WGS/WES QC Pipeline Documentation

Table of Contents

Overview

This pipeline performs quality control (QC) on whole-genome (WGS) or whole-exome (WES) sequencing data. It calculates per-chromosome coverage statistics and runs VerifyBamID2 to estimate sample contamination. The pipeline is designed to be run on a set of samples listed in a sample sheet, with BAM files either already present or downloaded from an S3 bucket.

Main steps:

1. **Prepare autosomal BED** – Creates a BED file covering autosomal chromosomes, excluding known gap regions (for WGS) or using exon/intron coordinates from a GTF file (for WES).
2. **Download BAMs (optional)** – Downloads BAM and index files from a configured S3 bucket.
3. **QC metrics calculation** – For each sample, iterates over autosomal regions, computes coverage per position, and writes per-chromosome JSON/TSV files.
4. **VerifyBamID2** – Runs VerifyBamID2 on each sample to estimate contamination.
5. **Combine results** – Merges per-chromosome data into sample-level summaries and integrates VerifyBamID2 output into a final TSV.

6. **Report generation** – An optional script that reads the combined QC JSON and MultiQC outputs to produce an interactive HTML report and multiple TSV summaries.

Prerequisites

Software & Tools

- **Python 3.8+** with the following packages:
    - pandas
    - numpy
    - pysam
    - plotly
    - tqdm
    - configparser
- **bedtools** (for WGS gap subtraction)
- **verifybamid2** (must be in PATH)
- **awscli** (if downloading BAMs from S3; configure with the appropriate profile)
- **Linux utilities**: zcat, cut, egrep, sort

Reference Files

- **FASTA reference** (with .fai index)
- **Gap file** (gap.txt.gz) – e.g., from UCSC
- **GTF file** (for WES mode) – e.g., GENCODE annotation
- **VerifyBamID2 resource files** – download from the VerifyBamID2 website (e.g., 1000g.phase3.100k.b38.vcf.gz.dat*)

Sample Sheet

A tab-separated file with at least the following columns:

- Library_ID – unique sample identifier
- Path – subdirectory under the base BAM path (see configuration) that leads to the recalibrated BAM folder.

Example:

Library_ID	Path
WHB13294	project_A/sample_WHB13294
WHB13306	project_A/sample_WHB13306

Installation & Setup

1. **Clone or copy** the pipeline scripts to a directory, e.g., /home/ubuntu/WGSQC_pipeline/.
2. **Create the required directory structure** (or let the pipeline create it on first run):

```
WGSQC_pipeline/
├── data/
│   ├── config.ini
│   ├── thresholds.ini          (optional)
│   ├── sample_sheet.tsv
│   └── resources/
│       └── verifybamid/         (VerifyBamID2 resource files)
├── shellScript/
│   ├── downloadBAM.sh
│   └── runVerifyBamID2.sh
├── run_pipeline.py
├── utils.py
├── qc_metrics.py
├── verify_bamid.py
└── generate_qc_report.py
```

3. **Install Python dependencies**:

pip install pandas numpy pysam plotly tqdm

4. **Verify that** verifybamid2**,** bedtools**, and** aws **are in your** PATH**.**

Configuration

All pipeline parameters are set in data/config.ini. Below is an example with explanations:

```
[DEFAULT]

pipeline_path = /home/ubuntu/WGSQC_pipeline


[PATHS]

workdir = /home/ubuntu/DATA_DRIVE/WGS_QC_Full_39

bamdir = /home/ubuntu/DATA_DRIVE/WGS_QC_Full_39/BAMs

refpath = /home/ubuntu/DATA_DRIVE_REF/references

bucketupload = enabl-wgs-processed.store.emulsion.sg

bucketdownload = enabl-wgs-alignment.store.emulsion.sg

name = WGS_QC

results = /home/ubuntu/DATA_DRIVE/WGS_QC_Full_39/results

work = /home/ubuntu/DATA_DRIVE/WGS_QC_Full_39/work

sampleinfo = /home/ubuntu/WGSQC_pipeline/data/WGS_QC_Full_39.tsv

references =
/home/ubuntu/DATA_DRIVE_REF/references/Homo_sapiens/GATK/GRCh38/Sequence/Wh
oleGenomeFasta/Homo_sapiens_assembly38.fasta

fai =
/home/ubuntu/DATA_DRIVE_REF/references/Homo_sapiens/GATK/GRCh38/Sequence/Wh
oleGenomeFasta/Homo_sapiens_assembly38.fasta.fai

interval =
/home/ubuntu/DATA_DRIVE_REF/references/Homo_sapiens/GATK/GRCh38/Annotation/
intervals/wholegenome.interval_list

gap = /home/ubuntu/DATA_DRIVE_REF/other_annotation_hg38/gap.txt.gz

gtf = /home/ubuntu/DATA_DRIVE_REF/references/gencode.v46.annotation.gtf   ;
required for WES

uploaddirname = WGS_ENABL_39_FULL_QC_RESULTS

runbatchname = Sample_WGS_QC_39

num_threads = 30

analysis_type = WGS                          ; WGS or WES

verifybamid_resource_path =
/home/ubuntu/WGSQC_pipeline/data/resources/verifybamid

bam_base_dir = /home/ubuntu/ENABL_AWS/enabl-wgs-alignment

tmp_dir = /home/ubuntu/DATA_DRIVE/tmp
```

```
num_processes = 8                          ; number of parallel chunks
```

Important notes:

- bam_base_dir is combined with Path from the sample sheet to form the full BAM directory.
  Example: bam_base_dir + / + Path + /results_Library_ID/preprocessing/recalibrated/Library_ID
- tmp_dir is set as TMPDIR for any subprocesses.
- num_processes controls how many sample chunks are processed simultaneously.

Sample Information Sheet

The sample sheet is a TSV file with columns:

- Library_ID – must match the BAM file name (Library_ID.recal.bam).
- Path – subdirectory under bam_base_dir.

The pipeline reads this file and constructs the full BAM path for each sample.

Example:

Library_ID	Path
WHB13294	project_A/sample_WHB13294
WHB13306	project_A/sample_WHB13306

Running the Pipeline

The main orchestrator is run_pipeline.py. It accepts several command-line options to run only specific steps.

Basic usage

python run_pipeline.py --config data/config.ini --steps download qc verify combine

This executes:

- download – download BAM files (if not already present)
- qc – calculate QC metrics
- verify – run VerifyBamID2
- combine – merge results and produce final TSV

If you omit --steps, the default is qc verify combine.

Options

- --config PATH : path to configuration file (default: data/config.ini relative to script)
- --steps STEP [STEP ...] : space-separated list of steps to run (download, qc, verify, combine)
- --sampleinfo PATH : override the sample sheet path given in config

Step details

- **download** : Uses downloadBAM.sh to fetch BAM and .bai from the S3 bucket. Skips if both files already exist.
- **qc** : Creates the autosomal BED (if missing), then splits samples into chunks and processes them in parallel. For each sample, per-chromosome JSON/TSV files are written to workdir/QC_metrics/Library_ID/chromosome_regions/.
- **verify** : Splits samples into chunks and runs runVerifyBamID2.sh in parallel. Outputs are written to workdir/res_VerifyBamID2/.
- **combine** : Reads all per-chromosome JSONs, merges them by sample, adds VerifyBamID2 data, and writes a final merged JSON and TSV in workdir/QC_metrics/.

Note: The qc and verify steps run **simultaneously** as separate processes when both are requested, exactly as in the original pipeline.

Outputs

After a successful run, the working directory (workdir) contains:

```
workdir/
├── QC_metrics/
|   ├── Library_ID_1/
|   |   ├── chromosome_regions/
|   |   |   ├── chr1.json
|   |   |   ├── chr1.tsv
|   |   |   ├── chr2.json
|   |   |   ├── chr2.tsv
|   |   |   └── ...
|   |   ├── log.txt
```

```
│   │   └── Library_ID_1_merged_all_chrom_qc_metrics.json
│   │   └── Library_ID_1_merged_all_chrom_qc_metrics.tsv
│   ├── Library_ID_2/
│   │   └── ...
│   └── QC_metricses_data_all_samples_all_chrom.json      (combined across samples)
│   └── QC_metricses_data_all_samples.tsv                 (final summary TSV)
├── res_VerifyBamID2/
│   ├── Library_ID_1.selfSM
│   ├── Library_ID_1.geno
│   ├── Library_ID_2.selfSM
│   └── ...
└── sorted_autosomes.bed
```

File descriptions

- **Per-chromosome JSON/TSV** – contain coverage metrics for one chromosome of one sample.
- log.txt – timing information for the sample.
- **Merged JSON/TSV** – per-sample aggregation of all chromosomes.
- QC_metricses_data_all_samples_all_chrom.json – data for all samples and chromosomes, used by the report generator.
- QC_metricses_data_all_samples.tsv – final sample-level summary with columns from the original combine_JSONs function.
- **VerifyBamID2 outputs** – standard output files; the .selfSM file contains the contamination estimate (FREEMIX).
- sorted_autosomes.bed – BED file used for coverage calculation.

Post-Processing Report

The script generate_qc_report.py produces an interactive HTML report and multiple TSV summaries from the pipeline outputs and MultiQC data.

Usage

```
python generate_qc_report.py \
  --qc-json /path/to/QC_metricses_data_all_samples_all_chrom.json \
  --multiqc-dir /path/to/multiqc_data \
  --outdir /path/to/report_output \
  --config data/thresholds.ini
```

Arguments

- --qc-json – the combined JSON file generated by the pipeline.
- --multiqc-dir – directory containing MultiQC outputs (e.g., multiqc_data from a MultiQC run). The script looks for files like mosdepth-cumcoverage-dist-id_1.txt, coverage-per-contig_1.txt, etc.
- --outdir – directory where all reports and the HTML file will be saved.
- --config – optional thresholds file (see below).

Outputs

In the specified output directory:

- qc_report.html – an interactive HTML file with all plots.
- Autosomal_Coverage_Samples_report.tsv
- Autosomal_Coverage_Chromosomes_report.tsv
- Samples_Contamination_report.tsv
- Cumulative_Coverage_Samples_report.tsv
- Coverage_per_contig_Samples_Chromosome_report.tsv
- Chromosome_coverage_imbalance_report.tsv
- Percent_mapped_reads_report.tsv
- insert_quality_report.tsv
- base_quality_report.tsv

These TSV files contain summary statistics and pass/fail statuses based on the thresholds.

Thresholds File (Optional)

You can supply a thresholds file (INI format) to customise the cut-offs used in the report. Example:

```
[THRESHOLDS]
autosomal_coverage_cutoff = 30
percent_coverage_cutoff = 90
mean_median_ratio_cutoff = 1.5
freemix_cutoff = 0.01
mapped_percent_cutoff = 95
base_quality_cutoff = 30
cv_cutoffs = 5,10,15,20,25,30
```

If no file is given, these defaults are used.

Troubleshooting

1. BAM files not found

- Verify that the bam_base_dir and Path column in the sample sheet correctly point to the BAM directory.
- Ensure the BAM file is named exactly Library_ID.recal.bam.

2. bedtools or verifybamid2 not found

- Add the installation directory to your PATH.
- Alternatively, provide full paths in the shell scripts.

3. Download step fails

- Check that awscli is configured with the correct profile (--profile enabl-wgs-alignment in downloadBAM.sh).
- Confirm that the S3 bucket and path are accessible.

4. Memory issues

- Reduce num_processes in the config.
- Ensure tmp_dir has enough free space.

5. Logs

Each sample writes a log.txt file with timings; check these for errors. Also examine the console output from the pipeline.

Example Commands

Full pipeline run (download + qc + verify + combine)

cd /home/ubuntu/WGSQC_pipeline

python run_pipeline.py --config data/config.ini --steps download qc verify combine

Only run QC and combine (if BAMs are already present)

python run_pipeline.py --config data/config.ini --steps qc combine

Generate report after MultiQC

python generate_qc_report.py \

--qc-json /home/ubuntu/DATA_DRIVE/WGS_QC_Full_39/QC_metrics/QC_metricses_data_all_samples_all_chrom.json \

--multiqc-dir /home/ubuntu/DATA_DRIVE/multiqc_data_319 \

--outdir /home/ubuntu/DATA_DRIVE/WGS_QC_Full_39/QC_Results \

--config data/thresholds.ini