Processing Sequencing Data Utilizing Twist Unique Molecular Identifier (UMI) Adapter System

INTRODUCTION

This document is a recommended guideline for processing NGS data generated from libraries constructed with Twist's Unique Molecular Identifier (UMI) adapter system. The bioinformatic steps outlined are based on Twist's internal troubleshooting pipeline and begin with FastQ files from an Illumina sequencing platform that have been demultiplexed upstream. Certain input files, such as the reference genome build and command parameters, should be modified to meet assay-specific validation criteria.

This document details how to extract the UMI sequences from the sequencing reads, prepare BAM files with appropriate metadata, perform alignment, group reads by UMI, and call duplex consensus reads. The tools used in this workflow are open source and can be used to process UMI data. Several other tools are also available, which can be used to call duplex consensus reads following similar steps.

PACKAGES AND VERSIONS

bwa	version: 0.7.17
samtools	version: 1.15.1
fgbio	version: 1.5.1
gatk	version: 4.2.6.1
picard	version: 2.27.1

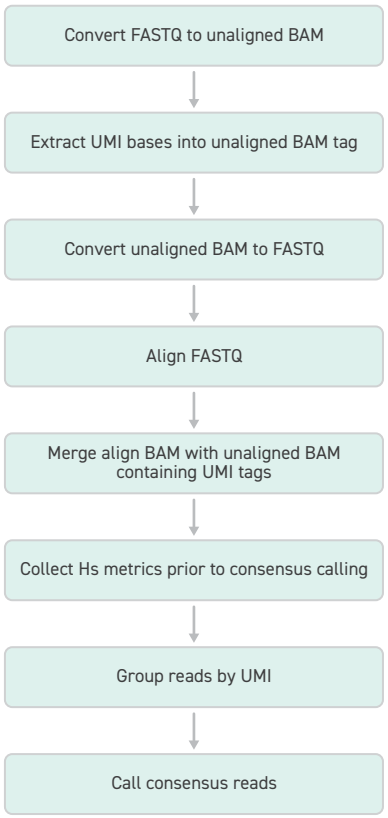
**Note: Twist recommends pinning fgbio version 1.5.1 when implementing this workflow to avoid changes in more recent updates which impact the "Merge aligned BAM with unaligned BAM containing UMI tags" step.*

Sample datasets are available by request through Twist's Customer Support team to assist with verifying successful analysis implementation. They include:

- Raw FastQ files
- Aligned BAM files prior to consensus collapse (aligned with bwa mem)
- Aligned consensus BAM files (aligned with bwa mem)
- Picard HsMetrics
- Target (target_file_UMI_demo_data_hg38.bed) and probe (probe_file_UMI_demo_data_hg38.bed) bed files for the Twist capture panel used

OUTLINE

The bioinformatic processing steps presented here begin with FastQ files generated after demultiplexing and finish with duplex consensus BAM files. These individual steps can be combined into a single pipeline for analyzing sequencing data in a high throughput setting. Additionally, the number of cores/heap memory space for each task can be set by the user based on the machine being used and the size of the target data.



TWIST RECOMMENDED PIPELINE LOGIC

1. Convert FastQ to unaligned BAM

fgbio processes sequencing data on BAM files, which have expanded functionality to store metadata in BAM tags. The original FastQ is converted to a BAM that stores the sequence, read name and quality score information, but not any alignment information.

```
java -Xmx4g -jar /path/to/picard.jar FastqToSam \
   O={unaligned_bam} \
   F1={sample_1_r1.fastq} \
   F2={sample_1_r2.fastq} \
   SM={sample_name} \
   LB=Library1 \
   PU=Unit1 \
   PL=Illumina
```

2. Extract UMI bases into unaligned BAM tag

This step utilizes fgbio's ExtractUMIsFromBam function which extracts the UMI bases from the sequencing reads. The UMI bases are stored in three tags: ZA for the 5' UMI, ZB for the 3' UMI, and RX for a string containing both the 5' and 3' UMIs separated by a dash. Twist UMI adapters are 5 base pairs with a 2 base pair skip, resulting in the read structure 5M2S+T.

```
java -Xmx4g -jar /path/to/fgbio.jar ExtractUmisFromBam --input={unaligned_bam} \
   --output={unaligned_bam_umi_extracted}\
   --read-structure=5M2S+T 5M2S+T \
   --molecular-index-tags=ZA ZB \
   --single-tag=RX
```

3. Convert unaligned BAM to FastQ

After the UMI bases have been extracted from the sequencing reads and stored in tags in the unaligned BAM file, we convert the unaligned BAM to FastQ to align the reads.

```
java -Xmx4g -jar /path/to/picard.jar SamToFastq \
   I={unaligned_bam_umi_extracted} \
   F={fastq_umi_extracted} \
   INTERLEAVE=true
```

4. Align FastQ

This step aligns the UMI extracted FastQ files to the reference genome using bwa.

```
bwa mem -p -t 8 {local_reference_fa_path} {fastq_umi_extracted} | samtools sort -@ 8 -o
{aligned_bam_umi_extracted}
```

5. Merge aligned BAM with unaligned BAM containing UMI tags

To prepare the BAM files for grouping by UMI using fgbio, the aligned BAM file from the previous step must be merged with the unaligned BAM file that contains the ZA, ZB, and RX tags from step 2. To achieve this, picard's MergeBamAlignment command is utilized, resulting in BAM files with sequences aligned to the reference, sequencing reads with the UMI bases extracted, and necessary UMI tags in the header.

```
java -Xmx4g -jar /path/to/picard.jar MergeBamAlignment \
   UNMAPPED={unaligned_bam_umi_extracted} \
   ALIGNED={aligned_bam_umi_extracted} \
   O={aligned_tag_umi_bam} \
   R={local_reference_fasta_path} \
   CLIP_ADAPTERS=false \
   VALIDATION_STRINGENCY=SILENT \
   CREATE_INDEX=true \
   EXPECTED_ORIENTATIONS=FR \
   MAX_GAPS=-1 \
   SO=coordinate \
   ALIGNER_PROPER_PAIR_FLAGS=false
```

6. Optional: Collect Hs metrics prior to consensus calling

Prior to grouping reads by UMI, collecting HsMetrics on the primary alignments can provide important performance metrics such as fold 80, coverage, and off target rates. The target and probe interval files in the following code uses the Twist customer facing panel as an example, but any bait and target bed files from a target enrichment panel can be used.

```
java -Xmx4g -jar /path/to/picard.jar CollectHsMetrics -I {aligned_tag_umi_bam} \
   -O {metrics_file} \
   -R {local_reference_fasta_path} \
   --BAIT_INTERVALS {probes_intervals} \
   --TARGET_INTERVALS target_file_UMI_demo_data_hg38.bed \
   --PER_TARGET_COVERAGE probe_file_UMI_demo_data_hg38.bed
```

7. Group reads by UMI

This step groups reads together that appear to have come from the same original molecule. Reads are grouped by template, and then templates are sorted by the 5' mapping positions of the reads from the template, used from earliest mapping position to latest. Reads that have the same end positions are then sub-grouped by UMI sequence.

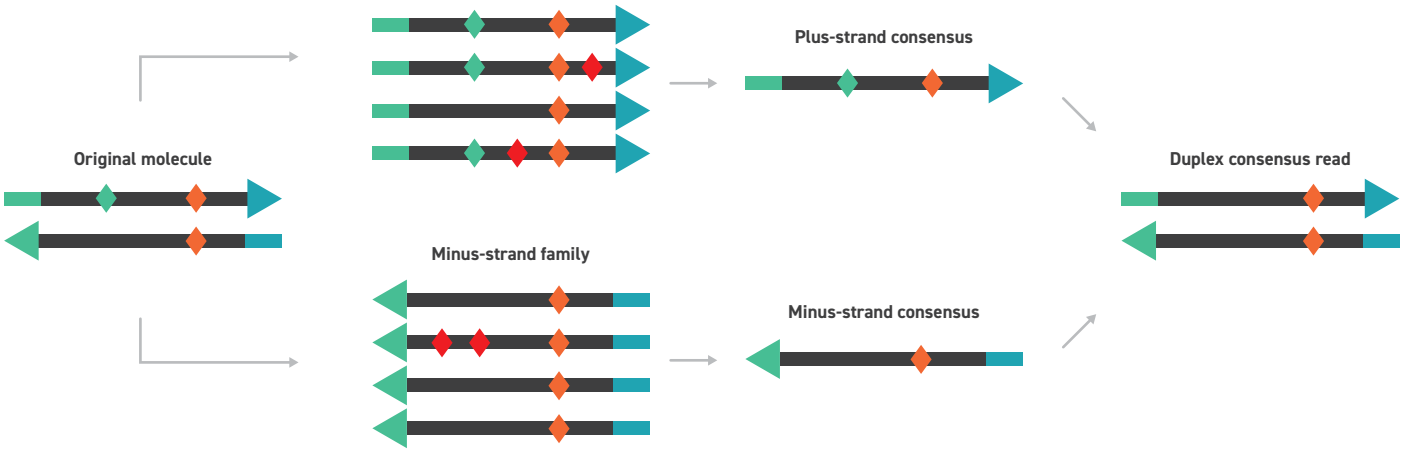
There are 4 strategies available from fgbio to group reads. Twist utilizes CallDuplexConsensusReads in the subsequent step, which is only compatible with the "paired" strategy. CallDuplexConsensusReads error corrects UMI pairs considering duplex information (i.e. takes into account both 5' and 3' UMIs during error correction).

```
java -Xmx4g -jar /path/to/fgbio.jar GroupReadsByUmi
   --strategy={paired} \
   --input={aligned_tag_umi_bam} \
   --output={grouped_by_umi_bam} \
   --raw-tag=RX \
   --min-map-q=10 \
   --edits=1
```

8. Call consensus reads

Reads from the same unique molecule are first partitioned by source strand and assembled into single strand consensus molecules as described by fgbio CallMolecularConsensusReads. Subsequently, for molecules that have at least one observation of each strand, duplex consensus reads are assembled by combining the evidence from the two single strand consensus reads. The BAM output file from this step will result in unaligned consensus duplex reads which can be converted back to FastQ and aligned to the reference for downstream processing.

The min-reads flag is used to apply stringency to the duplex collapse. The first integer is the duplex filter flag, which sets the minimum number of input reads to form a consensus read to 2. The second and third integers are the first and second strand filter flags respectively. Twist recommends setting the first and second strand filters to 1 to ensure that duplex consensus reads consist of at least 1 read from both the top and bottom strand.




```
java -Xmx4g -jar /path/to/fgbio.jar CallDuplexConsensusReads \
   --input={grouped_by_umi_bam} \
   --output={unaligned_consensus_bam} \
   --error-rate-pre-umi=45 \
   --error-rate-post-umi=30 \
   --min-input-base-quality=30 \
   --min-reads 2 1 1
```

9. Align duplex consensus reads

After consensus duplex collapse, reads will need to be converted back to FastQ for alignment to the genome using the same commands from steps 3 and 4:

```
java -Xmx4g -jar /path/to/picard.jar SamToFastq \
   I={unaligned_consensus_bam} \
   F={fastq_consensus} \
   INTERLEAVE=true
bwa mem -p -t 8 {local_reference_fa_path} -o {aligned_bam_consensus}
{fastq_consensus}
```

10. Merge unaligned consensus BAM and aligned consensus BAM to retain UMI tag metadata

Aligned consensus BAM files can be merged with the unaligned consensus BAM file to produce finalized aligned reads which contain the UMI tag information for additional downstream processing if desired. Subsequently, we utilize Picard's AddOrReplaceReadGroups to assign all the reads in the final BAM file to a single new read-group.

```
java -Xmx4g -jar /path/to/picard.jar MergeBamAlignment \
   VALIDATION_STRINGENCY=SILENT \
   UNMAPPED={unaligned_consensus_bam} \
   ALIGNED={aligned_bam_consensus} \
   O={merged_no_read_group_consensus_bam} \
   R={local_reference_fasta_path} \
   CLIP_ADAPTERS=false \
   VALIDATION_STRINGENCY=SILENT \
   CREATE_INDEX=true \
   EXPECTED_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_RETAIN=ad \
   ATTRIBUTES_TO_RETAIN=bd \
   ATTRIBUTES_TO_RETAIN=cd \
   ATTRIBUTES_TO_RETAIN=ae \
   ATTRIBUTES_TO_RETAIN=be \
   ATTRIBUTES_TO_RETAIN=ce

java -Xmx4g -jar /path/to/picard.jar AddOrReplaceReadGroups
I={merged_no_read_group_bam} \
   O={consensus_aligned_merged_bam} \
   RGID={sample_id} \
   RGLB={sample_lib} \
   RGPL=Illumina RGSM={sample_name} \
   RGPU=NA
```

10a. Optional: Collect HsMetrics preceding duplex consensus calling

Twist recommends assessing key performance metrics such as coverage, fold-80, and off bait after duplex consensus calling. This can be achieved using the Picard CollectHsMetrics package.

```
java -Xmx4g -jar /path/to/picard.jar CollectHsMetrics -I
{consensus_aligned_merged_bam} \
   -O {metrics_file} \
   -R {local_reference_fasta_path} \
   --BAIT_INTERVALS {probes_intervals} \
   --TARGET_INTERVALS target_file_UMI_demo_data_hg38.bed \
   --PER_TARGET_COVERAGE probe_file_UMI_demo_data_hg38.bed
```

ADDITIONAL BIOINFORMATIC RESOURCES: DNANEXUS

Twist has partnered with DNAnexus, a cloud computing and bioinformatics platform, to provide our customers with a solution to process sequencing data generated using Twist's Unique Molecular Identifier adapter system. DNAnexus has developed a bioinformatic pipeline for secondary analysis utilizing the steps outlined in this document, and tested using Twist example datasets, which can be used to process FastQ files in the event customers require additional support. To utilize the DNAnexus platform for bioinformatic processing of libraries with Twist's UMI adapter systems, please visit dnanexus.com and use the "Contact Us" field to set up a meeting.

DNAnexus®