Documentation for NYUAD Date Palm GATK Variant-Calling Workflow

NYUAD Date Palm Diversity Project
M.Dhar (mid224) 4-24-23

INTRODUCTION

Documentation and scripts stored in shared Google Drive directory at:
https://drive.google.com/drive/folders/1K6kMVOkhpA3xueNYX0dBd2f9ECpY1FmE?usp=share_link

Background and Purpose

This is a variant-calling workflow using the GATK (Genome Analysis Toolkit: https://gatk.broadinstitute.org/hc/en-us) tools in the Nextflow pipeline framework (https://www.nextflow.io/). The core Nextflow variant-calling script was developed by M. Dhar (mid224) in 2021 for use by NYU Date Palm Diversity Project researchers. It is an adaptation of the "Variant Calling Pipeline using GATK4" developed by M. Khalfan in 2020 for the Genomics Core in 2020 (https://gencore.bio.nyu.edu/variant-calling-pipeline-gatk4), a simplification of that pipeline, with adaptations and fine-tuning for the needs of the Date Palm group. Like that pipeline, it uses GATK to call variants, "including single nucleotide polymorphisms (SNPs) and DNA insertions and deletions (indels), from next generation sequencing data." See the documentation for the Genomics Core pipeline for more details on that original pipeline and the variant-calling process (https://gencore.bio.nyu.edu/variant-calling-pipeline-gatk4).

This Date Palm workflow additionally includes a separate Nextflow script for conducting FastP preprocessing of sequence reads and a BASH script for producing symlinks to read files in the format needed by the Nextflow scripts. This workflow runs on the NYU Greene high-performance computing (HPC) cluster (https://sites.google.com/nyu.edu/nyu-hpc/hpc-systems/greene).

Reference Pipeline Adaptations

The Genomics Core Nextflow pipeline script was adapted for use in this workflow in several ways (see that original script on GitHub here: https://github.com/gencorefacility/variant-calling-pipeline-gatk4/blob/master/main.nf):

- It was simplified to focus on the first four steps ("processes" in Nextflow terminology): align, markDuplicatesSpark, getMetrics, and haplotypeCaller.
- An additional process was added: IndexFeatureFile (another GATK tool).
- The haplotypeCaller command was updated to produce a GVCF for each sample, which can be merged after running the pipeline.
- The "Base Quality Score Recalibration" (BSQR) step of the Genomics Core pipeline (in which variants are called first without BSQR, then used as input to BSQR, then called again) was removed.
- To reduce disk usage, code was added to various steps to delete output produced by previous steps in Nextflow's "work" directories as the pipeline proceeds. Similarly, to

avoid overwhelming the default temp directory in the markDuplicatesSpark process, the temp directory was set as a folder on the user's "vast" directory on Greene. (You will need to create a folder "vast<NetID>/tmp" if it doesn't already exist and add your NetID to the GATK pipeline configuration file [nextflow.config]. See below.)

- Code was added to create and sort a file containing checksum "fingerprint" for each output.
- Code was added to trim sample pair IDs (including only up until the first underscore) to a more readable format when used in output names.
- Other fine-tunings and updates have been done as needed and desired for use by the Date Palm group researchers.

The Sarek variant-calling pipeline produced by Nextflow itself was also used as background reference (https://nf-co.re/sarek).

Dates: Development and Previous Production Run
The main GATK Nextflow pipeline script in this workflow was developed primarily in 2021, with additional fine-tuning and adaptation since then. The symlink BASH script for inputs was developed at the same time. An additional Nextflow pipeline script for running FastP preprocessing was created in early 2023.

A production run of the GATK pipeline (without the FastP script) was completed in November 2021 on a preliminary version 2 date palm genome then in production by Tiago Capote. Output, scripts, logs, and notes from that run are available in the shared Greene drive for the NYU Michael Purugganan Lab at: */scratch/projects/puruggananlab/GVCF_pipeline_11-6-21/*

PIPELINE SCRIPTS AND STEPS

Included Scripts and Needed Files
Scripts: The current Date Palm variant-calling workflow consists of three main scripts and their associated support scripts:

- **Symlink creator (BASH script)**: *symlink_shareable_4-25-23.sh*
- **Nextflow FastP script and supporting scripts/files:**
  - Nextflow script: *fastp_shareable_4-25-23.nf*
  - Slurm wrapper script: *fastp_shareable_4-25-23.s*
  - Nextflow config file: *nextflow.config*
- **Nextflow GATK variant-calling script and supporting scripts/files:**
  - Nextflow script: *main_shareable_4-25-23.nf*
  - Slurm wrapper script: *slurm_shareable_4-25-23.s*
  - Nextflow config file: *nextflow.config*

[Note that each Nextflow script needs an accompanying config file named "nextflow.config". This same name must be used. Be sure to save the respective nextflow scripts and their associated config file and Slurm scripts in separate directories. See RUNNING THE PIPELINE below.]

Needed Input and Reference Files:

- **Sample reads:**
  - **Reads:** Paired forward and reverse g-zipped FastQ reads in one directory.
  - **Manifest:** A (3-column) tab-separated file listing each read's sample ID, the forward read file name, and the reverse read file name. (An example of such a manifest, named: *manifest.3.2.2021.tsv*, is also included in the *symlink_shareable_4-25-23.sh* script folder in the shared script directory).
- **Reference genome and associated indices:**
  - **Reference:**
    - We are currently using the NCBI date palm genome *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic.fna* (originally downloaded 6-14-22 from https://ftp.ncbi.nlm.nih.gov/genomes/refseq/plant/Phoenix_dactylifera/latest_assembly_versions/GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p/) and adapted 4-8-23 to truncate header lines (using Picard NormalizeFasta (https://gatk.broadinstitute.org/hc/en-us/articles/360037067312-NormalizeFasta-Picard-).
    - This adapted version of the genome is available at: */scratch/projects/purugganan lab/palm/PlantGenomeDownloads_2022/RefSeq_Transfer/11InitialDownloads/Phoenix_dactylifera_date_palm/GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_mod_headers_4-8-13* with file name: *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.fna*
  - Dictionary and indices: Several support files are needed with the genome. These have all been created for the above genome and are available in the same directory, as required by the pipelines:
    - FASTA dict file: *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.dict*
    - FASTA index file: *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.fna.fai*
    - BWA index files (for BWA alignment step):
      - *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.fna.pac*
      - *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.fna.amb* *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.fna.sa*

        - *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.fna.ann README.txt*
        - *GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.fna.bwt*

Shared Location

- Scripts are located on the shared Google Drive at: https://drive.google.com/drive/folders/14Y9iCZzHg0cX28kT5fsYizyAKNjFW-JX?usp=share_link
- NCBI adjusted genome and associated indices are available on Greene shared directory at: */scratch/projects/purugganan1ab/palm/PlantGenomeDownloads_2022/RefSeq_Transfer/11InitialDownloads/Phoenix_dactylifera_date_palm/GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_mod_headers_4-8-13*

Tools Used in Workflow:

- FastP
- GATK4
- BWA
- Samtools
- R (as dependency for some steps)

Steps

- The major steps of this pipeline are as follows:
    - **Symlinks**: Create symlinks of read files with name format required by Nextflow scripts.
    - **FastP**: Run FastP Nextflow script to preprocess reads. (This is run as a separate pipeline script so that you can move/archive the original samples to save space before running the GATK pipeline script. See below.)
    - **Variant calling**: Run GATK Nextflow script to call variants:
        - **align**: Map to reference genome using BWA MEM.
        - **markDuplicatesSpark**: Mark duplicates and sort aligned reads (BAMs) using GATK4 MarkDuplicatesSpark.
        - **getMetrics:** Collect alignment and insert size metrics.
        - **cleanMetrics**: Remove previous steps (large) metrics files.
        - **haplotypeCaller**: Call variants, producing GVCF, using GATK's HaplotypeCaller tool.
        - **IndexFeatureFile**: Create feature file for above-created GVCF using GATK's IndexFeatureFile too.
- See **Schematic** of steps in this workflow in the same Google Drive directory as this documentation: *GVCF_Pipeline_Schematic_3-27-23.pdf*. For more background on these steps, see original "Workflow Overview" in original Genome Core documentation

(though note above modifications in the Date Palm workflow):
https://gencore.bio.nyu.edu/variant-calling-pipeline-gatk4

How the Scripts Work (Slurm and Config Files):

- **Symlink BASH:** The symlink script is a BASH script that will create appropriate names for your sample reads. Bash scripts are launched by entering *bash <script_name>.sh*
- **Slurm script wrappers:** The Nextflow scripts are each launched by an associated Slurm script (Slurm is the job-handling program in the HPC). (This allows the script to run in the background without the need to keep your terminal open and avoids failure if you lose power or connection to the terminal.) These are launched by entering: *sbatch <script_name>.s*. The Slurm script will load the Nextflow module and launch the Nextflow script. (It will also sort the checksum file produced by the GATK pipeline.)
- **Nextflow scripts:** Each process within the Nextflow scripts is launched as a separate Slurm job. (This happens automatically once the overall pipeline script is launched.) Nextflow reads in read pairs by looking for a specific pattern ending file names; this script uses files that end in 1.fastq.gz or 2.fastq.gz (eg, “PADC1_read1.fastq.gz”). The symlink script creates symbolic links to your sample files with the expected name endings.
- **Nextflow config file:** The config file contains various parameters needed by the Nextflow script, including setting Slurm as the “process.executor” and setting the resource requests for each process.

RUNNING THE PIPELINE

To run this pipeline, you will need to prepare input files, copy scripts to directories in your Greene scratch, and run the scripts, as follows:

- **Prepare input files:**
  - *Reference genome:*
    - **Copy NCBI genome:** If using the same NCBI date palm genome (with adjusted headers) as the Date Palm group, copy that genome and its associated dict and index files (see above lists), to your scratch. Recommended to create a specific directory for this, eg: */scratch/<net_ID>/data/genomes*. These files are available at: */scratch/projects/puruggananlab/palm/PlantGenomeDownloads_2022/RefSeq_Transfer/11InitialDownloads/Phoenix_dactylifera_date_palm/GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_mod_headers_4-8-13*
    - **Create indices for different genome:** If using a different genome, you will need to copy or create its indices and dict file. See GATK documentation for creating dict and fai index files: https://gatk.broadinstitute.org/hc/en-us/articles/360035531652-FASTA-Reference-genome-format. For BWA index documentation, on Greene, enter *module load bwa/intel/0.7.17* and type *bwa index* for help menu. The Slurm script in the above shared

directory for the genome shows the commands used to create those index and dict files.

- *Sequence reads:*
  - All pairs of forward and reverse sequence reads should be collected in **one** directory.
  - These should be g-zipped FASTQ files with extension *.fastq.gz*
  - Create or copy in a 3-column tab-separated manifest listing, in order:
    - Col 1: Pair ID name
    - Col 2: Forward read file name
    - Col 3: Reverse read file name
  - Forward and reverse reads should ideally be identified by including “read1” vs “read2” or “R1” vs “R2” somewhere in the name. However, this is not essential as long as forward read (read 1) is listed in the second column of the manifest and reverse in the third.
  - Copy the symlink script into the same directory as all reads.
    - Update it as follows:
      - Sample path: Set the path to the directory containing your samples at: *sample_path="<PATH_TO_SAMPLE_DIR>"*
      - Set the path and name of your manifest of sample names at: *manifest="<MANIFEST_FILE>"*
    - Run it using: *bash symlink_shareable_4-25-23.sh*
    - This will create a folder, within the directory containing your samples, called *Sample_softlinks*. With in that folder, each read pair will get its own subdirectory with the sample ID name. Within each subdirectory, will be symlinks pointing to your samples. The files will end in “1.fastq.gz” or “2.fastq.gz” as expected by Nextflow scripts.

- **Copy Nextflow scripts**
  - Place the two Nextflow scripts and their associated config files and Slurm in separate directories, eg:
    - Directory *FastP_Pipeline*, containing:
      - *fastp_shareable_4-25-23.nf*
      - *fastp_shareable_4-25-23.s*
      - *nextflow.config*
    - Directory *GATK_Pipeline*, containing:
      - *main_shareable_4-25-23.nf*
      - *slurm_shareable_4-25-23.s*
      - *nextflow.config*
  - Note that the Nextflow config file for each Nextflow script above (FastP and GATK scripts) must have the name “nextflow.config”, which is automatically identified as the configuration file by the Nextflow script. Place in separate directories to avoid overwriting.
- **Update scripts and config file:**

- *Slurm wrapper scripts:*
    - Update to send the job notice to your email address:
        - *#SBATCH --mail-user=<email_address>*
        - [Note: the Slurm wrapper script doesn't access the Nextflow config script, so set your email address with your NetID manually here.]
    - Update the job name in:
        - *#SBATCH --job-name=<job_name>*
- *Config files:* You will need to set several parameters in the nextflow.config files for each Nextflow script.
    - **Input reads--FastP:** For the FastP config file, set the location of the input reads by pointing to the directory created by the symlink script (*Sample_softlinks*). This done in the line: *params.reads = "$SCRATCH/<PATH>/Sample_softlinks/*/*{1,2}.fastq.gz"*
        - The ending is a pattern (glob pattern) that tells the Nextflow step that reads in sample names to look for file pairs that end in "1.fastq.gz" and "2.fastq.gz".
        - The asterisk within slash marks (*/*/*) tells the pattern to go into each subdirectory for each read pair in *Sample_softlinks*, eg, *Sample_softlinks/PDAC1*.
    - **Input reads:--GATK** For the GATK config file, set the location of the input reads by pointing to the output file of the FastP nextflow script: *params.reads = "$SCRATCH/<PATH_to_FastP_Directory>/out/out/fastp/*/*{1,2}.fastq.gz"*
    - **Out directory:** For both config files, set the output directory as: *params.outdir = "$SCRATCH/<PATH_to_Script_Directory>/out"*. That is, it should point to the directory in which you've copied the respective nextflow scripts.
    - **Reference genome:** For GATK, set the reference genome in the line, eg: *params.ref = "$SCRATCH/data/genome_assemblies/GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_mod_headers_4-8-13/GCF_009389715.1_palm_55x_up_171113_PBpolish2nd_filt_p_genomic_mod_headers.fna"*
    - **Platform:** For GATK, set the platform used to produce the reads (likely, Illumina), as in: *params.pl = "ILLUMINA"*
        - This is used to create the read group in the script.
    - **NetID:** For GATK, enter your NetID (the ID before the @ symbol in your NYU email address) at: *params.netID = "<NetID>"*.
        - This is used to direct temp files from markDuplicatesSpark to a temp folder in the vast directory named */vast/<NetID>/tmp*.
        - Create this *tmp* folder if it doesn't already exist in your vast directory. (Don't overwrite it if it does already exist.)

            - This is done to avoid overwhelming the default temp directory in large pipeline runs.
        - **Resource Requests:** Resource requests for each process within the Nextflow pipeline are set at the end of the config file. You may want to adjust these depending on the size of your job and files.
- **Run Nextflow script Slurm wrappers:**
    - **FastP:** Launch the FastP Nextflow script via its Slurm wrapper by entering: *sbatch fastp_shareable_4-25-23.s*
        - Slurm will output a slurm job number. Copy this number to monitor your job.
        - Once this job is completed, you may want to archive or move the original (pre-FastP) samples to free up space.
    - **GATK:** Once FastP has completed and you have verified the preprocessed outputs, launch the GATK Nextflow script via its Slurm wrapper by entering: *slurm_shareable_4-25-23.s*
        - Again, copy Slurm job number.
- **Output:**
    - *Work directory*: Nextflow will do its processing within a work directory called *nextflow_work_dir* within the out directory you set in the config file above. Find it at *out/nextflow_work_dir* in your script directory.
    - *Output directory*: Readable, organized output from each process will be "published' to output folders within a directory called "out" within the overall output directory you defined above.
        - That means output will be within an *out/out* path.
        - Each process has its own subdirectory, so for example, output for haplotype caller (ie, the GVCFs), will be published to *out/out/haplotype_caller*.
        - To save space, original BAMs from the alignment step are not published; only deduplicated BAMs are published.
        - The overall *out/out* folder will also contain the checksum file (*checksum.txt*). At the end of the pipeline, entries will be sorted, as: *checksum_sorted.txt*.
- **Monitoring script runs and reports:**
    - You can monitor your ongoing Nextflow pipelines in several ways:
        - Enter *squeue -u <netID>* to see the status of your Slurm jobs. This will include entries for the overall wrapper Slurm scripts you launched above as well as the jobs spawned by the Nextflow script. (Use the Slurm job numbers generated above to find the overall wrapper script runs.)
        - Read the Slurm job log for your overall script. You can view this using the Slurm job number copied above. The log will be in the same directory where you launched the job. Enter *cat *<job_number>** to output contents to screen.
        - After the pipeline finishes or fails, view the Nextflow report in the file named "*report_slurm_%j.html*". If you run multiple jobs in the same

directory, the old report will get a number appended (eg, report_slurm_%j.html.1). These will always go in reverse order, ie, the newest job will have no number, the next newest will have 1, and the next newest 2, etc. You can view these html files by downloading to your home machine and opening with a web browser.

- After the pipeline finishes or fails, you should also receive an email (although, this can sometimes fail to send if the job fails).
- To view the Nextflow log for troubleshooting, list invisible files in your pipeline script directory using *ls -a*. This file will be named *.nextflow*
    - Note the period before the name.

TROUBLESHOOTING/TIPS

- **Resume**: Nextflow has a "resume" function that allows you to restart a failed or stopped pipeline after its last completed step. This setting is used automatically in the Slurm wrapper scripts that launch the Nextflow scripts. However, it will not work for many of the steps in this pipeline due to the deletion of previous output for disk space-saving purposes. If you want to be able to resume at each step (eg, during testing), you'll need to comment out the deletion steps for old files. M. Dhar may be able to help with this.
- **Space**: If you need to save more space, you may want to comment out the Samtools depth command from the getMetrics step (as well as its reference in the output of that step). M. Dhar may be able to help with this if necessary.
- **Time**: Particularly with haplotypeCaller jobs, you may need to monitor your Slurm queue to see if any are going to run out of time and ask HPC support (Shenglong at hpc@nyu.edu) to add clock time to that job. Monitor by entering *squeue -u <netID>*
- **Testing/subfolders:** If you're doing testing and/or multiple runs, you may want to create a new subfolder and copy in the scripts and config file. Be sure to update the "out" file path in config to the new folder. If you want to run a new job in the same directory, archive your "out" directory to avoid overwriting.
- **Testing samples:** M.Dhar can provide a set of 1 short sample for quick testing (PACA1) or a set of two samples (one longer) for testing of multiple sample runs (PACA1 and PDAC106). (You might want to try saving these with longer names to test the symlink renaming function.) J. Flowers may have test files to use as well.
- **Nextflow vs BASH variables:** Within each Nextflow process, the "script" is a BASH command running the respective tool (this BASH script is contained within 2 sets of 3 quotation marks). Note, however, that it does not behave perfectly as BASH. For example, to use a BASH variable within the script—and not interpret it as a Nextflow variable—it must be escaped with a slash sign. Instead of using *$VARIABLE_NAME*, you would use *\$VARIABLE_NAME*.