ngs week4 assignment – Xiaoyan Wen

XiaoyanWen

2/14/2022

Next Generation Sequence Analysis Homework Week 4

In this assignment, you will call snps and genotypes from 30 samples from the 1000 Human Genomes Project. You will also apply a set of hard filters to the callset, and select the high quality snps for population genomic analysis in Week 5.

About the data

This week we will continue working with re-sequencing data from the 1000 Genomes Project from Weeks 2 and 3. We will work with the same 30 samples whose reads you or your instructor processed with fastp and aligned to the human reference genome.

Your instructor has conducted additional processing of the alignments to coordinate sort the alignments, mark duplicate read pairs (PCR artifacts), and run the HaplotypeCaller step of HaplotypeCaller/GenotypeGVCFs workflow for variant discovery with the Genome Analysis Toolkit.

The output of this workflow is a combined gvcf representing 29 sample gvcfs. Your instructor dropped one of the 30 samples for technical reasons. You will now run the final step of the workflow to *call SNPs and genotypes* and then use GATK tools to *refine the snp callset*.

Task 1: Call SNPs and Genotypes with GenotypeGVCFs

In a pre-recorded video, your instructor introduced the GATK workflow for snp-calling and genotyping.

This workflow includes the following conducted separately on each sample:

- Align reads with BWA-MEM
- Coordinate-sort
- Mark PCR duplicates
- Base Quality Score Recalibration (BQSR)
- Run HaplotypeCaller to create .gvcf files

After HaplotypCaller is completed, we have a .gvcf file for each sample. The workflow at this point proceeds by conducting the remaining steps on all samples simultaneously

- Run CombineGVCFs
- Run GenotypeGVCFs
- Perform Variant Quality Score Recalibration
- Perform hard filtering of VCF

In Week 3, you performed the BWA-MEM alignment step on each sample. Your instructor did the same and converted the output .sam files to .bam and then proceed with all steps above up to the GenotypeGVCFs step which is where this assignment begins. The input file to GenotypeGVCFs is a multi-sample .gvcf.

In addition to the above steps, your instructor completed additional minor steps required **such as creating BAM index files at each step (every coordinate-sorted .sam or .bam file requires a new index file).** Your instructor also created a sequence dictionary (.dict file) of the reference genome fasta. This is a type of index file of the reference genome which is required for all GATK tools.

You may review your instructor's scripts here:

```
/scratch/work/courses/BI7653/hw4.2022/hw4_alignment_processing.slurm
/scratch/work/courses/BI7653/hw4.2022/hw4_combinegvcfs_intervals.slurm
/scratch/work/courses/BI7653/hw4.2022/hw4_haplotypecaller.slurm
```

You will now create a script to run *GenotypeGVCFs*. This requires a single command line (NOT an array job). If you would like, you may do the following in preparation for this Task.

```
srun --time=4:00:00 --mem=10GB --pty /bin/bash
```

Now create a directory (e.g., ngs.week4) in your /scratch, create a subdirectory for Task 1, and create a slurm script to execute GenotypeGVCFs. You may wish to use the template script from Week 1

```
/scratch/work/courses/BI7653/hw1.2022/slurm_template.sh
```

Before beginning, review this guide to constructing command lines for GATK version 4 including how to provide arguments to the java virtual machine (JVM) with the –java-options argument. This is a helpful introduction to executing java archive file (.jar) that will be helpful to you in the future:

https://software.broadinstitute.org/gatk/documentation/article?id=11050 (https://software.broadinstitute.org/gatk/documentation/article?id=11050)

Instructions for the GATK version 4 GenotypeGVCFs command can be found here:

https://gatk.broadinstitute.org/hc/en-us/articles/360036820431-GenotypeGVCFs (https://gatk.broadinstitute.org/hc/en-us/articles/360036820431-GenotypeGVCFs)

Proceed with constructing your job submission script to run GenotypeGVCFs as follows:

1. Load the most recent version of GATK (must be version 4 or higher).
2. Use the GenotypeGVCFS example listed at the above page. Ignore the GenomicsDB example which is typically used when you have hundreds of samples.
3. Add to your GenotypeGVCFs command the following option:

```
--allow-old-rms-mapping-quality-annotation-data
```

4. Use the following reference genome fasta for your -R argument. **The required index files including a new dictionary file for GATK tools (produced by Picard Tool CreateSequenceDictionary) are all found in the same directory as the fasta:**

```
/scratch/work/courses/BI7653/hw3.2022/hg38/Homo_sapiens.GRCh38.dna_sm.primary_assembly.normalized.fa
```

5. The multi-sample .gvcf input produced by CombineGVCFs required for the -V argument is here:

```
/scratch/work/courses/BI7653/hw4.2022/cohort.g.vcf.gz
```

6. You may need to provide additional memory to the Java Virtual Machine (JVM) for your GenotypeGVCFs command using –java-options "-Xmx8G". If you increase the memory in this fashion, you should also increase the memory requested by your slurm job to 8 + 2 = 10GB. **It is generally advisable to provide an additional 1 or 2 GB of RAM to your job above the amount passed to the JVM.**

Now, submit your job using sbatch.

Q1.1 Please paste the contents of your sbatch script here

$2 points$

.

```
#!/bin/bash
#
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=8:00:00
#SBATCH --mem=8GB
#SBATCH --job-name=slurm_template
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=xw2470@nyu.edu

module purge

module load gatk/4.2.4.1

gatk --java-options "-Xmx8g" GenotypeGVCFs \
--allow-old-rms-mapping-quality-annotation-data \
-R /scratch/work/courses/BI7653/hw3.2022/hg38/Homo_sapiens.GRCh38.dna_sm.primary_assembly.normalized.fa \
-V /scratch/work/courses/BI7653/hw4.2022/cohort.g.vcf.gz \
-O hs29.vcf.gz
```

Q1.2 When your script has completed, report the first 20 lines of the output gzipped vcf using

$1 point$

:

```
gunzip -c hs29.vcf.gz | head -n 20
```

Q1.3 The GenotypeGVCFs command will produce a VCF file with both indels and snps. You can use a grep command that excludes header lines beginning with a "#" and extract only variant records from a VCF.

How many total variants are in the VCF file

$1 point$

?

```
gunzip -c hs29.vcf.gz | grep -c -v '^#'
```

Answer:: 91636

Q1.4 Answer the following

$1 point$

.

Q1.4a The HaplotypeCaller + CombineGVCFs + GenotypeGVCFs workflow addresses what is known as the n + 1 problem. What is the n + 1 problem?

Answer:: SNP caller program utilize all sample BAMs simultaneously in one step. if add one more sample to the dataset, it has to repeat the snp-calling program. This is problematic because this process is computational intensive and takes time, and thus known as a 'N+1' problem.

Q1.4b If after completing your assignment your instructor provides you with an additional .gvcf file to include in your snp callset, which steps in the workflow would you need to re-execute to generate a VCF with all samples?

Answer:: the process can be started from the GVCF file for each sample, joint-called the GVCFs using GenotypeGVCFs, and yield an unfiltered VCF callset for the new dataset.

Task 2: Subset SNPs from VCF

The output of Task 1 is a Variant Call Format (VCF) file that contains both snps and indel ("insertion/deletion") polymorphisms. We will focus our analysis on snps so will need to create a snp-only VCF.

The following link has instructions for how to do this. Please skip the VQSR steps and proceed to section 2 "Hard filter a cohort callset with VariantFiltration" where you will find a command under "

A

Subset to SNPs-only callset with SelectVariants":

https://gatk.broadinstitute.org/hc/en-us/articles/360035531112--How-to-Filter-variants-either-with-VQSR-or-by-hard-filtering (https://gatk.broadinstitute.org/hc/en-us/articles/360035531112--How-to-Filter-variants-either-with-VQSR-or-by-hard-filtering)

Now create a new sbatch script to select only snps from the VCF created in Task 1 using the SelectVariants command mentioned in the link with appropriate adjustments made to it so it will work on your data.

Execute your script with sbatch when you are ready.

Q2.1 Paste the contents of your script into your answers file

$$1 point$$

.

```
#!/bin/bash
#
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=8:00:00
#SBATCH --mem=8GB
#SBATCH --job-name=slurm_template
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=xw2470@nyu.edu

module purge

module load gatk/4.2.4.1

gatk SelectVariants \
-V /scratch/xw2470/ngs.week4/task1/hs29.vcf.gz \
-select-type SNP \
-O snps.vcf.gz
```

Q2.2 Indel ("insertion/deletion") variants are important in many contexts including studies of frameshift mutations in protein-coding genes. Please review the VCF format specification for how indels are specified in VCF format in section 5, p. 13:

https://samtools.github.io/hts-specs/VCFv4.3.pdf (https://samtools.github.io/hts-specs/VCFv4.3.pdf) For each VCF-encoded variant below, answer the following.

For each of Is the variant a SNP or indel? If it is an indel, is the reference or the alternate allele the deletion allele? If it is an indel, how many bases are deleted relative to the insertion allele? If it is an indel, for each allele, which base is found at the genomic position in the POS column

$$1 point$$

?

```
#CHROM    POS   ID  REF ALT     QUAL  FILTER  INFO  <additional columns not shown>
20        20    .   AT  A       .     PASS    DP=100
20        10    .   C   G       .     PASS    DP=100
20        20    .   C   CATATAT .     PASS    DP=100
```

Answer::

```
variant 1:    indel   alternate allele is the deletion allele     1 base is deleted       A
variant 2:    SNP       --
variant 3:    indel   reference allele is the deletion allele     6 bases are deleted     C
```

Task 3: Hard filtering SNPs

Creating a high quality snp callset is a challenging task. In a snp-calling and genotyping analysis for production (e.g., publication or clinical application) we would conduct additional steps of *Base Quality Score Recalibration (BQSR)* and *Variant Quality Score Recalibration (VQSR)* to produce the highest possible quality SNP callset.

BQSR uses comparisons to calibrate PHRED quality scores associated with each base in the raw reads (after they have been aligned to the reference). This allows accurate calculation of error probabilities for SNPs and genotypes (uncalibrated base qualities lead to inaccurate error probabilities). As mentioned in the pre-recorded video, VQSR is a machine learning procedure that takes an input database of known SNPs and outputs a score for each SNP in the SNP callset that can be used to filter the callset based on a desired false positive rate. We are not performing these steps, but they are important steps in generating publication quality outputs for human data.

Even when VQSR is conducted, the GATK still recommends *eliminating outlier SNPs using hard filtering*. Unfortunately, this requires deciding on thresholds on features like read depth. For example, SNPs with very high or very low read depths (summed across samples) are often removed. Other common filters include removing strand-biased SNPs, where reads predominantly map to either plus or minus strand of the reference (they should map roughly 50:50), or SNPs that deviate strongly from Hardy-Weinberg Equilibrium (e.g., a deficit of heterozygotes often due to low sequencing depth). The impact of a set of hard filtering thresholds on output data quality is rarely known with any degree of accuracy and it remains a major area of uncertainty especially when analyzing non-model organism snp data for which VQSR is not possible (because a set of independently acquired high quality SNP data necessary for VQSR is not available).

You can read more here:

https://software.broadinstitute.org/gatk/documentation/article?id=11069 (https://software.broadinstitute.org/gatk/documentation/article?id=11069)

Now you will create a job submission script to conduct hard-filtering on your raw snp callset produced in Task 2. Return to the following webpage and skip to Section 2: "

$$C$$

Hard-filter SNPs on multiple expressions using *VariantFiltration*"

https://gatk.broadinstitute.org/hc/en-us/articles/360035531112--How-to-Filter-variants-either-with-VQSR-or-by-hard-filtering (https://gatk.broadinstitute.org/hc/en-us/articles/360035531112--How-to-Filter-variants-either-with-VQSR-or-by-hard-filtering)

Create a job submission script to execute the VariantFiltration command at the above website on your SNPs-only VCF from above then execute your script.

Q3.1 Paste the contents of your script here

$$1 point$$

.

```
#!/bin/bash
#
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=8:00:00
#SBATCH --mem=8GB
#SBATCH --job-name=slurm_template
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=xw2470@nyu.edu

module purge

module load gatk/4.2.4.1

gatk VariantFiltration \
-V /scratch/xw2470/ngs.week4/task2/snps.vcf.gz \
-filter "QD < 2.0" --filter-name "QD2" \
-filter "QUAL < 30.0" --filter-name "QUAL30" \
-filter "SOR > 3.0" --filter-name "SOR3" \
-filter "FS > 60.0" --filter-name "FS60" \
-filter "MQ < 40.0" --filter-name "MQ40" \
-filter "MQRankSum < -12.5" --filter-name "MQRankSum-12.5" \
-filter "ReadPosRankSum < -8.0" --filter-name "ReadPosRankSum-8" \
-O snps_filtered.vcf.gz
```

Q3.2 Review the output VCF.

```
gunzip -c snps_filtered.vcf.gz | grep -v '^#' | less  # Type "q" to exit less when finished
```

Now answer the following

$1 point$

:

Q3.2a Report one record that passed the filtering criterion. What is the Depth of this variant across samples? What is the SNP quality? *Answer:*: depth = 248; snp quality = 713.57

```
1       17365   .       C       G       713.57  PASS    AC=1;AF=0.024;AN=42;BaseQRankSum=1.76;ClippingRankSum=0.00;DP=248;ExcessHet=0.0000;FS=0.000;InbreedingCoeff=
0.3699;MLEAC=1;MLEAF=0.024;MQ=49.31;MQRankSum=0.390;QD=9.03;RAW_MQ=194502.00;ReadPosRankSum=1.18;SOR=0.073  GT:AD:DP:GQ:PGT:PID:PL  0/0:42,2:44:42:0|1:17365_C_G:0,4
2,2415  0/0:13,0:13:36:.:.:0,36,540    ./.:0,0:0:0:.:.:0,0,0   0/0:2,0:2:6:.:.:0,6,66  0/1:55,24:79:99:.:.:725,0,2026  0/0:1,0:1:3:.:.:0,3,15  0/0:6,0:6:15:.:.:0,1
5,225       0/0:3,0:3:9:.:.:0,9,102 0/0:5,0:5:15:.:.:0,15,201       0/0:3,0:3:9:.:.:0,9,111 0/0:25,0:25:69:.:.:0,69,1035    0/0:13,0:13:36:.:.:0,36,540     0/0:7,0:
7:21:.:.:0,21,279       0/0:6,0:6:18:.:.:0,18,166       0/0:14,0:14:39:.:.:0,39,585     0/0:11,0:11:33:.:.:0,33,425     0/0:4,0:4:12:.:.:0,12,138       ./.:0,0:0:0:
.:.:0,0,0   ./.:0,0:0:0:.:.:0,0,0   0/0:2,0:2:6:.:.:0,6,81  0/0:2,0:2:3:.:.:0,3,45  ./.:0,0:0:0:.:.:0,0,0   ./.:0,0:0:0:.:.:0,0,0   ./.:0,0:0:0:.:.:0,0,0   ./.:0,0:
0:0:.:.:0,0,0   0/0:3,0:3:9:.:.:0,9,94  0/0:2,0:2:6:.:.:0,6,48  ./.:0,0:0:0:.:.:0,0,0   0/0:2,0:2:6:.:.:0,6,47
```

Q3.2b Report one record that failed one or more filters then answer which filters did it fail? What is the threshold of the filter(s) that it failed and what is the value(s) for the filter for the SNP in question? *Answer:*: this record failed on MQ40 filter (MQ = 22.00) with the threshold MQ < 40.

```
1       16966   .       G       A       30.90   MQ40    AC=1;AF=0.026;AN=38;BaseQRankSum=-6.740e-01;ClippingRankSum=0.00;DP=81;ExcessHet=0.0000;FS=0.000;InbreedingC
oeff=0.3530;MLEAC=2;MLEAF=0.053;MQ=22.00;MQRankSum=0.00;QD=7.72;RAW_MQ=1936.00;ReadPosRankSum=-6.740e-01;SOR=0.693  GT:AD:DP:GQ:PL  0/0:1,0:1:3:0,3,43      0/0:7,0:
7:21:0,21,253   0/0:2,0:2:6:0,6,53      ./.:0,0:0:0:0,0,0       0/0:1,0:1:3:0,3,32      ./.:0,0:0:0:0,0,0       0/0:4,0:4:12:0,12,135   0/0:2,0:2:6:0,6,70      0/0:
4,0:4:12:0,12,122   0/0:2,0:2:6:0,6,53      0/0:18,0:18:54:0,54,651 0/0:8,0:8:24:0,24,280   0/0:4,0:4:12:0,12,152   0/0:4,0:4:12:0,12,145   0/0:6,0:6:18:0,18,196
0/1:2,2:4:41:41,0,41    ./.:0,0:0:0:0,0,0       0/0:3,0:3:9:0,9,131     ./.:0,0:0:0:0,0,0       ./.:0,0:0:0:0,0,0       0/0:2,0:2:6:0,6,74      0/0:4,0:4:12:0,12,15
5   ./.:0,0:0:0:0,0,0       ./.:0,0:0:0:0,0,0       0/0:4,0:4:12:0,12,136   ./.:0,0:0:0:0,0,0       ./.:0,0:0:0:0,0,0       ./.:0,0:0:0:0,0,0       0/0:1,0:1:3:0,3,
27
```

Q3.3 Create a job submission script with the following command line to remove SNPs that failed the filter criteria from the VCF.

```
gatk SelectVariants \
-R /scratch/work/courses/BI7653/hw3.2022/hg38/Homo_sapiens.GRCh38.dna_sm.primary_assembly.normalized.fa \
-V snps_filtered.vcf.gz \
--exclude-filtered \
-O snps_filrevmoved.vcf.gz
```

```
#!/bin/bash
#
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=8:00:00
#SBATCH --mem=8GB
#SBATCH --job-name=slurm_template
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=xw2470@nyu.edu

module purge

module load gatk/4.2.4.1

gatk SelectVariants \
-R /scratch/work/courses/BI7653/hw3.2022/hg38/Homo_sapiens.GRCh38.dna_sm.primary_assembly.normalized.fa \
-V snps_filtered.vcf.gz \
--exclude-filtered \
-O snps_filrevmoved.vcf.gz
```

How many snps are in your final filtered callset?

$1 point$

```
gunzip -c snps_filrevmoved.vcf.gz | grep -c -v '^#'
```

Answer:: 74265

You are finished, please review the Completing your assignment section above before submitting your report.