Validating files at each step in a sequence processing pipeline

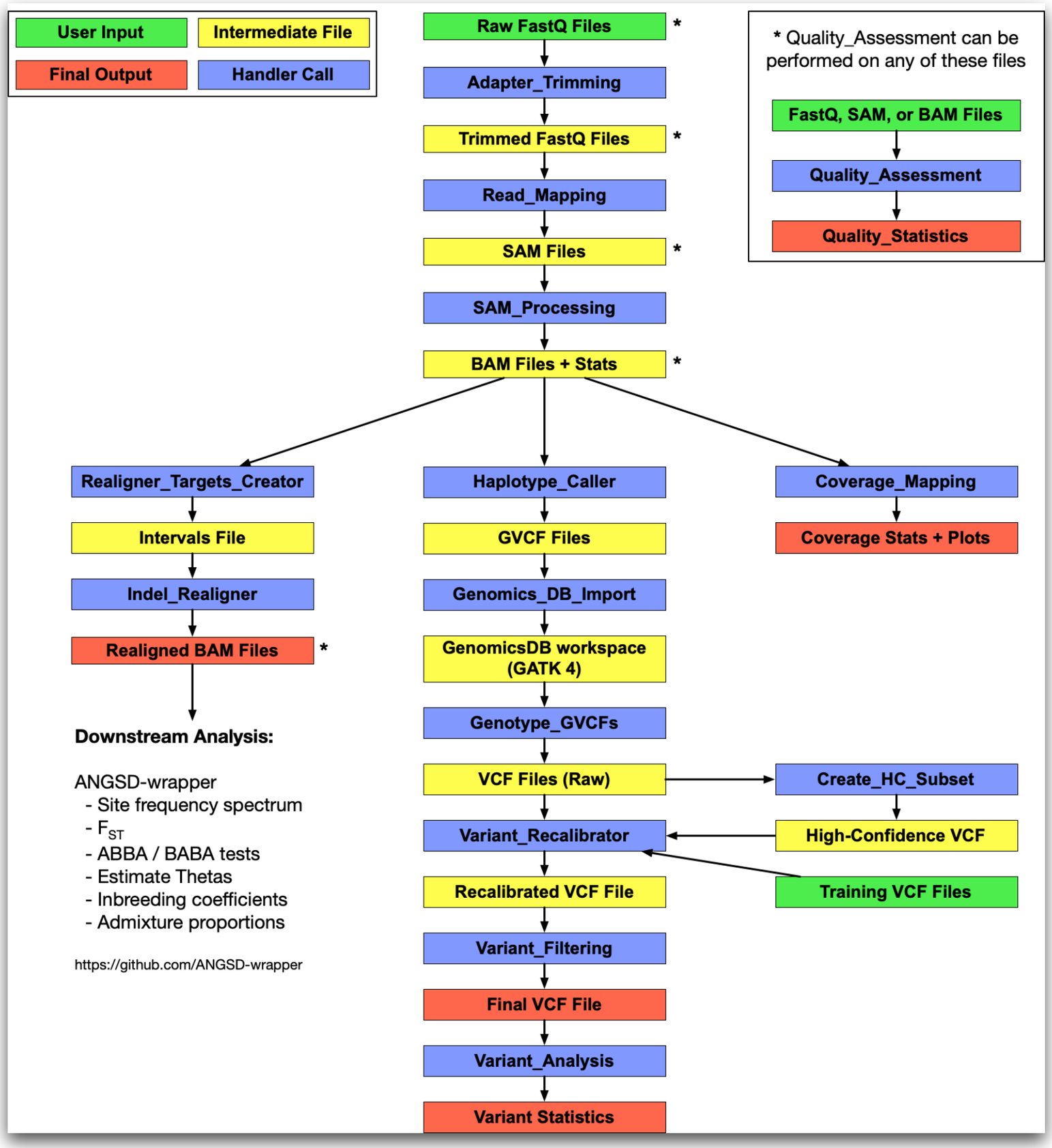
Does[0]Compute?

May 27, 2020

Overview

- Validating files scope
    - Sequence processing pipeline
    - sequence_handling as an example
- How do we know if our output files are usable?
- Common issues when processing sequence data
    - What are the checks built-in to sequence_handling that make our lives easier?
    - What are additional “sanity” checks we can perform?

sequence_handling pipeline

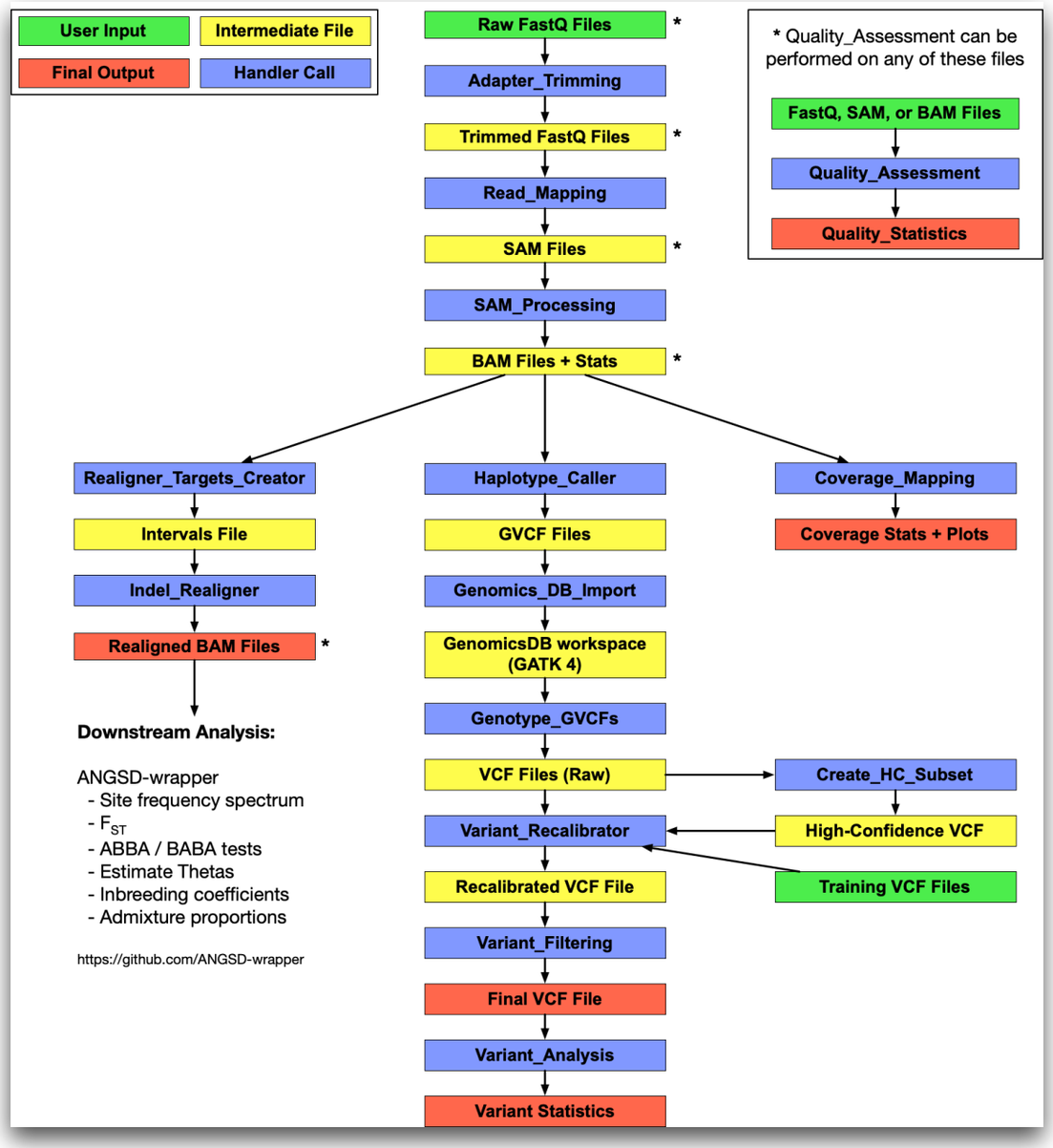


sequence_handling pipeline

All handlers automatically check for the following:

- Dependencies
- Samples and sample list exists
- If we are using a PBS job scheduler

Additional checks are handler specific (e.g., check for adapters file is specific to the Adapter_Trimming handler)



“Manual” checks for any step

- Quick sanity check for any unusual file sizes
  - **`ls -lhS`** lists files sorted by file size
    - Good first check for raw FASTQ files, adapter trimmed FASTQ files, SAM/BAM files, etc.

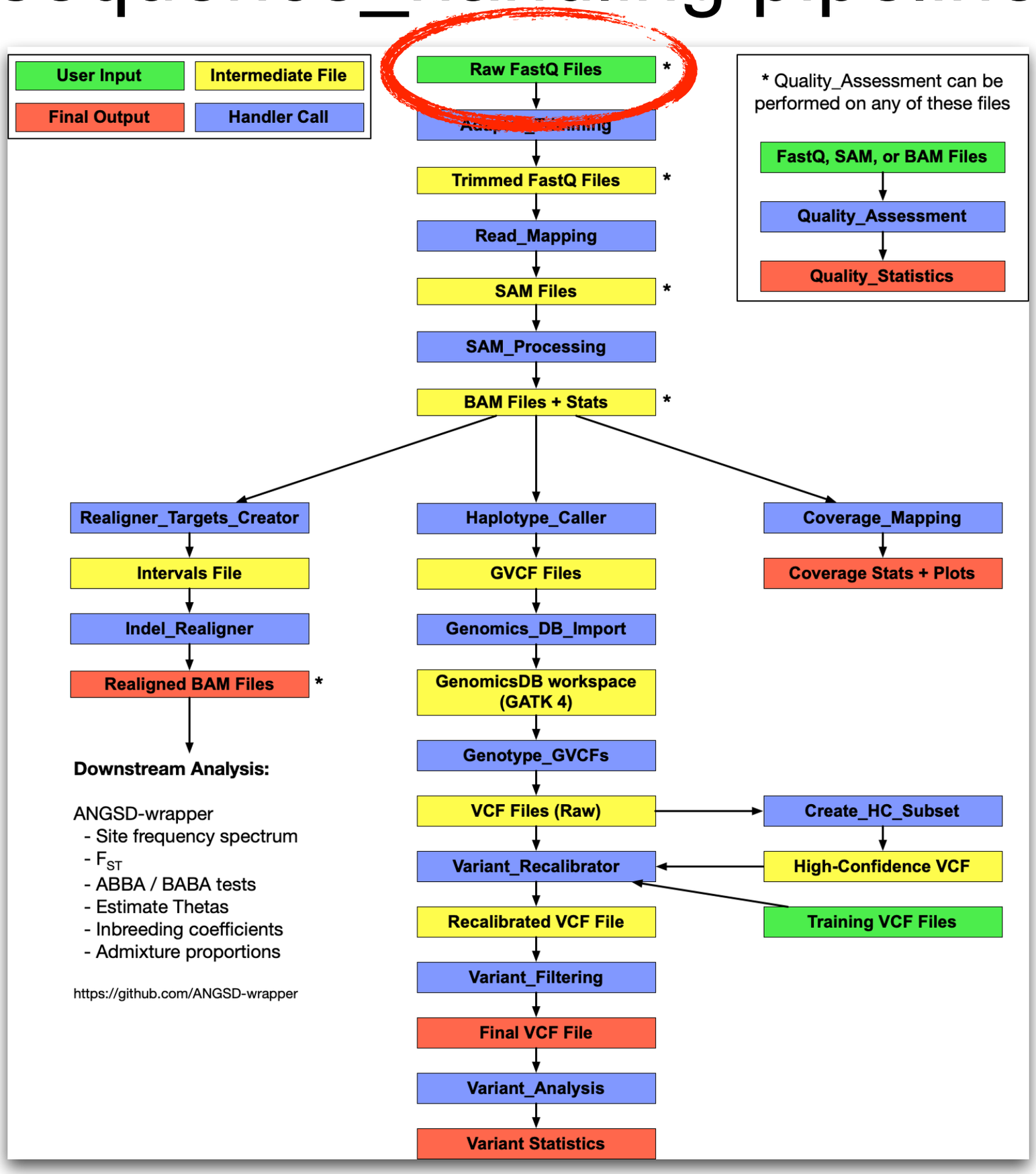
```
liux1299@ln0004:~/Shared/Datasets/NGS/Barley_Exome/WBDC_Inversion_Samples/100bp_concatenateastq $ ls -lhS *.fastq.gz
-rw-rwx---. 1 llei morrellp 2.5G Aug 21  2018 WBDC_005_R1.fastq.gz
-rw-rwx---. 1 llei morrellp 2.5G Aug 21  2018 WBDC_005_R2.fastq.gz
-rw-rwx---. 1 llei morrellp 2.5G Aug 21  2018 WBDC_004_R1.fastq.gz
-rw-rwx---. 1 llei morrellp 2.5G Aug 21  2018 WBDC_004_R2.fastq.gz
-rw-rwx---. 1 llei morrellp 2.5G Aug 21  2018 WBDC_020_R1.fastq.gz
-rw-rwx---. 1 llei morrellp 2.5G Aug 21  2018 WBDC_020_R2.fastq.gz
-rw-rwx---. 1 llei morrellp 2.5G Aug 21  2018 WBDC_009_R1.fastq.gz
-rw-rwx---. 1 llei morrellp 2.5G Aug 21  2018 WBDC_009_R2.fastq.gz
-rw-rwx---. 1 llei morrellp 2.4G Aug 21  2018 WBDC_018_R1.fastq.gz
-rw-rwx---. 1 llei morrellp 2.4G Aug 21  2018 WBDC_018_R2.fastq.gz
-rw-rwx---. 1 llei morrellp 2.4G Aug 21  2018 WBDC_010_R1.fastq.gz
-rw-rwx---. 1 llei morrellp 2.4G Aug 21  2018 WBDC_010_R2.fastq.gz
-rw-rwx---. 1 llei morrellp 2.4G Aug 21  2018 WBDC_001_R2.fastq.gz
-rw-rwx---. 1 llei morrellp 2.4G Aug 21  2018 WBDC_001_R1.fastq.gz
-rw-rwx---. 1 llei morrellp 2.4G Aug 21  2018 WBDC_008_R1.fastq.gz
-rw-rwx---. 1 llei morrellp 2.4G Aug 21  2018 WBDC_008_R2.fastq.gz
-rw-rwx---. 1 llei morrellp 2.3G Aug 21  2018 WBDC_006_R1.fastq.gz
```

“Manual” checks for any step

- Do we have the expected number of files?
  - **`ls *pattern* | wc -l`**
    - Good first check for raw FASTQ files, adapter trimmed FASTQ files, SAM/BAM files, split vcf files, etc.

```
liux1299@ln0004:~/Shared/Datasets/NGS/Barley_Exome/WBDC_Inversion_Samples/100bp_concatenated_fastq $ ls
missing_metadata_samples            WBDC_004_R2.fastq.gz  WBDC_009_R1.fastq.gz  WBDC_020_R2.fastq.gz
sra_submission_wbdc100bp_list.txt   WBDC_005_R1.fastq.gz  WBDC_009_R2.fastq.gz  WBDC_022_R1.fastq.gz
WBDC_001_R1.fastq.gz                WBDC_005_R2.fastq.gz  WBDC_010_R1.fastq.gz  WBDC_022_R2.fastq.gz
WBDC_001_R2.fastq.gz                WBDC_006_R1.fastq.gz  WBDC_010_R2.fastq.gz  wbdc_100bp_raw_fastq_list.txt
WBDC_002_R1.fastq.gz                WBDC_006_R2.fastq.gz  WBDC_018_R1.fastq.gz
WBDC_002_R2.fastq.gz                WBDC_008_R1.fastq.gz  WBDC_018_R2.fastq.gz
WBDC_004_R1.fastq.gz                WBDC_008_R2.fastq.gz  WBDC_020_R1.fastq.gz
liux1299@ln0004:~/Shared/Datasets/NGS/Barley_Exome/WBDC_Inversion_Samples/100bp_concatenated_fastq $ ls *R1* | wc -l
11
liux1299@ln0004:~/Shared/Datasets/NGS/Barley_Exome/WBDC_Inversion_Samples/100bp_concatenated_fastq $ ls *R2* | wc -l
11
```

sequence_handling pipeline



Raw FASTQ Files

“Manual” checks for raw data

- File size and expected number of files
- Quality assessment using tools like FastQC (in sequence_handling)
- Compare checksums after transferring/downloading data
  - Good check for raw FASTQ files to find truncated files

Generating checksums:

- On MSI use **`md5sum`**
- On Mac use **`md5`**

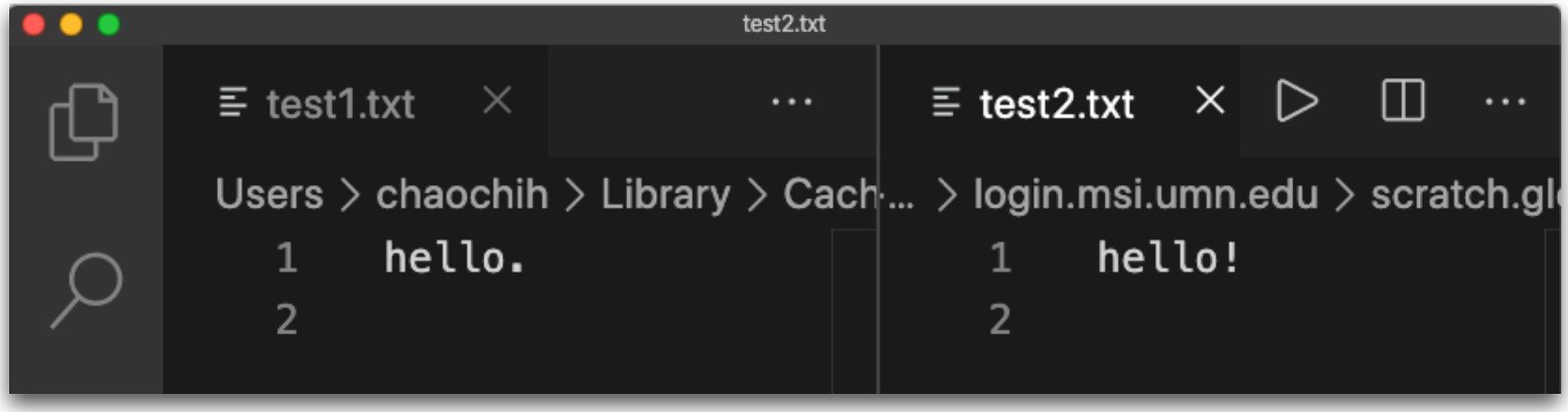
```
# Generate checksum for downloaded/transferred files
find *.fastq.gz | parallel "md5sum {}" > md5_downloaded.txt

# Make sure both lists are sorted in same order
# Here, we only sort list we generated
sort -k 2,2 md5_downloaded.txt

# Compare checksums
diff -y md5_downloaded.txt md5_original.txt
```

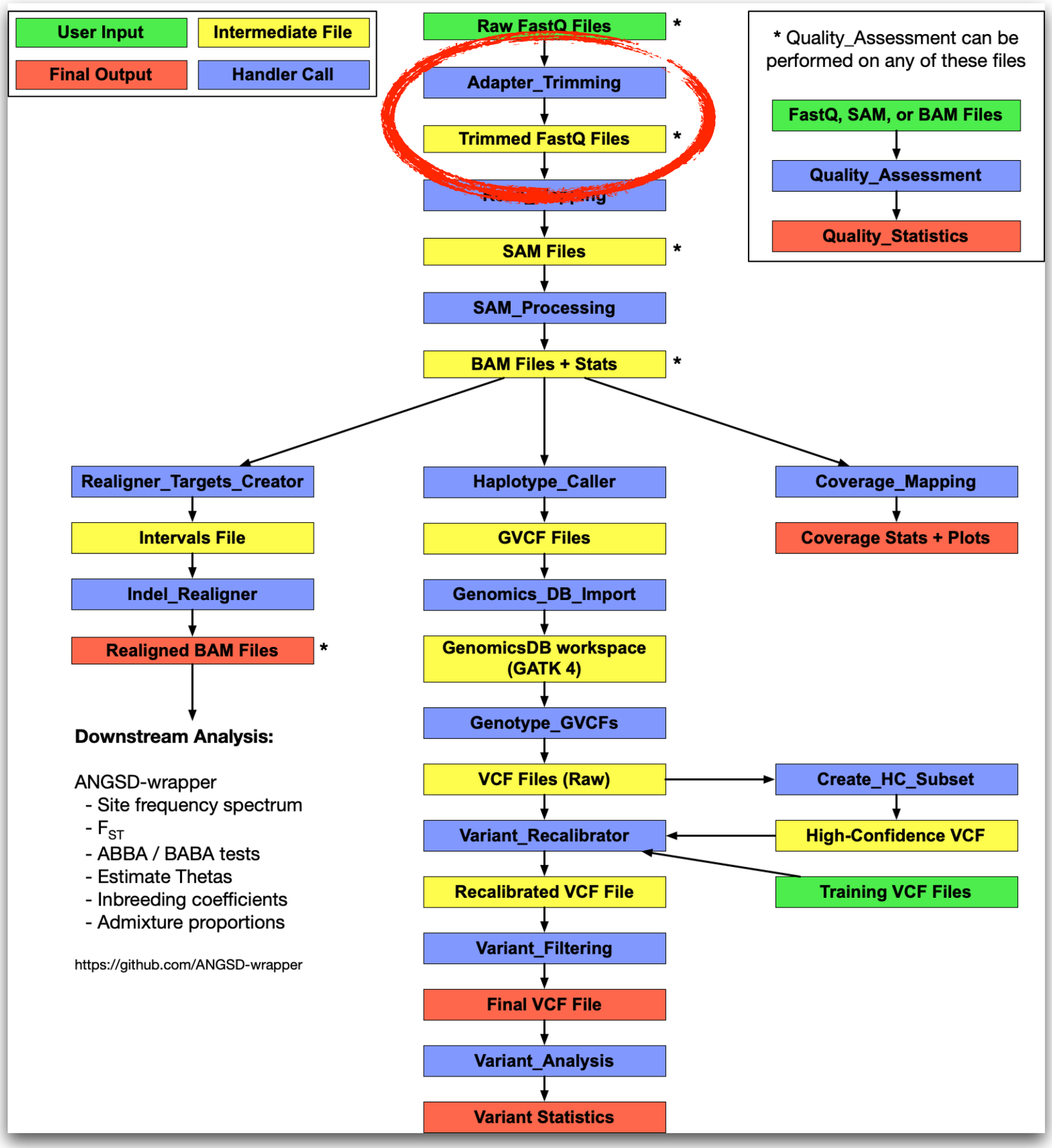
What are checksums?

- Many flavors: MD5, SHA-1, SHA-256, SHA-512
- Uses an algorithm to produce a sequence of numbers and letters of a fixed length
- Differences in files (even very small changes) produce different checksums




```
liux1299@ln0004:~/scratch $ diff -y md5_test1.txt md5_test2.txt
8563c2a9b1b1593b4b576bf7e1e32366  test1.txt                         | 8b8db3dfa426f6bdb1798d578f5239ae  test2.txt
```

sequence_handling pipeline



Trimmed FASTQ

Automated checks: Adapter_Trimming handler

- Do we have a valid adapters file?
- Have we specified a quality encoding in the config file?
    - Choose from: 'sanger', 'illumina', 'solexa', or ‘phred'

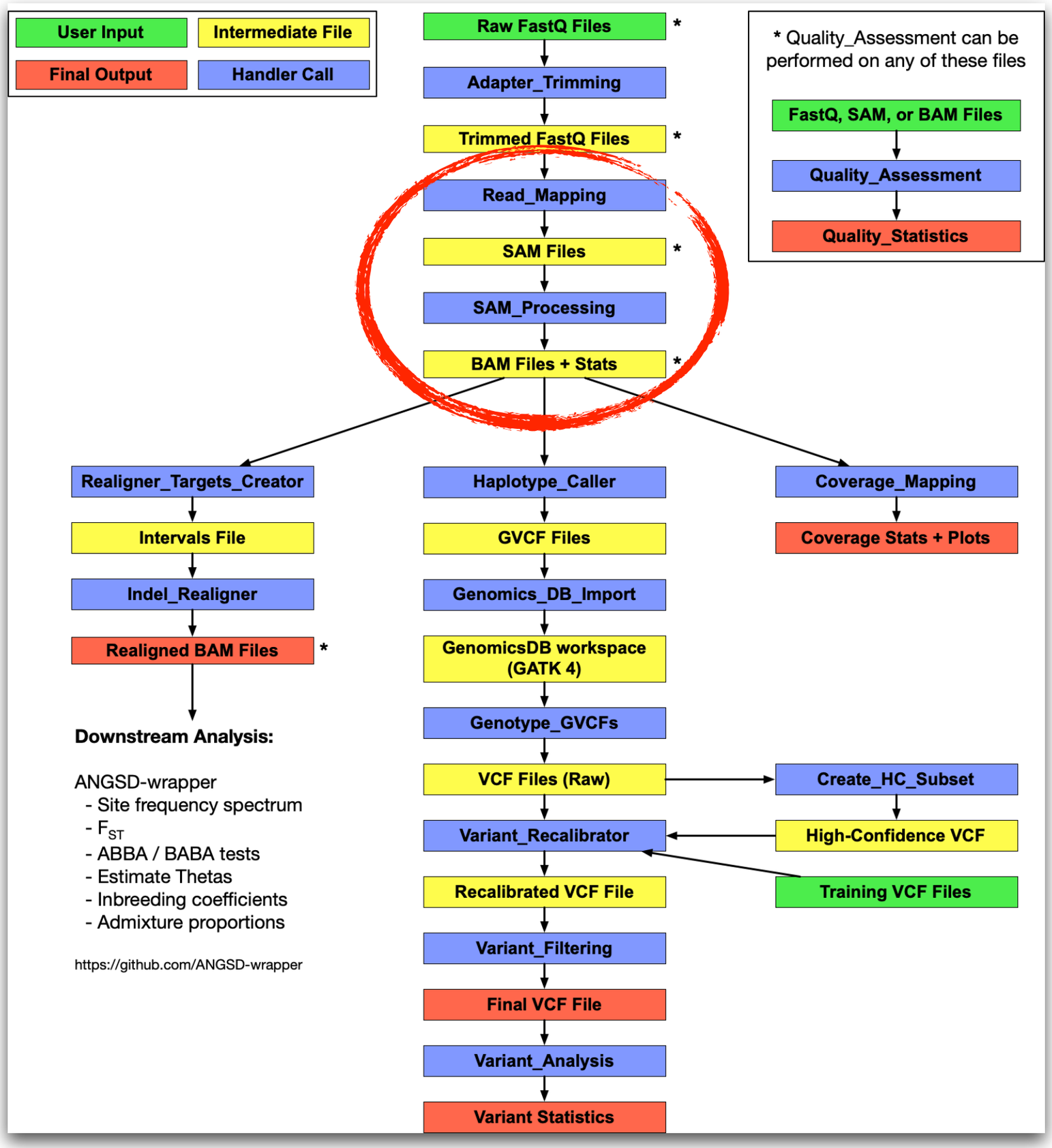
Trimmed FASTQ

“Manual” checks for adapter trimmed files

- File size and expected number of files
- Quality assessment output report

```
# Go into output directory containing trimmed files
cd ~/Path/to/output/dir/Adapter_Trimming
# Check file sizes
# Use globbing before and after in case some files
#   have not been compressed yet
ls -lhS *.fastq*
# Can also just look at largest and smallest files
ls -lhS *.fastq* | head
ls -lhS *.fastq* | tail
# Check if # of forward and reverse fastq files match
ls *Forward_ScytheTrimmed.fastq.gz | wc -l
ls *Reverse_ScytheTrimmed.fastq.gz | wc -l
```

sequence_handling pipeline



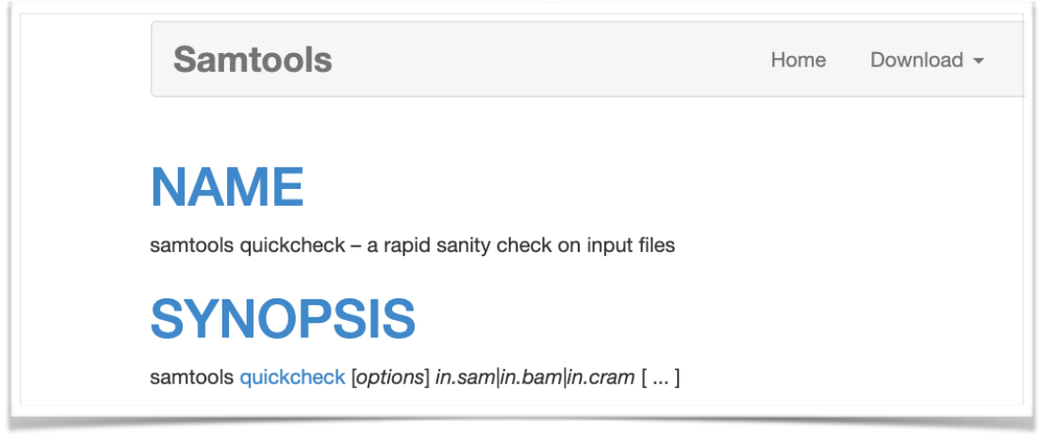
SAM/BAM Files

Automated checks: Read_Mapping and SAM_Processing handlers

- Is our reference genome indexed?
- Have we specified a valid sequence platform in the config file? (e.g., ILLUMINA)
- For Read_Mapping, if we have paired end samples, check if we have equal numbers of forward and reverse samples.

“Manual” checks for SAM/BAM files

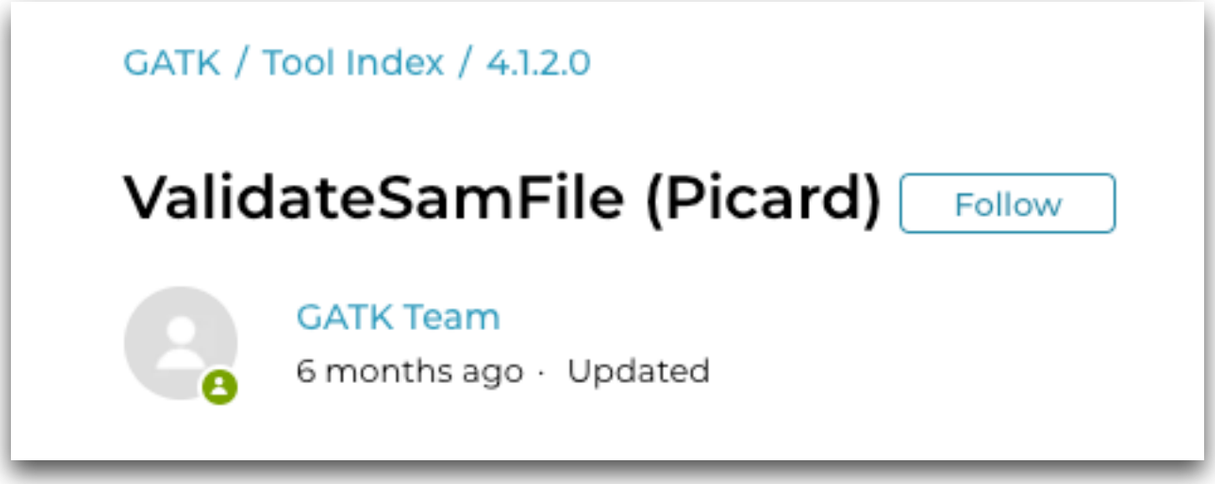
- File size and expected number of files
- Are any of our SAM/BAM files truncated?
  - Use samtools quickcheck
  - Important: Does NOT check for internal corruption, only checks headers plus one target sequence and end-of-file (EOF) presence




```
samtools quickcheck -qv *.bam > bad_bams.fofn \
    && echo 'all ok' \
    || echo 'some files failed check, see bad_bams.fofn'
```

“Manual” checks for SAM/BAM files

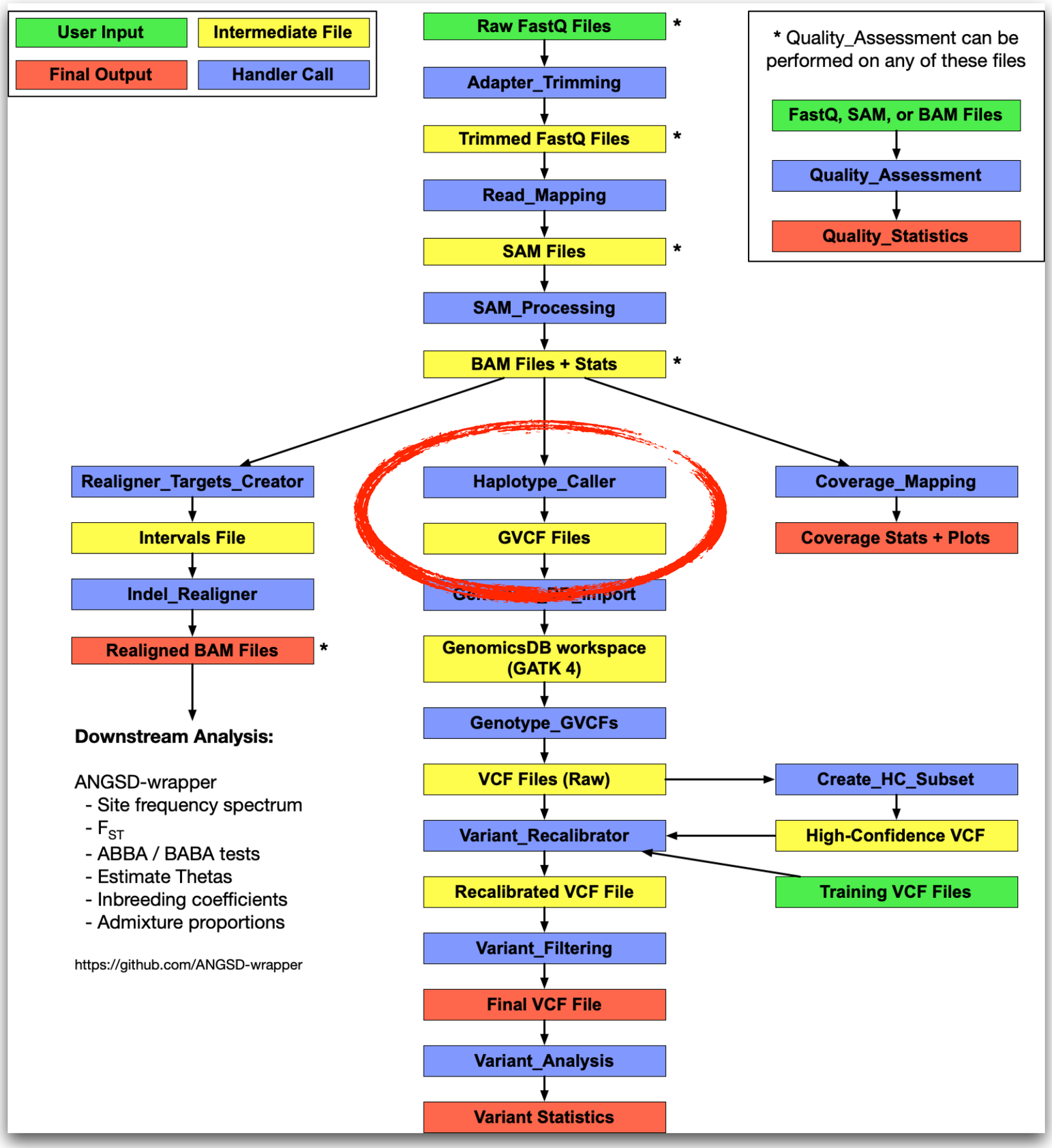
- Alternatively, use ValidateSamFile for both SAM and BAM files to catch:
  - Improper formatting (relative to SAM format specification)
  - Faulty alignments
  - Incorrect flag values



Usage example:

```
java -jar picard.jar ValidateSamFile \
      I=input.bam \
      MODE=SUMMARY
```

sequence_handling pipeline



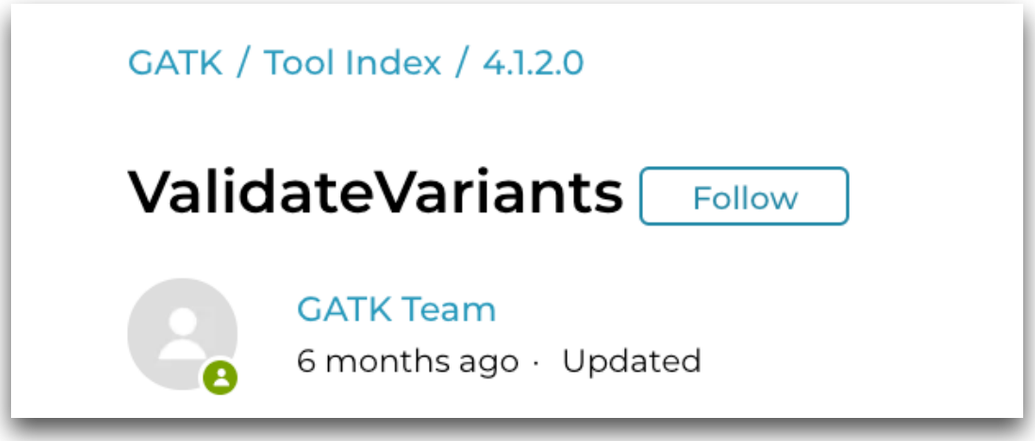
GVCF Files

Automated checks: Haplotype_Caller handler

- Are the BAM files indexed?
- Is GATK installed? If so, are we running GATK v3 or v4?
- Does our reference genome have a dict file?

“Manual” checks for GVCF files

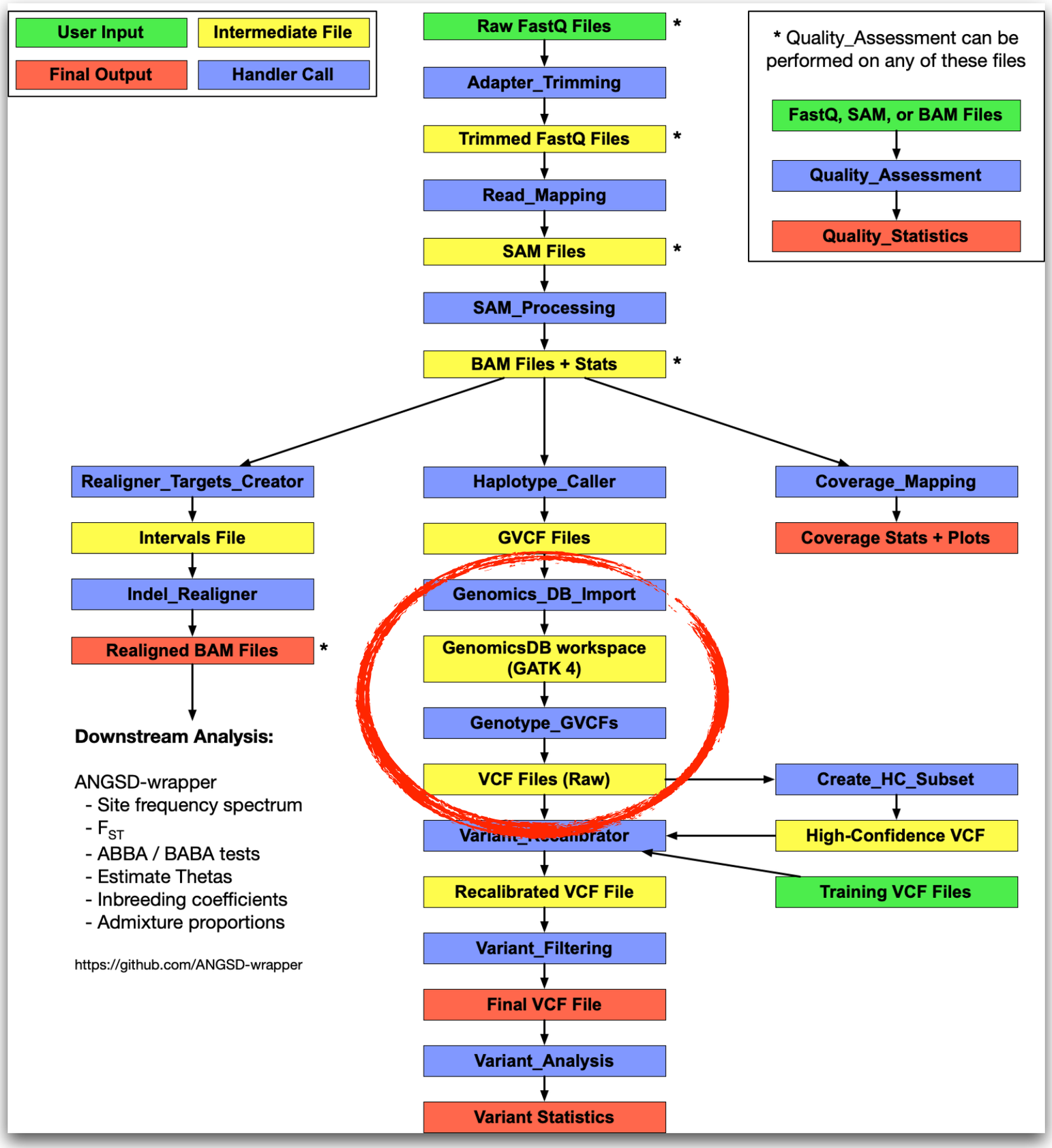
- File size
- Expected number of files (especially if we are parallelizing across regions)
- GATK’s [ValidateVariants]



Validate a GVCF for adherence to VCF format, including REF allele match:

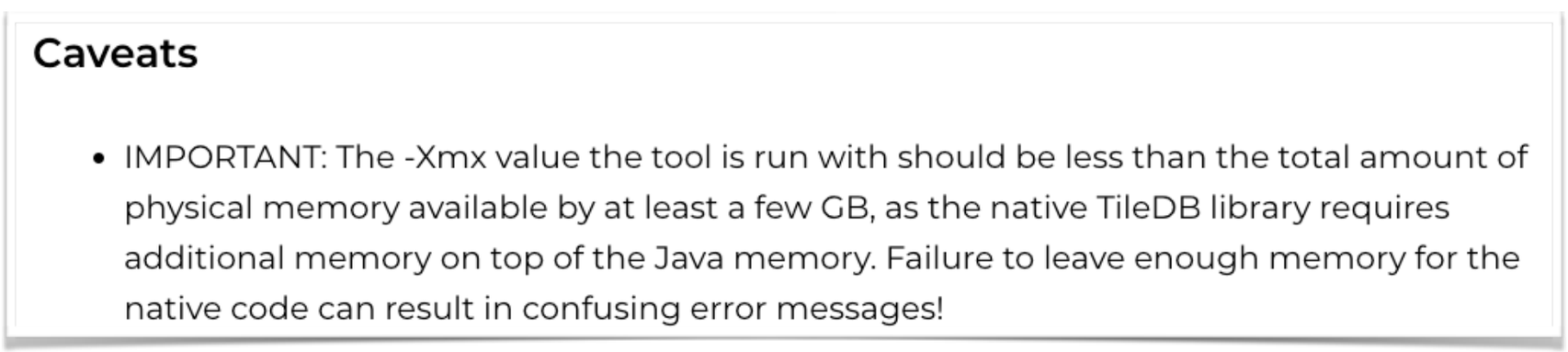
```
gatk ValidateVariants \ -V sample.g.vcf.gz \ -R reference.fasta -gvcf
```

sequence_handling pipeline



Automated checks: Genomics_DB_Import handler

- Are the GVCF files indexed?
- Is GATK installed? If so, are we running GATK v3 or v4?
- Are we running GATK v3 or v4?
    - For GATK v4, automatically adjusts memory to leave enough for the TileDB library on top of Java



Caveats

- IMPORTANT: The -Xmx value the tool is run with should be less than the total amount of physical memory available by at least a few GB, as the native TileDB library requires additional memory on top of the Java memory. Failure to leave enough memory for the native code can result in confusing error messages!

“Manual” checks for GenomicsDB workspaces

```
liux1299@ln0004:~/scratch/gendb_wksp_chr1H_part1_105196836-109695587 $ tree
.
├── callset.json
├── chr1H_part1$105196837$109695587
│   ├── __array_schema.tdb
│   ├── __c11348e1-01fe-4d16-8961-8b58077993a947937991378688_1576215421817
│   │   ├── AD.tdb
│   │   ├── AD_var.tdb
│   │   ├── ALT.tdb
│   │   ├── ALT_var.tdb
│   │   ├── BaseQRankSum.tdb
│   │   ├── __book_keeping.tdb.gz
│   │   ├── __coords.tdb
│   │   ├── DP_FORMAT.tdb
│   │   ├── DP.tdb
│   │   ├── DS.tdb
│   │   ├── END.tdb
│   │   ├── ExcessHet.tdb
│   │   ├── FILTER.tdb
│   │   ├── FILTER_var.tdb
│   │   ├── GQ.tdb
│   │   ├── GT.tdb
│   │   ├── GT_var.tdb
│   │   ├── ID.tdb
│   │   ├── ID_var.tdb
│   │   ├── InbreedingCoeff.tdb
│   │   ├── MIN_DP.tdb
│   │   ├── MLEAC.tdb
│   │   ├── MLEAC_var.tdb
│   │   ├── MLEAF.tdb
│   │   ├── MLEAF_var.tdb
│   │   ├── MQRankSum.tdb
│   │   ├── PGT.tdb
│   │   ├── PGT_var.tdb
│   │   ├── PID.tdb
│   │   ├── PID_var.tdb
│   │   ├── PL.tdb
│   │   ├── PL_var.tdb
│   │   ├── PS.tdb
│   │   ├── QUAL.tdb
│   │   ├── RAW_MQandDP.tdb
│   │   ├── ReadPosRankSum.tdb
│   │   ├── REF.tdb
│   │   ├── REF_var.tdb
│   │   ├── SB.tdb
│   │   └── __tiledb_fragment.tdb
│   └── genomicsdb_meta_dir
│       └── genomicsdb_meta_684afb33-0d2e-4407-9f70-671cf03a28b3.json
├── __tiledb_workspace.tdb
├── vcfheader.vcf
└── vidmap.json

3 directories, 46 files
```

Expected number of workspaces (especially if we are parallelizing across regions)

The tree command for linux:
http://mama.indstate.edu/users/ice/tree/

VCF Files

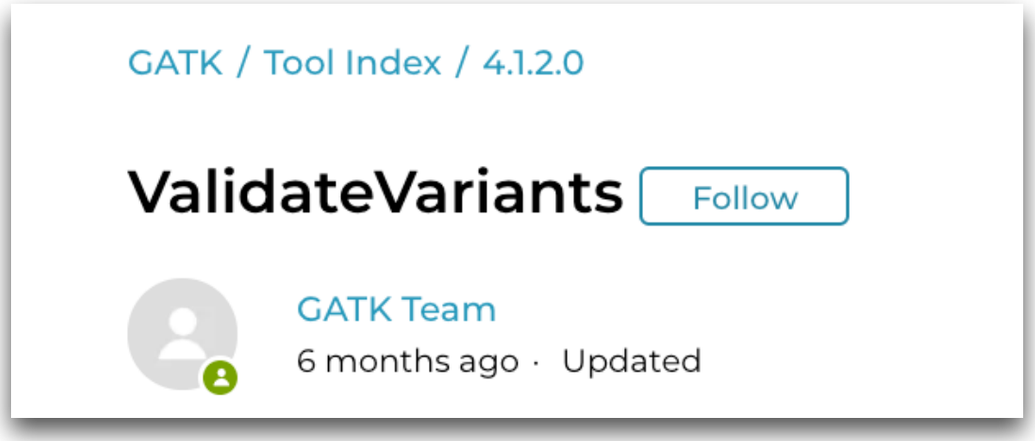
Automated checks: Genotype_GVCFs handler

- Is GATK installed? If so, are we running GATK v3 or v4?

- Does our reference genome have a dict file?

“Manual” checks for VCF files

- File size
- Expected number of files (especially if we are parallelizing across regions)
- GATK’s ValidateVariants



Minimally validate a file for adherence to VCF format:

```
gatk ValidateVariants \ -V cohort.vcf.gz
```