS 6.5, Debian 7.7, OpenSUSE-13.2, or Ubuntu-14.04.
- 16GB of memory for small panel or whole exome or 64GB for whole genome.
- (Recommended) High-speed SSD drives are preferred for ideal I/O performance to get maximum CPU utilization.

2. Software requirements:

    - Python 2.6.x, Python 2.7.x, or python3.x is required. You can check Python version by typing the following:

```
python --version
```

3. Sentieon® software release package:

- Download the package from the link provided by the technical support at Sentieon.
- Decompress the package by running the following command, where VERSION is the version you are using, for example 202503.02:

```
tar xvzf sentieon-genomics-VERSION.tar.gz
```

4. License requirements: Please see the Appendix for more details on how to set up your license. IT support may be needed.
5. Environment requirements:

- If Python 2.6.x, Python 2.7.x, or python3.x is not the default Python version, you can set the following environment variable.

```
export SENTIEON_PYTHON=Python_location
```

- If you are using a localhost license file, set the following environment variable, where LICENSE_DIR is where the license file is located, and LICENSE_FILE.lic is the license file name.

```
export SENTIEON_LICENSE=LICENSE_DIR/LICENSE_FILE.lic
```

- If user is using a license server, set the following environmental variable, where LICSRVR_HOST and LICSRVR_PORT are the hostname and port of the license server. Please see the next section for more details.

```
export SENTIEON_LICENSE=LICSRVR_HOST:LICSRVR_PORT
```

- For convenience, set the binary path as shown below, where PATH_TO_ SENTIEON_BINARY_DIRECTORY is where Sentieon® binary is installed.

```
export SENTIEON_INSTALL_DIR=PATH_TO_SENTIEON_BINARY_DIRECTORY
```

- For improved performance when using NFS storage, set the SENTIEON_TMPDIR environmental variable to point to local scratch fast storage.

```
export SENTIEON_TMPDIR=/tmp
```

1.3 Start your first Sentieon® DNAscope job

Sentieon® Inc. provides a quick start package that includes a sample script and data to help you quickly test the installation and to diagnose potential problems.

The quick start package includes data for a single chromosome, both sequence data of a sample and reference materials. The job script uses the Sentieon DNAscope pipeline for a set of pair-ended Illumina fastq files:

- BWA: Map reads to the reference.
- Metrics and LocusCollector: Collect reads' statistics.
- Dedup: Remove duplicate reads.
- Variant calling: DNAscope variant calling.

> **Note**
>
> DNAscope is only recommended for use with samples from diploid organisms. For other samples, please use DNAseq.

1.3.1 Run the quick start package

To get started, copy the downloaded quick start package to a new directory, and unpack it by running the following:

```
tar xzvf sentieon_quickstart.tar.gz
```

Here is what is included in the package:

- `sentieon_quickstart.sh`: the sample shell script that drives the entire pipeline.
- `reference`: a directory that contains human genome reference files and database files of known SNP sites.
- `models`: a directory that contains DNAscope model files.
- FASTQ files: sample sequence files.

Before running the script, you need to make sure that the environment variables are properly set as described above, including the license and path to the directory.

Then open your favorite editor to edit the user settings in `sentieon_quickstart.sh`.

```
# Update with the location of the Sentieon software package
SENTIEON_INSTALL_DIR=/home/release/sentieon-genomics-202503.02

# Update with the location of temporary fast storage and uncomment
#SENTIEON_TMPDIR=/tmp

# It is important to assign meaningful names in actual cases.
# It is particularly important to assign different read group names.
sample="sample_name"
group="read_group_name"
platform="ILLUMINA"

# Other settings
nt=16 #number of threads to use in computation

# Is the data prepared with a PCR free library prep
PCRFREE=true
```

Note

In the user setting shell script `sentieon_quickstart.sh`:

- It is important to assign meaningful names in actual cases.
- It is particularly important to assign different read group names.

To get the number of the CPU cores, user can run nproc as shown below.

```
nproc
```

To better understand the rest of the `sentieon_quickstart.sh` script, please read the comment in each section, and the corresponding chapters in the manual.

Now, launch the script by simply running `sentieon_quickstart.sh`, and watch the result unfold. The entire run takes about 3 - 5 minutes on a typical Linux server. Actual time varies depending on the computation environment.

```
sh sentieon_quickstart.sh &
```

1.3.2 Understanding the results

Below is a list of the files, their meaning and references. For more details, please refer to documentation.

1. Quick start test output files

File name	Description
sorted.bam	Coordinate-sorted BAM file after alignment with Sentieon® BWA mem.
score.txt	Duplicate read data file.
aln_metrics.txt	Alignment and general statistics of the two pair sequence reads.
gc_summary.txt	GC bias statistics summary.
gc_metrics.txt, qc-report.pdf	GC bias statistics data file and report PDF.
qd_metrics.txt, qd-report.pdf	Base quality score distribution data file and report PDF.
mq_metrics.txt, mq-report.pdf	Cycle-dependence of the mean quality score data file and report PDF.
is_metrics.txt, is-report.pdf	Insert size distribution data file and report PDF.
deduped.bam	Output BAM file of Dedup stage, with duplicated reads removed.
dnascope.vcf.gz	Output VCF file of DNAscope variant calling.

1.4 Description of the Sentieon® tools

The table below shows the different Sentieon® products and tools and their purpose. It is also noted if a tool implements functionality equivalent to an existing GATK pipeline tool.

Table 1.1: Sentieon tools

Sentieon® product	Sentieon® tool	Typical use	Equivalent GATK pipeline tool
Sentieon® BWA	Sentieon® BWA	Read alignment and mapping	BWA
DNAscope	DNAscope	Improved germline SNV/Indel/SV calling	
DNAseq®	Genotyper	Germline SNV/Indel calling, non haplotype based	UnifiedGenotyper
DNAseq®	Haplotyper	Germline SNV/Indel calling	HaplotypeCaller
DNAseq®	GVCFtyper	Joint calling of cohorts, demonstrated up to 200,000 samples	GenotypeGVCFs
DNAseq®	VarCal	Calculate Variant Quality Score Recalibration	VariantRecalibrator
DNAseq®	ApplyVarCal	Apply Variant Quality Score Recalibration	ApplyRecalibration
RNAseq	RNASplitReadsAtJunction	RNA SNV/Indel calling	SplitNCigarReads
RNAseq	Haplotyper	RNA SNV/Indel calling	HaplotypeCaller
TNseq®	TNsnv	Somatic SNV calling, non haplotype based	MuTect
TNseq®	TNhaplotyper	Somatic SNV/Indel calling	MuTect2
TNseq®	TNhaplotyper2 + TNfilter	Somatic SNV/Indel calling	Mutect2 and FilterMutectCalls from GATK4

continues on next page

Table 1.1 – continued from previous page

Sentieon® product	Sentieon® tool	Typical use	Equivalent GATK pipeline tool
TNscope®	TNscope®	Improved somatic SNV/Indel/SV calling	
General tools	Dedup and LocusCollector	Perform deduplication	Picard MarkDuplicates
General tools	Realigner	Perform Indel relaignment for non-haplotype based callers	RealignerTargetCreator and IndelRealigner
General tools	QualCal	Perform Base Quality Score Recalibration	BaseRecalibrator, AnalyzeCovariates
General tools	ReadWriter	Create BAM files	PrintReads
General tools	AlignmentStat	QC metrics	Picard CollectAlignmentSummaryMetrics
General tools	BaseDistributionByCycle	QC metrics	Picard CollectBaseDistributionByCycle
General tools	CollectVCMetrics	QC metrics	Picard CollectVariantCallingMetrics
General tools	ContaminationAssessment	QC metrics	ContEst
General tools	CoverageMetrics	QC metrics	DepthOfCoverage
General tools	GCBias	QC metrics	Picard CollectGcBiasMetrics
General tools	HsMetricAlgo	QC metrics	Picard CollectHsMetrics
General tools	InsertSizeMetricAlgo	QC metrics	Picard CollectInsertSizeMetrics
General tools	MeanQualityByCycle	QC metrics	Picard MeanQualityByCycle
General tools	QualDistribution	QC metrics	Picard QualityScoreDistribution
General tools	QualityYield	QC metrics	Picard CollectQualityYieldMetrics
General tools	SequenceArtifactMetricsAlgo	QC metrics	Picard CollectSequencingArtifactMetrics, ConvertSequencingArtifactToOxoG
General tools	WgsMetricsAlgo	QC metrics	Picard CollectWgsMetrics

1.5 Appendix - Set up license

Sentieon® software is a license-controlled software. The user is required to properly set up the license in order to run the software.

We provide two types of the licenses:

- Single machine evaluation license: this license is used for evaluating the Sentieon® software in a single machine. It allows new users to get quickly started on using the software without requiring help from the IT department. In order to use this license, the computer where you plan on running the Sentieon® software requires external Internet access.
- Cluster license: this license is used in a cluster environment. With this license, a floating license server lightweight process is running on one node in the cluster, serving licenses though TCP to all other nodes that have network connection to the license server. This license server is running in a special

non-computing node on the cluster periphery that has unrestricted access to the outside world through HTTPS, and serves the licenses to the rest of the nodes in the cluster by listening to a specific TCP port that needs to be open within the cluster.

1.5.1 Setting up a single machine evaluation license

To use the single machine evaluation license, the computing node needs have access to the Internet. This allows Sentieon® software to validate the license.

To use a single machine evaluation license, follow the steps below:

1. Copy the license file to the computing node. For example, the license file **LICENSE_FILE.lic** is now located at **LICENSE_DIR** .
2. Set up environment variable as below:

```
export SENTIEON_LICENSE=LICENSE_DIR/LICENSE_FILE.lic
```

1.5.2 Setting up license server

As shown in Fig. 1.1, license server requires the following:

1. The license server should have access to the Internet to perform license validation.
2. The computing nodes should have access to the license server via a host name **LICSRVR_HOST**
3. The machine the license server is running has an open port for the license services to listen on, and the computing nodes have access to that port. Here we assume the available port is **LICSRVR_PORT**

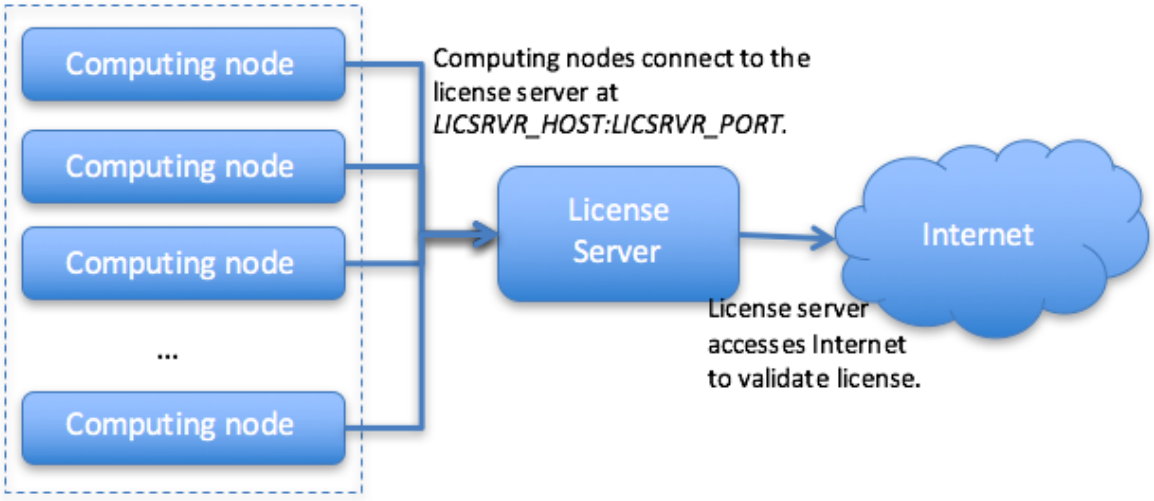


Fig. 1.1: Topology of the computing nodes and license server

You may need IT support to get **LICSRVR_HOST:LICSRVR_PORT**, and confirm that the above requirements are met.

Note

If the license server is behind a firewall, separated from the computing nodes through a NAT, the license server's hostname/IP visible to the nodes may be different from its actual hostname/IP. If this is the case, you will need to bind the license server on the actual IP address, while the compute node requests license from the IP address after NAT. Please contact Sentieon support for more details.

Follow these steps to obtain license file, set up and test the license server:

1. Send the following information to Sentieon® to receive the license file:

- FQDN (fully qualified domain name) **LICSRVR_HOST** of the designated machine to run license service.
- The designated port **LICSRVR_PORT** to Sentieon® to receive the license file.

2. Copy the received license file to the license server LICSRVR_HOST. We assume the license file is located in **LICENSE_PATH/LICENSE_FILE**. Run the following command *on the license server* to start the license server process:

```
<SENTIEON_INSTALL_DIR>/bin/sentieon licsrvr --start --log LOG_FILE LICENSE_PATH/LICENSE_
↪FILE
```

3. Alternatively, you can follow the instructions in section *8.8 - Running the license server (LICSRVR) as a system service* in the Sentieon® Genomics Manual, to configure and start the license server as a system daemon.

4. Go to the Sentieon® installation directory. Run the following commands *on the license server* to confirm the license server is up and running.

```
<SENTIEON_INSTALL_DIR>/bin/sentieon licclnt ping -s LICSRVR_HOST:LICSRVR_PORT
```

   If the command returns without an error message, the license server is up and running.

5. Login to one of the computing node, go to the Sentieon® installation directory, and run the above command again:

```
<SENTIEON_INSTALL_DIR>/bin/sentieon licclnt ping -s LICSRVR_HOST:LICSRVR_PORT
```

   If the command returns without an error message, the computing node now can access the license server, too.

6. Set up the following environment variable and you are good to go.

```
export SENTIEON_LICENSE=LICSRVR_HOST:LICSRVR_PORT
```

2 Typical usage for Pangenome

The Sentieon® Genomics software includes a pipeline for pangenome alignment and germline variant calling. Compared to the linear alignment *Typical usage for DNAseq®* or *Typical usage for DNAscope* pipelines, the Sentieon pangenome pipeline utilizes pangenome graph data structures to improve short-read alignment and variant calling accuracy. The Pangenome pipeline is recommended for datasets sequenced from human samples.

2.1 General

2.1.1 Pipeline Overview

The Sentieon® Pangenome pipeline will process the input reads through the following steps:

1. Extract the k-mer spectra: This step generates a k-mer spectrum from the input reads as a KFF file.
2. Generate a personalized pangenome: This step generates a personalized pangenome using the sample's k-mer spectrum. The personalized pangenome is then converted into a personalized reference genome.
3. Map reads to reference: This step aligns the reads contained in the FASTQ files to a reference genome in FASTA format. This step is skipped with aligned input data in BAM or CRAM format.
4. Map reads to the personalized reference: This step aligns a subset of reads to the sample's personalized reference genome and lifts these alignments back to the standard reference genome.
5. Duplicate marking and metrics collection: This step identifies DNA molecules that were sequenced multiple times and marks these reads as duplicate to exclude them from downstream analysis. This step also generates a statistical summary of the data quality from the aligned read data. If `multiqc` is available, a report will be generated from the collected metrics. The duplicate marked reads are output in BAM or CRAM format.
6. Small variant calling using DNAscope: This step identifies small variants (SNVs and indels) present in the sample relative to the reference genome and calculates sample genotypes.

These steps can be run as a single command using the `sentieon-cli`.

2.1.2 Input files

The Sentieon® Pangenome pipeline is implemented in the `sentieon-cli`. Instructions for installing the `sentieon-cli` can be found on GitHub at, https://github.com/Sentieon/sentieon-cli. A detailed description of the Sentieon® Pangenome pipeline can be found at sentieon_pangenome_cli.

In this bioinformatics pipeline you will need the following inputs:

- The FASTA file containing the nucleotide sequence of the GRCh38 reference genome corresponding to the sample you will analyze. Samtools and bwa index files for the reference genome are also required.
- The pangenome file in GBZ format with a corresponding haplotype (`.hapl`) file. The pipeline requires that the pangenome incorporates GRCh38 contigs as a starting graph.
- One or multiple GZIP-compressed FASTQ files containing the nucleotide sequence of the sample to be analyzed. These files contain the raw reads from the DNA sequencing. The software only supports files containing quality scores in Sanger format (Phred+33).
- A population VCF file containing allele frequency information from population databases.
- A pipeline and sequencing platform-specific model bundle file. Model bundle files can be accessed from https://github.com/Sentieon/sentieon-models.
- (Optional) A BED file containing variant calling intervals. Recommended for whole-genome sequencing data to restrict variant calling to the canonical contigs. Recommended for whole-exome sequencing data to restrict variant calling to target regions.
- (Optional) The Single Nucleotide Polymorphism database (dbSNP) data that you want to include in the pipeline. The data is used in the form of a VCF file; you can use a VCF file compressed with bgzip and indexed.

2.1.3 Third-party tools

In addition to the Sentieon® software, the bioinformatics pipeline requires the following third-party tools:

- samtools[1] version 1.16 or higher. `samtools` is used to manipulate the aligned read data. `samtools` installation instructions can be found at, https://github.com/samtools/samtools?tab=readme-ov-file#building-samtools.
- bcftools[2] version 1.22 or higher. `bcftools` is used to manipulate called variants. `bcftools` installation instructions can be found at, https://samtools.github.io/bcftools/howtos/install.html.
- vg[3]. `vg` is used to generate a personalized pangenome. x86 executables for `vg` can be downloaded from GitHub, https://github.com/vgteam/vg/releases/download/v1.68.0/vg.
- KMC[4] version 3 or higher. `kmc` is used for k-mer counting of the input sample. x86 executables for `kmc` can be downloaded from GitHub, https://github.com/Sentieon/KMC/releases/download/v3.2.4-pipe2/KMC3.2.4.linux.x64.tar.gz.

The following third-party tools are optional and will add additional functionality to the pipeline:

- MultiQC[5]. `multiqc` is used to generate a report from the pipeline metrics. `multiqc` installation instructions can be found at, https://github.com/MultiQC/MultiQC?tab=readme-ov-file#installation.

Tools will be called from the user's `PATH`.

2.2 Example usage

The following example shows step-by-step usage of the Sentieon® pangenome pipeline using recommended input files to process a test sample.

[1] https://www.htslib.org/
[2] http://samtools.github.io/bcftools/bcftools.html
[3] https://github.com/vgteam/vg
[4] https://github.com/refresh-bio/KMC
[5] https://github.com/MultiQC/MultiQC

2.2.1 Download input files

The following shell command is run to download the GRCh38, pangenome and population VCF input files required by the pipeline:

```
curl -L \
  -O 'https://ftp.sentieon.com/public/GRCh38/hg38_canonical.bed' \
  -O 'https://human-pangenomics.s3.amazonaws.com/pangenomes/freeze/release2/minigraph-cactus/
↪hprc-v2.0-mc-grch38.gbz' \
  -O 'https://human-pangenomics.s3.amazonaws.com/pangenomes/freeze/release2/minigraph-cactus/
↪hprc-v2.0-mc-grch38.hapl' \
  -O 'https://ftp.sentieon.com/public/GRCh38/population/pop-v20g41-20251216.vcf.gz' \
  -O 'https://ftp.sentieon.com/public/GRCh38/population/pop-v20g41-20251216.vcf.gz.tbi' \
  -o 'hg38_ucsc.fa' 'https://ngi-igenomes.s3.amazonaws.com/igenomes/Homo_sapiens/UCSC/hg38/
↪Sequence/WholeGenomeFasta/genome.fa' \
  -o 'hg38_ucsc.fa.fai' 'https://ngi-igenomes.s3.amazonaws.com/igenomes/Homo_sapiens/UCSC/
↪hg38/Sequence/WholeGenomeFasta/genome.fa.fai' \
  -o 'hg38_ucsc.fa.amb' 'https://ngi-igenomes.s3.amazonaws.com/igenomes/Homo_sapiens/UCSC/
↪hg38/Sequence/BWAIndex/genome.fa.amb' \
  -o 'hg38_ucsc.fa.ann' 'https://ngi-igenomes.s3.amazonaws.com/igenomes/Homo_sapiens/UCSC/
↪hg38/Sequence/BWAIndex/genome.fa.ann' \
  -o 'hg38_ucsc.fa.bwt' 'https://ngi-igenomes.s3.amazonaws.com/igenomes/Homo_sapiens/UCSC/
↪hg38/Sequence/BWAIndex/genome.fa.bwt' \
  -o 'hg38_ucsc.fa.pac' 'https://ngi-igenomes.s3.amazonaws.com/igenomes/Homo_sapiens/UCSC/
↪hg38/Sequence/BWAIndex/genome.fa.pac' \
  -o 'hg38_ucsc.fa.sa' 'https://ngi-igenomes.s3.amazonaws.com/igenomes/Homo_sapiens/UCSC/
↪hg38/Sequence/BWAIndex/genome.fa.sa' \
  -O 'https://storage.googleapis.com/gcp-public-data--broad-references/hg38/v0/Homo_sapiens_
↪assembly38.dbsnp138.vcf.gz' \
  -O 'https://storage.googleapis.com/gcp-public-data--broad-references/hg38/v0/Homo_sapiens_
↪assembly38.dbsnp138.vcf.gz.tbi'
```

A model bundle file for the pipeline can be downloaded from the sentieon models[6] page.

2.2.2 Run the pangenome pipeline with fastq input

A single command is run to execute the pangenome pipeline from the input files.

```
sentieon-cli sentieon-pangenome \
  -r hg38_ucsc.fa \
  --hapl hprc-v2.0-mc-grch38.hapl \
  --gbz hprc-v2.0-mc-grch38.gbz \
  -m SentieonIlluminaPangenomeRealignWGS1.0.bundle \
  --pop_vcf pop-v20g41-20251216.vcf.gz \
  --r1_fastq HG002.novaseq.pcr-free.30x.R1.fastq.gz \
  --r2_fastq HG002.novaseq.pcr-free.30x.R2.fastq.gz \
  --readgroup "@RG\tID:HG002-1\tSM:HG002\tLB:HG002-LB-1\tPL:ILLUMINA" \
  -b hg38_canonical.bed \
  --dbsnp Homo_sapiens_assembly38.dbsnp138.vcf.gz \
  --pcr_free \
  HG002_pangenome.vcf.gz
```

Depending on whether PCR is involved, DNAscope uses different priors for finding significant INDEL variants.

[6] https://github.com/Sentieon/sentieon-models

The default setting is appropriate for samples sequenced with standard library preps. Setting `--pcr_free` uses a prior appropriate for samples sequenced using a PCR-free library prep.

2.2.3 Output files

The following files are output by the pipeline with all optional features:

- `HG002_pangenome.vcf.gz`: SNV and indel calls in VCF format.
- `HG002_pangenome_bwa_deduped.cram`: bwa aligned, coordinate-sorted, and duplicate-marked read data from the input FASTQ file.
- `HG002_pangenome_mm2_deduped.cram`: pangenome aligned, coordinate-sorted, and duplicate-marked read data from the input FASTQ file. The reads in this file are aligned to the pangenome and lifted back to GRCh38.
- `HG002_pangenome_metrics/`: A directory containing QC metrics for the analyzed sample.

2.3 Limitations of the Sentieon® pangenome pipeline

The Sentieon® pangenome pipeline currently only supports Minigraph-Cactus pangenomes with a GRCh38 reference sequence, such as those generated by the Human Pangenome Reference Consortium (HPRC). Please reach out to Sentieon® support for information on using the pipeline with other pangenomes.

3 Typical usage for DNAseq®

One of the typical uses of Sentieon® Genomics software is to perform the bioinformatics pipeline for DNA analysis recommended in the Broad institute best practices described in https://www.broadinstitute.org/gatk/guide/best-practices. *Recommended bioinformatics pipeline for DNA variant calling analysis* illustrates such a typical bioinformatics pipeline.

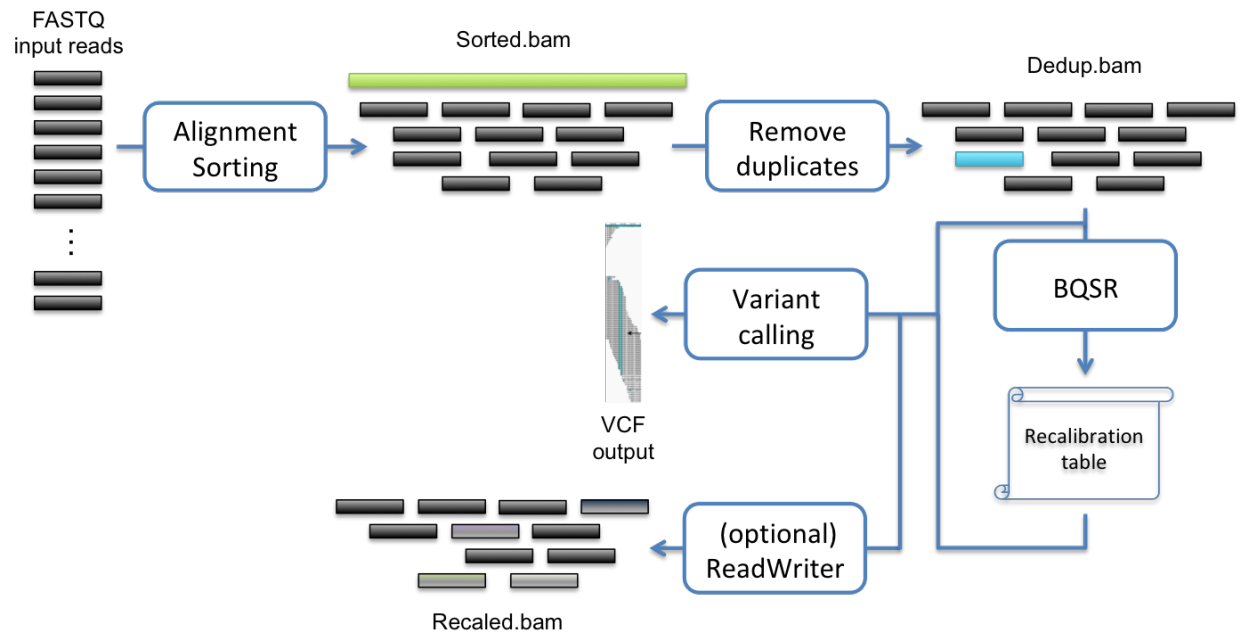


Fig. 3.1: Recommended bioinformatics pipeline for DNA variant calling analysis

3.1 General

In this bioinformatics pipeline you will need the following inputs:

- The FASTA file containing the nucleotide sequence of the reference genome corresponding to the sample you will analyze. The reference data needs to be pre-processed such that the data specified in *Data requirements for the reference nucleotide sequence* is available to the software. You can refer to *Preparing reference file for use* for instructions on how to generate the required files.

Table 3.1: Data requirements for the reference nucleotide sequence

Data	Description
.dict	Dictionary file
.fasta	Reference sequence file
.fasta.amb	BWA index file
.fasta.ann	BWA index file
.fasta.bwt	BWA index file
.fasta.fai	Index file
.fasta.pac	BWA index file
.fasta.sa	BWA index file

- One or multiple FASTQ files containing the nucleotide sequence of the sample to be analyzed. These files contain the raw reads from the DNA sequencing. The software supports inputting FASTQ files compressed using GZIP. The software only supports files containing quality scores in Sanger format (Phred+33).
- (Optional) The Single Nucleotide Polymorphism database (dbSNP) data that you want to include in the pipeline. The data is used in the form of a VCF file; you can use a VCF file compressed with bgzip and indexed.
- (Optional) Multiple collections of known sites that you want to include in the pipeline. The data is used in the form of a VCF file; you can use a VCF file compressed with bgzip and indexed.

The following steps compose the typical bioinformatics pipeline:

1. Map reads to reference: This step aligns the reads contained in the FASTQ files to map to a reference genome contained in the FASTA file. This step ensures that the data can be placed in context.
2. Calculate data metrics: This step produces a statistical summary of the data quality and the pipeline data analysis quality.
3. Remove or mark duplicates: This step detects reads indicative that the same DNA molecules were sequenced several times. These duplicates are not informative and should not be counted as additional evidence.
4. (Optional) Base quality score recalibration (BQSR): This step modifies the quality scores assigned to individual read bases of the sequence read data. This action removes experimental biases caused by the sequencing methodology.
5. Variant calling: This step identifies the sites where your data displays variation relative to the reference genome, and calculates genotypes for each sample at that site.

3.2 Step by step usage for DNAseq®

3.2.1 Map reads to reference

A single command is run to efficiently perform the alignment using BWA, and the creation of the BAM file and the sorting using Sentieon® software:

```
(sentieon bwa mem -R '@RG\tID:GROUP_NAME\tSM:SAMPLE_NAME\tPL:PLATFORM' \
  -t NUMBER_THREADS REFERENCE SAMPLE [SAMPLE2] || echo -n 'error' ) \
  | sentieon util sort -r REFERENCE -o SORTED_BAM -t NUMBER_THREADS --sam2bam -i -
```

Inputs and options for BWA are described in its manual[7].

[7] http://bio-bwa.sourceforge.net/bwa.shtml

Alternatively, you can use other aligners that produce a file following the SAM format in stdout and replace the BWA portion of the command.

The following inputs are required for the command:

- **GROUP_NAME**: Readgroup identifier that will be added to the readgroup header line. The `RG:ID` needs to be unique among all the datasets that you plan on using, which is important when working with multiple input files as described in *Working with multiple input files*, or when performing a Tumor-Normal analysis as described in *Typical usage for TNscope®*.
- **SAMPLE_NAME**: name of the sample that will be added to the readgroup header line.
- **PLATFORM**: name of the sequencing platform used to sequence the DNA. Possible options are ILLUMINA when the fastq files have been produced in an Illumina™ machine; IONTORRENT when the fastq files have been produced in a Life Technologies™ Ion-Torrent™ machine; ELEMENT when the fastq files have been produced in an Element Biosciences™ machine; DNBSEQ when the fastq files have been produced in a MGI™ machine; ULTIMA when the fastq files have been produced in an Ultima Genomics™ machine.
- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system. We recommend that you use the same number of threads for both BWA and for the util binary.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that all additional reference data specified in *Data requirements for the reference nucleotide sequence* is available in the same location with consistent naming.
- **SAMPLE**: the location of the sample FASTQ file. If the data comes from pair ended sequencing technology, you will also need to input **SAMPLE2** as the corresponding pair sample FASTQ file.
- **SORTED_BAM**: the location and filename of the sorted mapped BAM output file. A corresponding index file (.bai) will be created.

BWA will produce slightly different results depending on the number of threads used in the command. This is due to the fact that BWA computes the insert size distribution on a chunk, whose size is dependent on the number of threads. To guarantee that the results are independent of the number of threads used, you should fix the chunk size in bases using option `-K 10000000`.

3.2.2 Calculate data metrics

A single command is run to generate 5 statistical summaries of the data quality and the pipeline data analysis quality results:

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i SORTED_BAM \
  --algo GCBias --summary GC_SUMMARY_TXT GC_METRIC_TXT \
  --algo MeanQualityByCycle MQ_METRIC_TXT \
  --algo QualDistribution QD_METRIC_TXT \
  --algo InsertSizeMetricAlgo IS_METRIC_TXT \
  --algo AlignmentStat ALN_METRIC_TXT
```

Four commands are run to generate the plots from the statistical summaries:

```
sentieon plot GCBias -o GC_METRIC_PDF GC_METRIC_TXT
sentieon plot MeanQualityByCycle -o MQ_METRIC_PDF MQ_METRIC_TXT
sentieon plot QualDistribution -o QD_METRIC_PDF QD_METRIC_TXT
sentieon plot InsertSizeMetricAlgo -o IS_METRIC_PDF IS_METRIC_TXT
```

The following inputs are required for the commands:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **SORTED_BAM**: the location where the previous mapping stage stored the result.
- **GC_SUMMARY_TXT**: the location and filename of the GC bias metrics summary results output file.
- **GC_METRIC_TXT**: the location and filename of the GC bias metrics results output file.
- **MQ_METRIC_TXT**: the location and filename of the mapping quality metrics results output file.
- **QD_METRIC_TXT**: the location and filename of the quality/depth metrics results output file.
- **IS_METRIC_TXT**: the location and filename of the insertion size metrics results output file.
- **ALN_METRIC_TXT**: the location and filename of the alignment metrics results output file.
- **GC_METRIC_PDF**: the location and filename of the GC bias metrics report output file.
- **MQ_METRIC_PDF**: the location and filename of the mapping quality metrics report output file.
- **QD_METRIC_PDF**: the location and filename of the quality/depth metrics report output file.
- **IS_METRIC_PDF**: the location and filename of the insertion size metrics report output file.

3.2.3 Remove or mark duplicates

Two individual commands are run to remove or mark duplicates on the BAM file after alignment and sorting. The first command collects read information, and the second command performs the deduplication; the option `--rmdup` controls whether the duplicated reads are removed, if the option is present, or only marked as duplicated.

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --fun score_info SCORE.gz
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo Dedup [--rmdup] --score_info SCORE.gz \
  --metrics DEDUP_METRIC_TXT DEDUPED_BAM
```

The following inputs are required for the commands:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **SORTED_BAM**: the location where the previous mapping stage stored the result.
- **SCORE.gz**: the location and filename of the temporary score output file. Make sure that the same file is used for both commands.
- **DEDUP_METRICS_TXT**: the location and filename of the dedup metrics results output file.
- **DEDUPED_BAM**: the location and filename of the deduped BAM output file. A corresponding index file (.bai) will be created.

3.2.4 Base quality score recalibration (BQSR; optional)

A single command is run to calculate the required modification of the quality scores assigned to individual read bases of the sequence read data; the actual recalibration is applied during the variant calling stage.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE \
  -i DEDUPED_BAM --algo QualCal [-k KNOWN_SITES] RECAL_DATA.TABLE
```

Three commands are run to apply the recalibration and create a report on the base quality sore recalibration. The first command applies the recalibration to calculate the post calibration data table and additionally apply the recalibration on the BAM file, the second one creates the data for plotting; the third command plots the calibration data tables, both pre and post, into graphs in a pdf.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i DEDUPED_BAM \
  -q RECAL_DATA.TABLE --algo QualCal [-k KNOWN_SITES] \
  RECAL_DATA.TABLE.POST [--algo ReadWriter RECALIBRATED_BAM]
sentieon driver -t NUMBER_THREADS --algo QualCal --plot \
  --before RECAL_DATA.TABLE --after RECAL_DATA.TABLE.POST RECAL_RESULT.CSV
sentieon plot QualCal -o BQSR_PDF RECAL_RESULT.CSV
```

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **DEDUPED_BAM**: the location where the previous deduping stage stored the result.
- **RECAL_DATA.TABLE**: the location and filename of the recalibration table.
- **RECAL_DATA.TABLE.POST**: the location and filename of the temporary post recalibration table
- **RECAL_RESULT.CSV**: the location and filename of the temporary recalibration results output file used for plotting.
- **BQSR_PDF**: the location and filename of the BSQR results output file.

The following inputs are optional for the command:

- **KNOWN_SITES**: the location of the VCF file used as a set of known sites. You can include multiple collections of known sites by repeating the `-k KNOWN_SITES` option.
- **RECALIBRATED_BAM**: the location and filename of the recalibrated BAM output file. A corresponding index file (.bai) will be created. This output is optional as Sentieon® variant callers can perform the recalibration on the fly using the before recalibration BAM plus the recalibration table

3.2.5 Variant calling

A single command is run to call variants and additionally apply the BQSR calculated before.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i DEDUPED_BAM \
  -q RECAL_DATA.TABLE --algo Haplotyper [-d dbSNP] VARIANT_VCF
```

You may want to rerun only the variant calling, for example to use Unified Genotyper variant calling algorithm. In that case, you do not need to re-apply the BQSR, and can use the previously recalibrated BAM:

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i RECALIBRATED_BAM \
  --algo Genotyper [-d dbSNP] VARIANT_VCF
```

In both cases, using the recalibrated BAM or the BAM before recalibration plus the recalibration data table will give identical results; however, you should be careful not to use the recalibration data table together with the already recalibrated BAM, as that would apply the recalibration twice, leading to incorrect results.

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.

- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **DEDUPED_BAM**: the location where the previous deduping stage stored the result.
- **RECAL_DATA.TABLE**: the location where the previous BQSR stage stored the result.
- **RECALIBRATED_BAM**: the location of the recalibrated BAM file.
- **VARIANT_VCF**: the location and filename of the variant calling output file. A corresponding index file will be created. The tool will output a compressed file by using .gz extension.

The following inputs are optional for the command:

- **dbSNP**: the location of the Single Nucleotide Polymorphism database (dbSNP) that will be used to label known variants. You can only use one dbSNP file.

4 Typical usage for DNAscope

The Sentieon® Genomics software includes an improved algorithm to perform the variant calling step of germline DNA analysis. The pipeline used for DNAscope is similar to the DNAseq® one described in *Typical usage for DNAseq®*, but with differences in both alignment and variant calling. DNAscope accepts model files to improve processing speed and accuracy, and it can perform structural variant calling in addition to calling SNPs and small indels. DNAscope is recommended for datasets sequenced from human or other mammalian samples.

4.1 General

In this bioinformatics pipeline you will need the following inputs:

- The FASTA file containing the nucleotide sequence of the reference genome corresponding to the sample you will analyze.
- One or multiple FASTQ files containing the nucleotide sequence of the sample to be analyzed. These files contain the raw reads from the DNA sequencing. The software supports inputting FASTQ files compressed using GZIP. The software only supports files containing quality scores in Sanger format (Phred+33).
- A sequencing platform specific machine learning model file that can be accessed from https://github.com/Sentieon/sentieon-models.
- (Optional) A BED file containing variant calling intervals. Recommended for whole-exome or targeted sequencing data.
- (Optional) The Single Nucleotide Polymorphism database (dbSNP) data that you want to include in the pipeline. The data is used in the form of a VCF file; you can use a VCF file compressed with bgzip and indexed.

The following steps compose the typical bioinformatics pipeline for DNAscope:

1. Map reads to reference: This step aligns the reads contained in the FASTQ files to map to a reference genome contained in the FASTA file. This step ensures that the data can be placed in context.
2. Calculate data metrics: This step produces a statistical summary of the data quality and the pipeline data analysis quality.
3. Remove or mark duplicates: This step detects reads indicative that the same DNA molecules were sequenced several times. These duplicates are not informative and should not be counted as additional evidence.

4. Variant calling using DNAscope with a machine learning model: This step identifies the sites where your data displays variation relative to the reference genome, and calculates genotypes for each sample at that site.

4.2 Step by step usage

4.2.1 Map reads to reference

A single command is run to efficiently perform the alignment using BWA, and the creation of the BAM file and the sorting using Sentieon® software:

```
(sentieon bwa mem -R '@RG\tID:GROUP_NAME\tSM:SAMPLE_NAME\tPL:PLATFORM' \
  -t NUMBER_THREADS -x DNASCOPE_MODEL/bwa.model REFERENCE SAMPLE [SAMPLE2] \
  || echo -n 'error' ) \
  | sentieon util sort -r REFERENCE -o SORTED_BAM -t NUMBER_THREADS --sam2bam -i -
```

Inputs and options for BWA are described in its manual[8].

Compared to the BWA usage described in *Typical usage for DNAseq®*, the **DNASCOPE_MODEL** is added with the argument `-x DNASCOPE_MODEL/bwa.model`.

The following inputs are required for the command:

- **GROUP_NAME**: Readgroup identifier that will be added to the readgroup header line. The `RG:ID` needs to be unique among all the datasets that you plan on using, which is important when working with multiple input files as described in *Working with multiple input files*, or when performing a Tumor-Normal analysis as described in *Typical usage for TNscope®*.
- **SAMPLE_NAME**: name of the sample that will be added to the readgroup header line.
- **PLATFORM**: name of the sequencing platform used to sequence the DNA. Possible options are ILLUMINA when the fastq files have been produced in an Illumina™ machine; ELEMENT when the fastq files have been produced in an Element Biosciences™ machine; DNBSEQ when the fastq files have been produced in a MGI™ machine; ULTIMA when the fastq files have been produced in an Ultima Genomics™ machine.
- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system. We recommend that you use the same number of threads for both BWA and for the util binary.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that all additional reference data specified in *Data requirements for the reference nucleotide sequence* is available in the same location with consistent naming.
- **SAMPLE**: the location of the sample FASTQ file. If the data comes from pair ended sequencing technology, you will also need to input **SAMPLE2** as the corresponding pair sample FASTQ file.
- **SORTED_BAM**: the location and filename of the sorted mapped BAM output file. A corresponding index file (.bai) will be created.
- **DNASCOPE_MODEL**: the location of the DNAscope model bundle. The model will be used to determine the settings used in alignment and variant calling.

BWA will produce slightly different results depending on the number of threads used in the command. This is due to the fact that BWA computes the insert size distribution on a chunk, whose size is dependent on the number of threads. To guarantee that the results are independent of the number of threads used, you should fix the chunk size in bases using option `-K 10000000`.

[8] http://bio-bwa.sourceforge.net/bwa.shtml

For the metrics collection and duplicate removal stages, please refer to *Calculate data metrics* for detailed usage instructions.

4.2.2 Germline variant calling with a machine learning model

It is recommended to use DNAscope with a machine learning model to perform variant calling with higher accuracy by improving the candidate detection and filtering.

Sentieon® can provide you with a sequencing platform specific model trained using a subset of the data from the GiAB truth-set found in https://github.com/genome-in-a-bottle. The models were created by processing reference samples HG001-HG007 through a pipeline consisting of Sentieon® BWA-mem alignment and Sentieon® deduplication, and using the variant calling results to calibrate a model to fit the truth-set.

In addition, Sentieon® can assist you in the creation of models using your own data, which will calibrate the specifics of your sequencing and bio-informatics processing.

4.2.2.1 Using a machine learning model with DNAscope

Two individual commands are run to call variants and to apply the machine learning model. The input BAM file should come from a pipeline where only alignment and deduplication have been performed (no BQSR or indel realignment), to match the model creation methodology.

```
PCRFREE=true #PCRFREE=true means the sample is PCRFree, change it to false for PCR samples.
if [ "$PCRFREE" = true ] ; then
    sentieon driver -t NUMBER_THREADS -r REFERENCE -i DEDUPED_BAM \
    [--interval INTERVAL_FILE] --algo DNAscope [-d dbSNP] \
    --pcr_indel_model none --model DNASCOPE_MODEL/dnascope.model \
    TMP_VARIANT_VCF
else
    sentieon driver -t NUMBER_THREADS -r REFERENCE -i DEDUPED_BAM \
    [--interval INTERVAL_FILE] --algo DNAscope [-d dbSNP] \
    --model DNASCOPE_MODEL/dnascope.model TMP_VARIANT_VCF
fi
sentieon driver -t NUMBER_THREADS -r REFERENCE --algo DNAModelApply \
  --model DNASCOPE_MODEL/dnascope.model -v TMP_VARIANT_VCF VARIANT_VCF
```

Reminder

It is important to add option `--pcr_indel_model NONE` when running `DNAscope` if the data you are using is PCR Free.

Depending on whether PCR is involved, `DNAscope` uses different priors for finding significant INDEL variants, which could be controlled by the `--pcr_indel_model` option. The default `--pcr_indel_model` setting is for PCR samples. Thus it is important to set `--pcr_indel_model none` for PCR Free samples.

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **DEDUPED_BAM**: the location of the input BAM file.
- **TMP_VARIANT_VCF**: the location and filename of the variant calling output of DNAscope. This is a temporary file.

- **VARIANT_VCF**: the location and filename of the variant calling output. A corresponding index file will be created. The tool will output a compressed file by using .gz extension.
- **DNASCOPE_MODEL**: the location of the machine learning model file. In the DNAscope command the model will be used to determine the settings used in alignment and variant calling.

Reminder

When running DNAscope with a machine learning model, most of the advanced settings are determined by the model itself, and setting specific values to options other than the `--pcr_indel_model` option could negatively affect the results.

The following inputs are optional for the command:

- **INTERVAL_FILE**: the location of the BED file.
- **dbSNP**: the location of the Single Nucleotide Polymorphism database (dbSNP) that will be used to label known variants. You can only use one dbSNP file.

4.2.2.2 Using DNAscope to produce GVCF output files

From version 202112.04 onwards, DNAscope accepts a model to produce variant calls in the Genomic VCF (GVCF) format. The GVCF format contains additional information on sites that are homozygous for the reference allele in the sample being processed. Recently trained DNAscope models are required and using a DNAscope model trained with Sentieon version 202112.01 or earlier will result in an error.

Two individual commands are run to call variants and to apply the machine learning model. The input BAM file should come from a pipeline where only alignment and deduplication have been performed, to match the model creation methodology.

```
PCRFREE=true #PCRFREE=true means the sample is PCRFree, change it to false for PCR samples.
if [ "$PCRFREE" = true ] ; then
    sentieon driver -t NUMBER_THREADS -r REFERENCE -i DEDUPED_BAM \
    [--interval INTERVAL_FILE] --algo DNAscope [-d dbSNP] \
    --pcr_indel_model none --model DNASCOPE_MODEL/dnascope.model \
    --emit_mode gvcf TMP_VARIANT_GVCF
else
    sentieon driver -t NUMBER_THREADS -r REFERENCE -i DEDUPED_BAM \
    [--interval INTERVAL_FILE] --algo DNAscope [-d dbSNP] \
    --model DNASCOPE_MODEL/dnascope.model --emit_mode gvcf TMP_VARIANT_GVCF
fi
sentieon driver -t NUMBER_THREADS -r REFERENCE --algo DNAModelApply \
  --model DNASCOPE_MODEL/dnascope.model -v TMP_VARIANT_GVCF VARIANT_GVCF
```

Reminder

It is important to add option `--pcr_indel_model NONE` when running DNAscope if the data you are using is PCR Free.

Depending on whether PCR is involved, DNAscope uses different priors for finding significant INDEL variants, which could be controlled by the `--pcr_indel_model` option. The default `--pcr_indel_model` setting is for PCR samples. Thus it is important to set `--pcr_indel_model none` for PCR Free samples.

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **DEDUPED_BAM**: the location of the input BAM file.
- **TMP_VARIANT_GVCF**: the location and filename of the GVCF output of DNAscope. This is a temporary file.
- **VARIANT_GVCF**: the location and filename of the GVCF output. A corresponding index file will be created. The tool will output a compressed file by using .gz extension.
- **DNASCOPE_MODEL**: the location of the machine learning model file. In the DNAscope command the model will be used to determine the settings used in variant calling.

The following inputs are optional for the command:

- **INTERVAL_FILE**: the location of the BED file.
- **dbSNP**: the location of the Single Nucleotide Polymorphism database (dbSNP) that will be used to label known variants. You can only use one dbSNP file.

4.2.2.3 Genotyping GVCF files generated by DNAscope

The GVCF output file can be genotyped either individually, or jointly with GVCFs from other samples using the GVCFtyper algo in Sentieon version 202112.06 and later and will output a single sample or multi-sample VCF.

```
sentieon driver -r REFERENCE --algo GVCFtyper \
  -v s1_VARIANT_GVCF -v s2_VARIANT_GVCF -v s3_VARIANT_GVCF VARIANT_VCF
```

Please check the Sentieon manual for additional details about the *GVCFtyper ALGORITHM*.

Reminder

GVCFtyper can be used genotype DNAscope GVCFs from multiple sequencing platforms into a single multi-sample VCF.

GVCFtyper does not support joint genotyping of DNAscope GVCFs mixed with GVCFs produced by DNAscope without a machine learning model or DNAscope GVCFs mixed with GVCFs produced by other tools.

4.2.3 Structural variant calling

In order to perform structural variant calling you need to add the option to output break-end (BND) information to the DNAscope command; this is done by enabling the bnd (break-end) variant type. Two individual commands are run to perform structural variant calling.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i DEDUPED_BAM \
  --algo DNAscope --var_type bnd \
  [-d dbSNP] TMP_VARIANT_VCF
sentieon driver -t NUMBER_THREADS -r REFERENCE --algo SVSolver  \
  -v TMP_VARIANT_VCF STRUCTURAL_VARIANT_VCF
```

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **DEDUPED_BAM**: the location of the input BAM file.
- **TMP_VARIANT_VCF**: the location and filename of the variant calling output file from DNAscope, which includes the BND information. This is a temporary file when calling structural variants.
- **STRUCTURAL_VARIANT_VCF**: the location and filename of the variant calling output file containing the structural variants. A corresponding index file will be created. The tool will output a compressed file by using .gz extension.

The following inputs are optional for the command:

- **dbSNP**: the location of the Single Nucleotide Polymorphism database (dbSNP) that will be used to label known variants. You can only use one dbSNP file.

Note that structural variant calling is incompatible with the DNAscope model input file. In particular, the `--model` option should be avoided when `--var_type BND` is set.

5 Typical usage for TNseq®

Another of the typical uses of Sentieon® Genomics software is to perform the bioinformatics pipeline for Tumor-Normal analysis recommended in the Broad institute Somatic short variant discovery (SNVs + Indels)[9]. *Recommended bioinformatics pipeline for Tumor-Normal analysis* illustrates such a typical bioinformatics pipeline.

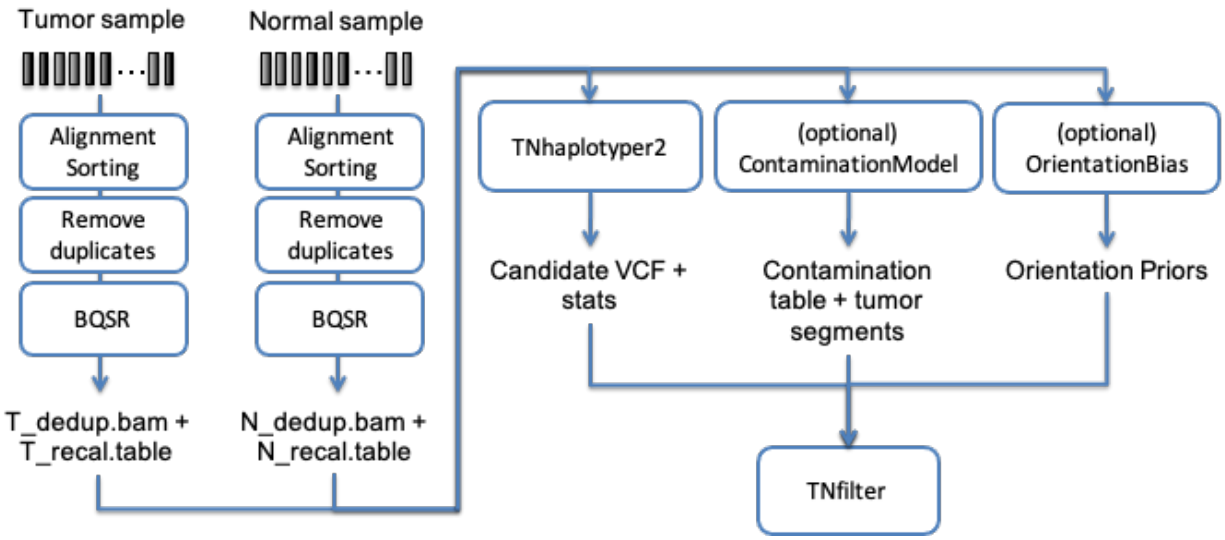


Fig. 5.1: Recommended bioinformatics pipeline for Tumor-Normal analysis

Please check the appnote, *Somatic Variant Calling for SNPs and Indels*, for recommended somatic variant calling pipelines with different inputs.

5.1 General

In this bioinformatics pipeline you will need the following inputs:

- The FASTA file containing the nucleotide sequence of the reference genome corresponding to the sample you will analyze.

[9] https://gatk.broadinstitute.org/hc/en-us/articles/360035894731-Somatic-short-variant-discovery-SNVs-Indels-

- Two sets of FASTQ files containing the nucleotide sequence of the sample to be analyzed, one for the tumor sample and one for the matched normal sample. These files contain the raw reads from the DNA sequencing. The software supports inputting FASTQ files compressed using GZIP. The software only supports files containing quality scores in Sanger format (Phred+33).

You can also include in the pipeline the following optional inputs that will help the algorithms detect artifacts and remove false positives:

- Panel of normal VCF: list of common errors that appear as variants from multiple unrelated normal samples. The contents of this file will be used to identify variants that are more likely to be germline variants, and filter them as such.
- Population resource VCF: list of population allele specific frequencies that will be used for filtering possible germline variants and to annotate the results.

The following steps compose the typical bioinformatics pipeline for a tumor-normal matched pair:

1. Independently pre-process the Tumor and Normal samples using a DNAseq pipeline like the one introduced in *Typical usage for DNAseq®*, with the following stages:
    a. Map reads to reference; you need to make sure that the SM sample tag is different between the tumor and the normal samples, as you will need it as an argument in the somatic variant calling.
    b. Calculate data metrics.
    c. Remove duplicates.
    d. (Optional) Base quality score recalibration (BQSR).
2. Somatic variant discovery, with the following stages:
    a. (Optional) Estimate the cross-sample contamination and tumor segmentation.
    b. (Optional) Estimate any possible orientation bias present in the sequencing.
    c. Somatic candidate variant calling on the two individual BAM files: this step identifies the potential sites where the cancer genome data displays somatic variations relative to the normal genome, and calculates genotypes at that site.
    d. Filter the variants.

5.2 Step by step usage

For the mapping, duplicate removal, and base quality score recalibration stages, please refer to *Step by step usage for DNAseq®* for detailed usage instructions.

5.2.1 Variant discovery with matched normal sample

Two commands are run to call variants on the tumor-normal matched pair. We recommend that the optional steps of estimation of cross-sample contamination and estimation of orientation bias be run in the same command as TNhaplotyper to improve performance.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE \
  -i TUMOR_DEDUPED_BAM [-q TUMOR_RECAL_DATA.TABLE] \
  -i NORMAL_DEDUPED_BAM [-q NORMAL_RECAL_DATA.TABLE] \
  --algo TNhaplotyper2 --tumor_sample TUMOR_SAMPLE_NAME \
     --normal_sample NORMAL_SAMPLE_NAME \
     [--germline_vcf GERMLINE_RESOURCE] \
     [--pon PANEL_OF_NORMAL] \
     TMP_OUT_TN_VCF \
```

(continues on next page)

(continued from previous page)

```
   [ --algo OrientationBias --tumor_sample TUMOR_SAMPLE_NAME \
      ORIENTATION_DATA ] \
   [ --algo ContaminationModel --tumor_sample TUMOR_SAMPLE_NAME \
      --normal_sample NORMAL_SAMPLE_NAME \
      --vcf GERMLINE_RESOURCE \
      --tumor_segments CONTAMINATION_DATA.segments \
      CONTAMINATION_DATA ]

sentieon driver -r REFERENCE \
   --algo TNfilter --tumor_sample TUMOR_SAMPLE_NAME \
   --normal_sample NORMAL_SAMPLE_NAME \
   -v TMP_OUT_TN_VCF \
   [--contamination CONTAMINATION_DATA] \
   [--tumor_segments CONTAMINATION_DATA.segments] \
   [--orientation_priors ORIENTATION_DATA] \
   OUT_TN_VCF
```

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **TUMOR_DEDUPED_BAM**: the location of the pre-processed BAM file after deduplication for the TUMOR sample.
- **NORMAL_DEDUPED_BAM**: the location of the pre-processed BAM file after deduplication for the NORMAL sample.
- **TUMOR_SAMPLE_NAME**: sample name used for tumor sample in Map reads to reference stage.
- **NORMAL_SAMPLE_NAME**: sample name used for normal sample in Map reads to reference stage.
- **TMP_OUT_TN_VCF**: the location and file name of the output file from TNhaplotyper2; this is a temporary file.
- **OUT_TN_VCF**: the location and file name of the output file containing the variants.

The following inputs are optional for the command:

- **TUMOR_RECAL_DATA.TABLE**: the location where the BQSR stage for the TUMOR sample stored the result.
- **NORMAL_RECAL_DATA.TABLE**: the location where the BQSR stage for the NORMAL sample stored the result.
- **PANEL_OF_NORMAL**: the location and name of panel of normal VCF file.
- **GERMLINE_RESOURCE**: the location of the population germline resource. The AF INFO field in the resource file will be used to annotate variant alleles with the population allele frequencies, which will then be used to identify possible germline variants that are not true somatic variants.
- **ORIENTATION_DATA**: the location and file name of the file containing the orientation bias information produced by OrientationBias.
- **CONTAMINATION_DATA**: the location and file name of the file containing the contamination information produced by ContaminationModel.

- **CONTAMINATION_DATA.segments**: the location and file name of the file containing the tumor segments information produced by ContaminationModel.

5.2.2 Special considerations when missing a matched normal sample

When missing a matched normal sample, we recommend the following changes to the pipeline shown in the previous section:

- The *PANEL_OF_NORMAL* and *GERMLINE_RESOURCE* should be included, as they replace the missing matched normal sample. Missing both the matched normal and these germline resources will result in a large number of germline false positive calls in the final output VCF.
- The *–normal_sample* argument and the input BAM and recalibration table for the normal sample should not be included.

5.2.3 Generating GERMLINE_RESOURCE germline population resource

The germline population resource file containing the population allele frequencies can be obtained from gnomAD[10] by post-processing the file following this procedure:

- Download all per chromosome .vcf.bgz files for the version you want.
- For each chromosome, remove unnecessary annotations to reduce file size:

```
bcftools annotate -x ^INFO/AF,INFO/AC INPUT.chrN.vcf.bgz | bcftools norm -m␣
↪+any -Oz -o OUTPUT.chrN.vcf.gz
```

- Concatenate all files onto a single file, and create the corresponding index:

```
bcftools concat -Oz -o OUTPUT.vcf.gz OUTPUT.chr*.vcf.gz && bcftools index -t␣
↪OUTPUT.vcf.gz
```

For instance, to download the GnomAD v3.1 release from Google, you can run:

```
for chr in 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 X Y; do
        curl https://storage.googleapis.com/gcp-public-data--gnomad/release/3.1/
↪vcf/genomes/gnomad.genomes.v3.1.sites.chr$chr.vcf.bgz \
          | bcftools annotate -x ^INFO/AF,INFO/AC  - | bcftools norm -m +any -Oz -
↪o tmp_OUTPUT.chr$chr.vcf.gz
        file_list="$file_list tmp_OUTPUT.chr$chr.vcf.gz"
done
bcftools concat -Oz -o OUTPUT.vcf.gz $file_list && bcftools index -t OUTPUT.vcf.gz
rm $file_list
```

[10] https://gnomad.broadinstitute.org/downloads/

6 Typical usage for TNscope®

A use of Sentieon® Genomics software is to perform the bioinformatics pipeline for Tumor only or Tumor-Normal analysis using the new TNscope® algorithms for somatic variant and structural variant detection. *Recommended pipeline for TNscope® based Tumor-Normal analysis* illustrates such a typical bioinformatics pipeline.

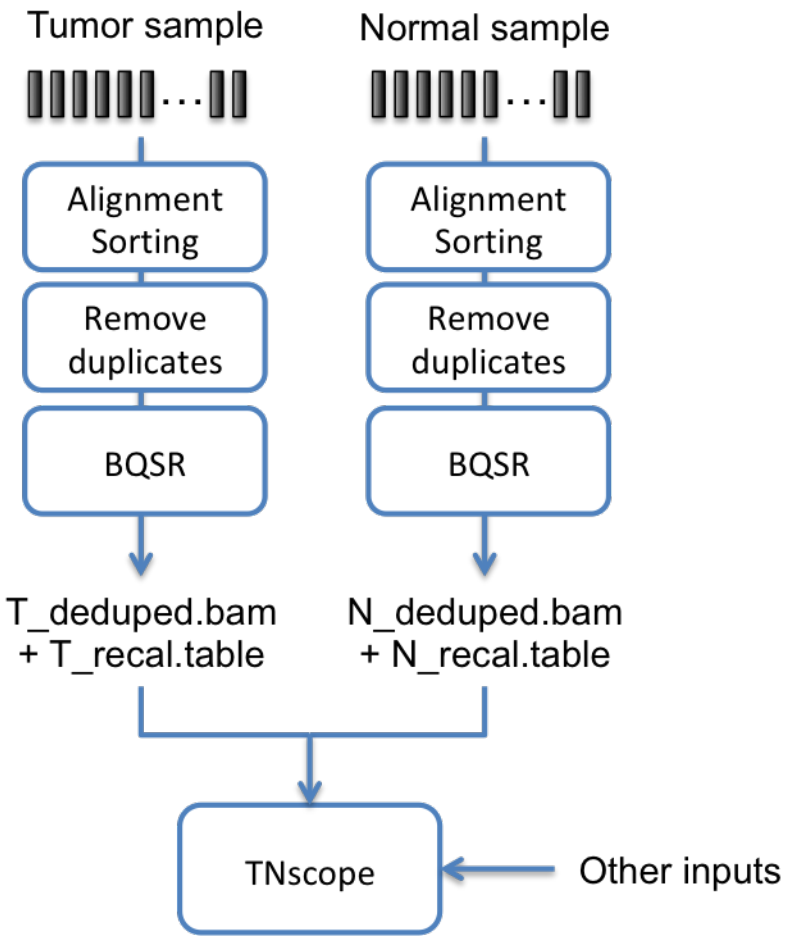


Fig. 6.1: Recommended pipeline for TNscope® based Tumor-Normal analysis

Please check the appnote, *Somatic Variant Calling for SNPs and Indels*, for recommended somatic variant calling pipelines with different inputs.

6.1 General

In this bioinformatics pipeline you will need the following inputs:

- The FASTA file containing the nucleotide sequence of the reference genome corresponding to the sample you will analyze.
- Two sets of FASTQ files containing the nucleotide sequence of the sample to be analyzed, one for the tumor sample and one for the matched normal sample. These files contain the raw reads from the DNA sequencing. The software supports inputting FASTQ files compressed using GZIP. The software only supports files containing quality scores in Sanger format (Phred+33).
- (Optional) The Single Nucleotide Polymorphism database (dbSNP) data that you want to include in the pipeline. The data is used in the form of a VCF file.
- (Optional) Multiple collections of known sites that you want to include in the pipeline. The data is used in the form of a VCF file.

The following steps compose the typical bioinformatics pipeline for a tumor-normal matched pair:

1. Independently pre-process the Tumor and Normal samples using a DNA seq pipeline like the one introduced in *Typical usage for DNAseq®*, with the following stages:
    a. Map reads to reference; you need to make sure that the SM sample tag is different between the tumor and the normal samples, as you will need it as an argument in the somatic variant calling. You also need to make sure that the RG:ID for both samples is different and unique.
    b. Calculate data metrics.
    c. Remove duplicates.
    d. (Optional) Base quality score recalibration (BQSR).
2. Somatic variant calling: this step identifies the sites where the cancer genome data displays somatic variations relative to the normal genome, and calculates genotypes at that site.

6.2 Step by step usage

For the mapping, duplicate removal, and base quality score recalibration stages, please refer to *Step by step usage for DNAseq®* for detailed usage instructions.

6.2.1 Somatic variant discovery with or without matched normal sample

A single command is run to call variants on the tumor-normal matched pair.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE \
  -i TUMOR_DEDUPED_BAM [-q TUMOR_RECAL_DATA.TABLE] \
  -i NORMAL_DEDUPED_BAM [-q NORMAL_RECAL_DATA.TABLE] \
  --algo TNscope \
  --tumor_sample TUMOR_SAMPLE_NAME --normal_sample NORMAL_SAMPLE_NAME \
  [--dbsnp DBSNP] OUT_TN_VCF
```

If you do not have a matched normal sample, you can skip the normal sample BAM file and sample name inputs, as those are not required; however, due to the absence of normal sample, germline variants will be present in the output file:

```
sentieon driver -t NUMBER_THREADS -r REFERENCE \
  -i TUMOR_DEDUPED_BAM [-q TUMOR_RECAL_DATA.TABLE] \
  --algo TNscope --tumor_sample TUMOR_SAMPLE_NAME \
  [--dbsnp DBSNP] OUT_TN_VCF
```

In order to filter out germline variants when missing a matched normal sample, you can replace the matched normal sample with a generic panel of normal file generated using the methodology described below in *Generating a Panel of Normal VCF file*.

A single command is run to call variants on the tumor only sample.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE \
  -i TUMOR_DEDUPED_BAM [-q TUMOR_RECAL_DATA.TABLE] \
  --algo TNscope --tumor_sample TUMOR_SAMPLE_NAME \
  --pon PANEL_OF_NORMAL_VCF [--dbsnp DBSNP] OUT_TN_VCF
```

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **TUMOR_DEDUPED_BAM**: the location of the pre-processed BAM file after deduplication for the TUMOR sample.
- **NORMAL_DEDUPED_BAM**: the location of the pre-processed BAM file after deduplication for the NORMAL sample.
- **TUMOR_SAMPLE_NAME**: sample name used for tumor sample in Map reads to reference stage.
- **OUT_TN_VCF**: the location and file name of the output file containing the variants.

The following inputs are optional for the command:

- **TUMOR_RECAL_DATA.TABLE**: the location where the BQSR stage for the TUMOR sample stored the result.
- **NORMAL_RECAL_DATA.TABLE**: the location where the BQSR stage for the NORMAL sample stored the result.
- **NORMAL_SAMPLE_NAME**: sample name used for normal sample in Map reads to reference stage.
- **DBSNP**: the location of the Single Nucleotide Polymorphism database (dbSNP). The variants in the dbSNP will be more likely to be marked as germline as they require more evidence of absence in the normal. You can only use one dbSNP file.
- **PANEL_OF_NORMAL_VCF**: the location and name of panel of normal VCF file.

6.2.2 Generating a Panel of Normal VCF file

In order to generate your own Panel of Normal VCF file to use with TNscope®, you will need to run the following command on every normal sample you want to use in the panel:

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i NORMAL_DEDUPED_BAM \
   [-q NORMAL_RECAL_DATA.TABLE] --algo TNscope \
   --tumor_sample NORMAL_SAMPLE_NAME OUT_NORMAL_VCF
```

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **NORMAL_DEDUPED_BAM**: the location of the pre-processed BAM file after deduplication for the NORMAL sample.
- **NORMAL_SAMPLE_NAME**: sample name used for normal sample in Map reads to reference stage.
- **OUT_NORMAL_VCF**: the location and name of the output VCF file containing the relevant variants for the input normal sample.

The following inputs are optional for the command:

- **NORMAL_RECAL_DATA.TABLE**: the location where the BQSR stage for the NORMAL sample stored the result.

After you have generated all VCF files you want to include in the panel, you need to merge them into a single Panel of Normal VCF. You can use bcftools[11] for that purpose:

```
BCF=/path_to_bcftools
export BCFTOOLS_PLUGINS=$BCF/plugins
DIR=/path_to_normal_vcf_file
$BCF/bcftools merge -m all -f PASS,. --force-samples $DIR/*.vcf.gz |\
$BCF/bcftools plugin fill-AN-AC |\
$BCF/bcftools filter -i 'SUM(AC)>1' > panel_of_normal.vcf
```

[11] https://samtools.github.io/bcftools/bcftools.html

7 Typical usage for RNA variant calling

One of the typical uses of Sentieon® Genomics software is to perform the bioinformatics pipeline for RNA analysis recommended in the Broad institute best practices described in https://gatk.broadinstitute.org/hc/en-us/articles/360035531192-RNAseq-short-variant-discovery-SNPs-Indels-.

7.1 General

In this bioinformatics pipeline you will need the following inputs:

- The FASTA file containing the nucleotide sequence of the reference genome corresponding to the sample you will analyze.
- One or multiple FASTQ files containing the nucleotide sequence of the sample to be analyzed. These files contain the raw reads from the DNA sequencing. The software supports inputting FASTQ files compressed using GZIP. The software only supports files containing quality scores in Sanger format (Phred+33).
- (Optional) The Single Nucleotide Polymorphism database (dbSNP) data that you want to include in the pipeline. The data is used in the form of a VCF file; you can use a VCF file compressed with bgzip and indexed.
- (Optional) Multiple collections of known sites that you want to include in the pipeline. The data is used in the form of a VCF file; you can use a VCF file compressed with bgzip and indexed.

The following steps compose the typical bioinformatics pipeline:

1. Map reads to reference: This step aligns the reads contained in the FASTQ files to map to a reference genome contained in the FASTA file. This step ensures that the data can be placed in context.
2. Remove duplicates: This step detects reads indicative that the same RNA molecules were sequenced several times. These duplicates are not informative and should not be counted as additional evidence.
3. Split reads at junction: This step splits the RNA reads into exon segments by getting rid of Ns while maintaining grouping information, and hard-clips any sequences overhanging into the intron regions. Additionally, the step will reassign the mapping qualities from STAR to be consistent with what is expected in subsequent steps by converting from quality 255 to 60.
4. (Optional) Base quality score recalibration (BQSR): This step modifies the quality scores assigned to individual read bases of the sequence read data. This action removes experimental biases caused by the sequencing methodology.

5. Variant calling: This step identifies the sites where your data displays variation relative to the reference genome, and calculates genotypes for each sample at that site.

7.2 Step by step usage

7.2.1 Map reads to reference

A single command is run to efficiently perform the alignment using STAR, and the creation of the BAM file and the sorting using Sentieon® software:

```
sentieon STAR --runThreadN NUMBER_THREADS --genomeDir STAR_REFERENCE \
  --readFilesIn SAMPLE SAMPLE2 --readFilesCommand "zcat" \
  --outStd BAM_Unsorted --outSAMtype BAM Unsorted --outBAMcompression 0 \
  --outSAMattrRGline ID:GROUP_NAME SM:SAMPLE_NAME PL:PLATFORM \
  --twopassMode Basic --twopass1readsN -1 --sjdbOverhang READ_LENGTH_MINUS_1 \
  | sentieon util sort -r REFERENCE -o SORTED_BAM -t NUMBER_THREADS -i -
```

Inputs and options for STAR are described in its manual[12].

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system. We recommend that you use the same number of threads for both STAR and for the util binary.
- **STAR_REFERENCE**: the location of the reference FASTA file STAR genomeDir. You should make sure that all the necessary reference data required by STAR is available in the same location with consistent naming.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the FASTA file and corresponding FAI index file match the build and naming convention of the one in the **STAR_REFERENCE** genomeDir.
- **SAMPLE**: the location of the sample FASTQ file. If the data comes from pair ended sequencing technology, you will also need to input **SAMPLE2** as the corresponding pair sample FASTQ file. You will need to make sure that the program used in option *–readFilesCommand* matches the compression status of the input FASTQ, i.e.: if the FASTQ files are compressed, you should use *zcat*, while if the FASTQ files are not compressed, you should use *cat*.
- **GROUP_NAME**: Readgroup identifier that will be added to the readgroup header line. The `RG:ID` needs to be unique among all the datasets that you plan on using.
- **SAMPLE_NAME**: name of the sample that will be added to the readgroup header line.
- **PLATFORM**: name of the sequencing platform used to sequence the DNA. Possible options are ILLUMINA when the fastq files have been produced in an Illumina™ machine; IONTORRENT when the fastq files have been produced in a Life Technologies™ Ion-Torrent™ machine; ELEMENT when the fastq files have been produced in an Element Biosciences™ machine; DNBSEQ when the fastq files have been produced in a MGI™ machine; ULTIMA when the fastq files have been produced in an Ultima Genomics™ machine.
- **READ_LENGTH_MINUS_1**: the read length of your input data, minus 1.
- **SORTED_BAM**: the location and filename of the sorted mapped BAM output file. A corresponding index file (.bai) will be created.

[12] https://github.com/alexdobin/STAR/blob/master/doc/STARmanual.pdf

7.2.2 Remove or mark duplicates

Two individual commands are run to remove or mark duplicates on the BAM file after alignment and sorting. The first command collects read information, and the second command performs the deduplication; the option `--rmdup` controls whether the duplicated reads are removed, if the option is present, or only marked as duplicated.

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --rna [--consensus] [--umi_tag XR]
  --fun score_info SCORE.gz
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo Dedup [--rmdup] --score_info SCORE.gz DEDUPED_BAM
```

The following inputs are required for the commands:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **SORTED_BAM**: the location where the previous mapping stage stored the result.
- **SCORE.gz**: the location and filename of the temporary score output file. Make sure that the same file is used for both commands.
- **DEDUPED_BAM**: the location and filename of the deduped BAM output file. A corresponding index file (.bai) will be created.

7.2.3 Split reads at junction

A single command is run to perform the splitting of reads into exon segments and reassigning the mapping qualities from STAR.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i DEDUPED_BAM \
  --algo RNASplitReadsAtJunction --reassign_mapq 255:60 SPLIT_BAM
```

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **DEDUPED_BAM**: the location where the previous deduping stage stored the result.
- **SPLIT_BAM**: the location and filename of the BAM output file containing the split reads. A corresponding index file (.bai) will be created.

7.2.4 Base quality score recalibration (BQSR; optional)

The Base quality score recalibration (BQSR) stage has identical usage to the DNAseq stage; for this stage, you can refer to *Step by step usage for DNAseq®* for detailed usage instructions.

7.2.5 RNA Variant calling

A single command is run to call variants and additionally apply the BQSR calculated before. The RNA variant calling can be done using either the Haplotyper algorithm or the DNAscope algorithm. For the command you should use the option `--trim_soft_clip` and a lower minimum phred-scaled confidence threshold than for DNAseq® variant calling, which means you should set `call_conf` to 20 and `emit_conf` to 20 instead of the default of 30.

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i SPLIT_BAM \
  [-q RECAL_DATA.TABLE] --algo Haplotyper --trim_soft_clip  \
  --call_conf 20 --emit_conf 20 [-d dbSNP] VARIANT_VCF
```

If you want to use DNAscope for the calling, the command would be:

```
sentieon driver -t NUMBER_THREADS -r REFERENCE -i SPLIT_BAM \
  [-q RECAL_DATA.TABLE] --algo DNAscope --trim_soft_clip  \
  --call_conf 20 --emit_conf 20 [-d dbSNP] VARIANT_VCF
```

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **SPLIT_BAM**: the location where the previous RNASplitReadsAtJunction stage stored the result.
- **VARIANT_VCF**: the location and filename of the variant calling output file. A corresponding index file will be created. The tool will output a compressed file by using .gz extension.

The following inputs are optional for the command:

- **RECAL_DATA.TABLE**: the location where the previous BQSR stage stored the result.
- **dbSNP:** the location of the Single Nucleotide Polymorphism database (dbSNP) that will be used to label known variants. You can only use one dbSNP file.

8 Examples of tool capabilities and applications

Additional examples of typical pipelines can be found in the Sentieon GitHub page in https://github.com/Sentieon/sentieon-scripts/tree/master/example_pipelines.

8.1 DNA pipeline example script

Below is an example of how to process a set of 2 fastq files for a sample and perform variant calling according to the Broad institute best practices described in https://www.broadinstitute.org/gatk/guide/best-practices.

```
#!/bin/sh
# ****************************************
# Script to perform DNA seq variant calling
# using a single sample with fastq files
# named 1.fastq.gz and 2.fastq.gz
# ****************************************

# Update with the fullpath location of your sample fastq
fastq_folder="/home/pipeline/samples"
fastq_1="1.fastq.gz"
fastq_2="2.fastq.gz" #If using Illumina paired data
sample="sample_name"
group="read_group_name"
platform="ILLUMINA"

# Update with the location of the reference data files
fasta="/home/regression/references/b37/human_g1k_v37_decoy.fasta"
dbsnp="/home/regression/references/b37/dbsnp_138.b37.vcf.gz"
known_Mills_indels="/home/regression/references/b37/Mills_and_1000G_gold_standard.indels.b37.
↪vcf.gz"
known_1000G_indels="/home/regression/references/b37/1000G_phase1.indels.b37.vcf.gz"

# Update with the location of the Sentieon software package and license file
export SENTIEON_INSTALL_DIR=/home/release/sentieon-genomics-202010
export SENTIEON_LICENSE=/home/Licenses/Sentieon.lic

# Other settings
```

(continues on next page)

(continued from previous page)

```
nt=$(nproc) #number of threads to use in computation, set to number of cores in the server
workdir="$PWD/test/DNAseq" #Determine where the output files will be stored

# ****************************************
# 0. Setup
# ****************************************
mkdir -p $workdir
logfile=$workdir/run.log
exec >$logfile 2>&1
cd $workdir

# ****************************************
# 1. Mapping reads with BWA-MEM, sorting
# ****************************************
#The results of this call are dependent on the number of threads used. To have number of␣
↪threads independent results, add chunk size option -K 10000000
( $SENTIEON_INSTALL_DIR/bin/sentieon bwa mem -R "@RG\tID:$group\tSM:$sample\tPL:$platform" -
↪t $nt -K 10000000 $fasta $fastq_folder/$fastq_1 $fastq_folder/$fastq_2 || echo -n 'error'␣
↪) | $SENTIEON_INSTALL_DIR/bin/sentieon util sort -r $fasta -o sorted.bam -t $nt --sam2bam -
↪i -

# ****************************************
# 2. Metrics
# ****************************************
$SENTIEON_INSTALL_DIR/bin/sentieon driver -r $fasta -t $nt -i sorted.bam --algo␣
↪MeanQualityByCycle mq_metrics.txt --algo QualDistribution qd_metrics.txt --algo GCBias --
↪summary gc_summary.txt gc_metrics.txt --algo AlignmentStat --adapter_seq '' aln_metrics.
↪txt --algo InsertSizeMetricAlgo is_metrics.txt
$SENTIEON_INSTALL_DIR/bin/sentieon plot GCBias -o gc-report.pdf gc_metrics.txt
$SENTIEON_INSTALL_DIR/bin/sentieon plot QualDistribution -o qd-report.pdf qd_metrics.txt
$SENTIEON_INSTALL_DIR/bin/sentieon plot MeanQualityByCycle -o mq-report.pdf mq_metrics.txt
$SENTIEON_INSTALL_DIR/bin/sentieon plot InsertSizeMetricAlgo -o is-report.pdf is_metrics.txt

# ****************************************
# 3. Remove Duplicate Reads. It is possible
# to mark instead of remove duplicates
# by ommiting the --rmdup option in Dedup
# ****************************************
$SENTIEON_INSTALL_DIR/bin/sentieon driver -t $nt -i sorted.bam --algo LocusCollector --fun␣
↪score_info score.txt
$SENTIEON_INSTALL_DIR/bin/sentieon driver -t $nt -i sorted.bam --algo Dedup --rmdup --score_
↪info score.txt --metrics dedup_metrics.txt deduped.bam

# ****************************************
# 5. Base recalibration
# ****************************************
$SENTIEON_INSTALL_DIR/bin/sentieon driver -r $fasta -t $nt -i deduped.bam --algo QualCal -k
↪$dbsnp -k $known_Mills_indels -k $known_1000G_indels recal_data.table
$SENTIEON_INSTALL_DIR/bin/sentieon driver -r $fasta -t $nt -i deduped.bam -q recal_data.
↪table --algo QualCal -k $dbsnp -k $known_Mills_indels -k $known_1000G_indels recal_data.
↪table.post
$SENTIEON_INSTALL_DIR/bin/sentieon driver -t $nt --algo QualCal --plot --before recal_data.
```

(continues on next page)

(continued from previous page)

```
↪table --after recal_data.table.post recal.csv
$SENTIEON_INSTALL_DIR/bin/sentieon plot QualCal -o recal_plots.pdf recal.csv

# ****************************************
# 6b. HC Variant caller
# ****************************************
$SENTIEON_INSTALL_DIR/bin/sentieon driver -r $fasta -t $nt -i deduped.bam -q recal_data.
↪table --algo Haplotyper -d $dbsnp --emit_conf=30 --call_conf=30 output-hc.vcf.gz

# ****************************************
# 5b. ReadWriter to output recalibrated bam
# This stage is optional as variant callers
# can perform the recalibration on the fly
# using the before recalibration bam plus
# the recalibration table
# ****************************************
$SENTIEON_INSTALL_DIR/bin/sentieon driver -r $fasta -t $nt -i deduped.bam -q recal_data.
↪table --algo ReadWriter recaled.bam
```

8.2 Working with multiple input files

Typically, there will be two occasions when you will be having multiple input files that need to be processed together: a single sample has been sequenced in multiple lanes, or you are processing multiple samples from a cohort.

8.2.1 Working with multiple input files from the same sample

When you have more than one set of input fastq files for the sample, you should perform an individual alignment stage for each set of input fastq files, and then use the multiple sorted BAM files as input of the next stage. The next stage will take care of merging all the BAM files into one. You need to make sure that each set of input fastq files are aligned using the same SM sample name, but a different RG readgroup name, so that the rest of the stages properly deal with the different data.

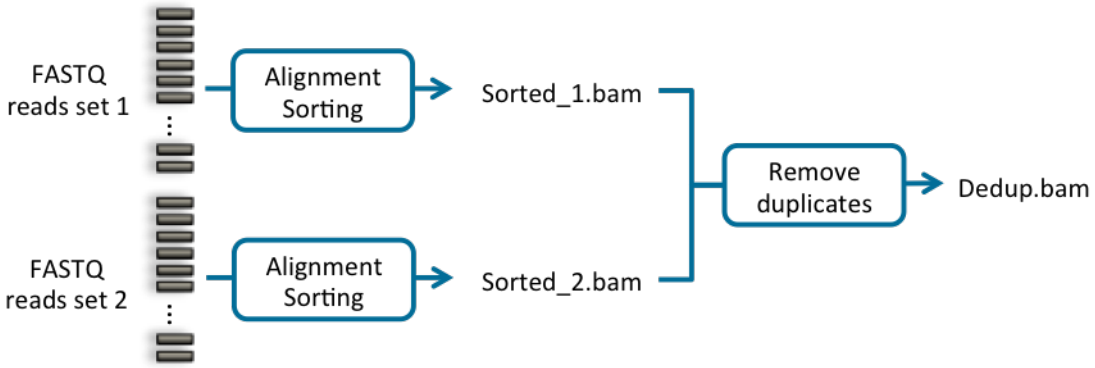


Fig. 8.1: Bioinformatics pipeline using multiple input files from the same sample

```
#Run alignment for 1st input file set
(sentieon bwa mem -R '@RG\tID:GROUP_NAME_1 \tSM:SAMPLE_NAME \tPL:PLATFORM' \
```

(continues on next page)

(continued from previous page)

```
  -t NUMBER_THREADS REFERENCE SAMPLE_1 [SAMPLE_1_2] || echo -n 'error' ) \
  | sentieon util sort -o SORTED_BAM_1 -t NUMBER_THREADS --sam2bam -i -
#Run alignment for 2nd input file set
(sentieon bwa mem -R '@RG\tID:GROUP_NAME_2 \tSM:SAMPLE_NAME \tPL:PLATFORM' \
  -t NUMBER_THREADS REFERENCE SAMPLE_2 [SAMPLE_2_2] || echo -n 'error' \
  sentieon util sort -o SORTED_BAM_2 -t NUMBER_THREADS --sam2bam -i -
#Run dedup on both BAM files
sentieon driver -t NUMBER_THREADS -i SORTED_BAM_1 -i SORTED_BAM_2 \
  --algo LocusCollector --fun score_info SCORE_TXT
sentieon driver -t NUMBER_THREADS -i SORTED_BAM_1 -i SORTED_BAM_2 \
  --algo Dedup --rmdup --score_info SCORE_TXT --metrics DEDUP_METRIC_TXT DEDUPED_BAM
```

Alternatively you can input the results of multiple BWA alignments onto a single util sort command to merge the results onto a single sorted bam files as soon as possible. You still need to make sure that each set of input fastq files are aligned using the same SM sample name, but a different RG readgroup name, so that the rest of the stages properly deal with the different data.

```
#Run alignment for both input file sets sequentially
(sentieon bwa mem -R '@RG\tID:GROUP_NAME_1\tSM:SAMPLE_NAME\tPL:PLATFORM' \
  -t NUMBER_THREADS REFERENCE SAMPLE_1 [SAMPLE_1_2]  && sentieon bwa mem -R \
  '@RG\tID:GROUP_NAME_2 \tSM:SAMPLE_NAME \tPL:PLATFORM' \
  -t NUMBER_THREADS REFERENCE SAMPLE_2 [SAMPLE_2_2] || echo -n 'error' ) \
  | sentieon util sort -o SORTED_COMBINED_BAM -t NUMBER_THREADS --sam2bam -i -
#Run dedup on combined BAM file
sentieon driver -t NUMBER_THREADS -i SORTED_COMBINED_BAM_1 \
  --algo LocusCollector --fun score_info SCORE_TXT
sentieon driver -t NUMBER_THREADS -i SORTED_COMBINED_BAM_1 \
  --algo Dedup --rmdup --score_info SCORE_TXT --metrics DEDUP_METRIC_TXT DEDUPED_BAM
```

8.2.2 Working with multiple input files from different samples

Processing multiple samples from a cohort together, also known as joint variant calling, can be done in two ways:

- Process each sample individually and use the Haplotyper algorithm with option `--emit_mode gvcf` to create a GVCF file containing additional information, then process all GVCF using the GVCFtyper algorithm. This method allows for easy and fast reprocessing after additional samples have been processed.

```
#Process all samples independently until GVCF
for sample in s1 s2 s3; do
  (sentieon bwa mem -R '@RG\tID:GROUP_${sample}\tSM:${sample}\tPL:PLATFORM' \
    -t NUMBER_THREADS REFERENCE ${sample}_FASTQ_1 ${sample}_FASTQ_2 || echo -n \
    'error') | sentieon util sort -o ${sample}_SORTED_BAM -t NUMBER_THREADS \
    --sam2bam -i -
  sentieon driver -t NUMBER_THREADS -i ${sample}_SORTED_BAM \
    --algo LocusCollector --fun score_info ${sample}_SCORE_TXT
  sentieon driver -t NUMBER_THREADS -i SORTED_BAM_1 -i SORTED_BAM_2 \
    --algo Dedup --rmdup --score_info ${sample}_SCORE_TXT ${sample}_DEDUPED_BAM
  sentieon driver -r REFERENCE -t NUMBER_THREADS -i ${sample}_DEDUPED_BAM \
    --algo QualCal -k $known_sites ${sample}_RECAL_DATA_TABLE
  sentieon driver -r REFERENCE -t NUMBER_THREADS -i ${sample}_DEDUPED_BAM \
    -q ${sample}_RECAL_DATA_TABLE --algo Haplotyper --emit_mode gvcf \
```

(continues on next page)

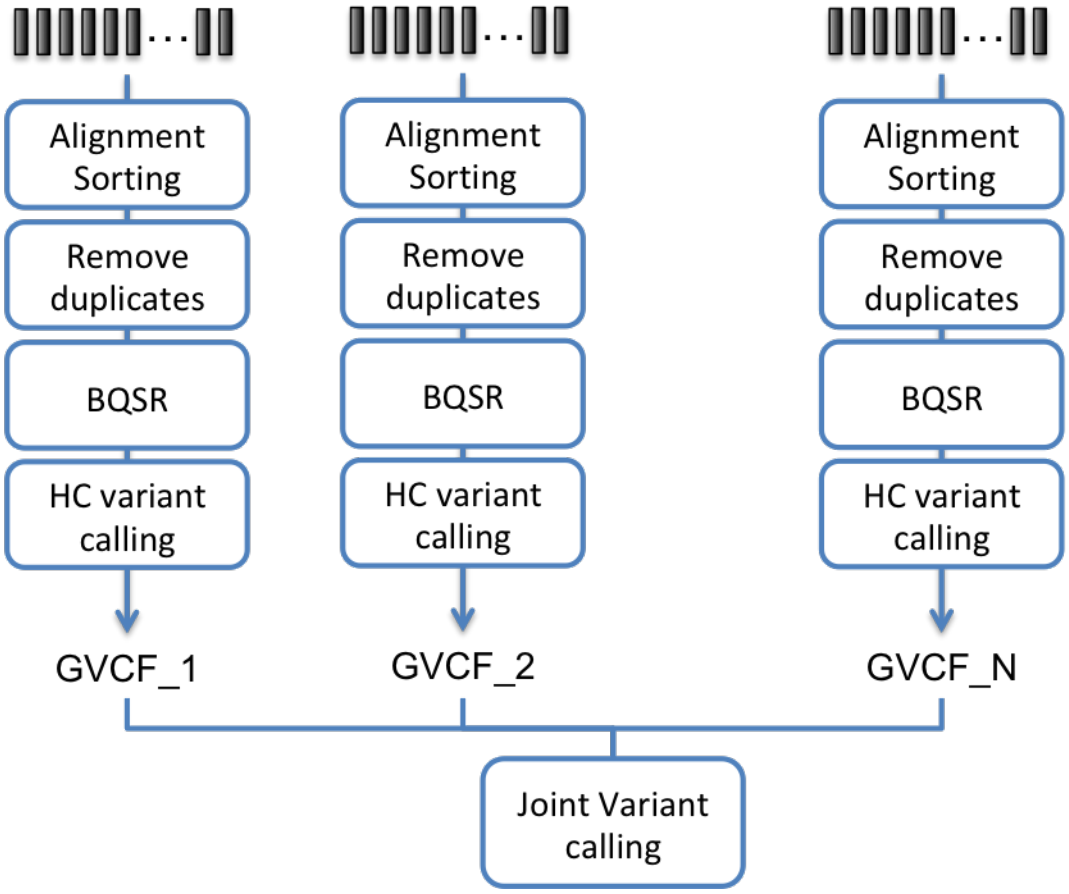


Fig. 8.2: Bioinformatics pipeline for joint calling using Haplotyper in GVCF emit mode and GVCFtyper

(continued from previous page)

```
    ${sample}_VARIANT_GVCF
done
#Run joint variant calling
sentieon driver -r REFERENCE --algo GVCFtyper -v s1_VARIANT_GVCF \
  -v s2_VARIANT_GVCF -v s3_VARIANT_GVCF VARIANT_VCF
```

- Process each sample individually and create either a recalibrated BAM file or a deduped BAM and a recalibration table for each sample, then process all files using the Haplotyper algorithm.

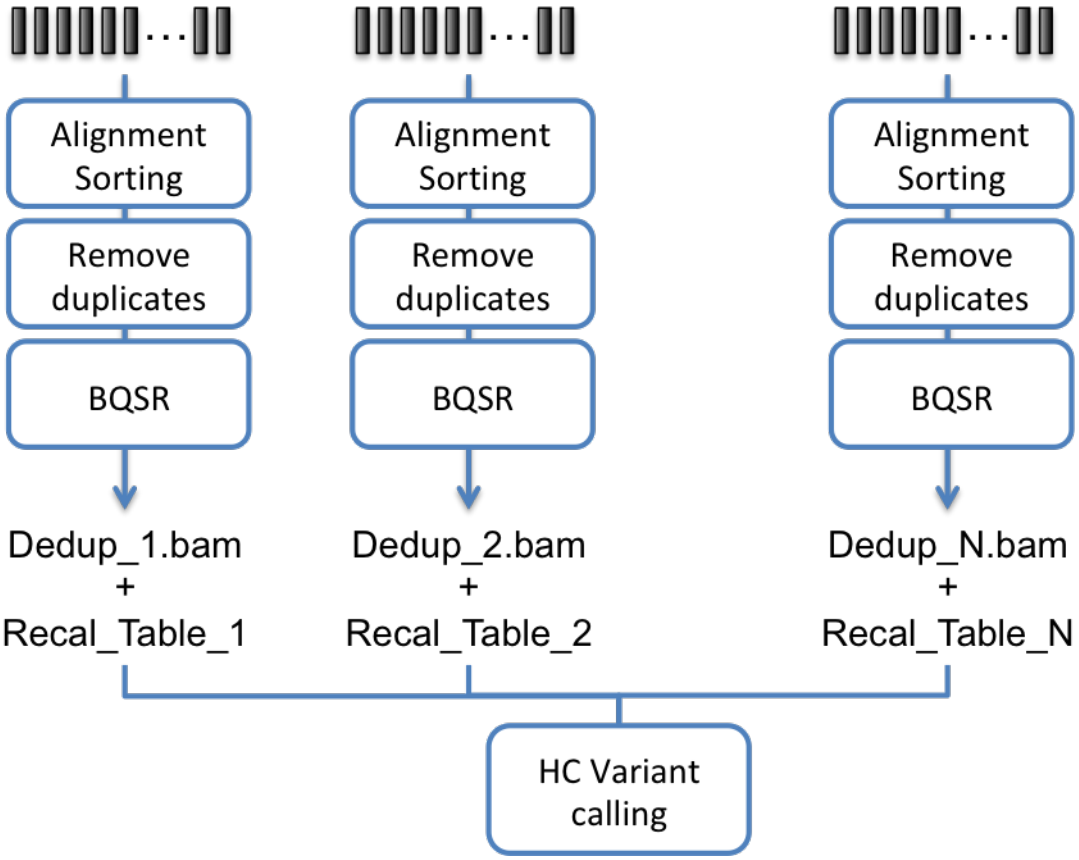


Fig. 8.3: Bioinformatics pipeline for joint calling using Haplotyper and multiple BAM files

```
#Process all samples independently until BQSR
for sample in s1 s2 s3; do
  (sentieon bwa mem -R '@RG\tID:GROUP_${sample}\tSM:${sample}\tPL:PLATFORM' \
    -t NUMBER_THREADS REFERENCE ${sample}_FASTQ_1 ${sample}_FASTQ_2 || echo -n \
    'error' )| sentieon util sort -o ${sample}_SORTED_BAM -t NUMBER_THREADS \
    --sam2bam -i -
  sentieon driver -t NUMBER_THREADS -i ${sample}_SORTED_BAM \
    --algo LocusCollector --fun score_info ${sample}_SCORE_TXT
  sentieon driver -t NUMBER_THREADS -i SORTED_BAM_1 -i SORTED_BAM_2 \
    --algo Dedup --rmdup --score_info ${sample}_SCORE_TXT ${sample}_DEDUPED_BAM
  sentieon driver -r REFERENCE -t NUMBER_THREADS -i ${sample}_DEDUPED_BAM \
    --algo QualCal -k $known_sites ${sample}_RECAL_DATA_TABLE
done
```

(continues on next page)

(continued from previous page)

```
#Run joint variant calling
sentieon driver -t NUMBER_THREADS -i s1_DEDUPED_BAM -q s1_RECAL_DATA_TABLE \
  -i s2_DEDUPED_BAM -q s2_RECAL_DATA_TABLE -i s3_DEDUPED_BAM \
  -q s3_RECAL_DATA_TABLE --algo Haplotyper VARIANT_VCF
```

8.3 Supported annotations in Haplotyper and Genotyper

Both Haplotyper and Genotyper germline variant calling methods accept the option `--annotation` to output additional annotations to the result VCF. The following are the possible annotations supported:

- AC: Allele count in genotypes, for each ALT allele, in the same order as listed
- AF: Allele frequency, for each ALT allele, in the same order as listed
- AN: Total number of alleles in called genotypes
- BaseQRankSum: Z-score from Wilcoxon rank sum test of Alt Vs. Ref base qualities
- ClippingRankSum: Z-score From Wilcoxon rank sum test of Alt vs. Ref number of hard clipped bases
- DP: Combined depth across samples
- ExcessHet: Phred-scaled p-value for exact test of excess heterozygosity
- FS: Phred-scaled p-value using Fisher's exact test to detect strand bias
- InbreedingCoeff: Inbreeding coefficient as estimated from the genotype likelihoods per-sample when compared against the Hardy-Weinberg expectation
- MLEAC: Maximum likelihood expectation (MLE) for the allele counts, for each ALT allele, in the same order as listed
- MLEAF: Maximum likelihood expectation (MLE) for the allele frequency, for each ALT allele, in the same order as listed
- MQ: RMS mapping quality
- MQ0: Number of MAPQ == 0 reads
- MQRankSum: Z-score From Wilcoxon rank sum test of Alt vs. Ref read mapping qualities
- QD: Variant Confidence/Quality by Depth
- RAW_MQ: Raw data for RMS mapping quality
- ReadPosRankSum: Z-score from Wilcoxon rank sum test of Alt vs. Ref read position bias
- SOR: Symmetric Odds Ratio of 2x2 contingency table to detect strand bias
- SAC: Number of reads on the forward and reverse strand supporting each allele (including reference)
- AS_BaseQRankSum: Allele specific
- AS_FS: Allele specific Phred-scaled p-value using Fisher's exact test to detect strand bias
- AS_InbreedingCoeff: Allele specific Inbreeding coefficient as estimated from the genotype likelihoods per-sample when compared against the Hardy-Weinberg expectation
- AS_MQRankSum: Allele specific Z-score From Wilcoxon rank sum test of Alt vs. Ref read mapping qualities
- AS_QD: Allele specific Variant Confidence/Quality by Depth
- AS_MQ: Allele specific RMS mapping quality

- AS_ReadPosRankSum: Allele specific Z-score from Wilcoxon rank sum test of Alt vs. Ref read position bias
- AS_SOR: Allele specific Symmetric Odds Ratio of 2x2 contingency table to detect strand bias

8.4 Removing reads after alignment with low mapping quality

Below is an example of how to remove reads with low mapping quality after alignment, so that they do not take space in the output BAM file.

```
sentieon bwa mem -R '@RG\tID:GROUP_NAME_1 \tSM:SAMPLE_NAME \tPL:PLATFORM' \
  -t NUMBER_THREADS REFERENCE SAMPLE_1 [SAMPLE_2] \
  |samtools view -h -q MAP_QUALITY_THRESHOLD - \
  | sentieon util sort -o SORTED_BAM_1 -t NUMBER_THREADS --sam2bam -i -
```

8.5 Performing Dedup to mark primary and non-primary reads

The standard 2 pass dedup will ignore non-primary reads and never change their duplicate flag.

Below is an example of how to perform dedup to mark both primary and non-primary reads:

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --fun score_info SCORE_TXT
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo Dedup --score_info SCORE_TXT --metrics DEDUP_METRIC_TXT \
  -- output_dup_read_name DUPLICATED_READNAMES_TXT
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo Dedup --rmdup --dup_read_name DUPLICATED_READNAMES_TXT DEDUPED_BAM
```

8.6 Pipeline modifications when using data with quantized quality scores

When your input data has quantized base quality scores, there is a high chance that the base quality reported in the input FASTQ is much higher than the empirical base quality; in this case, it is possible that the variant calling step may take longer to process the data, as low quality bases that would not have been taken into account in the local assembly will cause the generation of many irrelevant low-quality haplotypes. This issue should be addressed by running the Base Quality Score Recalibration step of the pipeline; alternatively, you can modify the default value of the min_base_qual argument of Haplotyper, or TNhaplotyper to reduce the impact: add the option `--min_base_qual 15` to your command line to prevent this issue.

8.7 Modify RG information on BAM files when both Tumor and Normal inputs have the same RGID

If you input multiple different BAM files containing readgroups with the same ID but different attributes, the Sentieon® tools will error out. This can happen when in TNseq® and TNscope® the tumor and normal sample BAM files have an RG ID of "1". In order to work around this, you can use the `--replace_rg` functionality to modify the RG information.

```
#first lane tumor, RGID incorrectly set to uninformative 1
sentieon bwa mem -R '@RG\tID:1\tSM:TUMOR_SM\tPL:PLATFORM' -t NT \
  REFERENCE TUMOR_1_1.fq.gz TUMOR_1_2.fq.gz | sentieon util sort \
```

(continues on next page)

(continued from previous page)

```
  -o TUMOR_1.bam -t NT --sam2bam -i -

#second lane tumor, RGID incorrectly set to uninformative 1
sentieon bwa mem -R '@RG\tID:1\tSM:TUMOR_SM\tPL:PLATFORM' -t NT \
  REFERENCE TUMOR_2_1.fq.gz TUMOR_2_2.fq.gz | sentieon util sort \
  -o TUMOR_2.bam -t NT --sam2bam -i -

#normal, RGID incorrectly set to uninformative 1
sentieon bwa mem -R '@RG\tID:1\tSM:NORMAL_SM\tPL:PLATFORM' -t NT \
  REFERENCE TUMOR_1.fq.gz TUMOR_2.fq.gz | sentieon util sort \
  -o NORMAL.bam -t NT --sam2bam -i -

#using replace_rg argument to modify the RGID and make them unique
#each replace_RG argument will only affect the following input BAM file
sentieon driver -r REFERENCE \
  --replace_rg 1='ID:T_1\tSM:TUMOR\tPL:PLATFORM' -i TUMOR_1.bam \
  --replace_rg 1='ID:T_2\tSM:TUMOR\tPL:PLATFORM' -i TUMOR_2.bam \
  --replace_rg 1='ID:N\tSM:NORMAL\tPL:PLATFORM'-i NORMAL.bam \
  --algo TNscope --tumor_sample TUMOR --normal_sample NORMAL OUT_TN_VCF
```

8.8 Running the license server (LICSRVR) as a system service

8.8.1 Running the license server as a system service using sysvinit

If your system follows the traditional System V init startup scripts you can setup the license server to be automatically started in your system by running the following commands as root:

1. Create and customize the configuration file; the configuration file is typically /etc/sysconfig/licsrvr; however, in Ubuntu the configuration file will be /etc/default/licsrvr. Below is an example of the configuration file, for the recommended setup using the `sentieon` username:

   - `/home/sentieon/release/latest` is a symlink to the installation directory of the latest Sentieon® software package
   - `/home/sentieon/licsrvr` is a folder to run the licsrvr service
   - `/home/sentieon/licsrvr/licsrvr.lic` is the Sentieon® license file

   ```
   licsrvr="/home/sentieon/release/latest/bin/sentieon licsrvr"
   licfile=/home/sentieon/licsrvr/licsrvr.lic
   logfile=/home/sentieon/licsrvr/licsrvr.log
   ```

2. Install the license server startup script into the /etc/init.d directory. The startup script is included with the release package.

   ```
   install -m 0755 $SENTIEON_INSTALL_DIR/doc/licsrvr.sh /etc/init.d/licsrvr
   ```

3. Install and enable the service. Depending on your system, you will run different commands:

   - If your system has Linux Standard Base Core Specifications[13] installed, execute the system init script installation script.

[13] http://refspecs.linuxfoundation.org/LSB_3.1.0/LSB-Core-generic/LSB-Core-generic/tocsysinit.html

```
/usr/lib/lsb/install_initd /etc/init.d/licsrvr
```

- If your system does not have the lsb conformance packages installed, use the `chkconfig` command to enable the service.

```
chkconfig --add licsrvr
chkconfig licsrvr on
```

- For Ubuntu and Debian systems, if you do not have the lsb/install_initd binary and choose not to install the lsb-core package, use the `update-rc.d` command to install and enable the service.

```
update-rc.d licsrvr defaults
update-rc.d licsrvr enable
```

4. You can use the `service` command to start/stop/restart/check status of the service.

```
service licsrvr [start|stop|restart|status]
```

8.8.2 Running the license server as a system service using systemd

You can use the systemd system and service capabilities of your OS to setup the license server to be automatically started in your system. To do so, run the following commands as root:

1. If you are using the licsrvr.service license server startup script found in the doc folder of the Sentieon® software release package, you will need to create the necessary files that the script requires, including using the `sentieon` username:
   - `/home/sentieon/release/latest` is a symlink to the installation directory of the latest Sentieon® software package
   - `/home/sentieon/licsrvr` is a folder to run the licsrvr service
   - `/home/sentieon/licsrvr/licsrvr.lic` is the Sentieon® license file

   Alternatively, you can edit the license server startup script to point to your specific username and/or file location information.
2. Install the license server startup script into the /etc/systemd/system directory.

```
install -m 0644 $SENTIEON_INSTALL_DIR/doc/licsrvr.service /etc/systemd/system
```

3. Run the following command to enable automatic start of the license server on the computer start:

```
systemctl enable licsrvr.service
```

4. You can use the `systemctl` command to manually start and stop the service.

```
systemctl start licsrvr.service

systemctl stop licsrvr.service
```

9 Sentieon-cli

The `sentieon-cli` provides complete implementations of Sentieon® pipelines in single commands. The `sentieon-cli` is designed to improve ease-of-use for complex multi-stage alignment and variant calling pipelines.

Instructions for installing the `sentieon-cli` can be found on GitHub at, https://github.com/Sentieon/sentieon-cli.

9.1 DNAscope

Sentieon® DNAscope is a pipeline for alignment and germline variant calling (SNVs, SVs, indels, and CNVs) from short-read DNA sequence data. The sentieon-cli provides a complete implementation of the DNAscope pipeline in a single command. The pipeline is designed for accuracy, efficiency, and ease-of-use.

The DNAscope pipeline uses a combination of Bayesian statistical approaches and machine learning to achieve high variant calling accuracy. The DNAscope pipeline supports samples sequenced using whole-genome or targeted (hybrid-capture) enrichment library preps.

The pipeline accepts as input aligned reads in BAM or CRAM format, or un-aligned reads in FASTQ, uBAM, or uCRAM format. The pipeline will output variants in the VCF (or gVCF) formats and aligned reads in BAM or CRAM formats.

9.1.1 Setup

9.1.1.1 Prerequisites

- Sentieon® software package version 202308 or higher.
- **samtools**[14] **version 1.16 or higher for**
  alignment of reads in uBAM or uCRAM format or re-alignment of previously aligned reads.
- **MultiQC**[15] **version 1.18 or higher for metrics**
  report generation.

The `sentieon`, `samtools`, and `multiqc` executables will be accessed through the user's `PATH` environment variable.

[14] https://www.htslib.org/
[15] https://multiqc.info/

9.1.1.2 The Reference genome

DNAscope will call variants present in the sample relative to a high quality reference genome sequence. Besides the reference genome file, a samtools fasta index file (.fai) needs to be present. Read alignment also requires bwa index files.

We recommend aligning to a reference genome without alternate contigs. If alternate contigs are present in the genome, please also supply a ".alt" file to activate alt-aware alignment[16] in bwa.

9.1.2 Usage

9.1.2.1 Alignment and variant calling from FASTQ

A single command is run to align, preprocess, and call SNVs, indels, and structural variants from FASTQ:

```
sentieon-cli dnascope [-h] \
  -r REFERENCE \
  --r1_fastq R1_FASTQ ... \
  --r2_fastq R2_FASTQ ... \
  --readgroups READGROUPS ... \
  -m MODEL_BUNDLE \
  [-d DBSNP] \
  [-b INTERVAL_FILE] \
  [--interval_padding INTERVAL_PADDING] \
  [-t NUMBER_THREADS] \
  [--pcr_free] \
  [-g] \
  [--duplicate_marking DUP_MARKING] \
  [--assay ASSAY] \
  [--consensus] \
  [--dry_run] \
  [--bam_format] \
  SAMPLE_VCF
```

With FASTQ input, the DNAscope pipeline requires the following arguments:

- `-r REFERENCE`: the location of the reference FASTA file. A reference fasta index, ".fai" file, and bwa index files, are also required.
- `--r1_fastq R1_FASTQ`: the R1 input FASTQ. Can be used multiple times. `R1_FASTQ` files without a corresponding `R2_FASTQ` are assumed to be single-ended. Be aware that the pipeline performs single-sample processing, and all fastq are expected to be from the same sample.
- `--r2_fastq R2_FASTQ`: the R2 input FASTQ. Can be used multiple times.
- `--readgroups READGROUPS`: readgroup information for each FASTQ. The pipeline will expect the same number of arguments to `--r1_fastq` and `--readgroups` arguments.

  An example argument is, `--readgroups "@RG\tID:HG002-1\tSM:HG002\tLB:HG002-LB-1\tPL:ILLUMINA"`
- `-m MODEL_BUNDLE`: the location of the model bundle. Model bundle files can be found in the sentieon-models[17] repository.
- `SAMPLE_VCF`: the location of the output VCF file for SNVs and indels. The pipeline requires the output file end with the suffix, `.vcf.gz`. The file path without the suffix will be used as the basename for other output files.

[16] https://github.com/lh3/bwa/blob/master/README-alt.md
[17] https://github.com/Sentieon/sentieon-models

The DNAscope pipeline accepts the following optional arguments:

- `-d DBSNP`: the location of the Single Nucleotide Polymorphism database (dbSNP) used to label known variants in VCF (`.vcf`) or bgzip compressed VCF (`.vcf.gz`) format. Only one file is supported. Supplying this file will annotate variants with their dbSNP refSNP ID numbers. A VCF index file is required.
- `-b INTERVAL_FILE`: interval in the reference to restrict variant calling, in BED file format. Supplying this file will limit variant calling to the intervals inside the BED file. If a BED file is not supplied, the software will process the whole genome.
- `--interval_padding INTERVAL_PADDING`: adds `INTERVAL_PADDING` bases padding to the edges of the input intervals. The default value is 0.
- `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The argument is optional; if omitted, the pipeline will use as many threads as the server has.
- `--pcr_free`: Call variants using `--pcr_indel_model NONE`, which is appropriate for libraries prepared with a PCR-free library prep. Deduplication is still performed to identify optical duplicates.
- `-g`: output variants in the gVCF format, in addition to the VCF output file. The tool will output a bgzip compressed gVCF file with a corresponding index file.
- `--duplicate_marking DUP_MARKING`: setting for duplicate marking. `markdup` will mark duplicate reads. `rmdup` will remove duplicate reads. `none` will skip duplicate marking. The default setting is `markdup`.
- `--assay ASSAY`: assay setting for metrics collection `WGS` or `WES`. The default setting is `WGS`.
- `--consensus`: generate consensus reads during duplicate marking.
- `-h`: print the command-line help and exit.
- `--dry_run`: print the pipeline commands, but do not actually execute them.
- `--bam_format`: use BAM format instead of CRAM for output aligned files.

9.1.2.2 Alignment and variant calling from uBAM or uCRAM

A single command is run to align, preprocess, and call SNVs, indels, and structural variants from uBAM or uCRAM files:

```
sentieon-cli dnascope [-h] \
  -r REFERENCE \
  -i SAMPLE_INPUT ... \
  --align \
  [--input_ref INPUT_REF] \
  -m MODEL_BUNDLE \
  [-d DBSNP] \
  [-b BED] \
  [--interval_padding INTERVAL_PADDING] \
  [-t NUMBER_THREADS] \
  [--pcr_free] \
  [-g] \
  [--duplicate_marking DUP_MARKING] \
  [--assay ASSAY] \
  [--consensus] \
  [--dry_run] \
  [--bam_format] \
  SAMPLE_VCF
```

With uBAM or uCRAM input, the DNAscope pipeline requires the following new arguments:

- `-i SAMPLE_INPUT`: the input input sample file in uBAM or uCRAM format. One or more files can be supplied by passing multiple files after the `-i` argument.
- `--align`: directs the pipeline to align the input reads.

The DNAscope pipeline accepts the following new optional arguments:

- `--input_ref INPUT_REF`: a reference fasta used for decoding the input file(s). Required with uCRAM input. Can be different from the fasta used with the `-r` argument.

9.1.2.3 Alignment and variant calling from coordinate-sorted BAM or CRAM

A single command is run to align, preprocess, and call SNVs, indels, and structural variants from BAM or CRAM files:

```
sentieon-cli dnascope [-h] \
  -r REFERENCE \
  -i SAMPLE_INPUT ... \
  --collate_align \
  [--input_ref INPUT_REF] \
  -m MODEL_BUNDLE \
  [-d DBSNP] \
  [-b BED] \
  [--interval_padding INTERVAL_PADDING] \
  [-t NUMBER_THREADS] \
  [--pcr_free] \
  [-g] \
  [--duplicate_marking DUP_MARKING] \
  [--assay ASSAY] \
  [--consensus] \
  [--dry_run] \
  [--bam_format] \
  SAMPLE_VCF
```

With BAM or CRAM input, the DNAscope pipeline requires the following new arguments:

- `--collate_align`: directs the pipeline to collate and then align the input reads.

9.1.2.4 Variant calling from sorted BAM or CRAM

A single command is run to preprocess, and call SNVs, indels, and structural variants from BAM or CRAM files:

```
sentieon-cli dnascope [-h] \
  -r REFERENCE \
  -i SAMPLE_INPUT ... \
  -m MODEL_BUNDLE \
  [-d DBSNP] \
  [-b BED] \
  [--interval_padding INTERVAL_PADDING] \
  [-t NUMBER_THREADS] \
  [--pcr_free] \
  [-g] \
  [--duplicate_marking DUP_MARKING] \
  [--assay ASSAY] \
```

(continues on next page)

(continued from previous page)

```
[--consensus] \
[--dry_run] \
[--bam_format] \
SAMPLE_VCF
```

Not supplying the `--align` and `--collate_align` arguments will direct the pipeline to call variants directly from the input reads.

9.1.3 Pipeline output

9.1.3.1 List of output files

The following files are output when processing WGS FASTQ with default arguments if the output file is `sample.vcf.gz`:

- `sample.vcf.gz`: SNV and indel variant calls across the regions of the genome as defined in the `-b BED` file.
- `sample_deduped.cram` or `sample_deduped.bam`: aligned, coordinate-sorted and duplicate-marked read data from the input FASTQ files.
- `sample_svs.vcf.gz`: structural variant calls from DNAscope and SVSolver.
- `sample_metrics`: a directory containing QC metrics for the analyzed sample.
- `sample_metrics/coverage*`: coverage metrics for the processed sample. Only available for WGS samples.
- `sample_metrics/{sample}.txt.alignment_stat.txt`: Metrics from the AlignmentStat algo.
- `sample_metrics/{sample}.txt.base_distribution_by_cycle.txt`: Metrics from the BaseDistributionByCycle algo.
- `sample_metrics/{sample}.txt.dedup_metrics.txt`: Metrics from the Dedup algo.
- `sample_metrics/{sample}.txt.gc_bias*`: Metrics from the GCBias algo. Only available for WGS samples.
- `sample_metrics/{sample}.txt.insert_size.txt`: Metrics from the InsertSizeMetricAlgo algo.
- `sample_metrics/{sample}.txt.mean_qual_by_cycle.txt`: Metrics from the MeanQualityByCycle algo.
- `sample_metrics/{sample}.txt.qual_distribution.txt`: Metrics from the QualDistribution algo.
- `sample_metrics/{sample}.txt.wgs.txt`: Metrics from the WgsMetricsAlgo algo. Only available for WGS samples.
- `sample_metrics/{sample}.txt.hybrid-selection.txt`: Metrics from the HsMetricAlgo algo.
- `sample_metrics/multiqc_report.html`: collected QC metrics aggregated by MultiQC.

9.2 DNAscope LongRead

Sentieon® DNAscope LongRead is a pipeline for alignment and germline variant calling (SNVs, SVs, CNVs, and indels) from long-read sequence data. The DNAscope LongRead pipeline is able to take advantage of longer read lengths to perform quick and accurate variant calling using specially calibrated machine learning models. The sentieon-cli provides a complete implementation of the DNAscope LongRead pipeline in a single command. The pipeline is designed for accuracy, efficiency, and ease-of-use.

The pipeline will accept as input aligned reads in BAM or CRAM format, or un-aligned reads in FASTQ, uBAM, or uCRAM format. The pipeline will output variants in the VCF (or gVCF) formats and aligned reads in BAM or CRAM formats.

9.2.1 Setup

9.2.1.1 Prerequisites

- Sentieon® software package version 202308.01 or higher.
- Python[18] version 3.8 or higher.
- bcftools[19] version 1.10 or higher for the SNV and indel calling pipeline.
- bedtools[20] for the SNV and indel calling pipeline.
- samtools[21] version 1.16 or higher for alignment of read data in uBAM or uCRAM format or re-alignment of previously aligned reads.
- mosdepth[22] version 0.2.6 or higher for coverage metrics of long-read data.
- hificnv[23] version 1.0.0 or higher for CNV calling.

The `sentieon`, `python`, `bcftools`, `bedtools`, `samtools`, `hificnv`, and `mosdepth` executables will be accessed through the user's `PATH` environment variable.

9.2.1.2 The Reference genome

DNAscope LongRead will call variants present in the sample relative to a high quality reference genome sequence. Besides the reference genome file, a samtools fasta index file (.fai) needs to be present.

We recommend aligning to a reference genome without alternate contigs.

9.2.2 Usage

9.2.2.1 Alignment and variant calling from FASTQ

A single command is run to align and call SNVs, indels and structural variants from PacBio HiFi or ONT reads in the FASTQ format:

```
sentieon-cli dnascope-longread [-h] \
  -r REFERENCE \
  --fastq INPUT_FASTQ ... \
  --readgroups READGROUP ... \
  -m MODEL_BUNDLE \
  [-d DBSNP] \
  [-b DIPLOID_BED] \
  [-t NUMBER_THREADS] \
  [-g] \
  --tech HiFi|ONT \
  [--haploid_bed HAPLOID_BED] \
  [--cnv_excluded_regions CNV_EXCLUDE_BED] \
  SAMPLE_VCF
```

[18] https://www.python.org/
[19] http://samtools.github.io/bcftools/bcftools.html
[20] https://bedtools.readthedocs.io/en/latest/
[21] https://www.htslib.org/
[22] https://github.com/brentp/mosdepth
[23] https://github.com/PacificBiosciences/HiFiCNV

With FASTQ input, the DNAscope LongRead pipeline requires the following arguments:

- `-r REFERENCE`: the location of the reference FASTA file. A reference fasta index, ".fai" file, is also required.
- `--fastq INPUT_FASTQ`: the input sample file in FASTQ format. One or more files can be supplied by passing multiple files after the `--fastq` argument.
- `--readgroups READGROUP`: readgroup information for the read data. The `--readgroups` argument is required if the input data is in the FASTQ format. This argument expects complete readgroup strings and these strings will be passed to `minimap2` through the `-R` argument.

  An example argument is, `--readgroups '@RG\tID:foo\tSM:bar'`.
- `-m MODEL_BUNDLE`: the location of the model bundle. Model bundle files can be found in the sentieon-models[24] repository.
- `--tech HiFi|ONT`: Sequencing technology used to generate the reads. Supported arguments are `ONT` or `HiFi`.
- `SAMPLE_VCF`: the location of the output VCF file for SNVs and indels. The pipeline requires the output file end with the suffix, ".vcf.gz". The file path without the suffix will be used as the basename for other output files.

The Sentieon® LongRead pipeline accepts the following optional arguments:

- `-d DBSNP`: the location of the Single Nucleotide Polymorphism database (dbSNP) used to label known variants in VCF (`.vcf`) or bgzip compressed VCF (`.vcf.gz`) format. Only one file is supported. Supplying this file will annotate variants with their dbSNP refSNP ID numbers. A VCF index file is required.
- `-b DIPLOID_BED`: interval in the reference to restrict diploid variant calling, in BED file format. Supplying this file will limit diploid variant calling to the intervals inside the BED file.
- `--haploid_bed HAPLOID_BED`: interval in the reference to restrict haploid variant calling, in BED file format. Supplying this file will perform haploid variant calling across the intervals inside the BED file.
- `--cnv_excluded_regions`: a BED file of excluded CNV regions passed to hificnv. See the hificnv documentation for more details, https://github.com/PacificBiosciences/HiFiCNV.
- `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The argument is optional; if omitted, the pipeline will use as many threads as the server has.
- `-g`: output variants in the gVCF format, in addition to the VCF output file. The tool will output a bgzip compressed gVCF file with a corresponding index file.
- `-h`: print the command-line help and exit.
- `--dry_run`: print the pipeline commands, but do not actually execute them.

9.2.2.2 Alignment and variant calling from uBAM, uCRAM, BAM, or CRAM

A single command is run to align and call SNVs, indels, and structural variants from PacBio HiFi or ONT reads in the uBAM, uCRAM, BAM, or CRAM formats:

```
sentieon-cli dnascope-longread [-h] \
  -r REFERENCE \
  -i SAMPLE_INPUT ... \
  --align \
  -m MODEL_BUNDLE \
```

(continues on next page)

[24] https://github.com/Sentieon/sentieon-models

(continued from previous page)

```
  [-d DBSNP] \
  [-b DIPLOID_BED] \
  [-t NUMBER_THREADS] \
  [-g] \
  --tech HiFi|ONT \
  [--haploid_bed HAPLOID_BED] \
  [--cnv_excluded_regions CNV_EXCLUDE_BED] \
  [--input_ref INPUT_REF] \
  SAMPLE_VCF
```

With uBAM, uCRAM, BAM, or CRAM input, the DNAscope LongRead pipeline requires the following new arguments:

- `-i SAMPLE_INPUT`: the input sample file in uBAM or uCRAM format. One or more files can be supplied by passing multiple files after the `-i` argument.
- `--align`: re-align the input read data to the reference genome using Sentieon® minimap2.

The DNAscope LongRead pipeline accepts the following new optional arguments:

- `--input_ref INPUT_REF`: a reference fasta used for decoding the input file(s). Required with uCRAM or CRAM input. Can be different from the fasta used with the `-r` argument.

9.2.2.3 Variant calling from BAM or CRAM

A single command is run to call SNVs, indels, and structural variants from PacBio HiFi or ONT reads in the BAM, or CRAM formats:

```
sentieon-cli dnascope-longread [-h] \
  -r REFERENCE \
  -i SAMPLE_INPUT ... \
  -m MODEL_BUNDLE \
  [-d DBSNP] \
  [-b DIPLOID_BED] \
  [-t NUMBER_THREADS] \
  [-g] \
  --tech HiFi|ONT \
  [--haploid_bed HAPLOID_BED] \
  [--cnv_excluded_regions CNV_EXCLUDE_BED] \
  SAMPLE_VCF
```

Not supplying the `--align` argument will direct the pipeline to call variants directly from the input reads.

9.2.3 Pipeline output

9.2.3.1 List of output files

The following files are output when processing FASTQ data or uBAM, uCRAM, BAM, or CRAM files with the `--align` argument:

- `sample.vcf.gz`: SNV and indel variant calls across the regions of the genome as defined in the `-b DIPLOID_BED` file.
- `sample.sv.vcf.gz`: structural variant calls from the Sentieon® LongReadSV tool.
- `sample_mm2_sorted_fq_*.cram`: aligned and coordinate-sorted reads from the input FASTQ files.

- `sample_mm2_sorted_*.cram`: aligned and coordinate-sorted reads from the input uBAM, uCRAM, BAM, or CRAM files.
- `sample.hificnv`: the base name of HiFiCNV output files (with HiFi input).

9.2.4 Other considerations

9.2.4.1 Diploid and haploid variant calling

The default pipeline is recommended for use with samples from diploid organisms. For samples with both diploid and haploid chromosomes, the `-b DIPLOID_BED` option can be used to limit diploid variant calling to diploid chromosomes and the `--haploid_bed HAPLOID_BED` argument can be used to perform haploid variant calling across haploid chromosomes.

Diploid and haploid BED files for the human hg38 reference genome (with male samples) can be found in the `/data`[25] folder in this repository.

9.2.4.2 Modification

Scripts in this repository are made available under the BSD 2-Clause license[26].

The Python scripts in the `sentieon_cli/scripts` folder perform low-level manipulation of intermediate gVCF and VCF files generated by the pipeline. Due to the low-level data handling performed by these scripts, modification of these files by users is discouraged.

9.2.5 References

Sentieon DNAscope LongRead – A highly Accurate, Fast, and Efficient Pipeline for Germline Variant Calling from PacBio HiFi reads[27] - A preprint describing the DNAscope LongRead pipeline for calling variants from PacBio HiFi data.

9.3 DNAscope Hybrid

Sentieon® DNAscope Hybrid is a pipeline for germline variant calling using combined short-read and long-read data from a single sample. The DNAscope Hybrid pipeline is able to utilize the strengths of both short and long-read technologies to generate variant callsets that are more accurate than either short-read or long-read data alone. The sentieon-cli provides a complete implementation of the DNAscope hybrid pipeline in a single command.

The pipeline supports input data in the following formats; both short-read and long-read input are required:

- Unaligned short-read data in gzipped FASTQ format.
- Aligned short-reads in BAM or CRAM format.
- Unaliged long-read data in the uBAM or uCRAM format.
- Aligned long-read data in BAM or CRAM format.

By default, the pipeline will generate the following output files:

- Small variants (SNVs and indels) in the VCF format.
- Structural variants in the VCF format.
- Copy-number variants in the VCF format.

If unaligned reads are used as input, the pipeline will also output aligned reads in BAM or CRAM format.

[25] https://github.com/Sentieon/sentieon-cli/tree/main/data
[26] https://github.com/Sentieon/sentieon-cli/blob/main/LICENSE
[27] https://www.biorxiv.org/content/10.1101/2022.06.01.494452v1

9.3.1 Setup

9.3.1.1 Prerequisites

- Sentieon® software package version 202503.01 or higher.
- Python[28] version 3.8 or higher.
- bcftools[29] version 1.10 or higher.
- bedtools[30]
- MultiQC[31] version 1.18 or higher for metrics report generation.
- samtools[32] version 1.16 or higher.
- mosdepth[33] version 0.2.6 or higher for coverage metrics collection from long-read data.

The `sentieon`, `python`, `bcftools`, `bedtools`, `samtools`, `multiqc`, and `mosdepth` executables will be accessed through the user's `PATH` environment variable.

9.3.1.2 The Reference genome

DNAscope LongRead will call variants present in the sample relative to a high quality reference genome in FASTA format. Besides the reference genome file, a samtools fasta index file (.fai) needs to be present. Short-read alignment also requires bwa index files.

We recommend aligning to a reference genome without alternate contigs. If alternate contigs are present in the genome and the pipeline is performing short-read alignment, please also supply a ".alt" file to activate alt-aware alignment[34] in bwa.

9.3.2 Usage

9.3.2.1 Germline variant calling from aligned short and long-read data

A single command is run to call SNVs, indels, SVs and CNVs from aligned short and long reads:

```
sentieon-cli dnascope-hybrid \
  -r REFERENCE \
  --sr_aln SR_ALN [SR_ALN ...] \
  --lr_aln LR_ALN [LR_ALN ...] \
  -m MODEL_BUNDLE \
  [-d DBSNP] \
  [-b DIPLOID_BED] \
  [-t NUMBER_THREADS] \
  sample.vcf.gz
```

The DNAscope Hybrid pipeline requires the following arguments:

- `-r REFERENCE`: the location of the reference FASTA file. A reference fasta index, ".fai" file, is also required.
- `--sr_aln`: the input short-read data in BAM or CRAM format. One or more files can be supplied by passing multiple files after the `--sr_aln` argument.

[28] https://www.python.org/
[29] http://samtools.github.io/bcftools/bcftools.html
[30] https://bedtools.readthedocs.io/en/latest/
[31] https://multiqc.info/
[32] https://www.htslib.org/
[33] https://github.com/brentp/mosdepth
[34] https://github.com/lh3/bwa/blob/master/README-alt.md

- `--lr_aln`: the input long-read data in BAM or CRAM format. One or more files can be supplied by passing multiple files after the `--lr_aln` argument.
- `-m MODEL_BUNDLE`: the location of the model bundle. Model bundle files can be found in the sentieon-models[35] repository.
- `sample.vcf.gz`: the location of the output VCF file for SNVs and indels. The pipeline requires the output file end with the suffix, ".vcf.gz".

The DNAscope Hybrid pipeline accepts the following optional arguments:

- `-d DBSNP`: the location of the Single Nucleotide Polymorphism database (dbSNP) used to label known variants in VCF (`.vcf`) or bgzip compressed VCF (`.vcf.gz`) format. Only one file is supported. Supplying this file will annotate variants with their dbSNP refSNP ID numbers. A VCF index file is required.
- `-b DIPLOID_BED`: interval in the reference to restrict diploid variant calling, in BED file format. Supplying this file will limit diploid variant calling to the intervals inside the BED file.
- `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The argument is optional; if omitted, the pipeline will use as many threads as the server has.
- `-h`: print the command-line help and exit.
- `--dry_run`: print the pipeline commands, but do not actually execute them.

9.3.2.2 Germline variant calling from unaligned short and long-read data

A single command is run to call SNVs, indels, SVs and CNVs from unaligned short and long reads:

```
sentieon-cli dnascope-hybrid \
  -r REFERENCE \
  --sr_r1_fastq SR_R1_FQ [SR_R1_FQ ...] \
  --sr_r2_fastq SR_R2_FQ [SR_R2_FQ ...] \
  --sr_readgroups SR_READGROUP [SR_READGROUP ...] \
  --lr_aln LR_ALN [LR_ALN ...] \
  --lr_align_input \
  -m MODEL_BUNDLE \
  [-d DBSNP] \
  [-b DIPLOID_BED] \
  [-t NUMBER_THREADS] \
  sample.vcf.gz
```

The DNAscope Hybrid pipeline requires the following arguments:

- `--sr_r1_fastq`: the input R1 short-read data in gzipped FASTQ format. One or more files can be supplied by passing multiple files after the `--sr_r1_fastq` argument.
- `--sr_r2_fastq`: the input R2 short-read data in gzipped FASTQ format. One or more files can be supplied by passing multiple files after the `--sr_r2_fastq` argument.
- `--sr_readgroups`: readgroup information for each FASTQ. The pipeline will expect the same number of arguments to `--sr_r1_fastq` and `--sr_readgroups`. An example argument is, `--sr_readgroups "@RG\tID:HG002-1\tSM:HG002\tLB:HG002-LB-1\tPL:ILLUMINA"`
- `--lr_aln`: the input long-read data in uBAM or uCRAM format. One or more files can be supplied by passing multiple files after the `--lr_aln` argument. - `--lr_align_input`: directs the pipeline to align the input long-reads.

[35] https://github.com/Sentieon/sentieon-models

The DNAscope Hybrid pipeline accepts the following optional arguments:

- `--sr_duplicate_marking`: setting for duplicate marking. `markdup` will mark duplicate reads. `rmdup` will remove duplicate reads. `none` will skip duplicate marking. The default setting is `markdup`.
- `--lr_input_ref`: a reference fasta used for decoding the input long-read file(s). Required with long-read uCRAM or CRAM input. Can be different from the fasta used with the `-r` argument.
- `--bam_format`: use BAM format instead of CRAM for output aligned files.

9.3.3 Pipeline output

9.3.3.1 List of output files

The following files are output by the DNAscope Hybrid pipeline:

- `sample.vcf.gz`: SNV and indel variant calls across the regions of the genome as defined in the `-b DIPLOID_BED` file.
- `sample.sv.vcf.gz`: structural variant calls from the Sentieon® LongReadSV tool.
- `sample.cnv.vcf.gz`: copy-number variant calls from the Sentieon® CNVscope tool.
- **`sample_deduped.cram`: aligned, coordinate-sorted and duplicate-marked** short-read data from the input FASTQ files.
- **`sample_mm2_sorted_*.cram`: aligned and coordinate-sorted long-reads** from the input uBAM, uCRAM, BAM, or CRAM files.
- **`sample_metrics`: a directory containing QC metrics for the analyzed** sample.

9.3.4 Troubleshooting

9.3.4.1 The pipeline complains, `Input . . . has a different RG-SM tag`

This error will occur if the pipeline detects that the input files have (or will have) different readgroup `SM` tags. To fix this error, please use the `--rgsm` argument to adjust the `SM` tags of the input files during variant calling. Note that with this argument, all reads in the input files will be used during variant calling.

9.4 Sentieon Pangenome

Sentieon® Pangenome is a pipeline for alignment and variant calling from short-read DNA sequence data using pangenome graphs. The Pangenome pipeline leverages graph-based reference representations to improve alignment and variant calling accuracy, particularly in complex genomic regions with high sequence diversity.

The pipeline accepts as input un-aligned reads in FASTQ format or aligned reads in BAM or CRAM format. The pipeline will output variants in the VCF format and aligned reads in BAM or CRAM formats.

9.4.1 Setup

9.4.1.1 Prerequisites

- Sentieon® software package version 202503.02 or higher.
- samtools[36] version 1.16 or higher.
- vg[37] for personalized pangenome creation and pangenome manipulation.

[36] https://www.htslib.org/
[37] https://github.com/vgteam/vg

- KMC[38] version 3 or higher for k-mer counting.
- bcftools[39] version 1.22 or higher for VCF operations.
- MultiQC[40]. `MultiQC` is used to generate a report from the pipeline metrics.

The executables will be accessed through the user's `PATH` environment variable.

9.4.1.2 The Reference genome

The Sentieon® Pangenome will call variants relative to a high quality reference genome sequence. Besides the reference genome file, a samtools fasta index file (.fai) needs to be present. Read alignment also requires bwa index files.

We recommend using a reference genome without alternate contigs.

Currently, only GRCh38 with UCSC-style contig names (`chr1`, `chr2`, etc.) is supported.

9.4.1.3 Pangenome files

The pipeline requires the following pangenome files:

- `GBZ file`: The pangenome graph in GBZ format.
- `Haplotype file`: Pangenome haplotype information.

The pangenome needs to incorporate GRCh38 as a reference sequence.

The sentieon-cli will check that the `.gbz` and `.hapl` files follow the HPRC naming convention, ending with `grch38.gbz` and `grch38.hapl`, respectively. This check can be disabled using the hidden argument, `--skip_pangenome_name_checks`.

9.4.1.4 Model bundle

A Sentieon model bundle containing machine learning models for variant calling is required. Model bundle files can be found in the sentieon-models[41] repository.

9.4.1.5 Population VCF

The pipeline requires a population VCF file as input. The population VCF must match the model bundle that is used by the pipeline.

9.4.1.6 Optional input files

Additional input files are required for optional functionality:

- A BED file containing variant calling intervals. Recommended for whole-genome sequencing data to restrict variant calling to the canonical contigs. Recommended for whole-exome sequencing data to restrict variant calling to target regions.
- The Single Nucleotide Polymorphism database (dbSNP) data that you want to include in the pipeline. The data is used in the form of a VCF file; you can use a VCF file compressed with bgzip and indexed.

[38] https://github.com/refresh-bio/KMC
[39] http://samtools.github.io/bcftools/bcftools.html
[40] https://github.com/MultiQC/MultiQC
[41] https://github.com/Sentieon/sentieon-models

9.4.2 Usage

9.4.3 Alignment and variant calling from FASTQ

A single command is run to perform alignment, preprocessing and metrics collection, and SNV and indel calling from FASTQ files:

```
sentieon-cli sentieon-pangenome [-h] \
  -r REFERENCE \
  --hapl HAPL \
  --gbz GBZ \
  --r1_fastq R1_FASTQ ... \
  --r2_fastq R2_FASTQ ... \
  --readgroups READGROUP \
  --pop_vcf POP_VCF \
  -m MODEL_BUNDLE \
  [-b INTERVAL_FILE] \
  [--bam_format] \
  [-d DBSNP] \
  [--dry_run] \
  [-t NUMBER_THREADS] \
  [--pcr_free] \
  SAMPLE_VCF
```

With FASTQ input, the Sentieon® pangenome pipeline requires the following arguments:

- `-r REFERENCE`: the location of the reference FASTA file. A reference fasta index, ".fai" file, and bwa index files, are also required.
- `--hapl HAPL`: the location of the pangenome haplotype file.
- `--gbz GBZ`: the location of the pangenome graph file in GBZ format.
- `--r1_fastq R1_FASTQ`: the R1 input FASTQ. Can be used multiple times. Each `R1_FASTQ` must have a corresponding `R2_FASTQ` file. All fastq are expected to be from the same sample.
- `--r2_fastq R2_FASTQ`: the R2 input FASTQ. Can be used multiple times.
- `--readgroup READGROUP`: readgroup information for the sample. A single readgroup can be supplied for the sample. An example argument is, `--readgroup "@RG\tID:HG002-1\tSM:HG002\tLB:HG002-LB-1\tPL:ILLUMINA"`
- `--pop_vcf POP_VCF`: the location of the population VCF file.
- `-m MODEL_BUNDLE`: the location of the model bundle. Model bundle files can be found in the sentieon-models[42] repository.
- `SAMPLE_VCF`: the location of the output VCF file for SNVs and indels. The pipeline requires the output file end with the suffix, `.vcf.gz`. The file path without the suffix will be used as the basename for other output files.

The Sentieon® pangenome pipeline accepts the following optional arguments:

- `-b INTERVAL_FILE`: interval in the reference to restrict variant calling, in BED file format. Supplying this file will limit variant calling to the intervals inside the INTERVAL_FILE. We recommend using an `INTERVAL_FILE` to restrict variant calling to canonical contigs.
- `--bam_format`: use BAM format instead of CRAM for output aligned files.

[42] https://github.com/Sentieon/sentieon-models

- `-d DBSNP`: the location of the Single Nucleotide Polymorphism database (dbSNP) used to label known variants in VCF (`.vcf`) or bgzip compressed VCF (`.vcf.gz`) format. Only one file is supported. Supplying this file will annotate variants with their dbSNP refSNP ID numbers. A VCF index file is required.
- `--dry_run`: print the pipeline commands, but do not actually execute them.
- `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The argument is optional; if omitted, the pipeline will use as many threads as the server has.
- `--pcr_free`: Call variants using `--pcr_indel_model NONE`, which is appropriate for libraries prepared with a PCR-free library prep. Deduplication is still performed to identify optical duplicates.
- `-h`: print the command-line help and exit.

9.4.4 Variant calling from sorted BAM or CRAM

A single command is run to perform alignment, preprocessing and metrics collection, and SNV and indel calling from BAM or CRAM files:

```
sentieon-cli sentieon-pangenome [-h] \
  -r REFERENCE \
  --hapl HAPL \
  --gbz GBZ \
  -i SAMPLE_INPUT ... \
  --pop_vcf POP_VCF \
  -m MODEL_BUNDLE \
  [-b INTERVAL_FILE] \
  [-d DBSNP] \
  [--dry_run] \
  [-t NUMBER_THREADS] \
  SAMPLE_VCF
```

With BAM or CRAM input, the pangenome pipeline requires the following new arguments:

- `-i SAMPLE_INPUT`: the input sample file in uBAM or uCRAM format. One or more files can be supplied by passing multiple files after the `-i` argument.

9.4.5 Pipeline output

9.4.5.1 List of output files

The following files are output when processing WGS FASTQ with all features if the output file is `sample.vcf.gz`:

- `sample.vcf.gz`: SNV and indel variant calls across the regions of the genome as defined in the `-b BED` file.
- `sample_bwa_deduped.cram` or `sample_bwa_deduped.bam`: bwa aligned, coordinate-sorted and duplicate-marked read data from the input FASTQ file(s).
- `sample_mm2_deduped.cram` or `sample_mm2_deduped.bam`: pangenome aligned, coordinate-sorted and duplicate-marked read data from the input FASTQ files. The reads in this file are aligned to the pangenome and lifted back to the reference genome.
- `sample_metrics/`: A directory containing QC metrics for the analyzed sample.
  - `sample_metrics/sample.txt.alignment_stat.txt`: Metrics from the Sentieon® AlignmentStat algo.

- `sample_metrics/sample.txt.base_distribution_by_cycle.txt`: Metrics from the Sentieon® BaseDistributionByCycle algo.
- `sample_metrics/coverage*`: Coverage metrics from the Sentieon® CoverageMetrics algo.
- `sample_metrics/sample.txt.gc_bias*`: Metrics from the Sentieon® GCBias algo.
- `sample_metrics/sample.txt.insert_size.txt`: Metrics from the Sentieon® InsertSizeMetricAlgo algo.
- `sample_metrics/sample.txt.mean_qual_by_cycle.txt`: Metrics from the Sentieon® MeanQualityByCycle algo.
- `sample_metrics/sample.txt.qual_distribution.txt`: Metrics from the Sentieon® QualDistribution algo.
- `sample_metrics/sample.txt.wgs.txt`: Metrics from the Sentieon® WgsMetricsAlgo algo.
- `sample_metrics/multiqc_report.html`: Collected QC metrics aggregated by MultiQC.

9.4.6 Limitations of the Sentieon® pangenome pipeline

The Sentieon® pangenome pipeline currently only supports Minigraph-Cactus pangenomes with a GRCh38 reference sequence, such as those generated by the Human Pangenome Reference Consortium (HPRC). Please reach out to Sentieon® support for information on using the pipeline with other pangenomes.

10 Deduplication and UMI Handling

10.1 Introduction

This document describes how to remove PCR duplicates using Sentieon® Genomics tool. This step uses two individual commands to collect read information and perform the deduplication. The option `--consensus` of `LocusCollector` controls whether to output the consensus of PCR duplicates. If the unique molecular identifiers (UMI) tag is applicable, use the option `--umi_tag` for `LocusCollector` to turn on the barcode-aware deduplication.

10.2 Non-consensus-based deduplication

With Non-consensus-based deduplication, a representative read from a group of duplicate reads is selected as the primary read.

10.2.1 Non-consensus-based deduplication without UMI

This workflow matches the default outcome of Picard `MarkDuplicates`.

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --fun score_info SCORE.gz
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo Dedup [--rmdup] --score_info SCORE.gz  \
  --metrics DEDUP_METRIC_TXT DEDUPED_BAM
```

There is a special 3-pass deduplication flow to mark both primary and non-primary reads. This workflow, however, is only available for non-consensus-based deduplication without UMI.

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --fun score_info SCORE.gz
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo Dedup --score_info SCORE.gz --output_dup_read_name \
  --metrics DEDUP_METRIC_TXT TMP_DUP_QNAME.gz
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo Dedup --dup_read_name TMP_DUP_QNAME.gz \
  DEDUPED_BAM
```

10.2.2 Non-consensus-based deduplication with UMI

This workflow utilizes the UMI information in addition to the 5' positions of both reads and read-pairs to determine PCR duplicates. Use the option `--umi_tag` in `LocusCollector` to specify the logic UMI tag.

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --umi_tag XR --fun score_info SCORE.gz
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo Dedup [--rmdup] --score_info SCORE.gz  \
  --metrics DEDUP_METRIC_TXT DEDUPED_BAM
```

10.3 Consensus-based deduplication

With consensus-based deduplication, one single consensus read is generated from a group of duplicate reads. This process will correct errors introduced by PCR and sequencing. It will also estimate base quality scores at each position to reflect the new probability that a base in the consensus read is called incorrectly. Additional base quality adjustments should not be performed after consensus-based deduplication.

Set the option `--consensus` in `LocusCollector` to turn on the consensus-based deduplication function. Moreover, the reference FASTA file is now required for `Dedup`.

Note

The mate information of consensus reads may be incorrect after consensus-based deduplication. Mate information is not used by Sentieon® variant callers, but may be used by other bioinformatics software.

If correct mate information is required, `samtools fixmate` can be used to update the mate information of reads after consensus-based deduplication.

```
samtools collate -@ NUMBER_THREADS -Ou DEDUPED_BAM \
  | samtools fixmate --reference REFERENCE -@ NUMBER_THREADS - - \
  | sentieon util sort -r REFERENCE -o FIXMATE_BAM -t NUMBER_THREADS -i -
```

10.3.1 Consensus-based deduplication without UMI

Without UMI, this workflow uses the alignment coordinates alone to cluster sequencing reads.

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --consensus --fun score_info SCORE.gz
sentieon driver -t NUMBER_THREADS -r REFERENCE -i SORTED_BAM \
  --algo Dedup [--rmdup] --score_info SCORE.gz  \
  --metrics DEDUP_METRIC_TXT DEDUPED_BAM
```

10.3.2 Consensus-based deduplication with UMI

With UMI, this workflow uses both the alignment coordinates and their UMIs to cluster sequencing reads.

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --consensus --umi_tag XR --fun score_info SCORE.gz
sentieon driver -t NUMBER_THREADS -r REFERENCE -i SORTED_BAM \
  --algo Dedup [--rmdup] --score_info SCORE.gz  \
  --metrics DEDUP_METRIC_TXT DEDUPED_BAM
```

10.3.2.1 UMI barcode error correction

Errors in the UMI barcodes are corrected automatically based on the edit distance with other barcodes. To disable this function, use the option `--umi_ecc_dist 0` in `LocusCollector`.

10.3.2.2 RNA sequence data

Set the option `--rna` in `LocusCollector` when using RNA sequence data aligned with STAR.

```
sentieon driver -t NUMBER_THREADS -i SORTED_BAM \
  --algo LocusCollector --rna [--consensus] [--umi_tag XR] --fun score_info SCORE.gz
sentieon driver -t NUMBER_THREADS -r REFERENCE -i SORTED_BAM \
  --algo Dedup [--rmdup] --score_info SCORE.gz DEDUPED_BAM
```

10.4 Appendix

10.4.1 Determine the read structure and extract UMI barcode sequences

As a first step, you need to extract the barcode sequences from the input reads. This is performed with the Sentieon® `umi extract` command, which extracts the barcode sequence information from the reads and adds it to the read description. The adapter sequence should be removed from the input reads before running umi tag extraction. This can be performed by other third-party tools.

The output of `umi extract` is in FASTQ format with interleaved R1 and R2 reads. By default, the output from the extract command will be sent to standard output unless otherwise defined with the option `-o`.

The syntax of the `umi extract` command is as follows:

```
sentieon umi extract [options] read_structure fastq1 [fastq2] [fastq3]

Options:
  -o      Output file (default: stdout)
  -d      Turn on duplex mode
  --umi_tag     Logic umi tag (default 'XR')
  --output_format       Output format FASTQ or SAM (default 'FASTQ')
```

The first argument of the `umi extract` command defines the read structure. For paired-end reads, two read structures separated by comma ',' should be specified.

A read structure is defined by <number><operator> pairs. The number can be any digit or '+' to indicate the end of the read. Possible operators include:

- **T** The template sequence.
- **B** The cell barcode sequence.
- **M** The molecular barcode sequence.
- **S** A sequence of bases that should be ignored.

The `-d` option is used to extract duplex UMI and label the strand from which it originates. Duplex UMI extraction requires an identical read structure for both strands.

As an example, the following command demonstrates a single-ended UMI extraction on pair-ended reads. In this case, the first read in the pair contains an 8bp molecular barcode followed by a 12bp spacer and then the template sequence. The second read contains only the template sequence. The paired reads will be interleaved in the output file. Please note that in this example the output is piped to `gzip` to generate a compressed FASTQ file. In general, we recommend piping the output directly to the next step (Sentieon® `bwa mem`).

```
sentieon umi extract "8M12S+T,+T" \
  sample_R1.fastq.gz \
  sample_R2.fastq.gz | \
  gzip -c \
  > sample_extracted_pair.fastq.gz
```

The command below demonstrates duplex UMI extraction where both reads contain 4bp molecular barcode followed by template sequence.

```
sentieon umi extract \
  -d \
  "4M+T,4M+T" \
  sample_R1.fastq.gz \
  sample_R2.fastq.gz | \
  gzip -c \
  > sample_extracted_pair.fastq.gz
```

Below is a code example including both UMI extraction and alignment to the reference genome.

```
sentieon umi extract \
  8M12S+T,+T \
  sample_R1.fastq.gz \
  sample_R2.fastq.gz | \
sentieon bwa mem \
  -R "@RG\tID:$GROUP\tSM:$SAMPLE\tLB:$LIBRARY\tPL:$PLATFORM" \
  -t $NT \
  -K $BWA_K_SIZE \
  -p \
  -C \
  $REF \
  - | \
sentieon util sort \
  -i - \
  -o sorted.bam \
  --sam2bam
```

Below is a use case when the UMI sequence is already in a separate FASTQ file `sample_I1.fastq.gz`. When running in this mode, only one additional UMI index read is allowed. The UMI index read should contain no template sequences. This mode does not allow duplex UMI extraction.

```
sentieon umi extract \
  "+M,+T,+T" \
  sample_I1.fastq.gz \
  sample_R1.fastq.gz \
  sample_R2.fastq.gz | \
  gzip -c \
  > sample_extracted_pair.fastq.gz
```

The command below produces a SAM format output. This format is useful when using the RNA-seq aligner `STAR`. In this example, the first read in the pair contains a 16bp cell barcode and a 10bp molecular barcode. The second read contains only the template sequence. The output is single-end reads in the SAM format.

```
sentieon umi extract \
  --output_format SAM \
```

(continues on next page)

(continued from previous page)

```
"16B10M+S,+T" \
sample_R1.fastq.gz \
sample_R2.fastq.gz \
> sample_extracted.sam
```

The output of `umi extract` contains additional tags. By default, the output contains a CR tag for the cell or sample barcode and an XR tag for UMI sequence to be used by `umi consensus`.

Table 10.1: Additional tags generated by `umi extract`

Tags	Meaning
RX	Extracted UMI sequence bases.
XR	UMI tag for grouping in `umi consensus`.
CR	Cell barcode.

11

Somatic Variant Calling for SNPs and Indels

11.1 Introduction

This document describes somatic variant calling pipelines using TNscope® and TNseq® . TNscope® uses an improved variant calling algorithm to obtain higher accuracy and improved runtimes, while TNseq® matches the GATK's Mutect2 somatic variant calling with substantially improved runtime and parallelization.

For complete somatic variant calling pipelines, please visit our github page: https://github.com/Sentieon/sentieon-scripts/tree/master/example_pipelines/somatic.

11.2 Data processing without unique molecular identifiers (UMIs)

11.2.1 Step1. Alignment

```
# ****************************************
# 1a. Mapping reads with BWA-MEM, sorting for the tumor sample
# ****************************************
( sentieon bwa mem -R "@RG\tID:$tumor\tSM:$tumor\tPL:$platform" \
  -t $nt -K 10000000 $fasta $tumor_fastq_1 $tumor_fastq_2 || \
  echo -n 'error' ) | \
  sentieon util sort -o tumor_sorted.bam -t $nt --sam2bam -i -

# ****************************************
# 1b. Mapping reads with BWA-MEM, sorting for the normal sample
# ****************************************
( sentieon bwa mem -R "@RG\tID:$normal\tSM:$normal\tPL:$platform" \
  -t $nt -K 10000000 $fasta $normal_fastq_1 $normal_fastq_2 || \
  echo -n 'error' ) | \
  sentieon util sort -o normal_sorted.bam -t $nt --sam2bam -i -
```

11.2.2 Step2. PCR duplicate removal (skip for targeted amplicon sequencing)

```
# ****************************************
# 2a. Remove duplicate reads from the tumor sample
# ****************************************
sentieon driver -t $nt -i tumor_sorted.bam \
```

(continues on next page)

(continued from previous page)

```
  --algo LocusCollector \
  --fun score_info \
  tumor_score.txt
sentieon driver -t $nt -i tumor_sorted.bam \
  --algo Dedup \
  --score_info tumor_score.txt \
  --metrics tumor_dedup_metrics.txt \
  tumor_deduped.bam

# ****************************************
# 2b. Remove duplicate reads from the normal sample
# ****************************************
sentieon driver -t $nt -i normal_sorted.bam \
  --algo LocusCollector \
  --fun score_info \
  normal_score.txt
sentieon driver -t $nt -i normal_sorted.bam \
  --algo Dedup \
  --score_info normal_score.txt \
  --metrics normal_dedup_metrics.txt \
  normal_deduped.bam
```

11.2.3 Step3. Base quality score recalibration (skip for panels)

```
# ****************************************
# 3a. Base recalibration for the tumor sample
# ****************************************
sentieon driver -r $fasta -t $nt -i tumor_deduped.bam --interval $BED \
  --algo QualCal \
  -k $dbsnp \
  -k $known_Mills_indels \
  -k $known_1000G_indels \
  tumor_recal_data.table

# ****************************************
# 3b. Base recalibration for the normal sample
# ****************************************
sentieon driver -r $fasta -t $nt -i normal_deduped.bam --interval $BED \
  --algo QualCal \
  -k $dbsnp \
  -k $known_Mills_indels \
  -k $known_1000G_indels \
  normal_recal_data.table
```

11.2.4 Step4. Variant calling and filtration

Generation of the final variant callset is divided into two commands for variant calling and filtration. Depending on sample and library types, different options and filters should be used.

Please find the python script :code:`tnscope_filter.py` for filtering at Sentieon's GitHub page at https://github.com/Sentieon/sentieon-scripts.

11.2.4.1 Sample type 1 - whole genome/exome sequencing

This section describes the pipeline for WGS/WES using TNseq®. For more information about TNseq®, please visit *Typical usage for TNseq®*.

```
sentieon driver -r $fasta -t $nt -i tumor_deduped.bam -i normal_deduped.bam \
  [--interval $BED] \
  --algo TNhaplotyper2 \
  --tumor_sample $TUMOR_SM \
  --normal_sample $NORMAL_SM \
  [--pon $PON] \
  --germline_vcf $GERMLINE_VCF \
  output-tnhap2-tmp.vcf.gz \
  --algo OrientationBias \
  --tumor_sample $TUMOR_SM \
  output-orientation \
  --algo ContaminationModel \
  --tumor_sample $TUMOR_SM \
  --normal_sample $NORMAL_SM \
  --vcf $CONTAMINATION_VCF \
  --tumor_segments output-contamination-segments \
  output-contamination

sentieon driver -r $fasta \
  --algo TNfilter \
  -v output-tnhap2-tmp.vcf.gz \
  --tumor_sample $TUMOR_SM \
  --normal_sample $NORMAL_SM \
  [--contamination output-contamination] \
  [--tumor_segments output-contamination-segments] \
  [--orientation_priors output-orientation] \
  output-tnhap2.vcf.gz
```

11.2.4.2 Sample type 2 - Target panel (hybridization capture)

This section describes TNscope® parameters for somatic variant calling from targeted panel sequencing (hybridization capture, 200-5000x depth, AF >= 1%). The thresholds for `--min_tumor_af` and `--min_depth` should be adjusted as needed.

```
sentieon driver -r $fasta -t $nt -i tumor_deduped.bam -i normal_deduped.bam \
  --interval $BED \
  --algo TNscope \
  --tumor_sample $TUMOR_SM \
  --normal_sample $NORMAL_SM \
  --dbsnp $dbsnp \
  --min_tumor_allele_frac 0.009 \
  --max_fisher_pv_active 0.05 \
  --max_normal_alt_cnt 10 \
  --max_normal_alt_frac 0.01 \
  --max_normal_alt_qsum 250 \
  --sv_mask_ext 10 \
  --prune_factor 0 \
  [--pon panel_of_normal.vcf] \
  output_tnscope.pre_filter.vcf.gz
```

(continues on next page)

(continued from previous page)

```
sentieon pyexec tnscope_filter.py \
    -v output_tnscope.pre_filter.vcf.gz \
    --tumor_sample $TUMOR_SM \
    --normal_sample $NORMAL_SM \
    -x tissue_panel --min_tumor_af 0.0095 --min_depth 100 \
    output_tnscope.filter.vcf.gz
```

11.2.4.3 Sample type 3 - Target panel (amplicon)

This section describes TNscope® parameters for somatic variant calling from targeted panel sequencing (amplicon, 200-5000x depth, AF >= 1%). The thresholds for `--min_tumor_af` and `--min_depth` should be adjusted as needed.

```
sentieon driver -r $fasta -t $nt -i tumor_deduped.bam -i normal_deduped.bam \
  --interval $BED --interval_padding 10 \
  --algo TNscope \
  --tumor_sample $TUMOR_SM \
  --normal_sample $NORMAL_SM \
  --dbsnp $dbsnp \
  --min_tumor_allele_frac 0.009 \
  --max_fisher_pv_active 0.05 \
  --max_normal_alt_cnt 10 \
  --max_normal_alt_frac 0.01 \
  --max_normal_alt_qsum 250 \
  --sv_mask_ext 10 \
  --prune_factor 0 \
  --assemble_mode 2 \
  [--pon panel_of_normal.vcf ] \
  output_tnscope.pre_filter.vcf.gz

sentieon pyexec tnscope_filter.py \
    -v output_tnscope.pre_filter.vcf.gz \
    --tumor_sample $TUMOR_SM \
    --normal_sample $NORMAL_SM \
    -x amplicon --min_tumor_af 0.0095 --min_depth 200 \
    output_tnscope.filter.vcf.gz
```

11.2.4.4 Sample type 4 - ctDNA (tumor-only without UMI)

This section describes TNscope® parameters for ctDNA and other high-depth cases (6000x-10000x depth, AF > 0.5%). The thresholds for `--min_tumor_af` and `--min_depth` should be adjusted as needed. You can also use the consensus-based deduplication to correct errors introduced by PCR and sequencing. Please see our Application Note on *consensus-based deduplication* without UMI for more details.

```
sentieon driver -t $nt -i tumor_sorted.bam \
  --algo LocusCollector --consensus --fun score_info tumor_score.txt

sentieon driver -r $fasta -t $nt -i tumor_sorted.bam \
  --algo Dedup [--rmdup] --score_info tumor_score.txt  \
  tumor_deduped.bam
```

(continues on next page)

(continued from previous page)

```
sentieon driver -r $fasta -t $nt -i tumor_deduped.bam --interval $BED \
  --algo TNscope \
  --tumor_sample $TUMOR_SM \
  --dbsnp $dbsnp \
  --disable_detector sv \
  --min_tumor_allele_frac 3e-3 \
  --min_tumor_lod 3.0 \
  --min_init_tumor_lod 1.0 \
  --assemble_mode 4 \
  --pcr_indel_model NONE \
  --resample_depth 100000
  [--pon panel_of_normal.vcf ] \
  output_tnscope.pre_filter.vcf.gz

sentieon pyexec tnscope_filter.py \
    -v output_tnscope.pre_filter.vcf.gz \
    --tumor_sample $TUMOR_SM \
    -x ctdna --min_tumor_af 0.005 --min_depth 400 \
    output_tnscope.filter.vcf.gz
```

11.3 UMI data processing with Sentieon® UMI and TNscope®

This section describes TNscope® parameters for ctDNA and other high-depth samples with UMI-tagged reads (20000x-50000x depth, AF > 0.1%). The thresholds for `--min_tumor_af` and `--min_depth` should be adjusted as needed.

Please see our Application Note on *Deduplication and UMI Handling* for more details.

Please find the python script `tnscope_filter.py` for filtering at Sentieon's GitHub page at https://github.com/Sentieon/sentieon-scripts.

11.3.1 Step1. Alignment and consensus generation

```
if [ "$DUPLEX_UMI" = "true" ] ; then
      READ_STRUCTURE="-d $READ_STRUCTURE"
fi
sentieon umi extract $READ_STRUCTURE $fastq_1 $fastq_2 | \
  sentieon bwa mem -p -C -R "@RG\tID:$tumor\tSM:$tumor\tPL:$platform" -t $nt \
  -K 10000000 $fasta - | \
  sentieon util sort --sam2bam -i - -o tumor_sorted.bam

sentieon driver -t $nt -i tumor_sorted.bam --algo LocusCollector \
  --consensus --umi_tag XR --fun score_info tumor_score.txt

sentieon driver -t $nt -i tumor_sorted.bam -r $FASTA --algo Dedup \
  --score_info tumor_score.txt tumor_deduped.bam
```

11.3.2 Step2. Variant calling with UMI consensus reads

```
sentieon driver -r $fasta -t $nt -i tumor_deduped.bam --interval $BED \
  --algo TNscope \
  --tumor_sample $TUMOR_SM \
  --dbsnp $dbsnp \
  --min_tumor_allele_frac 0.001 \
  --min_tumor_lod 3.0 \
  --min_init_tumor_lod 3.0 \
  --pcr_indel_model NONE \
  --resample_depth 100000 \
  --assemble_mode 4 \
  output_tnscope.pre_filter.vcf.gz

sentieon pyexec tnscope_filter.py \
    -v output_tnscope.pre_filter.vcf.gz \
    --tumor_sample $TUMOR_SM \
    -x ctdna_umi --min_tumor_af 0.001 --min_depth 1000 \
    output_tnscope.filtered.vcf.gz
```

11.4 Appendix

11.4.1 Description of optional command-line arguments

TNscope:

- –assemble_mode: assemble_mode=2 achieves a good balance between accuracy and speed. assemble_mode=4 may be used to handle more complex regions.
- –sv_mask_ext: Prevent SNP/Indel detection in the proximity of an SV breakpoint.
- –max_fisher_pv_active: Turn on and set a threshold for the PV value filter, which measures the statistical difference between the tumor and normal sample at a particular site.
- –min_tumor_allele_frac: Set the minimum tumor AF to be considered as a potential variant site.
- –min_init_tumor_lod: Minimum tumor log odds ratio in initial variant calling pass.
- –min_tumor_lod: Minimum tumor log odds ratio to filter in the output VCF with the TLOD annotation.
- –prune_factor: Use prune_factor=0 to turn on the adaptive graph pruning algorithm.

11.4.2 FAQ

1. **The AD ratio does not match the reported AF in the output VCF:**
    - They are calculated and defined differently, for legacy reasons from GATK
    - AD: based on the pile-up of the input bam, before local assembly is done
    - AF: the ratio of the number of supporting reads for REF and ALT alleles, after local assembly is done
    - Filtered differently: Reads go through slightly different sets of filters before the calculation
2. **Using the Panel of Normal file:**
    - Panel of Normal can be used with or without matching normal samples
    - When a variant is found in the PoN, it is flagged with the `panel_of_normal` filter

- The Panel of Normal is helpful when matching normal is not available
- Follow this link for instructions on *generating a panel of normal VCF* with TNscope

3. **Role of COSMIC and dbSNP database file:**
    - They only affect Tumor-Normal cases, and only impact the filtering
    - Both COSMIC and dbSNP only determine the threshold of NOLD for `germline_risk`
4. **Complex variants:**
    - Sentieon provides an experimental Python script for merging neighboring variants in the same phase. This script is only experimental and is available at https://github.com/Sentieon/sentieon-scripts/blob/master/merge_mnp/merge_mnp.py.

11.5 References

- Freed, Donald, Renke Pan, and Rafael Aldana. "TNscope: Accurate Detection of Somatic Mutations with Haplotype-based Variant Candidate Detection and Machine Learning Filtering." bioRxiv (2018): 250647. Link[43]
- Pei, Surui, et al. "Benchmarking variant callers in next-generation and third-generation sequencing analysis." Briefings in Bioinformatics (2020). Link[44]

[43] https://www.biorxiv.org/content/10.1101/250647v1.full
[44] https://academic.oup.com/bib/advance-article-abstract/doi/10.1093/bib/bbaa148/5875142

12 Sentieon Application Notes

12.1 Arguments Correspondence

12.1.1 Introduction

This documents describes how to execute the Broad institute GATK Best Practices described in https://www.broadinstitute.org/gatk/guide/best-practices using the Sentieon® Genomics software. The document also described the correspondence between arguments of the different tools used.

This document should help you determine how to convert your existing pipelines to using Sentieon® and allow you to provide feedback to the Sentieon® team on what arguments are required for your work but are unavailable in the Sentieon® Genomics software.

12.1.1.1 Correspondence of tools

The table below shows the Sentieon® tool that implement functionality consistent with existing GATK pipeline tools.

Table 12.1: Broad/GATK matching tools

Sentieon tool	GATK pipeline tool	Version correspondence for 202503.02
Sentieon BWA	BWA	BWA 0.7.17
Sentieon STAR	STAR	STAR 2.7.10b
Sentieon minimap2	minimap2	minimap2 2.26 (r1175)
Dedup and LocusCollector	Picard MarkDuplicates	Picard 2.9.0
Realigner	RealignerTargetCreator and IndelRealigner	GATK 3.7/GATK3.8
QualCal	BaseRecalibrator	GATK 3.7/GATK3.8/GATK 4.0/GATK 4.1
ReadWriter	PrintReads	GATK 3.7/GATK3.8/GATK 4.0/GATK 4.1
QualCal	AnalyzeCovariates	GATK 3.7/GATK3.8/GATK 4.0/GATK 4.1
Genotyper	UnifiedGenotyper	GATK 3.7/GATK3.8
Haplotyper	HaplotypeCaller	GATK 3.7/GATK3.8/GATK 4.0/GATK 4.1

continues on next page

Table 12.1 – continued from previous page

Sentieon tool	GATK pipeline tool	Version correspondence for 202503.02
GVCFtyper	GenotypeGVCFs	GATK 3.7/GATK3.8/GATK 4.0/GATK 4.1
VarCal	VariantRecalibrator	GATK 3.7/GATK3.8/GATK 4.0/GATK 4.1
ApplyVarCal	ApplyRecalibration/ApplyVQSR	GATK 3.7/GATK3.8/GATK 4.0/GATK 4.1
TNsnv	MuTect	MuTect 1.1.5
TNhaplotyper	MuTect2	GATK 3.7/GATK3.8
TNhaplotyper2	GATK4 Mutect2	GATK 4.2.0.0
RNASplitReadsAtJunction	SplitNCigarReads	GATK 3.7/GATK3.8/GATK 4.0/GATK 4.1
AlignmentStat	Picard CollectAlignmentSummaryMetrics	Picard 2.9.0
BaseDistributionByCycle	Picard CollectBaseDistributionByCycle	Picard 2.9.0
CollectVCMetrics	Picard CollectVariantCallingMetrics	Picard 2.9.0
ContaminationAssessment	ContEst	GATK 3.7/GATK3.8
CoverageMetrics	DepthOfCoverage	GATK 3.7/GATK3.8
GCBias	Picard CollectGcBiasMetrics	Picard 2.9.0
HsMetricAlgo	Picard CollectHsMetrics	Picard 2.9.0
InsertSizeMetricAlgo	Picard CollectInsertSizeMetrics	Picard 2.9.0
MeanQualityByCycle	Picard MeanQualityByCycle	Picard 2.9.0
QualDistribution	Picard QualityScoreDistribution	Picard 2.9.0
QualityYield	Picard CollectQualityYieldMetrics	Picard 2.9.0
SequenceArtifactMetricsAlgo	Picard CollectSequencingArtifactMetrics	Picard 2.9.0
WgsMetricsAlgo	Picard CollectWgsMetrics	Picard 2.9.0
ContaminationModel	GetPileupSummaries and CalculateContamination	GATK 4.2.0.0
OrientationBias	Mutect2 and LearnReadOrientationModel	GATK 4.2.0.0
TNfilter	FilterMutectCalls	GATK 4.2.0.0

12.1.2 Detailed description per stage

12.1.2.1 Map to Reference - Alignment

GATK Best Practices command line

```
bwa mem -M -R '@RG\tID:GROUP_NAME \tSM:SAMPLE_NAME \tPL:PLATFORM' -p \
   -t NUMBER_THREADS REFERENCE.FASTA SAMPLE.FQ > ALIGNED.SAM
java -jar picard.jar SortSam INPUT=ALIGNED.SAM \
   OUTPUT=SORTED.SAM SORT_ORDER=coordinate
samtools view -bS SORTED.SAM > SORTED.BAM
samtools index SORTED.BAM
```

Sentieon® command line

```
sentieon bwa mem -M -R '@RG\tID:GROUP_NAME \tSM:SAMPLE_NAME \tPL:PLATFORM' -p \
   -t NUMBER_THREADS REFERENCE.FASTA SAMPLE.FQ | sentieon util sort \
   -o SORTED.BAM -t NUMBER_THREADS --sam2bam -i -
```

The BWA alignment command is identical, except that we recommend that the results from BWA be piped to the sorting stage in Sentieon®, instead of outputting to a SAM file.

The sorting using Sentieon® can only be ordered by coordinate.

Sentieon® will automatically create an index file for the sorted bam file.

12.1.2.2 Mark Duplicates - Dedup

GATK Best Practices command line

```
java -jar picard.jar MarkDuplicates INPUT=SORTED.BAM \
   OUTPUT=DEDUP.BAM METRICS_FILE=DEDUP_METRICS.TXT \
   REMOVE_DUPLICATES=true
java -jar picard.jar BuildBamIndex INPUT=DEDUP.BAM
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -i SORTED.BAM --algo LocusCollector\
   --fun score_info SCORE.TXT.GZ
sentieon driver -t NUMBER_THREADS -i SORTED.BAM --algo Dedup --rmdup \
   --score_info SCORE.TXT.GZ --metrics DEDUP_METRICS.TXT DEDUP.BAM
```

The last argument of the Sentieon® command line is the output bam file. Sentieon® will automatically create an index file for the deduped bam file.

Table 12.2: Argument correspondence for Dedup

Picard option	Sentieon option	Meaning
INPUT=SORTED.BAM	-i SORTED.BAM	Input the bam file
OUTPUT=DEDUP.BAM	N/A	Output bam file
METRICS_FILE=METRICS.TXT	–metrics METRICS.TXT	Output metrics
REMOVE_DUPLICATES=true	–rmdup	Remove duplicates from bam
OPTICAL_DUPLICATE_PIXEL_DISTANCE=	–optical_dup_pix_dist DISTANCE	Optical duplicate distance

12.1.2.3 Realign Indels - Realignment

GATK3 Best Practices command line

```
java -jar GenomeAnalysisTK.jar -T RealignerTargetCreator \
   -R REFERENCE.FASTA -I DEDUP.BAM -L INTERVAL \
   -known KNOWN_SITES.VCF -o REALIGNEMNT_TARGETS.LIST
java -jar GenomeAnalysisTK.jar -T IndelRealigner \
   -R REFERENCE.FASTA -I DEDUP.BAM \
   -targetIntervals REALIGNEMNT_TARGETS.LIST \
   -known KNOWN_SITES.VCF -o REALIGNED.BAM
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA -i DEDUP.BAM \
   --algo Realigner -k KNOWN_SITES.VCF --interval_list INTERVAL \
   REALIGNED.BAM
```

The last argument of the Sentieon® command line is the output bam file.

Table 12.3: Argument correspondence for Realign

GATK option	Sentieon option	Meaning
-I DEDUP.BAM	-i DEDUP.BAM	Input the bam file
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-o REALIGNED.BAM	N/A	Output bam file
-known KNOWN_SITES.VCF	-k KNOWN_SITES.VCF	Known sites
-L INTERVAL	–interval_list INTERVAL	Interval to restrict calculation

12.1.2.4 Recalibrate Bases - BQSR

12.1.2.4.1 BQSR - calculate recalibration

GATK3 Best Practices command line to generate the recalibration table

```
java -jar GenomeAnalysisTK.jar -T BaseRecalibrator \
   -R REFERENCE.FASTA -I REALIGNED.BAM -L INTERVAL \
   -knownSites KNOWN_SITES.VCF -o RECAL_DATA.TABLE
```

GATK4 Best Practices command line to generate the recalibration table

```
gatk BaseRecalibrator \
   -R REFERENCE.FASTA -I REALIGNED.BAM -L INTERVAL \
   --enable-baq --known-sites KNOWN_SITES.VCF \
   -O RECAL_DATA.TABLE
```

Sentieon command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA -i REALIGNED.BAM\
   --interval INTERVAL \
   --algo QualCal -k KNOWN_SITES.VCF RECAL_DATA.TABLE
```

The last argument of the Sentieon® command line is the recalibrated data table.

GATK4 removed the calculation of Per-Base Alignmnet Qualities (BAQ) to help reduce the runtime of BaseRecalibrator, https://github.com/broadinstitute/gatk/issues/2060. Adding the argument `--enable-baq` turns on BAQ calculation in the GATK4, which matches the behavior of Sentieon® QualCal algorithm.

Table 12.4: Argument correspondence - calculate BQSR - GATK3

GATK3 option	Sentieon option	Meaning
-I REALIGNED.BAM	-i REALIGNED.BAM	Input the bam file
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-o RECAL_DATA.TABLE	N/A	Output file
-knownSites KNOWN_SITES.VCF	-k KNOWN_SITES.VCF	Known sites
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation

Table 12.5: Argument correspondence - calculate BQSR - GATK4

GATK4 option	Sentieon option	Meaning
-I REALIGNED.BAM	-i REALIGNED.BAM	Input the bam file
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-O RECAL_DATA.TABLE	N/A	Output file
–known-sites KNOWN_SITES.VCF	-k KNOWN_SITES.VCF	Known sites
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation

12.1.2.4.2 BQSR - apply recalibration

GATK3 Best Practices command line to apply recalibration

```
java -jar GenomeAnalysisTK.jar -T PrintReads \
   -R REFERENCE.FASTA -I REALIGNED.BAM -L INTERVAL \
   -BQSR RECAL_DATA.TABLE -o RECALED.BAM
```

GATK4 Best Practices command line to apply recalibration

```
gatk ApplyBQSR \
   -R REFERENCE.FASTA -I REALIGNED.BAM -L INTERVAL \
   -bqsr RECAL_DATA.TABLE -O RECALED.BAM
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA -i REALIGNED.BAM\
   -q RECAL_DATA.TABLE --interval INTERVAL \
   --algo QualCal -k KNOWN_SITES RECAL_DATA.TABLE.POST \
   --algo ReadWriter RECALED.BAM
```

The last argument of the Sentieon® command line is the output bam file.

The Sentieon® ReadWriter command can be run together either with the step generating the **RECAL_DATA.TABLE.POST** above, or with the variant calling step to speed up the pipeline.

GATK4 removed the base quality score recalibration of INDELs when using default settings, so the GATK4 Best Practices command line shown above will not produce the same results as those produced by Sentieon®; in particular, the Sentieon® BAM output will contain BI/BD tags from the INDEL recalibration that will be missing from the GATK4 BAM output. This removal of INDEL recalibration was done to reduce GATK runtime at the expense of accuracy, so it is not recommend when using Sentieon® as the speed improvement is neglegible. In order to fully match the results of GATK4 ApplyBQSR, it is possible to use the `--read_filter QualCalFilter` option instead of the `-q RECAL_DATA.TABLE` in the ReadWriter command, which allows skipping the INDEL recalibration:

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA -i REALIGNED.BAM\
   --read_filter QualCalFilter,table=RECAL_DATA.TABLE.table,indel=false \
   --interval INTERVAL \
   --algo ReadWriter RECALED.BAM
```

Table 12.6: Argument correspondence - apply BQSR - GATK3

GATK3 option	Sentieon option	Meaning
-I REALIGNED.BAM	-i REALIGNED.BAM	Input the bam file
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-o RECALED.BAM	N/A	Output file
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation
-BQSR RECAL_DATA.TABLE	-q RECAL_DATA.TABLE	Recalibration table

Table 12.7: Argument correspondence - apply BQSR - GATK4

GATK4 option	Sentieon option	Meaning
-I REALIGNED.BAM	-i REALIGNED.BAM	Input the bam file
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-O RECALED.BAM	N/A	Output file
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation
-bqsr RECAL_DATA.TABLE	-q RECAL_DATA.TABLE	Recalibration table

12.1.2.4.3 BQSR - plot recalibration

GATK3 Best Practices command line to plot BQSR metrics

```
java -jar GenomeAnalysisTK.jar -T BaseRecalibrator \
   -R REFERENCE.FASTA -I REALIGNED.BAM -L INTERVAL \
   -knownSites KNOWN_SITES.VCF -BQSR RECAL_DATA.TABLE \
   -o RECAL_DATA.TABLE.POST
java -jar GenomeAnalysisTK.jar -T AnalyzeCovariates \
   -R REFERENCE.FASTA -before RECAL_DATA.TABLE \
   -after RECAL_DATA.TABLE.POST -csv RECAL_RESULT.CSV -plots BQSR.PDF
```

GATK4 Best Practices command line to plot BQSR metrics

```
 gatk BaseRecalibrator \
    -R REFERENCE.FASTA -I RECALED.BAM -L INTERVAL \
    --enable-baq --known-sites KNOWN_SITES.VCF \
    -O RECAL_DATA.TABLE.POST
gatk AnalyzeCovariates \
   -before RECAL_DATA.TABLE -after RECAL_DATA.TABLE.POST \
   -csv RECAL_RESULT.CSV -plots BQSR.PDF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS --algo QualCal --plot --before \
   RECAL_DATA.TABLE --after RECAL_DATA.TABLE.POST RECAL_RESULT.CSV
sentieon plot QualCal -o BQSR.PDF RECAL_RESULT.CSV
```

Table 12.8: Argument correspondence - plot BQSR - GATK3/GATK4

GATK option	Sentieon option	Meaning
-R REFERENCE.FASTA	N/A	Reference file
-before RECAL_DATA.TABLE	–before RECAL_DATA.TABLE	Recalibration table
-after RECAL_DATA.TABLE	–after RECAL_DATA.TABLE	After-recalibration table
-plots BQSR.PDF	-o BQSR.PDF	Report file
-csv RECAL_RESULT.CSV	N/A	Output csv file

12.1.2.5 Unified Genotyper - Genotyper

GATK3 Best Practices command line

```
java -jar GenomeAnalysisTK.jar -T UnifiedGenotyper \
   -R REFERENCE.FASTA -I RECALED.BAM -L INTERVAL \
   -D DBSNP.VCF --glm [SNP/INDEL/BOTH] -mbq QUALITY \
   -stand_emit_conf CONFIDENCE -stand_call_conf CONFIDENCE \
   --output_mode [EMIT_VARIANTS_ONLY/EMIT_ALL_CONFIDENT_SITES/EMIT_ALL_SITES] \
   -o OUTPUT.VCF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA -i RECALED.BAM\
   --interval INTERVAL \
   --algo Genotyper \
   -d DBSNP.VCF --var_type [SNP/INDEL/BOTH] --min_base_qual QUALITY \
   --emit_conf CONFIDENCE --call_conf CONFIDENCE \
   --emit_mode [VARIANT/CONFIDENT/ALL] \
   OUTPUT.VCF
```

The last argument of the Sentieon® command line is the variant vcf file. The tool will output a compressed VCF file when using .gz extension.

Bear in mind that since GATK 3.7, the `stand_emit_conf` is no longer supported, and the default value for `stand_call_conf` has been changed from 30 to 10, while the default in Sentieon® `call_conf` stayed at 30.

Table 12.9: Argument correspondence - UnifiedGenotyper

GATK option	Sentieon option	Meaning
-I RECALED.BAM	-i RECALED.BAM	Input the bam file
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-D DBSNP.VCF	-d DBSNP.VCF	dbSNP file
–glm [SNP/INDEL/BOTH]	–var_type [SNP/INDEL/BOTH]	Variant output type
-mbq QUALITY	–min_base_qual QUALITY	Minimum base quality
-stand_emit_conf CONFIDENCE	–emit_conf CONFIDENCE	Emit confidence threshold
-stand_call_conf CONFIDENCE	–call_conf CONFIDENCE	Call confidence threshold
–output_mode MODE	–emit_mode MODE	Emit mode
-ploidy PLOIDY	–ploidy PLOIDY	Ploidy of the sample
-o OUTPUT.VCF	N/A	Output variant file
-alleles GIVEN.VCF -gt_mode GENOTYPE_GIVEN_ALLELES	–given GIVEN.VCF	Perform variant calling using only the variants provided in the GIVEN_VCF

12.1.2.6 HaplotypeCaller - Halotyper

GATK3 Best Practices command line - VCF output

```
java -jar GenomeAnalysisTK.jar -T HaplotypeCaller \
   -R REFERENCE.FASTA -I RECALED.BAM -L INTERVAL \
   -D DBSNP.VCF -mbq QUALITY --minPruning FACTOR \
   -stand_emit_conf CONFIDENCE -stand_call_conf CONFIDENCE \
   -pcrModel [HOSTILE/AGGRESSIVE/CONSERVATIVE/NONE] \
   --output_mode [EMIT_VARIANTS_ONLY/EMIT_ALL_CONFIDENT_SITES/EMIT_ALL_SITES] \
   -o OUTPUT.VCF
```

GATK4 Best Practices command line - VCF output

```
gatk HaplotypeCaller \
   -R REFERENCE.FASTA -I RECALED.BAM -L INTERVAL \
   -D DBSNP.VCF -mbq QUALITY --min-pruning FACTOR \
   -stand-call-conf CONFIDENCE -new-qual false \
   --pcr-indel-model [HOSTILE/AGGRESSIVE/CONSERVATIVE/NONE] \
   --output-mode [EMIT_VARIANTS_ONLY/EMIT_ALL_CONFIDENT_SITES/EMIT_ALL_SITES] \
   -O OUTPUT.VCF
```

Sentieon® command line - VCF output

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA -i RECALED.BAM\
   --interval INTERVAL \
   --algo Haplotyper -d DBSNP.VCF \
   --min_base_qual QUALITY --prune_factor FACTOR \
   --emit_conf CONFIDENCE --call_conf CONFIDENCE \
   --emit_mode [VARIANT/CONFIDENT/ALL]
   --pcr_indel_model [HOSTILE/AGGRESSIVE/CONSERVATIVE/NONE] \
   OUTPUT.VCF
```

GATK3 Best Practices command line - gVCF output

```
java -jar GenomeAnalysisTK.jar -T HaplotypeCaller \
   -R REFERENCE.FASTA -I RECALED.BAM -L INTERVAL \
   -D DBSNP.VCF -mbq QUALITY --minPruning FACTOR \
   -stand_emit_conf CONFIDENCE -stand_call_conf CONFIDENCE \
   -pcrModel [HOSTILE/AGGRESSIVE/CONSERVATIVE/NONE] \
   --emitRefConfidence GVCF \
   -o OUTPUT.VCF
```

GATK4 Best Practices command line - gVCF output

```
gatk HaplotypeCaller \
   -R REFERENCE.FASTA -I RECALED.BAM -L INTERVAL \
   -D DBSNP.VCF -mbq QUALITY --min-pruning FACTOR \
   -stand-call-conf CONFIDENCE \
   --pcr-indel-model [HOSTILE/AGGRESSIVE/CONSERVATIVE/NONE] \
   -ERC GVCF \
   -O OUTPUT.VCF
```

Sentieon® command line - gVCF output

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA -i RECALED.BAM \
   --interval INTERVAL \
   --algo Haplotyper -d DBSNP.VCF \
   --min_base_qual QUALITY --prune_factor FACTOR \
   --emit_conf CONFIDENCE --call_conf CONFIDENCE \
   --emit_mode GVCF
   --pcr_indel_model [HOSTILE/AGGRESSIVE/CONSERVATIVE/NONE] \
   OUTPUT.VCF
```

The last argument of the Sentieon® command line is the output vcf file. The tool will output a compressed VCF file when using .gz extension.

Bear in mind that since GATK 3.7, the `stand_emit_conf` is no longer supported. Also, the default value for `stand_call_conf` was changed from 30 to 10 in the GATK 3.7 to GATK 4.0 and was reverted to 30 in the GATK 4.1, while the default in Sentieon® `call_conf` has remained at 30.

Since the GATK 4.1 `-newQual` is default genotyping model.

Table 12.10: Argument correspondence - HaplotypeCaller - GATK3

GATK3 option	Sentieon option	Meaning
-I RECALED.BAM	-i RECALED.BAM	Input the bam file
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-D DBSNP.VCF	-d DBSNP.VCF	dbSNP file
-mbq QUALITY	–min_base_qual QUALITY	Minimum base quality
-stand_emit_conf CONFIDENCE	–emit_conf CONFIDENCE	Emit confidence threshold
-stand_call_conf CONFIDENCE	–call_conf CONFIDENCE	Call confidence threshold
–output_mode MODE	–emit_mode MODE	Emit mode
–emitRefConfidence GVCF	–emit_mode gvcf	Produce a g.vcf output
-ploidy PLOIDY	–ploidy PLOIDY	Ploidy of the sample
-o OUTPUT.VCF	N/A	Output variant file
-alleles GIVEN.VCF -gt_mode GENOTYPE_GIVEN_ALLELES	–given GIVEN.VCF	Perform variant calling using only the variants provided in the GIVEN_VCF
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation
-mmq QUALITY	–min_map_qual QUALITY	Minimum mapping quality
–minPruning FACTOR	–prune_factor FACTOR	Pruning factor
-pcrModel MODEL	–pcr_indel_model MODEL	PCR model
–dontUseSoftClippedBases	–trim_soft_clip	Trim off soft-clipped bases
–annotation ANNOTATION	–annotation ANNOTATION	Annotations to apply to the variant calls
–excludeAnnotation ANNOTATION	–annotation !ANNOTATION	Annotations to exclude in the variant calls by using the ‘!’ prefix
-newQual	–genotype_model multinomial	Use the new simplified allele count model

Table 12.11: Argument correspondence - HaplotypeCaller - GATK4

GATK4 option	Sentieon option	Meaning
-I RECALED.BAM	-i RECALED.BAM	Input the bam file
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-D DBSNP.VCF	-d DBSNP.VCF	dbSNP file
-mbq QUALITY	–min_base_qual QUALITY	Minimum base quality
N/A	–emit_conf CONFIDENCE	Emit confidence threshold
-stand-call-conf CONFIDENCE	–call_conf CONFIDENCE	Call confidence threshold
–output-mode MODE	–emit_mode MODE	Emit mode
-ERC GVCF	–emit_mode gvcf	Produce a g.vcf output
-ploidy PLOIDY	–ploidy PLOIDY	Ploidy of the sample
-O OUTPUT.VCF	N/A	Output variant file
–alleles GIVEN.VCF –genotyping-mode GENOTYPE_GIVEN_ALLELES	–given GIVEN.VCF	Perform variant calling using only the variants provided in the GIVEN_VCF
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation
–minimum-mapping-quality QUALITY	–min_map_qual QUALITY	Minimum mapping quality
–min-pruning FACTOR	–prune_factor FACTOR	Pruning factor
–pcr-indel-model MODEL	–pcr_indel_model MODEL	PCR model
–dont-use-soft-clipped-bases	–trim_soft_clip	Trim off soft-clipped bases
–annotation ANNOTATION	–annotation ANNOTATION	Annotations to apply to the variant calls
–annotations-to-exclude ANNOTATION	–annotation !ANNOTATION	Annotations to exclude in the variant calls by using the '!' prefix
-new-qual	–genotype_model multinomial	Use the new simplified allele count model

12.1.2.7 Joint Genotype - GVCFtyper

GATK3 Best Practices command line

```
java -jar GenomeAnalysisTK.jar -T GenotypeGVCFs \
   -R REFERENCE.FASTA -L INTERVAL \
   -D DBSNP.VCF \
   -stand_emit_conf CONFIDENCE -stand_call_conf CONFIDENCE \
   -V INPUT_GVCF_1 -V INPUT_GVCF_2 -V INPUT_GVCF_3 \
   -o OUTPUT.VCF
```

GATK4 Best Practices command line

```
gatk GenotypeGVCFs \
   -R REFERENCE.FASTA -L INTERVAL \
   -D DBSNP.VCF -new-qual false \
   -stand-call-conf CONFIDENCE \
   -V INPUT_GVCF_1 -V INPUT_GVCF_2 -V INPUT_GVCF_3 \
   -O OUTPUT.VCF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   --interval INTERVAL \
   --algo GVCFtyper \
   -d DBSNP.VCF \
   --emit_conf CONFIDENCE --call_conf CONFIDENCE \
   --emit_mode [VARIANT/CONFIDENT/ALL] \
   -v INPUT_GVCF_1 -v INPUT_GVCF_2 -v INPUT_GVCF_3 \
   OUTPUT.VCF
```

The last argument of the Sentieon® command line is the output vcf file. The tool will output a compressed VCF file when using .gz extension.

Bear in mind that since GATK 3.7, the `stand_emit_conf` is no longer supported. Also, the default value for `stand_call_conf` was changed from 30 to 10 in the GATK 3.7 to GATK 4.0 and was reverted to 30 in the GATK 4.1, while the default in Sentieon® `call_conf` has remained at 30.

Since the GATK 4.1 `-newQual` is default genotyping model.

Table 12.12: Argument correspondence - GenotypeGVCF - GATK3

GATK3 option	Sentieon option	Meaning
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation
-D DBSNP.VCF	-d DBSNP.VCF	dbSNP file
-stand_emit_conf CONFIDENCE	–emit_conf CONFIDENCE	Emit confidence threshold
-stand_call_conf CONFIDENCE	–call_conf CONFIDENCE	Call confidence threshold
N/A	–emit_mode MODE	Emit mode
-V INPUT_GVCF_X	-v INPUT_GVCF_X	g.vcf input files
-o OUTPUT.VCF	N/A	Output variant file
-newQual	–genotype_model multinomial	Use the new simplified allele count model

Table 12.13: Argument correspondence - GenotypeGVCF - GATK4

GATK4 option	Sentieon option	Meaning
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation
-D DBSNP.VCF	-d DBSNP.VCF	dbSNP file
N/A	–emit_conf CONFIDENCE	Emit confidence threshold
-stand-call-conf CONFIDENCE	–call_conf CONFIDENCE	Call confidence threshold
N/A	–emit_mode MODE	Emit mode
-V INPUT_GVCF_X	-v INPUT_GVCF_X	g.vcf input files
-O OUTPUT.VCF	N/A	Output variant file
-new-qual	–genotype_model multinomial	Use the new simplified allele count model

12.1.2.8 Filter Variants - VQSR

12.1.2.8.1 VQSR - calculate recalibration

GATK3 Best Practices command line

```
java -jar GenomeAnalysisTK.jar -T VariantRecalibrator \
   -R REFERENCE.FASTA -input INPUT.VCF \
   -an ANNOTATION_1 -an ANNOTATION_2 . . .  \
   -mode [SNP/INDEL] \
   --resource:RESOURCE_PARAM RESOURCE.VCF . . . \
   -tranche TRANCH_THRES -tranche TRANCH_THRES . . . \
   --maxGaussians MAX_GAUSS --maxNegativeGaussians MAX_GAUSS \
   --maxIterations MAX_ITERATIONS \
   --aggregate AGREGATE_VCF \
   -tranchesFile TRANCHES_FILE \
   -rscriptFile R_PLOT_FILE \
   -recalFile RECAL_FILE
```

GATK4 Best Practices command line

```
gatk VariantRecalibrator \
   -R REFERENCE.FASTA -V INPUT.VCF \
   -an ANNOTATION_1 -an ANNOTATION_2 . . .  \
   -mode [SNP/INDEL] \
   --resource:RESOURCE_PARAM RESOURCE.VCF . . . \
   -tranche TRANCH_THRES -tranche TRANCH_THRES . . . \
   --max-gaussians MAX_GAUSS --max-negative-gaussians MAX_GAUSS \
   --max-iterations MAX_ITERATIONS \
   --aggregate AGREGATE_VCF \
   --tranches-file TRANCHES_FILE \
   --rscript-file R_PLOT_FILE \
   -O RECAL_FILE
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   --algo VarCal -v INPUT.VCF \
   --annotation ANNOTATION_1 --annotation ANNOTATION_2 . . . \
   --var_type [SNP/INDEL]  \
   --resource RESOURCE.VCF --resource_param RESOURCE_PARAM . . . \
   --tranche TRANCH_THRES --tranche TRANCH_THRES . . . \
   --max_gaussian MAX_GAUSS --max_neg_gaussian MAX_GAUSS \
   --max_iter MAX_ITERATIONS \
   --nthr NUMBER_THREADS_EM --srand RANDOM_SEED \
   --aggregate_data AGREGATE_VCF \
   --tranches_file TRANCHES_FILE \
   --plot_file PLOT_FILE \
   RECAL_FILE
```

The last argument of the Sentieon® command line is the output recal file.

The resource argument in Sentieon® is split into 2 consecutive arguments, one with the resource file and one with the resource parameters.

Table 12.14: Argument correspondence - calculate VQSR - GATK3

GATK3 option	Sentieon option	Meaning
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-input INPUT.VCF	-v INPUT.VCF	vcf input file
-an ANNONTATION	–annotation ANNOTATION	Annotation to use
-mode [SNP/INDEL]	–var_type [SNP/INDEL]	Mode to use
–resource	–resource/–resource_param	Resources to use
-tranche TRANCH_THRES	–tranche TRANCH_THRES	Thresholds for tranches
–maxGaussians MAX_GAUSS	–max_gaussians MAX_GAUSS	Max number of Gaussians used for positive model
–maxNegativeGaussians MAX_GAUSS	–max_neg_gaussians MAX_GAUSS	Max number of Gaussians used for negative model
–maxIterations MAX_ITERATIONS	–max_iter MAX_ITERATIONS	Max number of iterations
N/A	–srand RANDOM_SEED	Random seed for the EM calculation
–aggregate AGREGATE_VCF	–aggregate_data AGREGATE_VCF	Input aggregate data
-tranchesFile TRANCHES_FILE	–tranches_file TRANCHES_FILE	Output tranches file
-rscriptFile R_PLOT_FILE	–plot_file PLOT_FILE	Output file for plotting
-recalFile RECAL_FILE	N/A	Output recalibration file
-MQCap NUMBER	–max_mq NUMBER	Maximum MQ in the data

Table 12.15: Argument correspondence - calculate VQSR - GATK4

GATK4 option	Sentieon option	Meaning
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
–variant INPUT.VCF	-v INPUT.VCF	vcf input file
-an ANNONTATION	–annotation ANNOTATION	Annotation to use
-mode [SNP/INDEL]	–var_type [SNP/INDEL]	Mode to use
–resource	–resource/–resource_param	Resources to use
-tranche TRANCH_THRES	–tranche TRANCH_THRES	Thresholds for tranches
–max-gaussians MAX_GAUSS	–max_gaussians MAX_GAUSS	Max number of Gaussians used for positive model
–max-negative-gaussians MAX_GAUSS	–max_neg_gaussians MAX_GAUSS	Max number of Gaussians used for negative model
–max-iterations MAX_ITERATIONS	–max_iter MAX_ITERATIONS	Max number of iterations
N/A	–srand RANDOM_SEED	Random seed for the EM calculation
–aggregate AGREGATE_VCF	–aggregate_data AGREGATE_VCF	Input aggregate data
–tranches-file TRANCHES_FILE	–tranches_file TRANCHES_FILE	Output tranches file
–rscript-file R_PLOT_FILE	–plot_file PLOT_FILE	Output file for plotting
-O RECAL_FILE	N/A	Output recalibration file
-mq-cap NUMBER	–max_mq NUMBER	Maximum MQ in the data

12.1.2.8.2 VQSR - apply recalibration

GATK3 Best Practices command line

```
java -jar GenomeAnalysisTK.jar -T ApplyRecalibration \
   -R REFERENCE.FASTA -input INPUT.VCF \
   -mode [SNP/INDEL] --ts_filter_level SENSITIVITY \
```

(continues on next page)

(continued from previous page)

```
   -tranchesFile TRANCHES_FILE -recalFile RECAL_FILE \
   -o OUTPUT.VCF
```

GATK4 Best Practices command line

```
gatk ApplyVQSR \
   -R REFERENCE.FASTA -V INPUT.VCF \
   -mode [SNP/INDEL] -ts-filter-level SENSITIVITY \
   --tranches-file TRANCHES_FILE --recal-file RECAL_FILE \
   -O OUTPUT.VCF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   --algo ApplyVarCal -v INPUT.VCF \
   --var_type [SNP/INDEL] --sensitivity SENSITIVITY \
   --tranches_file TRANCHES_FILE --recal RECAL_FILE \
   OUTPUT.VCF
```

The last argument of the Sentieon® command line is the output vcf file. The tool will output a compressed VCF file when using .gz extension.

Table 12.16: Argument correspondence - apply VQSR - GATK3

GATK option	Sentieon option	Meaning
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-input INPUT.VCF	-v INPUT.VCF	vcf input file
–ts_filter_level SENSITIVITY	–sensitivity SENSITIVITY	Sensitivity
-mode [SNP/INDEL]	–var_type [SNP/INDEL]	Mode to use
-tranchesFile TRANCHES_FILE	–tranches_file TRANCHES_FILE	Input tranches file
-recalFile RECAL_FILE	–recal RECAL_FILE	Input recalibration file
-o OUTPUT.VCF	N/A	Output variant file

Table 12.17: Argument correspondence - apply VQSR - GATK4

GATK option	Sentieon option	Meaning
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-V INPUT.VCF	-v INPUT.VCF	vcf input file
-ts-filter-level SENSITIVITY	–sensitivity SENSITIVITY	Sensitivity
-mode [SNP/INDEL]	–var_type [SNP/INDEL]	Mode to use
–tranches-file TRANCHES_FILE	–tranches_file TRANCHES_FILE	Input tranches file
–recal-file RECAL_FILE	–recal RECAL_FILE	Input recalibration file
-O OUTPUT.VCF	N/A	Output variant file

12.1.2.9 MuTect - TNsnv

MuTect Best Practices command line

```
java -jar mutect.jar -T MuTect \
   -R REFERENCE.FASTA -L INTERVAL \
   -I:normal NORMAL_RECALED.BAM -I:tumor TUMOR_RECALED.BAM \
   --dbsnp DBSNP.VCF -o CALL_STATS_OUTPUT.TXT -vcf OUTPUT.VCF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i NORMAL_RECALED.BAM -i TUMOR_RECALED.BAM \
   --interval INTERVAL \
   --algo TNsnv --dbsnp DBSNP.VCF \
   --tumor_sample TUMOR_SM --normal_sample NORMAL_SM \
   -call_stats_out CALL_STATS_OUTPUT.TXT OUTPUT.VCF
```

The last argument of the Sentieon® command line is the output vcf file. The tool will output a compressed VCF file when using .gz extension.

The `normal_sample` and `tumor_sample` arguments are required in Sentieon as the BAM files are not explicitly differentiated, and could be a single co-realigned BAM file.

Table 12.18: Argument correspondence - TNsnv

GATK option	Sentieon option	Meaning
N/A	-i COREALIGNED.BAM	Input the corealigned bam file
-I:normal NORMAL_RECALED.BAM	-i NORMAL_RECALED.BAM	Input the bam files
-I:tumor TUMOR_RECALED.BAM	-i TUMOR_RECALED.BAM	Input the bam files
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
–normal_sample_name NORMAL_SAMPLE	–normal_sample NORMAL_SAMPLE	Input normal sample name
–tumor_sample_name TUMOR_SAMPLE	–tumor_sample TUMOR_SAMPLE	Input tumor sample name
–dbsnp DBSNP.VCF	–dbsnp DBSNP.VCF	dbSNP file
–cosmic COSMIC.VCF	–cosmic COSMIC.VCF	Input cosmic VCF file
–normal_panel PON.VCF	–pon PON.VCF	Input panel-of-normal VCF file
–artifact_detection_mode	–detect_pon	Turn on mode to detect artifacts in normal sample, used to generate the panel-of-normal
-vcf OUTPUT.VCF	N/A	Output tumor variants file
-o CALL_STATS.OUT	–call_stats_out CALL_STATS.OUT	Output call statistics file
–coverage_file COVERAGE_FILE	–stdcov_out COVERAGE_FILE	Output standard coverage wiggle file
–tumor_depth_file FILE	–tumor_depth_out FILE	Output wiggle file of depth of tumor reads
–normal_depth_file FILE	–normal_depth_out FILE	Output wiggle file of depth of normal reads
–power_file FILE	–power_out FILE	Output power file
–min_qscore QUALITY	–min_base_qual QUALITY	Filtering quality of the bases used in variant calling
–initial_tumor_lod NUMBER	–min_init_tumor_lod NUMBER	Minimum tumor log odds in the initial pass calling variants
–tumor_lod NUMBER	–min_tumor_lod NUMBER	Minimum tumor log odds in the final call of variants
–normal_lod NUMBER	–min_normal_lod NUMBER	Minimum normal log odds used to check that the tumor variant is not a normal variant

continues on next page

Table 12.18 – continued from previous page

GATK option	Sentieon option	Meaning
–fraction_contamination NUMBER	–contamination_frac NUMBER	Estimation of the contamination fraction from other samples
–minimum_mutation_cell_fraction NUMBER	–min_cell_mutation_frac NUMBER	Minimum fraction of cells which have mutation
–strand_artifact_lod NUMBER	–min_strand_bias_lod NUMBER	Minimum log odds for calling strand bias
–strand_artifact_power_threshold NUMBER	–min_strand_bias_power NUMBER	Minimum power for calling strand bias
–dbsnp_normal_lod NUMBER	–min_dbsnp_normal_lod NUMBER	Minimum log odds for calling normal non-variant at dbsnp sites
–minimum_normal_allele_fraction NUMBER	–min_normal_allele_frac NUMBER	Minimum allele fraction to be considered in normal
–tumor_f_pretest NUMBER	–min_tumor_allele_frac NUMBER	Minimum allelic fraction in tumor sample
–gap_events_threshold NUMBER	–max_indel NUMBER	Maximum of nearby indel events that are allowed
–heavily_clipped_read_fraction NUMBER	–max_read_clip_frac NUMBER	Maximum fraction of soft/hard clipped bases in a read
–fraction_mapq0_threshold NUMBER	–max_mapq0_frac NUMBER	Maximum ratio of reads whose mapq are 0 used to determine poor mapped area
–pir_median_threshold NUMBER	–min_pir_median NUMBER	Minimum read position median
–pir_mad_threshold NUMBER	–min_pir_mad NUMBER	Minimum read position median absolute deviation
–required_maximum_alt_allele_mapping_quality_score NUMBER	–max_alt_mapq NUMBER	Required maximum value of alt allele mapping quality score
–max_alt_alleles_in_normal_count NUMBER	–max_normal_alt_cnt NUMBER	Maximum alt alleles count in normal pileup
–max_alt_alleles_in_normal_qscor NUMBER	–max_normal_alt_qsum NUMBER	Maximum quality score sum of alt allele in normal pileup
–max_alt_allele_in_normal_fractio NUMBER	–max_normal_alt_frac NUMBER	Maximum fraction of alt allele in normal pileup
–power_constant_af NUMBER	–power_allele_frac NUMBER	Allele fraction used in power calculations

12.1.2.10 MuTect2 - TNhaplotyper

GATK3 MuTect2 Best Practices command line

```
java -jar GenomeAnalysisTK.jar -T MuTect2 \
   -R REFERENCE.FASTA -L INTERVAL \
   -I:normal NORMAL_RECALED.BAM -I:tumor TUMOR_RECALED.BAM \
   -D DBSNP.VCF -o OUTPUT.VCF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i NORMAL_RECALED.BAM -i TUMOR_RECALED.BAM \
```

(continues on next page)

(continued from previous page)

```
  --interval INTERVAL \
  --algo TNhaplotyper --dbsnp DBSNP.VCF \
  --tumor_sample TUMOR_SM --normal_sample NORMAL_SM \
  OUTPUT.VCF
```

The last argument of the Sentieon® command line is the output vcf file. The tool will output a compressed VCF file when using .gz extension.

The `normal_sample` and `tumor_sample` arguments are required in Sentieon as the BAM files are not explicitly differentiated, and could be a single co-realigned BAM file.

Table 12.19: Argument correspondence - TNhaplotyper

GATK option	Sentieon option	Meaning
N/A	-i COREALIGNED.BAM	Input the corealigned bam file
-I:normal NORMAL_RECALED.BAM	-i NORMAL_RECALED.BAM	Input the bam files
-I:tumor TUMOR_RECALED.BAM	-i TUMOR_RECALED.BAM	Input the bam files
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
N/A	–normal_sample NORMAL_SAMPLE	Input normal sample name
N/A	–tumor_sample TUMOR_SAMPLE	Input tumor sample name
-D DBSNP.VCF	–dbsnp DBSNP.VCF	dbSNP file
–cosmic COSMIC.VCF	–cosmic COSMIC.VCF	Input cosmic VCF file
–normal_panel PON.VCF	–pon PON.VCF	Input panel-of-normal VCF file
–artifact_detection_mode	–detect_pon	Turn on mode to detect artifacts in normal sample. It is used to generate the panel-of-normals
-o OUTPUT.VCF	N/A	Output tumor variants file.
-mbq QUALITY	–min_base_qual QUALITY	Filtering quality of the bases used in variant calling
–minPruning FACTOR	–prune_factor FACTOR	Pruning factor
-pcrModel MODEL	–pcr_indel_model MODEL	PCR model
–initial_tumor_lod NUMBER	–min_init_tumor_lod NUMBER	Minimum tumor log odds in the initial pass calling variants
–initial_normal_lod NUMBER	–min_init_normal_lod NUMBER	Minimum normal log odds in the initial pass calling variants
–tumor_lod NUMBER	–min_tumor_lod NUMBER	Minimum tumor log odds in the final call of variants
–normal_lod NUMBER	–min_normal_lod NUMBER	Minimum normal log odds used to check that the tumor variant is not a normal variant
–max_alt_alleles_in_normal_count NUMBER	–max_normal_alt_cnt NUMBER	Maximum alt alleles count in normal pileup
–max_alt_alleles_in_normal_qscor NUMBER	–max_normal_alt_qsum NUMBER	Maximum quality score sum of alt allele in normal pileup
–max_alt_allele_in_normal_fractio NUMBER	–max_normal_alt_frac NUMBER	Maximum fraction of alt allele in normal pileup
-contaminationFile TAB_FILE	–tumor_contamination_frac NUMBER	Estimation of the contamination fraction from other samples on the tumor sample
	–normal_contamination_frac NUMBER	Estimation of the contamination fraction from other samples on the normal sample

12.1.2.11 GATK4 Mutect2 - TNhaplotyper2 and TNfilter

GATK4 Mutect2 Best Practies command line

```
gatk Mutect2 -R REFERENCE.FASTA -I TUMOR_RECALED.BAM \
   -tumor TUMOR_SM -I NORMAL_RECALED.BAM -normal NORMAL_SM \
   --germline-resource GNOMAD.VCF -O TMP.VCF \
```

(continues on next page)

(continued from previous page)

```
   --f1r2-tar-gz F1R2.TAR.GZ -L INTERVAL
gatk GetPileupSummaries -R REFERENCE.FASTA -I TUMOR_RECALED.BAM \
   -V GNOMAD.VCF -O TUMOR.PILEUPS -L INTERVAL
gatk GetPileupSummaries -R REFERENCE.FASTA -I NORMAL_RECALED.BAM \
   -V GNOMAD.VCF -O NORMAL.PILEUPS -L INTERVAL
gatk LearnReadOrientationModel -I F1R2.TAR.GZ -O PRIORS
gatk CalculateContamination -I TUMOR.PILEUPS \
   -matched NORMAL.PILEUPS --tumor-segmentation SEGMENTS \
   -O CONTAMINATION.TABLE
gatk FilterMutectCalls -V TMP.VCF -R REFERENCE.FASTA \
   -O OUTPUT.VCF --contamination-table CONTAMINATION.TABLE \
   --tumor-segmentation SEGMENTS -ob-priors PRIORS \
   --stats TMP.VCF.stats --filtering-stats OUTPUT.VCF.stats
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i TUMOR_RECALED.BAM -i NORMAL_RECALED.BAM \
   --interval INTERVAL \
   --algo TNhaplotyper2 \
   --tumor_sample TUMOR_SM --normal_sample NORMAL_SM \
   --germline_vcf GNOMAD.VCF TMP.VCF \
   --algo OrientationBias --tumor_sample TUMOR_SM PRIORS \
   --algo ContaminationModel \
   --tumor_sample TUMOR_SM --normal_sample NORMAL_SM \
   -v GNOMAD.VCF --tumor_segments SEGMENTS CONTAMINATION.TABLE
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   --algo TNfilter -v TMP.VCF --tumor_sample TUMOR_SM \
   --normal_sample NORMAL_SM \
   --contamination CONTAMINATION.TABLE
   --tumor_segments SEGMENTS --orientation_priors PRIORS \
   OUTPUT.VCF
```

Some VCFs split multi-allelic sites into separate records. Mutect2 will ignore later split multi-allelic records in VCFs passed through the `--germline-resource` argument while TNhaplotyper2 will process all multi-allelic records in VCFs passed through the `--germline_vcf` argument. This can cause different results when the `GNOMAD.VCF` has multi-allelic sites split into separate records.

The above commands use the same VCF file, `GNOMAD.VCF`, for both Mutect2/TNhaplotyper2 and GetPileupSummaries/ContaminationModel. The GATK best practices typically uses a GnomAD VCF for Mutect2 but uses the same GnomAD VCF filtered for variants with `AF > 0.01 || AF < 0.2` in GetPileupSummaries. Using the same VCF with `--min_af 0.01 --max_af 0.2` (default) has the same effect.

Table 12.20: Argument correspondence - Mutect2 and TNhaplotyper2 - GATK4

GATK4 option	Sentieon option	Meaning
-I TUMOR_RECALED.BAM	-i TUMOR_RECALED.BAM	Input the bam files
-I NORMAL_RECALED.BAM	-i NORMAL_RECALED.BAM	Input the bam files
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference fasta
-L INTERVAL	–interval INTERVAL	Interval to restrict calculation
-tumor TUMOR_SM	–tumor_sample TUMOR_SM	Input tumor sample name
-normal NORMAL_SM	–normal_sample NORMAL_SM	Input normal sample name
-pon PON_FILE	–pon PON_FILE	A panel-of-normal file
-mbq MIN_BQ	–min_base_qual MIN_BQ	Minimum base quality
–min-pruning PRUNE	–prune_factor PRUNE	Pruning factor in local assembly
–pcr-indel-model INDEL_MODEL	–pcr_indel_model INDEL_MODEL	PCR indel error model
-init-lod INIT_T_LOD	–min_init_tumor_lod INIT_T_LOD	Minimum tumorLOD for candidate selection
-emit-lod T_LOD	–min_tumor_lod T_LOD	Minimum tumorLOD for called variants
–normal-lod N_LOD	–min_normal_lod N_LOD	Minimum normalLOD for called variants
–germline-resource GERMLINE.VCF	–germline_vcf GERMLINE.VCF	A germline VCF containing allele frequencies
–af-of-alleles-not-in-resource AF	–default_af AF	Allele frequency for variants not found in the germline VCF
–max-population-af MAX_AF	–max_germline_af MAX_AF	Maximum germline allele frequency in tumor-only mode
–genotype-pon-sites true	–call_pon_sites	Call candidate variants in the PoN
–callable-depth	–callable_depth	Minimum depth to be considered for statistics

Arguments in the OrientationBias algo map to arguments in both Mutect2 and LearnReadOrientationModel.

Table 12.21: Argument correspondence - Mutect2/LearnReadOrientationModel and OrientationBias

GATK4 option	Sentieon option	Meaning
-I TUMOR_RECALED.BAM	-i TUMOR_RECALED.BAM	Input the bam files
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference fasta
-tumor TUMOR_SM	–tumor_sample TUMOR_SM	Input tumor sample name
–f1r2-min-bq MIN_BQ	–min_base_qual MIN_BQ	Minimum base quality
–f1r2-median-mq MIN_MAPQ	–min_median_map_qual MIN_MAPQ	Minimum median mapping quality
–f1r2-max-depth MAX_DEPTH	–max_depth MAX_DEPTH	Sites with a higher depth will be grouped

Arguments in the ContaminationModel algo map to arguments in both GetPileupSummaries and CalculateContamination.

Table 12.22: Argument correspondence - GetPileupSummaries/CalculateContamination and ContaminationModel

GATK4 option	Sentieon option	Meaning
–input TUMOR_RECALED.BAM	-i TUMOR_RECALED.BAM	Input the bam files
–reference REFERENCE.FASTA	-r REFERENCE.FASTA	Reference fasta
–intervals INTERVAL	–interval INTERVAL	Interval to restrict calculation
N/A	–tumor_sample TUMOR_SM	Input tumor sample name
N/A	–normal_sample NORMAL_SM	Input normal sample name
–min-mapping-quality MIN_MAPQ	–min_map_qual MIN_MAPQ	Minimum mapping quality
–variant GNOMAD.VCF	-v GNOMAD.VCF	A VCF with population allele frequencies
–minimum-population-allele-frequency MIN_AF	–min_af MIN_AF	Minimum population allele frequency
–maximum-population-allele-frequency MIN_AF	–max_af MAX_AF	Maximum population allele frequency
–tumor-segmentation SEGMENTS	–tumor_segments SEGMENTS	Allele frequency segmentations output

Table 12.23: Argument correspondence - FilterMutectCalls and TNfilter

GATK4 option	Sentieon option	Meaning
-V TMP.VCF	-v TMP.VCF	The input VCF
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference fasta
N/A	–tumor_sample TUMOR_SM	Input tumor sample name
N/A	–normal_sample NORMAL_SM	Input normal sample name
–contamination-table CONTAMINATION.TABLE	–contamination CONTAMINATION.TABLE	The contamination table file
–tumor-segmentation SEGMENTS	–tumor_segments SEGMENTS	The tumor segmentation table
–ob-priors PRIORS	–orientation_priors PRIORS	The orientation prior table
–threshold-strategy STRATEGY	–threshold_strategy STRATEGY	Method to determine the filtering threshold
–f-score-beta BETA	–f_score_beta BETA	Relative weight of recall to precision in the F-score
–false-discovery-rate MAX_FP	–max_fp_rate MAX_FP	Maximum expected false-positive rate
–initial-threshold	–threshold THRESHOLD	Threshold for the constant strategy
–min-median-base-quality MIN_BQ	–min_median_base_qual MIN_BQ	Minimum median base quality
–max-events-in-region MAX_COUNT	–max_event_count MAX_COUNT	Maximum number of events in an active region
–unique-alt-read-count MIN_READS	–unique_alt_reads MIN_READS	Minimum number of unique reads supporting the alt allele
–max-median-fragment-length-difference MAX_MFRL	–max_mfrl_diff MAX_MFRL	Maximum median fragment length difference
–distance-on-haplotype MAX_DIST	–max_haplotype_distance MAX_DIST	Maximum distance to determine an artifact
–min-allele-fraction MIN_AF	–min_tumor_af MIN_AF	Minimum alternate allele fraction in the tumor sample
–min-median-mapping-quality MIN_MAPQ	–min_median_map_qual MIN_MAPQ	Minimum median mapping quality
–long-indel-length MAX_LEN	–long_indel_length MAX_LEN	Longer indels will use the reference mapping quality
–max-alt-allele-count MAX_ALT	–max_alt_count MAX_ALT	Maximum number of alternate alleles at a site
–max-n-ratio MAX_N	–max_n_ratio MAX_N	Maximum ratio of N to alt bases
–normal-p-value-threshold P_VALUE	–normal_p_value P_VALUE	P-value theshold for normal artifacts
–min-median-read-position MIN_DIST	–min_median_pos MIN_DIST	Minimum median distanct to the end of the read
–min-slippage-length MIN_SLIPPAGE	–min_slippage_length MIN_SLIPPAGE	Minimum length for polymerase slippage in STR regions
–pcr-slippage-rate SLIPPAGE_RATE	–slippage_rate SLIPPAGE_RATE	The rate of PCR slippage
–min-reads-per-strand MIN_ALT	–min_alt_reads_per_strand MIN_ALT	Number of reads supporting the alt allele per strand

12.1.2.12 SplitNCigarReads - RNASplitReadsAtJunction

GATK3 SplitNCigarReads Best Practices command line

```
java -jar GenomeAnalysisTK.jar -T SplitNCigarReads \
   -R REFERENCE.FASTA -I DEDUPED.BAM -o SPLIT.BAM \
   -rf ReassignOneMappingQuality -RMQF 255 -RMQT 60 \
   -U ALLOW_N_CIGAR_READS
```

GATK4 SplitNCigarReads Best Practices command line

```
gatk SplitNCigarReads \
   -R REFERENCE.FASTA -I DEDUPED.BAM -O SPLIT.BAM
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo RNASplitReadsAtJunction \
   --reassign_mapq 255:60 SPLIT.BAM
```

The last argument of the Sentieon® command line is the output bam file.

Table 12.24: Argument correspondence - SplitNCigarReads and RNASplitReadsAtJunction - GATK3

GATK3 option	Sentieon option	Meaning
-I DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference fasta
-rf ReassignOneMappingQuality -RMQF IN_QUAL -RMQT OUT_QUAL	–reassign_mapq IN_QUAL:OUT_QUAL	Reassign Mapping Quality from STAR
–doNotFixOverhangs	–ignore_overhang	Whether to ignore overhang
–maxBasesInOverhang NUMBER	–overhang_max_bases NUMBER	Max number of bases allowed in a hard-clipped overhang. Overhang will not be clipped if there are more than this value of bases
–maxMismatchesInOverhang NUMBER	–overhang_max_mismatches NUMBER	Max number of mismatches allowed in a non-hard-clipped overhang. Complete overhang will be hard-clipped if # of mismatches is above this value

Table 12.25: Argument correspondence - SplitNCigarReads and RNASplitReadsAtJunction - GATK4

GATK4 option	Sentieon option	Meaning
-I DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference fasta
N/A	–reassign_mapq IN_QUAL:OUT_QUAL	Reassign Mapping Quality from STAR
–do-not-fix-overhangs	–ignore_overhang	Whether to ignore overhang
–max-bases-in-overhang NUMBER	–overhang_max_bases NUMBER	Max number of bases allowed in a hard-clipped overhang. Overhang will not be clipped if there are more than this value of bases
–max-mismatches-in-overhang NUMBER	–overhang_max_mismatches NUMBER	Max number of mismatches allowed in a non-hard-clipped overhang. Complete overhang will be hard-clipped if # of mismatches is above this value

12.1.2.13 CollectAlignmentSummaryMetrics - AlignmentStat

Picard CollectAlignmentSummaryMetrics command line

```
java -jar picard.jar CollectAlignmentSummaryMetrics \
   I=ALIGNED.BAM O=ALN_METRICS.TXT \
   R=REFERENCE.FASTA \
   ADAPTER_SEQUENCE=ADAPTERS_SEQ
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i ALIGNED.BAM --algo AlignmentStat \
   --adapter_seq ADAPTERS_SEQ ALN_METRICS.TXT
```

Table 12.26: Argument correspondence - CollectAlignmentSummaryMetrics and AlignmentStat

Picard option	Sentieon option	Meaning
I=ALIGNED.BAM	-i ALIGNED.BAM	Input the bam files
O=ALN_METRICS.TXT	N/A	Output metrics
R=REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
ADAPTER_SEQUENCE=ADAPTERS_S	–adapter_seq ADAPTERS_SEQ	A string of adapters

12.1.2.14 CollectBaseDistributionByCycle - BaseDistributionByCycle

Picard CollectBaseDistributionByCycle command line

```
java -jar picard.jar CollectBaseDistributionByCycle \
   I=ALIGNED.BAM O=BASE_DISTRIBUTION_METRICS.TXT \
   CHART_OUTPUT=BASE_DISTRIBUTION.PDF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i ALIGNED.BAM --algo BaseDistributionByCycle \
   BASE_DISTRIBUTION_METRICS.TXT
```

Table 12.27: Argument correspondence - CollectBaseDistribution-ByCycle and BaseDistributionByCycle

Picard option	Sentieon option	Meaning
I=ALIGNED.BAM	-i ALIGNED.BAM	Input the bam files
O=BASE_DISTRIBUTION_METRICS.T	N/A	Output metrics
CHART_OUTPUT=BASE_DISTRIBUTI	N/A	Output chart
ALIGNED_READS_ONLY=true	–aligned_reads_only true	Calculate the base distribution over aligned reads only
PF_READS_ONLY=true	–pf_reads_only true	Calculate the base distribution over PF reads only

12.1.2.15 CollectVariantCallingMetrics - CollectVCMetrics

Picard CollectVariantCallingMetrics command line

```
java -jar picard.jar CollectVariantCallingMetrics \
   I=CALLS.VCF O=VC_METRICS_OUT DBSNP=DBSNP.VCF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   --algo CollectVCMetrics -d DBSNP.VCF -v CALLS.VCF \
   VC_METRICS_OUT
```

Table 12.28: Argument correspondence - CollectVCMetrics and CollectVariantCallingMetrics

Picard option	Sentieon option	Meaning
I=CALLS.VCF	-v CALLS.VCF	vcf input file
O=VC_METRICS_OUT	N/A	Output basename
DBSNP=DBSNP.VCF	-d DBSNP.VCF	dbSNP file

12.1.2.16 ContEst - ContaminationAssessment

GATK3 Best Practices command line

```
java -jar GenomeAnalysisTK.jar -T ContEst -I TUMOR_RECALED.BAM \
   -R REFERENCE.FASTA -pf POPULATION.VCF --genotypes GENOTYPES.VCF \
   -o OUTPUT.TXT
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA -i TUMOR_RECALED.BAM \
   --algo ContaminationAssessment --pop_vcf POPULATION.VCF \
   --genotype_vcf GENOTYPES.VCF OUTPUT.TXT
```

Table 12.29: Argument correspondence - ContaminationAssessment and ContEst

GATK option	Sentieon option	Meaning
-I TUMOR_RECALED.BAM	-i TUMOR_RECALED.BAM	Input the bam files
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference fasta
-pf POPULATION.VCF	–pop_vcf POPULATION.VCF	The VCF file containing allele frequency information for the population
–genotypes GENOTYPES.VCF	–genotype_vcf GENOTYPES.VCF	The VCF file containing variants reported for the inidividual
-llc [META/SAMPLE/READGROUP]	–type [META/SAMPLE/READGROUP]	Assess contamination by sample, lane or all reads
–min_qscore MIN_BQ	–min_base_qual MIN_BQ	Any bases with a quality less than MIN_BQ will be ignored
–min_mapq MIN_MAPQ	–min_map_qual MIN_MAPQ	Any reads with a mapping quality less than MIN_MAPQ will be ignored
-mbc MINIMUM_BASE_COUNT	–min_basecount MINIMUM_BASE_COUNT	The minimum number of bases present at a locus for contamination to be assessed
–beta_threshold TRIM	–trim_thresh TRIM	Theshold that will be used to trim sites
–trim_fraction TRIM_FRACTION	–trim_frac TRIM_FRACTION	Maximum fraction of sites that may be trimmed
-pc PRECISION	–precision PRECISION	The precision on the output percent number
-br BASE_REPORT	–base_report BASE_REPORT	The output file that will contain an extended report on the processed data
-population POPULATION	–population POPULATION	A population for the baseline allele frequency of the sample
-o OUTPUT.TXT	N/A	The output file

12.1.2.17 DepthOfCoverage - CoverageMetrics

GATK3 Best Practices command

```
java -jar GenomeAnalysisTK.jar -T DepthOfCoverage \
   -R REFERENCE.FASTA -I DEDUPED.BAM \
   -geneList GENE_LIST.REFSEQ -ct THRESHOLD \
   -o OUTPUT_BASE
```

GATK4 Best Practices command

```
gatk DepthOfCoverage \
   -R REFERENCE.FASTA -I DEDUPED.BAM \
   -gene-list GENE_LIST.REFSEQ \
   --summary-coverage-threshold THRESHOLD \
   -O OUTPUT_BASE
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo CoverageMetrics \
   --gene_list GENE_LIST.REFSEQ --cov_thresh THRESHOLD \
   OUTPUT_BASE
```

Table 12.30: Argument correspondence - CoverageMetrics and DepthOfCoverage - GATK3

GATK3 option	Sentieon option	Meaning
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-I DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
-geneList GENE_LIST.REFSEQ	–gene_list GENE_LIST.REFSEQ	RefSeq file used to aggregate results to the gene level
–countType COUNT_TYPE	–count_type COUNT_TYPE	Determines how to deal with overlapping paried ends
-pt PARTITION	–partition PARTITION	Determines how to partition the data
-ct THRESHOLD	–cov_thresh THRESHOLD	Add aggregation metrics for the percentage of bases with coverage greater than THRESHOLD
–start MIN_DEPTH	–histogram_low MIN_DEPTH	The smallest histogram bin
–stop MAX_DEPTH	–histogram_high MAX_DEPTH	The largest histogram bin
–nBins NUM_BINS	–histogram_bin_count NUM_BINS	The number of histogram bins
-mmq MIN_MAPQ	–min_map_qual MIN_MAPQ	Minimum mapping quality of reads used
–maxMappingQuality MAX_MAPQ	–max_map_qual MAX_MAPQ	Maximum mapping quality of reads used
-mbq MIN_BASEQ	–min_base_qual MIN_BASEQ	Minimum base quality of bases used
–maxBaseQuality MAX_BASEQ	–max_base_qual MAX_BASEQ	Maximum base quality of bases used
-omitBaseOutput	–omit_base_output	Omit output of the per locus coverage
-omitSampleSummary	–omit_sample_stat	Omit output of the summary results
-omitLocusTable	–omit_locus_stat	Omit output of histogram files
-omitIntervals	–omit_interval_stat	Omit output of interval statistics
-baseConts	–print_base_counts	Include the number of “ACGTND” in the output per locus coverage
–includeRefNSites	–include_ref_N	Include coverage data in loci where the reference genome is set to N
–ignoreDeletionSites	–ignore_del_sites	Ignore coverage data in loci where there are deletions
-dels	–include_del	Include deletions and add deletion counts
-o OUTPUT_BASE	N/A	Output file basename

Table 12.31: Argument correspondence - CoverageMetrics and DepthOfCoverage - GATK4

GATK4 option	Sentieon option	Meaning
-R REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
-I DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
-gene-list GENE_LIST.REFSEQ	–gene_list GENE_LIST.REFSEQ	RefSeq file used to aggregate results to the gene level
–count-type COUNT_TYPE	–count_type COUNT_TYPE	Determines how to deal with overlapping paried ends
-pt PARTITION	–partition PARTITION	Determines how to partition the data
–summary-coverage-threshold THRESHOLD	–cov_thresh THRESHOLD	Add aggregation metrics for the percentage of bases with coverage greater than THRESHOLD
–nBins NUM_BINS	–histogram_bin_count NUM_BINS	The number of histogram bins
-mbq MIN_BASEQ	–min_base_qual MIN_BASEQ	Minimum base quality of bases used
–min-base-quality MAX_BASEQ	–max_base_qual MAX_BASEQ	Maximum base quality of bases used
–omit-depth-output-at-each-base	–omit_base_output	Omit output of the per locus coverage
–omit-per-sample-statistics	–omit_sample_stat	Omit output of the summary results
–omit-locus-table	–omit_locus_stat	Omit output of histogram files
–omit-interval-statistics	–omit_interval_stat	Omit output of interval statistics
–print-base-counts	–print_base_counts	Include the number of “ACGTND” in the output per locus coverage
–include-ref-n-sites	–include_ref_N	Include coverage data in loci where the reference genome is set to N
–ignore-deletion-sites	–ignore_del_sites	Ignore coverage data in loci where there are deletions
–include-deletions	–include_del	Include deletions and add deletion counts
-O OUTPUT_BASE	N/A	Output file basename

12.1.2.18 CollectGcBiasMetrics - GCBias

Picard CollectGcBiasMetrics command line

```
java -jar picard.jar CollectGcBiasMetrics \
   I=DEDUPED.BAM O=GC_METRICS.TXT CHART=GC_BIAS.PDF \
   S=SUMMARY.TXT R=REFERENCE.FASTA ASSUME_SORTED=true
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo GCBias --summary SUMMARY.TXT \
   GC_METRICS.TXT
sentieon plot GCBias -o GC_BIAS.PDF GC_METRICS.TXT
```

Table 12.32: Argument correspondence - GCBias and CollectGcBiasMetrics

Picard option	Sentieon option	Meaning
I=DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
R=REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
O=GC_METRICS.TXT	N/A	GC bias metrics results
CHART=GC_BIAS.PDF	-o GC_BIAS.PDF	GC bias metrics report
S=SUMMARY.TXT	–summary SUMMARY.TXT	GC bias metrics summary results
LEVEL=LEVEL	–accum_level LEVEL	The accumulation level

12.1.2.19 CollectHsMetrics - HsMetricAlgo

Picard CollectHsMetrics command line

```
java -jar picard.jar CollectHsMetrics \
   I=DEDUPED.BAM O=HS_METRICS.TXT R=REFERENCE.FASTA \
   BAIT_INTERVALS=BAITS TARGET_INTERVALS=TARGETS
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo HsMetricAlgo --targets_list TARGETS \
   --baits_list BAITS HS_METRICS.TXT
```

Table 12.33: Argument correspondence - HsMetricAlgo and CollectHsMetrics

Picard option	Sentieon option	Meaning
I=DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
R=REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
O=HS_METRICS.TXT	N/A	HS metrics results
BAIT_INTERVALS=BAITS	–baits_list BAITS	Interval list input file of baits
TARGET_INTERVALS=TARGETS	–targets_list TARGETS	Interval list input file of targets
CLIP_OVERLAPPING_READS	–clip_overlapping_reads	Clip overlapping reads
MINIMUM_MAPPING_QUALITY=MIN	–min_map_qual MIN_MAPQ	Minimum read mapping quality
MINIMUM_BASE_QUALITY=MIN_BA	–min_base_qual MIN_BASEQ	Minimum base quality
COVERAGE_CAP=COVERAGE	–coverage_cap COVERAGE	Maximum coverage limit in the histogram

12.1.2.20 CollectInsertSizeMetrics - InsertSizeMetricAlgo

Picard CollectInsertSizeMetrics command line

```
java -jar picard.jar CollectInsertSizeMetrics \
   I=DEDUPED.BAM O=IS_METRICS.TXT R=REFERENCE.FASTA \
   H=IS_METRICS.PDF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo InsertSizeMetricAlgo \
   IS_METRICS.TXT
sentieon plot InsertSizeMetricAlgo -o IS_METRICS.PDF IS_METRICS.TXT
```

Table 12.34: Argument correspondence - InsertSizeMetricAlgo and CollectInsertSizeMetrics

Picard option	Sentieon option	Meaning
I=DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
O=IS_METRICS.TXT	N/A	IS metrics results
R=REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
H=IS_METRICS.PDF	-o IS_METRICS.PDF	Insert size metrics report

12.1.2.21 MeanQualityByCycle - MeanQualityByCycle

Picard MeanQualityByCycle command line

```
java -jar picard.jar MeanQualityByCycle \
   I=DEDUPED.BAM O=MQ_METRICS.TXT R=REFERENCE.FASTA \
   CHART=MQ_METRICS.PDF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo MeanQualityByCycle \
   MQ_METRICS.TXT
sentieon plot MeanQualityByCycle -o MQ_METRICS.PDF MQ_METRICS.TXT
```

Table 12.35: Argument correspondence - MeanQualityByCycle and MeanQualityByCycle

Picard option	Sentieon option	Meaning
I=DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
O=MQ_METRICS.TXT	N/A	MQ metrics results
R=REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
CHART=MQ_METRICS.PDF	-o MQ_METRICS.PDF	Mean quality metrics report

12.1.2.22 QualityScoreDistribution - QualDistribution

Picard QualityScoreDistribution command line

```
java -jar picard.jar QualityScoreDistribution \
   I=DEDUPED.BAM O=QD_METRICS.TXT \
   CHART=QD_METRICS.PDF
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo QualDistribution \
   QD_METRICS.TXT
sentieon plot QualDistribution -o QD_METRICS.PDF QD_METRICS.TXT
```

Table 12.36: Argument correspondence - QualDistribution and QualityScoreDistribution

Picard option	Sentieon option	Meaning
I=DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
O=QD_METRICS.TXT	N/A	QD metrics results
N/A	-r REFERENCE.FASTA	Reference file
CHART=QD_METRICS.PDF	-o QD_METRICS.PDF	Quality distribution metrics report

12.1.2.23 CollectQualityYieldMetrics - QualityYield

Picard CollectQualityYieldMetrics command line

```
java -jar picard.jar CollectQualityYieldMetrics \
   I=DEDUPED.BAM O=YIELD_METRICS.TXT
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo QualityYield \
   YIELD_METRICS.TXT
```

Table 12.37: Argument correspondence - QualityYield and CollectQualityYieldMetrics

Picard option	Sentieon option	Meaning
I=DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
O=YIELD_METRICS.TXT	N/A	Quality yield metrics results
INCLUDE_SECONDARY_ALIGNMENT	–include_supplementary	Include supplementary alignments in the calculation
INCLUDE_SUPPLEMENTAL_ALIGNMI	–include_secondary	Include secondary alignments in the calculation

12.1.2.24 CollectSequencingArtifactMetrics - SequenceArtifactMetricsAlgo

Picard CollectSequencingArtifactMetrics command line

```
java -jar picard.jar CollectSequencingArtifactMetrics \
   I=DEDUPED.BAM O=ARTIFACT_METRICS_BASE R=REFERENCE.FASTA \
   DB_SNP=DBSNP.VCF
java -jar picard.jar ConvertSequencingArtifactToOxoG \
   I=DEDUPED.BAM O=ARTIFACT_METRICS_BASE R=REFERENCE.FASTA \
   OUTPUT_BASE=oxog_metrics
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo SequenceArtifactMetricsAlgo \
   --dbsnp DBSNP.VCF ARTIFACT_METRICS_BASE
```

Table 12.38: Argument correspondence - SequenceArtifactMetricsAlgo and CollectSequencingArtifactMetrics

Picard option	Sentieon option	Meaning
I=DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
O=ARTIFACT_METRICS_BASE	N/A	Artifact metrics output base
R=REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
DB_SNP=DBSNP.VCF	–dbsnp DBSNP.VCF	A dbSNP file to exclude known polymorphisms
Q=MIN_BQ	–min_base_qual MIN_BQ	Minimum base quality for a base to be included
MQ=MIN_MAPQ	–min_map_qual MIN_MAPQ	Minimum mapping quality for a read to be included
MIN_INS=MIN_INSERT	–min_insert_size MIN_INSERT	Minimum insert size to include a read
MAX_INS=MAX_INSERT	–max_insert_size MAX_INSERT	Maximum insert size to include a read
UNPAIRED=true	–include_unpaired	Include unpaired reads
TANDEM=true	–tandem_reads	Include tandem reads
INCLUDE_DUPLICATES=true	–include_duplicates	Include duplicate reads
INCLUDE_NON_PF_READS=true	–include_non_pf_reads	Include non-PF reads
CONTEXT_SIZE=CONTEXT	–context_size CONTEXT	The number of context bases to include on each size

12.1.2.25 CollectWgsMetrics - WgsMetricsAlgo

Picard CollectWgsMetrics command line

```
java -jar picard.jar CollectWgsMetrics \
   I=DEDUPED.BAM O=WGS_METRICS.TXT R=REFERENCE.FASTA
```

Sentieon® command line

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i DEDUPED.BAM --algo WgsMetricsAlgo \
```

(continues on next page)

(continued from previous page)

```
WGS_METRICS.TXT
```

Table 12.39: Argument correspondence - WgsMetricsAlgo and CollectWgsMetrics

Picard option	Sentieon option	Meaning
I=DEDUPED.BAM	-i DEDUPED.BAM	Input the bam files
O=WGS_METRICS.TXT	N/A	WGS metrics results
R=REFERENCE.FASTA	-r REFERENCE.FASTA	Reference file
MQ=MIN_MAPQ	–min_map_qual MIN_MAPQ	Minimum mapping quality for a read to be included
Q=MIN_BQ	–min_base_qual MIN_BQ	Minimum base quality for a base to be included
CAP=COVERAGE_CAP	–coverage_cap COVERAGE_CAP	Maximum coverage limit for the histogram
COUNT_UNPAIRED=true	–include_unpaired true	Count unpaired reads and paired reads with one end unmapped
INCLUDE_BQ_HISTOGRAM=true	–base_qual_histogram true	Report a base quality histogram
SAMPLE_SIZE=SAMPLE_SIZE	–sample_size SAMPLE_SIZE	Sample size used for theoretical het sensitivity sampling

12.1.3 Other differences in usage

Sentieon® refers to tools as algorithms, so the option `-T` in GATK3 corresponds to the option `--algo` in Sentieon® .

Sentieon® produces log files directly to stdout and stderr, so the option `-log` is not available.

Sentieon® tries to use as many threads as the system has available, while GATK uses 1 thread by default. As such omitting option `-nt` in GATK, is not the same as omitting the option `-t` in Sentieon® .

Sentieon® does not do any down-sampling, so the following options are not available: `--downsample_to_coverage`, `--downsample_to_fraction`, `--downsampling_type`, ...

Other general level arguments that are currently supported by Sentieon are:

- `--bam_compression`: for algorithms that output a bam
- `--cram_write_options`: for algorithms that output a cram

12.2 Deployment Guide for Amazon Web Services

12.2.1 Introduction

12.2.1.1 Overview of the software

The Sentieon® software provides fast and efficient data processing for genomic data. Supported applications include secondary analysis of sequence data from whole-genome, whole-exome, targeted, and RNA sequence data for alignment, preprocessing, germline and somatic variant calling.

The Sentieon® Genomics software enables rapid and accurate analysis of next-generation sequence data. The Sentieon® DNAseq pipelines enable germline variant calling on hundreds of thousands of samples

simultaneously. The Sentieon® TNseq pipelines enable accurate calling of somatic variants in paired tumor-normal samples or in an unpaired tumor samples. The Sentieon® Genomics software produces more accurate results than other tools on third-party benchmarks with results obtained in one tenth the time of comparable pipelines.

12.2.1.2 Platform requirements

Sentieon® Genomics software is designed to run on Linux and other POSIX-compatible platforms.

For Linux systems, we recommended using the following Linux distributions or higher: RedHat/CentOS 6.5, Debian 7.7, OpenSUSE-13.2, or Ubuntu-14.04.

Sentieon® Genomics software requires at least 16GB of memory, although we recommend using 64 GB of memory. The alignment step is typically the most memory consuming stage in most pipelines.

Sentieon® Genomics software processes file data at speeds around 20-30MB/s per core using a state of the art server. In order to take advantage of the maximum speed that the software can provide, we recommend that you use high-speed SSD hard drives in your system, preferably two identical drives in a high speed RAID 0 striped volume configuration. When using a RAID 0 configuration, it is advised to use the drives to only store intermediate files, and move out any final results or important information out of those hard drives; this way, your server should have additional storage space to store the results.

The required hard disk space is pipeline and data specific and can vary significantly between use-cases. A general rule of thumb is that the software can use approximately 3x the input file size for intermediate file storage during data processing.

12.2.2 Multi-Availability Zone Deployment on AWS

The Sentieon® software is controlled by a license and requires an active Sentieon® license server to run. Deployment on AWS requires setup and configuration of the Sentieon® license server along with setup and configuration of one or more instances for genomic data processing. A minimum deployment will provision the following resources inside your default VPC and should take approximately ten minutes:

- Security groups that can be used to run the Sentieon® license server and the compute nodes.
- A CloudWatch log group for the license server logs.
- An IAM role and instance profile for the license server.
- A (t3.nano) EC2 instance running the Sentieon® license server.
- A private hosted zone in AWS Route53.

The Sentieon® software can be deployed onto additional instances for additional data processing capacity.

The provisioned IAM role will grant the license server instance read access to the Sentieon software package and to your Sentieon license file in AWS s3. It will also grant write access to the CloudWatch log group to record the license server logs in CloudWatch.

The root EBS disk for the license server instance will be encrypted. By default, the encryption will use an AWS-managed encryption key. Optionally, a KMS key can be specified to use a customer-managed encryption key.

With a minimum deployment, customers will be charged by AWS EC2 for the instance running the license server and for outbound communication from the Sentieon® license server to the Sentieon® master license server. Customers will also be charged for the Rout53 hosted zone and for DNS queries resolved by Route53. Estimated AWS costs for the minimal license server deployment with a c6a.2xlarge instance for compute at 60% utilization is $134.03 per month in On-Demand costs[45].

[45] https://calculator.aws/#/estimate?id=099986a1d2f73bf694c8f74c08dc022299844ebd

Adding additional compute instances to the deployment will increase the AWS EC2. Data processing may incur additional costs for data transfer, etc. The Sentieon® software is a proprietary software and this deployment requires a license subscription for the Sentieon® software.

This deployment is supported in the following AWS Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Africa (Cape Town)
- Asia Pacific (Hong Kong)
- Asia Pacific (Hyderabad)
- Asia Pacific (Jakarta)
- Asia Pacific (Melbourne)
- Asia Pacific (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Milan)
- Europe (Paris)
- Europe (Spain)
- Europe (Stockholm)
- Europe (Zurich)
- Middle East (Bahrain)
- Middle East (UAE)
- South America (São Paulo)

This deployment assumes basic familiarity with the VPC and EC2 services on AWS and familiarity with the Linux command-line interface.

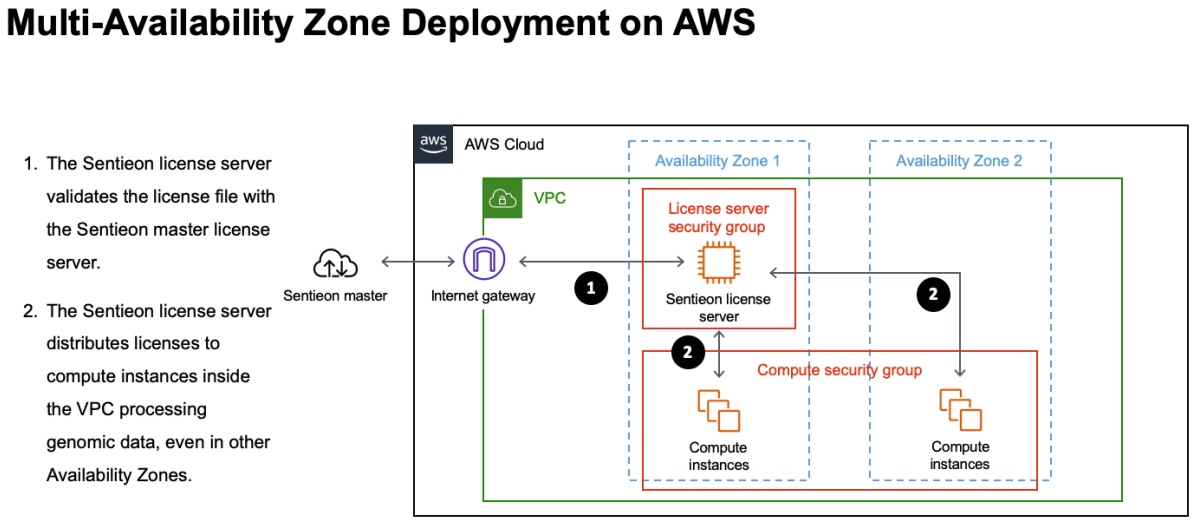


Fig. 12.1: Overview of the Sentieon deployment architecture on AWS

12.2.2.1 Prerequisites

Terraform[46] is an open-source infrastructure as code (IaC) tool for the provisioning and management of cloud infrastructure. This guide uses Terraform to automate the provisioning of the required AWS infrastructure.

Terraform configuration files used in this deployment can be found on GitHub at, https://github.com/Sentieon/terraform.

The following are prerequisites for the deployment:

- The Terraform CLI[47].
- The AWS CLI[48].
- An AWS account and IAM credentials with permission to provision the required resources.

12.2.2.2 Deployment of the Sentieon® license server

In order to use the software on AWS, you will need to follow the following installation instructions to deploy a Sentieon® License server to your VPC.

1. Choose an AWS Region to work in. Choose a fully qualified domain name (FQDN) to associate with the license server. This might be something like `licsrvr.sentieon.example.com`.
2. Send your Sentieon® support representative the FQDN. Sentieon will send back a license file that can be used to start a Sentieon license server at the FQDN. Move the license file into an s3 bucket in the AWS region.
3. Use the following commands to download the Terraform configuration files to your local machine, configure your AWS credentials and initialize the directory with Terraform.

[46] https://www.terraform.io/
[47] https://developer.hashicorp.com/terraform/downloads
[48] https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html

```
# Download the configuration files
git clone https://github.com/sentieon/terraform
cd terraform/aws_license-server

# Configure your AWS credentials
export AWS_ACCESS_KEY_ID=<AWS_ACCESS_KEY_ID>
export AWS_SECRET_ACCESS_KEY=<AWS_SECRET_ACCESS_KEY>
```

4. Use Terraform to provision the infrastructure and start the Sentieon license server. Use the `-var` arguments to pass the selected AWS region and FQDN along with the URI of your Sentieon® license file in s3 to Terraform. After the initial `terraform apply` command, Terraform will print the proposed infrastructure plan and ask to confirm the proposed changes. Replying `yes` will cause Terraform to provision the infrastructure.

   ```
   # Provision the license server infrastructure
   terraform apply \
     -var 'aws_region=<AWS_REGION>' \
     -var 'licsrvr_fqdn=<FQDN>' \
     -var 'license_s3_uri=s3://<S3_URI>'
   ```

   The command should provision the infrastructure and start the Sentieon license server in under five minutes.

After the license server has been deployed, the license server can be monitored using the log files written to the CloudWatch log group, `/sentieon/licsrvr/LicsrvrLog`.

The infrastructure deployment can be destroyed by running `terraform apply` with the `-destroy` argument:

```
# Destroy the provisioned infrastructure
terraform apply \
  -destroy \
  -var 'aws_region=<AWS_REGION>' \
  -var 'licsrvr_fqdn=<FQDN>' \
  -var 'license_s3_uri=s3://<S3_URI>'
```

12.2.2.3 Deployment of one or more instances for genomic data processing

After the license server is deployed in your VPC, you can deploy the software to one or more additional instances for genomic data processing. These instances can be deployed to any Availability Zone in the VPC. The following steps provide guidance for deploying the Sentieon® software onto newly launched EC2 instances:

1. Start an EC2 instance that meets the platform requirements for the Sentieon® software inside your VPC. Assign the instance to the `sentieon_compute` security group created by Terraform.
2. Download the Sentieon® tools to the newly launched instance.
3. Set an environment variable in your system to indicate the location of the license server.

   ```
   export SENTIEON_LICENSE=<FQDN>:8990
   ```

The new instance is now ready to process data and data processing does not require root privileges. Please see the *Sentieon Software Manual* for more information on the functionality included in the Sentieon® software.

12.2.2.4 Upgrading the deployment and restarting the license server

Most new releases of the Sentieon® software will not affect the function of the license server. In the event of a software update affecting the license server, the following steps can be used to update the license server.

1. In the `main.tf` file, update the `sentieon_version` variable to the desired version.
2. Re-run the `terraform apply` command to terminate the running license server instance and start a new instance using the updated Sentieon software version.

```
# Update the license server
terraform apply \
  -var 'aws_region=<AWS_REGION>' \
  -var 'licsrvr_fqdn=<FQDN>' \
  -var 'license_s3_uri=s3://<S3_URI>'
```

In the event that the license server becomes unreachable is otherwise non-functional, you can recover the license server by destroying current instance and starting new instance. The instance can be destroyed with `terraform destroy`:

```
# Destroy the license server instance
terraform destroy \
  -target aws_instance.sentieon_licsrvr \
  -var 'aws_region=<AWS_REGION>' \
  -var 'licsrvr_fqdn=<FQDN>' \
  -var 'license_s3_uri=s3://<S3_URI>'
```

The license server can then be restarted with `terraform apply`:

```
# Restart the license server
terraform apply \
  -var 'aws_region=<AWS_REGION>' \
  -var 'licsrvr_fqdn=<FQDN>' \
  -var 'license_s3_uri=s3://<S3_URI>'
```

12.2.3 Troubleshooting

If you encounter issues with the software, please refer to the *Sentieon Software Manual* and *Quick Start Guide*. You may also contact the Sentieon® support team at support@sentieon.com.

12.3 Deployment Guide for Azure

12.3.1 Multi-Availability Zone Deployment on Azure

The Sentieon® software is controlled by a license and requires an active Sentieon® license server to run. Deployment on Azure requires setup and configuration of the Sentieon® license server along with setup and configuration of one or more instances for genomic data processing. A minimum deployment will use the following resources and should take approximately 20 minutes:

- An Azure Virtual Network. The default network configuration with the default subnet(s), internet gateway, and route table is assumed in this deployment guide.
- Security groups that can be used to run the Sentieon® license server and the compute nodes.
- A (Standard_B1s) instance running the Sentieon® license server.

The Sentieon® software can be deployed onto additional instances for additional data processing capacity.

With a minimum deployment, customers will be charged by Azure for the instance running the license server and for outbound communication from the Sentieon® license server to the Sentieon® master license server. Adding additional comptute instances to the deployment will increase costs by the price of those instances. Data processing may incur additional costs for data transfer, etc. The Sentieon® software is a proprietary software and this deployment requires a license for the Sentieon® software.

This deployment is supported in the following Azure Regions:

- (Africa) South Africa North
- (Africa) South Africa West
- (Asia Pacific) Australia Central
- (Asia Pacific) Australia Central 2
- (Asia Pacific) Australia East
- (Asia Pacific) Australia Southeast
- (Asia Pacific) Central India
- (Asia Pacific) East Asia
- (Asia Pacific) Japan East
- (Asia Pacific) Japan West
- (Asia Pacific) Jio India Central
- (Asia Pacific) Jio India West
- (Asia Pacific) Korea Central
- (Asia Pacific) Korea South
- (Asia Pacific) South India
- (Asia Pacific) Southeast Asia
- (Asia Pacific) West India
- (Canada) Canada Central
- (Canada) Canada East
- (Europe) France Central
- (Europe) France South
- (Europe) Germany North
- (Europe) Germany West Central
- (Europe) North Europe
- (Europe) Norway East
- (Europe) Norway West
- (Europe) Sweden Central
- (Europe) Switzerland North
- (Europe) Switzerland West
- (Europe) UK South
- (Europe) UK West

- (Europe) West Europe
- (Middle East) Qatar Central
- (Middle East) UAE Central
- (Middle East) UAE North
- (South America) Brazil South
- (South America) Brazil Southeast
- (US) Central US
- (US) Central US EUAP
- (US) East US
- (US) East US 2
- (US) East US 2 EUAP
- (US) East US STG
- (US) North Central US
- (US) South Central US
- (US) West Central US
- (US) West US
- (US) West US 2
- (US) West US 3

This deployment assumes basic familiarity with the Virtual Network and VM services on Azure and familiarity with the Linux command-line interface.

12.3.1.1 Deployment of the Sentieon® license server

In order to use the software on Azure, you will need to follow the following installation instructions to deploy a Sentieon® License server to your Virtual Network.

1. Choose an Azure Region to work in. Choose (or create) the Virtual Network where you will run the Sentieon® tools.
2. Find and record the CIRD block for your Virtual Network.
3. Launch a Standard_B1s instance to run the license server in the selected Virtual Network.
4. Update the security group for the license server (Fig. 12.2 and Fig. 12.3). The security group must:

- Allow inbound TCP communication at a specific port (we use 8990 by default as it is not typically used by other applications). We highly recommend that users choose a port above 1024 so the license server can be run as a non-root user. This rule is used to accept inbound communication from the compute nodes to the license server. You should open TCP at the desired port across your Virtual Network's CIDR block (172.31.0.0/16 in Fig. 12.2) to whitelist traffic from within your network.
- Allow outbound HTTPS communication to Sentieon® license master at master.sentieon.com (IP 52.89.132.242). This is necessary for license validation.
- Allow inbound SSH. This is necessary for administration.
- (Recommended) Allow ICMP communication. This allows for PMTU discovery.

5. Send your Sentieon® support representative the Private IP address of your instance, together with the TCP port you opened in step 4.

6. Download the Sentieon® tools and your license file to the instance.
7. On the Standard_B1s instance, start the license server with the following command. Note that the license server does not need to be started as a root user if a port above 1024 is chosen in step 4.

```
sentieon licsrvr --start [-l <licsrvr_log>] <license_file>
```

8. (Optional) Confirm the license server is working correctly and serving licenses to the license server instance by running following commands. The second command will return the number of available licenses.

```
sentieon licclnt ping -s <IP_addresss>:<PORT> || (echo "Ping Failed"; exit 1)
sentieon licclnt query -s <IP_address>:<PORT> klib
```

9. Launch a Standard_B1s instance within the same Virtual Network to test the license server.
10. Update the security group for the second Virtual Machine (Fig. 12.4 and Fig. 12.5). The security group should:

- Allow inbound SSH. This is necessary for administration.
- Allow outbound TCP communication at the port chosen in step 3, which will be open across your Virtual Network's CIDR block.
- (Recommended) Allow ICMP communication to facilitate communication between the license server and compute nodes.

11. Download the Sentieon® software to the newly launched instance and confirm the license server is working and serving licenses within the Virtual Network by running the following commands on the instance. The second command will return the number of available licenses.

```
sentieon licclnt ping -s <IP_addresss>:<PORT> || (echo "Ping Failed"; exit 1)
sentieon licclnt query -s <IP_address>:<PORT> klib
```

Inbound port rules | Outbound port rules | Application security groups | Load balancing

Network security group sentieonLicProxy-nsg (attached to subnet: default)
Impacts 1 subnets, 0 network interfaces

Add inbound port rule

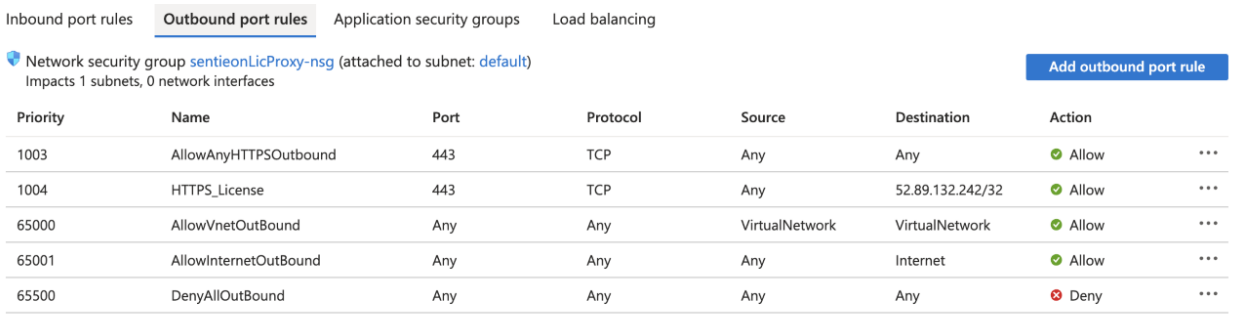
Priority	Name	Port	Protocol	Source	Destination	Action
1001	SSH	22	TCP	Any	Any	Allow
1002	Port_8990	8990	TCP	Any	10.4.0.0/24	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

Fig. 12.2: Example license server inbound security group rules

After the license server has been deployed, the license server can be monitored by running the `sentieon licclnt ping` and `sentieon licclnt query` commands on a separate instance, as described in steps 10 and 11, above.

12.3.1.2 Deployment of one or more instances for genomic data processing

After the license server is deployed in your Virtual Network, you can deploy the software to one or more additional instances for genomic data processing. These instances can be deployed to any Availability Zone in the Virtual Network. The following steps provide guidance for deploying the Sentieon® software onto newly launched Azure Virtual Machines:

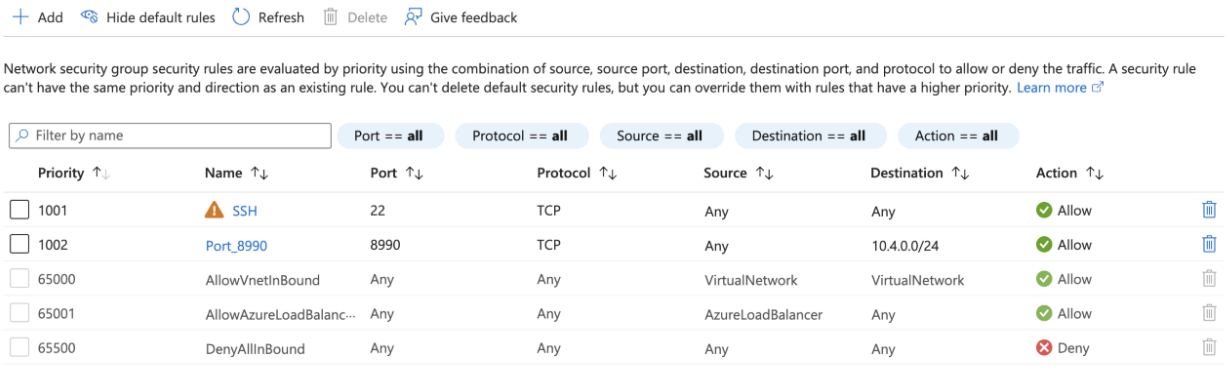

Inbound port rules | Outbound port rules | Application security groups | Load balancing

Network security group sentieonLicProxy-nsg (attached to subnet: default)
Impacts 1 subnets, 0 network interfaces

Add outbound port rule

Priority	Name	Port	Protocol	Source	Destination	Action
1003	AllowAnyHTTPSOutbound	443	TCP	Any	Any	Allow
1004	HTTPS_License	443	TCP	Any	52.89.132.242/32	Allow
65000	AllowVnetOutBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowInternetOutBound	Any	Any	Any	Internet	Allow
65500	DenyAllOutBound	Any	Any	Any	Any	Deny

Fig. 12.3: Example license server outbound security group rules

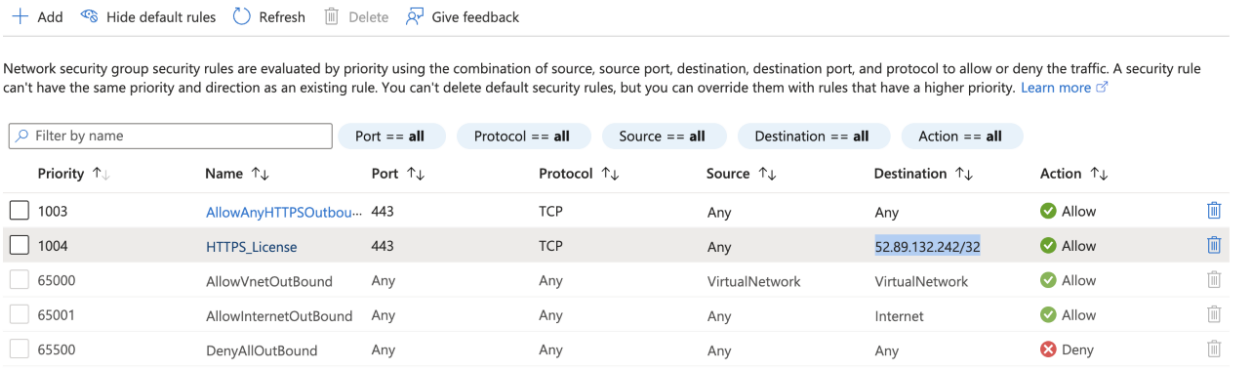

Add | Hide default rules | Refresh | Delete | Give feedback

Network security group security rules are evaluated by priority using the combination of source, source port, destination, destination port, and protocol to allow or deny the traffic. A security rule can't have the same priority and direction as an existing rule. You can't delete default security rules, but you can override them with rules that have a higher priority. Learn more

Filter by name | Port == all | Protocol == all | Source == all | Destination == all | Action == all

Priority	Name	Port	Protocol	Source	Destination	Action
1001	SSH	22	TCP	Any	Any	Allow
1002	Port_8990	8990	TCP	Any	10.4.0.0/24	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalanc...	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

Fig. 12.4: Example compute nodes inbound security group rule


Add | Hide default rules | Refresh | Delete | Give feedback

Network security group security rules are evaluated by priority using the combination of source, source port, destination, destination port, and protocol to allow or deny the traffic. A security rule can't have the same priority and direction as an existing rule. You can't delete default security rules, but you can override them with rules that have a higher priority. Learn more

Filter by name | Port == all | Protocol == all | Source == all | Destination == all | Action == all

Priority	Name	Port	Protocol	Source	Destination	Action
1003	AllowAnyHTTPSOutbou...	443	TCP	Any	Any	Allow
1004	HTTPS_License	443	TCP	Any	52.89.132.242/32	Allow
65000	AllowVnetOutBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowInternetOutBound	Any	Any	Any	Internet	Allow
65500	DenyAllOutBound	Any	Any	Any	Any	Deny

Fig. 12.5: Example compute nodes outbound security group rules

1. Start an Azure Virtual Machine that meets the platform requirements for the Sentieon® software inside your Virtual Network. Assign the instance to the security group created in *step 10* in the above section on deployment of the license server.
2. Download the Sentieon® tools to the newly launched instance.
3. Set an environment variable in your system to indicate the location of the license server.

```
export SENTIEON_LICENSE=<LICSRVR_INTERNAL_IP>:<LICSRVR_PORT>
```

The new instance is now ready to process data and data processing does not require root privileges. Please see the *Sentieon Software Manual* for more information on the functionality included in the Sentieon® software.

12.4 Germline Copy Number Variant Calling for Whole-Genome-Sequencing with CNVscope

12.4.1 Introduction

This document describes the capabilities of CNVscope for germline copy number variation (CNV) calling for whole-genome sequencing (WGS). If you have any additional questions, please contact the technical support at Sentieon® Inc. at support@sentieon.com.

12.4.2 Germline CNV calling with CNVscope

12.4.2.1 Basic usage of CNVscope

Two individual commands are run to call CNV and to apply the machine learning model. The input BAM file should come from a pipeline where alignment and deduplication have been performed.

```
sentieon driver [--interval INTERVAL_FILE] -t NUMBER_THREADS \
  -r REFERENCE -i DEDUPED_BAM --algo CNVscope \
  --model ML_MODEL/cnv.model TMP_VARIANT_VCF

sentieon driver -t NUMBER_THREADS -r REFERENCE --algo CNVModelApply \
  --model ML_MODEL/cnv.model -v TMP_VARIANT_VCF VARIANT_VCF
```

Reminder

It is important to use the same model for `CNVscope` and `CNVModelApply`. If different models are used, `CNVModelApply` will give an error.

The following inputs are required for the command:

- **NUMBER_THREADS**: the number of computer threads that will be used in the calculation. We recommend that the number does not exceed the number of computing cores available in your system.
- **REFERENCE**: the location of the reference FASTA file. You should make sure that the reference is the same as the one used in the mapping stage.
- **DEDUPED_BAM**: the location of the input BAM file.
- **TMP_VARIANT_VCF**: the location and filename of the variant calling output of CNVscope. This is a temporary file.

- **VARIANT_VCF**: the location and filename of the variant calling output. A corresponding index file will be created. The tool will output a compressed file by using `.gz` extension.
- **ML_MODEL**: the location of the machine learning model file. In the CNVscope command the model will be used to determine the settings used in variant calling.

The following inputs are optional for the command:

- **INTERVAL_BED**: the location of a BED file containing intervals for variant calling.

The final output VCF file uses `CN` annotation for the copy-number state for each region called by CNVscope machine-learning model. Possible copy-number states called by CNVscope are from 0 to 4, with `CN=4` representing copy number states equal to or larger than 4.

12.4.2.2 Using CNVscope with male samples

CNVscope will perform diploid variant calling across the whole genome or the genomic regions in the supplied `--interval` file. In human male samples, variant calling should be performed separately across diploid and haploid chromosomes. This can be accomplished by running `CNVscope` and `CNVModelApply` twice, once across the diploid autosomes and once across the haploid X and Y chromosomes using the `--interval` argument.

```
 sentieon driver --interval AUTOSOMES_BED -t NUMBER_THREADS \
   -r REFERENCE -i DEDUPED_BAM --algo CNVscope \
   --model ML_MODEL/cnv.model TMP_DIPLOID_VCF

 sentieon driver --interval  -t NUMBER_THREADS \
   -r REFERENCE --algo CNVModelApply --model ML_MODEL/cnv.model \
   -v TMP_DIPLOID_VCF DIPLOID_VCF

sentieon driver --interval HAPLOID_BED -t NUMBER_THREADS \
   -r REFERENCE -i DEDUPED_BAM --algo CNVscope \
   --model ML_MODEL/cnv.model TMP_HAPLOID_VCF

 sentieon driver --interval  -t NUMBER_THREADS \
   -r REFERENCE --algo CNVModelApply --model ML_MODEL/cnv.model \
   -v TMP_HAPLOID_VCF HAPLOID_VCF
```

The following inputs are required for the command:

- **AUTOSOMES_BED**: the location of a BED file containing the diploid autosomes.
- **HAPLOID_BED**: the location of a BED file containing the haploid chromosomes.

After generating VCFs containing haploid and diploid calls, the two VCF files can be combined using bcftools.

```
bcftools bcftools -aD DIPLOID_VCF HAPLOID_VCF | \
  sentieon util vcfconvert - VARIANT_VCF
```

12.4.2.3 Limitations of CNVscope machine learning model

Currently, CNVscope is trained on diploid WGS samples with CNV event size above 5000 base pairs and it should not be used to identify smaller CNVs. When performing diploid variant calling across haploid samples, intermediate copy-number states (`CN=1` and `CN=3`) are undefined.

12.5 Distributed Mode

12.5.1 Introduction

This document describes how to use the sharding capabilities of the Sentieon® Genomics tools to distribute a DNAseq® pipeline onto multiple servers; distribution of other pipelines such as a TNseq® one follow the same principles, as all Sentieon® genomcis tools have the same built in distribution processing capabilities. The goal of such distribution is to reduce the overall runtime of the pipeline to produce results quicker; however such distribution has some overhead that makes the computation cost higher.

Using the distribution capabilities, each stage of the pipeline is divided into small tasks; each of these tasks processes part of the genome, and can be run in different servers in parallel. Each of these tasks produces a partial result that needs to be merged sequentially into a final single output; such merging needs to be carefully done to make sure it considers the boundary and produces the same results as a pipeline run without sharding.

The execution framework for the distribution is not in the scope of this document, and the user needs to distribute the data/files and launch the correct processes while keeping the correct data dependency among the stages.

12.5.2 Shard and sharding

We divide the genome into many sequential non-overlapping parts, with each part called a shard. Each shard is defined as either a single chromosome contig name, or part of a chromosome following the convention contig:shard_start-shard_end. The special shard NO_COOR is used for all unmapped reads that do not have coordinate.

The Sentieon® binaries support sharding for distribution into multiple servers, and can process multiple shards in a single command by adding one or multiple shard options with the `--shard SHARD` argument. When using multiple `--shard` options in a single command line, these shards need to be contiguous according to the reference contig list; for example, a command can have one shard covering the end of chr2 and one shard covering the beginning of chr3, but it should not have a shard covering the end of chr2 and the beginning of chr22.

You can refer to the `generate_shards.sh` script in the appendix for a sample file that creates shards for the genome based on either the dictionary associated with the reference or the bam header in the input bam files. We recommend using 100 M bases as the shard size.

12.5.3 Data flow and data/file dependency for distribution

A pipeline for DNAseq® following the recommended workflow processes a pair of fastq files through the following stages: BWA alignment to produce a sorted.bam, Deduplication to produce a dedup.bam, BQSR to produce a recal.table and Haplotyper to produce an output.vcf.gz file. Fig. 12.6 illustrates the data flow for such a pipeline.

To distribute the above pipeline onto multiple servers, each of the stages is divided into commands that process the data on a shard, requiring inputs from both the specific shard as well as the right and left neighboring shards; the exceptions to this are the Dedup stage, which requires ALL the score files from the LocusCollector commands on all shards, and the Haplotyper stages, which requires a complete merged recalibration table.

As an example, Fig. 12.7 illustrates the data flow for a pipeline distributed as 4 shards; this is not a typical use case as using the recommended shard size of 100M bases will result in requiring more than 30 shards. In the example of Fig. 12.7, the stages require the following inputs and produce the following outputs:

- The LocusCollector stage (dedup pass 1) for the `i-th` shard requires the sorted.bam input. The stage produces a `part_deduped.bam$shard_i.score.gz` file.

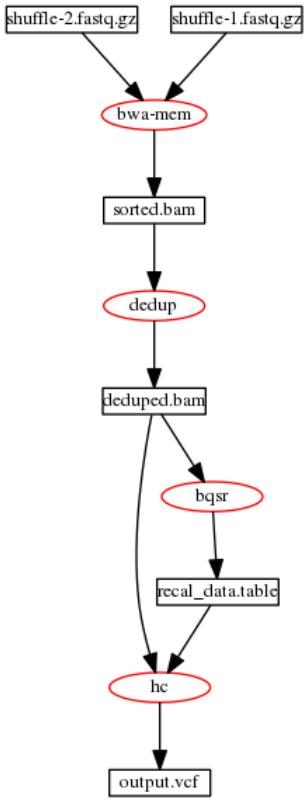


Fig. 12.6: Typical data flow for DNAseq® pipeline

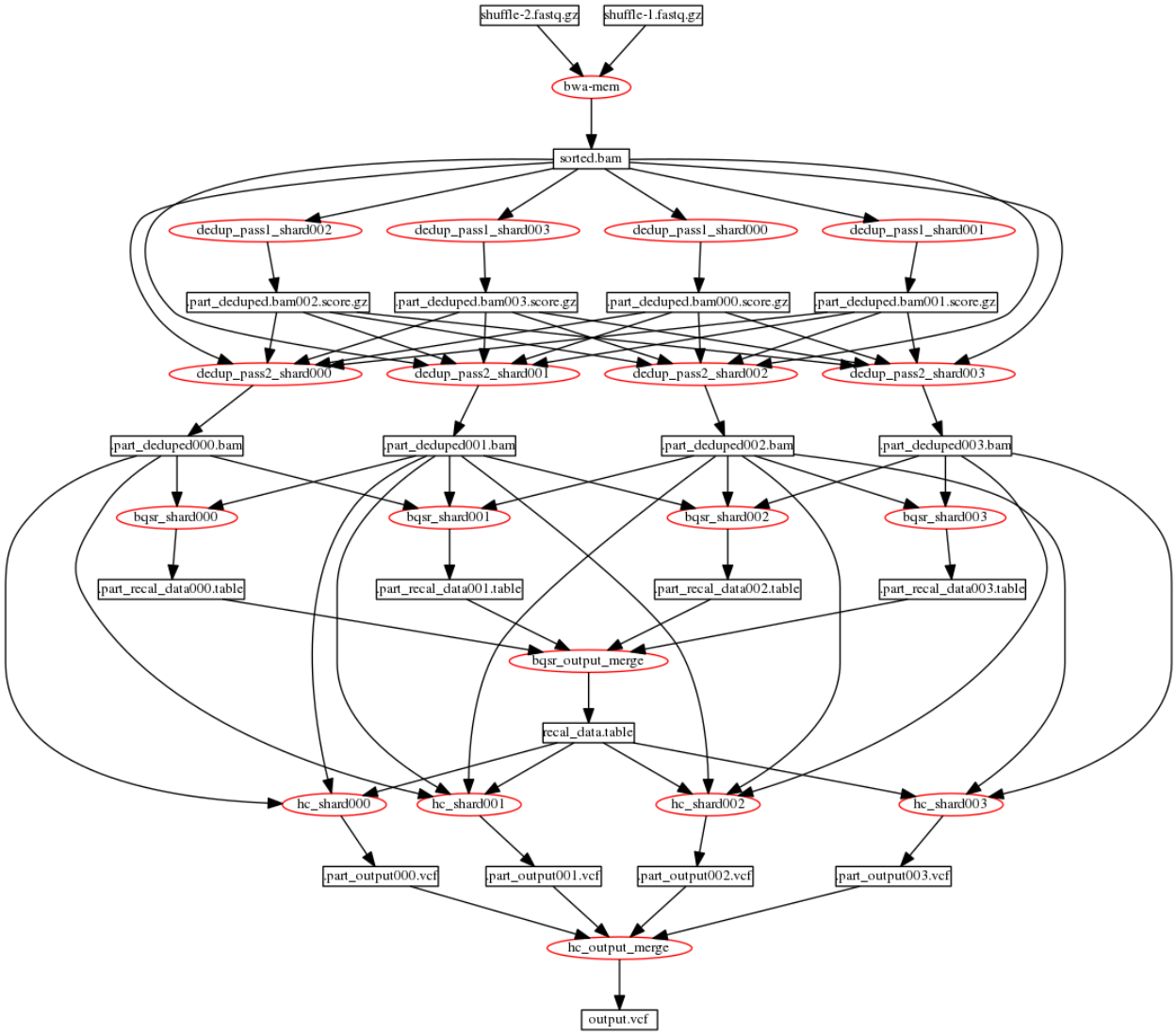


Fig. 12.7: Data flow for DNAseq® pipeline distributed as 4 shards without processing unmapped reads

- The Dedup stage (dedup pass 2) for the `i-th` shard requires the sorted.bam input as well as all the results from all the LocusCollector stages. The stage produces a `part_deduped$shard_i.bam` file.
- The QualCal stage for the `i-th` shard requires the `part_deduped$shard_i.bam` file as well as the `part_deduped$shard_i+1.bam` file and `part_deduped$shard_i-1.bam` file if available. The stage produces a `part_recal_data$shard_i.table` file.
- After QualCal, all `part_recal_data$shard_i.table` files need to be merged into a single calibration table file, to be used in variant calling. The options `--passthru` in the driver and `--merge` of QualCal can be used to perform the boundary aware merging.
- The Haplotyper stage for the `i-th` shard requires the `part_deduped$shard_i.bam` file as well as the `part_deduped$shard_i+1.bam` file and `part_deduped$shard_i-1.bam` file if available; additionally, the complete merged recalibration table needs to be an input. The stage produces a `part_output$shard_i.vcf.gz` file
- After Haplotyper, all `part_output$shard_i.vcf.gz` files need to be merged into a single output VCF file. The options `--passthru` in the driver and `--merge` of Haplotyper are used to perform the boundary aware merging.
- If at any stage in the process you require to have the merged output bam file, you can use the util binary to perform the boundary aware merging; you need to add the option `--mergemode=10` in the command so that util merge does not process the reads and only copies them by block.

Important

When merging the results, be careful to input the partial results sequentially in the correct order, as otherwise the results will be incorrect.

12.5.4 Distributing BWA

The above instructions do not include any information about distributing the alignment using BWA. In order to distribute the BWA alignment, you can use the `fqidx` tool contained in the Sentieon® tools to create index files for the input FASTQ files; then you can extract certain parts of the FASTQ files for processing on different servers, using the results of the fqidx command as the input to BWA mem; you will need to make sure that the option `-p` is included in the BWA command, as the output of the fqidx contains interleaved reads in a single output. The results of this flow are identical to executing on a single run.

```
BWA_K_size=100000000
num_splits=10
SAMPLE="sample_name" #Sample name SM tag in bam
GROUP="read_group_name" #Read Group name RGID tag in bam
PLATFORM="ILLUMINA"
NT=$(nproc) #number of threads to use in computation, set to all threads available

FASTA="/home/regression/references/b37/human_g1k_v37_decoy.fasta"
FASTQ_1="WGS-30x_1.fastq.gz"
FASTQ_2="WGS-30x_2.fastq.gz"
FASTQ_INDEX="WGS-30x.fastq.gz.index"

#create FASTQ indices
sentieon fqidx index -K $BWA_K_size -o $FASTQ_INDEX $FASTQ_1 $FASTQ_2

#get the number of runs that the inputs will be split into given the size
num_K=$(sentieon fqidx query -i $FASTQ_INDEX | cut -d' ' -f1)
```

(continues on next page)

(continued from previous page)

```
BWA_K_size=$(sentieon fqidx query -i $FASTQ_INDEX | cut -d' ' -f2)
num_K_splits=$(expr $num_K / $num_splits + 1)
#run multiple BWA on multiple servers
file_list=""
for run in $(seq 0 $((num_splits-1))); do
        region="$((run*num_K_splits))-$((run*num_K_splits+num_K_splits))"
        #send each of these to a different server
        sentieon bwa mem -R "@RG\tID:$GROUP\tSM:$SAMPLE\tPL:$PLATFORM" \
          -K $BWA_K_size -t $NT -p $FASTA \
          "<sentieon fqidx extract -i $FASTQ_INDEX -r $region $FASTQ_1 $FASTQ_2" | \
          sentieon util sort -r $FASTA -t $NT --sam2bam -o sorted_run$run.bam -i -
        file_list="$file_list sorted_run$run.bam"
done

#merge the results
sentieon util merge -o sorted.bam $file_list
#or perform deduplication on all the intermediate BAM files
file_list_bam=""
for file in $file_list; do file_list_bam="$file_list_bam -i $file"; done
sentieon driver -r $FASTA -t $NT $file_list --algo LocusCollector --fun score_info score.txt.
↪gz
sentieon driver -r $FASTA -t $NT $file_list --algo Dedup --score_info score.txt.gz deduped.
↪bam
```

12.5.4.1 Distributing BWA when you cannot create FASTQ index file (version 201808.02+)

In the event that it is not feasible to create the FASTQ index files, you can use util fqidx with the fractional option `-F m/n` while also using the `-K` option. This divides the input FASTQs in splits of `n` chuncks of reads and then extracts every `m-th` element from each split to be processed in a different server, with `m` going from `0` to `n-1`.

Bear in mind that this feature is only recommended in an IT environment where the file storage is fast enough to support each fqidx process reading the input FASTQ files at the same time, which is common in cloud environments, or in local cluster environments with a high bandwidth NFS system.

```
BWA_K_size=100000000
num_splits=10
SAMPLE="sample_name" #Sample name SM tag in bam
GROUP="read_group_name" #Read Group name RGID tag in bam
PLATFORM="ILLUMINA"
NT=$(nproc) #number of threads to use in computation, set to all threads available

FASTA="/home/regression/references/b37/human_g1k_v37_decoy.fasta"
FASTQ_1="WGS-30x_1.fastq.gz"
FASTQ_2="WGS-30x_2.fastq.gz"

#run multiple BWA on multiple servers
file_list=""
for run in $(seq 0 $((num_splits-1))); do
        #send each of these to a different server
        sentieon bwa mem -R "@RG\tID:$GROUP\tSM:$SAMPLE\tPL:$PLATFORM" \
          -K $BWA_K_size -t $NT -p $FASTA \
```

(continues on next page)

(continued from previous page)

```
        "<sentieon fqidx extract -F $((run))/$num_splits -K $BWA_K_size $FASTQ_1 $FASTQ_2"␣
↪| \
        sentieon util sort -r $FASTA -t $NT --sam2bam -o sorted_run$run.bam -i -
      file_list="$file_list sorted_run$run.bam"
done

#merge the results
sentieon util merge -o sorted.bam $file_list
#or perform deduplication on all the intermediate BAM files
file_list_bam=""
for file in $file_list; do file_list_bam="$file_list_bam -i $file"; done
sentieon driver -r $FASTA -t $NT $file_list --algo LocusCollector --fun score_info score.txt.
↪gz
sentieon driver -r $FASTA -t $NT $file_list --algo Dedup --score_info score.txt.gz deduped.
↪bam
```

12.5.5 Distributing GVCFtyper for large cohort joint calling

For large joint calls with more than 1000 samples, we recommend setting `--genotype_model multinomial` in GVCFtyper. While the default genotyping mode is theoretically more accuracy for smaller cohorts, the multinomial mode is equally accurate with large cohorts and scales better with a very large numbers of samples.

When running large number of samples (>3000) joint cohort calling, distribution using similar techniques as the ones described above are recommended; however, a few challenges need to be taken into account:

- The overall runtime of the joint calling.
- The resource requirements of the machines where the distributed GVCFtyper command will be run.
- The logistics of storing and accessing the large number of GVCF input files.
- The file size of the output VCF file.

In general, the recommendation is to use the Sentieon® sharding capabilities of GVCFtyper to process different genomic shards in different machines. The commands below access the complete list of GVCF inputs, and assume those inputs are available in a common location such as a NFS storage or a remote object storage location accessible either through `s3` or `http/https` protocols:

```
#individual shard calling in machine 1
sentieon driver -r $FASTA -t $NT --shard $shard1 --shard $shard2 \
  --algo GVCFtyper $gvcf_argument GVCFtyper-shard_1.vcf.gz
#individual shard calling in machine 2
sentieon driver -r $FASTA -t $NT --shard $shard3 \
  --algo GVCFtyper $gvcf_argument GVCFtyper-shard_2.vcf.gz
. . .
```

Important

The list of input GVCFs needs to be ordered consistently when processing each shard, as the sample order in the final output will depend on the input order and merging will require all partial files to have the same sample order.

The output VCF files from GVCFtyper when using –shard option are not valid VCF files, so they should

not be used until you merge them as described below.

After all shards are processed, you need to run a GVCFtyper merge command to merge the results, making sure that the order of the intermediate VCF inputs is consistent with the reference. The input files need to be available in a common location such as a NFS storage or a remote object storage location accessible either through `s3` or `http/https` protocols:

```
#merge step
sentieon driver --passthru -t $nt --algo GVCFtyper --merge joint_final.vcf.gz \
  GVCFtyper-shard_1.vcf.gz GVCFtyper-shard_2.vcf.gz . . .
```

12.5.5.1 Overall runtime and resource requirements

In order to reduce the overall runtime, it is recommended to use enough shards to allow granular distribution of the runtime into multiple servers. The shards can be created either by shard size or expected complexity. However, the final merge step of the individual shard results cannot be distributed and needs to be run in a single server; this fact sets a lower bound on the number of servers to be used for the distribution, as the merging could dominate the overall runtime.

The GVCFtyper algorithm is a very efficient algorithm that will likely be I/O limited by the performance of the storage location of the GVCF inputs. As such, it is recommended to store the GVCF inputs in a file system that provides 600 MBps transfer rate.

For extremely large number of samples (>10000) joint cohort calling, memory may become a problem and certain OS restrictions may also come into play. For such a case, it is recommended to do the following:

- Set the OS open file limit to a big enough number, to allow the software to open enough file handles. This is done via the command `ulimit -n NUM`.
- Set the OS stack limit to a big enough number, to allow the software to allocate enough memory for the operation, as the memory is proportional to the number of samples, and may be too big to fit in the stack allocated by default. This is done via the command `ulimit -s NUM`.
- Use a file containing the list of input GVCFs, to prevent the command to be longer that the argument list OS length. You can do this by using the following command:

  ```
  sentieon driver ... --algo GVCFtyper output.vcf.gz - < input_files.txt
  ```

- Use jemalloc memory allocation as described in the *jemalloc appnote* and update the vcf cache size by adding the following code to your scripts:

  ```
  export VCFCACHE_BLOCKSIZE=4096
  ```

- Set the shard size to a smaller number, i.e. 50MBases. In addition, use the driver option `--traverse_param` in the GVCFtyper commands to make sure that all threads are fully utilized. The command would become:

  ```
  sentieon driver -r $FASTA -t $NT --shard $shard1 --shard $shard2 \
   --traverse_param 10000/200 \
   --algo GVCFtyper GVCFtyper-shard_1.vcf.gz - < input_files.txt
  ```

12.5.5.2 Challenge of the large output VCF file

When running a very large cohort, the output VCF file will contain a very large number of columns with the genotype information of each sample. This large number of columns may make the output file unwieldy, for instance making running VQSR on the full file impractical, so you may need to look into alternatives to VCF storage of the output.

Depending on the method you will use to store the output, splitting the output by either sample groups or genomic coordinates may ameliorate the issue of very large output files; please feel free to contact sentieon support with your specific request on how you will store the output of joint calling.

12.5.5.2.1 Splitting output by genomic regions

In order to make the output VCF file smaller, you can perform the shard merging across specific genomic sub-regions, for instance across individual chromosomes. You would do this by merging only a subset of the intermediate VCF files. For example you can create per chromosome VCF files by merging only the shards that cover each chromosome: if shards 1-4 cover chr1 and shard 5 covers both chr1 and chr2, the following code would create a VCF containing only chr1 vartiants:

```
#merge the necessary intermediate sharded VCFs
sentieon driver --passthru -t $nt --algo GVCFtyper --merge \
  GVCFtyper_chr1_tmp.vcf.gz \
  GVCFtyper-shard_1.vcf.gz GVCFtyper-shard_2.vcf.gz GVCFtyper-shard_3.vcf.gz \
  GVCFtyper-shard_4.vcf.gz GVCFtyper-shard_5.vcf.gz
#remove variants not in the proper contig and index the VCF
bcftools view GVCFtyper_chr1_tmp.vcf.gz  -r chr1 \
  -o - | sentieon util vcfconvert - GVCFtyper_chr1.vcf.gz
```

Using the above methodology allows you to create valid VCF files with a fraction of the size of the full VCF.

In order to run VQSR on the output above, you need to supply the `VarCal` algorithm with a VCF file containing all variant records across the complete genome, as they are required to perform the proper calibration of the VQSR model. However, `VarCal` only requires the first 8 columns of VCF data, so you do not need to concatenate all the VCF from each specific genomic sub-regions and can create a smaller VCF file that contains the necesary information by extracting and concatenating the first 8 columns of each file. Using the code below will create the necessary file:

```
vcf_list=(GVCFtyper_chr1.vcf.gz GVCFtyper_chr2.vcf.gz) # include more VCFs if applicable

#extract the first 8 columns from each region to a new compressed VCF
mkfifo tmp.fifo
sentieon util vcfconvert tmp.fifo GVCFtyper_annotations.vcf.gz &
convert_pid=$!
exec 3>tmp.fifo #a file descriptor to hold the fifo open
bcftools view -h ${vcf_list[0]} | grep "^##" > tmp.fifo
bcftools view -h ${vcf_list[0]} | grep "^#CHROM" | cut -f 1-8 > tmp.fifo
for vcf in ${vcf_list[@]}; do
  bcftools view -H $vcf | cut -f 1-8 > tmp.fifo
done
exec 3>&-
wait $convert_pid
rm tmp.fifo
```

The `VarCal` algo can then be run on the merged VCF containing the first eight columns of the VCF to generate a tranches and recal file for both SNPs and INDELs. VQSR can then be applied directly to the VCFs split by genomic region with the `ApplyVarCal` algo:

```
vcf_list=(GVCFtyper_chr1.vcf.gz GVCFtyper_chr2.vcf.gz) # include more VCFs if applicable

#apply variant quality score recalibration
for vcf in ${vcf_list[@]}; do
  out_vcf=${vcf%%.vcf.gz}_snp-indel-recal.vcf.gz
  sentieon driver -r $FASTA -t $nt --algo ApplyVarCal -v $vcf \
    --vqsr_model var_type=SNP,tranches_file=${snp_tranches},sensitivity=99.0,recal=${snp_
↪recal} \
    --vqsr_model var_type=INDEL,tranches_file=${indel_tranches},sensitivity=99.0,recal=$
↪{indel_recal} \
    $out_vcf
done
```

12.5.5.2.2 Splitting output by sample groups

Alternatively, the Sentieon® GVCFtyper merge includes the `--split_by_sample` algo option as a potential solution to deal with this challenge. The `--split_by_sample` algo option is used during the merge step to generate valid partial VCFs, each containing a subset of samples, thus separating the complete VCF file into smaller, more manageable output files. The usage of the `--split_by_sample` algo option is:

```
sentieon driver --passthru -t $nt --algo GVCFtyper --merge \
  --split_by_sample split.conf GVCFtyper_main.vcf.gz \
  GVCFtyper-shard_1.vcf.gz GVCFtyper-shard_2.vcf.gz . . .
```

The `split.conf` input file used in the `--split_by_sample` algo option is a tab separated file where each line starts with the output file of the specific sample group, followed by a tab separated list of the samples of the corresponding group. You can use more than one line with the same output file, which will group all samples from the multiple lines. The example below shows two groups of samples where samples 1 to 5 will be output to the group1 output file, while samples 6 to 8 will be output to the group2 output file:

```
GVCFtyper_file_group1.vcf.gz Sample1 Sample2 Sample3
GVCFtyper_file_group1.vcf.gz Sample4 Sample5
GVCFtyper_file_group2.vcf.gz Sample6 Sample7 Sample8
```

The GVCFtyper merge command above will generate the following outputs:

- GVCFtyper_main.vcf.gz: a partial VCF file containing all the VCF information up to and including the INFO column. No sample information will be included. This file is useful to run VQSR, as it contains all the necessary information for the variant recalibration.
- GVCFtyper_file_group1.vcf.gz: a partial VCF file containing the up to and including the INFO column, plus the FORMAT column, and all the columns for the samples including in group1.
- GVCFtyper_file_group2.vcf.gz: a partial VCF file containing the up to and including the INFO column, plus the FORMAT column, and all the columns for the samples including in group2.

You can then use bcftools to merge the partial VCFs and select the samples you are interested in. You can use the following code to do it:

```
bash extract.sh GVCFtyper_main.vcf.gz \
  split.conf Samle1,Sample4,Sample7
```

Where the 3rd argument of the script is a comma separated list of the samples of interest and the extract.sh script is:

```
#!/bin/bash
MAIN=$1
GRPS_CONF=$2
SAMPLES=$3
REGIONS=$4
BCF=/home/release/other_tools/bcftools-1.3/bcftools
if [ $# -eq 4 ] || [ $# -eq 3 ]; then
  if [ $REGIONS == "" ]; then
    BED_ARG=""
  else
    BED_ARG="-R $REGIONS"
  fi
  #parse input files from group config file
  GRPS=$(grep -v "^#" $GRPS_CONF| cut -f1 | sort | uniq)
  $BCF view -h $MAIN | grep -v '^#CHROM' | grep -v '^##bcftools' > out.vcf
  hdr=$($BCF view -h $MAIN | grep '^#CHROM')
  hdr="$hdr\tFORMAT"
  arg="<($BCF view $BED_ARG -H -I $MAIN | cut -f -8)"
  col=9
  for g in $GRPS; do
    s=$($BCF view --force-samples -s $SAMPLES -h $g 2>/dev/null | grep '^#CHROM' | \
       cut -f 10-)
    [ -z "$s" ] && continue
    hdr="$hdr\t$s"
    c="<($BCF view --force-samples -s $SAMPLES $BED_ARG -H -I $g 2>/dev/null | \
      cut -f $col-)"
    arg="$arg $c"
    col=10
  done
echo -e "$hdr" >> out.vcf
eval paste "$arg" >> out.vcf
else
  echo "usage $0 main_vcf_file group_config_file csv_sample_list [bed_file]"
fi
```

The solution presented here allows to easily run VQSR on the GVCFtyper_main.vcf.gz to create a file after recalibration, which you can then use in the extract.sh script to obtain the after VQSR results.

12.5.5.3 Special considerations for Cloud environments

In cloud environments, there is typically no NFS storage that can accommodate a large number of GVCF inputs. For joint genotyping, the Sentieon® GVCFtyper allows GVCF input files hosted in an object storage location such as AWS s3 or in a location accessible via HTTP. However, for large cohorts (100+), the memory requirements for the GVCFtyper command using object storage inputs will likely be too high, so this method of accessing the inputs is not recommended.

The recommended methodology in a cloud environment is to download partial GVCF input files to the computing node depending on the shard that the node is processing. The partial download of the GVCF inputs can be done using bcftools, but it is important to add the option *–no-version* to the bcftools command to make sure that the headers of the different shards are not different enough to trigger the GVCFtyper merge from rejecting them:

```
#use bcftools to download shard1 GVCF partial inputs and create the index
bcftools view --no-version -r $shard1_csv_intervals -o - $URL_sample.g.vcf.gz | \
```

(continues on next page)

(continued from previous page)

```
  sentieon util vcfconvert - ${sample}_s1.g.vcf.gz
#or download multiple files in parallel using xargs using a list
cat list_inputs.txt | xargs -P $NUM_PARALLEL_DOWNLOAD -I @ | \
  sh -c "bcftools view -r $shard1_csv_intervals -o - $S3BUCKET_INPUTS/@ | \
  sentieon util vcfconvert - @"
```

To gain additional efficiency, a "waterfall" approach can be used, as shown in Fig. 12.8 and Fig. 12.9. In this approach a computing node will process multiple shards sequentially, running the download of the partial GVCF inputs for the next shard in parallel with the GVCFtyper processing of the current shard, thus more efficiently sharing resources as a CPU intensive and I/O intensive process would. The pipeline would be as follows:

- Download the partial GVCF inputs for shard1.
- Start the GVCFtyper of shard1, and in parallel download the partial GVCF inputs for shard2.
- After the previous step is completed, start the GVCFtyper of shard2, and in parallel use bcftools to download shard3.

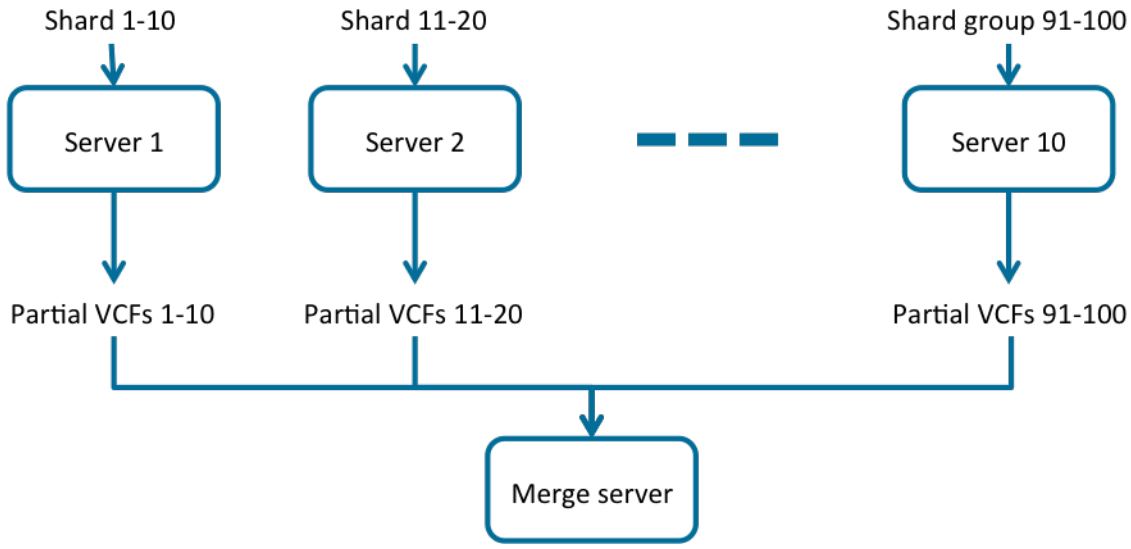


Fig. 12.8: Distributing GVCFtyper onto multiple servers in a cloud environment

12.5.6 Shell examples

Below is a shell example for the distribution commands, using shard size 1G bases, which was selected for demo purposes only. The recommended shard size is 100M bases.

```
# Sample file for distributing DNAseq pipeline onto 4 1GBase shards
# Each stage command can be distributed to a different server for faster processing,
# but the user needs to make sure that the necessary files are present in each machine

FASTA="/home/b37/human_g1k_v37_decoy.fasta"
FASTQ_1="WGS-30x_1.fastq.gz"
FASTQ_2="WGS-30x_2.fastq.gz"
FASTQ_INDEX="WGS-30x.fastq.gz.index"
KNOWN1="/home/b37/1000G_phase1.indels.b37.vcf.gz"
```

(continues on next page)

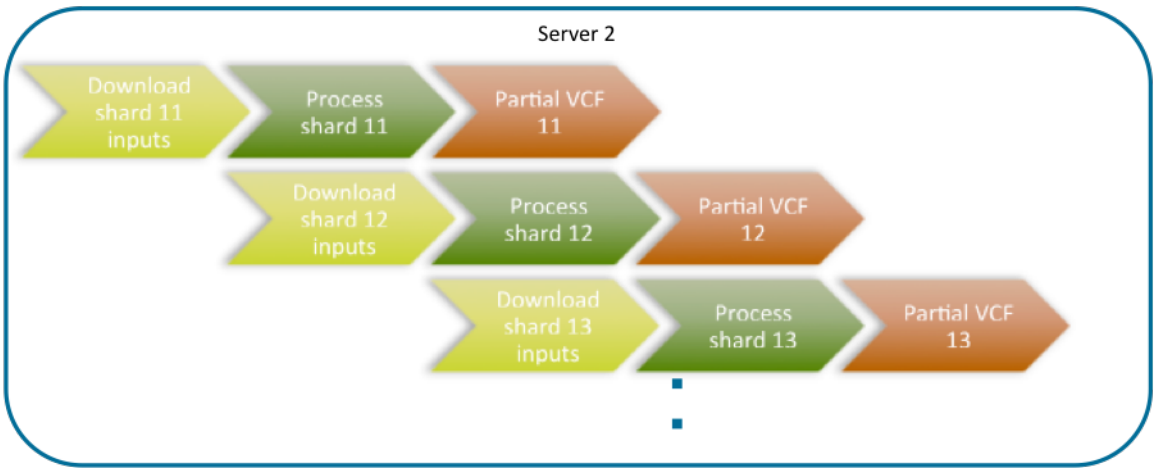


Fig. 12.9: Distributing GVCFtyper onto multiple servers in a cloud environment, detail of the waterfall approach of downloading the inputs of the next shard in parallel with the processing of the previous shard

(continued from previous page)

```
KNOWN2="/home/b37/Mills_and_1000G_gold_standard.indels.b37.vcf.gz"
DBSNP="/home/b37/dbsnp_138.b37.vcf.gz"

#######################################
# BWA mapping, distributed on 4 servers
#######################################
BWA_K_size=100000000
num_srvr=4

#get the number of runs that the inputs will be split into given the size
num_K=$(sentieon fqidx query -i $FASTQ_INDEX | cut -d' ' -f1)
BWA_K_size=$(sentieon fqidx query -i $FASTQ_INDEX | cut -d' ' -f2)
num_K_srvr=$(expr $num_K / $num_srvr + 1)
#run multiple BWA on multiple servers
file_list=""
for run in $(seq 0 $((num_srvr-1))); do
        region="$((run*num_K_srvr))-$((run*num_K_srvr+num_K_srvr))"
        #send each of these to a different server
        sentieon bwa mem -R "@RG\tID:$GROUP\tSM:$SAMPLE\tPL:$PLATFORM" \
          -K $BWA_K_size -t $NT -p $FASTA \
          "< sentieon fqidx extract -i $FASTQ_INDEX -r $region $FASTQ_1 $FASTQ_2" | \
          sentieon util sort -r $FASTA -t $NT --sam2bam -o sorted_run$run.bam -i -
        file_list="$file_list sorted_run$run.bam"
done

#merge the results
sentieon util merge -o sorted.bam $file_list

#######################################
# define 4 shards
#######################################
SHARD_0="--shard 1:1-249250621 --shard 2:1-243199373 --shard 3:1-198022430 \
  --shard 4:1-191154276 --shard 5:1-118373300"
```

(continues on next page)

(continued from previous page)

```
SHARD_1="--shard 5:118373301-180915260 --shard 6:1-171115067 --shard 7:1-159138663 \
  --shard 8:1-146364022 --shard 9:1-141213431 --shard 10:1-135534747 \
  --shard 11:1-135006516 --shard 12:1-49085594"
SHARD_2="--shard 12:49085595-133851895 --shard 13:1-115169878 --shard 14:1-107349540 \
   --shard 15:1-102531392 --shard 16:1-90354753 --shard 17:1-81195210 \
   --shard 18:1-78077248 --shard 19:1-59128983 --shard 20:1-63025520 \
   --shard 21:1-48129895 --shard 22:1-51304566 --shard X:1-118966714"
SHARD_3="--shard X:118966715-155270560 --shard Y:1-59373566 --shard MT:1-16569 \
   --shard GL000207.1:1-4262 --shard GL000226.1:1-15008 --shard GL000229.1:1-19913 \
   --shard GL000231.1:1-27386 --shard GL000210.1:1-27682 --shard GL000239.1:1-33824 \
   --shard GL000235.1:1-34474 --shard GL000201.1:1-36148 --shard GL000247.1:1-36422 \
   --shard GL000245.1:1-36651 --shard GL000197.1:1-37175 --shard GL000203.1:1-37498 \
   --shard GL000246.1:1-38154 --shard GL000249.1:1-38502 --shard GL000196.1:1-38914 \
   --shard GL000248.1:1-39786 --shard GL000244.1:1-39929 --shard GL000238.1:1-39939 \
   --shard GL000202.1:1-40103 --shard GL000234.1:1-40531 --shard GL000232.1:1-40652 \
   --shard GL000206.1:1-41001 --shard GL000240.1:1-41933 --shard GL000236.1:1-41934 \
   --shard GL000241.1:1-42152 --shard GL000243.1:1-43341 --shard GL000242.1:1-43523 \
   --shard GL000230.1:1-43691 --shard GL000237.1:1-45867 --shard GL000233.1:1-45941 \
   --shard GL000204.1:1-81310 --shard GL000198.1:1-90085 --shard GL000208.1:1-92689 \
   --shard GL000191.1:1-106433 --shard GL000227.1:1-128374 \
   --shard GL000228.1:1-129120 --shard GL000214.1:1-137718 \
   --shard GL000221.1:1-155397 --shard GL000209.1:1-159169 \
   --shard GL000218.1:1-161147 --shard GL000220.1:1-161802 \
   --shard GL000213.1:1-164239 --shard GL000211.1:1-166566 \
   --shard GL000199.1:1-169874 --shard GL000217.1:1-172149 \
   --shard GL000216.1:1-172294 --shard GL000215.1:1-172545 \
   --shard GL000205.1:1-174588 --shard GL000219.1:1-179198 \
   --shard GL000224.1:1-179693 --shard GL000223.1:1-180455 \
   --shard GL000195.1:1-182896 --shard GL000212.1:1-186858 \
   --shard GL000222.1:1-186861 --shard GL000200.1:1-187035 \
   --shard GL000193.1:1-189789 --shard GL000194.1:1-191469 \
   --shard GL000225.1:1-211173 --shard GL000192.1:1-547496 \
   --shard NC_007605:1-171823 --shard hs37d5:1-35477943"

#######################################
# Locus collector
#######################################
#Locus Collector for shard 000
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam $SHARD_0 \
   --algo LocusCollector --fun score_info .part_deduped.bam000.score.gz \
   2> collect000.log
#Locus Collector for shard 001
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam $SHARD_1 \
   --algo LocusCollector --fun score_info .part_deduped.bam001.score.gz \
   2> collect001.log
#Locus Collector for shard 002
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam $SHARD_2 \
   --algo LocusCollector --fun score_info .part_deduped.bam002.score.gz \
   2> collect002.log
#Locus Collector for shard 003
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam $SHARD_3 \
   --algo LocusCollector --fun score_info .part_deduped.bam003.score.gz \
```

(continues on next page)

(continued from previous page)

```
   2> collect003.log
#Locus Collector for shard with unmapped reads (NO_COOR)
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam --shard NO_COOR \
   --algo LocusCollector --fun score_info .part_deduped.bam004.score.gz \
   2> collect004.log

#######################################
# Dedup using all score.gz files
#######################################
#Dedup for shard 000
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam $SHARD_0 \
   --algo Dedup --score_info .part_deduped.bam000.score.gz \
   --score_info .part_deduped.bam001.score.gz \
   --score_info .part_deduped.bam002.score.gz \
   --score_info .part_deduped.bam003.score.gz \
   --score_info .part_deduped.bam004.score.gz \
   --rmdup .part_deduped000.bam 2> dedup000.log
#Dedup for shard 001
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam $SHARD_1 \
   --algo Dedup --score_info .part_deduped.bam000.score.gz \
   --score_info .part_deduped.bam001.score.gz \
   --score_info .part_deduped.bam002.score.gz \
   --score_info .part_deduped.bam003.score.gz \
   --score_info .part_deduped.bam004.score.gz \
   --rmdup .part_deduped001.bam 2> dedup001.log
#Dedup for shard 002
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam $SHARD_2 \
   --algo Dedup --score_info .part_deduped.bam000.score.gz \
   --score_info .part_deduped.bam001.score.gz \
   --score_info .part_deduped.bam002.score.gz \
   --score_info .part_deduped.bam003.score.gz \
   --score_info .part_deduped.bam004.score.gz \
   --rmdup .part_deduped002.bam 2> dedup002.log
#Dedup for shard 003
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam $SHARD_3 \
   --algo Dedup --score_info .part_deduped.bam000.score.gz \
   --score_info .part_deduped.bam001.score.gz \
   --score_info .part_deduped.bam002.score.gz \
   --score_info .part_deduped.bam003.score.gz \
   --score_info .part_deduped.bam004.score.gz \
   --rmdup .part_deduped003.bam 2> dedup003.log
#Dedup for shard with unmapped reads (NO_COOR)
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -i sorted.bam --shard NO_COOR \
   --algo Dedup --score_info .part_deduped.bam000.score.gz \
   --score_info .part_deduped.bam001.score.gz \
   --score_info .part_deduped.bam002.score.gz \
   --score_info .part_deduped.bam003.score.gz \
   --score_info .part_deduped.bam004.score.gz \
   --rmdup .part_deduped004.bam 2> dedup004.log
#Merge bam files from all shards into final output
$SENTIEON_FOLDER/bin/sentieon util merge -i .part_deduped000.bam \
   -i .part_deduped001.bam -i .part_deduped002.bam \
```

(continues on next page)

(continued from previous page)

```
   -i .part_deduped003.bam -i .part_deduped004.bam \
   -o deduped.bam --mergemode=10 2> dedup_merge.log

#####################################
# BQSR
#####################################
#QualCal for shard 000
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped000.bam \
   -i .part_deduped001.bam $SHARD_0 --algo QualCal -k $KNOWN1 -k $KNOWN2 \
   .part_recal_data000.table 2> bqsr000.log
#QualCal for shard 001
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped000.bam \
   -i .part_deduped001.bam -i .part_deduped002.bam $SHARD_1 --algo QualCal \
   -k $KNOWN1 -k $KNOWN2 .part_recal_data001.table 2> bqsr001.log
#QualCal for shard 002
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped001.bam \
   -i .part_deduped002.bam -i .part_deduped003.bam $SHARD_2 --algo QualCal \
   -k $KNOWN1 -k $KNOWN2 .part_recal_data002.table 2> bqsr002.log
#QualCal for shard 003
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped002.bam \
   -i .part_deduped003.bam $SHARD_3 --algo QualCal -k $KNOWN1 -k $KNOWN2 \
   .part_recal_data003.table 2> bqsr003.log
#QualCal for shard with unmapped reads (NO_COOR)
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped004.bam \
   --shard NO_COOR --algo QualCal -k $KNOWN1 -k $KNOWN2 \
   .part_recal_data004.table 2> bqsr004.log
#Merge table files into complete calibration table
$SENTIEON_FOLDER/bin/sentieon driver --passthru --algo QualCal \
   --merge recal_data.table .part_recal_data000.table .part_recal_data001.table \
   .part_recal_data002.table .part_recal_data003.table .part_recal_data004.table \
   2> bqsr_merge.log

#####################################
# Variant Calling
#####################################
#Haplotyper for shard 000
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped000.bam \
   -i .part_deduped001.bam -q recal_data.table $SHARD_0 --algo Haplotyper \
   -d $DBSNP .part_output000.vcf.gz 2> hc000.log
#Haplotyper for shard 001
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped000.bam \
   -i .part_deduped001.bam -i .part_deduped002.bam  -q recal_data.table \
   $SHARD_1 --algo Haplotyper -d $DBSNP .part_output001.vcf.gz 2> hc001.log
#Haplotyper for shard 002
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped001.bam \
   -i .part_deduped002.bam -i .part_deduped003.bam -q recal_data.table \
   $SHARD_2 --algo Haplotyper -d $DBSNP .part_output002.vcf.gz 2> hc002.log
#Haplotyper for shard 003
$SENTIEON_FOLDER/bin/sentieon driver -t 16 -r $FASTA -i .part_deduped002.bam \
   -i .part_deduped003.bam -q recal_data.table $SHARD_3 --algo Haplotyper \
   -d $DBSNP .part_output003.vcf.gz 2> hc003.log
#There is no need to do variant calling on the NO_COOR shard, as there will
```

(continues on next page)

(continued from previous page)

```
# be no variants on unmapped reads
#Merge output files into output VCF
$SENTIEON_FOLDER/bin/sentieon driver --passthru --algo Haplotyper \
   --merge output.vcf.gz .part_output000.vcf.gz .part_output001.vcf.gz \
   .part_output002.vcf.gz .part_output003.vcf.gz 2> hc_merge.log
```

Below is a shell example `generate_shards.sh` to create the shards.

```
determine_shards_from_bam(){
    local bam step tag chr len pos end
    bam="$1"
    step="$2"
    pos=1
    samtools view -H $bam |\
    while read tag chr len; do
        [ $tag == '@SQ' ] || continue
        chr=$(expr "$chr" : 'SN:\(.*\)')
        len=$(expr "$len" : 'LN:\(.*\)')
        while [ $pos -le $len ]; do
            end=$(($pos + $step - 1))
            if [ $pos -lt 0 ]; then
                start=1
            else
                start=$pos
            fi
            if [ $end -gt $len ]; then
                echo -n "$chr:$start-$len "
                pos=$(($pos-$len))
                break
            else
                echo "$chr:$start-$end"
                pos=$(($end + 1))
            fi
        done
    done
    echo "NO_COOR"
}

determine_shards_from_dict(){
    local bam step tag chr len pos end
    dict="$1"
    step="$2"
    pos=1
    cat $dict |\
    while read tag chr len UR; do
        [ $tag == '@SQ' ] || continue
        chr=$(expr "$chr" : 'SN:\(.*\)')
        len=$(expr "$len" : 'LN:\(.*\)')
        while [ $pos -le $len ]; do
            end=$(($pos + $step - 1))
            if [ $pos -lt 0 ]; then
                start=1
```

(continues on next page)

(continued from previous page)

```
            else
                start=$pos
            fi
            if [ $end -gt $len ];then
                echo -n "$chr:$start-$len "
                pos=$(($pos-$len))
                break
            else
                echo "$chr:$start-$end"
                pos=$(($end + 1))
            fi
        done
    done
    echo "NO_COOR"
}


determine_shards_from_fai(){
    local bam step tag chr len pos end
    fai="$1"
    step="$2"
    pos=1
    cat $fai |\
    while read chr len other; do
        while [ $pos -le $len ]; do
            end=$(($pos + $step - 1))
            if [ $pos -lt 0 ]; then
                start=1
            else
                start=$pos
            fi
            if [ $end -gt $len ]; then
                echo -n "$chr:$start-$len "
                pos=$(($pos-$len))
                break
            else
                echo "$chr:$start-$end"
                pos=$(($end + 1))
            fi
        done
    done
    echo "NO_COOR"
}

if [ $# -eq 2 ]; then
    filename=$(basename "$1")
    extension="${filename##*.}"
    if [ "$extension" = "fai" ]; then
        determine_shards_from_fai $1 $2
    elif [ "$extension" = "bam" ]; then
        determine_shards_from_bam $1 $2
    elif [ "$extension" = "dict" ]; then
```

(continues on next page)

(continued from previous page)

```
        determine_shards_from_dict $1 $2
    fi
else
    echo "usage $0 file shard_size"
fi
```

12.6 Functional Equivalent Pipeline from CCDG using Sentieon®

12.6.1 Introduction

This documents describes how to implement the "Functional Equivalent Pipeline", also known as the CCDG pipeline standard, described in https://github.com/CCDG/Pipeline-Standardization/blob/master/PipelineStandard.md and published in https://www.nature.com/articles/s41467-018-06159-4 using the Sentieon® tools. In order to match the version requirements of this pipeline, you should use the Sentieon® tools with version 201704 or higher. Starting with Sentieon® tools version 201911, the Sentieon® BWA was updated to version 0.7.17; BWA version 0.7.17 produces MC MateTags in its output, and samblaster addMateTags will not remove this MC tag and add its own MC tag to the BAM file, creating a duplicated MC tag.

12.6.2 Command line equivalence

12.6.2.1 Alignment

The alignment stage in the CCDG functional equivalent pipeline is done using BWA-MEM version 0.7.15:

```
FASTA=Homo_sapiens_assembly38.fasta
NT=$(nproc)
bwa mem -R "@RG\tID:$RGID\tSM:$SM\tPL:$PL" -t $NT -K 100000000 -Y $FASTA $FASTQ1 $FASTQ2 | \
samblaster --addMateTags -a | \
samtools view -Sbhu - | \
sambamba sort -n -t $nt --tmpdir tmp -o sorted.bam /dev/stdin
```

To run the equivalent command with Sentieon®:

```
sentieon bwa mem -R "@RG\tID:$RGID\tSM:$SM\tPL:$PL" -t $NT -K 100000000 -Y $FASTA $FASTQ1
↪$FASTQ2 | \
samblaster --addMateTags -a | \
util sort --sam2bam -i - -r $FASTA -t $nt -o sorted.bam
```

To run the equivalent command when using Sentieon® version 201911 or higher:

```
sentieon bwa mem -R "@RG\tID:$RGID\tSM:$SM\tPL:$PL" -t $NT -K 100000000 -Y $FASTA $FASTQ1
↪$FASTQ2 | \
sed 's|MC:Z:[^\t]*\t||' | \
samblaster --addMateTags -a | \
util sort --sam2bam -i - -r $FASTA -t $nt -o sorted.bam
```

12.6.2.2 Duplicate marking

The deduplication stage in the CCDG functional equivalent pipeline is done using Picard version 2.4 or higher:

```
java -Xmx48g -jar $picard MarkDuplicates I=sorted.bam METRICS_FILE=markdup_metrics.txt \
   ASSUME_SORT_ORDER=queryname QUIET=true COMPRESSION_LEVEL=0 O=/dev/stdout | \
sambamba sort -t $NT --tmpdir tmp -o markduped.bam /dev/stdin
```

To run the equivalent command with Sentieon®:

```
sentieon driver -t $nt -r $FASTA -i sorted.bam --algo LocusCollector --fun score_info tmp_
↪score.gz && \
sentieon driver -t $nt -r $FASTA -i sorted.bam --algo Dedup --score_info tmp_score.gz \
   --output_dup_read_name --metrics dedup_metrics.txt tmp_dup_qname.txt.gz && \
sentieon driver -t $nt -r $FASTA -i sorted.bam --algo Dedup --dup_read_name tmp_dup_qname.
↪txt.gz markduped.bam
```

The Sentieon® command uses a special 3-pass Deduplication flow to mark both primary and non-primary reads.

12.6.2.3 Base quality score recalibration with binning scheme

The BQSR stage in the CCDG functional equivalent pipeline is done using GATK3 or GATK4:

```
INTERVAL_ARG="-L chr1 -L chr2 -L chr3 -L chr4 -L chr5 -L chr6 -L chr7 -L chr8 -L chr9 -L␣
↪chr10 \
   -L chr11 -L chr12 -L chr13 -L chr14 -L chr15 -L chr16 -L chr17 -L chr18 -L chr19 -L chr20␣
↪-L chr21 -L chr22"
DOWNSAMPLE_ARG="--downsample_to_fraction .1"
KNOWN_MILLS_INDELS="Mills_and_1000G_gold_standard.indels.hg38.vcf.gz"
KNOWN_1000G_INDELS="Homo_sapiens_assembly38.known_indels.vcf.gz"
KNOWN_DBSNP="Homo_sapiens_assembly38.dbsnp138.vcf"
java -Xmx48g -jar $GATK_37 -T BaseRecalibrator -R $FASTA -I markduped.bam $DOWNSAMPLE_ARG
↪$INTERVAL_ARG \
   -knownSites $KNOWN_MILLS_INDELS -knownSites $KNOWN_1000G_INDELS -knownSites $KNOWN_DBSNP \
   -o recal_data_37.table && \
java -Xmx15g -jar $GATK_37 -T PrintReads -R $fasta -I markduped.bam --BQSR recal_data_37.
↪table -o recaled_37.bam \
   --globalQScorePrior -1.0 --preserve_qscores_less_than 6 --static_quantized_quals 10 \
   --static_quantized_quals 20 --static_quantized_quals 30 --disable_indel_quals && \
samtools view -C -T $fasta -@ 2 -o recaled_37.cram recaled_37.bam && \
samtools index -c recaled_37.cram recaled_37.cram.index
```

To run the equivalent command with Sentieon®:

```
INTERVAL_ARG="--interval chr1,chr2,chr3,chr4,chr5,chr6,chr7,chr8,chr9,chr10,chr11,\
chr12,chr13,chr14,chr15,chr16,chr17,chr18,chr19,chr20,chr21,chr22"
sentieon driver -t $NT -r $FASTA --interval $INTERVAL_ARG -i markduped.bam --algo QualCal -k
↪$KNOWN_MILLS_INDELS \
   -k $KNOWN_1000G_INDELS -k $KNOWN_DBSNP recal_data_Sentieon.table && \
sentieon driver -t $NT -r $FASTA -i markduped.bam \
   --read_filter QualCalFilter,table=recal_data_Sentieon.table,prior=-1.0,indel=false,
↪levels=10/20/30,min_qual=6 \
   --algo ReadWriter recaled_RW.cram
```

Bear in mind that Sentieon® does not do any downsampling, as the Sentieon® tools are efficient enough that they are able to handle all the depth in your sequencing. In addition, this flow is differernt from the normal best practices flow to implement the special binning required in the CCDG functional equivalent

pipeline.

12.6.3 Pipeline script using Sentieon®

The following script will perform the CCDG functional equivalent pipeline on you input FASTQs using Sentieon®:

```
#!/bin/sh
# ****************************************************************************
# Script to perform DNA seq variant calling using Sentieon following
# the functional equivalent pipeline described in
# https://github.com/CCDG/Pipeline-Standardization/blob/master/PipelineStandard.md
# ****************************************************************************

# Update with the fullpath location of your sample fastq
SM="sample" #sample name
RGID="rg_$SM" #read group ID
PL="ILLUMINA" #or other sequencing platform
FASTQ_1="${SAMPLE}_r1.fastq.gz"
FASTQ_2="${SAMPLE}_r2.fastq.gz" #if using 2 FASTQ inputs

# Update with the location of the reference data files
FASTA_DIR="/home/regression/references/hg38bundle"
FASTA="$FASTA_DIR/Homo_sapiens_assembly38.fasta"
KNOWN_DBSNP="$FASTA_DIR/Homo_sapiens_assembly38.dbsnp138.vcf.gz"
KNOWN_INDELS="$FASTA_DIR/Homo_sapiens_assembly38.known_indels.vcf.gz"
KNOWN_MILLS="$FASTA_DIR/Mills_and_1000G_gold_standard.indels.hg38.vcf.gz"

# Update with the location of the Sentieon software package and license file
SENTIEON_INSTALL_DIR=/home/release/sentieon-genomics-|release_version|
export SENTIEON_LICENSE=/home/Licenses/Sentieon.lic #or using licsrvr: c1n11.sentieon.
↪com:5443

# Other settings
NT=$(nproc) #number of threads to use in computation
SAMBLASTER=/home/release/other_tools/samblaster-0.1.23/samblaster
START_DIR=$PWD

# ****************************************
# 0. Setup
# ****************************************
workdir="$START_DIR/${SM}" #Determines where the output files will be stored
mkdir -p $workdir
logfile=$workdir/run.log
exec >>$logfile 2>&1
cd $workdir

# ****************************************
# main pipeline with Sentieon
# ****************************************
# 1. Mapping BWA-MEM 0.7.15 util sort
SENTIEON_VERSION=$($SENTIEON_INSTALL_DIR/bin/sentieon driver --version)
if (( $(echo "${SENTIEON_VERSION##*-} < 201911" |bc -l) )); then
        $SENTIEON_INSTALL_DIR/bin/sentieon bwa mem -R "@RG\tID:$RGID\tSM:$SM\tPL:$PL" -t $NT␣
```

(continues on next page)

(continued from previous page)

```
↪\
          -K 100000000 -Y $FASTA $FASTQ_1 $FASTQ_2 | \
       $SAMBLASTER --addMateTags -a | \
       $SENTIEON_INSTALL_DIR/bin/sentieon util sort -r $FASTA -o sorted.bam -t $NT --
↪sam2bam -i -
else
       #Sentieon 201911 and higher use BWA 0.7.17, which already produce MC tags in the␣
↪output
       $SENTIEON_INSTALL_DIR/bin/sentieon bwa mem -R "@RG\tID:$RGID\tSM:$SM\tPL:$PL" -t $NT␣
↪\
          -K 100000000 -Y $FASTA $FASTQ_1 $FASTQ_2 | \
       $SENTIEON_INSTALL_DIR/bin/sentieon util sort -r $FASTA -o sorted.bam -t $NT --
↪sam2bam -i -
fi

# 2. Mark Duplicates with Sentieon
$SENTIEON_INSTALL_DIR/bin/sentieon driver -t $NT -i sorted.bam --algo LocusCollector --fun␣
↪score_info score.txt
$SENTIEON_INSTALL_DIR/bin/sentieon driver -t $NT -i sorted.bam --algo Dedup --score_info␣
↪score.txt \
  --metrics mark_dup_metrics.txt --output_dup_read_name tmp_dup_qname.txt
$SENTIEON_INSTALL_DIR/bin/sentieon driver -t $NT -i sorted.bam --algo Dedup \
  --dup_read_name tmp_dup_qname.txt markduped.bam

# 3. Base Quality Score Recalibration with Sentieon
interval_arg="--interval chr1,chr2,chr3,chr4,chr5,chr6,chr7,chr8,chr9,chr10,chr11,\
chr12,chr13,chr14,chr15,chr16,chr17,chr18,chr19,chr20,chr21,chr22"
$SENTIEON_INSTALL_DIR/bin/sentieon driver $interval_arg -r $FASTA -t $NT -i markduped.bam \
  --algo QualCal -k $KNOWN_MILLS -k $KNOWN_INDELS -k $KNOWN_DBSNP recal_data.table
$SENTIEON_INSTALL_DIR/bin/sentieon driver -r $FASTA -t $NT -i markduped.bam \
  --read_filter QualCalFilter,table=recal_data.table,prior=-1.0,indel=false,levels=10/20/30,
↪min_qual=6 \
  --algo ReadWriter recaled_RW.cram

# 4. Haplotyper with Sentieon
$SENTIEON_INSTALL_DIR/bin/sentieon driver -r $FASTA -t $NT -i recaled_RW.cram --algo␣
↪Haplotyper Haplotyper.vcf.gz
```

12.7 Building and Installing gnuplot

12.7.1 Introduction

This document describes the proper steps to install `gnuplot` on a user's Linux system. For any questions regarding this document, please contact Sentieon support at support@sentieon.com.

12.7.2 Background

gnuplot is a command-line graphics utility that for Linux and other operating systems. For more general information about gnuplot, please refer to http://www.gnuplot.info/.

The Sentieon software uses gnuplot to create figures using the output of the `QualCal`, `VarCal`, `GCBias`, `QualDistribution`, `InsertSizeMetricAlgo`, and `MeanQualityByCycle` algos.

12.7.3 Installation

Sentieon recommends building `gnuplot` from the repository source code. The latest version of `gnuplot` can be found at, http://www.gnuplot.info/download.html. More information on gnuplot configuration and installation can be found in the `INSTALL` and `README` files in the gnuplot source package.

Below, we show example installations on different operating systems.

12.7.3.1 Rocky Linux/AlmaLinux

```
yum install gcc make patch gzip tar cairo-devel pango-devel  # may require sudo
curl -L "https://downloads.sourceforge.net/project/gnuplot/gnuplot/6.0.2/gnuplot-6.0.2.tar.gz
↪" | \
  tar -zxf -
cd gnuplot-6.0.2
./configure
make
make install  # may require sudo
```

12.7.3.2 Ubuntu/Debian

```
apt update  # may require sudo
apt install make gcc gzip libgd-dev libcairo2-dev libpango1.0-dev curl  # may require sudo
curl -L "https://downloads.sourceforge.net/project/gnuplot/gnuplot/6.0.2/gnuplot-6.0.2.tar.gz
↪" | \
  tar -zxf -
cd gnuplot-6.0.2
./configure
make
make install  # may require sudo
```

12.7.4 Usage in Sentieon

The `sentieon plot` commands will call `gnuplot` from the `PATH` to produce pdf files.

12.8 Using jemalloc to Optimize Memory Allocation

12.8.1 Introduction

This document describes the proper steps to install `libjemalloc.so` that is optimized for a user's Linux system. For any questions regarding this document, please contact Sentieon support at support@sentieon.com.

12.8.2 Background

jemalloc is a memory allocator, optimized for high memory allocation performance and fewer memory fragments in multi-thread scenarios. For more general information about jemalloc, please refer to https://github.com/jemalloc/jemalloc

Sentieon recommends using jemalloc to improve memory management and overall performance in Sentieon applications, especially Sentieon bwa-mem.

12.8.3 Installation

Sentieon recommends installing a pre-built `libjemalloc.so`. Users may need root access to complete the installation.

12.8.3.1 RHEL/CentOS 8.x

```
yum install epel-release
yum install jemalloc
```

By default, the `libjemalloc.so` is installed at

```
/usr/lib64/libjemalloc.so.2
```

12.8.3.2 RHEL/CentOS 7.x

```
yum install epel-release
yum install jemalloc
```

By default, the `libjemalloc.so` is installed at

```
/usr/lib64/libjemalloc.so.1
```

12.8.3.3 Ubuntu 20.04

```
apt update
apt install libjemalloc2
```

By default, the `libjemalloc.so` is installed at

```
/usr/lib/x86_64-linux-gnu/libjemalloc.so.2
```

12.8.3.4 Ubuntu 18.04

```
apt update
apt install libjemalloc1
```

By default, the `libjemalloc.so` is installed at

```
/usr/lib/x86_64-linux-gnu/libjemalloc.so.1
```

12.8.3.5 Other systems without a prebuilt package

Please refer to `INSTALL.md` on the jemalloc GitHub page, https://github.com/jemalloc/jemalloc, for more information on how to build and install jemalloc.

12.8.4 Loading jemalloc in a Sentieon pipeline

The `LD_PRELOAD` environment variable can be used to load the jemalloc library in Sentieon at run time.

For example, on a CentOS 8.x system, you can use the following code to set this environment variable before running the Sentieon tools:

```
export LD_PRELOAD=/usr/lib64/libjemalloc.so.2
```

12.9 License Server Extension

12.9.1 Introduction

This document describes how to extend the license server to provide additional authentication functionality. If you have any additional questions, please contact the technical support at Sentieon® Inc. at support@sentieon.com.

12.9.2 How the license server extension works

The license server extension adds another layer of user-defined authentication on top of the license server. It allows the user to define a custom authentication agent to be launched by the license server whenever a new license request comes in. Here is how the extension works.

1. On the license server side, the user needs to provide a script to be launched with the license server. The script can be written in any scripting language, for example, Python or shell. In this document, we will assume the script is a Python script named auth.py.
2. Launch the license server with option –auth pointing to the authentication script.

   ```
   sentieon licsrvr --start --auth /path/to/auth.py /path/to/licfile
   ```

3. On the computing node, set two environment variables SENTIEON_AUTH_MECH and SENTIEON_AUTH_DATA. Through these environment variables, user can set any preset content or information gathered locally for authentication or other purposes.

   ```
   export SENTIEON_AUTH_MECH=$SECURITY_MECHANISM

   export SENTIEON_AUTH_DATA=$SECURITY_CONTEXT
   ```

4. Start a new Sentieon® job. When the job starts, the content of `$SENTIEON_AUTH_MECH` and `$SENTIEON_AUTH_DATA`, together with other local environment data will be sent to the license server in JSON format with the following keys:
   - mech: `$SENTIEON_AUTH_MECH`.
   - data: `$SENTIEON_AUTH_DATA`.
   - addr: Local IP address.
   - user: Current user.
   - groups: User groups current user belongs to.
5. On the license server, the above JSON string passed from the computing node will be piped to `auth.py` script through standard input. The script will parse and authenticate the JSON data. If the authentication is successful, the script will return with exit status 0. Otherwise, it will return with a non-zero exit status.
6. If `auth.py` returns non-zero, the license server will return license validation failure status to the requesting job. In that case, the job will quit with a “No license” error message.

12.9.3 License server extension script

12.9.3.1 Specification

The license server extension script needs to meet the following:

1. The script file's permission is executable by the license server launcher.
2. The script accepts the input JSON string through standard input.
3. When authentication is successful, the script returns 0. Otherwise, it returns non-zero.

12.9.3.2 Example

On the license server, the license server is started with the following command:

```
sentieon licsrvr --start --auth /path/to/auth_unix.py lic_file.lic
```

An example of `auth_unix.py` file is below:

```
#!/usr/bin/env python
import json
import sys
import urllib

secret_key = "My_scret_key"

def main(argv):
    inputs = json.load(sys.stdin)
    print >>sys.stderr, inputs
    if inputs.get('mech') != 'unix':
        print >>sys.stderr, "mech not unix"
        return -1
    if inputs.get('data') != secret_key:
        print >>sys.stderr, "Incorrect key"
        return -1
    if not isinstance(inputs.get('groups'), list):
        return -1
    if 'Developers' not in inputs.get('groups'):
        print >>sys.stderr, "Developers not in groups"
        return -1
    if 'baduser' == inputs.get('user'):
        print >>sys.stderr, "baduser is running the test"

    return 0

if __name__ == '__main__':
    sys.exit(main(sys.argv))
```

On the computing node, the following environment variable is set.

```
export SENTIEON_LICENSE=mylicsrvr:8992
export SENTIEON_AUTH_MECH=METHOD
export SENTIEON_AUTH_DATA=ASK_ME
```

Launch a job the computing node will pass the following JSON string to the license server:

```
{u'mech': u'METHOD', u'data': u'ASK_ME', u'addr': u'10.1.2.3',\
 u'groups': [u'Domain Users', u'Developers'], u'user': u'goodguy'}
```

Since 'METHOD' is not 'unix' as expected by the `auth_unix.py` script, `auth_unix.py` script will exit -1. The license server will print out the following output, and the requesting job will exit with a "No license" error message.

```
mech not unix
```

12.9.4 Cloud Examples

12.9.4.1 Google Computing Platform - Running a license server in a persistent e2-micro instance

In cloud environments, the most straightforward way to set up a license server is to have a very small continually running virtual machine acting as the license server. This setup will allow the license server to serve licenses to any other VM within the license server's Subnet.

The following instructions will guide you through setting up a e2-micro instance as a license server:

1. Choose an GCP Region to work in. Choose (or create) the Subnet where you will run the Sentieon® tools.
2. Note the CIRD block for your Subnet.
3. Create a firewall rule set for the license server (Fig. 12.10 and Fig. 12.11). The rules must:

   a. Allow inbound TCP communication at a specific port (we use 8990 by default as it is not typically used by other applications). This port is used to allow communication between the compute nodes and the license server. You should open TCP at the desired port across your Subnet's CIDR block (10.128.0.0/20 in Fig. 12.10) to whitelist traffic from within your Subnet.

   b. Allow outbound HTTPS communication to Sentieon® license master at master.sentieon.com (IP 52.89.132.242). This is necessary to enable license validation and updates.

   c. Allow SSH communication. This is necessary for administration.

   d. (Recommended) Allow ICMP communication. This allows PMTU with minimal security risk.
4. Launch a e2-micro instance to run the license server. Be sure to launch the instance in the region you selected earlier and connect to the network interface with the updated firewall rules created in step 3.
5. Send your Sentieon® support representative the Private IP address of your instance, together with the TCP port you opened in step 3.a.
6. Download the Sentieon® tools and your license file to the instance.
7. Start the license server with the following command.

   ```
   sentieon licsrvr --start [-l <licsrvr_log>] <license_file>
   ```
8. (Optional) Confirm the license server is working correctly and serving licenses to the license server instance, by running following commands; the second command will return the number of available licenses.

   ```
   sentieon licclnt ping -s <IP_addresss>:<PORT> || (echo "Ping Failed"; exit 1)
   sentieon licclnt query -s <IP_address>:<PORT> klib
   ```
9. Create a firewall rule set for the compute nodes (Fig. 12.12 and Fig. 12.13). The rules should:

a. Allow inbound SSH communication for administration.

b. Allow outbound TCP communication at the port chosen in step 3-a, which will be open across your Subnet's CIDR block.

c. (Recommended) Allow ICMP communication to facilitate communication between the license server and compute nodes.

10. Launch another e2-micro instance within the same Subnet to test the license server. Be sure launch the instance in the region you selected earlier and connect to the network interface with the updated firewall rules created in step 9.

11. Confirm the license server is working and serving licenses within the Subnet by running the following commands; the second command will return the number of available licenses.

```
sentieon licclnt ping -s <IP_addresss>:<PORT> || (echo "Ping Failed"; exit 1)
sentieon licclnt query -s <IP_address>:<PORT> klib
```

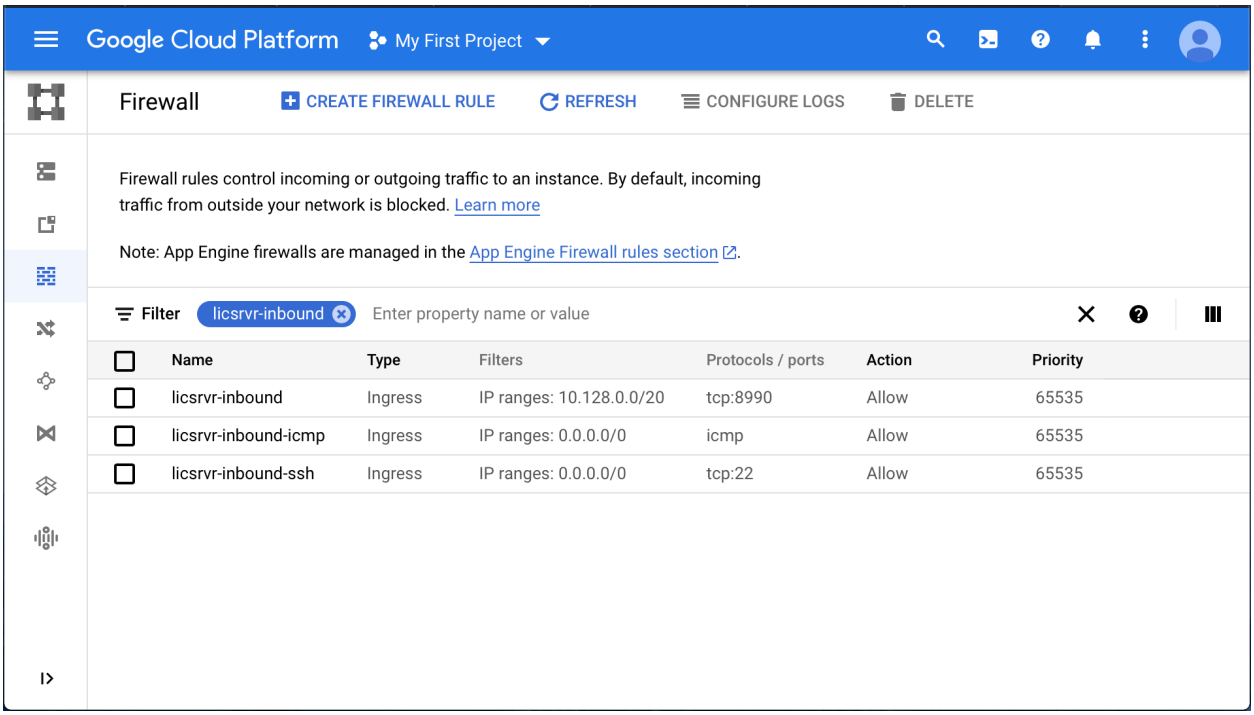


Fig. 12.10: Example license server inbound firewall rules

12.10 Description of output files and fields

12.10.1 Introduction

This document describes the output files of Sentieon® TNsnv, TNhaplotyper, TNhaplotyper2 and TNscope® algorithms and the meaning of the fields in those files. You can use the information in this document to better understand the files produced by Sentieon® tumor-normal variant calling software.

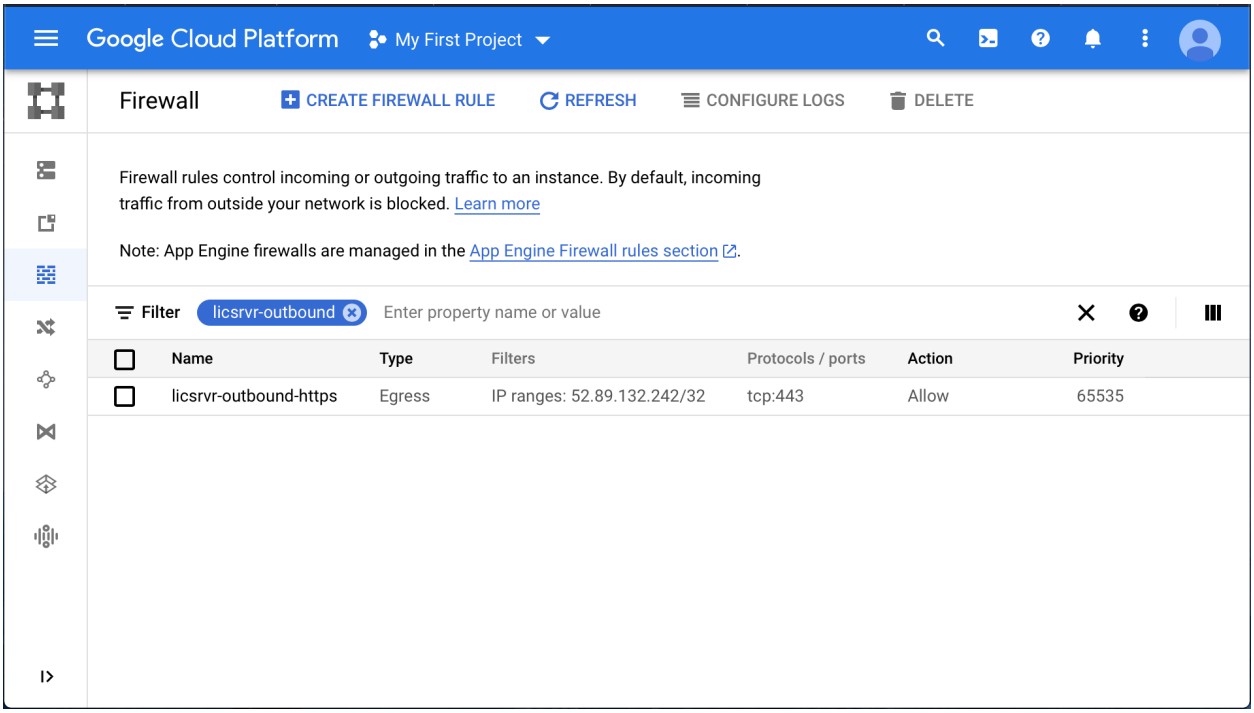


Fig. 12.11: Example license server outbound firewall rules

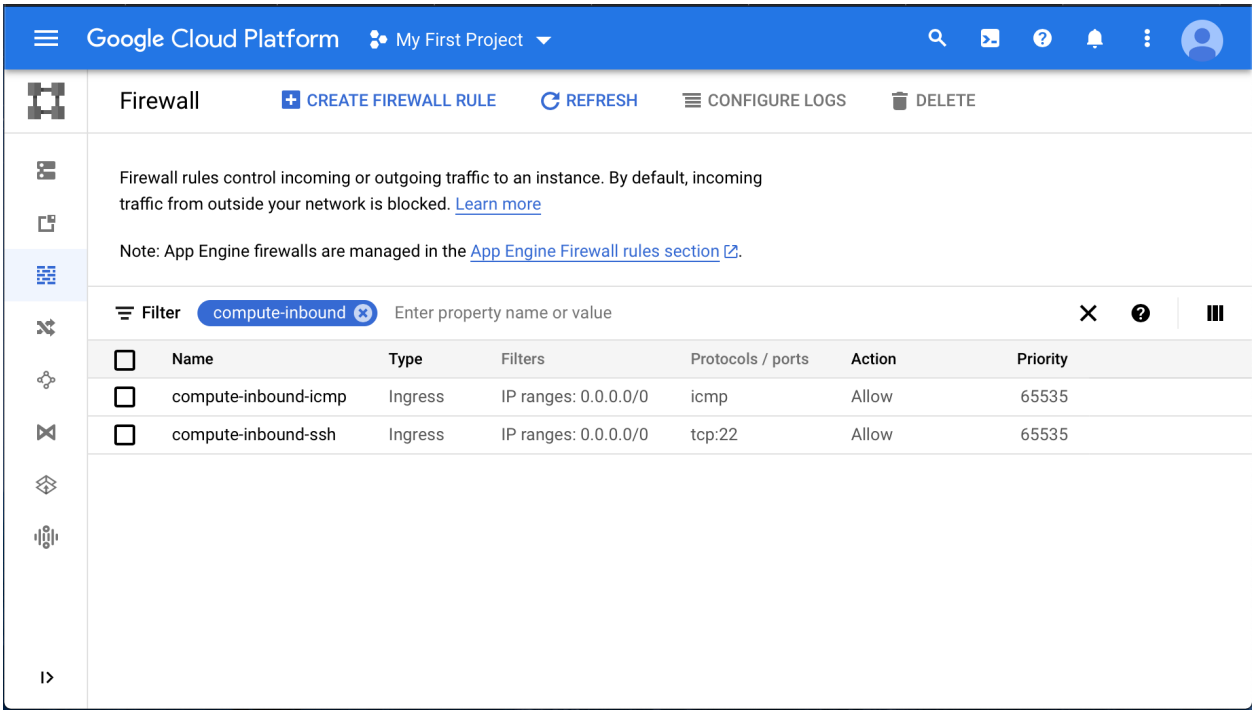


Fig. 12.12: Example compute nodes inbound firewall rule

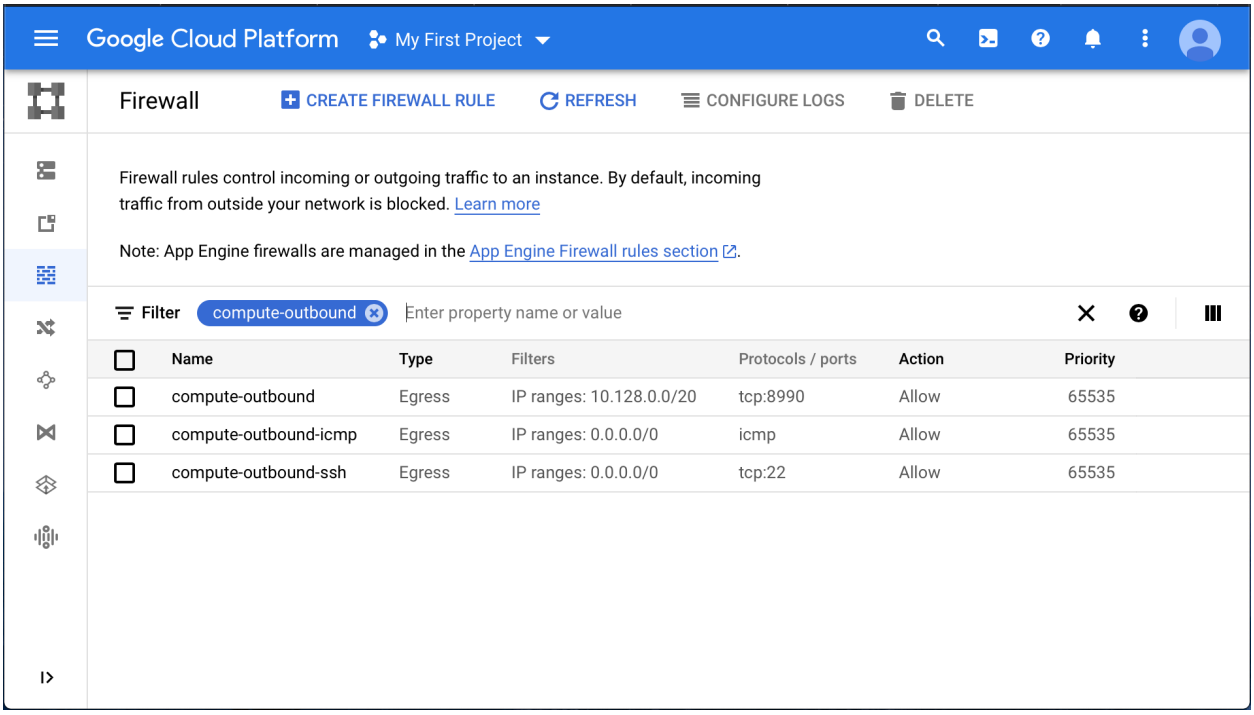


Fig. 12.13: Example compute nodes outbound firewall rules

12.10.2 TNsnv

12.10.2.1 Introduction

An example command with TNsnv is as follows

```
sentieon driver -t NUMBER_THREDS -r REFERENCE.FASTA \
   -i NORMAL_RECALED.BAM -i TUMOR_RECALED.BAM \
   --interval INTERVAL \
   --algo TNsnv --dbsnp DBSNP.VCF \
   --tumor_sample TUMOR_SM --normal_sample NORMAL_SM \
   -call_stats_out CALL_STATS_OUTPUT.TXT
   --stdcov_out STD_COVERAGE.TXT \     Standard coverage output file
   --q20cov_out Q20_COVERAGE.TXT \     Q20 coverage output file
   --power_out POWER.TXT --tumor_depth_out TUMOR_DP.TXT \
   --normal_depth_out NORMAL_DP.TXT OUTPUT.VCF
```

This command line produces the following required output files:

- **OUTPUT.VCF**

In addition, the following optional output files are produced:

- **CALL_STATS_OUTPUT.TXT**
- **STD_COVERAGE.TXT**
- **Q20_COVERAGE.TXT**
- **POWER.TXT**
- **TUMOR_DP.TXT**

- **NORMAL_DP.TXT**

The **OUTPUT.VCF** of TNsnv contains only limited output information. Users who desired a more detailed output format should examine the **CALL_STATS_OUTPUT.TXT** file.

12.10.2.2 OUTPUT.VCF

The **OUTPUT.VCF** file conforms to the VCF 4.2 specification. More information on the VCF format can be found at https://samtools.github.io/hts-specs/VCFv4.2.pdf. The INFO field annotations are described in detail below.

INFO annotation	Description
DB	The variant is present in the VCF file supplied with the –dbsnp option
MQ0	Total number of reads with Mapping Quality equal to 0
SOMATIC	The variant occurs uniquely in the sample supplied with the –tumor_sample option
VT	Variant type, can be SNP, INS or DEL

TNsnv also populates the FILTER field of the output VCF file. Variants are filtered using TNsnv internal quality filters. More information on the applied filters can be found in the failure_reasons row in the table in section 2.3.

FILTER	Description
PASS	The variant passes TNsnv internal quality filters
REJECT	The variant fails TNsnv internal quality filters

Standard genotype fields are defined by the format specification. However, TNsnv also outputs the following non-standard fields.

GENOTYPE field	Description
BQ	Average base quality of bases supporting the alternate alleles
FA	Fraction of reads supporting the alternate allele
SS	Status of the variant. Not currently implemented, always set to 2

12.10.2.3 CALL_STATS_OUTPUT.TXT

The **CALL_STATS_OUTPUT.TXT** file is a tab-separated text file with the following columns for each candidate variant. The core statistic of the software is t_lod_fstar which is a measurement of the support for the mutation relative to the expected level of sequencing noise at the candidate site.

Column	Description
Contig	The contig (chromosome) with the candidate
Position	The genomic coordinate of the candidate along the contig
Context	The sequence 3bp to either side of the candidate
Ref_allele	The reference allele at the candidate site
Alt_allele	The alternate allele at the candidate site
Tumor_name	The name of the tumor sample with the candidate mutation
Normal_name	The name of the paired normal sample
Score	Variant score. Not currently implemented, always set to 0.0

continues on next page

Table 12.40 – continued from previous page

Column	Description
Dbsnp_site	The variant is present in the VCF file supplied with the –dbsnp option (DBSNP) or is novel (NOVEL)
Covered	The site has sufficient read coverage to detect a variant with a 0.3 allele fraction at 80% power
Power	The product of tumor power and normal power, described below.
Tumor_power	The power to detect a mutation at a 0.3 allele fraction at the observed sequencing depth in the tumor sample
Normal_power	The power to detect a germline mutation at this site taking into account the presence of the site in dbSNP at the observed sequencing depth in the normal sample
Normal_power_nsp	The power to detect a germline mutation in the normal sample given that the mutation is not in dbSNP
Normal_power_wsp	The power to detect a germline mutation in the normal sample given that the mutation is in dbSNP
Total_reads	Total number of reads in both the tumor and normal samples at this site
Map_Q0_reads	Total number of reads in both the tumor and normal samples with mapping quality 0 at this site
Init_t_lod	Log odds of the likelihood that the candidate mutation is real over the likelihood that the candidate mutation is a sequencing error before any read-based filters are applied
t_lod_fstar	Log odds of the likelihood that the candidate mutation is real over the likelihood that the candidate mutation is a sequencing error
t_lod_fstar_forward	t_lod_fstar calculated using only reads on the forward strand
t_lod_fstar_reverse	t_lod_fstar calculated using only reads on the reverse strand
tumor_f	Estimated allele fraction of the candidate mutation in the tumor sample
Contaminant_fraction	Estimate of contamination of normal cells in the tumor sample
Contaminant_lod	Log odds of the likelihood that the candidate is contamination over the likelihood that the candidate is a sequencing error
t_q20_count	Count of the number of reads in the tumor sample with a base quality of at least 20
t_ref_count	Number of reads supporting the reference allele in the tumor sample
t_alt_count	Number of reads supporting the alternate allele in the tumor sample
t_ref_sum	Sum of the quality scores of the bases supporting the reference allele in the tumor sample
t_alt_sum	Sum of the quality scores of the bases supporting the alternate allele in the tumor sample
t_ref_max_mapq	The maximum mapping quality of tumor reads supporting the reference allele
t_alt_max_mapq	The maximum mapping quality of tumor reads supporting the alternate allele
t_ins_count	The number of reads in the tumor sample that have an insertion in the surrounding five bases
t_del_count	The number of reads in the tumor sample that have an insertion in the surrounding five bases
Normal_best_gt	The most likely genotype of the normal sample
Init_n_lod	Log odds of the likelihood that the normal sample is reference over the normal sample having the variant before any read-based filters are applied
normal_f	Estimated allele fraction of the candidate mutation in the normal sample
n_q20_count	Count of the number of reads in the normal sample with a base quality of at least 20
n_ref_count	Number of reads supporting the reference allele in the normal sample
n_alt_count	Number of reads supporting the alternate allele in the normal sample

continues on next page

Table 12.40 – continued from previous page

Column	Description
n_ref_sum	Sum of the quality scores of the bases supporting the reference allele in the normal sample
n_alt_sum	Sum of the quality scores of the bases supporting the alternate allele in the normal sample
power_to_detect_positive	The power to detect strand bias to the positive strand at the given sequencing depth
power_to_detect_negativ	The power to detect strand bias to the negative strand at the given sequencing depth
strand_bias_counts	A vector of counts for the tumor sample in the order of (tumor_ref_pos, tumor_ref_neg, tumor_alt_pos, tumor_alt_neg) where ref and alt specify the reference and alternate alleles and pos and neg specify the positive and negative strands. The numbers do not match those in earlier columns due to differential filtering
tumor_alt_fpir_median	Median position along forward strand reads for bases supporting the alternate allele in the tumor sample
tumor_alt_fpir_mad	Mean absolute deviation of the positions along forward strand reads for bases supporting the alternate allele in the tumor sample
tumor_alt_rpir_median	Median position along reverse strand reads for bases supporting the alternate allele in the tumor sample
tumor_alt_rpir_mad	Mean absolute deviation of the positions along reverse strand reads for bases supporting the alternate allele in the tumor sample
observed_in_normals_count	The number of reads supporting the candidate mutation in the normal sample
failure_reasons	Reasons for rejecting the candidate somatic mutation. Possibilities include: (1) alt_allele_in_normal - The alternate allele has significant support in the normal sample. (2) clustered_read_position - The alternate allele is not distributed evenly over the length of the read. (3) fstar_tumor_lod - the candidate does not have significant support above noise. (4) germline_risk - there is evidence for the mutation in the normal sample at a dbSNP site (5) nearby_gap_events - Insertion and deletion events were identified at the locus. (6) normal_lod - there is evidence for the mutation in the normal sample. (7) poor_mapping_region_alternate_allele_mapq - Low mapping quality for the alternate allele. (8) poor_mapping_region_mapq0 - Too many reads with a mapping quality of 0 at the locus. (9) possible_contamination - Possible contamination of the normal sample with tumor. (10) strand_artifact - The mutation is likely a strand bias artifact. (11) triallelic_site - The site is not biallelic.
judgement	The candidate is a true somatic variant (KEEP) or the candidate is not a likely somatic variant (REJECT).

12.10.2.4 STD_COVERAGE.TXT

A WIGGLE format file describing whether there is sufficient coverage to detect somatic variants at a 0.3 allele fraction in the tumor with 80% power. 1 indicates that the coverage at the locus passes this threshold, 0 otherwise.

12.10.2.5 Q20_COVERAGE.TXT

A WIGGLE format file describing whether there is sufficient coverage to detect somatic variants at a 0.3 allele fraction in the tumor with 80% power examining only bases with a quality of greater than 20. 1 indicates that the coverage at the locus passes this threshold, 0 otherwise.

12.10.2.6 POWER.TXT

A WIGGLE format file describing the power to detect a somatic variant at the observed coverage in the tumor and normal samples.

12.10.2.7 TUMOR_DP.TXT

A WIGGLE format file describing the observed sequence read depth in the tumor sample.

12.10.2.8 NORMAL_DP.TXT

A WIGGLE format file describing the observed sequence read depth in the normal sample.

12.10.3 TNhaplotyper

12.10.3.1 Introduction

An example command with TNhaplotyper is as follows

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i NORMAL_RECALED.BAM -i TUMOR_RECALED.BAM \
   --interval INTERVAL \
   --algo TNhaplotyper --dbsnp DBSNP.VCF \
   --tumor_sample TUMOR_SM --normal_sample NORMAL_SM \
   OUTPUT.VCF
```

This command line produces the following required output files:

- **OUTPUT.VCF**

12.10.3.2 OUTPUT.VCF

The **OUTPUT.VCF** file conforms to the VCF 4.2 specification. More information on the VCF format can be found at https://samtools.github.io/hts-specs/VCFv4.2.pdf. The INFO field annotations are described in detail below.

The core statistics of the software are TLOD, which is a measure of the support for the mutation relative to the expected level of sequencing noise at the candidate site, and NLOD, which is a measure of the odds that the mutation is absent from the normal sample.

INFO annotation	Description
DB	The variant is present in the VCF file supplied with the –dbsnp option
ECNT	Number of candidate variants in the active region, typically the number of candidate variants in the +/- 50 to 300 bp region
HCNT	Number of haplotypes observed in the active region after assembly of the sequence reads
MAX_ED	Maximum edit distance between the observed haplotypes in the active region
MIN_ED	Minimum edit distance between the observed haplotypes in the active region
NLOD	Log odds that the variant is not present in the normal sample (confidence that the variant is not a germline variant)
PON	Number of times the variant is observed in the panel of normal samples
RPA	The number of times the repeat is present for each allele for an indel within a short tandem repeat
RU	The sequence of the repeated nucleotides for an indel within a short tandem repeat
STR	The variant is an expansion or contraction of a short tandem repeat
TLOD	Log odds that the variant is present in the tumor sample relative to the expected noise

TNhaplotyper also populates the FILTER field for the variants.

FILTER	Description
PASS	The variant is confidently a somatic mutation
alt_allele_in_normal	The alternate allele is present in the paired normal sample and is unlikely to be a somatic variant
clustered_events	Multiple events are present on the same haplotype as the variant which is indicative of a false-positive call
germline_risk	There is evidence that the variant is present in the normal sample given that the variant is present in supplied dbSNP VCF and not present in the supplied COSMIC vcf
homologous_mapping_event	More than three events are present at this locus in the tumor which is indicate of a false-positive call
low_t_alt_frac	The variant is filtered due to a low alternate allele fraction in the tumor sample
multi_event_alt_allele_in	Multiple events are present in the tumor sample and the alternate allele appears in the normal sample
panel_of_normals	The mutation is present in at least two samples in the panel of normals.
str_contraction	The mutation is a contraction of a short tandem repeat
t_lod_fstar	The mutation does not have significant support above noise
triallelic_site	The mutation occurs at a triallelic site

Standard genotype fields are defined by the format specification. However, TNhaplotyper also outputs the following non-standard fields.

GENOTYPE	Description
AF	Fraction of reads supporting the alternate allele
ALT_F1R2	The number of reads in the F1R2 orientation supporting the alternate allele
ALT_F2R1	The number of reads in the F1R2 orientation supporting the alternate allele
FOXOG	The fraction of alt reads indicating OxoG error. OxoG error is induced by DNA oxidation during library preparation and is a frequent source of false-positive calls. See PMID: 23303777.
PGT	Physical phasing haplotype information describing how the alternate alleles are phased in relation to one another
PID	Physical phasing ID information, connecting records within a phasing group by using unique IDs within a given sample, but not across samples
QSS	Sum of base quality scores for each allele
REF_F1R2	The number of reads in the F1R2 orientation supporting the reference allele
REF_F2R1	The number of reads in the F2R1 orientation supporting the reference allele

12.10.4 TNhaplotyper2

12.10.4.1 Introduction

An example command with TNhaplotyper2 is as follows

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
   -i NORMAL_RECALED.BAM -i TUMOR_RECALED.BAM \
   --algo TNhaplotyper2 --tumor_sample TUMOR_SM \
   --normal_sample NORMAL_SM \
   TMP.VCF \
   --algo OrientationBias --tumor_sample TUMOR_SM \
```

(continues on next page)

(continued from previous page)

```
   ORIENTATION_DATA \
   --algo ContaminationModel --tumor_sample TUMOR_SM \
   --normal_sample NORMAL_SM \
   --vcf GERMLINE_RESOURCE \
   --tumor_segments CONTAMINATION_DATA.segments \
   CONTAMINATION_DATA

sentieon driver -r REFERENCE.FASTA \
   --algo TNfilter --tumor_sample TUMOR_SM \
   --normal_sample NORMAL_SM \
   -v TMP.VCF \
   --contamination CONTAMINATION_DATA \
   --tumor_segments CONTAMINATION_DATA.segments \
   --orientation_priors ORIENTATION_DATA \
   OUTPUT.VCF
```

This command line produces the following required output files:

- **OUTPUT.VCF**

12.10.4.2 OUTPUT.VCF

The **OUTPUT.VCF** file conforms to the VCF 4.2 specification. More information on the VCF format can be found at https://samtools.github.io/hts-specs/VCFv4.2.pdf. The INFO field annotations are described in detail below.

The core statistics of the software are TLOD, which is a measure of the support for the mutation relative to the expected level of sequencing noise at the candidate site, and NLOD, which is a measure of the odds that the mutation is absent from the normal sample.

INFO annotation	Description
AS_FilterStatus	The filter status of each allele, with alleles separated by the pipe character
AS_SB_TABLE	Forward and reverse read counts for each allele, with alleles separated by the pipe character
AS_UNIQ_ALT_RE	The number of reads with unique start and mate-end positions for each alternate allele
CONTQ	Phred-scaled probability that the variant alleles are not due to contamination
DP	Approximate read depth
ECNT	Number of candidate variants in the active region, typically the number of candidate variants in the +/- 50 to 300 bp region
GERMQ	The phred-scaled posterior probability that the alternate allele(s) are not germline variants
MBQ	Median base quality of each allele
MFRL	Median fragment length of each allele
MMQ	Median mapping quality of each allele
MPOS	Median distance from the end of the read for each alternate allele
NALOD	Negative log 10 odds of the variant being an an artifact in the normal sample with the same allele fraction as the tumor sample for each alternate allele
NCount	Count of N-bases in the read pileup
NLOD	Log odds that the variant is not present in the normal sample (confidence that the variant is not a germline variant) for each alternate allele
OCM	Number of reads supporting the alternate allele whose original alignment does not match the current contig
PON	The variant is found in the panel of normal samples
POPAF	Population allele frequency of the alternate alleles
ROQ	Phred-scaled probability that the variant alleles are not due to a read orientation artifact
RPA	The number of times the repeat is present for each allele for an indel within a short tandem repeat
RU	The sequence of the repeated nucleotides for an indel within a short tandem repeat
SEQQ	Phred-scaled probability that the variant alleles are not due to sequencing error
STR	The variant is an expansion or contraction of a short tandem repeat
STRANDQ	Phred-scaled probability of a strand-bias artifact
STRQ	Phred-scaled probability that the alternate alleles are errors due to polymerase slippage
TLOD	Log odds that the variant is present in the tumor sample relative to the expected noise

TNfilter also populates the FILTER field for the variants.

FILTER	Description
PASS	The site contains at least one allele that passes all filters
FAIL	All variant alleles are filtered, but for different reasons
base_qual	The median base quality of bases supporting the alternate allele is too low
clustered_events	Multiple events are present on the same haplotype as the variant which is indicative of a false-positive call
contamination	The alternate allele is present due to contamination
duplicate	The alternate allele is overrepresented by apparent sequencing duplicates
fragment	A large difference is observed in the median fragment length for reads supporting the reference and alternate alleles
germline	There is evidence that the variant is germline
haplotype	Variant is on the same haplotype as other filtered variants
low_allele_frac	The variant allele fraction is below the threshold
map_qual	The median mapping quality of reads supporting the alternate allele is too low
multiallelic	The mutation occurs at a multialleleic site
n_ratio	Too many 'N' bases at the site
normal_artifact	The variant is likely an artifact in the normal sample
orientation	The variant is likely an artifact due to orientation bias
panel_of_norm	The site is present in the panel of normals
position	The allele is close to the ends of the reads
slippage	The variant is likely an artifact due to polymerase slippage
strand_bias	Evidence for the alternate allele comes from only one read direction
strict_strand	Evidence for the alternate allele is not significant on both directions
weak_evidence	The mutation does not have significant support above noise

Standard genotype fields are defined by the format specification. However, TNhaplotyper2 also outputs the following non-standard fields.

GENOTYPE	Description
AF	Fraction of reads supporting the alternate allele
AD	Allelic depths for the reference and alternate alleles
DP	Approximate read depth
F1R2	The number of reads in the F1R2 orientation supporting each allele
F2R1	The number of reads in the F2R1 orientation supporting each allele
PGT	Physical phasing haplotype information describing how the alternate alleles are phased in relation to one another
PID	Physical phasing ID information, connecting records within a phasing group by using unique IDs within a given sample, but not across samples
PS	Phasing set; typically the position of the first variant in the set
SB	The forward and reverse read counts for the reference and alternate alleles

12.10.5 TNscope®

12.10.5.1 Introduction

An example command with TNscope® is as follows

```
sentieon driver -t NUMBER_THREADS -r REFERENCE.FASTA \
  -i NORMAL_RECALED.BAM -i TUMOR_RECALED.BAM \
  --interval INTERVAL \
  --algo TNscope --tumor_sample TUMOR_SM \
  --normal_sample NORMAL_SM --dbsnp DBSNP.VCF OUTPUT.VCF
```

This command line produces the following required output files:

- **OUTPUT.VCF**

12.10.5.2 OUTPUT.VCF

The **OUTPUT.VCF** file conforms to the VCF 4.2 specification. More information on the VCF format can be found at https://samtools.github.io/hts-specs/VCFv4.2.pdf. The INFO field annotations are described in detail below.

The core statistics of the software are TLOD, which is a measure of the support for the mutation relative to the expected level of sequencing noise at the candidate site, and NLOD, which is a measure of the odds that the mutation is absent from the normal sample.

INFO annotation	Description
CIEND	The confidence interval around the END position for imprecise structural variants
CIPOS	Confidence interval around POS for imprecise structural variants
DB	The variant is present in the VCF file supplied with the –dbsnp option
DPR	Average depth in the region surrounding the variant (+/-1bp)
ECNT	Number of candidate variants in the active region, typically the number of candidate variants in the +/- 50 to 300 bp region
END	The end position of the structural variant
FS	Phred-scale p-value using Fisher's exact test to detect strand bias
HCNT	The number of haplotypes observed in the active region after assembly of the sequence reads
IMPRECISE	The breakpoints of the structural variant are not precisely known
MATEID	Breakend mate
MAX_ED	Maximum edit distance between the observed haplotypes in the active region
MIN_ED	Minimum edit distance between the observed haplotypes in the active region
NLOD	Log odds that the variant is not present in the normal sample (confidence that the variant is not a germline variant)
NLODF	Log odds that the variant is not present in the normal sample (confidence that the variant is not a germline variant) given the allele fraction in the tumor sample
PON	Number of times the variant is observed in the panel of normal samples
PV	The p-value from a Fisher's exact test of the number of reads supporting the reference and alternate alleles in the tumor and normal samples
PV2	The p-value from a Fisher's exact test of the number of reads supporting the reference and alternate alleles in the tumor and normal samples using only high-confidence reads
RPA	The number of times the repeat is present for each allele for an indel within a short tandem repeat
RU	The sequence of the repeated nucleotides for an indel within a short tandem repeat
SOMATIC	The variant occurs uniquely in the sample supplied with the –tumor_sample option
SOR	Symmetric Odds Ratio of 2x2 contingency table to detect strand bias
STR	The variant is an expansion or contraction of a short tandem repeat
SVLEN	The difference in length between REF and ALT alleles of structural variants
SVTYPE	The type of structural variant
TLOD	Log odds that the variant is present in the tumor sample relative to expected noise
VAF	The variant allele frequency. The fraction of reads supporting the alternate allele in the tumor sample.

TNscope® also populates the FILTER field for the variants.

FILTER	Description
PASS	The variant is confidently a somatic mutation
alt_allele_in_normal	The alternate allele is present in the paired normal sample and is unlikely to be a somatic variant
clustered_events	Multiple events are present on the same haplotype as the variant which is indicative of a false-positive call
germline_risk	There is evidence that the variant is present in the normal sample given that the variant is present in supplied dbSNP VCF and not present in the supplied COSMIC vcf
homologous_mapping_event	More than three events are present at this locus in the tumor which is indicate of a false-positive call
low_t_alt_frac	The variant is filtered due to a low alternate allele fraction in the tumor sample
multi_event_alt_allele_in	Multiple events are present in the tumor sample and the alternate allele appears in the normal sample
panel_of_normals	The mutation is observed in at least two samples in the panel of normals
str_contraction	The mutation is a contraction of a short tandem repeat
t_lod_fstar	The mutation does not have significant support above noise
triallelic_site	The mutation occurs at a triallelic site

Standard genotype fields are defined by the format specification. However, TNscope® also outputs the following non-standard fields.

GENOTYPE field	Description
AD	Allele depths for the ref and alt alleles
AF	Fraction of reads supporting the alternate allele
AFDPLOWMQ	Read depth used to calculate AF including reads with low mapping quality
AFLOWMQ	Allele fraction of the event in the tumor including low mapq reads
ALT_F1R2	The number of reads in the F1R2 orientation supporting the alternate allele
ALT_F2R1	The number of reads in the F1R2 orientation supporting the alternate allele
ALTHC	Depth of reads supporting the highest confidence alternate allele
ALTHCLOWMQ	Depth of reads supporting the highest confidence alternate allele including reads with low mapping quality
BaseQRankSumPS	Z-score from Wilcoxon rank sum test of Alt vs. Ref base qualities per sample
ClippingRankSumPS	Z-score from Wilcoxon rank sum test of Alt vs. Ref number of hard clipped bases per sample
DPHC	Depth of high-confidence reads supporting the reference or alternate allele
DPHCLOWMQ	Depth of high-confidence reads supporting the reference or alternate allele including reads with low mapping quality
FOXOG	The fraction of alt reads indicating OxoG error. OxoG error is induced by DNA oxidation during library preparation and is a frequent source of false-positive calls. See PMID: 23303777.
MQRankSumPS	Z-score from Wilcoxon rank sum test of Alt vs. Ref read mapping qualities per sample
NBQPS	Mean Neighboring Base Quality, including 5bp on both sides per sample
PGT	Physical phasing haplotype information, describing how the alternate alleles are phased in relation to one another
PID	Physical phasing ID information, connecting records within a phasing group by using unique IDs within a given sample, but not across samples
QSS	Sum of base quality scores for each allele
ReadPosEndDistPS	Z-score from Wilcoxon rank sum test of mean distance from either end of read per sample
ReadPosRankSumPS	Z-score from Wilcoxon rank sum test of Alt vs. Ref read position bias per sample
REF_F1R2	The number of reads in the F1R2 orientation supporting the reference allele
REF_F2R1	The number of reads in the F2R1 orientation supporting the reference allele

12.11 Recommendations on Read Groups

12.11.1 Introduction

This documents describes the recommended usage of the RGID fields to minimize potential problems using the Sentieon® Genomics software.

This document should help you determine the best practices for setting the different fields in the RG tags of the bam files used.

12.11.2 Detailed description of the RG fields and its usage

12.11.2.1 Detailed description of the RG field

The SAM format specification http://samtools.github.io/hts-specs/SAMv1.pdf defines the Read Group as an identifier that groups reads together. The Read Group field in the BAM file can contain the following tags:

- `ID`: IDentifier. A unique identifier for the Read Group. You need to make sure that the RG-ID is unique within the BAM file, and within multiple BAM files that will be used in the same command in a pipeline.

This field is required.

- `CN`: Center Name. Name of the sequencing center that sequenced the reads in the Read Group. Typically this tag is not used.
- `DS`: DeScription. Freeform description of the Read Group. Typically this tag is not used.
- `DT`: DaTe. Date the run was produced, following ISO8601 date or date/time. Typically this tag is not used.
- `FO`: Flow Order. The array of nucleotide bases that correspond to the nucleotides used for each flow of each read. Typically this tag is not used.
- `KS`: Key Sequence. The array of nucleotide bases that correspond to the key sequence of each read. Typically this tag is not used.
- `LB`: LiBrary. The library used to sequence the reads.
- `PG`: ProGram. The programs used for processing the read group. Typically the information is included in the PG field of the BAM file, instead of doing this within each Read Group.
- `PI`: Predicted median Insert size. Typically this tag is not used.
- `PL`: PLatform. The technology used to sequence the reads. This tag is required if you plan on running BQSR, as it is used to determine the correct error model to apply.
- `PM`: Platform Model. Freeform text providing further details of the platform/technology used. Typically this tag is not used.
- `PU`: Platform Unit. Unique identifier for the sequencer unit used to perform the sequencing. This tag is recommended if you plan on running BQSR, as BQSR will model together all reads belonging to the same PU; if the PU is missing, BQSR will model together reads with the same RG-ID.
- `SM`: Sample name. The sample the reads belong to. This field is required.

12.11.2.2 RG field tags and Sentieon®

The following are general principles of how RG field tags are used with the Sentieon® tools:

- When using multiple input bam files, the ID tags of the bam files need to be unique; there cannot be a RG with the same ID in two different bam input files.
- The tools use the SM tag to identify the reads that belong to the same sample and process them accordingly.
- The Deduplication uses the LB tag to determine which groups may contain duplicates, duplicate reads need to belong to the same library.
- The BQSR model requires the PL tag to determine the error model to apply. If no PL tag is present, the BQSR will not be performed.
- The BQSR modeling will be performed independently on groups of reads identified by the PU tag if it is present; if the PU tag is not present, the BQSR modeling will be performed independently on groups of reads identified by the ID tag.

12.11.3 Recommendation on how to fill in the RG fields

Sentieon® recommends using the following conventions for the RG field tags:

- `ID`: sample_name.flowcell.lane.barcode
- `SM`: sample_name
- `PL`: technology, i.e. ILLUMINA

- PU: flowcell.lane
- LB: sample_name.library_preparation

The above recommendation makes sure that:

- The read group ID will be unique even across multiple bam files, even for the same sample sequenced in different lanes or using different libraries.
- The BQSR will create a recalibration based on the actual unique sequencing unit, and can be performed on multiple samples if they are sequenced on the same sequencing unit.
- The tumor and normal sample names will be unique for somatic variant calling.

13 Introduction

13.1 Description

Sentieon® Genomics software is a set of software tools that perform analysis of genomic data obtained from DNA sequencing.

13.2 Benefits and Value

The Sentieon® Genomics software enables rapid and accurate analysis of next-generation sequence data. The Sentieon® DNAseq pipelines enable germline variant calling on hundreds of thousands of samples simultaneously. The Sentieon® TNseq pipelines enable accurate calling of somatic variants in paired tumor-normal samples or in an unpaired tumor samples. The Sentieon® Genomics software produces more accurate results than other tools on third-party benchmarks with results obtained in one tenth the time of comparable pipelines.

13.3 Platform Requirements

Sentieon® Genomics software is designed to run on Linux and other POSIX-compatible platforms.

For Linux systems, we recommended using the following Linux distributions or higher: RedHat/CentOS 6.5, Debian 7.7, OpenSUSE-13.2, or Ubuntu-14.04.

Sentieon® Genomics software requires at least 16GB of memory, although we recommend using 64 GB of memory. The alignment step is typically the most memory consuming stage; you can check section *Controlling memory usage in BWA* for ways to control the memory used in alignment.

Sentieon® Genomics software processes file data at speeds around 20-30MB/s per core using a state of the art server. In order to take advantage of the maximum speed that the software can provide, we recommend that you use high-speed SSD hard drives in your system, preferably two identical drives in a high speed RAID 0 striped volume configuration. When using a RAID 0 configuration, it is advised to use the drives to only store intermediate files, and move out any final results or important information out of those hard drives; this way, your server should have additional storage space to store the results.

Sentieon® Genomics software requires that your system have either python2.6.x, python2.7.x, or python3.x. You can check the version of the default python in your system by running:

```
python --version
```

If neither python2.6.x, python2.7.x, or python3.x is not the default python version in your system, you will need to install it and set an environmental variable to tell the software where the python binary is located

```
export SENTIEON_PYTHON=Python_binary_location
```

13.4 Installation procedure for a Linux Based system

In order to install the Sentieon® Genomics software in a Linux based system, you need to follow these installation instructions:

1. Obtain the software release package. The software release package is named sentieon-genomics-202503.02.tar.gz, where 202503.02 is the release version.
2. Obtain the license file from Sentieon®. The license file has extension .lic. The Sentieon® license control is based on a lightweight floating license server process running on a node, and serving licenses though TCP. In a HPC cluster, the license server process is typically running in a special non-computing node on the cluster periphery that has unrestricted access to the outside world through HTTPS, and serves the licenses to the rest of the nodes in the cluster by listening to a specific TCP port that needs to be open within the cluster. The license server needs to have external https access to validate the license, while the computing client nodes do not need to have access to the internet. You need to provide Sentieon® with the hostname (FQDN or IP) and port where you plan on deploying the license server.
3. Upload the software release package and license file to your server.
4. Decompress the release package with the command below. This will create a folder named sentieon-genomics-202503.02.

   ```
   tar xvzf sentieon-genomics-202503.02.tar.gz
   ```

5. Copy the license file to the directory that stores the license files (`LICENSE_DIR`). If you are running a license server process, start the license server.

   ```
   <SENTIEON_FOLDER>/bin/sentieon licsrvr --start LICENSE_DIR/LICENSE_FILE.lic
   ```

6. Set an environment variable in your system to indicate where the license is located (either `LICSRVR_HOST:LICSRVR_PORT` if you are using a license server or `LICENSE_DIR/LICENSE_FILE.lic` if you are using a license file)

   We recommend that you add this line to the shell scripts you use to run the Sentieon® software, or to your shell profile:

   ```
   #if you are using a license server
   export SENTIEON_LICENSE=LICSRVR_HOST:LICSRVR_PORT
   #if you are using a license file
   export SENTIEON_LICENSE=LICENSE_DIR/LICENSE_FILE.lic
   ```

13.5 Useful environmental variables when using the software

When using Sentieon® Genomics software there are certain environmental variables that may be useful or are required:

1. *SENTIEON_LICENSE*: tells the software where the license is located. This variable is required.

```
#if you are using a license server
export SENTIEON_LICENSE=LICSRVR_HOST:LICSRVR_PORT
#if you are using a license file
export SENTIEON_LICENSE=LICENSE_DIR/LICENSE_FILE.lic
```

2. *SENTIEON_PYTHON*: tells the software the location of the python binary. This variable is useful when python2.6.x, python2.7.x, or python3.x is not the default python version in your system.

```
export SENTIEON_PYTHON=Python_binary_location
```

3. *SENTIEON_TMPDIR*: tells the software where to store the intermediate temporary files. This variable is recommended and should point to a fast storage location to prevent the software from being I/O limited; this is specially important when using a slow NFS storage for input/output. When using this variable, the software will create a unique temporary subfolder in SENTIEON_TMPDIR for each command, which will reduce contention if multiple commands output to the same folder.

```
export SENTIEON_TMPDIR=/local_fast_scratch
```

4. Environmental settings to use jemalloc memory allocation: this is recommended in systems with large memory, or a large number of CPUs (more than 32 vCPU). Please check the *jemalloc appnote* for more information.

13.6 Keywords

Bioinformatics, DNA sequencing, Genomics.

14 DRIVER binary

DRIVER is the binary used to execute all stages of the bioinformatics pipeline. A single call to the driver binary can run multiple algorithms; for example, the metrics stage is implemented as a single command call to driver running multiple algorithms.

14.1 DRIVER syntax

The general syntax of the DRIVER binary is:

```
sentieon driver OPTIONS --algo ALGORITHM ALGO_OPTION OUTPUT \
[--algo ALGORITHM2 ALGO_OPTION2 OUTPUT2]
```

In the case of running multiple algorithms in the same driver call, the OPTIONS are shared among all the algorithms.

The arguments (OPTIONS) for this command include:

- `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The argument is optional; if omitted, the driver tool will use as many threads as the server has.
- `-r REFERENCE`: location of the reference FASTA file. This argument is required in all algorithms except LocusCollector and Dedup without consensus.
- `-i INPUT_FILE`: location of the BAM input file. This argument can be used multiple times to have the software use multiple input files and the result would be the same as if the input BAM files are merged into one BAM file. Sentieon® only support bam files that are sorted by coordinates, the `@HD` line with `SO:coordinate` attribute should be present. Sentieon® also requires the input bam files have `RG` tag in the alignment section and a corresponding `@RG` line with matching ID tag defined in the header. This argument is required in all algorithms except for `CollectVCMetrics`, `GVCFtyper`, `DNAModelApply`, `SVSolver`, `TNModelApply`, `VarCal`, `ApplyVarCal` and `TNfilter`.
- `-q QUALITY_RECALIBRATION_TABLE`: location of the quality recalibration table output from the BQSR stage that will be used as an input during the BQSR stage and Variant Calling stage. This argument can be used multiple times to have the software use multiple input calibration tables; each calibration table will apply a recalibration to BAM files by matching the readgroup of the reads
- `--interval INTERVAL`: interval in the reference that will be used in the software. This argument can be used multiple times to perform the calculation on the union of all the intervals. `INTERVAL` can be specified as:

  - CONTIG[:START-END]: calculation will be done only on the corresponding contig. START and END are optional numbers to further reduce the interval; both are first-base-1 based. You can input a comma-separated list of multiple contigs.
  - BED_FILE: location of the BED file containing the intervals. Providing an empty file will result in no processing done, as if the interval had 0 length. A compressed BGZF file is also supported.
  - PICARD_INTERVAL_FILE: location of the file containing the intervals, following the Picard interval standard. Providing an empty file will result in no processing done, as if the interval had 0 length. A compressed BGZF file is also supported.
  - VCF_FILE: location of VCF containing variant records whose genomic coordinates will be used as intervals. A compressed VCF.gz file is also supported.
- --interval_padding PADDING_SIZE: adds PADDING_SIZE bases padding to the edges of the input intervals. The default value is 0.
- --help: option to display help. The option can be used together with the --algo ALGORITHM to display help for the specific algorithm.
- --read_filter FILTER,OPTION=VALUE,OPTION=VALUE: perform a filter or transformation of reads prior to the application of the algorithm. Please refer to *DRIVER read_filter options* to get additional information on the available filters and their functionality.
- --temp_dir DIRECTORY: determines where the temporary files will be stored. The default is the folder where the command is run ($PWD).
- --skip_no_coor: determines whether to skip unmapped reads.
- --cram_read_options decode_md=0: CRAM input option to turn off the NM/MD tag in the input CRAM.
- --replace_rg ORIG_RG="NEW_RG_STRING": modifies the @RG Read group tag of the next BAM input file to update the ORIG_RG with the new information; the NEW_RG_STRING needs to be a valid and complete string. This argument can be used multiple times, and each will affect the next -i BAM input. You can check *Modify RG information on BAM files when both Tumor and Normal inputs have the same RGID* for a detail example on its usage

The supported algorithms (ALGORITHM) for this command are:

- LocusCollector, Dedup: used in the remove duplicates stage.
- Realigner: used in the indel realignment stage.
- QualCal: used in the base quality recalibration stage.
- MeanQualityByCycle, QualDistribution, GCBias, AlignmentStat, InsertSizeMetricAlgo, HsMetricAlgo, CoverageMetrics, BaseDistributionByCycle, QualityYield, WgsMetricsAlgo, SequenceArtifactMetricsAlgo: used to calculate QC metrics.
- Genotyper, Haplotyper: used for germline variant calling analysis.
- DNAscope and DNAModelApply: used in DNAscope germline variant calling analysis.
- DNAscope and SVsolver: used in the DNAscope germline structural variant analysis.
- DNAscope and VariantPhaser: used in the DNAscope germline variant calling analysis for PacBio® HiFi® reads.
- ReadWriter: used to output the BAM file after base quality recalibration or merge multiple BAM files.
- VarCal, ApplyVarCal: used in the VQSR stage.
- GVCFtyper: used for germline joint variant calling analysis.

- TNsnv, TNhaplotyper, TNhaplotyper2, ContaminationModel, OrientationBias, TNfilter: used for somatic tumor-normal or tumor only analysis.
- TNscope, TNModelApply: used for somatic tumor-normal somatic and structural variant analysis.
- RNASplitReadsAtJunction: used in the stage to split RNA reads at junctions.
- ContaminationAssessment: used to identify cross-sample contamination from BAM files.
- CollectVCMetrics: used to calculate post-variant calling metrics.

14.2 DRIVER ALGORITHM syntax

14.2.1 LocusCollector ALGORITHM

The LocusCollector algorithm collects read information that will be used for removing duplicate reads.

The input to the LocusCollector algorithm is a BAM file; its output is the score file indicating which reads are likely duplicates.

The LocusCollector algorithm requires the following ALGO_OPTION:

- `--fun SCORE`: scoring function to use. Possible values for SCORE are:
  - score_info: calculates the score of a read pair for the Dedup algorithm. This is the default score function.
- `--consensus`: set this option to turn on consensus deduplication.
- `--umi_tag TAG`: Logic UMI tag for UMI barcode aware deduplication. The default value is None.
- `--umi_ecc_dist DISTANCE`: UMI barcode error correction distance. Set it to 0 to turn off the barcode error correction. The default value is 1.
- `--rna`: set this option for RNA sequence data aligned with STAR.

14.2.2 Dedup ALGORITHM

The Dedup algorithm performs the marking/removing of duplicate reads.

The input to the Dedup algorithm is a BAM file; its output is the BAM file after removing duplicate reads.

The Dedup algorithm requires the following ALGO_OPTION:

- `--score_info LOCUS_COLLECTOR_OUTPUT`: location of the output file of the LocusCollector command call.

The Dedup algorithm accepts the following optional ALGO_OPTION:

- `--rmdup`: set this option to remove duplicated reads in the output BAM file. If this option is not set, the duplicated reads will be marked as such with a flag, but not removed from the BAM file.
- `--cram_write_options compressor=[gzip|bzip2|lzma|rans|tok|fqz|arith]`: CRAM output compression options. compressor=gzip+rans is default if not defined.
- `--cram_write_options version=[3.0|3.1]`: CRAM output version options. version=3.0 is default if not defined.
- `--metrics METRICS_FILE`: location and filename of the output file containing the metrics data from the deduping stage.
- `--optical_dup_pix_dist DISTANCE`: determine the maximum distance between two duplicate reads for them to be considered optical duplicates. The default value is 100.

- `--bam_compression COMPRESSION_LEVEL[0-9]`: gzip compression level for the output BAM file. The default value is 6.
- `--output_dup_read_name`: when using this option, the output of the command will not be a BAM file with marked/removed duplicates, but a list of read names for reads that were marked as duplicate.
- `--dup_read_name DUPLICATE_READ_NAME_FILE`: when using this input all reads contained in the DUPLICATE_READ_NAME_FILE will be marked as duplicate, regardless of whether they are primary or non-primary.

14.2.3 Realigner ALGORITHM

The Realigner algorithm performs the indel realignment.

The input to the Realigner algorithm is a BAM file; its output is the BAM file after realignment.

The Realigner algorithm accepts the following optional ALGO_OPTION:

- `-k KNOWN_SITES`: location of the VCF file used as a set of known sites. The known sites will be used to help identify likely sites where the realignment is necessary; only indel variants in the file will be used. You can include multiple collections of known sites by specifying multiple files and repeating the `-k KNOWN_SITES` option.
- `--interval_list INTERVAL`: interval in the reference that will be used in the calculation of the realign targets. Only a single input INTERVAL will be considered: if you repeat the –interval_list option, only the INTERVAL in the last one will be considered. INTERVAL can be specified as:
  - BED_FILE: location of the BED file containing the intervals.
  - PICARD_INTERVAL_FILE: location of the file containing the intervals, following the Picard interval standard.
- `--bam_compression COMPRESSION_LEVEL[0-9]`: gzip compression level for the output BAM file. The default value is 6.
- `--cram_write_options compressor=[gzip|bzip2|lzma|rans|tok|fqz|arith]`: CRAM output compression options. compressor=gzip+rans is default if not defined.
- `--cram_write_options version=[3.0|3.1]`: CRAM output version options. version=3.0 is default if not defined.

14.2.4 QualCal ALGORITHM

The QualCal algorithm calculates the recalibration table necessary to do the BQSR. The QualCal algorithm also applies the recalibration to calculate the data required to create a report, and creates the data required to create a report.

The recalibration math depends on platform (PL) tag of the ReadGroup; the QualCal algorithm supports the following platforms: ILLUMINA, ION_TORRENT, LS454, PACBIO, COMPLETE_GENOMICS, DNBSEQ. Support for sequencing data from the SOLID platform is not currently implemented.

The input to the QualCal algorithm is a BAM file; its output is a recalibration table or the csv file containing the data required to create a report.

The QualCal algorithm accepts the following optional ALGO_OPTION:

- `-k KNOWN_SITES`: location of the VCF file used as a set of known sites. The known sites will be used to make sure that known locations do not get artificially low quality scores by misidentifying true variants errors in the sequencing methodology. You can include multiple collections of known sites by specifying multiple files and repeating the `-k KNOWN_SITES` option. We strongly recommend using as many known sites as possible, as otherwise the recalibration will consider variant sites to be sequencing errors.

- `--plot`: indicates whether the command is being used to generate the data required to create a report.
- `--cycle_val_max`: maximum allowed cycle value for the cycle covariate.
- `--before RECAL_TABLE`: location of the previously calculated recalibration table; it will be used to apply the recalibration.
- `--after RECAL_TABLE.POST`: location of the previously calculated results of applying the recalibration table; it will be used to calculate the data required to create a report.

14.2.5 MeanQualityByCycle ALGORITHM

The MeanQualityByCycle algorithm calculates the mean base quality score for each sequencing cycle.

The input to the MeanQualityByCycle algorithm is a BAM file; its output is the metrics data.

The MeanQualityByCycle algorithm does not accept any ALGO_OPTION.

14.2.6 QualDistribution ALGORITHM

The QualDistribution algorithm calculates the number of bases with a specific base quality score.

The input to the QualDistribution algorithm is a BAM file; its output is the metrics data.

The QualDistribution algorithm does not accept any ALGO_OPTION.

14.2.7 GCBias ALGORITHM

The GCBias algorithm calculates the GC bias in the reference and the sample.

The input to the GCBias algorithm is a BAM file; its output is the metrics data.

The GCBias algorithm accepts the following optional ALGO_OPTION:

- `--summary SUMMARY_FILE`: location and filename of the output file summarizing the GC Bias metrics.
- `--accum_level LEVEL`: determines the accumulation levels. The possible values of LEVEL are ALL_READS, SAMPLE, LIBRARY, READ_GROUP. The default value is ALL_READS.
- `--also_ignore_duplicates`: determines whether the output metrics will be calculated using unique non duplicated reads.
- `--is_bisulfite_seq`: set this option to indicate the input BAM file consists of bisulfite sequenced reads.

14.2.8 HsMetricAlgo ALGORITHM

The HsMetricAlgo algorithm calculates the Hybrid Selection specific metrics for the sample and the AT/GC dropout metrics for the reference.

The input to the HsMetricAlgo algorithm is a BAM file; its output is the metrics data.

The HsMetricAlgo algorithm requires the following ALGO_OPTION:

- `--targets_list TARGETS_FILE`: location and filename of the interval list input file that contains the locations of the targets.
- `--baits_list TARGETS_FILE`: location and filename of the interval list input file that contains the locations of the baits used.
- `--clip_overlapping_reads`: determines whether to clip overlapping reads during the calculation.
- `--min_map_qual QUALITY`: determines the filtering quality of the reads used. Any reads with mapping quality less than QUALITY will be filtered out.

- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored.
- `--coverage_cap COVERAGE`: determines the maximum coverage limit used in the histogram.

14.2.9 AlignmentStat ALGORITHM

The AlignmentStat algorithm calculates statistics about the alignment of the reads.

The input to the AlignmentStat algorithm is a BAM file; its output is the metrics data.

The AlignmentStat algorithm accepts the following optional ALGO_OPTION:

- `--adapter_seq SEQUENCE_LIST`: the sequence of the adapters used in the sequencing, provided as a comma separated list. The default value is the list of default Illumina adapters.
- `--is_bisulfite_seq`: set this option to indicate the input BAM file consists of bisulfite sequenced reads.

14.2.10 InsertSizeMetricAlgo ALGORITHM

The InsertSizeMetricAlgo algorithm calculates the statistical distribution of insert sizes.

The input to the InsertSizeMetricAlgo algorithm is a BAM file; its output is the metrics data.

The InsertSizeMetricAlgo algorithm accepts the following optional ALGO_OPTION:

- `--deviation DEVIATION`: Maximum multiples of standard deviation before the histogram is trimmed down. The default value is 10.0.
- `--hist_width HISTOGRAM\_WIDTH`: sets the HISTOGRAM_WIDTH to override the automatic truncation of histogram tail. The default value is 0.
- `--min_read_ratio RATIO`: Minimum ratio of reads for a read category to be included in the histogram. The default value is 0.05.
- `--se_tag TAG`: include single ended reads with the TAG in histogram. The default value is `MI`.

14.2.11 CoverageMetrics ALGORITHM

The CoverageMetrics algorithm calculates the depth coverage of the BAM file. The coverage is aggregated by interval if the `--interval` option is included at the driver level; if no `--interval` option is included, the aggregation per interval will be done per contig for all bases in the reference and the per locus coverage output file will be about 60GB. The coverage is aggregated by gene if a RefSeq file is included.

The input to the CoverageMetrics algorithm is a BAM file, it is recommended to use the after Deduplication BAM file for this analysis; its outputs are files containing the metrics data organized by partition, aggregation and output. Possible outputs are:

- summary: contains the depth data.
- statistics: contains the histogram of loci with specific depth.
- cumulative_coverage_counts: contains the histogram of loci with depth larger than x.
- cumulative_coverage_proportions: contains the normalized histogram of loci with depth larger than x.

Examples of output files when the output name is OUTPUT:

- OUTPUT: the per locus coverage with no partition.
- OUTPUT.sample_summary: the summary for PARTITION_GROUP sample, aggregated over all bases.
- OUTPUT. library_interval_statistics: the statistics for PARTITION_GROUP library, aggregated by interval.

The CoverageMetrics algorithm accepts the following optional ALGO_OPTION:

- `--partition PARTITION_GROUP`: determine how to partition the data. Possible values are readgroup or a comma separated combination of the RG attributes, namely sample, platform, library, center. The default value is "sample". You can include multiple partition groups by repeating the `--partition` option, and each ouput file will be created once per `--partition` option.
- `--gene_list REFSEQ_FILE`: location of the RefSeq file used to aggregate the results of the CoverageMetrics algorithm to the gene level.
- Filtering options:
  - `--min_map_qual MAP_QUALITY`: determines the filtering quality of the reads used. Any reads with mapping quality less than QUALITY will be filtered out.
  - `--max_map_qual MAP_QUALITY`: determines the filtering quality of the reads used. Any reads with mapping quality larger than QUALITY will be filtered out.
  - `--min_base_qual QUALITY`: determines the filtering quality of the bases used. Any base with quality less than QUALITY will be ignored.
  - `--max_base_qual QUALITY`: determines the filtering quality of the bases used. Any base with quality larger than QUALITY will be ignored.
  - `--cov_thresh THRESHOLD`: add percentage of bases in the aggregation that have coverage larger than the threshold. You can include multiple thresholds by repeating the `--cov_thresh` argument.
- Omit output options:
  - `--omit_base_output`: skip the output of the per locus coverage with no partition. This option can be used when you do not use intervals to save space.
  - `--omit_sample_stat`: skip the output of summary results aggregated over all bases (_summary)
  - `--omit_locus_stat`: skip the output of all histogram files (both _cumulative_coverage_counts and _cumulative_coverage_proportions).
  - `--omit_interval_stat`: skip the output of all interval statistics files (_interval_statistics).
- `--count_type TYPE`: determines how to deal with overlapping reads from the same fragment. Possible options are:
  - 0: to count overlapping reads even if they come from the same fragment. This is the default value.
  - 1: to count overlapping reads
  - 2: to count overlapping reads only if the reads in the fragment have consistent bases.
- `--print_base_counts`: include the number of "AGCTND" in the output per locus coverage with no partition.
- `--include_ref_N`: include the coverage data in loci where the reference genome is set to N.
- `--ignore_del_sites`: ignore the coverage data in loci where there are deletions.
- `--include_del`: this argument will interact with others as follows:
  - if ignore_del_sites is off, count Deletion as depth
  - if print_base_counts is on, include number of 'D'
- `--histogram_scale [log/linear] --histogram_low MIN_DEPTH, --histogram_high MAX_DEPTH, --histogram_bin_count NUM_BINS`: determine the scale type, bin and sizes for histograms. The default values are log, 1, 500, 499.

14.2.12 CollectVCMetrics ALGORITHM

The CollectVCMetrics algorithm collects metrics related to the variants present in the input VCF.

The input to the CollectVCMetrics algorithm is a VCF file and a DBSNP file; its output is a pair of files containing information about the variants from the VCF file.

The CollectVCMetrics algorithm requires the following ALGO_OPTION:

- `-d dbSNP_FILE`: location of the Single Nucleotide Polymorphism database (dbSNP). Only one file is supported.
- `-v INPUT`: location of the VCF file on which the metrics will be calculated. Only one file is supported.

14.2.13 BaseDistributionByCycle

The BaseDistributionByCycle algorithm calculates the nucleotide distribution per sequencer cycle.

The input to the BaseDistributionByCycle algorithm is a BAM file; its output is the metrics data.

The BaseDistributionByCycle algorithm accepts the following optional ALGO_OPTION:

- `--aligned_reads_only`: determines whether to calculate the base distribution over aligned reads only.
- `--pf_reads_only`: determines whether to calculate the base distribution over PF reads only.

14.2.14 QualityYield

The QualityYield algorithm collects metrics related to reads that pass quality thresholds and Illumina-specific filters.

The input to the QualityYield algorithm is a BAM file; its output is the metrics data.

The QualityYield algorithm accepts the following optional ALGO_OPTION:

- `--include_secondary`: determines whether to include bases from secondary alignments in the calculation.
- `--include_supplementary`: determines whether to include bases from supplementary alignments in the calculation.

14.2.15 WgsMetricsAlgo

The WgsMetricsAlgo algorithm collects metrics related to the coverage and performance of whole genome sequencing (WGS) experiments.

The input to the WgsMetricsAlgo algorithm is a BAM file; its output is the metrics data.

The WgsMetricsAlgo algorithm accepts the following optional ALGO_OPTION:

- `--min_map_qual QUALITY`: determines the filtering quality of the reads used in the calculation. Any read with quality less than QUALITY will be ignored. The default value is 20.
- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in the calculation. Any base with quality less than QUALITY will be ignored. The default value is 20.
- `--coverage_cap COVERAGE`: determines the maximum coverage limit for the histogram. Any position with coverage higher than COVERAGE will have its coverage set to COVERAGE.
- `--sample_size SIZE`: determines the Sample Size used for the Theoretical Het Sensitivity sampling. The default value is 10000.
- `--include_unpaired`: determines whether to count unpaired reads and paired reads with one end unmapped.

- `--base_qual_histogram`: determines whether to report the base quality histogram.

14.2.16 SequenceArtifactMetricsAlgo

The SequenceArtifactMetricsAlgo algorithm collects metrics that quantify single-base sequencing artifacts and OxoG artifacts.

The input to the SequenceArtifactMetricsAlgo algorithm is a BAM file; its output is the metrics data.

The SequenceArtifactMetricsAlgo algorithm accepts the following optional ALGO_OPTION:

- `--dbsnp FILE`: location of the Single Nucleotide Polymorphism database (dbSNP) used to exclude regions around known polymorphisms. Only one file is supported.
- `--min_map_qual QUALITY`: determines the filtering quality of the reads used in the calculation. Any read with quality less than QUALITY will be ignored. The default value is 30.
- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in the calculation. Any base with quality less than QUALITY will be ignored. The default value is 20.
- `--include_unpaired`: determines whether to count unpaired reads and paired reads with one end unmapped.
- `--include_duplicates`: determines whether to count duplicated reads.
- `--include_non_pf_reads`: determines whether to count non-PF reads.
- `--min_insert_size ISIZE`: determines the filtering insert size of the reads used in the calculation. Any read with insert size less than ISIZE will be ignored. The default value is 60.
- `--max_insert_size ISIZE`: determines the filtering insert size of the reads used in the calculation. Any read with insert size larger than ISIZE will be ignored. The default value is 600.
- `--tandem_reads`: determines whether the mate pairs are being sequenced from the same strand.
- `--context_size SIZE`: determined the number of context bases to include on each side. The default value is 1.

14.2.17 Genotyper ALGORITHM

The Genotyper algorithm performs the Unified Genotyper variant calling.

The input to the Genotyper algorithm is a BAM file; its output is a VCF file.

The Genotyper algorithm accepts the following optional ALGO_OPTION:

- `--annotation 'ANNOTATION_LIST'`: determines additional annotations that will be added to the output VCF. Use a comma separated list to enable or disable annotations. Include 'none' to remove the default annotations; prefix annotations with the exclamation point (!) to disable the specific annotation. See *Supported annotations in Haplotyper and Genotyper* for supported annotations.
- `-d dbSNP_FILE`: location of the Single Nucleotide Polymorphism database (dbSNP) used to label known variants. Only one file is supported.
- `--var_type VARIANT_TYPE`: determine which variant types will be called; possible values for VARIANT_TYPE are:
  - SNP to call only Single Nucleotide Polymorphism. This is the default behavior.
  - INDEL to call only insertion-deletions.
  - both to call both SNPs and INDELs.
- `--call_conf CONFIDENCE`: determine the threshold of variant quality to call a variant. Variants with quality less than CONFIDENCE will be removed.

- `--emit_conf CONFIDENCE`: determine the threshold of variant quality to emit a variant. Variants with quality less than CONFIDENCE will be not be added to the output VCF file.
- `--emit_mode MODE`: determines what calls will be emitted. Possible values for MODE are:
  - variant: emit calls only at confident variant sites. This is the default behavior.
  - confident: emit calls at confident variant sites or confident reference sites.
  - all: emit all calls, regardless of their confidence.
- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored. The default value is 17.
- `--ploidy PLOIDY`: determines the ploidy number of the sample being processed. The default value is 2.
- `--given GIVEN_VCF`: perform variant calling using only the variants provided in the GIVEN_VCF. The calling will only evaluate the locus and alleles provided in the file, and only if the variant has the FILTER column as *PASS* or ..
- `--genotype_model MODEL`: determines which model to use for genotyping and QUAL calculation. MODEL can be `coalescent` to use the "so called exact model" based on the coalescent theory in population genetics or `multinomial` to use the simplified model that assumes variants are independent; the default is `coalescent`.

14.2.18 Haplotyper ALGORITHM

The Haplotyper algorithm performs the Haplotype variant calling.

The input to the Haplotyper algorithm is a BAM file; its output is a VCF file.

The Haplotyper algorithm accepts the following optional ALGO_OPTION:

- `--annotation 'ANNOTATION_LIST'`: determines additional annotations that will be added to the output VCF. Use a comma separated list to enable or disable annotations. Include 'none' to remove the default annotations; prefix annotations with the exclamation point (!) to disable the specific annotation. See *Supported annotations in Haplotyper and Genotyper* for supported annotations.
- `-d dbSNP_FILE`: location of the Single Nucleotide Polymorphism database (dbSNP) used to label known variants. Only one file is supported.
- `--call_conf CONFIDENCE`: determine the threshold of variant quality to call a variant. Variants with quality less than CONFIDENCE will be removed. This option is ignored when the `--emit_mode` is gvcf.
- `--emit_conf CONFIDENCE`: determine the threshold of variant quality to emit a variant. Variants with quality less than CONFIDENCE will be not be added to the output VCF file. This option is ignored when the `--emit_mode` is gvcf.
- `--emit_mode MODE`: determines what calls will be emitted. Possible values for mode are:
  - variant: emit calls only at confident variant sites. This is the default behavior.
  - confident: emit calls at confident variant sites or confident reference sites.
  - all: emit all calls, regardless of their confidence.
  - gvcf: emits additional information required for joint calling. This option is required if you want to perform joint calling using the GVCFtyper algorithm.
- `--gq_bands LIST_OF_BANDS`: determines the bands that will be used to compress variants of similar genotype quality (GQ) that will be emitted as a single VCF record in the GVCF output file. The LIST_OF_BANDS is a comma-separated list of bands where each band is defined by START-END/STEP. The default value is 1-60,60-99/10,99.

- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored. The default value is 10.
- `--pcr_indel_model MODEL`: PCR indel model used to weed out false positive indels more or less aggressively. The possible MODELs are: NONE (used for PCR free samples), and HOSTILE, AGGRESSIVE and CONSERVATIVE, in order of decreasing aggressiveness. The default value is CONSERVATIVE.
- `--phasing [1/0]`: flag to enable or disable phasing in the output when using `emit_mode GVCF`. The default value is 1 (on) and this flag has no impact when using an emit_mode other than GVCF. Phasing is only calculated for diploid samples.
- `--ploidy PLOIDY`: determines the ploidy number of the sample being processed. The default value is 2.
- `--prune_factor FACTOR`: minimum pruning factor in local assembly; paths with fewer supporting kmers than FACTOR will be pruned from the graph. The default value is 2.
- `--trim_soft_clip`: determines whether soft clipped bases in the reads should be excluded from the variant calling. This argument is only recommended to process RNA reads.
- `--given GIVEN_VCF`: perform variant calling using only the variants provided in the GIVEN_VCF. The calling will only evaluate the locus and alleles provided in the file, and only if the variant has the FILTER column as *PASS* or .. This option cannot be used in conjunction with `--emit_mode gvcf`.
- `--bam_output OUTPUT_BAM`: output a BAM file containing modified reads after the local reassembly done by the variant calling. This option should only be used in conjunction with a small bed file for troubleshooting purposes.
- `--genotype_model MODEL`: determines which model to use for genotyping and QUAL calculation. MODEL can be `coalescent` to use the "so called exact model" based on the coalescent theory in population genetics or `multinomial` to use the simplified model that assumes variants are independent; the default is `coalescent`.

14.2.19 ReadWriter ALGORITHM

The ReadWriter algorithm outputs the result of applying the Base Quality Score Recalibration to a file.

The ReadWriter algorithm can also merge BAM files, and/or convert them into cram files.

The input to the ReadWriter algorithm is one or multiple BAM files and one or multiple recalibration tables; its output is the BAM file after recalibration. If the output file extension is CRAM, a CRAM file will be created. If multiple input files were used, the output file will be the result of merging all the files.

The ReadWriter algorithm accepts the following optional ALGO_OPTION:

- `--bam_compression COMPRESSION_LEVEL[0-9]`: gzip compression level for the output BAM file. The default value is 6.
- `--cram_write_options compressor=[gzip|bzip2|lzma|rans|tok|fqz|arith]`: CRAM output compression options. compressor=gzip+rans is default if not defined.
- `--cram_write_options version=[3.0|3.1]`: CRAM output version options. version=3.0 is default if not defined.
- `--output_mapq_filter min_map_qual=MIN_QUAL,max_map_qual=MAX_QUAL`: filter reads based on their mapping quality; ReadWriter will only output reads that have a mapping quality equal or larger than MIN_QUAL and equal or smaller than MAX_QUAL.
- `--output_flag_filter MASK:VALUE[,VALUE,VALUE...]`: filter reads based on the flags. MASK is the set of bits that should be considered, and VALUE is a comma separated list of FLAGS that need to be set for a read to be kept, or 0 if all bits in the MASK need to be unset. You can include multiple instances

of this option and they will be applied sequentially in the same order they are in the command line. Some examples:

- To only output reads that are PROPER_PAIR you would use `--output_flag_filter PROPER_PAIR:PROPER_PAIR` or `--output_flag_filter 0x2:0x2`.
- To output all reads except unmapped reads with UNMAP flag set, you would use `--output_flag_filter UNMAP:0` or `--output_flag_filter 0x4:0`
- To write a BAM file that does not contain either duplicated reads or unmapped reads you would use two options that will be applied sequentially `--output_flag_filter UNMAP:0 --output_flag_filter DUP:0` which will first only allow reads with the UNMAP flag unset, and then only allow reads that have the DUP flag unset.
- To write a BAM file that does not contain reads where both mates are unmapped you would use `--output_flag_filter UNMAP+MUNMAP:0,UNMAP,MUNMAP`, which will allow reads as long as both UNMAP and MUNMAP are not set together.
- More complex cases can be constructed with the help of a truth table, where the MASK is the sum of all bits you want to consider, and the VALUEs are the sum of those bits set for each of the different conditions; for instance, to find read pairs where the two non-secondary and non-supplementary reads have the same orientation (F1F2 or R1R2) hinting at a SV, you would create a table as follows:

Flag	#	F1F2	R1R2
PAIRED	0x001	x	x
REVERSE	0x010	0	x
MREVERSE	0x020	0	x
SECONDARY	0x100	0	0
SUPPLEMENTARY	0x800	0	0
	0x931	0x001	0x031

and the option would be `--output_flag_filter 0x931:0x001,0x031` or, in more complex terms, `--output_flag_filter PAIRED+SECONDARY+SUPPLEMENTARY+DUP+REVERSE+MREVERSE:PAIRED+REVERSE+MREVERSE,PAIRED+REVERSE+MREVERSE`

The table below is a reminder of the flag definition and their naming convention.

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

We recommend running the ReadWriter algorithm at the same command call as one of the variant calls, to reduce overhead.

14.2.20 GVCFtyper ALGORITHM

The GVCFtyper algorithm performs the joint variant calling of multiple samples, provided that each single sample has been previously processed using the Haplotyper algorithm with the option `--emit_mode gvcf`.

The GVCFtyper algorithm has no input in the driver level other than the reference file, its output is a VCF containing the joint called variants for all samples.

The GVCFtyper algorithm requires the following ALGO_OPTION:

- `-v INPUT`: location of the GVCF file from the variant calling algorithm performed on a single sample using the extra option `--emit_mode gvcf`. You can include GVCF files from multiple samples by specifying multiple files and repeating the `-v INPUT` option. You can use VCF files compressed with bgzip and indexed.
- Alternatively, you can input a list of GVCF files at the end of the command after the output file. Thus, the following 2 commands are interchangeable:

```
sentieon driver -r REFERENCE --algo GVCFtyper \
  -v s1_VARIANT_GVCF -v s2_VARIANT_GVCF -v s3_VARIANT_GVCF VARIANT_VCF
sentieon driver -r REFERENCE --algo GVCFtyper \
  VARIANT_VCF s1_VARIANT_GVCF s2_VARIANT_GVCF s3_VARIANT_GVCF
```

- You can read in the GVCF file list from a text file using the following commands:

```
gvcf_argument=""
while read -r line; do
 gvcf_argument=$gvcf_argument" -v $line"
done < "list_of_gvcfs"
sentieon driver -r REFERENCE --algo GVCFtyper $gvcf_argument output-joint.vcf
```

- You can also read in the GVCF file list from a text file using the following command leveraging the stdin pipe (-):

```
cat list_of_gvcfs | sentieon driver -r REFERENCE --algo GVCFtyper output-joint.vcf -
```

- You could input all files from a specific folder using the following command:

```
sentieon driver -r REFERENCE --algo GVCFtyper output-joint.vcf sample*.g.vcf
```

The GVCFtyper algorithm accepts the following optional ALGO_OPTION:

- `--annotation 'ANNOTATION_LIST'`: determines additional annotations that will be added to the output VCF. Use a comma separated list to enable or disable annotations. Include 'none' to remove the default annotations; prefix annotations with the exclamation point (!) to disable the specific annotation. See *Supported annotations in Haplotyper and Genotyper* for supported annotations.
- `-d dbSNP_FILE`: location of the Single Nucleotide Polymorphism database (dbSNP) used to label known variants. Only one file is supported.
- `--call_conf CONFIDENCE`: determine the threshold of variant quality to call a variant. Variants with quality less than CONFIDENCE will be removed. The default value is 30.
- `--emit_conf CONFIDENCE`: determine the threshold of variant quality to emit a variant. Variants with quality less than CONFIDENCE will be not be added to the output VCF file. The default value is 30.
- `--emit_mode MODE`: determines what calls will be emitted. Possible values for mode are:
  - variant: emit calls only at confident variant sites. This is the default behavior.
  - confident: emit calls at confident variant sites or confident reference sites.

  - all: emit all calls, regardless of their confidence.

- `--max_alt_alleles NUMBER`: Maximum number of alternate alleles. The default value is 100.
- `--genotype_model MODEL`: determines which model to use for genotyping and QUAL calculation. MODEL can be `coalescent` to use the "so called exact model" based on the coalescent theory in population genetics or `multinomial` to use the simplified model that assumes variants are independent; the default is `coalescent`.

14.2.21 VarCal ALGORITHM

The VarCal algorithm calculates the Variant Quality Score Recalibration (VQSR). VQSR assigns a well-calibrated probability score to individual variant calls, to enable more accurate control in determining the most likely variants. For that, VQSR uses highly confident known sites to build a recalibration model and determine the probability that called sites are true. For more information about the algorithm, you can check http://gatkforums.broadinstitute.org/discussion/39/variant-quality-score-recalibration-vqsr. For information on the recommended resources to use in VQSR, you can check https://www.broadinstitute.org/gatk/guide/article?id=1259.

The VarCal algorithm has no input in the driver level other than the reference file, its output is a recalibration file containing additional annotations related to the VQSR.

The VarCal algorithm requires the following ALGO_OPTION:

- `-v INPUT`: location of the VCF file from the variant calling algorithm; you can use a VCF file compressed with bgzip and indexed.
- `--tranches_file TRANCHES_FILE`: location and filename of the file containing the partition of the call sets into quality tranches.
- `--annotation ANNOTATION`: determine annotation that will be used during the recalibration. You can include multiple annotations in the optimization by repeating the `--annotation ANNOTATION` option. You can use all annotations present in the original variant call file.
- `--resource RESOURCE_FILE --resource_param PARAM`: location of the VCF file used as a training/truth resource in VQSR, followed by parameters determining how the file will be used. You can include multiple collections by specifying multiple files and repeating the `--resource RESOURCE_FILE --resource_param PARAM` option. The PARAM argument follows the syntax:

```
LABEL,known=IS_KNOWN,training=IS_TRAIN,truth=IS_TRUTH,prior=PRIOR
```

  - **LABEL** is a descriptive name for the resource.
  - **IS_KNOWN** can be *true* or *false*, and determines whether the sites contained in the resource will be used to stratify output metrics.
  - **IS_TRAIN** can be *true* or *false*, and determines whether the sites contained in the resource will be used for training the recalibration model.
  - **IS_TRUTH** can be *true* or *false*, and determines whether the sites contained in the resource will be considered true sites.
  - **PRIOR** is a value that reflects your confidence in how reliable the resource is as a truth set.

The VarCal algorithm accepts the following optional ALGO_OPTION:

- `--srand RANDOM_SEED`: determines the seed to use in the random number generation. You can set RANDOM_SEED to 0 and the software will use the random seed from your computer. In order to generate a deterministic result, you should use a non-zero RANDOM_SEED and set the NUMBER_THREADS to 1.

- `--var_type VARIANT_TYPE`: determine which variant types will be recalibrated; possible values for VARIANT_TYPE are:
  - SNP to recalibrate only Single Nucleotide Polymorphism. This is the default behavior.
  - INDEL to recalibrate only insertion-deletions.
  - (do not use) BOTH to recalibrate both SNPs and INDELs. This setting SHOULD NOT be used, as VQSR should be performed independently for SNPs and INDELs.
- `--tranche TRANCH_THRESHOLD`: normalized quality threshold for each tranche; the TRANCH_THRESHOLD number is a number between 0 and 100. Multiple instances of the option are allowed that will create as many tranches as there are thresholds. The default values are 90, 99, 99.9 and 100.
- `--max_gaussians MAX_GAUSS`: determines the maximum number of Gaussians that will be used for the positive recalibration model. The default value is 8 for SNP and 4 for INDEL.
- `--max_neg_gaussians MAX_GAUSS`: determines the maximum number of Gaussians that will be used for the negative recalibration model. The default value is 2.
- `--max_iter MAX_ITERATIONS`: determines the maximum number of iterations for the Expectation Maximization (EM) optimization. The default value is 150.
- `--max_mq MAPQ`: indicates the maximum MQ in your data, which will be used to perform a logit jitter transform of the MQ to make the distribution closer to a Gaussian.
- `--aggregate_data AGREGATE_VCF`: location of an additional VCF file containing variants called from other similar samples; these additional data will increase the effective sample size for the statistical model calibration. Multiple instances of the option are allowed.
- `--plot_file PLOT_FILE`: location of the temporary file containing the necessary data to generate the reports from the VarCal algorithm.

14.2.22 ApplyVarCal ALGORITHM

The ApplyVarCal algorithm combines the output information from the VQSR with the original variant information.

The ApplyVarCal algorithm has no input in the driver level other than the reference file; its output is a copy of the original VCF containing additional annotations from the VQSR.

The ApplyVarCal algorithm requires the following ALGO_OPTION:

- `-v INPUT`: location of the VCF file from the variant calling algorithm. It should be the same as the one used in the VarCal algorithm; you can use a VCF file compressed with bgzip and indexed.
- `--recal VARIANT_RECAL_DATA`: location of the VCF file output from the VarCal algorithm.
- `--tranches_file TRANCHES_FILE`: location of the tranches file output from the VarCal algorithm.
- `--var_type VARIANT_TYPE`: determine which variant types will be recalibrated. This option should be consistent with the one used in the VarCal algorithm.

Alternatively, you can use the option `--vqsr_model` to input a comma-separated list of the required information for multiple VQSR models; this option allows you to apply both a SNP and INDEL mode in a single command line. The syntax of the option is:

```
--vqsr_model var_type=VARIANT_TYPE,\
             recal=VARIANT_RECAL_DATA,\
             tranches_file=TRANCHES_FILE,\
             sensitivity=SENSITIVITY
```

The ApplyVarCal algorithm accepts the following optional ALGO_OPTION:

- `--sensitivity SENSITIVITY`: determine the sensitivity to the available truth sites; only tranches with threshold larger than the sensitivity will be included in the recalibration. We recommend you use a sensitivity number that is included in the tranche threshold list of VarCal algorithm; this will reduce rounding issues. The default value is NULL, so that no tranches filtering is applied, and only the LOW_VQSLOD filter is applied.

14.2.23 TNsnv ALGORITHM

The TNsnv algorithm performs the somatic variant calling on the tumor-normal matched pair or the tumor and panel of normal data, using a Genotyper algorithm.

The input to the TNsnv algorithm is a BAM file; its output is a VCF file.

The TNsnv algorithm requires the following ALGO_OPTION:

- `--tumor_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the tumor sample.

Depending on the mode it is run, the TNsnv algorithm may require the following ALGO_OPTION:

- `--normal_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the normal sample.
- `--detect_pon`: indicates that you are using the TNsnv algorithm to create a VCF file that will be part of a panel of normal.
- `--cosmic COSMIC_VCF`: location of the Catalogue of Somatic Mutations in Cancer (COSMIC) VCF file used to create the panel of normal file. Only one file is supported.
- `--pon PANEL_OF_NORMAL_VCF`: location of the file containing the variants detected in the Panel of Normal analysis that will be used to remove false positives. Only one file is supported.

The TNsnv algorithm accepts the following optional ALGO_OPTION:

- `--dbsnp dbSNP_FILE`: location of the Single Nucleotide Polymorphism database (dbSNP). The variants in the dbSNP will be more likely to be marked as germline as they require more evidence of absence in the normal. Only one file is supported.
- `--call_stats_out CALL_STATS_FILE`: location and filename of the file containing the call stats information from the somatic variant calling.
- `--stdcov_out COVERAGE_FILE`: location and filename of the wiggle file containing the standard coverage.
- `--tumor_depth_out TUMOR_DEPTH_FILE`: location and filename of the wiggle file containing the depth of the tumor sample reads.
- `--normal_depth_out NORMAL_DEPTH_FILE`: location and filename of the wiggle file containing the depth of the normal sample reads.
- `--power_out POWER_FILE`: location and filename of the power file.
- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored. The default value is 5.
- `--min_init_tumor_lod NUMBER`: minimum tumor log odds in the initial pass calling variants. The default value is 4.
- `--min_tumor_lod NUMBER`: minimum tumor log odds in the final call of variants. The default value is 6.3.
- `--min_normal_lod NUMBER`: minimum normal log odds used to check that the tumor variant is not a normal variant. The default value is 2.2.

- `--contamination_frac NUMBER`: estimation of the contamination fraction from other samples. The default value is 0.02.
- `--min_cell_mutation_frac NUMBER`: minimum fraction of cells which have mutation. The default value is 0.
- `--min_strand_bias_lod NUMBER`: minimum log odds for calling strand bias. The default value is 2.
- `--min_strand_bias_power NUMBER`: minimum power for calling strand bias. The default value is 0.9.
- `--min_dbsnp_normal_lod NUMBER`: minimum log odds for calling normal non-variant at dbsnp sites. The default value is 5.5.
- `--min_normal_allele_frac NUMBER`: minimum allele fraction to be considered in normal; this parameter is useful when the normal sample is contaminated with the tumor sample. The default value is 0.
- `--min_tumor_allele_frac NUMBER`: minimum allelic fraction in tumor sample. The default value is 0.005.
- `--max_indel NUMBER`: maximum nearby indel events that are allowed. The default value is 3.
- `--max_read_clip_frac NUMBER`: maximum fraction of soft/hard clipped bases in a read. The default value is 0.3.
- `--max_mapq0_frac NUMBER`: maximum ratio of reads whose mapq are 0 used to determine poor mapped area. The default value is 0.5.
- `--min_pir_median NUMBER`: minimum read position median. The default value is 10.
- `--min_pir_mad NUMBER`: minimum read position median absolute deviation. The default value is 3.
- `--max_alt_mapq NUMBER`: maximum value of alt allele mapping quality score. The default value is 20.
- `--max_normal_alt_cnt NUMBER`: maximum alt alleles count in normal pileup. The default value is 2.
- `--max_normal_alt_qsum NUMBER`: maximum quality score sum of alt allele in normal pileup. The default value is 20.
- `--max_normal_alt_frac NUMBER`: maximum fraction of alt allele in normal pileup. The default value is 0.03.
- `--power_allele_frac NUMBER`: allele fraction used in power calculations. The default value is 0.3.

14.2.24 TNhaplotyper ALGORITHM

The TNhaplotyper algorithm performs the somatic variant calling on the tumor-normal matched pair or the tumor and panel of normal data, using a Haplotyper algorithm.

The input to the TNhaplotyper algorithm is a BAM file; its output is a VCF file.

The TNhaplotyper algorithm requires the following ALGO_OPTION:

- `--tumor_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the tumor sample.

Depending on the mode it is run, the TNhaplotyper algorithm may require the following ALGO_OPTION:

- `--normal_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the normal sample.
- `--detect_pon`: indicates that you are using the TNhaplotyper algorithm to create a VCF file that will be part of a panel of normal.
- `--cosmic COSMIC_VCF`: location of the Catalogue of Somatic Mutations in Cancer (COSMIC) VCF file used to create the panel of normal file. Only one file is supported.

- `--pon PANEL_OF_NORMAL_VCF`: location of the file containing the variants detected in the Panel of Normal analysis that will be used to remove false positives. Only one file is supported.

The TNhaplotyper algorithm accepts the following optional ALGO_OPTION:

- `--dbsnp dbSNP_FILE`: location of the Single Nucleotide Polymorphism database (dbSNP). The variants in the dbSNP will be more likely to be marked as germline as they require more evidence of absence in the normal. Only one file is supported.
- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored. The default value is 10.
- `--prune_factor FACTOR`: minimum pruning factor in local assembly; paths with fewer supporting kmers than FACTOR will be pruned from the graph. The default value is 2.
- `--pcr_indel_model MODEL`: PCR indel model used to weed out false positive indels more or less aggressively. The possible modes are`: NONE (used for PCR free samples), and HOSTILE, AGGRESSIVE and CONSERVATIVE, in order of decreasing aggressiveness. The default value is HOSTILE.
- `--phasing [1/0]`: flag to enable or disable phasing in the output.
- `--min_init_tumor_lod NUMBER`: minimum tumor log odds in the initial pass calling variants. The default value is 4.
- `--min_init_normal_lod NUMBER`: minimum tumor log odds in the initial pass calling variants. The default value is 0.5.
- `--min_tumor_lod NUMBER`: minimum tumor log odds in the final call of variants. The default value is 6.3.
- `--min_normal_lod NUMBER`: minimum normal log odds used to check that the tumor variant is not a normal variant. The default value is 2.2.
- `--min_strand_bias_lod NUMBER`: minimum log odds for calling strand bias. The default value is 2.
- `--min_strand_bias_power NUMBER`: minimum power for calling strand bias. The default value is 0.9.
- `--min_pir_median NUMBER`: minimum read position median. The default value is 10.
- `--min_pir_mad NUMBER`: minimum read position median absolute deviation. The default value is 3.
- `--max_normal_alt_cnt NUMBER`: maximum alt alleles count in normal pileup. The default value is 2.
- `--max_normal_alt_qsum NUMBER`: maximum quality score sum of alt allele in normal pileup. The default value is 20.
- `--max_normal_alt_frac NUMBER`: maximum fraction of alt allele in normal pileup. The default value is 0.03.
- `--tumor_contamination_frac NUMBER`: estimation of the contamination fraction on the tumor sample from other samples. The default value is 0.
- `--normal_contamination_frac NUMBER`: estimation of the contamination fraction on the normal sample from other samples. The default value is 0.
- `--filter_clustered_read_position`: filters variants that are clustered at the start or end of sequencing reads
- `--filter_strand_bias`: filters variants that show evidence of strand bias
- `--bam_output OUTPUT_BAM`: output a BAM file containing modified reads after the local reassembly done by the variant calling. This option should only be used in conjunction with a small bed file for troubleshooting purposes.

- `--trim_soft_clip`: determines whether soft clipped bases in the reads should be excluded from the variant calling.

14.2.25 TNhaplotyper2 ALGORITHM

The TNhaplotyper2 algorithm performs the somatic variant calling on the tumor-normal matched pair or the tumor and panel of normal data, using a Haplotype based algorithm.

The input to the TNhaplotyper2 algorithm is one of multiple BAM files; its output is a VCF file that will be used when filtering the results in TNfilter; TNhaplotyper2 will output an additional file with the same output file name and *.stats* extension that contains statistics that can help the filtering.

The TNhaplotyper2 algorithm requires the following ALGO_OPTION:

- `--tumor_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the tumor sample.

Depending on the mode it is run, the TNhaplotyper2 algorithm may require the following ALGO_OPTION:

- `--normal_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the normal sample.
- `--pon PANEL_OF_NORMAL_VCF`: location of the file containing the variants detected in the Panel of Normal analysis that will be used to remove false positives. Only one file is supported.

The TNhaplotyper2 algorithm accepts the following optional ALGO_OPTION:

- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored. The default value is 10.
- `--prune_factor FACTOR`: minimum pruning factor in local assembly; paths with fewer supporting kmers than FACTOR will be pruned from the graph. Setting the prune factor to 0 will turn on adaptive pruning. The default value is 0.
- `--pcr_indel_model MODEL`: PCR indel model used to weed out false positive indels more or less aggressively. The possible modes are`: NONE (used for PCR free samples), and HOSTILE, AGGRESSIVE and CONSERVATIVE, in order of decreasing aggressiveness. The default value is CONSERVATIVE.
- `--min_init_tumor_lod NUMBER`: minimum tumor log odds in the initial pass calling variants. The default value is 2.0.
- `--min_tumor_lod NUMBER`: minimum tumor log odds in the final call of variants. The default value is 3.0.
- `--min_normal_lod NUMBER`: minimum normal log odds used to check that the tumor variant is not a normal variant. The default value is 2.2.
- `--germline_vcf VCF`: location of the VCF containing the population allele frequency.
- `--default_af AF`: determines the allele frequency value for alleles not found in the germline vcf. The default value is 1E-6 when running tumor-normal mode, and 5E-8 when running without a matched normal in tumor-only mode.
- `--max_germline_af AF`: determines the maximum germline allele frequency in tumor-only mode. The default value is 0.01.
- `--call_pon_sites`: determines whether to call candidate variants even if they are present in the Panel of Normal input.
- `--callable_depth DEPTH`: determines the minimum depth for a site to be considered callable for the additional *.stats* file containing statistics that can help the filtering.
- `--given GIVEN_VCF`: perform variant calling using the variants provided in the GIVEN_VCF. In addition to looking for variants in discovery mode, the calling will evaluate the locus and alleles provided in the file, but only if the variant has the FILTER column as *PASS* or ..

- `--bam_output OUTPUT_BAM`: output a BAM file containing modified reads after the local reassembly done by the variant calling. This option should only be used in conjunction with a small bed file for troubleshooting purposes.
- `--trim_soft_clip`: determines whether soft clipped bases in the reads should be excluded from the variant calling.
- `--call_germline_sites`: determines whether to call candidate variants if they are present in the germline sites.

14.2.26 TNfilter ALGORITHM

The TNfilter algorithm performs filtering on the output of TNhaplotyper2.

The input to the TNfilter algorithm is a VCF containing candidate variants to be filtered; its output is the VCF containing the filtered variants; TNfilter will output an additional file with the same output file name and *.stats* extension that contains statistics about the filtering.

The TNfilter algorithm requires the following ALGO_OPTION:

- `-v CANDIDATE_VCF`: the location and file name of the file containing the unfiltered variants produced by TNhaplotyper2. You can use VCF files compressed with bgzip and indexed. In addition to this file, the binary requires the file containing statictics named CANDIDATE_VCF.stats created by TNhaplotyper2.
- `--tumor_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the tumor sample.

Depending on the mode it is run, the TNfilter algorithm may require the following ALGO_OPTION:

- `--normal_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the normal sample.

The TNfilter algorithm accepts the following optional ALGO_OPTION:

- `--orientation_priors PRIORS`: the location and file name of the file containing the orientation bias information produced by OrientationBias. This option affects the *Orientation Bias* Filter.
- `--contamination FILE`: the location and file name of the file containing the contamination information produced by ContaminationModel. This option affects the *Contamination Filter*.
- `--tumor_segments SEGMENTS`: the location and file name of the file containing the tumor segments information produced by ContaminationModel. This option affects the *Germline Risk Artifact* Filter.
- `--threshold_strategy STRATEGY`: determines the strategy that should be applied to optimize the posterior probability threshold. The possible values are: *f_score* to prioritize F-Score, *precision* to prioritize reducing false positives, and *constant*. The default value is *f_score*.
- `--f_score_beta SCORE`: when using *–threshold_strategy f_score*, determines the relative weigth of recall to precision that will be the goal of the filtering. The default value is 1 for equal weight of precision and recall.
- `--max_fp_rate FP_RATE`: when using *–threshold_strategy precision*, determines the maximum expected rate of false positive calls. The default value is 0.05.
- `--threshold THRESHOLD`: when using *–threshold_strategy constant*, determines the posterior probability threshold.
- `--min_median_base_qual QUALITY`: determines the minimum median base quality of alt reads. This option affects the *Median Base Quality Filter*. The default value is 20.
- `--max_event_count COUNT`: determines the maximum number of events allowed in the active region. If there are more than COUNT events in the active region, the variants will be filtered with the *Clustered Events Filter*. The default value is 2.
- `--unique_alt_reads COUNT`: determines the minimum number of unique ALT reads that need to support the variant. This option affects the *Duplicate Read Filter*. The default value is 0.

- `--max_mfrl_diff VALUE`: determines the maximum difference between the median ALT and REF fragments. This option affects the *Median Fragment Length Difference Filter*. The default value is 10000.
- `--max_haplotype_distance DISTANCE`: determines the maximum distance of two phased variants within the same haplotype. If a variant is filtered, all phased variants within the DISTANCE will also be filtered out, but variants beyond the DISTANCE will not be affected. This option affects the *Haplotype Filter*. The default value is 100.
- `--min_tumor_af AF`: determines the minimum tumor allele fraction. This option affects the *Low Allele Fection Filter*. The default value is 0.
- `--min_median_map_qual QUALITY`: determines the minimum median mapping quality of the reads supporting the variant. This argument affects the *Median Mapping Quality Filter*. The default value is 30.
- `--long_indel_length LENGTH`: determines whether INDELs will use the reference mapping quality; INDELs longer than LENGTH will use reference mapping quality. This argument affects the *Median Mapping Quality Filter*. The default value is 5.
- `--max_alt_count COUNT`: determines the maximum number of ALT alleles at a site. This option affects the *Multi-Allelic Filter*. The default value is 1.
- `--max_n_ratio RATIO`: determines the maximum ratio of N-bases to ALT bases. This option affects the *N-base Ratio Filter*. The default value is 1.
- `--normal_p_value VALUE`: determines the P-value threshold for the detection of normal artifacs. This option affects the *Normal Artifact Filter*. The default is 0.001.
- `--min_median_pos DISTANCE`: determines the minimum median distance from the variant to the end of the reads. This option affects the *Median Read Position Filter*. The default value is 1.
- `--min_slippage_length LENGTH`: determines the minimum length of REF bases in an STR for it to be likely to have polymerase slippage. This option affects the *Polymerase Slippage Filter*. The default value is 8.
- `--slippage_rate RATE`: determines the frequency of polymerase slippage in areas likely to have polymerase slippage. This option affects the *Polymerase Slippage Filter*. The default value is 0.1.
- `--min_alt_reads_per_strand COUNT`: determines the minimum number of ALT reads required on each strand. This option affects the *Hard Strand Bias Filter*. The default value is 0.

In addition to the filters shown above, TNfilter may apply the following filters:

- *Model-based strad bias artifact Filter*, for sites likely to be an artifact of strand bias.
- *Panel of Normal Filter*, for sites present in the Panel of Normals.
- *Tumor Evidence Filter*, for sites with weak evidence of a variant.

14.2.27 OrientationBias ALGORITHM

The OrientationBias algorithm estimates any possible orientation bias present in the sequencing data.

The input to the OrientationBias algorithm is one or multiple BAM files; its output is a file containing the orientation bias information that will be used when filtering the results of TNhaplotyper2 in TNfilter.

The OrientationBias algorithm requires the following ALGO_OPTION:

- `--tumor_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the tumor sample.

The OrientationBias algorithm accepts the following optional ALGO_OPTION:

- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored. The default value is 20.

- `--min_median_map_qual QUALITY`: determines the minimum median mapping quality. Sites where the median mapping quality of the reads is below QUALITY will be ignored. The default value is 50.
- `--max_depth DEPTH`: determines the depth threshold for grouping. Sites with depth higher than DEPTH will be grouped together. The default value is 200.

14.2.28 ContaminationModel ALGORITHM

The ContaminationModel algorithm estimates the cross-sample contamination and tumor segmentation to be used in the TNseq pipeline.

The input to the ContaminationModel algorithm is one or multiple BAM files; its output is a file containing the contamination information that can be used when filtering the results of TNhaplotyper2 in TNfilter. In addition the tool may output a *.segments* file containing the tumor segments information that can help the filtering.

The ContaminationModel algorithm requires the following ALGO_OPTION:

- `--tumor_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the tumor sample.
- `--vcf VCF`: location of the VCF containing the population allele frequency.

Depending on the mode it is run, the ContaminationModel algorithm may require the following ALGO_OPTION:

- `--normal_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the normal sample.

The ContaminationModel algorithm accepts the following optional ALGO_OPTION:

- `--tumor_segments CONTAMINATION.segments`: the location and file name of the optional output file containing the tumor segments information; this information will be used in TNfilter to filter calls that are likely Germline Risk Artifacts.
- `--min_map_qual QUALITY`: determines the filtering quality of the reads used. Any reads with mapping quality less than QUALITY will be filtered out. The default value is 50.
- `--min_af AF`: determines the minimum value of the population allele frequency. The default value is 0.01.
- `--max_af AF`: determines the maximum value of the population allele frequency. The default value is 0.2.

14.2.29 TNscope ALGORITHM

The TNscope algorithm performs the somatic variant calling on the tumor-normal matched pair or the tumor only data, using a Haplotyper algorithm.

The input to the TNscope algorithm is a BAM file; its output is a VCF file.

The TNscope algorithm requires the following ALGO_OPTION:

- `--tumor_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the tumor sample.

The TNscope algorithm accepts the following optional ALGO_OPTION:

- `--annotation 'ANNOTATION'`: determines additional annotations that will be added to the output VCF. Currently `FAD` is supported. Three additional annotations will be added under `--annotation FAD`: `FAD`, `F1R2`, and `F2R1`.
- `--normal_sample SAMPLE_NAME`: name of the SM tag in the BAM file for the normal sample. When doing tumor-only somatic calling, this argument is not required.
- `--cosmic COSMIC_VCF`: location of the Catalogue of Somatic Mutations in Cancer (COSMIC) VCF file used to create the panel of normal file. Only one file is supported.

- `--pon PANEL_OF_NORMAL_VCF`: location of the file containing the variants detected in the Panel of Normal analysis that will be used to remove false positives. Only one file is supported. This file is the same as the one used with TNhaplotyper.
- `--dbsnp dbSNP_FILE`: location of the Single Nucleotide Polymorphism database (dbSNP). The variants in the dbSNP will be more likely to be marked as germline as they require more evidence of absence in the normal. Only one file is supported.
- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored. The default value is 15.
- `--prune_factor FACTOR`: minimum pruning factor in local assembly; paths with fewer supporting kmers than FACTOR will be pruned from the graph. Setting the prune factor to 0 will turn on adaptive pruning. The default value is 2.
- `--pcr_indel_model MODEL`: PCR indel model used to weed out false positive indels more or less aggressively. The possible modes are`: NONE (used for PCR free samples), and HOSTILE, AGGRESSIVE and CONSERVATIVE, in order of decreasing aggressiveness. The default value is CONSERVATIVE.
- `--phasing [1/0]`: flag to enable or disable phasing in the output. The default value is 1 (on).
- `--min_init_tumor_lod NUMBER`: minimum tumor log odds in the initial pass calling variants. The default value is 4.
- `--min_init_normal_lod NUMBER`: minimum tumor log odds in the initial pass calling variants. The default value is 0.5.
- `--min_tumor_lod NUMBER`: minimum tumor log odds in the final call of variants. The default value is 6.3.
- `--min_normal_lod NUMBER`: minimum normal log odds used to check that the tumor variant is not a normal variant. The default value is 2.2.
- `--min_dbsnp_normal_lod NUMBER`: minimum log odds for calling normal non-variant at dbsnp sites. The default value is 5.5.
- `--tumor_contamination_frac NUMBER`: estimation of the contamination fraction on the tumor sample from other samples. The default value is 0.
- `--normal_contamination_frac NUMBER`: estimation of the contamination fraction on the normal sample from other samples. The default value is 0.
- `--given GIVEN_VCF`: perform variant calling using only the variants provided in the GIVEN_VCF. The calling will only evaluate the locus and alleles provided in the file, and only if the variant has the FILTER column as *PASS* or ..
- `--bam_output OUTPUT_BAM`: output a BAM file containing modified reads after the local reassembly done by the variant calling. This option should only be used in conjunction with a small bed file for troubleshooting purposes.
- `--disable_detector DETECTOR`: disable the variant calling for specific detectors: use 'sv' as DETECTOR to prevent calling of structural variants, and use 'snv_indel' as DETECTOR to prevent calling of small variants.
- `--trim_soft_clip`: determines whether soft clipped bases in the reads should be excluded from the variant calling.
- `--trim_primer AMPLICON_TABLE`: determines whether to trim primers based on the information provided in the AMPLICON_TABLE. The AMPLICON_TABLE is a BED file containins the location of the primers in the amplicon sequencing as a tab-delimited file with eigth columns: contig, amplicon_start, amplicon_end, name (ignored), score (ignored), strand (ignored), insert_start, insert_end; the tool will

trim bases from amplicon reads between amplicon_start and insert_start, and between amplicon_end and insert_end.

- `--trim_adaptor=[0|1]`: determines whether to trim off adaptor bases from the input reads. The default is to trim the adapter bases. Set `--trim_adaptor=0` to turn it off.

14.2.30 RNASplitReadsAtJunction ALGORITHM

The RNASplitReadsAtJunction algorithm performs the splitting of reads into exon segments by getting rid of Ns but maintaining grouping information, and hard-clipping any sequences overhanging into the intron regions.

The input to the RNASplitReadsAtJunction algorithm is a BAM file; its output is a BAM file.

The RNASplitReadsAtJunction algorithm requires the following ALGO_OPTION:

- `--reassign_mapq IN_QUAL:OUT_QUAL`: the algorithm will reassign mapping qualities from IN_QUAL to OUT_QUAL. This argument is required because STAR assigns a quality of 255 to good alignments instead of the expected default score of 60.

The RNASplitReadsAtJunction algorithm accepts the following optional ALGO_OPTION:

- `--ignore_overhang`: determines whether to ignore and not fix the overhanging sections of the reads.
- `--overhang_max_bases NUMBER`: determines the maximum number of bases allowed in a hard-clipped overhang, so that if there are more bases in the overhang, the overhang will not be hard-clipped. The default value is 40.
- `--overhang_max_mismatches NUMBER`: determines the maximum number of mismatches allowed in a non-hard-clipped overhang, so that the complete overhang will be hard-clipped if the number of mismatches is too high. The default value is 1.
- `--cram_write_options compressor=[gzip|bzip2|lzma|rans|tok|fqz|arith]`: CRAM output compression options. compressor=gzip+rans is default if not defined.
- `--cram_write_options version=[3.0|3.1]`: CRAM output version options. version=3.0 is default if not defined.

14.2.31 ContaminationAssessment ALGORITHM

The ContaminationAssessment algorithm assesses the contamination present in a sample BAM file; the output of this algorithm can be used as the value of argument contamination_frac, normal_contamination_frac and tumor_contamination_frac in the TNseq and TNscope tools.

The input to the ContaminationAssessment algorithm is a BAM file; its output is a text file.

The ContaminationAssessment algorithm requires the following ALGO_OPTION:

- `--pop_vcf VCF_FILE`: the location of the VCF file containing the allele frequency information for the specific population of the sample.
- `--genotype_vcf VCF_FILE`: the location of the VCF file containing the DNAseq variants reported for the individual; to calculate the contamination in the tumor sample, you should use the DNAseq variants reported for the normal sample. You can create this file by using Haplotyper or Genotyper on the sample bam.

The ContaminationAssessment algorithm accepts the following optional ALGO_OPTION:

- `--type ASSESS_TYPE`: determines the type for the estimate. The possible values are SAMPLE, READGROUP and META to assess the contamination by sample, by lane, or in aggregate across all the reads. Multiple instances of the option are allowed that will assess the contamination at multiple levels. The default value is META.

- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in the contamination assessment. Any base with quality less than QUALITY will be ignored. The default value is 20.
- `--min_map_qual QUALITY`: determines the filtering quality of the reads used in the contamination assessment. Any read with quality less than QUALITY will be ignored. The default value is 20.
- `--min_basecount NUMBER`: determines the minimum number of bases that need to be present at a locus before the contamination is assessed. The default value is 500.
- `--trim_thresh NUMBER`: threshold that will be used to trim sites; if the probability of the contamination ratio being larger than 0.5 is larger than the threshold, the site will not be included in the contamination assessment. The default value is 0.95.
- `--trim_frac NUMBER`: determines the maximum fraction of sites that may be trimmed based on the trim threshold. The default value is 0.01.
- `--precision NUMBER`: determines the PRECISION on the output percent number. The default value is 0.1.
- `--base_report FILE`: location and filename of the output file that will contain an extended report about the processed data.
- `--population POPULATION_NAME`: a population to use to determine the baseline allele frequency of the sample. The default value is CEU.

14.2.32 TNModelApply ALGORITHM

The TNModelApply algorithm applies a Machine Learning model on the results of TNscope to help with variant filtration. This algorithm is only supported in the Linux version of the Sentieon Genomics software.

The TNModelApply algorithm has no input in the driver level, its output is a VCF file.

The TNModelApply algorithm requires the following ALGO_OPTION:

- `-v INPUT`: location of the VCF file from the TNscope variant calling; you can use a VCF file compressed with bgzip and indexed.
- `-m MODEL_FILE`: location of the file containing the Machine Learning model.

The TNModelApply algorithm modifies the input VCF file by adding the MLrejected FILTER to the variants; since the FILTER is added, you may want to remove any FILTERs already present in the input VCF, as they may no longer be relevant. You can use bcftools(https://samtools.github.io/bcftools/bcftools.html) for that purpose:

```
$BCF/bcftools annotate -x "^FILTER/MLrejected,FILTER/PASS" -O z \
  -o $OUTPUT.vcf.gz $INPUT.vcf.gz
```

14.2.33 DNAscope ALGORITHM

The DNAscope algorithm performs an improved version of Haplotype variant calling.

The input to the DNAscope algorithm is a BAM file; its output is a VCF file.

The DNAscope algorithm accepts the following optional ALGO_OPTION:

- `--annotation 'ANNOTATION_LIST'`: determines additional annotations that will be added to the output VCF. Use a comma separated list to enable or disable annotations. Include 'none' to remove the default annotations; prefix annotations with the exclamation point (!) to disable the specific annotation. See *Supported annotations in Haplotyper and Genotyper* for supported annotations.
- `-d dbSNP_FILE`: location of the Single Nucleotide Polymorphism database (dbSNP) used to label known variants. Only one file is supported.

- `--var_type VARIANT_TYPE`: determine which variant types will be called; VARIANT_TYPE is a comma separated list of the following possible values:
  - SNP to call Single Nucleotide Polymorphism. This is included in the default behavior.
  - INDEL to call insertion-deletions. This is included in the default behavior.
  - BND to call break-end information required for the structural variant caller.
- `--call_conf CONFIDENCE`: determine the threshold of variant quality to call a variant. Variants with quality less than CONFIDENCE will be removed.
- `--emit_conf CONFIDENCE`: determine the threshold of variant quality to emit a variant. Variants with quality less than CONFIDENCE will be not be added to the output VCF file.
- `--emit_mode MODE`: determines what calls will be emitted. Possible values for mode are:
  - variant: emit calls only at confident variant sites. This is the default behavior.
  - confident: emit calls at confident variant sites or confident reference sites.
  - all: emit all calls, regardless of their confidence.
  - gvcf: emits additional information required for joint calling. This option is required if you want to perform joint calling using the GVCFtyper algorithm.
- `--gq_bands LIST_OF_BANDS`: determines the bands that will be used to compress variants of similar genotype quality (GQ) that will be emitted as a single VCF record in the GVCF output file. The LIST_OF_BANDS is a comma-separated list of bands where each band is defined by START-END/STEP. The default value is 1-60,60-99/10,99.
- `--min_base_qual QUALITY`: determines the filtering quality of the bases used in variant calling. Any base with quality less than QUALITY will be ignored. The default value is 10.
- `--pcr_indel_model MODEL`: PCR indel model used to weed out false positive indels more or less aggressively. The possible MODELs are: NONE (used for PCR free samples), and HOSTILE, AGGRESSIVE and CONSERVATIVE, in order of decreasing aggressiveness. The default value is CONSERVATIVE.
- `--phasing [1/0]`: flag to enable or disable phasing in the output. The default value is 1 (on). Phasing is only calculated for diploid samples.
- `--ploidy PLOIDY`: determines the ploidy number of the sample being processed. The default value is 2.
- `--prune_factor FACTOR`: minimum pruning factor in local assembly; paths with fewer supporting kmers than FACTOR will be pruned from the graph. The default value is 2.
- `--trim_soft_clip`: determines whether soft clipped bases in the reads should be excluded from the variant calling. This argument is only recommended to process RNA reads.
- `--given GIVEN_VCF`: perform variant calling using only the variants provided in the GIVEN_VCF. The calling will only evaluate the locus and alleles provided in the file, and only if the variant has the FILTER column as *PASS* or .. This option cannot be used in conjunction with `--emit_mode gvcf`.
- `--bam_output OUTPUT_BAM`: output a BAM file containing modified reads after the local reassembly done by the variant calling. This option should only be used in conjunction with a small bed file for troubleshooting purposes.
- `--filter_chimeric_reads`: determines whether chimeric reads will be used when calling variants. The default is to include chimeric reads only if the `var_type BND` is set.
- `--trim_adaptor`: determines whether to trim off adaptor bases from the input reads. The default is to NOT trim the adapter bases as that could affect the region around SVs.

- `--model MODEL_FILE`: the location of the machine learning model file that will be used with the DNAModelApply tool; the model will be used to determine the settings used in variant calling.

14.2.34 DNAModelApply ALGORITHM

The DNAModelApply algorithm performs the second step of variant calling using DNAscope. This algorithm is only supported in the Linux version of the Sentieon Genomics software.

The DNAModelApply algorithm has no input in the driver level, its output is a VCF containing the variants.

The DNAModelApply algorithm requires the following ALGO_OPTION:

- `-v INPUT`: location of the VCF file from the DNAscope variant calling algorithm performed with an input model file determining the correct settings. You can use VCF files compressed with bgzip and indexed.
- `-m MODEL_FILE`: the location of the machine learning model file; this file should be the same as the one used in the DNAscope command to generate the input VCF.

The DNAModelApply algorithm does not support any optional ALGO_OPTION.

The DNAModelApply algorithm modifies the input VCF file by adding the MLrejected FILTER to the variants; since the FILTER is added, you may want to remove any FILTERs already present in the input VCF, as they may no longer be relevant. You can use bcftools(https://samtools.github.io/bcftools/bcftools.html) for that purpose:

```
$BCF/bcftools annotate -x "^FILTER/MLrejected,FILTER/PASS" -O z \
  -o $OUTPUT.vcf.gz $INPUT.vcf.gz
```

14.2.35 SVSolver ALGORITHM

The SVSolver algorithm performs the structural variant calling of a sample, provided that the sample has been previously processed using the DNAscope algorithm with the option `--var_type bnd`.

The SVSolver algorithm has no input in the driver level, its output is a VCF containing the called structural variants.

The SVSolver algorithm requires the following ALGO_OPTION:

- `-v INPUT`: location of the VCF file from the DNAscope variant calling algorithm performed on a sample using the option `--var_type bnd`. You can use VCF files compressed with bgzip and indexed.

The SVSolver algorithm does not support any optional ALGO_OPTION.

14.2.36 VariantPhaser ALGORITHM

The VariantPhaser algorithm performs read-based phasing of variants from a VCF by using read information from a BAM file containing long reads.

The input to the VariantPhaser algorithm is a BAM file and a VCF file; its output is the VCF file after phasing.

The VariantPhaser algorithm requires the following ALGO_OPTION:

- `-v INPUT`: location of the VCF file containing the variants to be phased. You can use VCF files compressed with bgzip and indexed.

14.3 DRIVER read_filter options

The `--read_filter` argument of the DRIVER binary allows to filter or transform reads from the input BAM file before performing the calculation. It is possible to use multiple `--read_filter` arguments in the same command line, in which case they will be executed sequentially , so the order is important.

The syntax for the argument is: `--read_filter FILTER,OPTION=VALUE,OPTION=VALUE,. . .`

14.3.1 QualCalFilter read_filter

The QualCalFilter read filter is used to transform reads and perform base quality score recalibration while modifying the information contained in the recalibration table.

The QualCalFilter read filter requires one of the following OPTION:

- table=TABLE_FILEPATH: the location of the recalibration table that will be used as the basis to perform the base quality score recalibration.
- use_oq=[true/false]: determines whether to use the original base quality scores contained in the OQ tag in the BAM file. This option cannot be used in conjunction with the table option and is used to undo base quality score recalibration by setting the base quality scores of the output to the ones contained in the OQ tag in the input BAM file. Typically this option will be used before a second QualCalFilter read filter to first undo a possible recalibration done on the input BAM file.

The QualCalFilter read filter accepts the following optional OPTION:

- prior=PRIOR: determines the global bias for all the base quality scores.
- min_qual=QUAL: determines the quality threshold to perform recalibration; bases with quality scores less than QUAL will not be recalibrated.
- levels=LEVEL1/LEVEL2/...: determines the static quantization levels of the base quality scores.
- indel=[true/false]: determines whether to add the base quality scores for INDELs into the BAM tags.
- keep_oq=[true/false]: determines whether to keep the original before recalibration base quality scores by using the OQ tag in the bam file.

14.3.2 OverclippingFilter read_filter

The OverclippingFilter read filter is used to filter reads depending on their soft clipping characteristics.

The OverclippingFilter read filter accepts the following optional OPTION:

- `min_align_len=LENGTH`: filter reads where the number of bases that are not soft clipped is less than LENGTH.
- `count_both_ends=[true/false]`: if set to true, only filter reads where both ends of the read are soft clipped, so that reads with soft-clipping on one end only will not be filtered regardless of their non soft-clipped length. The default value is true.

14.3.3 SimplifyCigarTransform read_filter

The SimplifyCigarTransform read filter is used to simplify and standardize cigars with following rules:

- Operator with zero length: drop.
- EQUAL operator (`=`) transform to MATCH operator (`M`).
- DIFF operator (`X`) transform to MATCH operator (`M`).
- Adjacent INS (`I`) and DEL (`D`) operators are merged to MATCH (`M`).

- Trailing DEL operator (`D`): drop.

15 BWA binary

The BWA binary performs alignment of DNA-seq data.

In Sentieon® version 202112, a bug that resulting in an incorrect MAPQ for a small number of alignments in the original bwa mem is fixed. A compatibility environment variable `ksw_compat=1` is provided and setting this environment variable will cause Sentieon® bwa mem behave the same way as described in http://bio-bwa.sourceforge.net/bwa.shtml, providing an incorrect MAPQ for some alignments. The default behavior implements the correct MAPQ calculation for all alignments and provides a speedup on some newer hardware.

The BWA binary has two modes of interest, "mem" mode to align FASTQ files against a reference FASTA file, and "shm" mode to load the FASTA index file in memory to be shared among multiple BWA processes running in the same server.

15.1 BWA mem syntax

You can run the following command to align a single-ended FASTQ1 file or a pair-ended set of 2 FASTQ files against the FASTA reference, which will produce the mapped reads to stdout, to be piped onto util sort:

```
<SENTIEON_FOLDER>/bin/sentieon bwa mem OPTIONS FASTA FASTQ1 [FASTQ2]
```

The arguments (OPTIONS) for this command include:

- `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The argument is optional; if omitted the bwa binary will use 1 thread.
- `-p`: determines whether the first input FASTQ file contains interleaved pair-ended reads. If this argument is used, only use a single FASTQ input, as the second FASTQ2 file will be ignored.
- `-M`: determines whether to make split reads as secondary.
- `-R READGROUP_STRING`: Read Group header line that all reads will be attached to. The recommended READGROUP_STRING is `@RG\tID:$readgroup\tSM:$sample\tPL:$platform\tPU:$platform_unit`
  - $readgroup is a unique ID that identifies the reads.
  - $sample is the name of the sample the reads belong to.
  - $platform is the sequencing technology, typically ILLUMINA.
  - $platform_unit is the sequencing element that performed the sequencing.

- `-K CHUNK_SIZE`: determines the size of the group of reads that will be mapped at the same time. If this argument is not set, the results will depend on the number of threads used.

15.2 BWA shm syntax

You can run the following command to load the FASTA index file in memory:

```
<SENTIEON_FOLDER>/bin/sentieon bwa shm FASTA
```

You can run the following command to list FASTA indices files stored in memory:

```
<SENTIEON_FOLDER>/bin/sentieon bwa shm -l
```

You can run the following command to remove all FASTA indices files stored in memory, thus freeing memory when no longer necessary:

```
<SENTIEON_FOLDER>/bin/sentieon bwa shm -d
```

The arguments (OPTIONS) for this command include:

- `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The argument is optional; if omitted the bwa binary will use 1 thread.
- `-f FILE`: location of a temporary file that will be used to reduce peak memory usage.

15.3 Controlling memory usage in BWA

By default BWA will use about 24 GB in a Linux system and 8 GB in a Mac system. You can control the memory usage via the bwt_max_mem environment variable, which can be used to enhance the speed performance by using more memory, or limit the memory usage at the expense of speed performance. For example, you will get faster alignment by adding the following to your scripts:

```
export bwt_max_mem=50G
```

Bear in mind that the number you use in the `bwt_max_mem` environmental variable is not a hard limit, but an estimate of the memory used in BWA; as such, if BWA memory usage does not go beyond anything lower a certain value, that means it is the minimum required memory for the specific reference, setting `bwt_max_mem` to a smaller value than the minimum required memory won't change the BWA mem jobs's memory usage.

15.4 Using an existing BAM file as input

If you do not have access to the FASTQ inputs, but only have an already aligned and sorted BAM file, you can use it as input and redo the alignment by running *samtools*:

```
samtools collate -@ 32 -Ou INPUT_BAM tmp- | samtools fastq -@ 32 -s \
/dev/null -0 /dev/null - | <SENTIEON_FOLDER>/bin/sentieon bwa mem -t 32 -R \
'@RG\tID:id\tLB:lib\tSM:sample\tPL:ILLUMINA' -M -K 1000000 -p $ref /dev/stdin \
| <SENTIEON_FOLDER>/bin/sentieon util sort -t 32 -o OUTPUT_BAM --sam2bam -
```

Alternatively, you could first create the FASTQ files and then process them as you would normally do:

```
samtools collate -n -@ 32 -uO INPUT_BAM tmp- | samtools fastq -@ 32 \
-s >(gzip -c > single.fastq.gz) -0 >(gzip -c > unpaired.fastq.gz) \
-1 >(gzip -c > output_1.fastq.gz) -2 >(gzip -c > output_2.fastq.gz) -
```

If you do this, you may encounter an abnormal memory usage in BWA; if that is the case, you can follow the instructions in *BWA uses an abnormal amount of memory when using FASTQ files created from a BAM file* .

16

minimap2 binary

The minimap2 binary performs alignment of PacBio or Oxford Nanopore genomic reads data and will behave the same way as the tool described in https://github.com/lh3/minimap2.

17 STAR binary

The STAR binary performs alignment of RNA-seq data and will behave the same way as the tool described in https://github.com/alexdobin/STAR.

17.1 Using compressed FASTQ.gz input files

Using compressed FASTQ input files requires the use of the STAR option *–readFilesCommand COMMAND* to determine what external program will be used to decompress the input files. When using the Sentieon® implementation of STAR together with Sentieon® util sort, the efficient procesing of the inputs could cause the decompression to be a bottleneck, so it is important to use a fast decompression method.

18 UTIL binary

UTIL is the binary used to run some utility functions. This binary is mainly used to process the raw reads output from BWA.

18.1 UTIL syntax

The general syntax of the UTIL binary is:

```
sentieon util MODE [OPTIONS]
```

The supported modes (**MODE**) for this command are:

- index: build the index for a BAM file. The following command will generate a bai BAM index file at the same location as the input file:

  ```
  sentieon util index INPUT.bam
  ```

- vcfindex: build the index for a VCF file. The following command will generate a idx VCF index file at the same location as the input file:

  ```
  sentieon util vcfindex INPUT.vcf
  ```

- sort: sort a BAM file. The optional arguments (OPTIONS) for the UTIL command using the sort MODE include:
  - `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The default is as many threads as available in the server.
  - `-r REFERENCE`: location of the reference FASTA file. This argument is required if you are using a CRAM output file, otherwise it is optional.
  - `-i INPUT`: location of the input file.
  - `-o OUTPUT`: the location and filename of the output file.
  - `--temp_dir DIRECTORY`: determines where the temporary files will be stored. The default is the folder where the command is run (`$PWD`).
  - `--cram_read_options decode_md=0`: CRAM input option to turn off the NM/MD tag in the input CRAM.

  - `--cram_write_options compressor=[gzip|bzip2|lzma|rans|tok|fqz|arith]`: CRAM output compression options. compressor=gzip+rans is default if not defined.
  - `--cram_write_options version=[3.0|3.1]`: CRAM output version options. version=3.0 is default if not defined.
  - `--bam_compression COMPRESSION_LEVEL[0-9]`: gzip compression level for the output BAM file. The default value is 6.
  - `--sam2bam`: indicates that the input will be in the form of an uncompressed SAM file, that needs to be converted to BAM. If this option is not used, the input should have been converted to BAM format from the BWA output using samtools.
  - `--block_size BLOCK_SIZE`: size of the block to be used for sorting.
  - `--trim_primer AMPLICON_TABLE,clip=CLIPPING`: determines whether to mark primers as soft/hard clips, depending on whether the value of CLIPPING is *soft* or *hard*. When using option `--trim_primer AMPLICON_TABLE,clip=soft` you will need to make sure to use the `--trim_soft_clip` option in any subsequent variant caller command so that the soft-clipped primers are ignored. The primers are determined based on the information provided in the AMPLICON_TABLE. The AMPLICON_TABLE is a BED file containins the location of the primers in the amplicon sequencing as a tab-delimited file with eigth columns: contig, amplicon_start, amplicon_end, name (ignored), score (ignored), strand (ignored), insert_start, insert_end; the tool will trim bases from amplicon reads between amplicon_start and insert_start, and between amplicon_end and insert_end.

```
sentieon util sort -t NUMBER_THREADS --sam2bam -i INPUT.sam -o OUTPUT.bam
```

- vcfconvert: compress and decompress VCF and GVCF files.

  The following command will compress and index the input file:

```
sentieon util vcfconvert INPUT.vcf OUTPUT.vcf.gz
```

  The following command will decompress a non-indexed vcf file generated with gzip and then compress and index the file. When using this command make sure that the INPUT and OUTPUT files are not the same:

```
sentieon util vcfconvert INPUT.gz OUTPUT.vcf.gz
```

- stream: perform base quality correction in streaming mode. The optional arguments (OPTIONS) for the UTIL command using the stream MODE include:
  - `-r REFERENCE`: location of the reference FASTA file. This argument is required if you are using a CRAM output file, otherwise it is optional.
  - `-i INPUT`: location of the input file. By default the binary will use stdin as the input.
  - `-q RECAL_TABLE`: location of the recalibration table.
  - `-t NUMBER_THREADS`: number of computing threads that will be used by the software to run parallel processes. The default value is 1.
  - `-o OUTPUT`: the location and filename of the output file. By default the binary will output to stdout, to be able to stream the results.
  - `--output_format FORMAT`: determines the format of the output stream. The possible FORMAT values are BAM or CRAM. The default is BAM.
  - `--output_index_file OUTPUT_INDEX`: determines where the corresponding index file will be created. If this option is ommitted, the tool will not generate an index file.

- `--read_filter FILTER,OPTION=VALUE,OPTION=VALUE`: perform a filter or transformation of reads prior to the application of the algorithm. Please refer to *DRIVER read_filter options* in the driver usage to get additional information on the available filters and their functionality.
- `--cram_read_options decode_md=0`: CRAM input option to turn off the NM/MD tag in the input CRAM.
- `--cram_write_options compressor=[gzip|bzip2|lzma|rans|tok|fqz|arith]`: CRAM output compression options. compressor=gzip+rans is default if not defined.
- `--cram_write_options version=[3.0|3.1]`: CRAM output version options. version=3.0 is default if not defined.
- `--bam_compression COMPRESSION_LEVEL[0-9]`: gzip compression level for the output BAM file. The default value is 6.

The following command will apply the recalibration and output the corresponding recalibrated BAM file to stdout:

```
sentieon util stream -i INPUT.bam -q RECAL_TABLE \
  --output_index_file OUTPUT.bam.bai -o -
```

The above command will not generate an index file unless the option `output_index_file` is included.

19 UMI binary

UMI is the binary used to process reads containing UMI sequences.

19.1 UMI syntax

The general syntax of the UMI binary is:

```
sentieon umi MODE [OPTIONS]
```

The supported modes (**MODE**) for this command are:

- extract: pre-process FASTQ files containing reads with UMI sequences. The syntax of the extract MODE is:

  ```
  sentieon umi extract [OPTIONS] read_structure fastq1 [fastq2] [fastq3]
  ```

  where:

  - `read_structure` is the logical structure of the reads. It consists of a collection of integer+character pairs describing the #bases+type; the type can be M for molecular barcode, T for template and S for skip. The read structure consists of comma separated groups, where each group will be read from the corresponding input FASTQ (first group from first FASTQ, second group from second FASTQ...)
  - `fastq1/2/3` are the FASTQ files. Up to 3 input FASTQ files are supported to allow the use case when the UMI sequence is already in a separate FASTQ file.

  The optional arguments (OPTIONS) for the UMI binary using the extract MODE include:

  - `-o OUTPUT`: the location and filename of the output file. If ommitted, the output will be stdout.
  - `-d`: if present, the extraction will be done in duplex mode.
  - `--umi_tag TAG`: the logic UMI tag. The default value is *XR*.

20 PLOT script

PLOT is a script used to create plots of the results of the metrics and recalibration stages. The plots are stored in a PDF file.

20.1 PLOT syntax

The general syntax of the PLOT script is:

```
sentieon plot STAGE -o OUTPUPT_FILE INPUTS [OPTIONS]
```

The supported modes (STAGE) for this command are:

- GCBias: generate PDF file from the metrics results of GCBias.
- QualDistribution: generate PDF file from the metrics results of QualDistribution.
- InsertSizeMetricAlgo: generate PDF file from the metrics results of InsertSizeMetricAlgo.
- MeanQualityByCycle: generate PDF file from the metrics results of MeanQualityByCycle.
- QualCal: generate PDF file from the BQSR QualCal tool.
- VarCal: generate PDF file from the VQSR VarCal tool.

20.1.1 PLOT results of GCBias STAGE

The INPUTS to generate the plots from the GCBias stage are:

- GC_METRIC_TXT: where GC_METRIC_TXT is the output file of the GCBias algorithm from *GCBias ALGORITHM*.

The plotting of the GCBias metrics STAGE does not accept any OPTIONS.

20.1.2 PLOT results of MeanQualityByCycle STAGE

The INPUTS to generate the plots from the MeanQualityByCycle stage are:

- MQ_METRIC_TXT: where MQ_METRIC_TXT is the output file of the MeanQualityByCycle algorithm from *MeanQualityByCycle ALGORITHM*.

The plotting of the MeanQualityByCycle metrics STAGE does not accept any OPTIONS.

20.1.3 PLOT results of QualDistribution STAGE

The INPUTS to generate the plots from the QualDistribution stage are:

- QD_METRIC_TXT: where QD_METRIC_TXT is the output file of the QualDistribution algorithm from *QualDistribution ALGORITHM*.

The plotting of the QualDistribution metrics STAGE does not accept any OPTIONS.

20.1.4 PLOT results of InsertSizeMetricAlgo STAGE

The INPUTS to generate the plots from the InsertSizeMetricAlgo stage are:

- IS_METRIC_TXT: where IS_METRIC_TXT is the output file of the InsertSizeMetricAlgo algorithm from *InsertSizeMetricAlgo ALGORITHM*.

The plotting of the InsertSizeMetricAlgo metrics STAGE does not accept any OPTIONS.

20.1.5 PLOT results of QualCal STAGE

The INPUTS to generate the plots from the bqsr stage are:

- RECAL_RESULT.CSV: the output csv file of the QualCal algorithm from *QualCal ALGORITHM*.

The plotting of the bqsr STAGE does not accept any OPTIONS.

20.1.6 PLOT results of VarCal STAGE

The INPUTS to generate the plots from the vqsr stage is:

- PLOT_FILE: a file created by the VarCal algorithm from *VarCal ALGORITHM* containing the data required to create the report.

The plotting of the vqsr STAGE accept the following OPTIONS:

- tranches_file=TRANCHES_FILE: location of the file containing the partition of the call sets into quality tranches, generated by the VarCal algorithm from *VarCal ALGORITHM*.
- target_titv=TITV_THRES: expected TiTv number for the species; it is used calculate the True Positive and False Positive numbers in the plot.
- min_fp_rate=MIN_RATE: minimum False Positive number; it is used calculate the True Positive and False Positive numbers in the plot.

21 LICSRVR binary

LICSRVR is the binary used to run the license server to facilitate dynamic license assignment and record license utilization within a cluster.

21.1 LICSRVR syntax

The general syntax of the LICSRVR binary is:

```
<SENTIEON_FOLDER>/bin/sentieon licsrvr [--start|--stop] [--log LOG_FILE] LICENSE_FILE
```

The following inputs are optional for the command:

- **LOG_FILE**: location and filename of the output file containing the log of the server.
- **LICENSE_FILE**: location of the server license file.

After the license server is operational, the client applications can request license tokens from the server by setting the SENTIEON_LICENSE environment variable to the server address in the form of HOST:PORT.

The licsrvr binary supports the following additional modes:

- `--version`: will report the software package version the binary belongs to.
- `--dump`: will report the current status of the license server, including the number of available licenses.

  ```
  <SENTIEON_FOLDER>/bin/sentieon licsrvr --dump LICENSE_FILE
  ```

- `--dump=update`: if the license information has been updated and automatically pulled by the license server, this mode will dump the updated license information to the stdout. The following command will report the updated license information the license server has, if there has been any change:

  ```
  <SENTIEON_FOLDER>/bin/sentieon licsrvr --dump=update LICENSE_FILE
  ```

22 LICCLNT binary

LICCLNT is the binary used to test the license server functionality to help determine whether the license server is operational, and how many licenses of the different algorithms are available.

22.1 LICCLNT syntax

The LICCLNT binary has two modes, one to ping the license server and one to check the available licenses for specific algorithms.

You can run the following command to check if the license server is operational:

```
<SENTIEON_FOLDER>/bin/sentieon licclnt ping --server HOST:PORT
```

The command will return 0 if the server is operational.

You can run the following command to check how many licenses are:

```
<SENTIEON_FOLDER>/bin/sentieon licclnt query --server HOST:PORT FEATURE
```

The command will return the number of licenses that are available for the specific license feature, which can be used for managing your jobs: before submitting a job on a certain number of threads, you can check if there are enough licenses for those threads, preventing the tool from being idle while waiting for licenses.

23 Troubleshooting

23.1 Preparing reference file for use

If your reference FASTA file has not been pre-processed such that the data specified in *Data requirements for the reference nucleotide sequence* is not available to the software, you will need to process it as explained in https://www.broadinstitute.org/gatk/guide/article?id=2798.

You will need to do the following steps:

1. Generate the BWA index using BWA. This will create the ".fasta.amb", ".fasta.ann", ".fasta.bwt", ".fasta.pac" and ".fasta.sa" files.

```
sentieon bwa index reference.fasta
```

2. Generate the FASTA file index using samtools. This will create the ".fasta.fai" file.

```
samtools faidx reference.fasta
```

3. Generate the sequence dictionary using Picard. This will create the ".dict" file.

```
java -jar picard.jar CreateSequenceDictionary REFERENCE=reference.fasta \
  OUTPUT=reference.dict
```

23.2 Preparing RefSeq file for use

RefSeq files are used to aggregate the results of the CoverageMetrics algorithm to the gene level.

In order to use RefSeq files downloaded from the ucsc genome browser, they need to be sorted by chromosome and loci. To perform the sorting you will need to do the following steps:

1. Strip the header from the file

```
grep -v "^#" FILE.refSeq > FILE.refSeq.headerless
grep -e "^#" FILE.refSeq > FILE.refSeq.header
```

2. Sort the loci first using unix sort.

```
sort -k 5 -n FILE.refSeq.headerless > FILE.refSeq.presorted
```

3. Use GATK sortByRef.pl (available from https://raw.githubusercontent.com/broadgsa/gatk/3.4/public/perl/sortByRef.pl) using the FASTA index fai to sort by chromosome.

```
perl sortByRef.pl --k 3 FILE.refSeq.presorted FASTA.fai --tmp ~/tmp \
    > FILE_sorted_headerless.refSeq
```

4. Put the header back to the file.

```
cat FILE.refSeq.header > FILE_sorted.refSeq
cat FILE_sorted_headerless.refSeq >> FILE_sorted.refSeq
```

23.3 Common usage problems

Following is a list of symptoms of common problems as well as solutions for them.

23.3.1 Driver or Util fails with Error: can not open file (xxx) in mode(r), Too many open files

The root cause for this error is that the limit of concurrently open files is not set to be high enough for your system.

You can solve this error by setting the system ulimit -n. In a Linux based system:

1. Check the limit of maximum number of open files in your system, by running the following command:

```
ulimit -n
```

2. Set a higher limit by editing file `/etc/security/limits.conf` as root, and add the following 2 lines:

```
* soft nofile 16384
* hard nofile 16384
```

3. If your system is running Ubuntu, you also need to add this line to you shell profile `~/.bashrc`

```
ulimit -n 16384
```

4. You need to log out of your system and log back in for the changes to take effect. After logging in, check that the change was applied correctly by running the following command:

```
ulimit -n
```

5. The command should return 16384.

23.3.2 Driver fails with error: Contig XXX from vcf/bam is not present in the reference, or error Contig XXX has different size in vcf/bam than in the reference

The root cause for this error is that the input VCF or BAM file is incompatible with the reference fasta file. Either there are contigs in the file not present in the reference, or the contigs have different sizes. This is most likely caused by using VCF or BAM files processed with a different reference.

23.3.3 Driver reports warning: Contigs in the vcf file XXX do not match any contigs in the reference

The root cause for this warning is that the input VCF file is incompatible with the reference fasta file, and the contigs in the file are not present in the reference. This is most likely caused by using VCF files from a

different reference.

23.3.4 License message: No more license available for Sentieon...

This message is produced when you request to run the Sentieon® software in more threads than your license currently allows you to. This happens because you are concurrently running commands that collectively request more threads than the number of cores your license supports.

The Sentieon® commands will be idle while waiting for free licenses, but the commands will not fail.

23.3.5 Driver fails with error: Readgroup XX is present in multiple BAM files with different attributes

This error is produced when you input two different BAM files containing readgroups with the same ID but different attributes, for instance when in TNseq® and TNscope® the tumor and normal sample BAM files have an RG ID of "1".

Before you are able to use the BAM files you will need to edit them to make the RG ID unique, for instance by adding the SM name to the RG ID. You can check *Modify RG information on BAM files when both Tumor and Normal inputs have the same RGID* for an example of a work-around for this issue.

Alternatively, you can use the samtools addreplacerg functionality to modify the RG ID of the input BAM files and make them unique:

```
#add the new RG and modify all reads in the BAM file
RGtag=$(samtools view -H $INPUT_BAM|grep ^@RG|sed "s|ID:$ORIG_RGID|ID:$NEW_RGID|g")
samtools addreplacerg -r "$RGtag" -o $TMP_BAM $INPUT_BAM
#reheader the BAM to remove the original RG that is no longer used
samtools view -H $TMP_BAM|grep -v "^@RG.*$ORIGINAL_RGID" \
    |samtools reheader - $TMP_BAM > $OUTPUT_BAM
rm $TMP_BAM
```

23.3.6 Driver reports warning: none of the QualCal tables is applicable to the input BAM files

This warning means that none of the information in the recalibration table input file can be applied to the input BAM file, which is likley due to using a recalibration table that does not correspond to the BAM file.

This warning could be produced when the input BAM file to QualCal does not have the correct fields in the RG fields. For instance, this could happen if the PL tag of the RG is set to something different than ILLUMINA; in this case, you will need to modify the BAM header to include/modify the missing/incorrect fields, for which you can use the samtools reheader functionality.

23.3.7 BWA uses an abnormal amount of memory when using FASTQ files created from a BAM file

When you use FASTQ files created by converting an already sorted BAM file, it may happen that all the unmapped reads are grouped together at the end of the FASTQ inputs. In that case, BWA may use an abnormal amount of memory at the end of the alignment because poorly mapped or unmappable reads require additional memory.

In order to reduce the abnormal memory usage, you should first re-sort the bam file to make sure that the unmapped reads are not grouped together. You can use *samtools* to do that:

```
samtools sort -n -@ 32 input.bam | samtools fastq -@ 32 \
-s >(gzip -c > single.fastq.gz) -0 >(gzip -c > unpaired.fastq.gz) \
-1 >(gzip -c > output_1.fastq.gz) -2 >(gzip -c > output_2.fastq.gz) -
```

23.3.8 BWA fails with the error: Killed

This error is produced when BWA receives a SIGKILL signal from the operating system. If the system is low on available memory, SIGKILL may have been sent by the kernel's Out Of Memory (OOM) manager. You can check the kernel logs on your system to confirm the SIGKILL signal was sent by the OOM manager.

To resolve this error, you might reduce BWA's memory usage with the `bwt_max_mem` environmental variable. You can refer to *Controlling memory usage in BWA* for more information.

23.4 KPNS - Known Problems No Solutions

23.4.1 Lack of support for gzipped vcf files not compressed with bgzip

Normal gzipped files do not allow for random or indexed access to the information contained on them, only files compressed with bgzip are indexable. As such, the Sentieon® software does not support gzipped VCF files as input. In order to use these files you will need to uncompress them using gunzip and either use them uncompressed or recompress them with bgzip. Alternatively, you can use `util vcfconvert` to recompress and index the files.

```
sentieon util vcfconvert INPUT.vcf.gz OUTPUT.vcf.gz
```

23.4.2 Lack of support for gzipped fasta files

Currently the software does not support gzipped FASTA files as input. You need to gunzip the files before using them.

23.4.3 FASTQ files required to have SANGER quality format

If your FASTQ files have been encoded with Illumina™ sequencing technology before 1.8, the read quality scores will not be in SANGER format, which may produce unexpected results. The Sentieon® Genomics software will not detect that you are using the unsupported format.

23.4.4 Driver fails with error: ImportError: No module named argparse

This error is produced when running tnhapfilter in an environment where the python version is 2.6.x and the argparse module is not present. You will need to install the argparse module to your python installation; you can do this by running `pip install argparse` or whichever other package manager you use.

24

Release notes and usage changes

24.1 Updates from previous releases

24.1.1 Release 202503.02

Type	Description
Feature	Introduced β-version of Pangenome pipeline with accelerated alignment.
Feature	Improved *InsertSizeMetricAlgo* to include single-end consensus reads in the insert size metrics calculation.
Bug-fix	Solved issue in *GVCFtyper* that caused a third-party tool fail when InbreedingCoeff is *-nan*.
Bug-fix	Solved issue in Consensus Dedup that caused a segmentation fault when the input is RNA reads with UMI.
Bug-fix	Solved issue in TNscope with PoN filtering that incorrectly filtered variants based on locus.
Bug-fix	Removed unnecessary environment variables in plot.

24.1.2 Release 202503.01

Type	Description
Feature	Improved DNAscope model apply speed substantially.
Bug-fix	Solved issue in TNscope that could call a false positive in somatic amplicon data.
Bug-fix	Solved issue that could produce malformed CRAM file under rare circumstances.
Bug-fix	Solved issue in Consensus-based Dedup that could cause an assertion failure when the input is RNA reads with UMI.
Bug-fix	Solved issue in DNAscope that caused a segmentation fault when a machine learning model is used on a non-human sample.

24.1.3 Release 202503

Type	Description
Feature	Added support for hybrid short and long-read variant calling.
Feature	Added support for reading and writing CRAM version 3.1 files.
Feature	Added support in GVCFtyper algorithm for joint calling DNAscope GVCFs from long reads pipeline.
Feature	Added support in TNscope algorithm to output *FAD*, *F1R2*, and *F2R1* annotations.
Feature	Added option in GCBias and AlignmentStat algorithms to match with Picard IS_BISULFITE_SEQUENCED option.
Feature	Reduced memory utilization in HsMetricAlgo algorithm.
Bug-fix	Solved issue in driver where minor inconsistencies in alignment *@SQ* records can cause an error.
Bug-fix	Solved issue in consensus Dedup where paired consensus reads may not have matching read names.
Bug-fix	Solved issue in LongReadSV algorithm that could cause a crash under rare circumstances.
Bug-fix	Solved issue in DNAscope and Haplotyper that could cause an assertion when using the *–bam_output* option.
Bug-fix	Solved issue in consensus Dedup that would cause the job to hang when *–metrics* is turned on.
Bug-fix	Solved issue in TNhaplotyper2 in given mode that may produce incorrect results.
Bug-fix	Solved issue in TNscope that missed low-AF variants at the site of germline homozygous variants.
Other	Removed umi consensus from the software package.
Other	Removed gnuplot binary from the software package.

24.1.4 Release 202308.03

Type	Description
Feature	Support STAR for ARM based CPUs.
Feature	Support minimap2 for ARM based CPUs.
Feature	Optimized performance of DNAModelApply on some newer ARM CPUs.
Feature	Added read filter SimplifyCigarTransform to simplify and standardize cigars.
Feature	Added support for consensus and UMI-aware deduplication metrics in Dedup.
Bug-fix	Solved issue in util sort to handle empty type Z and type H tags in input SAM records.
Bug-fix	Solved some phasing issues in the long-reads DNAscope pipeline.
Bug-fix	Solved potential stack overflow issue when input bam depth is extremely high.

24.1.5 Release 202308.02

Type	Description
Feature	Added support in Haplotyper algorithm to output RAW_MQandDP annotation.
Feature	Added support in GVCFtyper algorithm for gVCFs containing the RAW_MQandDP annotation.
Bug-fix	Solved an issue in GVCFtyper that could cause an assertion when input gVCF file's tbi index is from third-party tool.
Bug-fix	Solved issue in GVCFtyper that could cause a segmentation fault when input gVCF file is from CombineGVCFs.
Bug-fix	Solved issue in TNscope that may miss large INDELs in single-end sequencing data.

24.1.6 Release 202308.01

Type	Description
Feature	Added support in TNhaplotyper2 algorithm to run in distributed mode.
Feature	Added support in ContaminationModel algorithm to run in distributed mode.
Feature	Added support in OrientationBias algorithm to run in distributed mode.
Feature	Added support for CSI index for compressed VCF files.
Feature	Maintenance update of GeneEditEvaluator.
Feature	Added better error message in util sort when input fastq to aligner is not legitimate.
Bug-fix	Solved issue in Haplotyper that could cause an error when using the –bam_output option.
Bug-fix	Solved issue in TNscope where variants may report incorrect annotations under rare circumstances.
Bug-fix	Solved issue in LongReadSV that failed to report variants in non-canonical chromosomes.
Bug-fix	Solved issue in util sort when –output_format CRAM is used without a reference.

24.1.7 Release 202308

Type	Description
Feature	Improved DNAscope pipeline speed and accuracy with a BWA model.
Feature	Improved long-reads DNAscope pipeline speed and accuracy with minimap2 models.
Feature	Improved speed of STAR when input format is SAM.
Feature	Improved speed of deduplication on RNA data.
Feature	Improved consensus based deduplication and UMI barcode aware deduplication for RNA data.
Feature	Improved consensus of INDELs in Dedup algorithm.
Feature	Improved speed of DNAscope on long-reads small variant call.
Feature	Improved machine learning model for TNscope.
Feature	Introduced LongReadUtil to extract cell barcode from Oxford Nanopore single-cell RNA data.
Feature	Introduced α-version of GeneEditEvaluator algorithm for genome editing sequence analysis.
Feature	Added support in Dedup to handle INDEL errors in barcode.
Feature	Added support to encapsulate multiple machine learning models into a single file.
Feature	Added support in GVCFtyper to output PL and ML FORMAT fields.
Feature	Added support in umi extract to output SAM format.
Feature	Added support for small variant calling on Oxford Nanopore data.
Feature	Maintenance update of STAR.
Feature	Maintenance update of minimap2.
Feature	Reduced memory utilization in util sort for long reads.
Bug-fix	Solved an issue in umi consensus that could cause an assertion failure under rare circumstances.
Bug-fix	Solved issue in DNAscope that may filter softclip reads at high coverage sites.

24.1.8 Release 202112.07

Type	Description
Feature	Introduced β-version of consensus based deduplication and UMI barcode aware deduplication for RNA data.
Bug-fix	Solved an issue in Dedup that could cause an assertion failure when the output format is CRAM.
Bug-fix	Solved issue in util sort to handle type B tags in input SAM records.
Bug-fix	Improved license server fault tolerance upon extreme high usage demand.

24.1.9 Release 202112.06

Type	Description
Feature	Added support in LocusCollector and Dedup algorithms to perform consensus based deduplication as well as UMI barcode aware deduplication.
Feature	Added support in Dedup algorithm to perform UMI barcode error correction.
Feature	Added support in GVCFtyper for joint calling DNAscope GVCFs from multiple sequencing platforms into a single multi-sample VCF.
Feature	Added support in GVCFtyper for gVCF files without PL fields.
Feature	Added support in LongReadSV algorithm to process Oxford Nanopore long reads in addition to PacBio HiFi long reads.
Feature	Improved TNscope SV to better handle different representations of the same alignment.
Bug-fix	Solved issue in umi consensus that could cause a segmentation fault when input has 0-length read.
Bug-fix	Solved issue in GVCFtyper that would produce <NON_REF> alleles under rare circumstances.
Bug-fix	Solved issue that Plot GCBias was left-shifted on some datasets.

24.1.10 Release 202112.05

Type	Description
Feature	Maintenance update of UltimaReadFilter.
Feature	Added option in TNscope to control adapter trimming.
Feature	Added environment variable in bwa mem for bug compatibility.
Feature	Added wrapper script for STAR in the bin directory to remove the requirement of setting LD_LIBRARY_PATH.
Bug-fix	Solved issue in STAR that printed out extra information with the argument –version.
Bug-fix	Solved issue in duplex umi consensus that could output zero-length reads.
Bug-fix	Solved issue in umi consensus that slowed down execution under rare circumstances.
Bug-fix	Solved issue in TNscope such that evidence from overlapping read pairs were not adequately accounted for.

24.1.11 Release 202112.04

Type	Description
Feature	Added support for DNAscope with machine learning model to generate gVCF file for subsequent joint call.
Feature	Introduced β-version of LongReadSV algorithm to perform germline structure variant call for long reads.
Feature	Added options in InsertSizeMetricAlgo to control the threshold to output FR, TANDEM, RF.
Feature	Maintenance update of read filter.
Feature	Improved reference assembly detection logic.

24.1.12 Release 202112.02

Type	Description
Feature	A license framework to support multiple sequencing platforms.
Feature	Improved bwa-mem speed on certain ARM CPU.
Bug-fix	Solved issue in GVCFtyper that would produce no-calls in certain ploidies when input is a multi-sample merged gVCF from third party tool.
Bug-fix	Solved issue in DNAscope that could produce no call at zero coverage site when using –given.
Bug-fix	Solved issue that triggered CRAM buffer overflow when minimap2 output zero-length secondary reads.
Bug-fix	Solved issue in minimap2 that would cause the job to hang under rare circumstances.

24.1.13 Release 202112.01

Type	Description
Feature	Improved accuracy of DNAscope machine learning model compared to NIST truth-sets v4.2.1.
Bug-fix	Solved issue in TNscope that missed SNPs with 1% AF under very sensitive settings.
Bug-fix	Solved issue in TNscope that failed to report variants when the sequence length is shorter than 30bp.
Bug-fix	Solved issue in TNscope that could report incorrect AF number for Amplicon data.
Bug-fix	Solved issue in umi consensus that could cause a crash under rare circumstances.
Bug-fix	Added additional error checks and error reporting in util sort.

24.1.14 Release 202112

Type	Description
Feature	Improved speed of bwa mem.
Feature	Improved speed of minimap2 when -x splice:hq argument is used.
Feature	Improved speed of LocusCollector when input bam has huge number of contigs.
Feature	Implemented a low memory usage mode for extremely large GVCFtyper jobs.
Feature	Added option in TNhaplotyper2 to match with GATK –genotype-germline-sites option.
Feature	Added ELEMENT platform to support Element Biosciences sequencers.
Feature	Modified TNscope® AF output format.
Feature	Added support in GVCFtyper to take multi-sample gVCF files with coverage gaps.
Bug-fix	Solved matching issue in STAR when using non-default value for chimMultimapNmax.
Bug-fix	Solved buffer overflow issue in STAR due to base count difference between input R1 and R2.
Bug-fix	Solved issue in TNhaplotyer2 that could cause matching difference with GATK-4.2.0.0 when the input bam has high QUAL N bases.
Bug-fix	Solved issue in TNhaplotyper2 that could cause a crash under rare circumstances.
Bug-fix	Solved minor matching issue in median annotations in TNhaplotyper2.
Bug-fix	Solved issue that triggered stack overflow when input bam has substantially dense region.
Bug-fix	Solved issue in duplex umi consensus that results depend on internal data order.
Bug-fix	Added additional error checks and error reporting.

24.1.15 Release 202010.04

Type	Description
Feature	Improved speed of Minimap2.
Feature	Mantainance update of Minimap2
Bug-fix	Solved issue in VariantPhaser that could cause a crash under rare circumstances.
Bug-fix	Solved issue in VariantPhaser that could output variants out of order.
Bug-fix	Solved issue in DNAscope that could cause an error when running with HiFi data.
Bug-fix	Reduced memory utilization in DNAscope when using HiFi data aligned with minimap 2.22.
Bug-fix	Added additional error checks and error reporting.
Bug-fix	Updated sentieon wrapper to default to use python system binary, then python3 and last python2.

24.1.16 Release 202010.03

Type	Description
Feature	Introduced β-version of PacBio HiFi reads pipeline.
Feature	Introduced β-version of VariantPhaser tool to do read-based phasing for long reads.
Feature	Introduced β-version of minimap2 aligner for PacBio and Nanopore reads.
Feature	Added support in the sentieon wrapper to run Sentieon python scripts.
Bug-fix	Added additional error checks and error reporting.

24.1.17 Release 202010.02

Type	Description
Feature	Maintenance update of the TNseq algorithm.
Feature	Added support in util for amplicon specific primer clipping.
Feature	Added support to access s3 objects from a region different than the one hosting the bucket.
Feature	Improved speed of TNhaplotyper2 for extreme depth cases.
Feature	Introduced β-version of STAR aligner.
Bug-fix	Solved an issue when using CRAM files that could incorrectly report reads as missing the RG under very rare circumstances.
Bug-fix	Solved issue in CoverageMetrics that could cause a small run-to-run results difference.
Bug-fix	Solved issue when using CRAM that could drop certain malformed reads from output.
Bug-fix	Solved issue in TNhaplotyper2 that could cause an error when using the –bam_out option.
Bug-fix	Solved issue in TNfilter that would cause an error when encountering unrecognized reference bases.
Bug-fix	Solved issue in util sort –umi_post_process that would cause an incorrect BI/BD tag.
Bug-fix	Solved issue when reading BAM files with convoluted PGs that would cause a crash.
Bug-fix	Solved issue in TNhaplotyper, TNhaplotyper2 and TNscope that could cause a crash when using options –given and –bam_output.
Bug-fix	Solve issue in TNhaplotyper2 that could produce incorrect PS values.
Bug-fix	Solved issue in some metrics algorithms that was producing trailing tabs causing an empty column output.
Bug-fix	Added additional error checks and error reporting.

24.1.18 Release 202010.01

Type	Description
Feature	Added support in TNhaplotyper2 for genotyping given alleles.
Feature	Improved TNscope® method for handling read trimming.
Feature	Added support in TNscope® for amplicon specific primer removal.
Bug-fix	Added additional error checks and error reporting.
Bug-fix	Fixed issue in TNscope® that could report incorrect AD numbers at tri-allelic sites.
Bug-fix	Solved issue in ContaminationModel that could calculate incorrect estimates when using very sparse pileups.
Bug-fix	Solved issue in Haplotyper that could cause a crash for inputs with high depth close to the start of a non-canonical chromosome.
Bug-fix	Solved issue in umi consensus that could cause the sequence and BI/BD to be inconsistent when using duplex UMI.
Bug-fix	Solved issue in ContaminationModel that could cause a crash under rare circumstances.
Bug-fix	Solved issue that could cause a crash when using the Sentieon® libraries with bcl2fastq.

24.1.19 Release 202010

Type	Description
Feature	Reduced memory usage when outputting CRAM files.
Feature	Maintenance update of the TNseq algorithm.
Feature	Added DNBSEQ platform to support BGI sequencers.
Feature	Improve speed of umi consensus when not outputting BI/BD tags.
Feature	Added support of BGZF compressed interval list files as well as VCF and VCF.gz files.
Feature	Modified the type of the AD, QSS and RPA FORMAT field in the VCF.
Feature	Override unlimited stack size limit in Linux to prevent Linux bug with thread local storage allocation.
Feature	Using an empty interval bed file will result in no processing done, as if the interval had 0 length.
Bug-fix	Added additional error checks and error reporting.
Bug-fix	Solved issue that would cause a crash when using BAM files containing reads without a sequence.
Bug-fix	Added option in util sort to handle supplementary reads after processing with the umi tools.
Bug-fix	Solved issue in umi consensus that could cause results to depend on the number of threads.
Bug-fix	Solved issue in Haplotyper that could cause an out of bound memory access error in very rare circumstances
Bug-fix	Solved issue in util sort that could cause it to hang for single threaded jobs under very rare circumstances.
Bug-fix	Solved issue in umi consensus that could cause a segmentation fault.
Bug-fix	Solved issue in umi consensus that produced a non informative group 0 histogram when using –min_reads 1.
Bug-fix	Error out when creating CRAM file containing reads beyond the reference end.
Bug-fix	Solved issue in plot that would not respect the SENTIEON_TMPDIR environmental variable.

24.1.20 Release 201911.01

Type	Description
Feature	Reduced number of temporary files in util sort.
Feature	Improved UMI consensus calculation for INDELs.
Feature	Reduced memory usage in UMI tool.
Feature	Added support in umi extract to read FASTQ files containing UMI tags already extracted.
Feature	Added support in Dedup metrics to report QCFAIL reads if present.
Feature	Made the rsID field population from dbSNP more robust.
Bug-fix	Solved issue that would cause a crash when using CRAM files and unmapped reads of zero length.
Bug-fix	Solved issue in SequenceArtifactMetricsAlgo that could cause a negative result.
Bug-fix	Solved issue in BWA that could cause an error when using a value larger than 520g for bwt_max_mem option.
Bug-fix	Solved issue in the vcflib python library that could cause an error processing VCFs from SVSolver
Bug-fix	Added additional error checks and error reporting.
Bug-fix	Solved issue in umi extract that could cause a crash.
Bug-fix	Solved issue that could slow down merging of large output files.
Bug-fix	Solved issue that could generate an incorrect CRAM index file when a slice contained multi contigs.
Bug-fix	Solved issue in licsrvr for Mac that could prevent it from stopping when running the –stop command.
Bug-fix	Solved issue that could cause a crash when using CRAM files and reads longer than 128KBases.
Bug-fix	Solved issue in util sort that could cause a crash with very large SAM records.
Bug-fix	Solved issue in Haplotyper that could cause an out of bound memory access error in very rare circumstances
Bug-fix	Solved issue in DNAscope that could produce no GT when using –given.
Bug-fix	Solved issue in CollectVCMetrics that could cause incorrect results for INDELs straddling beyond the input interval.

24.1.21 Release 201911

Type	Description
Feature	Introduced β-version of tools for processing reads containing UMI sequences.
Feature	Introduced β-version of software for ARM based CPUs.
Feature	Added support in ReadWriter to filter reads based on the mapping quality.
Feature	Maintenance update of BWA
Feature	Improved speed of util sort on high CPU count servers.
Feature	Added support in ReadWriter to filter reads based on their flags.
Feature	Added support to GVCFtyper to output very small AF numbers in scientific notation.
Feature	Modified the type of the AD FORMAT field in the VCF.
Feature	Added support in TNscope® to output ID and MATEID in its BND output.
Bug-fix	Solved issue in VarCal that could cause a numeric overflow
Bug-fix	Solved issue in Haplotyper that could cause a segmentation fault under very rare circumstances.
Bug-fix	Solved issue in TNscope® calling SVs that would output an invalid AD value for the normal sample, if present.
Bug-fix	Solved issue in DNAscope that would prevent generating a GVCF when using sharded mode.
Bug-fix	Solved issue in VarCal that would cause an issue when using a VCF containing an INF value annotation.
Bug-fix	Solved issue in Haplotyper and DNAscope that would not trim common bases in REF and ALT in a variant with a spanning delete.
Bug-fix	Solved issue in TNscope® SV calling that could shift the position by a few bases under rare circumstances.
Bug-fix	Solved issue in GVCFtyper that would create a SOR=nan for large cohorts
Bug-fix	Solved issue in the DNAscope Smith-Waterman that could cause a results difference in very rare circumstances.
Bug-fix	Solved issue that could cause an error when inputing sharded VCF files indexed by other tools.
Bug-fix	Added additional error checks and error reporting.

24.1.22 Release 201808.08

Type	Description
Bug-fix	Solved issue in Haplotyper that could create an incorrect tbi index file.
Bug-fix	Solved issue in DNAscope and Haplotyper that could cause an assertion when using the –bam_output option.
Bug-fix	Solved issue in LocusCollector that could cause an assertion when creating a compressed score file.

24.1.23 Release 201808.07

Type	Description
Feature	Improved speed of QualCal on high CPU count servers.
Feature	Improved speed of Dedup on high CPU count servers.
Feature	Improved speed of GVCFtyper merge for large cohorts.
Bug-fix	Solved issue preventing reading CRAM files with slices straddling across multiple contigs; the Sentieon® tools do not generate such CRAM files.
Bug-fix	Added additional error checks and error reporting.
Bug-fix	Solved issue that would cause an error when using BAM files containing incomplete PG records.
Bug-fix	Solved issue in GVCFtyper that would cause an error when using phased GT annotation.

24.1.24 Release 201808.06

Type	Description
Bug-fix	Solved issue in GVCFtyper that could cause an error when using multinomial genotyping and more than 1000 samples.

24.1.25 Release 201808.05

Type	Description
Feature	Improved speed of BWA alignment.
Feature	Changed default CRAM output version to 3.0
Feature	Reduced memory usage when outputting CRAM files.
Feature	Added error checking in tnhapfilter.
Feature	Improved accuracy of DNAscope model.
Feature	Removed Realigner stages from the example files
Feature	Added option in Haplotyer and GVCFtyper to use additional genotyping models.
Bug-fix	Solved issue in TNhaplotyer2 that could cause a run to run variation in the annotation calculation in 1 out of 100 runs.
Bug-fix	Solved issue in BWA that could cause an error when using an extremely large bwt_max_mem option.
Bug-fix	Solved issue that could cause an error when reading a CRAM file generated by cramtools.
Bug-fix	Improved error reporting.

24.1.26 Release 201808.03

Type	Description
Bug-fix	Solved issue in TNscope® that set the wrong default settings.

24.1.27 Release 201808.02

Type	Description
Feature	Improved speed of TNscope®.
Feature	Added option –trim_soft_clip to TNscope®, TNhaplotyper and TNhaplotyper2.
Feature	Added capability for output BAM from –bam_output option to keep the input BAM RG information.
Feature	Added option –disable_detector to TNscope® to control the type of variants called.
Feature	Added support in TNscope® SV calling to more accurately represent large INS.
Feature	Added mode to util fqidx to extract a fraction of the reads.
Feature	Added support in GVCFtyper merge mode to allow input files hosted in an object storage location.
Feature	Added support for using multiple –interval options.
Bug-fix	Solved issue in TNscope® that could call a false positive in a site that has a germline call and is neighboring another SNV.
Bug-fix	Improved error reporting.
Bug-fix	Solved issue in sentieon script that could prevent using a demo license on a shell that is not BASH.
Bug-fix	Solved issue that would prevent BED files containing an interval with identical start and end.
Bug-fix	Solved issue in DNAModelApply that would cause an error when the input VCF file is empty.
Bug-fix	Solved issue in plot that could cause it to run longer than necessary.
Bug-fix	Solved issue in DNAModelApply that would cause an error when over-scheduling.
Bug-fix	Solved issue in util sort that could cause an assertion error when over-scheduling.
Bug-fix	Solved issue in plot that would generate BQSR PDF plots without AA and AAA context covariate.

24.1.28 Release 201808.01

Type	Description
Feature	Improved performance of DNAscope and Machine learning model
Feature	Improved speed of Insert Size plotting when the sample has very large insert sizes
Bug-fix	Solved issue in GVCFtyper that could create a non conforming VCF when doing joint calling of DNAscope results.
Bug-fix	Solved issue in DNAModelApply that would cause an error when the input file does not contain a ModelID.
Bug-fix	Solved issue in Haplotyper when using the –given option that would generate a VCF without the LowQual filter definition in the header.
Bug-fix	Solved issue in tnhapfilter tool to include tumor sample in the header of the VCF

24.1.29 Release 201808

Type	Description
Feature	Added support in ReadWriter for customized flag read filters.
Feature	Added support to modify the read group information of an input BAM file.
Feature	Added support in TNscope® and TNhaplotyper to output a BAM including local reassembly of the reads.
Feature	Added support in TNModelApply to include command line to the VCF output.
Feature	Added support to QualCalFilter to keep the base quality scores before recalibration.
Feature	Introduced TNhaplotyper2 algorithm
Feature	Added support for outputting CRAM v 3.0 files.
Feature	Added support in Haplotyper, DNAscope and GVCFtyper to input the expected heterozygosity value used to compute prior likelihoods.
Feature	Updated interface of plot to better match other tools.
Feature	Added support for machine learning model for DNAscope
Feature	Modified the help for driver options to make it more user friendly.
Feature	Improved speed in Dedup.
Feature	Added support in Haplotyper, Genotyper and DNAscope to include @PG line in the output header of the `--bam_output` argument.
Feature	Updated the @PG information when processing in distribution mode to make it more informative.
Bug-fix	Solved issue in GVCFtyper on DNAscope GVCFs that may report incorrect phasing on a multiallelic variant
Bug-fix	Solved issue in util sort that would cause an error in older CPUs.
Bug-fix	Added additional error checks and error reporting.
Bug-fix	Solved issue in LocusCollector that would cause a crash when running 2 commands in parallel outputting to the same file.
Bug-fix	Solved issue in Haplotyper that could create a GVCF with REF allele incorrectly set to N under very rare conditions.
Bug-fix	Solved issue in TNscope® that may use uninitialized values
Bug-fix	Solved issue in AlignmentStat that would report an incorrect command line in the header of the output file.
Bug-fix	Solved issue that could slow down the startup when the LIBDIR contains many files and is stored in a NFS.

24.1.30 Release 201711.05

Type	Description
Bug-fix	Solved issue in in Haplotyper that could create a GVCF file that does not cover the entire region under rare circumstances.

24.1.31 Release 201711.04

Type	Description
Feature	Added support to all BAM writing tools to preserve existing PG tags in the header.
Bug-fix	Solved issue in TNscope® that could cause an error with references containing small contigs.
Bug-fix	Solved issue in SVSolver that could cause a segmentation fault.
Bug-fix	Solved issue in Haplotyper when using the —bam_output option that could generate a debug BAM file incompatible with other tools.
Bug-fix	Solved issue in ApplyVarCal that would generate a non-compliant list of FILTERs if the input file already had existing filters.

24.1.32 Release 201711.03

Type	Description
Feature	Reduced memory usage in GVCFtyper.
Feature	Improved the speed of BWA alignment.
Feature	Added command line to output of metrics tools.
Feature	Reduced memory usage in util sort.
Feature	Reduced TNscope® SV calling runtime for very large samples.
Bug-fix	Solved issue in GVCFtyper merge when using `--split_by_sample` that would cause an error in merging files with a large number of samples.
Bug-fix	Solved issue in WgsMetricsAlgo that would cause an error under very rare circumstances.
Bug-fix	Reduced memory usage in WgsMetricsAlgo and SequenceArtifactMetricsAlgo tools.
Bug-fix	Solved issue in SequenceArtifactMetricsAlgo that would produce an error when using reference FASTA files with non-ACGT bases.
Bug-fix	Solved issue in WgsMetricsAlgo that would cause an error when the `--coverage_cap` argument is not within the valid rage.
Bug-fix	Solved issue in SequenceArtifactMetricsAlgo that would cause an error when the `--context_size` argument is not within the valid rage.

24.1.33 Release 201711.02

Type	Description
Feature	Added support for BWA shm mode.
Feature	Added QC metrics tools: BaseDistributionByCycle, QualityYield, WgsMetricsAlgo, SequenceArtifactMetricsAlgo.
Feature	Improved speed in TNscope® when calling SVs.
Feature	Added support in TNscope® to show progress report during the SV calling.
Feature	Reduced memory usage in TNscope® while calling SVs.
Feature	Introduced β-version of Windows based tools.
Feature	Improved speed for the Windows version.
Feature	Added additional error checks and error reporting.
Feature	Modified the default behavior in DNAscope when only calling short variants to filter out chimeric reads during genotyping.
Feature	Reduced memory usage in GVCFtyper.
Bug-fix	Solved issue in Haplotyper that would cause an error if the input BAM file contained reads where the cigar is composed solely of soft and hard clips.
Bug-fix	Solved issue in TNscope® that could cause a segmentation fault when using an incomplete BAM file.
Bug-fix	Solved issue in TNscope® that would cause a segmentation fault under rare circumstances.
Bug-fix	Solved issue in Haplotyper and DNAscope that could cause an error when using the `bam_output` option.
Bug-fix	Solved issue in Realign that would not do the proper read pairing when dealing with secondary alignments.
Bug-fix	Maintenance update on the SVsolver algorithm.
Bug-fix	Solved issue in licsrvrv for Windows that would prevent it from working with HTTP proxy servers with authentication.
Bug-fix	Solved issue in GVCFtyper that would produce VCF records with non-conformant MQ0 when the input GVCFs include MQ0 annotation.

24.1.34 Release 201711.01

Type	Description
Feature	Improved QualCal speed for very small jobs.
Feature	Added support in licsrvr to report the version.
Feature	Added support in licsrvr to report an update in the license.
Bug-fix	Solved issue in licsrvr that would prevent from working HTTP proxy servers that return auth schemes in multiple header lines.
Bug-fix	Solved issue in Haplotyper, DNAscope, TNhaplotyper and TNscope® that could cause an "illegal instruction" error in AWS.
Bug-fix	Solved issue in TNscope® that would produce and "Error in function boost" under rare conditions.
Bug-fix	Solved issue in util that would prevent running vcfconvert on a compressed vcf.gz file.
Bug-fix	Solved issue in Haplotyper and DNAscope that could cause an error when using the `--bam_output` option.
Bug-fix	Reduced memory usage in Realigner.
Bug-fix	Solved issue in TNhaplotyper and TNscope® that could incorrectly filter a variant as present in the Panel of Normals when the variant is covered by a DEL present in the PoN.
Bug-fix	Changed emit_conf value in the sample scripts to use the default value.
Bug-fix	Solved issue in TNhaplotyper and TNscope® that could produce an assertion error when using a fractured bed file with small intervals close to each other.
Bug-fix	Solved issue in TNscope® that would prevent the job from finishing when prune_factor is set to 0.
Bug-fix	Solved issue in BWA that prevented it from working on certain older AMD cpus.

24.1.35 Release 201711

Type	Description
Feature	Added support in Haplotyper to output a BAM including the local reassembly of the reads.
Feature	Introduced β-version of new functionality for applying a Machine Learning model to help with variant filtration in TNscope®.
Feature	Introduced β-version of new functionality for Python based recipe creation to drive the Sentieon® tools.
Feature	Introduced β-version of new product DNAscope for germline variant calling and germline structural variant calling.
Feature	Added support for printing the command line to stderr log, usually as the first line, before the license check.
Feature	Added support in QualCal to calculate recalibration tables based on PU if present.
Feature	Added support in ApplyVarCal to use both SNP and INDEL models in a single command line.
Feature	Added additional error checks and error reporting.
Bug-fix	Solved issue in TNscope® SV that would cause an error if the input BAM file contained realigned reads with no matching bases.
Bug-fix	Solved issue that would cause a crash when using a BAM file containing reads with invalid mate tid.
Bug-fix	Solved issue in GVCFtyper that could cause a job to incorrectly think there are not enough licenses available when the communication to the license server is slow.
Bug-fix	Solved issue in TNscope® that failed to detect long INDELs under rare circumstances.
Bug-fix	Solved issue in ApplyVarCal that caused the output VCF file to miss the Sentieon® CommandLine in the header.
Bug-fix	Solved issue in TNscope® that prevented modifying the max_normalAlt_active option, which could cause loss of variants in areas of extreme depth.
Bug-fix	Solved issue in GCBias that would report results for a RG that did not contain any reads.
Bug-fix	Solved issue in util that would cause a crash when the BAM contained reads whose tid is out of bound.
Bug-fix	Solved issue in util stream that would not include the full path of util binary in the PG line of the BAM header.
Bug-fix	Solved issue in AlignmentStat that would produce an inaccurate PCT_ADAPTER value when the adapter_seq is not null.
Bug-fix	Solved issue in ApplyVarCal that would produce incorrect results when using sensitivity 0.
Bug-fix	Solved issue in the help of TNhaplotyper that would misrepresent the default pcr_indel_model.
Bug-fix	Solved issue in BWA that prevented it from working on certain older AMD cpus.
Bug-fix	Solved issue in InsertSizeMetrics that would generate a header using space instead of tab.

24.1.36 Release 201704.03

Type	Description
Feature	Added support for using a HTTP proxy with authentication.
Bug-fix	Solved issue that would prevent recalibration tables from older releases from being applied when the BAM file RGs have a defined PU.
Bug-fix	Solved issue in Realigner that could cause an error under extremely rare circumstances when reads in the input file have inconsistent information.
Bug-fix	Solved issue in TNscope® that could incorrectly error out when the tumor_contamination_frac parameter was set to 0.
Bug-fix	Solved issue that was preventing using BAM files with Contigs larger than 300M.

24.1.37 Release 201704.02

Type	Description
Feature	Added support in QualCal to allow input of `--cycle_val_max` parameter.
Bug-fix	Solved issue in Realigner when input BAM file has wrong MC tag type.
Bug-fix	Solved issue in TNhaplotyper in `--detect_pon` that would produce a non-compliant VCF.
Bug-fix	Solved issue in TNscope® in `--given` mode that would not report records on areas of 0 depth coverage.
Bug-fix	Solved issue in CoverageMetrics that would report the wrong cumulative_coverage_counts, coverage.statistics when input BAM has multipe different readgroups.
Bug-fix	Solved issue in InsertSizeMetricAlgo, GCBias and CoverageMetrics that could report the wrong results for extremely deep (600x) samples.
Bug-fix	Solved issue in QualCal where cycle_val_max parameter was not opened to user.

24.1.38 Release 201704.01

Type	Description
Feature	Added additional checks on the input BAM files.
Bug-fix	Solved issue in Dedup that would cause the algorithm to hang when the BAM file has more than 4 billion reads (200x WGS with 150 BP reads).
Bug-fix	Solved issue in QualCal that was slowing down the calculation.
Bug-fix	Solved issue in TNscope® that could report the wrong reference allele for structural variants.
Bug-fix	Solved issue that could cause a job to hang under extremely rare circumstances.
Bug-fix	Solved issue that could cause extra tags in the BAM header to disappear.

24.1.39 Release 201704

Type	Description
Feature	Maintenance update of algorithms.
Feature	Added support for defining a temporary directory, instead of using PWD.
Feature	Added support for type 2 VCF index files.
Feature	Added additional error checks and error reporting.
Feature	Added support for SAC annotation and Allele Specific annotations.
Feature	Added support for additional read filtering: MapQualFilter and OverclippingFilter.
Feature	Added support for CollectVCMetrics.
Feature	Added support for Deduplication equivalent to using BAM files sorted by read name.
Bug-fix	Solved issue in license control that would prevent run when the SENTIEON_AUTH_DATA is of a specific length.
Bug-fix	Solved issue that would not properly parse bed files containing both space and tab delimited fields.
Bug-fix	Solved issue in TNscope® that would report the wrong REF allele for structural variants.
Bug-fix	Solved issue in CoverageMetrics that would produce the wrong summary mean coverage when not using a bed file.

24.1.40 Release 201611.03

Type	Description
Feature	Added support for more options for base quality score recalibration.
Bug-fix	Solved issue in GVCFtyper that could report an incorrect GT or PL when the input GVCFs had been processed with different bed files.
Bug-fix	Solved issue in Realign that caused reads close to the edge of the contig to be realigned beyond the contig boundary.
Bug-fix	Solved issue in Realign that caused a segmentation fault when using known sites VCF including symbolic variants.
Bug-fix	Added support for Realign to update NM and MD tags in the realigned reads.

24.1.41 Release 201611.02

Type	Description
Feature	Added support for calling given known variants to Genotyper, Haplotyper, and TNscope®.
Feature	Added support for outputting physical phasing information of variants in TNhaplotyper and TNscope®.
Feature	Added support for base quality correction in streaming mode in util.
Feature	Maintenance update of HsMetricAlgo.
Feature	Removed redundant nthr argument in VarCal.
Bug-fix	Solved issue in TNscope® causing excessive memory usage in the structural variant calling.
Bug-fix	Solved issue in TNscope® causing a segmentation fault when an output file was not set.
Bug-fix	Solved issue in QualCal in `--plot` mode that was incorrectly reporting the need for known sites.
Bug-fix	Solved incorrect description of filter low_t_alt_frac in TNscope®.

24.1.42 Release 201611.01

Type	Description
Feature	Removed TNscope® temporary files novo_hap.data and novo_sv.data.
Feature	Increased default license timeout from client request.
Bug-fix	Solved an issue causing jobs to hang at the end of the processing when there were issues in the network communication.
Bug-fix	Solved an issue in TNscope® causing an assertion error when pairend reads have inconsistent flags.
Bug-fix	Solved an issue in TNscope® that incorrectly considered reads marked as duplicates in the calculation.

24.1.43 Release 201611

Type	Description
Feature	Added `--emit_mode all` in GVCFtyper.
Feature	Updates to β-version of TNscope®.
Feature	Speed improvement to the alignment and sorting tools.
Feature	Added ContaminationAssessment algorithm for contamination estimation.
Feature	Added detection of truncated VCF input and corrupted cram input files.
Bug-fix	Solved an issue in VarCal that was not producing a plot file when the recalibration failed.
Bug-fix	Solved an issue in VarCal that could cause a segmentation fault when there were not enough licenses for the requested threads.
Bug-fix	Solved an issue in util sort that was causing a segmentation fault when converting an empty sam file to BAM file.

24.1.44 Release 201608.01

Type	Description
Bug-fix	Solved issue in TNseq in tumor-only mode that added to the output VCF an empty column for non-existent NORMAL sample.

24.1.45 Release 201608

Type	Description
Feature	Introduced β-version of new product TNscope® for Tumor-Normal somatic variant calling and structural variant calling.
Feature	Reduced peak memory utilization in klib for alignment.
Feature	Speed improvement in the input file loading of GVCFtyper.
Feature	Speed improvement in the Haplotyper algorithm.
Feature	Made VarCal more robust when building the VQSR gaussian models.
Bug-fix	Cleared up certain error messages to make them more user friendly.
Bug-fix	Solved issue in Genotyper that caused a segmentation fault when using var_type BOTH.

24.1.46 Release 201606.02

Type	Description
Feature	Reduced memory utilization in GVCFtyper to reduce requirements for analysis of large (4000+) cohorts.
Bug-fix	Solved issue in TNsnv that created VCF files non-conforming to the standard.

24.1.47 Release 201606.01

Type	Description
Bug-fix	Solved an issue that prevented licsrvr from serving licenses when the user group list is longer than 1000 characters.

24.1.48 Release 201606

Type	Description
Feature	Maintenance update of TNhaplotyper algorithm (still in β).
Feature	Added support for RNAseq variant calling.
Feature	Added online help.
Feature	Added support for interval padding.
Bug-fix	Solved issue that produced no output in TNhaplotyper when using a PoN or no normal sample data.
Bug-fix	Solved issue preventing BED file with a header from being used.

24.1.49 Release 201603.03

Type	Description
Feature	Added method for inputting long list of VCF files to GVCFtyper.
Feature	Added command line to VCF header.
Feature	Added support for additional annotations on all variant callers.
Bug-fix	Solved issue producing an assertion error when too few variants are called.
Bug-fix	Solved issue reporting the wrong assembly in the VCF header when the FASTA file cannot be interpreted correctly.

24.1.50 Release 201603.02

Type	Description
Bug-fix	Solved issue in TNhaplotyper that incorrectly filtered INDELs.
Bug-fix	Solved issue in TNsnv and TNhaplotyper that made the dbsnp a required argument.
Bug-fix	Solved issue that creates non-conforming VCF files when calling variants in locations where the REF base is M.

24.1.51 Release 201603.01

Type	Description
Feature	Speed improvement in the VQSR algorithm.
Bug-fix	Solved issue that caused VQSR to apply to the wrong type of variants when the variants had a prior LOD annotation.
Bug-fix	Solved issue that slowed down dedup optical duplicate calculation when the number of duplicates at one locus is too large.

24.1.52 Release 201603

Type	Description
Feature	Maintenance update of algorithms.
Feature	Added TNsnv and TNhaplotyper (in β) algorithms for Tumor-Normal and Tumor only somatic variant calling.
Feature	Added support for CRAM files.
Feature	Added support for Haploid and Polyploid samples.
Feature	Added support for setting custom GQ bands for GVCF output.
Bug-fix	Solved issue that dropped reads when the reads extended beyond the contig boundary.

24.1.53 Release 201601.01

Type	Description
Bug-fix	Solved issue in GVCFtyper that produced the wrong variant quality on sites with AAF 0.5 when using more than 50 samples.

24.1.54 Release 201601

Type	Description
Feature	Added support for using interval files only on the realign target creation portion of Realigner.
Bug-fix	Solved issue preventing license server from running in systems with an Infiniband interface.
Bug-fix	Solved issue when merging BAM files that marks reads unmapped when their mate is unmapped.
Bug-fix	Solved issue in Haplotyper in GVCF mode when intervals are used that causes missed variants when the first interval of a contig contains no variants
Bug-fix	Solved issue in GVCFtyper to remove stray/empty SB annotations from the output VCF
Bug-fix	Solved issue causing the corruption of the output VCF under rare conditions involving large number of samples.
Bug-fix	Solved issue that prevented using BAM index files without the optional metadata pseudo bin.

24.1.55 Release 201511.01

Type	Description
Bug-fix	Solved issue in QualCal algorithm that slowed down execution when using BAM files with large number of Readgroups.
Bug-fix	Solved issue in ReadWriter algorithm that produced an incorrect BAM file when using a recalibration table and an input BAM that had PGZ auxiliary data.
Bug-fix	Solved issue in VQSR algorithm that caused the execution to hang when using the results from joint calling of more than 20 samples.

24.1.56 Release 201511

Type	Description
Feature	Maintenance update of algorithms.
Feature	Added pcr_indel_model flag to Haplotyper tool.
Feature	Added phasing information to the output of Haplotyper.
Feature	Added depth metrics tool.
Feature	Added support for compressed VCF/GVCF input and output.
Feature	Added support for Picard style interval files.
Feature	Added wrapper script to remove requirement of LD_LIBRARY_PATH environmental library.
Bug-fix	Added FILTER definition to the VQSR output file header.
Bug-fix	Solved issue that would cause an Out of Memory error in BQSR when using interval files.
Bug-fix	Solved issue that would cause a crash in IndelRealigner when the BAM file has reads with supplementary alignment.

24.1.57 Release 201509

Type	Description
Feature	Reduced resource requirements, including reducing the required maximum number of open files in the system.
Feature	Various improvements in error reporting.
Bug-fix	Solved issue that prevented using multiple BAM files as input to algorithms that produced another BAM file.
Bug-fix	Solved issue that prevented running BQSR when using Readgroups containing spaces.
Bug-fix	Solved issue in Haplotyper that could cause an incorrect variant quality score calculation in low coverage regions. The change could be up to 2% in the quality score, for 1 in 100000 variants.

24.1.58 Release 201508.01

Type	Description
Bug-fix	Solved issue in Realign algorithm that may cause an error when using known sites

24.1.59 Release 201508

Type	Description
Feature	Enhanced error reporting and error handling when dealing with inconsistent input data
Feature	Packaged BWA 0.7.12, and removed non-official options for BWA interface. The results of the packaged BWA are identical to the official BWA
Feature	Grouped the temporary files for VQSR plotting into a single file
Feature	New license server for large clusters
Bug-fix	Solved error in Haplotype Caller when using GVCF emit mode for joint calling
Bug-fix	Solved error in joint calling GVCFtyper that prevented using more than 20 samples

24.1.60 Release 201506

Type	Description
Feature	Added support for Hybrid Selection Analysis metrics
Feature	Unified Genotyper default behavior is to call only SNP variants, to be consistent with GATK 3.3 default behavior
α-Feature	Added initial support for joint variant calling of multiple samples that have ben previously processed individually
Bug-fix	Metrics properly reports unmapped reads when both mates in the pair are unmapped

24.1.61 Release 201505.02

Type	Description
Feature	Speed improvement in Alignment stage
Feature	Speed improvement in Dedup stage
Feature	Added support for temporary BAM files

24.1.62 Release 201505.01

Type	Description
Bug-fix	Issue running VQSR in Mac platform

24.1.63 Release 201505

Type	Description
Feature	Speed improvement via AVX optimization
Feature	Standarized option naming convention
Feature	tmp folder moved to the same location as the job
Feature	Added validation checks to input files
Bug-fix	Solved run to run differences in BQSR when reads have quality scores of 0.
Bug-fix	BQSR applied twice when running variant calling using a recaled BAM file.

24.2 Usage changes from previous release

24.2.1 Release 202503.02

This released introduced the following changes in interface and usage:

- Added new pgutil tool.
- `-se_tag` option is added in the InsertSizeMetricAlgo to includes single-end consensus reads in the insert size metrics calculation.

24.2.2 Release 202010.04

There are no changes in interface for this release.

24.2.3 Release 202010.03

This released introduced the following changes in interface and usage:

- Added new minimap2 tool.
- Added new VariantPhaser algorithm for the PacBio® pipeline.

24.2.4 Release 202010.02

This released introduced the following changes in interface and usage:

- Added new STAR tool.
- Added support in util sort for `--trim_primer` option to mark primers as soft or hard clips for amplicon samples.

24.2.5 Release 202010.01

This released introduced the following changes in interface and usage:

- Added support in TNhaplotyper2 for `--given` option.
- Added support in TNscope® for `--trim_primer` option to trim primers for amplicon samples.

24.2.6 Release 202010

This released introduced the following changes in interface and usage:

- Added 3 new algorithms (ContaminationModel, OrientationBias, and TNfilter) and updated TNhaplotyper2 for TNseq pipelines; in addition, deprecated the tnhapfilter tool, replaced by TNfilter.
- Added new option `--umi_post_process` to util sort for umi pipelines.
- Modified the way the tools deal with empty BED interval files: using an empty interval bed file will result in no processing done, as if the interval had 0 length.
- The tools now support using VCF files as interval files.
- Removed support for the –interval option in the Dedup algo to avoid unintentionally dropping reads from the output file.

24.2.7 Release 201911.01

This released introduced the following changes in interface and usage:

- Added suppport in `umi extract` for 3 input FASTQ files.

24.2.8 Release 201911

This released introduced the following changes in interface and usage:

- Added `umi extract` and `umi consensus` tools for processing reads containing UMI sequences.
- Added `--output_flag_filter` to ReadWriter to filter reads based on their flags. This option replaces the `--read_flag_mask` RedWriter option, which has been deprecated.

24.2.9 Release 201808.08

There are no changes in interface for this release.

24.2.10 Release 201808.07

There are no changes in interface for this release.

24.2.11 Release 201808.06

There are no changes in interface for this release.

24.2.12 Release 201808.05

This released introduced the following changes in interface and usage:

- Added `--genotype_model` option in Genotyper, Haplotyer and GVCFtyper algorithm to support new multinomial genotyping model.

24.2.13 Release 201808.03

There are no changes in interface for this release.

24.2.14 Release 201808.02

This released introduced the following changes in interface and usage:

- Added TNscope® option *–disable_detector* to control the type of variants called.
- Added support for using multiple *–interval* options.

24.2.15 Release 201808.01

This released introduced the following changes in interface and usage:

- The binaries licsrvr and licclnt require using the bin/sentieon wrapper to be expecuted.

24.2.16 Release 201808

This released introduced the following changes in interface and usage:

- Added `--bam_output` option to TNscope® and TNhaplotyper to output a BAM file including local reassembly of the reads.
- Added support for running BWA through the sentieon command wrapper.
- Modified plot interface to be consistent with other tools.
- Added support for CRAM v3 files.
- Added options `--snp_heterozygosity` and `--indel_heterozygosity` to Haplotyper and GVCFtyper to input the expected heterozygosity value used to compute prior likelihoods.
- Added options `--keep_oq` and `--use_oq` to QualCalFilter to keep and use the original qualities when performing BQSR.
- Added option `--read_flag_mask` to ReadWriter to perform customized read filtering.
- Added option `--replace_rg` to driver to modify the RG information of an input BAM file.
- Modified the flow for germline variant calling with DNAscope and added DNAModelApply algorithm.

24.2.17 Release 201711.05

There are no changes in interface for this release.

24.2.18 Release 201711.04

There are no changes in interface for this release.

24.2.19 Release 201711.03

This released introduced the following changes in interface and usage:

- Introduced multi threaded capability to BWA shm, so that BWA shm accepts the -t NUMBER_THREADS option.
- Introduced new environmental variable bwt_max_mem to limit the memory usage of BWA at the expense of speed performance.

24.2.20 Release 201711.02

This released introduced the following changes in interface and usage:

- Introduced new QC metrics algorithms: BaseDistributionByCycle, QualityYield, WgsMetricsAlgo, SequenceArtifactMetricsAlgo.
- Modified the default for option filter_chimeric_reads in DNAscope when var_type is NOT bnd.

24.2.21 Release 201711.01

This released introduced the following changes in interface and usage:

- Added options `--version` and ``--dump` to licsrvr binary.

24.2.22 Release 201711

This released introduced the following changes in interface and usage:

- Added new DNAscope algorithm for germline variant calling and structural variant calling.
- Added new TNModelApply algorithm for applying a Machine Learning model to help with variant filtration in TNscope®.
- Added new `--bam_output` option to Haplotyper to output a BAM file including local reassembly of the reads.
- Added new option `--vqsr_model` to ApplyVarCal to allow application of both SNP and INDEL models in a single command line.

24.2.23 Release 201704.03

There are no changes in interface for this release.

24.2.24 Release 201704.02

This released introduced the following changes in interface and usage:

- Added `--cycle_val_max` parameter in QualCal algorithm.
- Added `--cram_write_options` in Dedup, Realigner, ReadWriter, and RNASplitReadsAtJunction algorithms when output format is CRAM.
- Removed `--min_iter` in VarCal algorithm.

24.2.25 Release 201704.01

There are no changes in interface for this release.

24.2.26 Release 201704

This released introduced the following changes in interface and usage:

- Added new driver option `--temp_dir` option to override where the temporary files will be stored.
- Added extra annotations for the `--annotation` option: SAC, AS_BaseQRankSum, AS_FS, AS_InbreedingCoeff, AS_MQRankSum, AS_QD, AS_MQ, AS_ReadPosRankSum, AS_SOR
- Added new read filter driver options `--read_filter MapQualFilter` to filter the input BAM reads by mapping Quality and `--read_filter OverclippingFilter` to filter the input BAM reads according to their soft clipping characteristics.
- Added CollectVCMetrics algorithm to calculate metrics on the VCF output from variant calling.
- Added new Dedup options `--output_dup_read_name` and `--dup_read_name` to perform dedup to mark both primary and non-primary reads.
- Added new options for TNhaplotyper
- Added a new ContaminationAssessment option `--population` to specify the population to represent the baseline allele fraction.

24.2.27 Release 201611.03

This released introduced the following changes in interface and usage:

- Added new driver option `--read_filter QualCalFilter` to perform base quality score recalibration with additional options.

24.2.28 Release 201611.02

This released introduced the following changes in interface and usage:

- Introduced `--given` option in Genotyper, Haplotyper, and TNscope®, to perform calling at given variant sites.
- Introduced stream mode in util to perform base quality correction in streaming mode.
- Removed `--nthr` argument from VarCal.
- Added extra options to HsMetricAlgo.

24.2.29 Release 201611.01

There are no changes in interface for this release.

24.2.30 Release 201611

This released introduced the following changes in interface and usage:

- Introduced ContaminationAssessment algorithm to estimate the contamination in the tumor sample based on the normal variants called.
- Added `--emit_mode all` option in GVCFtyper.

24.2.31 Release 201608.01

There are no changes in interface for this release.

24.2.32 Release 201608

This released introduced the following changes in interface and usage:

- Introduced TNscope® algorithm as part of the new TNscope® product for Tumor-Normal somatic variant calling and structural variant calling.

24.2.33 Release 201606.02

There are no changes in interface for this release.

24.2.34 Release 201606.01

There are no changes in interface for this release.

24.2.35 Release 201606

This released introduced the following changes in interface and usage:

- Added RNA variant calling tool `--algo SplitReadsAtJunction` and Haplotyper option `--trim_soft_clip` to process RNA data aligned using STAR.
- Added `--help` driver option to driver and algorithms to show online help.
- Added `--interval_padding` driver option to pad the input intervals.

24.2.36 Release 201603.03

This released introduced the following changes in interface and usage:

- Added `--annotation` option to Haplotyper and Genotyper to output additional annotations to the result VCF.
- Added new method of inputting GVCF files to GVCFtyper.

24.2.37 Release 201603.02

There are no changes in interface for this release.

24.2.38 Release 201603.01

There are no changes in interface for this release.

24.2.39 Release 201603

This released introduced the following changes in interface and usage:

- Added TNsnv and TNhaplotyper algorithm for Tumor-Normal and Tumor only somatic variant calling.
- Haplotyper and Genotyper have a new optional argument `--ploidy` to indicate the ploidicity of the sample being processed.
- All locations where you can input and output BAM files now accept CRAM files.

24.2.40 Release 201601.01

There are no changes in interface for this release.

24.2.41 Release 201601

This released introduced the following changes in interface and usage:

- Realigner has a new optional argument `--interval_list` to use interval file on the target interval portion of the algorithm, to be in line with Broad's Best Practice recommendations. Using option `--interval` in the driver call for realigner would apply the interval to both target creation and realignment, dropping reads in the output BAM file.

24.2.42 Release 201511.01

There are no changes in interface for this release.

24.2.43 Release 201511

This released introduced the following changes in interface and usage:

- Haplotyper has a new optional argument `--pcr_indel_model` to determine the indel model used to weed out false positive indels.
- The single interval definition for the `--interval` driver option is now first-base-1 based as opposed to first-base-0 based. The `--interval` driver option supports Picard style interval lists.
- The driver and util binaries are now accessible through a wrapper script that takes care of setting the LD_LIBRARY_PATH environmental library.
- Output VCF files can be produced compressed by using .vcf.gz or .g.vcf.gz extensions.

24.2.44 Release 201509

This release introduced the following changes in interface and usage:

- It is no longer necessary to setup a large limit of open files.
- In the Remove duplicates stage, the Dedup command now creates an output file reporting the results of the de-duplication.
- Added option `--bam_compression` to all algorithms producing an output BAM file (Dedup, Realign and ReadWriter). The option controls the output BAM file compression level and can be used to achieve a faster speed at the expense of file size.

24.2.45 Release 201508.01

There are no changes in interface for this release.

24.2.46 Release 201508

This release introduced the following changes in interface and usage:

- The BWA mapping binary no longer requires or accepts the option -f.
- The procedure to generate the plots for base quality score recalibration (BQSR) is now composed of 3 commands instead of 2. Option `--pre` has been replaced by option `--before`, and a new option `--plot` is used to generate the plotting data.
- VQSR VarCal algorithm now produces a single file containing all data required to create the VQSR report; this file is specified via the new option `--plot_file`.

24.2.47 Release 201506

There are no changes in interface for this release.

24.2.48 Release 201505.02

This released introduced the following changes in interface and usage:

- In the Alignment stage, the samtools command to convert the output of BWA from SAM to BAM is no longer required, as it is now done in the util command via a new option `--sam2bam`.
- In QualCal algorithm there is a new option `--pre` RECAL_DATA.TABLE to calculate the data required to create the report.

24.2.49 Release 201505.01

There are no changes in interface for this release.

24.2.50 Release 201505

This release introduced the following changes in interface and usage:

- In the GCBias algorithm, the option -s has been replaced by `--summary`.
- In the LocusCollector algorithm, the option -f has been replaced by `--fun`.
- In the Remove duplicates stage, the option -s has been replaced by `--score_info`.
- In the Remove duplicates stage, the option -r has been replaced by `--rmdup`.
- In the BQSR stage, it is no longer necessary to run the varplot command to calculate the data required to create the BQSR report; this call has been merged with the call to apply the recalibration. New option `--csv` is now used.

25

Acknowledgements

Portions of the SENTIEON SOFTWARE may use the following copyrighted material. We acknowledge these developers for their contributions.

```
FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT
SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE
FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE,
ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
DEALINGS IN THE SOFTWARE.

* ====================================================================
* Copyright (c) 1998-2011 The OpenSSL Project. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 * notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in
 * the documentation and/or other materials provided with the
 * distribution.
 *
 * 3. All advertising materials mentioning features or use of this
 * software must display the following acknowledgment:
 * "This product includes software developed by the OpenSSL Project
 * for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
 *
 * 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
 * endorse or promote products derived from this software without
 * prior written permission. For written permission, please contact
 * openssl-core@openssl.org.
 *
 * 5. Products derived from this software may not be called "OpenSSL"
 * nor may "OpenSSL" appear in their names without prior written
 * permission of the OpenSSL Project.
 *
 * 6. Redistributions of any form whatsoever must retain the following
 * acknowledgment:
 * "This product includes software developed by the OpenSSL Project
 * for use in the OpenSSL Toolkit (http://www.openssl.org/)"
 *
 * THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ''AS IS'' AND ANY
 * EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR
 * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
 * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
 * LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
 * STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
 * OF THE POSSIBILITY OF SUCH DAMAGE.
 * ====================================================================
 *
```

```
 * This product includes cryptographic software written by Eric Young
 * (eay@cryptsoft.com). This product includes software written by Tim
 * Hudson (tjh@cryptsoft.com).
 *
 /

/ Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
 * All rights reserved.
 *
 * This package is an SSL implementation written
 * by Eric Young (eay@cryptsoft.com).
 * The implementation was written so as to conform with Netscapes SSL.
 *
 * This library is free for commercial and non-commercial use as long as
 * the following conditions are aheared to. The following conditions
 * apply to all code found in this distribution, be it the RC4, RSA,
 * lhash, DES, etc., code; not just the SSL code. The SSL documentation
 * included with this distribution is covered by the same copyright terms
 * except that the holder is Tim Hudson (tjh@cryptsoft.com).
 *
 * Copyright remains Eric Young's, and as such any Copyright notices in
 * the code are not to be removed.
 * If this package is used in a product, Eric Young should be given attribution
 * as the author of the parts of the library used.
 * This can be in the form of a textual message at program startup or
 * in documentation (online or textual) provided with the package.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the copyright
 * notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 * 3. All advertising materials mentioning features or use of this software
 * must display the following acknowledgement:
 * "This product includes cryptographic software written by
 * Eric Young (eay@cryptsoft.com)"
 * The word 'cryptographic' can be left out if the rouines from the library
 * being used are not cryptographic related :-).
 * 4. If you include any Windows specific code (or a derivative thereof) from
 * the apps directory (application code) you must include an acknowledgement:
 * "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
 *
 * THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ''AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
```

```
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 * The licence and distribution terms for any publically available version or
 * derivative of this code cannot be changed. i.e. this code cannot simply be
 * copied and put under another distribution licence
 * [including the GNU Public Licence.]
 /


Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction, and
   distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by the
   copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all other
   entities that control, are controlled by, or are under common control with
   that entity.  For the purposes of this definition, "control" means (i) the
   power, direct or indirect, to cause the direction or management of such
   entity, whether by contract or otherwise, or (ii) ownership of fifty percent
   (50%) or more of the outstanding shares, or (iii) beneficial ownership of
   such entity.

   "You" (or "Your") shall mean an individual or Legal Entity exercising
   permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications,
   including but not limited to software source code, documentation source, and
   configuration files.

   "Object" form shall mean any form resulting from mechanical transformation
   or translation of a Source form, including but not limited to compiled
   object code, generated documentation, and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or Object form,
   made available under the License, as indicated by a copyright notice that is
   included in or attached to the work (an example is provided in the Appendix
   below).

   "Derivative Works" shall mean any work, whether in Source or Object form,
   that is based on (or derived from) the Work and for which the editorial
   revisions, annotations, elaborations, or other modifications represent, as a
   whole, an original work of authorship. For the purposes of this License,
```

Derivative Works shall not include works that remain separable from, or
merely link (or bind by name) to the interfaces of, the Work and Derivative
Works thereof.

"Contribution" shall mean any work of authorship, including the original
version of the Work and any modifications or additions to that Work or
Derivative Works thereof, that is intentionally submitted to Licensor for
inclusion in the Work by the copyright owner or by an individual or Legal
Entity authorized to submit on behalf of the copyright owner. For the
purposes of this definition, "submitted" means any form of electronic,
verbal, or written communication sent to the Licensor or its
representatives, including but not limited to communication on electronic
mailing lists, source code control systems, and issue tracking systems that
are managed by, or on behalf of, the Licensor for the purpose of discussing
and improving the Work, but excluding communication that is conspicuously
marked or otherwise designated in writing by the copyright owner as "Not a
Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on
behalf of whom a Contribution has been received by Licensor and subsequently
incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this
   License, each Contributor hereby grants to You a perpetual, worldwide,
   non-exclusive, no-charge, royalty-free, irrevocable copyright license to
   reproduce, prepare Derivative Works of, publicly display, publicly perform,
   sublicense, and distribute the Work and such Derivative Works in Source or
   Object form.

3. Grant of Patent License. Subject to the terms and conditions of this
   License, each Contributor hereby grants to You a perpetual, worldwide,
   non-exclusive, no-charge, royalty-free, irrevocable (except as stated in
   this section) patent license to make, have made, use, offer to sell, sell,
   import, and otherwise transfer the Work, where such license applies only to
   those patent claims licensable by such Contributor that are necessarily
   infringed by their Contribution(s) alone or by combination of their
   Contribution(s) with the Work to which such Contribution(s) was submitted.
   If You institute patent litigation against any entity (including a
   cross-claim or counterclaim in a lawsuit) alleging that the Work or a
   Contribution incorporated within the Work constitutes direct or contributory
   patent infringement, then any patent licenses granted to You under this
   License for that Work shall terminate as of the date such litigation is
   filed.

4. Redistribution. You may reproduce and distribute copies of the Work or
   Derivative Works thereof in any medium, with or without modifications, and
   in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a
       copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating
       that You changed the files; and

(c) You must retain,in the Source form of any Derivative Works that You
distribute, all copyright, patent, trademark, and attribution notices
from the Source form of the Work, excluding those notices that do not
pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution,
then any Derivative Works that You distribute must include a readable
copy of the attribution notices contained within such NOTICE file,
excluding those notices that do not pertain to any part of the
Derivative Works, in at least one of the following places: within a
NOTICE text file distributed as part of the Derivative Works; within the
Source form or documentation, if provided along with the Derivative
Works; or, within a display generated by the Derivative Works, if and
wherever such third-party notices normally appear. The contents of the
NOTICE file are for informational purposes only and do not modify the
License. You may add Your own attribution notices within Derivative
Works that You distribute, alongside or as an addendum to the NOTICE
text from the Work, provided that such additional attribution notices
cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may
provide additional or different license terms and conditions for use,
reproduction, or distribution of Your modifications, or for any such
Derivative Works as a whole, provided Your use, reproduction, and
distribution of the Work otherwise complies with the conditions stated in
this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any
Contribution intentionally submitted for inclusion in the Work by You to the
Licensor shall be under the terms and conditions of this License, without
any additional terms or conditions. Notwithstanding the above, nothing
herein shall supersede or modify the terms of any separate license agreement
you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names,
trademarks, service marks, or product names of the Licensor, except as
required for reasonable and customary use in describing the origin of the
Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in
writing, Licensor provides the Work (and each Contributor provides its
Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
KIND, either express or implied, including, without limitation, any
warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or
FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining
the appropriateness of using or redistributing the Work and assume any risks
associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in
tort (including negligence), contract, or otherwise, unless required by
applicable law (such as deliberate and grossly negligent acts) or agreed to
in writing, shall any Contributor be liable to You for damages, including
any direct, indirect, special, incidental, or consequential damages of any
character arising as a result of this License or out of the use or inability

to use the Work (including but not limited to damages for loss of goodwill,
work stoppage, computer failure or malfunction, or any and all other
commercial damages or losses), even if such Contributor has been advised of
the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or
Derivative Works thereof, You may choose to offer, and charge a fee for,
acceptance of support, warranty, indemnity, or other liability obligations
and/or rights consistent with this License. However, in accepting such
obligations, You may act only on Your own behalf and on Your sole
responsibility, not on behalf of any other Contributor, and only if You
agree to indemnify, defend, and hold each Contributor harmless for any
liability incurred by, or claims asserted against, such Contributor by
reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

/***
The MIT License

Copyright (c) 2014 Intel Corporation

Permission is hereby granted, free of charge, to any person
obtaining a copy of this software and associated documentation
files (the "Software"), to deal in the Software without
restriction, including without limitation the rights to use,
copy, modify, merge, publish, distribute, sublicense, and/or
sell copies of the Software, and to permit persons to whom the
Software is furnished to do so, subject to the following
conditions:

The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the
Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR
OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
***/

Copyright (c) 2013-2015 Niels Lohmann.

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in
all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

The MIT License (MIT)

Copyright (c) Microsoft Corporation

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

Copyright © David Elworthy 2004.
All rights reserved.

Redistribution and use in source and binary forms for any purpose, with or
without modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright notice,
   this list of conditions, and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions, and the disclaimer that follows these conditions
   in the documentation and/or other materials provided with the distribution.

26

Acronyms and Abbreviations

GATK
Genome Analysis Tool Kit

VCF
Variant Call Format

SAM
Sequence Alignment/Map

BAM
Binary compressed SAM

BCF
Binary compressed variant Call Format

BED
Browser Extensible Display

indels
INsertion-DELetions

27

DISCLAIMER OF WARRANTIES AND LIMITATION OF LIABILITIES

SENTIEON, INC. RESERVES ALL RIGHTS IN THE PROGRAM AS DELIVERED. THE PROGRAM OR ANY PORTION THEREOF MAY NOT BE REPRODUCED IN ANY FORM WHATSOEVER EXCEPT AS PROVIDED BY LICENSE, WITHOUT THE CONSENT OF SENTIEON.

THIS NOTICE MAY NOT BE REMOVED FROM THE PROGRAM.

NEITHER SENTIEON, ANY MEMBER OF SENTIEON, THE ORGANIZATION(S) BELOW, NOR ANY PERSON ACTING ON BEHALF OF ANY OF THEM:

1. MAKES ANY WARRANTY OR REPRESENTATION WHATSOEVER, EXPRESS OR IMPLIED, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS OF ANY PURPOSE WITH RESPECT TO THE PROGRAM; OR
2. ASSUMES ANY LIABILITY WHATSOEVER WITH RESPECT TO ANY USE OF THE PROGRAM OR ANY PORTION THEREOF OR WITH RESPECT TO ANY DAMAGES WHICH MAY RESULT FROM SUCH USE.

TO THE MAXIMUM EXTENT PERMITTED BY LAW AND EXCEPT AS EXPRESSLY WARRANTED HEREIN, THE SENTIEON SOFTWARE, DOCUMENTATION, AND OTHER PRODUCTS AND SERVICES PROVIDED BY SENTIEON HEREUNDER ARE PROVIDED AS-IS WITHOUT WARRANTIES OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO WARRANTIES OF TITLE OR IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE. SENTIEON AND ITS LICENSORS DO NOT GUARANTEE THAT THE SENTIEON SOFTWARE, DOCUMENTATION OR SERVICES WILL MEET LICENSEE'S REQUIREMENTS, BE ERROR-FREE, OR OPERATE WITHOUT INTERRUPTION.

160 E Tasman Dr STE 208, San Jose, CA 95134-1619
www.sentieon.com