Variant (SNP/InDel) detection

Overview

Single nucleotide polymorphisms (SNPs) are one of the fundamental types of genetic variation, and with the growing popularity of next-generation sequencing, they are becoming the most ubiquitously utilized genetic markers in both evolutionary and biomedical research. An issue, however, is to separate true polymorphisms from sequencing or alignment artefacts, for which extensive filtering based on statistical models is necessary. There is to date no perfect way to do this, especially for non-model organisms, so it is always prudent to be stringent in filtering if unsure.

In this exercise, we will walk through the steps necessary to identify good quality SNPs from population-level next-generation sequencing data and we'll extract genotype information whenever possible from the SNP call dataset.

For all the data processing steps within this section, we have chosen to follow the recommendations of the Broad Institute, created for the Genome Analysis Toolkit (GATK): http://www.broadinstitute.org/gatk/guide/topic?name=best-practices

We highly recommend keeping an eye on the instructions of this site for more information and updated protocols. They also have an excellent forum for posting technical questions. The only step in their protocol that we do not use is the Base Quality Score recalibration, as this step requires a list of known variant sites as input. The focus of this course is on non-model organisms, for which there typically are no *a priori* known variant sites. If you do have access to this type of data, it is highly recommended to follow the instructions on the GATK site.

There are three major steps to this section of the protocol. First, we need to process our alignment files slightly. We start by merging all the deduplicated .bam files from section 4 into one file called `merged.bam`, which will be our base for SNP discovery. At this step, it is crucial that the "read group" headings for your samples (which we specified yesterday) are correct, as they will be used to keep track of the samples within the merged .bam file. We then index our merged .bam file and search through the file for areas containing indels, where the initial mapping might be of poor quality. By using information from all samples in the merged file in a realignment step, we improve our chances of correctly aligning these regions.

The merged realigned .bam file is what we will use in the next step, variant (SNP) detection and genotyping. An initial search for only very high-quality variant sites outputs a .vcf file, which is a list of all variant sites and the genotypes of all individuals for those sites. For information about the vcf file format, see (http://www.1000genomes.org/node/101).

For purposes of this exercise, we will consider the high-quality variants "true" sites for further processing. An additional search for variant sites, now with a lower quality threshold, is then conducted and by using our "true" variant sites from the first search we can build a Gaussian mixture model to separate true variants from false positives using a log-odds ratio (VQSLOD) of a variant being true vs. being false:

(http://www.broadinstitute.org/gatk/gatkdocs/org_broadinstitute_sting_gatk_walkers_variantrecalibration_VariantRecalibrator.html).

Following this, we can extract the genotype information for each individual from the .vcf file, while specifying a genotype quality threshold, and use this information to calculate allele and genotype frequencies. As Alex has covered, there are probability-based approaches available, which allow for a lower sequencing depth, but for simplicity we will today use Q=20 (p=0.99) as a threshold.

We can then use this data to e.g. conduct PCA and F_{ST} outlier analyses, as well as calculate any number of other standard population genetic metrics with software such as Genepop or DNAsp (not covered today).

Objectives

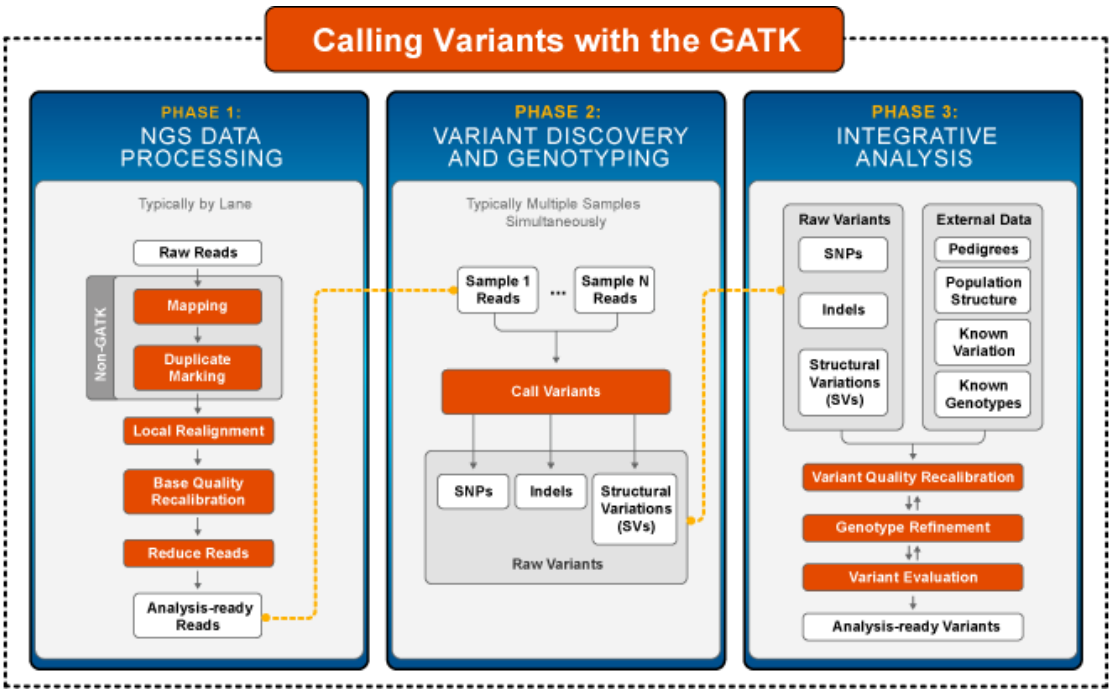
The objectives of this section are to 1) merge your alignment files and realign poorly mapped regions, 2) detect variant sites and filter out true sites from false positives, 3) extract genotype information for all individuals at all variant sites, 4) calculate allele and genotype frequencies.

Genome Analysis Toolkit (GATK): http://www.broadinstitute.org/gsa/wiki/index.php/The_Genome_Analysis_Toolkit : A collection of programs for searching through and manipulating .bam files; we use it for realigning .bam files, finding and filtering variant sites.

McKenna, A, Hanna, M, Banks, E, Sivachenko, A, Cibulskis, K, Kernytsky, A, Garimella, K, Altshuler, D, Gabriel, S, Daly, M, DePristo, MA. 2010. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Research* 20: 1297-1303.

DePristo, M, Banks, E, Poplin, R, Garimella, K, Maguire, J, Hartl, C, Philippakis, A, del Angel, G, Rivas, MA, Hanna, M, McKenna, A, Fennell, T, Kernytsky, A, Sivachenko, A, Cibulskis, K, Gabriel, S, Altshuler, D, Daly, M. 2011. A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nature Genetics* 43: 491-498.

Workflow (from GATK website)



Exercise

PHASE I

1) Merge your deduplicated .bam files into one file and index the merged file, using samtools, with the commands `samtools merge` and `samtools index`

a. Open the tab-delimited text file called `rg.txt`, which is located along with your data files. This file provides critical information for GATK to keep the individuals apart in the merged file. It should be formatted like this (new line for each sample):

@RG ID:READ_GROUP SM:SAMPLE_NAME PL:Illumina
….

b. Merge your deduplicated .bam files from this morning's exercise:

```
samtools merge -h rg.txt merged.bam *dedup.bam
```

c. Index your merged .bam file so that GATK will be able to search through it:

```
samtools index merged.bam
```

2) Realign around InDels using the GATK.

a. Open the `realigner.sh` script in nano and make sure that the path to your reference assembly (created in section 2) is correct, then re-save the script.
b. There are two parts to this script: 1) call on RealignerTargetCreator to search for poorly mapped regions near indels and save those intervals in an output.intervals file. 2) call on IndelRealigner to realign the intervals specified in step 1, and save the realigned file as
   `merged_realigned.bam`

c. Execute the script from the Terminal shell:

```
sh ../scripts/realigner.sh
```

PHASE II

Detect variant sites.

a. Open the `SNP_detection.sh` script in nano and make sure that the paths and file names are correct. Re-save the script.

   The script has three elements:

   1) It calls on the GATK HaplotypeCaller to only call variant sites with a Phred scale quality of more than 20 (probability of being a true variant site >0.99). This will be used as a set of "true" variant sites to train the Gaussian mixture model used by the Variant Quality Score Recalibrator (VQSR) in the next step. The VQSR depends on a set of true variant sites, so if you are working with an organism for which a validated set of variants exist, it is recommended to use that data here. However, as we are working with non-model organisms, we can not assume that this data will always be available so let's assume that we have no prior knowledge in this case. You will want the quality threshold to be as high as possible at this point, but with out limited dataset, we will have to settle for Q=20 as a threshold.

   2) The script then calls on the HaplotypeCaller to call SNPs with a threshold that is largely determined by the sequencing depth. As we have low coverage due to our truncated fastq files, we will use a low quality threshold here (Q=3). In reality, you would want to maximize this to reduce the chance of false positives.

   3) Finally, the script uses the VariantAnnotator to add annotations to the .vcf file output.

      The high-quality variant sites are stored in a file called: raw_snps_indels_Q20.vcf

While the variants that we will use for our final call set are in a file called: raw_snps_indels_Q3_annotated.vcf

b. Execute the script from the Terminal:

```
sh ../scripts/SNP_detection.sh
```

This step will take a little while, so while the GATK is running, let's familiarize ourselves with the output format (.vcf). There is a sample .vcf file called "sample.vcf" in your data folder that you can open using nano or copy onto you local drive and open in another text editor.

The first section of the file is a header, containing among others the history of the file and a list of all the contig names in the assembly. The header lines begin with ##. The last header line only has one # and contains the header to the SNP call columns that follow.

For each line, the first nine columns contain information about the variant site (location and quality metrics) as well as a filter flag column. Columns 10 and above contain genotype information for each individual in the dataset. More information about the .vcf file format can be found at: *http://www.1000genomes.org/wiki/Analysis/vcf4.0*

c. View the bottom 10 lines of the annotated Q3 .vcf file with:

```
tail raw_snps_indels_Q3_annotated.vcf
```

And try to find the genotype quality scores (GQ) for each individual at the last few SNPs in the file. These are what we are interested in extracting at the end of the day. But first we need to filter out false positives.

PHASE III

Variant Quality Score Recalibration (VQSR)

a. Open the `VQSR.sh` script in nano and make sure that all paths and file names are correct. Re-save the script. The script has five elements:

1) It uses the high-quality SNP dataset to train a model that can be used for filtering the true SNPs from false positives in our call dataset;
2) It uses the high-quality InDel dataset to train a model that can be used for filtering the true InDels from false positives in our call dataset;
3) It applies the SNP model to the call data and flags all SNPs failing the filter;
4) It applies the InDel model and flags all InDels failing the filter;
5) It saves only the variant sites that have passed the VQSR into a new file called `VQSR_PASS_SNPS.vcf`

b. Execute the script from the Terminal by typing:

```
sh ../scripts/VQSR.sh
```

It is quite likely that the VQSR will produce an error message due to the fact that we have very few SNPs in our call set with which to train the model. This is unfortunate, but can be circumvented in most cases by lowering the sensitivity of the model.

So, if you get this error message:

ERROR MESSAGE: NaN LOD value assigned. Clustering with this few variants and these annotations is unsafe. Please consider raising the number of variants used to train the negative model (via --percentBadVariants 0.05, for example) or lowering the maximum number of Gaussians to use in the model (via --maxGaussians 4, for example).

Try changing the settings for -percentBad, -minNumBad and --maxGaussians in the first two commands of VQSR.sh using nano, then re-saving and re-running the script.

Genotype extraction

1) Extract genotypes in variant sites shared by all individuals:

Extract genotypes of all individuals at all variable sites from the .vcf file into a format useable by Microsoft Excel, using a genotype quality threshold.
S: `getgenosfromvcf.py`
I: `VQSR_PASS_SNPS.vcf`
O: `shared_genotypes.txt`

a. Execute the python script by typing:

```
python ../scripts/getgenosfromvcf.py
VQSR_PASS_SNPS.vcf Genotypes.txt rows 20
```

The final argument specifies a genotype Phred quality score cutoff of 20 (99% probability of being true). This parameter can be changed according to your needs. The "rows" argument specifies that SNPs will be output in rows, with two columns per individual, one for each allele (specifying "cols" would return the same output, but with SNPs as columns, two columns per SNP).

b. Once the script has finished, we can create a new file containing only the variant sites for which we have genotype information for all individuals

(all lines that do not contain any "."). We can easily do this with the bash command 'grep':

```
grep -v "\." Genotypes.txt > genotypes_shared_by_all.txt
```

It is often a good idea to focus on the best quality SNPs when working with next-gen sequence data. However, many analyses do not require data for all individuals at every SNP.

To get an idea of the amount of SNP data produced from this small dataset, note how many SNPs are present in the `Genotypes.txt` file (number of rows -1), as well as how many SNPs are present in the `genotypes_shared_by_all.txt` file, using the `cat -b` command.

```
cat –b Genotypes.txt

cat –b genotypes_shared_by_all.txt
```

c. `genotypes_shared_by_all.txt` can also be copied to your local drive and opened in Microsoft Excel to visualize the genotype data. Although it is not typically practical to work with next-gen sequencing data using Excel, it can be a useful format to get comfortable with the data as well as to calculate and plot allele and genotype frequencies.

d. Use Excel to calculate the alternative allele frequencies for the SNPs in the `genotypes_shared_by_all` file, and plot a frequency histogram. In a real dataset, we would expect there to be a logarithmic decrease in loci with increasing allele frequency. But of course, in our small test dataset this relation ship might be obscured by random factors, and also since SNPs with low minor allele frequencies are more difficult to detect.