Avida DNA Targeted Sequencing Analysis

Analysis of Avida DNA sequencing data using open-source software packages

Technical Guide

Version B0, September 2024

For Research Use Only. Not for use in diagnostic procedures.

Notices

Manual Part Number

G9409-90001

Edition

Version B0, September 2024

Agilent Technologies, Inc.
5301 Stevens Creek Blvd
Santa Clara, CA 95051 USA

Technical Support

For US and Canada

Call (800) 227-9770 (option 3,4,4)

Or send an e-mail to:
ngs.support@agilent.com

For all other regions

Agilent's world-wide Sales and Support Center contact details for your location can be obtained at
www.agilent.com/en/contact-us/page.

Warranty

The material contained in this document is provided "as is," and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Safety Notices

CAUTION

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.

WARNING

A WARNING notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.

In this Guide...

This guide provides an analysis pipeline for Illumina NGS sequencing files generated from libraries prepared with the Avida DNA reagents.

What's New in Version B0

- Added bcftools to Table 1.
- Added note to step 2
- Corrected example script for step 4.
- Corrected tool listed for step 6 to Picard.
- Updated step 22 to allow for use of the covered regions BED file from SureDesign.
- Clarified steps in which the input or output SAM file is a BAM file.

1 Introduction to Avida Analysis

This chapter provides an overview of Avida products and an introduction to the purpose of the Avida Targeted Sequencing Analysis Technical Guide.

Overview of the Avida Reagent Kits and Panels

Detection of tumor-specific variants in cell-free DNA (cfDNA) samples can be challenging due to limiting sample quantities and low abundance of tumor DNA within a large amount of normal DNA. The Avida library preparation and target enrichment system was specifically designed to address these challenges to provide highly sensitive detection of somatic variants and methylated DNA using cfDNA (1–100 ng for detection of variants or 3–100 ng for detection of methylation changes) or 10–100 ng of sheared genomic DNA (gDNA).

The performance of the Avida platform is enabled through several technological innovations:

- A highly optimized library preparation formulation that efficiently converts cfDNA molecules and gDNA fragments into Illumina-sequenceable inserts.
- Dual unique molecule identifiers (UMIs) to tag top and bottom strands and allow for error correction.
- A PCR-free library preparation method that reduces bias and the risk of allele dropout.
- A target enrichment method that leverages cooperative probe binding to capture 70–80% of both strands of the DNA molecules.

Together, these features allow for detection of small variants down to a frequency of 0.1% as well as low-frequency structural variants (Table 2 through Table 4).

The Avida portfolio consists of three library preparation kit options that support DNA analysis (Avida DNA reagent kits), methylation analysis (Avida Methyl reagent kits), and DNA plus methylation analysis from a single sample (Avida Duo Methyl reagent kits). Along with the library preparation kits, three DNA and one methylation-specific catalog panels are available. The DNA panels, which are compatible with the Avida DNA and Avida Duo workflows, were designed for a range of applications and cover distinct gene sets. The 26-kb Avida DNA Focused Cancer Panel covers hot-spot regions in 14 oncogenes and tumor suppressor genes and is well-suited for low-allele frequency detection with a modest sequencing budget. The larger 345-kb Avida DNA Expanded Cancer Panel includes >100 genes, 80 of which have full exonic coverage, to allow for both mutation and copy number detection. The Expanded panel also includes key intronic regions for the detection of translocations. For genome profiling applications, the 2.7-Mb Avida DNA Discovery Cancer Panel covers the full exons of 682 genes and translocation hot-spot introns.

About the Avida Targeted Sequencing Analysis Technical Guide

This technical guide describes the steps for analysis of Avida DNA targeted sequencing data using open-source software packages. Detailed protocols outlining the library prep and target capture steps of the Avida DNA and Avida Duo workflows can be found on the Agilent website. See https://www.agilent.com/cs/library/usermanuals/public/G9409-90000.pdf and https://www.agilent.com/cs/library/usermanuals/public/G9439-90000.pdf.

The analysis pipeline for the Avida DNA workflow starts with paired-end sequencing data files in FASTQ format generated by the Illumina sequencer. The pipeline consists of UMI analysis, sequence alignment, data QC metrics creation, and variant calling, with options for generating single-stranded or double-stranded (duplex) consensus reads and performing position-based deduplication. This technical guide describes each of these analysis pipeline steps, providing detailed step-by-step command line examples. The information is intended to guide you as you use your FASTQ input files to locate genomic alterations, specifically single nucleotide variants (SNVs), small insertion and deletions (indels), copy number variations (CNVs), and translocations

(TLs). All steps described in this publication make use of publicly available open-source tools. Table 1 lists the open-source software packages used in the example scripts in Chapter 3, "Shell Script Examples."

Open-Source Tools for Avida Analysis

Table 1 Open-source software packages*

Package name	Versions tested in this guide
BEDTools	2.25.0
BWA	0.7.12-r1039
fgbio	1.5.2
Java	1.8.0_322
Miniconda3	3.9.5
Picard	2.20.1
SAMtools	1.9
SnpEff/SnpSift	4.2
VarDict	1.5.0
R	3.5.1
PureCN	2.8.0
bcftools	1.11
GATK	4.2.0.0
GeneFuse	0.6.0

* The Software may utilize third party software made available under various open source and third party software licenses ("Third Party Components"). The terms associated with the Third Party Components are available in the comprehensive-license-disclosure.txt within the ExDViewer Program Files Directory. You agree to comply with all applicable terms. In addition to the warranty disclaimers contained in the terms associated with the Third Party Components, Agilent makes the following disclaimers regarding the Third Party Components on behalf of itself, and the copyright holders, contributors, and licensors of the Third Party Components: To the maximum extent permitted by applicable law, the Third Party Components are provided by the copyright holders, contributors, licensors, and Agilent "as is" and AGILENT MAKES NO WARRANTIES OR REPRESENTATIONS OF ANY KIND, WHETHER ORAL OR WRITTEN, WHETHER EXPRESS, IMPLIED, OR ARISING BY STATUTE, CUSTOM, COURSE OF DEALING, OR TRADE USAGE, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF TITLE, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. In no event will the copyright owner, contributors, licensors, or Agilent 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 the Third Party Components.

Examples of Variant Analysis Results

The data in Table 2 through Table 4 summarize variant analysis results obtained using the pipeline described in this technical guide. The sequencing libraries used for this example data were prepared with LGC SeraCare ctDNA Reference Material enriched with Avida DNA catalog panels. The 0.5% allele frequency (AF) SeraCare ctDNA Reference Material V4 was titrated with the 0% AF SeraCare ctDNA Reference Material (beta products provided to Agilent by LGC) to create samples with a range of variant allele frequencies (VAF). In these experiments, the Avida DNA Expanded Cancer Panel ("Expanded panel") and Avida DNA Focused Cancer Panel ("Focused panel") were used for DNA capture with 20 ng and 40 ng of DNA input. Performance measures are calculated based on variants covered by the tested panel.

Table 2 SNV/indel analysis performance with SeraCare ctDNA reference material at multiple VAF levels and multiple input quantities.

Avida DNA Panel	Sequencing depth	Specificity	Detection rate				
			0.1% VAF 20 ng input	0.1% VAF 40 ng input	0.2% VAF 20 ng input	0.2% VAF 40 ng input	0.5% VAF 20 ng input
Focused panel	10M/20M PE	96%	85%	100%	100%	100%	100%
	5M PE	100%	72%	93%	100%	100%	100%
Expanded panel	80M/160M PE	93%	NA	92%	94%	98%	100%
	50M PE	100%	NA	86%	92%	96%	100%

Table 3 CNV analysis performance with SeraCare ctDNA reference material at 0.5% VAF and 20 ng input. Gene-level CNVs were identified with PureCN.

Avida DNA Panel	Sequencing depth	Detection rate	Positive Predictive Value (PPV)
Expanded panel	40M/80M PE	100%	100%

Table 4 Translocation detection performance with SeraCare ctDNA reference material at multiple VAF levels and multiple input quantities.

Avida DNA Panel	Sequencing depth	Detection rate			
		0.1% VAF 40 ng input	0.2% VAF 20 ng input	0.2% VAF 40 ng input	0.5% VAF 20 ng input
Expanded panel	80M/160M PE	71%	71%	86%	93%

2
Avida Analysis Pipeline Steps

This chapter describes the steps of the analysis pipeline for Avida DNA targeted sequencing data.

Avida DNA Targeted Sequencing Analysis Pipeline Summary

The sequence of steps for the Avida DNA analysis pipeline is illustrated in Figure 1. The division of steps used in this technical guide was designed for simplicity in concept and logic. For a production pipeline, some of these steps could be connected through pipes ("|") in shell command line where it is desirable to optimize the process to minimize disk I/O and processing time. An example of such practice can be found in the fgbio guide: github.com/fulcrumgenomics/fgbio/blob/main/docs/best-practice-consensus-pipeline.md.

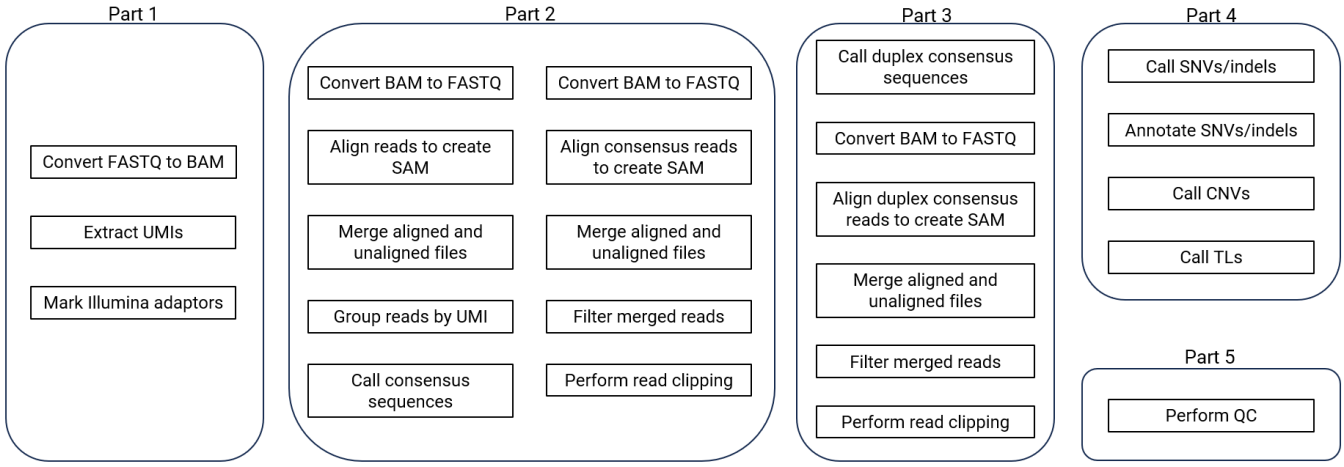

Figure 1 Avida DNA analysis pipeline

The steps shown in Figure 1 are described in further detail below. The steps are grouped into functional blocks ("parts") for conceptual clarification.

Analysis Pipeline Steps

Part 1. Read pre-processing (steps 1–3)

The input for part 1 of the analysis pipeline is the paired-end FASTQ files (R1.fq and R2.fq) for a single DNA library sample. You start by converting the FASTQ files to SAM format using Picard tools. Then, with the SAM file as input, use the fgbio consensus pipeline to extract and trim the in-line UMI sequences from both ends of the reads (5' ends of both R1 and R2) and keep the combined UMI in the RX tag of the SAM file. Finally, use Picard to mark the standard Illumina adaptors in the reads in the SAM file.

1 Convert FASTQ files to BAM file (Picard)

2 Extract UMIs from BAM file (fgbio)

3 Mark Illumina adaptors in BAM file (Picard)

Part 2. Single-stranded UMI processing and alignment (steps 4–13)

In part 2, you convert the SAM file created in part 1 to a single FASTQ file that contains the trimmed R1 and R2 reads. You then align the reads in BWA, generating a BAM file, and use that BAM file for single-stranded UMI consensus matching in fgbio. Next, you convert the BAM file containing the UMI consensus reads to a FASTQ file and use the FASTQ file to generate a more fine-tuned alignment. Using the resulting SAM file, the UMI-deduplicated reads are filtered based on quality metrics specified in fgbio and overlapping bases between the R1 and R2 reads are clipped to keep the base calls in only one read of the pair.

4 Convert BAM file to FASTQ file (Picard)

5 Align reads in FASTQ file to create SAM file (BWA)

6 Merge data from unaligned and aligned BAM files (Picard)

7 Group reads in BAM file by UMI (fgbio)

8 Call consensus sequence for each set of UMI-grouped reads (fgbio)

9 Convert BAM file to FASTQ file (Picard)

10 Align UMI consensus reads in FASTQ file to create SAM file (BWA)

11 Merge data from unaligned and aligned BAM files (Picard)

12 Filter merged reads in BAM file based on quality (fgbio)

13 Perform read clipping in BAM file to remove overlapping bases in R1 and R2 reads (fgbio)

Part 3. Double-stranded UMI (duplex) processing and alignment (steps 14–19)

Part 3 of the pipeline is optional. It is intended for users who want to perform double-stranded UMI-based (i.e., "duplex") deduplication. For users who only require single-stranded UMI-based deduplication, proceed to part 4. The steps in part 3 are similar to steps 8 to 13 in part 2.

14 Call duplex consensus sequence (fgbio)

15 Convert BAM file to FASTQ file (Picard)

16 Align UMI consensus reads to create SAM file (BWA)

17 Merge data from unaligned and aligned BAM files (Picard)

18 Filter merged reads in BAM file based on quality (fgbio)

19 Perform read clipping in BAM file to remove overlapping bases in R1 and R2 reads (fgbio)

Part 4. Variant analysis (steps 20–23)

Part 4 is for identifying variants in the sample. Select which steps in this part are appropriate for your DNA panel and assay needs. You can perform variant calling on the deduplicated reads in any one of the following output files: 1) BAM file generated in part 2 with single-stranded UMI-based deduplication; 2) BAM file generated in part 3 with double-stranded UMI-based deduplication (duplex); 3) BAM file generated using positional deduplication. Positional deduplication is not covered in this technical guide, but resources for performing that type of deduplication are readily available. If desired, you can combine the variant call results from these different methods of deduplication for a hybrid mode of variant analysis. Note that the shell script examples shown in the next chapter use an output file generated with single-stranded UMI-based deduplication.

20 Perform variant calling for SNVs and indels (VarDict)

21 Annotate SNV and indel calls (snpSift, snpEff)

22 Perform CNV calling (PureCN)

23 Perform translocation calling (GeneFuse)

Part 5. Alignment metrics and QC (steps 20)

Calculate QC metrics on alignment and sample quality for comparison with user-defined pass /fail criteria. The QC analyses use output files from parts 1 through 4.

24 Perform QC for alignment and coverage (BEDTools, Picard, SAMtools)

3
Shell Script Examples

This chapter contains shell script examples for each step described in Chapter 2, “Avida Analysis Pipeline Steps.”

Command line examples for each of the parts and steps are provided. The examples are based on the following folder structures. If your computing node uses different folder structures, adjust the text in the examples accordingly.

- “/localdata" - the local storage mount where reference genome, software packages, input FASTQ files and output files are stored.
- “/localdata/references" - the directory under which the indexed reference genome in FASTQ format is stored.
- “/localdata/tools" - the directory where all software packages such as SAMtools, BEDTools etc. are installed.
- "$sample_dn" - the directory that contains the input FASTQ files and the output results are to be stored for the sample. For example, sample_dn="/localdata/analyses/run1/sample1".

Part 1. Read pre-processing (steps 1–3)

1 Convert FASTQ files to BAM file (Picard)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar picard.jar FastqToSam \
  FASTQ=$sample_dn/R1.fq FASTQ2=$sample_dn/R2.fq \
  OUTPUT=$sample_dn/raw_unmapped.bam \
  READ_GROUP_NAME=$sid SAMPLE_NAME=$sid \
  LIBRARY_NAME=pe PLATFORM=illumina PLATFORM_UNIT=HiSeq \
  MAX_RECORDS_IN_RAM=1000000
```

2 Extract UMIs from BAM file (fgbio)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar fgbio.jar ExtractUmisFromBam \
  --input=$sample_dn/raw_unmapped.bam \
  --output=$sample_dn/raw_unmapped_umi.bam \
  --read-structure=3M2S146T 3M2S146T --molecular-index-tags=ZA ZB --single-tag=RX
```

Note: The script in step 2 is hard coded for 151 bp sequencing, but it also works for runs with shorter sequencing.

3 Mark Illumina adaptors in BAM file (Picard)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar picard.jar MarkIlluminaAdapters \
  INPUT=$sample_dn/raw_unmapped_umi.bam \
  OUTPUT=$sample_dn/raw_unmapped_markedAdpt.bam \
  METRICS=$sample_dn/readstats/adapter_metrics.txt MAX_RECORDS_IN_RAM=1000000
```

Part 2. Single-stranded UMI processing and alignment (steps 4–13)

4 Convert BAM file to FASTQ file (Picard)

```
java -Xmx16G -jar picard.jar SamToFastq \
  INPUT=$sample_dn/raw_unmapped_markedAdpt.bam \
  FASTQ=$sample_dn/raw_unmapped_markedAdpt.fq \
  MAX_RECORDS_IN_RAM=1000000 CLIPPING_MIN_LENGTH=36 \
  INTERLEAVE=true INCLUDE_NON_PF_READS=true \
  CLIPPING_ATTRIBUTE=XT CLIPPING_ACTION=X
```

5 Align reads in FASTQ file to create SAM file (BWA)

```
bwa mem -p -t 10 \
  /localdata/references/hg19/Sequence/bwaIndex/genome.fa \
  $sample_dn/raw_unmapped_markedAdpt.fq > $sample_dn/raw_aligned.sam
```

6 Merge data from unaligned and aligned BAM files (Picard)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar picard.jar MergeBamAlignment \
  UNMAPPED=$sample_dn/raw_unmapped_markedAdpt.bam \
  ALIGNED=$sample_dn/raw_aligned.sam \
  OUTPUT=$sample_dn/picard_fixed.bam \
  REFERENCE_SEQUENCE=/localdata/references/hg19/Sequence/genome.fa \
  CLIP_ADAPTERS=false VALIDATION_STRINGENCY=SILENT \
  CREATE_INDEX=true EXPECTED_ORIENTATIONS=FR MAX_GAPS=-1 \
  SORT_ORDER=coordinate ALIGNER_PROPER_PAIR_FLAGS=false \
  MAX_RECORDS_IN_RAM=1000000
```

7 Group reads in BAM file by UMI (fgbio)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar fgbio.jar GroupReadsByUmi \
  --input=$sample_dn/picard_fixed.bam \
  --output=$sample_dn/grouped_umi.bam \
  --strategy=paired --raw-tag=RX --assign-tag=MI --min-map-q=10 --edits=1
```

8 Call consensus sequence for each set of UMI-grouped reads (fgbio)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar fgbio.jar CallMolecularConsensusReads \
  --input=$sample_dn/grouped_umi.bam \
  --output=$sample_dn/ss_consensus_unmapped.bam \
  --rejects=$sample_dn/ss_rejects.bam \
  --error-rate-pre-umi=45 --error-rate-post-umi=40 \
  --min-input-base-quality=10 --min-reads=1
```

9 Convert BAM file to FASTQ file (Picard)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar picard.jar SamToFastq \
  INPUT=$sample_dn/ss_consensus_unmapped.bam \
  FASTQ=$sample_dn/ss_consensus_unmapped.fastq \
  INTERLEAVE=true INCLUDE_NON_PF_READS=true MAX_RECORDS_IN_RAM=1000000
```

10 Align UMI consensus reads in FASTQ file to create SAM file (BWA)

```
/localdata/tools/bwa.kit/bwa mem -p -t 10 \
  /localdata/references/hg19/Sequence/bwaIndex/genome.fa \
  $sample_dn/ss_consensus_unmapped.fastq \
  > $sample_dn/ss_consensus_mapped.sam
```

11 Merge data from unaligned and aligned BAM files (Picard)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar picard.jar MergeBamAlignment \
  UNMAPPED=$sample_dn/ss_consensus_unmapped.bam \
  ALIGNED=$sample_dn/ss_consensus_mapped.sam \
  OUTPUT=$sample_dn/ss_consensus_mapped.bam \
  CLIP_ADAPTERS=false VALIDATION_STRINGENCY=SILENT \
  CREATE_INDEX=true ORIENTATIONS=FR MAX_GAPS=-1 \
  SORT_ORDER=coordinate ALIGNER_PROPER_PAIR_FLAGS=false \
  ATTRIBUTES_TO_RETAIN=X ATTRIBUTES_TO_RETAIN=ZS \
  ATTRIBUTES_TO_RETAIN=ZI ATTRIBUTES_TO_RETAIN=ZM \
  ATTRIBUTES_TO_RETAIN=ZC ATTRIBUTES_TO_RETAIN=ZN \
  ATTRIBUTES_TO_REVERSE=cd ATTRIBUTES_TO_REVERSE=ce \
  REFERENCE_SEQUENCE=/localdata/references/hg19/Sequence/genome.fa \
  MAX_RECORDS_IN_RAM=1000000
```

12 Filter merged reads in BAM file based on quality (fgbio)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar fgbio.jar FilterConsensusReads \
  --input=$sample_dn/ss_consensus_mapped.bam \
  --output=$sample_dn/ss_consensus_filtered.bam \
  --ref=/localdata/references/hg19/Sequence/genome.fa \
  --max-read-error-rate=0.05 --max-base-error-rate=0.1 \
  --min-base-quality=10 --max-no-call-fraction=0.2 --min-reads=3
```

13 Perform read clipping in BAM file to remove overlapping bases in R1 and R2 reads (fgbio)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar fgbio.jar ClipBam \
  --input=$sample_dn/ss_consensus_filtered.bam \
  --output=$sample_dn/ss_consensus_filtered_clipped.bam \
  --ref=/localdata/references/hg19/Sequence/genome.fa \
  --soft-clip=false --clip-overlapping-reads=true
```

Part 3. Double-stranded UMI (duplex) processing and alignment (steps 14–19)

14 Call duplex consensus (fgbio)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar fgbio.jar CallDuplexConsensusReads \
  --input=$sample_dn/grouped_umi.bam \
  --output=$sample_dn/ds_consensus_unmapped.bam \
  --error-rate-pre-umi=45 --error-rate-post-umi=40 --min-input-base-quality=10
```

15 Convert BAM file to FASTQ file (Picard)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar picard.jar SamToFastq \
  INPUT=$sample_dn/ds_consensus_unmapped.bam \
  FASTQ=$sample_dn/ds_consensus_unmapped.fastq \
  INTERLEAVE=true INCLUDE_NON_PF_READS=true \
  MAX_RECORDS_IN_RAM=1000000
```

16 Align UMI consensus reads to create SAM file (BWA)

```
bwa mem -p -t 10 /localdata/references/hg19/Sequence/bwaIndex/genome.fa \
  $sample_dn/ds_consensus_unmapped.fastq > $sample_dn/ds_consensus_mapped.sam
```

17 Merge data from unaligned and aligned BAM files (Picard)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar picard.jar MergeBamAlignment \
  UNMAPPED=$sample_dn/ds_consensus_unmapped.bam \
  ALIGNED=$sample_dn/ds_consensus_mapped.sam \
  OUTPUT=$sample_dn/ds_consensus_mapped.bam \
  CLIP_ADAPTERS=false VALIDATION_STRINGENCY=SILENT \
  CREATE_INDEX=true ORIENTATIONS=FR MAX_GAPS=-1 \
  SORT_ORDER=coordinate ALIGNER_PROPER_PAIR_FLAGS=false \
  ATTRIBUTES_TO_RETAIN=X0 ATTRIBUTES_TO_RETAIN=ZS \
  ATTRIBUTES_TO_RETAIN=ZI ATTRIBUTES_TO_RETAIN=ZM \
  ATTRIBUTES_TO_RETAIN=ZC ATTRIBUTES_TO_RETAIN=ZN \
  ATTRIBUTES_TO_REVERSE=cd ATTRIBUTES_TO_REVERSE=ce \
  REFERENCE_SEQUENCE=/localdata/references/hg19/Sequence/genome.fa \
  MAX_RECORDS_IN_RAM=1000000
```

18 Filter merged reads in BAM file based on quality (fgbio)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar fgbio.jar FilterConsensusReads \
  --input=$sample_dn/ds_consensus_mapped.bam \
  --output=$sample_dn/ds_consensus_filtered.bam \
  --ref=/localdata/references/hg19/Sequence/genome.fa \
  --max-read-error-rate=0.2 --max-base-error-rate=0.4 \
  --min-base-quality=30 --max-no-call-fraction=0.4 --min-reads=2 1 1
```

19 Perform read clipping in BAM file to remove overlapping bases in R1 and R2 reads (fgbio)

```
java -Djava.io.tmpdir=$sample_dn -Xmx16G -jar fgbio.jar ClipBam \
  --input=$sample_dn/ds_consensus_filtered.bam \
  --output=$sample_dn/ds_consensus_filtered_clipped.bam \
  --ref=/localdata/references/hg19/Sequence/genome.fa \
  --soft-clip=false --clip-overlapping-reads=true
```

Part 4. Variant analysis (steps 20–23)

The following output files are suitable for variant analysis: 1) BAM file generated in part 2 with single-stranded UMI-based deduplication; 2) BAM file generated in part 3 with double-stranded UMI-based deduplication (duplex); 3) BAM file generated using positional deduplication (not covered in this technical note). The variant call results from these different methods of deduplication can be combined for a hybrid mode of variant analysis. The example shown below uses an output file generated with method 1. You can annotate the SNV and indel calls to add gene symbols, amino acid changes, transcript IDs, cDNA changes, and functional effects (e.g., protein coding, intronic, etc.) for further analysis. The annotated variants can be further filtered based on the annotation or compared with a population database such as dbSNP, ExAC, COSMIC, and others. For simplicity, an example script for filtering of annotated variants is not included here.

20 Perform variant calling for SNVs and indels (VarDict)

```
VarDict -G /localdata/references/hg19/Sequence/genome.fa \
  -N $sid -b $sample_dn/ss_consensus_filtered_clipped.bam \
  -f 0.001 -c 1 -S 2 -E 3 -g 4 -r 3 -F 0x700 -th 12 \
  $sample_dn/devel401_designed_regions.bed  | \
  /localdata/tools/VarDict/teststrandbias.R | \
  /localdata/tools/VarDict/var2vcf_valid.pl -N $sid -E -f 0.001 \
  > $sample_dn/vd.vcf
```

21 Annotate SNV and indel calls (snpSift, snpEff)

```
java -Xmx16G -jar /localdata/tools/snpEff/SnpSift.jar annotate \
  -dbsnp $sample_dn/vd_roi.vcf > $sample_dn/snpsift.vd.vcf

java -Xmx16G -jar /localdata/tools/snpEff/snpEff.jar \
  -config /localdata/tools/snpEff/snpEff.config \
  -noStats -noLog -nodownload hg19 \
  $sample_dn/snpsift.vd.vcf > $sample_dn/snpeff.vd.vcf
```

22 Perform CNV calling (PureCN)

Use the BAM files generated during single-stranded UMI-based deduplication (Part 2 above) and either the covered regions file (Covered.bed) or target regions file (Regions.bed) available from Agilent SureDesign. The Regions.bed file is used in the example script below. For robust CNV calling, replicates of the reference are required. The example below is based on one sample and one reference (Sample1.bam, Normal1.bam, and Normal2.bam). Best practice guidelines, with example scripts, are available at bioconductor.org. Agilent has tested these guidelines with the following minimal modifications for the IntervalFile.R.

- The *--off-target* option is omitted to minimize noise
- The *--average-target-width* and *--min-target-width* options are set to 100 and 50, respectively

```
# Prepare reference files. Could be done once per design and set of normal samples
Rscript /localdata/tools/IntervalFile.R \
  --in-file /localdata/Regions.bed \
  --fasta /localdata/hg19.fa \
  --out-file /localdata/baits_hg19_intervals.txt \
  --genome hg19 \
  --mappability /localdata/wgEncodeCrgMapabilityAlign100mer.bigWig \
  --average-target-width 100 \
  --min-target-width 50

# Prepare a file "normals.list" to include Normal1.bam and Normal2.bam
echo "/localdata/Normal1.bam" > /localdata/normals.list
echo "/localdata/Normal2.bam" >> /localdata/normals.list

# Calculate GC-normalized coverage
mkdir /localdata/normals

Rscript /localdata/tools/Coverage.R \
  --bam /localdata/normals.list \
  --out-dir /localdata/normals \
  --intervals /localdata/baits_hg19_intervals.txt \
  --cores 4

# Do variant calling for normal files
gatk Mutect2 \
  -R /localdata/hg19.fa \
  -I /localdata/Normal1.bam \
  -O /localdata/Normal1_mutect2.vcf.gz \
  --intervals /localdata/Regions.bed \
  --genotype-germline-sites true \
  --genotype-pon-sites true \
  --germline-resource /localdata/af-only-gnomad.vcf.gz \
  --interval-padding 200 \
  --native-pair-hmm-threads 8
```

Script for step 22 continues on following page

```
gatk Mutect2 \
  -R /localdata/hg19.fa \
  -I /localdata/Normal2.bam \
  -O /localdata/Normal2_mutect2.vcf.gz \
  --intervals /localdata/Regions.bed \
  --genotype-germline-sites true \
  --genotype-pon-sites true \
  --germline-resource /localdata/af-only-gnomad.vcf.gz \
  --interval-padding 200 \
  --native-pair-hmm-threads 8

gatk FilterMutectCalls \
  -V /localdata/Normal1_mutect2.vcf.gz \
  -R /localdata/hg19.fa \
  -stats /localdata/Normal1_mutect2.vcf.gz.stats \
  -filtering-stats /localdata/Normal1_mutect2.filtered.stats \
  -O /localdata/Normal1_mutect2.filtered.vcf.gz \
  -f-score-beta 0.1

gatk FilterMutectCalls \
  -V /localdata/Normal2_mutect2.vcf.gz \
  -R /localdata/hg19.fa \
  -stats /localdata/Normal2_mutect2.vcf.gz.stats \
  -filtering-stats /localdata/Normal2_mutect2.filtered.stats \
  -O /localdata/Normal2_mutect2.filtered.vcf.gz \
  -f-score-beta 0.1

# Merge normal samples VCF files
ls Normal*.filtered.vcf.gz > normal_vcfs.list

bcftools merge \
  -l normal_vcfs.list \
  -Oz \
  -o /localdata/panel_of_normals.vcf.gz \
  --force-samples

bcftools index \
  --tbi \
  /localdata/panel_of_normals.vcf.gz

# Build a normal database
ls -a /localdata/normals/*_loess.txt.gz | cat > /localdata/normal_coverages.list

Rscript /localdata/tools/NormalDB.R \
  --out-dir /localdata/ \
  --coverage-files normal_coverages.list \
  --normal-panel /localdata/panel_of_normals.vcf.gz \
  --genome hg19 \
  --assay avida_expanded

# Process sample
$SAMPLEID="Sample1"
mkdir /localdata/$SAMPLEID
```

Script for step 22 continues on following page

```
# Calculate GC-normalized coverage
Rscript /localdata/tools/Coverage.R \
  --out-dir /localdata/${SAMPLEID} \
  --bam /localdata/${SAMPLEID}.bam \
  --intervals /localdata/baits_hg19_intervals.txt

# Generate a VCF for each sample using Mutect2
gatk Mutect2 \
  -R /localdata/hg19.fa \
  -I /localdata/${SAMPLEID}.bam \
  -O /localdata/${SAMPLEID}_mutect2.vcf.gz \
  --intervals /localdata/Regions.bed \
  --genotype-germline-sites true \
  --genotype-pon-sites true \
  --germline-resource /localdata/af-only-gnomad.vcf.gz \
  --interval-padding 200 \
  --native-pair-hmm-threads 8

gatk FilterMutectCalls \
  -V /localdata/${SAMPLEID}_mutect2.vcf.gz \
  -R /localdata/hg19.fa \
  -stats /localdata/${SAMPLEID}_mutect2.vcf.gz.stats \
  -filtering-stats /localdata/${SAMPLEID}_mutect2.filtered.stats \
  -O /localdata/${SAMPLEID}_mutect2.filtered.vcf.gz \
  -f-score-beta 0.1

# Run PureCN to normalize, segment and determine purity and ploidy
Rscript /localdata/tools/PureCN.R \
  --out /localdata/${SAMPLEID} \
  --tumor $OUT/${SAMPLEID}/${SAMPLEID}_coverage_loess.txt.gz \
  --sampleid ${SAMPLEID} \
  --vcf /localdata/${SAMPLEID}_mutect2.filtered.vcf.gz \
  --normaldb /localdata/normalDB_avida_expanded_hg19.rds \
  --mapping-bias-file /localdata/mapping_bias_avida_expanded_hg19.rds \
  --intervals /localdata/baits_hg19_intervals.txt \
  --genome hg19 \
  --interval-padding 200 \
  --post-optimize
```

23 Perform translocation calling (GeneFuse)

```
# cut adapter
cutadapt -m 30 -q 15 \
  -a AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC \
  -A AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT \
  -o $sample_dn/r1_cutadapt.fq  -p $sample_dn/r2_cutadapt.fq \
  $sample_dn/R1.fq $sample_dn/R2.fq > $sample_dn/readstats/cutadapt.log

# call translocation
genefuse -u 1 -t 30 \
  -r /localdata/references/hg19/Sequence/genome.fa \
  -f /localdata/tools/genefuse-0.6.1/genes/cancer.hg19.csv \
  -1 $sample_dn/r1_cutadapt.fq. -2 $sample_dn/r2_cutadapt.fq \
  -h $sample_dn/genefuse_report.html
```

Part 5. Alignment metrics and QC (step 24)

Alignment and sample QC metrics are generated and collected to be compared with preset sample QC passing /failing criteria.

24 Perform QC for alignment & coverage (BEDTools, Picard, SAMtools)

a Generate chromosome lengths file:

```
# chrom lengths
samtools view -H $sample_dn/picard_fixed.bam \
 | grep @SQ | sed 's/@SQ\tSN:\|LN://g' | cut -f1,2 > $sample_dn/genome_length.tsv
```

b Collect unique coverage of every base in the panel. The same can be done for the raw BAM file and duplex deduplicated files.

```
bedtools coverage -sorted -a [panel_target_regions].bed \
 -b $sample_dn/ss_consensus_mapped.bam \
 -d -g $sample_dn/genome_length.tsv > $sample_dn/ss_base.cover.tsv
```

c SAMtools flagstat

```
samtools flagstat $sample_dn/picard_fixed.bam > $sample_dn/flagstats.txt
```

d Collect insert size

```
java -Xmx16G -jar picard.jar CollectInsertSizeMetrics \
  INPUT=$sample_dn/ss_consensus_mapped.bam \
  OUTPUT=$sample_dn/insert_metrics.tsv \
  HISTOGRAM_FILE=$sample_dn/insert_histo.pdf \
  HISTOGRAM_WIDTH=400 MINIMUM_PCT=0.01
```

www.agilent.com

In This Book

This technical guide provides an analysis pipeline for Illumina NGS sequencing files generated from libraries prepared with the Avida DNA or Avida Duo Methyl reagents.

Version B0, September 2024

G9409-90001

Agilent