MGI 华大智造

To Enable Effective and Affordable Healthcare Solutions for All

MegaBOLT Bioinformatics Analysis Accelerator

User Manual

Address: Main Building and Second floor of No.11 Building, Beishan Industrial Zone, Yantian District, Shenzhen, 518083, Guangdong, China

E-mail: MGI-service@genomics.cn

Website: www.mgitech.cn

Research Use Only

MGI Tech Co., Ltd.

Edition 1.0

About the user manual

This user manual is applicable to MegaBOLT Bioinformatics Analysis Accelerator (MegaBOLT_scheduler). The edition is 1.0 and the software version is V2.1.0.

Figures in this manual are all illustrations. The contents might be slightly different from the software, please refer to the software purchased.

Release date: December 20, 2019

Manufacturer information

Manufacturer	MGI Tech Co., Ltd.
Address	Main Building and Second floor of No.11 Building, Beishan Industrial Zone, Yantian District, Shenzhen, 518083, Guangdong, China
Technical support	MGI Tech Co., Ltd.
Technical support E-mail	MGI-service@genomics.cn

Revision history

Revision	Date
1.0	December 20, 2019

Contents

---This page is intentionally left blank.---

1 Introduction

MegaBOLT bioinformatics analysis accelerator (hereinafter called MegaBOLT) is a highly effective sequencing analysis system. MegaBOLT supports the analysis of Whole Genome Sequencing (WGS), Whole Exome Sequencing (WES), and Panel Sequencing on Germline or Somatic data; performs the calculation from fq.gz sequence file to vcf.gz variant output, including pre-processing data (QC step), and post-processing (statistical step) for BAM files and VCF files. MegaBOLT adopts FPGA heterogeneous computing and multi-stream system to accelerate analysis. It is 10X to 20X faster in massive data analysis compared with normal CPU computing, for example, GATK Best Practice.

The MegaBOLT Rack Server mode is a rack server equipped with the MegaBOLT analysis system for large-scale data analysis scenarios in a cluster environment. It is flexible because it supports a variety of analysis processes, and it is suitable for users with background of bioinformatics analysis.

The MegaBOLT Workstation mode is a workstation equipped with the MegaBOLT analysis system for small to medium data analysis scenarios. It provides one-stop services from sequencing to WGS/WES analysis. It provides an interactive web interface and analysis report, which is easy to operate and suitable for most users with non-biological information analysis backgrounds.

2 System description

Running mode

Item	Description
WGS	Whole Genome Sequencing data analysis
WES	Whole Exome Sequencing data analysis

Pipeline options

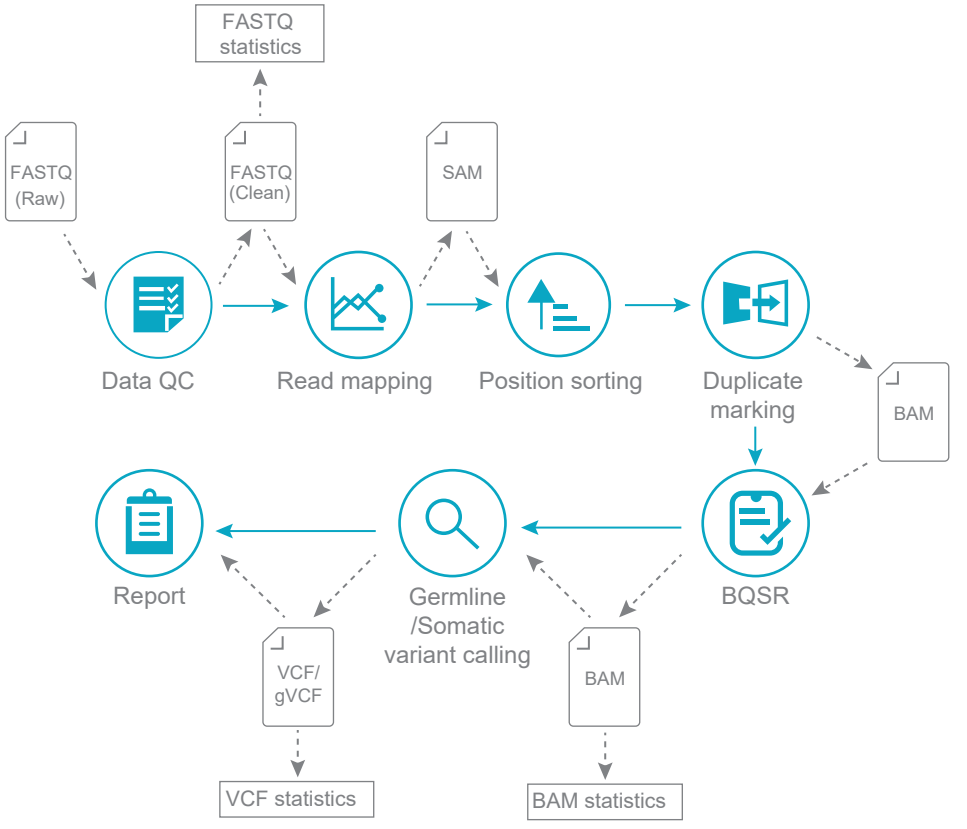

Figure 1 Flow chart

NOTE Items marked with an asterisk (*) is an optional pipeline or input/output.

Pipeline		Description
Basic pipeline (default)	Germline variant calling basic pipeline	Read mapping > position sorting > duplicate marking* > BQSR* > Germline variant calling. The input is clean FASTQ file. The output is BAM file* from BQSR and VCF/gVCF file from variant calling.
full	Germline variant calling full pipeline	Data QC > read mapping > position sorting > duplicate marking* > BQSR* > Germline variant calling > BAM/VCF statistics > analysis report. The input is raw FASTQ file. The output is clean FASTQ file*, BAM file* from BQSR, VCF/gVCF file from variant calling, FASTQ/BAM/VCF statistics, and finally analysis report generation.
somatic	Somatic variant calling full pipeline	Data QC* > read mapping > position sorting > duplicate marking* > BQSR* > Germline variant calling* > Somatic variant calling > BAM/VCF statistics* > analysis report. The input is the raw/clean FASTQ file for tumor and normal*. The output is clean FASTQ file*, BAM file* from BQSR, VCF/gVCF file* from Germline variant calling, VCF file from Somatic variant calling, FASTQ/BAM/VCF statistics*, and Germline variant calling analysis report.

MegaBOLT also supports more flexible pipelines. For details, refer to *Pipeline modes (--type) on Page 11*.

Configuration requirements

Item	Minimum configuration	Recommended configuration
CPU	2 × Intel Xeon E5-26XX Series	2 × Intel Xeon Gold 62XX Series
Memory	96 GB	128 GB
Storage	1 TB HDD	>2 TB SSD
OS	CentOS 7.3-7.5	CentOS 7.3-7.5
Network card	1 Gbps	10 Gbps

3 Quick start

Exemplary script

Input and execute *run.sh* as follows:

WGS Germline basic pipeline analysis:

```
MegaBOLT --type basic --runtype WGS --list sample.list
```

WGS Germline full pipeline analysis:

```
MegaBOLT --type full --runtype WGS --list sample.list
```

WGS Somatic pipeline analysis:

```
MegaBOLT --type somatic --runtype WGS --list sample.tumor.list
--list2 sample.normal.list
```

WES Germline basic pipeline analysis:

```
MegaBOLT --type basic --runtype WES --list sample.list --bed
BV4
```

WES Germline full pipeline analysis:

```
MegaBOLT --type full --runtype WES --list sample.list --bed
BV4
```

WES Somatic pipeline analysis:

```
MegaBOLT --type somatic --runtype WES --list sample.tumor.list
--list2 sample.normal.list --bed BV4
```

NOTE
- For information about other pipelines, refer to *Pipeline modes (--type) on Page 11*.
- For more examples, refer to *Use cases on Page 24*.

Description

The following takes WGS basic pipeline analysis for example:

```
MegaBOLT --type basic --runtype WGS --list sample.list
```

"MegaBOLT" is the name of the program, "--type basic" indicates that it is a basic pipeline, "--runtype WGS" indicates that the running mode is WGS, "--list sample.list" indicates that it is a sample list file.

The format of the list file for PE data is as follows:

```
SampleName Read1 Read2 Adaptor1 Adaptor2 RGID RGSM RGLB RGPL
```

For details, refer to *List file on Page 7*.

The script implies several default parameters, for example, the script does not specify a reference sequence (through --ref). Therefore, hg19.fa is used for analysis by default. "basic" and "WGS" are the default value of "--type" and "--runtype" respectively. Therefore, the script can be simplified as follows:

```
MegaBOLT --list sample.list
```

For details about the parameters, refer to *Parameter description on Page 16*, or access by executing "MegaBOLT -h".

4 List file

The file is a list of FASTQ files that support PE and SE data.

List file format for PE data

Format 1

```
SampleName    Read1    Read2
```

Description:

- “SampleName” refers to the sample name, “Read1” refers to Read1 FASTQ file path for PE data, “Read2” refers to Read2 FASTQ file path for PE data. The fields are separated by using a space or tab character.

  Example:

  ```
  sample    /data/example/read1.fq.gz    /data/example/
  read2.fq.gz
  ```

- The list file contains one or more samples. Each sample occupies a column. Comment lines that start with “#” are supported.

  Example:

  ```
  sample1    /data/example/read1_1.fq.gz    /data/
  example/read2_1.fq.gz
  sample2    /data/example/read1_2.fq.gz    /data/
  example/read2_2.fq.gz
  ```

- One sample can be related to multiple pairs of FASTQ files, with the name of the files separated by a comma “,” but no spaces.

  Example:

  ```
  sample    read1_1,read1_2,…,read1_N
  read2_1,read2_2,…,read2_N
  ```

Format 2

```
SampleName    Read1    Read2    Adaptor1    Adaptor2
```

Description:

- Adaptor1 and Adaptor2 are sequence adaptors for Read1 and Read2 respectively.
- Adaptor1 and Adaptor2 should appear in pair and only contain “A”, “T”, “G” , and “C”.
- One sample can only include one group of sequence adaptors.

Example:

```
sample    read1_1,read1_2    read2_1,read2_2    AAGTCGGA
AAGTCGGATC
```

Format 3

```
SampleName    Read1    Read2    RGID    RGSM    RGLB
RGPL
```

Description:

- “RGID”, “ RGSM”, “RGLB”, and “RGPL” refer to Read group ID, Read group sample name, Read group library name, and Read group sequencing platform.
- “RGID”, “ RGSM”, “RGLB”, and “RGPL” should appear in groups and their order cannot be changed.
- “RGSM” should be consistent with “SampleName”.
- “RGPL” only supports the following fields:

  MGISEQ, BGISEQ, ILLUMINA, SLX, SOLEXA, SOLID, 454, LS454, COMPLETE, PACBIO, IONTORRENT

- When a sample contains multiple pairs of FASTQ files, “RGID”, “ RGSM”, “RGLB”, and “RGPL” of each FASTQ file can be user-defined, but should be separated with a comma and ensure that the columns and read pairs are the same in quantity.

Example:

```
sample   read1_1,read1_2   read2_1,read2_2   id1,id2  sample
lb  COMPLETE
```

Format 4

```
SampleName   Read1   Read2   Adaptor1   Adaptor2   RGID
RGSM   RGLB   RGPL
```

Description:

- The sequence of “Adaptor1 Adaptor2” and “RGID RGSM RGLB RGPL” cannot be reversed.
- Other constraints are the same as format 1-3.

Example:

```
sample  read1  read2  AAGTCGGA  AAGTCGGATC  id  sample  lb
COMPLETE
```


Default value

Unspecified columns in the list file use the default value.

Format	Description
Adaptor1	AAGTCGGAGGCCAAGCGGTCTTAGGAAGACAA
Adaptor2	AAGTCGGATCGTAGCCATGTCGTTCTGTGAGCCAAGGAGTTG
RGID	rg
RGSM	SampleName in the list file.
RGLB	lb
RGPL	COMPLETE

List file format for SE data

NOTE Constraints of the list file format for SE data are the same as that for PE data.

Format 1

```
SampleName    Read
```

Format 2

```
SampleName    Read    Adaptor
```

Format 3

```
SampleName    Read    RGID    RGSM    RGLB    RGPL
```

Format 4

```
SampleName    Read    Adaptor    RGID    RGSM    RGLB
RGPL
```

5 Pipeline modes (--type)

MegaBOLT supports several pipeline modes, including single-module pipelines and combined-module pipelines. Pipeline modes can be selected by setting the parameter “--type”.

Combined-pipeline modes

MegaBOLT provides combined-pipeline modes that are commonly used in bioinformatics analysis, which are optimized by integrating the software and hardware. Compared with sequential execution of single-module pipeline modes, combined-pipeline modes can finish analysis tasks faster with the same results. Combined-pipeline modes are recommended.

The combined-pipeline modes are as follows:

Combined-pipeline modes		Description
basic (default pipeline)	Germline variant calling basic pipeline	Read mapping > position sorting > duplicate marking* > BQSR* > Germline variant calling The input is clean FASTQ file. The output is BAM file* from BQSR and VCF/gVCF files from variant calling.
full	Germline variant calling full pipeline	Data QC > read mapping > position sorting > duplicate marking* > BQSR* > Germline variant calling > BAM/VCF statistics > analysis report The input is raw FASTQ file. The output is clean FASTQ file*, BAM file* from BQSR, VCF/gVCF file from variant calling, FASTQ/BAM/VCF statistic results, and analysis reports.

Combined-pipeline modes		Description
somatic	Somatic variant calling pipeline	Data QC* > read mapping > position sorting > duplicate marking* > BQSR* > Germline variant calling* > Somatic variant calling > BAM/VCF statistics* > analysis report* The input is the raw/clean FASTQ file for tumor and normal* samples. The output is clean FASTQ file*, BAM file* from BQSR, VCF/gVCF file* from Germline variant calling, VCF/gVCF file from Somatic variant calling, FASTQ/BAM/VCF statistic results*, and analysis reports*. ● "Tumor + normal" mode and singular tumor mode are supported. ● Through parameter settings, you can choose whether to run data QC, duplicate marking, BQSR, Germline variant calling, BAM/VCF statistics, and generating analysis report. Germline variant calling analysis report is available only after running data QC, Germline variant calling, and BAM/VCF statistics. ● Set the tumor sample through "--list", and normal sample through "--list2".
alignmentsortmarkdup	Read mapping, position sorting, and duplicate marking	Run read mapping, position sorting, and duplicate marking* pipelines in order. The input is clean FASTQ file. The output is BAM files from position sorting and duplicate marking.
alignmentsortmarkdupbqsr	Read mapping, position sorting, duplicate marking, and BQSR.	Run read mapping, position sorting, duplicate marking*, and BQSR pipelines in order. The input is clean FASTQ file. The output is the BAM file from BQSR.

NOTE Items marked with an asterisk (*) is an optional pipeline or input/output.

To satisfy specific demands of the user, combined-pipeline modes can be customized through parameter settings:

- Select read mapping software from Minimap2 (default) and BWA.

- Select variant calling software from HaplotypeCaller 3.7 (default), HaplotypeCaller 4.0, and DeepVariant.
- Select whether to execute duplicate marking and BQSR pipelines through parameter settings.
- Select whether to output result files.

For details, refer to *Parameter description on Page 16*.

Single-mode pipeline

Single-mode pipeline		Description
qc	Raw FASTQ file QC	Finish filtering and statistics of the raw FASTQ file. The input is raw FASTQ file. The output is clean FASTQ file and statistics file.
alignment	Read mapping	Align the sequenced sequences to the reference genome. The input is clean FASTQ file. The output is clean FASTQ file and statistics file. NOTE - Support selecting alignment software: Minimap2 (default) or BWA. - The BWA is accelerated, and creates a large index under the reference genome directory.
sortmarkdup	Position sorting and duplicate marking	The aligned sequences are sorted by position on the genome, and the sequences aligned to the same position are marked as duplicates. The input is SAM/BAM file from read mapping. The output is BAM file from position sorting and duplicate marking. NOTE - Items marked with an asterisk (*) is an optional pipeline or input/output. - Support selecting whether to run duplicate marking through parameter settings.
bqsr	BQSR	Calibrate the sequencing quality values for result data. The input is BAM file. The output is BAM file from BQSR.

Single-mode pipeline		Description
haplotypecaller	Germline variant calling	Use the sequence and related information that have been aligned to the reference genome to identify and detect mutations. The input is BAM file. The output is VCF/gVCF file after variant calling. NOTE Support selecting variant calling software: GATK HaplotypeCaller 3.7 (default), GATK HaplotypeCaller 4.0 or DeepVariant.
mutect2	Somatic variant calling	The sequence and related information that have been aligned to the reference genome are used to identify and detect somatic mutations. The input is BAM file. The output is the VCF file for variant calling results.
genotypegvcfs	Joint genotyping	Run GenotypeGVCFs for the gVCF file from variant calling. The input is gVCF file. The output is VCF file from GenotypeGVCFs.
builddict	Build dict index	Build the dict index for the reference genome sequence. The input is the ref file (*.fa). The output is the dict index for ref. NOTE The index is stored under the same directory of the ref file. Ensure that the directory is writable and no dict index for the ref already exist.
buildfai	Build fai index	Build the fai index for the reference genome sequence. The input is the ref file (*.fa). The output is fai index for ref. NOTE The index is stored under the same directory of the ref file. Ensure that the directory is writable and no fai index for the ref already exists.

Single-mode pipeline		Description
bqsrindex	Build BQSR index	Build the index required for BQSR. The input is the reference genome sequence file (*.fa) and known SNP/INDEL file (*.vcf.gz). The output is the index (*.vcfi) of BQSR. NOTE • The index is stored under the same directory with the ref file by default. If the directory is not writable, the index will be stored under the user output directory. • The same ref file and database file need to be built only once.
bamstats	BAM statistics	BAM file statistic information. The input is BAM file. The output is the statistic information of the BAM file.
vcfstats	VCF statistics	VCF file information statistics. The input is VCF file. The output is the statistic information of the VCF file.

6 Parameter description

Program MegaBOLT

Version: V2.x.x

Usage: `MegaBOLT [options]`

NOTE
- Use the output directory for temporary storage when overcapacity occurs.
- Parameters are case sensitive.

MegaBOLT global parameters

Parameter	Description
`--type <string>`	Select pipeline modes, including: alignment, alignmentsortmarkdup, alignmentsortmarkdupbqsr, bamstats, basic, bqsr, bqsrindex, builddict, buildfai, full, genotypegvcfs, haplotypecaller, mutect2, qc, somatic, sortmarkdup, vcfstats (Default: basic) For details about the pipeline modes, refer to *Pipeline modes (--type) on Page 11*.
`--help | -h`	Output software descriptions.
`--version | -v`	Output software version.
`--list <sample.list>`	The list file is a list of FASTQ files that support PE and SE data. For details about the list file format, refer to *List file on Page 7*.
`--ref <hg19.fa>`	Reference genome sequence file (Default: hg19.fa).
`--vcf <dbsnp.vcf>`	dbSNP file (Default: dbsnp_151.vcf.gz).
`--outputdir | -outdir <Path>`	Output directory (Default: current directory).
`--outputprefix | -prefix <prefix>`	Output file prefix (Default: output). ● Only effective when list file is not an input. ● When inputting the list file, use SampleName in the list file as the prefix of the output file.

Parameter	Description
`--runtype <WGS|WES>`	Pipeline modes (Default: WGS). WGS: Whole Genome Sequencing data analysis WES: Whole Exome Sequencing data analysis
`--bed <BV4|BV5|AV2|AV5|AV6|ACV6|AV7|NV3|NME|TV1.2|IDT|AiJi|BV4-38|BV5-38|AV2-38|AV5-38|AV6-38|ACV6-38|AV7-38|NV3-38|NME-38|TV1.2-38|IDT- 38|AiJi-38|user_defined_path>`	Interval files corresponding to hg19: BV4 BGI_Exome_V4_kit BV5 BGI_Exome_V5_kit AV2 Agilent_Exome_V2 AV5 Agilent_Exome_V5 AV6 Agilent_Exome_V6 ACV6 Agilent.V6COSMIC AV7 Agilent_Exome_V7 NV3 Roche_SeqCapEZ_Exome_v3.0 NME Nimblegen_MedExome_V2 TV1.2 Illumina.truseq.v1.2 IDT xgen_target AiJi AIJI_Exome Interval files corresponding to hg38: BV4-38 BGI_Exome_V4_kit BV5-38 BGI_Exome_V5_kit AV2-38 Agilent_Exome_V2 AV5-38 Agilent_Exome_V5 AV6-38 Agilent_Exome_V6 ACV6-38 Agilent.V6COSMIC AV7-38 Agilent_Exome_V7 NV3-38 Roche_SeqCapEZ_Exome_v3.0 NME-38 Nimblegen_MedExome_V2 TV1.2-38 Illumina.truseq.v1.2 IDT-38 xgen_target AiJi-38 AIJI_Exome User-defined interval files (Please use absolute path for input), for example: */home/my.bed* NOTE Only effective in WES mode.
`--no-markdup <0|1>`	Do not run MarkDup (Default: 0, that is, run by default).
`--no-bqsr <0|1>`	Do not run BQSR (Default: 0, that is, run by default).

Parameter	Description
--no-fastq-output <0|1>	Do not output clean FASTQ file (Default: 0, that is, output by default).
--no-bam-output-for-alignment <0|1>	When running alignmentsortmarkdupbqsr, basic, full, or somatic pipelines, do not output BAM file from duplicate marking (Default: 1, that is, do not output by default).
--no-bam-output-for-sort <0|1>	When running basic, full, or somatic pipelines but not running BQSR, do not output BAM file from duplicate marking (Default: 0, that is, output by default).
--no-bam-output-for-bqsr <0|1>	When running basic, full, or somatic pipelines and running BQSR, do not output BAM file from BQSR (Default: 0, that is, output by default).
--traffic	View traffic information.

Alignment

Parameter	Description
--bwa <0|1>	Use BWA for alignment (Default: 0, that is, use Minimap2 for alignment by default).
--se <0|1>	Process SE reads (Default: 0, that is, process PE reads by default).
--mdtag <0|1>	Output MD tag for alignment (Default: 0, that is, do not output by default). NOTE Only for Minimap2.

SortMarkDup

Parameter	Description
--sort-markdup-input <file>	Input SAM/BAM file for SortMarkDup pipeline (necessary only when running SortMarkDup independently).
--sortmarkdup-input-type <sam|bam>	Input file format (Default: sam).

BQSR

Parameter	Description
`--bqsr-input <file>`	Input SAM/BAM file for SortMarkDup pipeline (necessary only when running SortMarkDup independently).
`--knownSites <file>`	Known SNP/INDEL database file (*.vcf.gz), The default files are: ● dbsnp_151.vcf.gz ● Mills_and_1000G_gold_standard.indels.hg19. vcf.gz ● 1000G_phase1.indels.hg19.vcf.gz NOTE This parameter can be set multiple times.
`--bqsrindex <file>`	Set the BQSR index that has been generated or will be generated.
`--diq <0|1>`	Whether block filed BD and BI for the BQSR BAM file (Default: 0, that is, output by default).

HaplotypeCaller

Parameter	Description
`--haplotypecaller-input <file>`	Input BAM file for Germline variant calling pipeline (necessary only when running Germline variant calling pipeline independently).
`--intervals <file>`	intervals file. NOTE ● Same as “--bed”. ● Only effective when running WGS mode and bed file is not an input.
`--ERC <NONE|GVCF>`	Set whether to output gVCF file (Default: NONE). NONE: Output VCF files. GVCF: Output gVCF files.

Parameter	Description
`--hc4 <0|1>`	Use GATK HaplotypeCaller 4.0 for Germline variant calling (Default: 0, that is, GATK HaplotypeCaller 3.7 is used by default). NOTE Do not use it with “--deepvariant” simultaneously.
`--interval-padding <integer>`	Set the interval padding (Default: 0). NOTE Only applicable to HaplotypeCaller 3.7/4.0.
`--stand-call-conf <integer>`	Set the stand call conf (Default: 30). NOTE Only applicable to HaplotypeCaller 3.7/4.0.
`--pcr-indel-model <CONSERVATIVE|NONE>`	Whether it is PCR-Free (Default: CONSERVATIVE). CONSERVATIVE: PCR data NONE: PCR-Free data NOTE Only applicable to HaplotypeCaller 3.7/4.0.

Parameter	Description
--scalafile <scalafile.scala>	scala file, can be used to set HaplotypeCaller (Default: ExampleHaplotypeCallerFPGA.scala). NOTE ● Only for HaplotypeCaller 3.7. ● If values need to be assigned to ERC, interval_padding, or stand_call_conf, use the corresponding parameter settings first. When these parameters are assigned with values in the scala file, the content of the scala file information will prevail and the values assigned through parameter settings will be ignored. ● We recommend that you use "--vcf" instead of the scala file to set dbSNP. ● For details about the scala file and scala file parameter settings, refer to *Setting HaplotypeCaller by using the scala file on Page 7*.
--arguments-file <file>	HaplotypeCaller file, for details, refer to GATK 4.0 parameter description (Default: Arguments_file(blank file)). NOTE Only applicable to HaplotypeCaller 4.0.
--deepvariant <0|1>	Use DeepVariant for Germline variant calling (Default: 0, that is, GATK HaplotypeCaller 3.7 is used by default). NOTE Do not use it with "--hc4" simultaneously.
--use-openvino <0|1>	Whether to use Intel OpenVINOTM toolkit to accelerate DeepVariant inference (Default: 1, that is, use by default). NOTE Only applicable to DeepVariant.

Parameter	Description
`--MGI-data <0|1>`	Whether to use Intel® OpenVINOTM toolkit to accelerate DeepVariant inference (Default: 1, that is, use by default). NOTE Only applicable to DeepVariant.
`--WGS-mode <PCR|PCR-free>`	For WGS data, choose the deep learning model based on the library preparation method (PCR/PCR-Free) (Default: PCR). NOTE Only applicable to DeepVariant.
`--fast-model <0|1>`	Whether to use the DeepVariant fast inference model in WGS tasks. NOTE • Only applicable to DeepVariant. • Do not set this parameter unless you know its meaning.
`--deepvariant-model`	Run DeepVariant with the user-defined model. NOTE • Only applicable to DeepVariant. • Do not set this parameter unless you know its meaning.

NOTE For details about DeepVariant parameter description, refer to *DeepVariant parameter description on Page 11*.

MuTect2

Parameter	Description
`--mutect2-input <file>`	Input BAM file for MuTect2 (necessary only when running MuTect2 independently). NOTE This parameter can be set multiple times.
`--tumor | -tumor <string>`	Sample name of the tumor sample (necessary only when running MuTect2 independently). NOTE The sample name should be consistent with the Read group sample name from the BAM file of the tumor sample.

Parameter	Description
--normal | -normal <string>	Name of the tumor sample. NOTE The sample name should be consistent with the read group sample (RGSM) name from the BAM file of the normal sample.

GenotypeGVCFs

Parameter	Description
--genotypegvcfs-input <file>	Input gVCF file for GenotypeGVCFs pipeline (necessary only when running GenotypeGVCFs independently). NOTE This parameter can be set multiple times.
--stand-call-conf_genotypegvcfs <integer>	Set the stand call conf of GenotypeGVCFs pipeline (Default: 10).
--allSites <0|1>	Output in allSites mode (Default: 0, that is, do not output by default).

BamStats

Parameter	Description
--bamstats-input <file>	Input BAM file for BamStats pipeline (necessary only when running BamStats independently).

VcfStats

Parameter	Description
--vcfstats-input <file>	Input VCF file for VcfStats pipeline (necessary only when running VcfStats independently).

Somatic

Parameter	Description
`--list2 <sample.list2>`	List file of the normal sample in Somatic pipeline. The file format is the same as the list.
`--no-stats <0|1>`	Do not run BAM/VCF statistics in the Somatic pipeline (Default: 0, that is, run by default). ● Germline variant calling report will not be generated if you do not run QC pipeline or Germline variant calling.
`--no-qc <0|1>`	Do not run QC statistics in the Somatic pipeline (Default: 0, that is, run by default).
`--no-hc <0|1>`	Do not run Germline variant calling in the Somatic pipeline (Default: 1, that is, do not run by default).

7 Use cases

basic

Use the default reference and dbSNP with WGS basic pipeline.

```
MegaBOLT --runtype WGS --list sample.list
```

Use the default reference and dbSNP, with Single End data to be analyzed and WGS basic pipeline.

```
MegaBOLT --runtype WGS --se 1 --list sample.list
```

Use the default reference, dbSNP, and knownSites, and the pipeline mode is WGS basic pipeline.

```
MegaBOLT --runtype WGS --list sample.list --ref ref.fa --vcf
b37.vcf --knownSites  b37.vcf
```

Use the default reference and dbSNP, with WGS pipeline. Use PCR-Free library preparation for input samples. Use BWA for alignment. Do not output BAM file after BQSR. Use HaplotypeCaller 4.0 for variant calling basic pipelines.

```
MegaBOLT --runtype WGS --list sample.list --pcr-indel-model
NONE --bwa 1 --hc4 1 --no-bam-output-for-bqsr 1
```

Use the default reference and dbSNP, with WGS pipeline. Use PCR-Free library preparation for input samples. Output BAM after position sorting. Use DeepVariant for variant calling basic pipelines.

```
MegaBOLT --runtype WGS --list sample.list --deepvariant 1
--no-bam-output-for-alignment 0 --WGS-mode PCR-free
```

Use the default reference and dbSNP, preset interval file BV5, output to the user-defined directory. The pipeline mode is WES basic pipeline.

```
MegaBOLT --runtype WES --list sample.list --bed BV5
--outputdir ./out
```

Use the default reference, dbSNP, knownSites, and intervals files, and the pipeline mode is WES basic pipeline.

```
MegaBOLT --runtype WES --list sample.list --ref ref.fa --vcf
b37.vcf --knownSites  b37.vcf --deepvariant 1 --bed user.bed
```

 NOTE The parameters above also functions the same in the full pipeline.

full

Use the default reference and dbSNP with WGS full pipeline.

```
MegaBOLT --type full --runtype WGS --list sample.list
```

Use the default reference, bed, and dbSNP, do not ouput FASTQ files after QC, do not run BQSR, and the pipeline mode is WGS full pipeline.

```
MegaBOLT --type full --runtype WGS --list sample.list --ref
ref.fa --vcf b37.vcf --no-fastq-output 1 --no-bqsr 1
```

Use the default reference, dbSNP, and intervals files, and the pipeline mode is WES basic pipeline.

```
MegaBOLT --type full --runtype WES --list sample.list
```

somatic

Use the default reference and dbSNP with WGS basic pipeline. Use tumor/normal Somatic variant calling pipeline.

```
MegaBOLT --type somatic --runtype WGS --list tumor.list --list2
normal.list
```

Use the default reference and dbSNP with WGS pipeline. Use tumor Somatic variant calling pipeline.

```
MegaBOLT --type somatic --runtype WGS --list tumor.list
```

Use the default reference and dbSNP. Do not run QC, Haplotype variant calling, and statistics. The pipeline mode is WGS. Use tumor/normal Somatic variant calling pipeline.

```
MegaBOLT --type somatic --runtype WGS --list tumor.list --list2
normal.list --ref ref.fa --vcf b37.vcf --no-qc 1 --no-hc 1
--no-stats 1
```

Use the default reference and dbSNP with WES pipeline. Use the preset interval file BV5. Output to the user-defined directory. Use tumor/normal variant calling pipeline.

```
MegaBOLT --type somatic --runtype WGS --bed BV5 --list tumor.
list --list2 normal.list --outputdir ./out
```

alignment

Use Minimap2 for alignment.

```
MegaBOLT --type alignment --list sample.list
```

Use BWA for alignment. The data to be analyzed is Single End data.

```
MegaBOLT --type alignment --list sample.list --bwa 1 --se 1
```

sortmarkdup

Use sortmarkdup for position sorting and duplicate marking.

```
MegaBOLT --type sortmarkdup --sortmarkdup-input input.sam
```

Use sortmarkdup for position sorting, set input type to bam, and specify the file name prefix.

```
MegaBOLT --type sortmarkdup --sortmarkdup-input input.bam
--sortmarddup-input-type bam --outputprefix myoutputprefix
--no-markdup 1
```

alignmentsortmarkdup

Use combined-pipeline modes for alignment and duplicate marking.

```
MegaBOLT --type alignmentsortmarkdup --list sample.list
```

alignmentsortmarkdupbqsr

Use combined-pipeline modes for alignment, position sorting, and BQSR.

```
MegaBOLT --type alignmentsortmarkdupbqsr --list sample.list
```

bqsrindex

Use the default reference, dbSNP, and knownSites to generate BQSR index, and save the index to a specific file format.

```
MegaBOLT --type bqsrindex --ref ref.fa --vcf dbsnp_151.vcf.gz
--bqsrindex ref.fa.vcfi --knownSites dbindel.vcf.gz
```

bqsr

Use the default reference, dbSNP, and knownSites for BQSR.

```
MegaBOLT --type bqsr --bqsr-input input.bam
```

Use the default reference, dbSNP, and knownSites for BQSR, specify the file name prefix, and save the BQSR index to a specific file format.

```
MegaBOLT --type bqsr --bqsrindex ref.fa.vcfi --bqsr-input
input.bam --ref ref.fa --vcf dbsnp_151.vcf.gz --knownSites
dbindel.vcf.gz --outputprefix myoutputprefix
```

haplotypecaller

Use the default reference and dbSNP with WGS pipeline. Use HaplotypeCaller 3.7 for variant calling.

```
MegaBOLT --type haplotypecaller --runtype WGS
--haplotypecaller-input input.bam
```

Use the default reference and dbSNP with WGS pipeline. Set the user-defined interval-padding and stand-call-conf. Use HaplotypeCaller 4.0 for variant calling.

```
MegaBOLT --type haplotypecaller --runtype WGS  --ref ref.fa
--vcf b37.vcf --scalafile example.scala --haplotypecaller-input
input.bam --interval-padding 10 --stand-call-conf 10 --hc4 1
```

Use the default reference, dbSNP, and interval files with WES pipeline. Use the user-defined scala file and specify the file name prefix. Use HaplotypeCaller 3.7 for variant calling and output genotype information.

```
MegaBOLT --type haplotypecaller --runtype WES
--haplotypecaller-input input.bam --scalafile exapmle.scala
--ERC GVCF --outputprefix myoutputprefix
```

deepvariant

Use the default reference and dbSNP. Use DeepVariant fast inference model for variant calling of the WGS alignment data that adopts PCR library preparation.

```
MegaBOLT --type haplotypecaller --runtype WGS --deepvariant 1
--haplotypecaller-input input.bam
```

Use the default reference and dbSNP. Use DeepVariant standard inference model for variant calling of the WGS alignment data, which adopts PCR-Free library preparation.

```
MegaBOLT --type haplotypecaller --deepvariant 1 --ref ref.fa
--vcf b37.vcf --haplotypecaller-input input.bam --runtype WGS
--WGS-mode PCR-free --fast-model 0
```

Use the default reference, dbSNP, and interval files. Use DeepVariant standard inference model for variant calling of the WES alignment data, and output genotype information.

```
MegaBOLT --type haplotypecaller --deepvariant 1
--haplotypecaller-input input.bam --runtype WES --intervals
BV4 --ERC GVCF
```

mutect2

Use the default reference and dbSNP. Use tumor/normal Somatic variant calling pipeline.

```
MegaBOLT --type mutect2 --mutect2-input tumor.bam
--mutect2-input normal.bam --tumor tumorsamplename --nomal
normalsamplename
```

Use the user-defined reference and dbSNP. Use tumor Somatic variant calling pipeline, and output the results to the specified directory.

```
MegaBOLT --type mutect2 --mutect2-input tumor.bam --tumor
tumorsamplename --ref ref.fa --vcf b37.vcf --outputdir ./out
```

genotypegvcfs

Use default reference and dbSNP, and run GenotypeGVCFs.

```
MegaBOLT --type genotypegvcfs --genotypegvcfs-input input.
g.vcf.gz
```

Use user-defined reference, dbSNP, and genotypegvcfs-stand-call-conf, run GenotypeGVCFs, and output allSites information.

```
MegaBOLT --type genotypegvcfs --genotypegvcfs-input input.
g.vcf.gz --ref ref.fa --vcf b37.vcf --genotypegvcfs-stand-
call-conf 30 --allSites 1
```

bamstats

Summarize the BAM file generated from Paired End data, and output to the specified directory.

```
MegaBOLT --type bamstats --bamstats-input input.bam --outputdir
./out
```

Summarize the BAM file generated from Single End data, and output to the specified directory.

```
MegaBOLT --type bamstats --bamstats-input input.bam --outputdir
./out --se 1
```

vcfstats

Summarize the vcf file and output to the specified directory.

```
MegaBOLT --type vcfstats --vcfstats-input input.vcf.gz
--outputdir ./out
```

builddict

Generate a dict file for the specified reference.

```
MegaBOLT --type builddict --ref ref.fa
```

buildfai

Generate a index file for the specified reference.

```
MegaBOLT --type buildfai --ref ref.fa
```

8 Output directory and results

Germline variant calling pipeline

If the program runs smoothly, the following navigation panel is generated under the task output directory:

```
├── megabolt.log
├── megabolt.out
└── samplename/
    ├── samplename_1.fq.gz
    ├── samplename_2.fq.gz
    ├── samplename.list
    ├── samplename.log
    ├── samplename.mm2.sortdup.bqsr.bam
    ├── samplename.mm2.sortdup.bqsr.bam.bai
    ├── samplename.mm2.sortdup.bqsr.bam.grp
    ├── samplename.mm2.sortdup.bqsr.HaplotypeCaller.vcf.gz
    ├── samplename.mm2.sortdup.bqsr.HaplotypeCaller.vcf.gz.out
    ├── samplename.mm2.sortdup.bqsr.HaplotypeCaller.vcf.gz.tbi
    ├── samplename.out
    ├── report/ (only for full pipeline)
    │   ├── samplename_cn.html
    │   ├── samplename_en.html
    │   └── samplename.report.zip
    └── stat/ (only for full pipeline)
        ├── bam_stats/
        │   ├── cumu.txt
        │   ├── depth_frequency.txt
        │   ├── samplename.bamstat.xls
        │   ├── samplename.CollectInsertSizeMetrics.txt
        │   ├── samplename.cumuPlot.png
        │   ├── samplename.depthstat.xls
        │   ├── samplename.gc_bias_metrics.xls
        │   ├── samplename.gcbias.png
```

```
│   ├── samplename.histPlot.png
│   ├── samplename.insertsize.png
│   ├── samplename.samtoolsstat.xls
│   └── samplename.Summary.xls
├── qc/
│   ├── samplename_1.fq.gz.check
│   ├── samplename_2.fq.gz.check
│   ├── samplename.base.png
│   ├── samplename.fqstat.xls
│   └── samplename.qual.png
└── vcf_stats/
    └── samplename.vcfstat.xls
```

The task output directory is as follows:

megabolt.log	Client standard output stream/standard error stream.
megabolt.out	Client program running log.
samplename/	Result directory.

The structure of the result output directory is as follows (Taking 100 GB gzip file for example, results from sub-modules and logs generated in the run are stored in the sibling directory):

samplename.fq.gz	Filtered Reads file (100 GB).
samplename.list	Transformed sample list.
samplename.bam*	Alignment file and index after duplicate marking and BQSR (200 GB).
samplename.vcf*	Variant calling results and index (200 MB for VCF and 6GB for gVCF).
samplename.log	Task analysis log.
samplename.out	Program running log.
report/	Sample report and its compressed package (only for full pipeline).
stat/	Statistics file (only for full pipeline).

The following files are generated after running the full pipeline:

Full pipeline result report (report/):

report.zip	Compressed package of all samples (*/*.html)
*_en.html	English Sample Analysis report
*_cn.html	Chinese Sample Analysis report

Full pipeline statistics file (stat/):

bam_stats/	Statistic information for alignment results.
qc/	Quality control information for QC results.

vcf_stats/	Statistic information for variant calling results.

samplename.fqstat.xls (qc/):

Sample	WGS
Read_length	100:100
Read_raw	941148380
Read_clean	941148366
Rate_clean	100%
Q20_raw	97.1%
Q30_raw	88.5%
GC_raw	41%
Q20_clean	97.1%
Q30_clean	88.5%
GC_clean	41%

samplename.bamstat.xls (bam_stats/):

Sample	WGS
Mapping_Rate	99.01%
PE_Mapping_Rate	98.62%
Duplication_Rate	1.77%
Mismatch_Rate	0.65%
Insert_size	388.0
Average_depth(rmdup)	31.22
Coverage(>=1X)	99.10%
Coverage(>=4X)	98.53%
Coverage(>=10X)	98.32%
Coverage(>=20X)	92.41%
Uniformity(>0.2f)	97.94%

samplename.vcfstat.xls (vcf_stats)

Sample	WGS
Total_SNP	3844409
dbSNP_rate	98.66%
Novel_SNP	51597
Novel_SNP_Rate	1.34%
Ti/Tv	2.01
Total_INDEL	859574
dbINDEL_Rate	86.29%

Somatic variant calling pipeline

If the program runs smoothly, the following directory tree is generated under the result output directory:

```
├── megabolt.log
├── megabolt.out
└── tumor-name_normal-name/
    ├── normal-name/
    ├── tumor-name/
    ├── tumor-name_normal-name.log
    ├── tumor-name_normal-name.Mutect2.vcf
    └── tumor-name_normal-name.out
```

The structure of the task output directory is as follows:

megabolt.log	Client standard output stream/standard error stream.
megabolt.out	Client program log.
tumor-name_normal-name/	Result directory.

NOTE For details about the MegaBOLT client, refer to Chapter 5 of *MegaBOLT_User_Manual_Advanced*.

The structure of the result output directory is as follows (Taking 100 GB gzip file for example, results from sub-modules and logs generated in the run are stored in the sibling directory):

normal-name/	Analysis result directory of the tumor sample.
tumor-name/	Analysis result directory of the normal sample.
tumor-name_normal-name.log	Task analysis log.
tumor-name_normal-name.Mutect2.vcf	VCF file for Somatic variant calling.
tumor-name_normal-name.out	Program running log.

The result directory structure for turmor sample analysis, normal sample analysis, and Germline variant calling pipeline are consistent.

9 Precautions for parameter settings

Relations between --ref, --vcf, and --knownSites

Reference genome file (hereinafter called ref file) set through --ref and dbSNP file set through --vcf should match the known SNP/INDEL database file (hereinafter called knownSites file) set through --knownSites. Otherwise, the analysis correctness might be affected.

When one or more of the --ref, --vcf, and --knownSites parameters are entered, the specific behavior of the program is different. The detailed description is shown in the following table:

Set --ref	Set --vcf	Set --knownSites	Allow	Description
No	No	No	Yes	Use the default ref file (hg19.fa), default dbSNP file (dbsnp_151.vcf.gz), default knownSites file (dbsnp_151.vcf.gz, Mills_and_1000G_gold_standard.indels.hg19. Vcf.gz, 1000G_phase1.indels.hg19.vcf.gz).
No	No	Yes	No	Because the user does not set the ref file, therefore, whether the dbSNP/knownSites file set by the user matches the default ref file (hg19.fa) is unknown.
No	Yes	No		
No	Yes	Yes		

Set --ref	Set --vcf	Set --knownSites	Allow	Description
Yes	No	No	Yes	The user set the ref file but not the dbSNP and knownSites file. Germline variant calling pipeline will not use dbSNP. The program tries to find the BQSR index in the ref file directory that is set by the user. If the operation succeeds, the index found is used to run BQSR, otherwise, the BQSR pipeline will not run.
Yes	No	Yes	Yes	The user set the ref file but not the dbSNP file. Germline variant calling pipeline will not use dbSNP.
Yes	Yes	No	Yes	The user set the ref file and dbSNP file but not the knownSites file. The program will use the dbSNP file set by the user as the knownSites file.
Yes	Yes	Yes	Yes	Use the ref file, dbSNP file, and knownSites file set by the user. Do not verify whether the files match.

Build BQSR index (--bqsrindex)

BQSR index need to be built before running BQSR pipeline. Normally, the BQSR index is built automatically. You can also run bqsrindex to build BQSR index based on practical requirements.

The policy for building BQSR index varies with whether you have set --ref, --vcf, --knownSites, and --bqsrindex, as shown in the table below:

Set --ref	Set --vcf or --knownSites	Set --bqsrindex	Allow	Description
No	No	No	Yes	Use the default BQSR index that has been set in advance.
No	No	Yes	No	Because the user does not set the ref file, therefore, whether the dbSNP/knownSites/BQSR index file set by the user matches the default ref file (hg19.fa) is unknown.
No	Yes	No		
No	Yes	Yes		
Yes	No	No	Yes	The program tries to find the BQSR index in the ref file directory that is set by the user. If the operation succeeds, the index found is used to run BQSR, otherwise, the BQSR pipeline will not run.
Yes	No	Yes	Yes	Run BQSR pipeline by using the file that is set through --bqsrindex by the user as the BQSR index.
Yes	Yes	No	Yes	The program tries to find the BQSR index in the ref file directory that is set by the user. If the operation succeeds, the index found is used to run the BQSR pipeline. Otherwise, the program first tries to build BQSR index in the ref file directory by using the ref/dbSNP/knownSites files that are set by the user. If the diretory is not writable, it then tries to build BQSR index in the user output directory.
Yes	Yes	Yes	Yes	If the file that is set through --bqsrindex by the user exists, use it as the BQSR index to run BQSR pipeline. Otherwise, use the ref/dbSNP/knownSites files set by the user to build BQSR index in the location specified by --bqsrindex, and use the file to run BQSR pipeline.

Relations between --ref, --bed, and --runtype

To use --ref, --bed, and --runtype simultaneously, pay attention to their constraints, as shown in the table below:

Set --ref	Set --bed	Set --runtype	Allow	Description
No	No	No/Yes (WGS)	Yes	Run WGS mode.
No	No	Yes (WES)	Yes	Run WES mode. BV4 is used as the bed file by default.
No	Yes	No/Yes (WES)	Yes	If bed set by the user exists in the known bed list, use it to run WES mode. Otherwise, the program cannot run because it cannot confirm if the bed file input by the user matches the default ref file (hg19.fa).
No	Yes	Yes (WGS)	No	Cannot specify bed file in the WGS mode.
Yes	No	No/Yes (WGS)	Yes	Run WGS mode.
Yes	No	Yes (WES)	No	The user sets the ref file and select the WES mode, but does not set the bed file that matches the ref file, the program cannot run.
Yes	Yes	No/Yes (WES)	Yes	When bed set by the user exists in the known bed list, and the ref file contains field “19” and “38”, run WES mode; but if the ref tile does not contain field “19” and “38”, the program cannot run. If bed set by the user does not exist in the known bed list, run WES mode and use the bed file set by the user.
Yes	Yes	Yes (WGS)	No	Cannot specify bed file in the WGS mode.

Setting HaplotypeCaller by using the scala file

Parameter setting description

If values need to be assigned to ERC, interval_padding, or stand_call_conf, use the corresponding parameter settings first. When these parameters are assigned with values in the scala file, the content of the scala file information will prevail and the values assigned through parameter settings will be ignored.

We recommend that you use “--vcf” instead of the scala file to set dbSNP.

Methods for parameter settings based on the default scala file are as follows:

GATK	Methods for setting scala file (varcall formula)
-stand_call_conf 30	this.stand_call_conf = 30
-emitRefConfidence	this.emitRefConfidence = ReferenceConfidenceMode.GVCF

Default scala file

File name: *ExampleHaplotypeCallerFPGA.scala*

File content:

```
package org.broadinstitute.gatk.queue.qscripts.examples
import org.broadinstitute.gatk.queue.QScript
import org.broadinstitute.gatk.queue.extensions.gatk._
import org.broadinstitute.gatk.utils.commandline.Hidden
import org.broadinstitute.gatk.utils.commandline._
import org.broadinstitute.gatk.queue.util.QScriptUtils
import org.broadinstitute.gatk.queue.function.
ListWriterFunction
import org.broadinstitute.gatk.utils.variant.GATKVCFIndexType
import org.broadinstitute.gatk.tools.walkers.haplotypecaller.
ReferenceConfidenceMode
import org.broadinstitute.gatk.utils.pairhmm.PairHMM.HMM_
IMPLEMENTATION
import org.broadinstitute.gatk.tools.walkers.haplotypecaller.
PairHMMLikelihoodCalculationEngine.PCR_ERROR_MODEL
```

```
class ExampleHaplotypeCaller extends QScript {
  qscript =>
  @Input(doc="The reference file for the bam files.",
shortName="R")
  var referenceFile: File = _ // _ is scala shorthand for null
  @Input(doc="Bam file to indel realigner.", shortName="I")
  var bamFile: File = _
  @Input(doc="Vcf file.", shortName="O")//, required=false
  var vcfFile: File = _
  @Input(doc="an intervals file to be used by GATK - output
bams at intervals only", fullName="gatk_interval_file",
shortName="intervals", required=false)
  var intervals: File = _
  @Argument(doc="Is output gvcf file.", shortName="ERC",
required=false)
  var emitRefConfidence: String = _
  @Argument(doc="Parameter stand_call_conf.", shortName="stand_
call_conf", required=false)
  var stand_call_conf: Int = 10
  @Input(doc="Parameter dbsnp.", shortName="dbsnp",
required=false)
  var dbsnp: File = _
  @Argument(doc="Parameter interval_padding.",
shortName="interval_padding", required=false)
  var interval_padding: Int = 0
  @Argument(doc="Parameter pcr_indel_model.", shortName="pcr_
indel_model", required=false)
  var pcr_indel_model: String = _
  @Hidden
  @Argument(doc="How many ways to scatter/gather",
fullName="scatter_gather", shortName="sg", required=false)
  var nContigs: Int = -1
  trait CommandLineGATKArgs extends CommandLineGATK {
    this.reference_sequence = qscript.referenceFile
  }

  case class varcall (inBam: File, outVCF: File) extends
HaplotypeCaller with CommandLineGATKArgs {
    this.input_file :+= inBam
```

```
    if(qscript.emitRefConfidence != null && qscript.
emitRefConfidence == “GVCF”){
        this.emitRefConfidence = ReferenceConfidenceMode.GVCF
    }
    if(this.emitRefConfidence == ReferenceConfidenceMode.GVCF){
        if(!outVCF.endsWith(“.g.vcf”) && !outVCF.endsWith(“.
g.vcf.gz”)){
                this.out = outVCF.replace(“.vcf”, “.g.vcf”)
                print(“Changing name for GVCF file to “ +
this.out + “\n”)
        }
        else{
                this.out = outVCF
        }
    }
    else{
        if(outVCF.endsWith(“.g.vcf”) || outVCF.endsWith(“.
g.vcf.gz”)){
                this.out = outVCF.replace(“.g.vcf”, “.vcf”)
                print(“Changing output name for VCF file to “
+ this.out + “\n”)
        }
        else{
                this.out = outVCF
        }
}
    if(qscript.pcr_indel_model != null)
    {
        if(qscript.pcr_indel_model == “NONE”)
        {
                this.pcr_indel_model = PCR_ERROR_MODEL.NONE
        }
        if(qscript.pcr_indel_model == “CONSERVATIVE”)
        {
                this.pcr_indel_model = PCR_ERROR_MODEL.
CONSERVATIVE
        }
        if(qscript.pcr_indel_model == “HOSTILE”)
        {
                this.pcr_indel_model = PCR_ERROR_MODEL.HOSTILE
```

```
        }
        if(qscript.pcr_indel_model == “AGGRESSIVE”)
        {
                this.pcr_indel_model = PCR_ERROR_MODEL.
AGGRESSIVE
        }
    }

//    this.pcr_indel_model = PCR_ERROR_MODEL.NONE
    this.interval_padding = qscript.interval_padding
    this.stand_call_conf = qscript.stand_call_conf
    if(qscript.dbsnp != null){
        this.dbsnp = qscript.dbsnp
    }
    this.intervals = if (qscript.intervals == null) Nil else
List(qscript.intervals)

// add or delete parameters

    this.nct = 3
    this.variant_index_type =  GATKVCFIndexType.LINEAR
    this.variant_index_parameter = 128000
    this.scatterCount = qscript.nContigs
    this.memoryLimit = 4
    this.pair_hmm_implementation = HMM_IMPLEMENTATION.VECTOR_
LOGLESS_CACHING_FPGA_EXPERIMENTAL
  }

  def script() {
      nContigs = 24
      val recalBam = qscript.bamFile
      val finalVCF = qscript.vcfFile
      add(varcall(recalBam, finalVCF))
    }
}
```

DeepVariant parameter description

DeepVariant variant calling module is an optional submodule of MegaBOLT. It inherits basic variant calling parameters from haplotypecaller pipeline. The following haplotypecaller pipeline parameters is effective for DeepVariant:

```
--haplotypecaller-input
--ERC
--deepvariant
--use-openvino
--MGI-data
--WGS-mode
--fast-model
--deepvariant-model
```

For different types of data to be analyzed, Deepvariant provides different deep learning models based on the following parameters that are specified by the user:

```
--mgi-data
--runtype
--WGS-mode
```

In addition, when you use --deepvariant-model, --fast-model, and --use-openvino simultaneously, pay attention to the constraint relations between them. The legality of the parameter combination is shown in the table below:

--deepvariant-model specified or not	--fast-model value	--use-openvino value	Allow	Description
No	0	0	Yes	Use TensorFlow for the deep learning of variant calling, which is slower than using OpenVINO.
No	0	1	Yes	Use OpenVINO to accelerate the model inference flow, use standard DeepVariant model structure.
No	1	0	No	MegaBOLT does not support the use of fast inference model through TensorFlow for variant calling at present.

--deepvariant-model specified or not	--fast-model value	--use-openvino value	Allow	Description
No	1	1	Yes	Use OpenVINO to accelerate the model inference flow, use DeepVariant fast inference model.
Yes	0	0	Yes	Use user-defined deep learning model for variant calling.
Yes	0	1	No	OpenVINO does not support TensorFlow model trained by the user.
Yes	1	0	Yes	Fast model selection will be skipped if you has specified a user-defined model.
Yes	1	1	No	OpenVINO does not support TensorFlow model trained by the user.

DeepVariant allows you to use user-defined inference model through --deepvariant-model for variant calling. But this function is only applicable to users who have some experience in deep learning development. Do not set this parameter unless you know its meaning. For example, use the default reference and dbSNP, and use DeepVariant fast inference model for variant calling of the WGS alignment data, as shown below:

```
MegaBOLT --type haplotypecaller --deepvariant 1
--haplotypecaller-input input.bam --sample-mode WGS
--deepvariant-model user.model.ckpt --fast-model 0 --use-
openvino 0
```

10 Software update log

Date	Revision	Description
2017/12/15	V1.0	Initial release
2019/05/07	V1.5.4	● Set Minimap2 as the default alignment software (Qaligner is optional) ● Added support for SE data. ● Added extractdepth pipeline. ● Update BQSR and the result is consistent with the original version. ● Including two workflows: MegaBOLT and MegaBOLT-full.
2019/09/23	V1.5.6	● Added allnomarkdup, alignmentsort, and sort pipelines. ● Added DeepVariant pipeline. ● Added BWA into alignment software options.
2019/11/30	V2.1.0	● Multi-task scheduler version. ● Merged the previous MegaBOLT (basic) and MegaBOLT-full (full) pipelines. ● Added Somatic full pipeline for analysis. ● Each step of the full pipelines can be run indepedently. ● Do not support extractdepth pipeline. ● Do not support Qaligner for alignment. ● Do not support parameter setting of SOAPnuke.

---This page is intentionally left blank.---