Variant Calling Workshop

Cesky Krumlov

January 8, 2020

Find the Data

We’re going to start with the same data we had from the alignment exercises. We will perform structural variant calling on the PacBio data and small variant calling on the Illumina data.

Mark Duplicates

Once you've merged all your bams together and indexed them, we're going to mark duplicates. We do this because sometimes you get exactly the same read or read pair more than once for technical reasons, but they're not unique looks at your underlying sample, so we only want to use one of the duplicate copies for analysis. Samtools has a duplicate marker, but even the samtools developers say that GATK's is better, so we're going to use that

gatk MarkDuplicates -I MERGEDBAM –O DEDUPBAM –M DUPMETRICS --CREATE_INDEX true

This should take about 2 minutes to run.

Note that it will write a new bam with no changes other than that the duplicates are marked (in the bit flag). We also need to provide a second file, because Picard will give us a text file detailing some metrics about the duplicate marking.

Base Quality Recalibration

We're going to skip base quality recalibration for this exercise.

The reason is that recalibrating base quality assumes that bases that don't match the reference are errors. When we do this for human sequencing, we generally have a long list (such as dbSNP) of known variant sites, and we don't count disagreements at those sites as errors. We don't have that for this yeast strain.

In practice, in a situation like this, the suggested practice is to run a full round of variant calling without recalibrating, then use the list of variant sites as known sites, then recalibrate quality score based on other sites, then cycle around and call variants again.

There is some debate in the field about whether recalibrating base quality even really matters for this depth of sequencing, and this process will take a long time, so we will skip it.

Adding Read Group Information

Before calling variants, GATK wants to know which reads came from the same library and the same machine run so that it can group those together for purposes of calling (and base quality recalibration, if we had done that).

This can be done as an argument to bwa when you run alignment, but if you get your data back from a sequencing core already aligned, you may need to do it yourself on the bam file, so we're going to see how to do that with GATK.

gatk AddOrReplaceReadGroups –I DEDUPBAM –O READGROUPBAM –LB SK1-1 –SM SK1 –PL Illumina –PU 1 –CREATE-INDEX true

This is the minimal information for haplotype caller to work, library, sample, platform, and platform unit (run). In reality, you might have multiple read groups in a data set and would need to have each run or library aligned separately to add the appropriate information.

This should take a few minutes to run and will write a new bam.

Sequence Dictionary

GATK requires one more thing we haven't done. We need a "sequence dictionary" for our reference that says how big all the chromosomes are and how they're ordered without having to read the whole fasta file. We make this with GATK:

gatk CreateSequenceDictionary –R reference/S288C.fa

This should create a small text file call S288C.dict in the reference directory (if this file is already there, you can skip this step).

Haplotype Caller

The following is the command for haplotype caller, where BAMFILE is your bam file with read groups added and GVCFFILE is the name of your gvcf file. It is important that your vcf file end in .g.vcf.

gatk HaplotypeCaller -R reference/S288C.fa -I BAMFILE -O GVCFFILE –ERC GVCF

This is going to take about 20-30 minutes. Feel free to take a break once you know it’s running.

Variant Genotyping

Haplotype Caller made a first pass at calling variants, creating the gvcf file. If we had multiple samples that were from the same population, we could merge multiple gvcfs together to jointly genotype them, but since we only have one, we're just going to run the genotyper on our one vcf to get a set of alternate calls only. Note that VCFFILE should end in .vcf.

gatk GenotypeGVCFs -R reference/S288C.fa -V GVCFFILE –O VCFFILE

This runs in a few minutes.

Variant Filtering

We have one last step to do, which is to filter the variants. As we discussed, one way to do this is by variant quality recalibration (VQSR), but we don’t have enough variants in this data set to run that (plus it takes a really long time), so we’re instead going to use a set of standard parameters that are considered “best practices” for human genomes (in other words, if you actually ran VQSR, chances are good you would get something very much like this). Because this cammand is so long, it’s on the next slide, and I've spread it across several lines, but it should be run on a single line.

Variant Filtering

*gatk4 VariantFiltration -R human_g1k_v37.fa *
*-V VCFFILE *
*-O FILTEREDVCF *
*-filter “QD < 2.0” --filter-name “QDfilter” *
*-filter “MQ < 30.0” --filter-name “MQfilter” *
*-filter “FS > 60.0” --filter-name “FSfilter” *
*-filter “SOR > 3.0” --filter-name “SORfilter” *
*-filter “ReadPosRankSum < -8.0” --filter-name “RPRSfilter” *
*-filter “MQRankSum < -12.5” --filter-name “MQRSfilter” *
--verbosity ERROR

Note:
The backslash (”\”) character when typed at the end of a unix line will tell the shell to treat the next line as part of that line. If you type this all on a single command line, don’t put that in.

You can leave out the verbosity option, but the homozygous variants generate warnings because they don’t have the rank sum metrics, so I recommend leaving it there.

Variant Filtering

You may get a lot of warnings, but you should get out a new vcf. To see if anything happened, you can try:

grep filter FILTEREDVCF | less

You should see a large number of filtered out variants.

Calling SVs on the PacBio Reads

We're going to use a program called sniffles2 to call structure variants on the long read data.

sniffles PBBAM -v PBSVVCF

This should run pretty quickly and produce a VCF with large insertions and deletions in it.

Looking at Your Calls

Now we can launch IGV again and load in the variant calls and see how they look compared to the read data we aligned. For this, you can follow the instructions from the alignment workshop to reload the genome annotations and your bams. IGV should remember you used the yeast genome last time and load that for you automatically. You also don't need to run igvtools again, as your read depth data files should still be there. (Don't worry about loading the new deduped and read group bams, as the underlying data are still the same, but you can if you want to; you will need to run igvtools Count feature on that file since we didn't do that before.)

Then you can go to File->Load from file and load your 2 VCF files, and you should be able to see marks where the variant callers called variants and see what they look like in the reads. It might be particularly interesting to see the difference between PacBio and Illumina in the regions where sniffles made SV calls.