Next Generation Sequencing Technologies: Data Analysis and Applications

VCF Files

Dr. Riddhiman Dhar, Department of Biotechnology

Indian Institute of Technology Kharagpur

Good day, everyone. Welcome to the course on generation sequencing technologies, data analysis, and applications. We have been discussing variant calling. So, in the last few classes, we have discussed methods on how to call variants, different types of variants, and we specifically focused on single nucleotide polymorphisms, or SNPs or SNVs, and small indels. So, we also tested with a script how to call these SNPs or SNVs in small indels from the same output file through a multi-step process. So, in this class, I will just summarize what we did in the last class hands-on and then we will talk about visual verification of the variant calls.

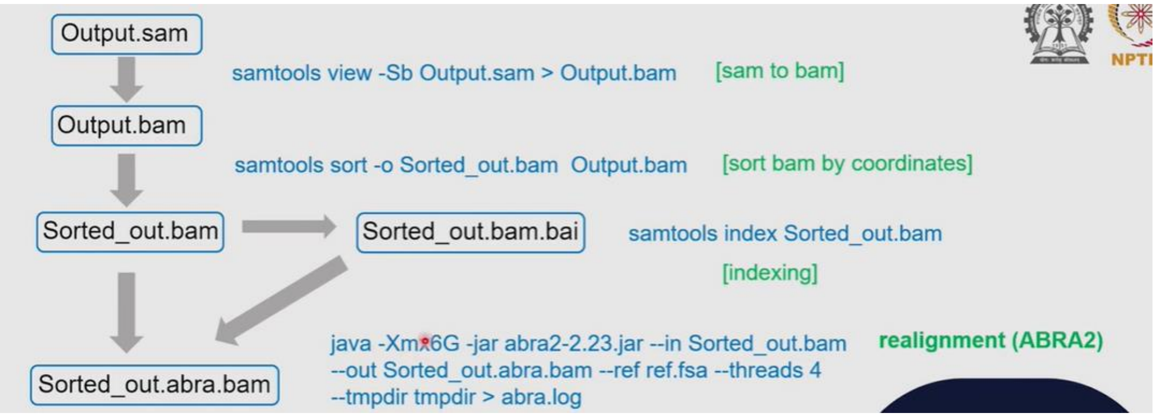
So, we can actually verify whether the variant calls are actually correct visually by looking at the alignment; there are tools for that, and then we will talk about the VCF file format. So, if you remember right, in the last class we got the output in this VCF file format. So, we will discuss what this format is, what fields are there, and what kind of information you can find in that file. So, these are the keywords that we will come across in this presentation.

So, VCF integrated a genome browser and data lines. So, briefly, I will summarize the steps that we used in the last class to call variants using SAM tools and BCF tools. There are other tools, but we use SAM tools and BCF tools to do the variant calling, and here are the steps I have just listed from the last class. So, again, you do not have to remember, but if you are following this pipeline, you will probably have to go step by step. So, one is that we first converted this SAM file to a BAM file binary format, and then we sorted this binary format, and this conversion and sorting are required for the variant calling. And in the third part, we did something called local realignment; we used a specific tool to do so, and then followed by that, we actually marked duplicates, and again, we discussed why duplicates are important, why you have to mark duplicates, etcetera.

The fifth step we did not do, but this is something you might want to do in cases where you have a lot of SNP data, something called base quality score recalibration. It is a time-consuming step. So, that is why it is not possible for us to cover this in this class or to do hands-on in this class, but we discussed the idea of why we do this and also mentioned the software that will do this for you. And then the final variant call. This is the actual variant call where we identify those variants. Again, just a flow chart of the scheme. So, we have this output dot SAM right. We use this samtools view command with these options, and the ultimate outcome is that we convert this SAM to BAM, which is why we gave this output dot BAM file.

So, I am actually mentioning the actual file names that we use. so that you can follow and go back and see the files yourself. Then we did the sorting using this command: SAM tool sort. Again, you have the option to specify the output file name and input file name, and this will sort by coordinates. So, this is the default sorting. So, it will sort by the coordinates or positions of the mappings, and this will give you this sorted-out dot BAM.

Now, once we have this sorted out dot BAM, we also generate this index file. So, we will see why this index file is important. It is also important for visualization. So, to sort out dot BAM dot bai, we use this function SAM tools index and this command SAM tools index to sort out dot BAM right. So, we are creating this index file for this sorted-out dot BAM file. Then we use these two files for realignment using this tool called ABRA 2 or ABRA 2 right.



```
samtools view -Sb Output.sam > Output.bam
```

[sam to bam]

```
samtools sort -o Sorted_out.bam  Output.bam
```

[sort bam by coordinates]

```
samtools index Sorted_out.bam
```

[indexing]

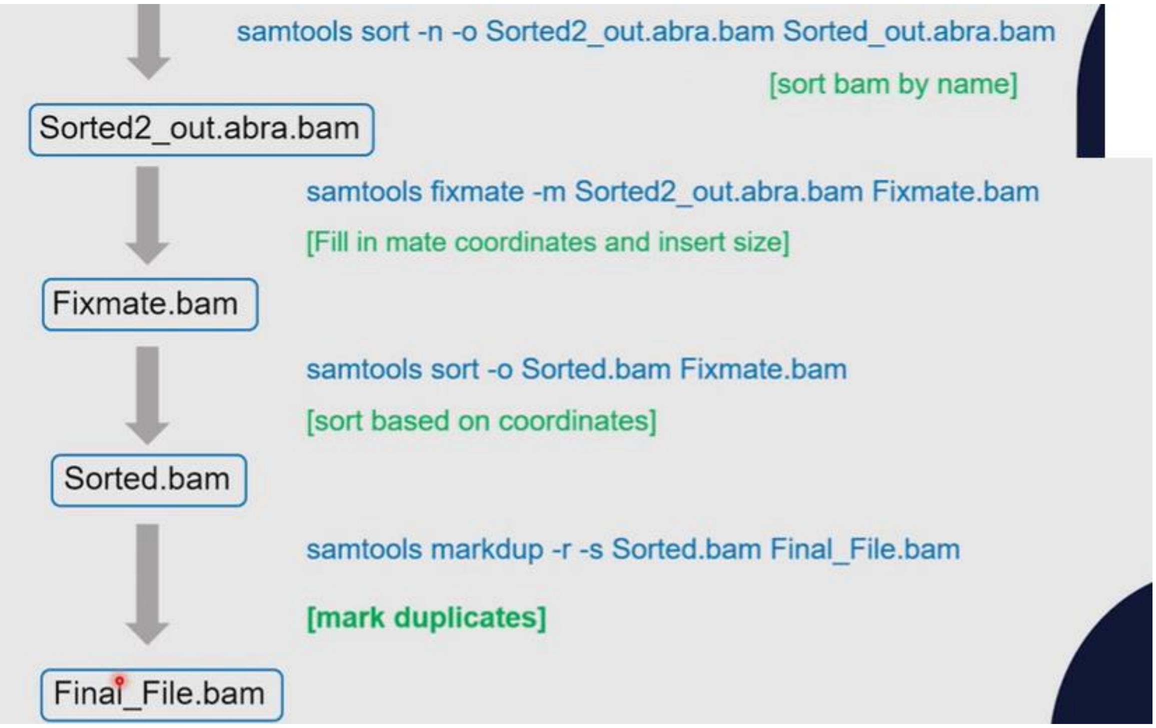
```
java -Xmx6G -jar abra2-2.23.jar --in Sorted_out.bam
--out Sorted_out.abra.bam --ref ref.fsa --threads 4
--tmpdir tmpdir > abra.log
```

realignment (ABRA2)

So, then we use this command specifically. Right again, I mention what this command actually means. So, we are actually calling Java here, and then we are limiting the memory usage to 6 GB using this command. Then we are calling the program version 2.23, and we are specifying the input file name; we are giving the output file name; and then we are also mentioning the reference file name, which is the reference sequence. Then the number of threads that will be used to run the processor on the computer, we have also created a temporary directory, and that is where this log will be generated.

So, we got this file sorted out dot ABRA dot BAM and then we again sorted by name, ok. So, these are actually required because of the tools that we are using; they can take data in certain formats after sorting, etcetera. So, this one we call sorted 2 out dot ABRA dot BAM, and this sorting is by name, not just by position. So, that is why this minus n option exists. Now what we want to do is mark duplicates, right?

So, these steps are preparing for marking duplicates. So, sorting by name means we are putting these pairs together and reading them side by side because the names of these pairs would be identical. So, the next step is actually using something called Samtools Fixmate. So, this will fill in the met coordinates and also fill in the insert size, and we created this file name called Fixmate Dot BAM, and then again, we sorted after we had marked the mates. So, we then sorted by the coordinates (default sorting in Samtools), and we created this sorted dot BAM.



```
samtools sort -n -o Sorted2_out.abra.bam Sorted_out.abra.bam
```

[sort bam by name]

```
samtools fixmate -m Sorted2_out.abra.bam Fixmate.bam
```

[Fill in mate coordinates and insert size]

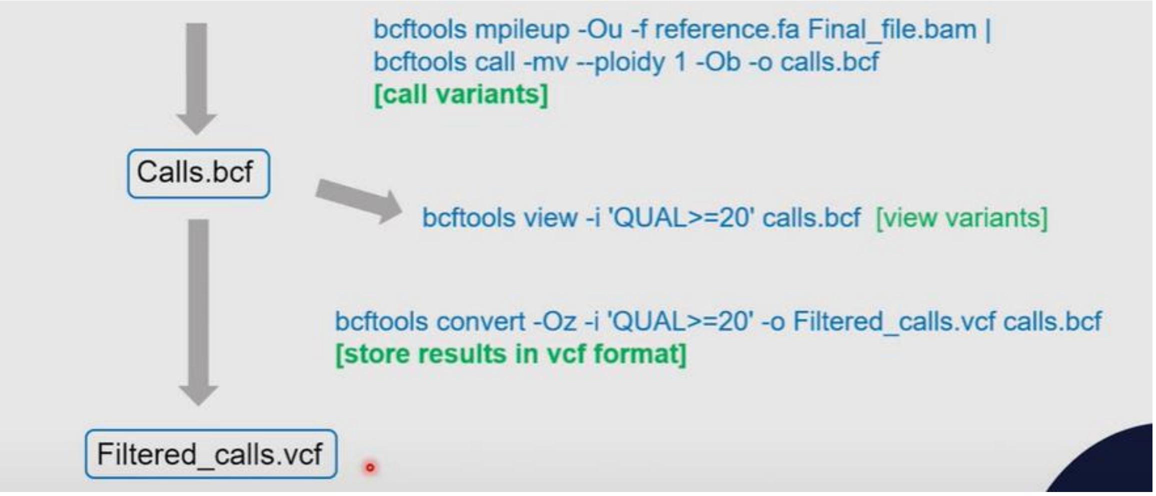
```
samtools sort -o Sorted.bam Fixmate.bam
```

[sort based on coordinates]

```
samtools markdup -r -s Sorted.bam Final_File.bam
```

[mark duplicates]

This is where we are marking the duplicates; actually, all those steps before preparing for this command are okay. So, we then use this command samtools mark to minus r minus s sorted dot BAM, and we create this final file dot BAM. So, the marking is done, the realignment has been done, and we did not do the base quality recalibration as we mentioned. So, what we did next was call the variance OK, and this is where we used this command: bcftools m pileup. So, it is collecting all the data from the final file, and then followed by that, we have this bcftools call.



```
bcftools mpileup -Ou -f reference.fa Final_file.bam |
bcftools call -mv --ploidy 1 -Ob -o calls.bcf
```

[call variants]

```
bcftools view -i 'QUAL>=20' calls.bcf
```

[view variants]

```
bcftools convert -Oz -i 'QUAL>=20' -o Filtered_calls.vcf calls.bcf
```

[store results in vcf format]

So, this is where the variant calling is happening, and we have to mention minus-minus ploid 1 because we are working with the haploid genome. So, this is a haploid yeast, but in case you are working with a diploid genome, this will be 2, and by default, this is 2. And then we stored these variant calls in this calls dot bcf file, ok? And we can see these bcf files, and we can also filter out some of these variants using this quality argument, right? We want high-quality variants. So, we can use this argument: qual greater than equal to 20.

And then finally, we stored the results using this bcftools convert option, and from this calls dot bcf file, we stored these filtered calls dot vcf in vcf format. And this is the final file filtered calls dot vcf, and we actually had a look also into that file for this file contents, etcetera. The commands or tools that we used are so, I will just list them here some tools viewsort pixmecmark2, and we also use the tools after 2. So, you can go and see the options on the manual page, right? These options mean that we used and kind of read about all the other options that are also out there. And similarly for bcftools, we use these commands in mpileup, call view, convert, etcetera.

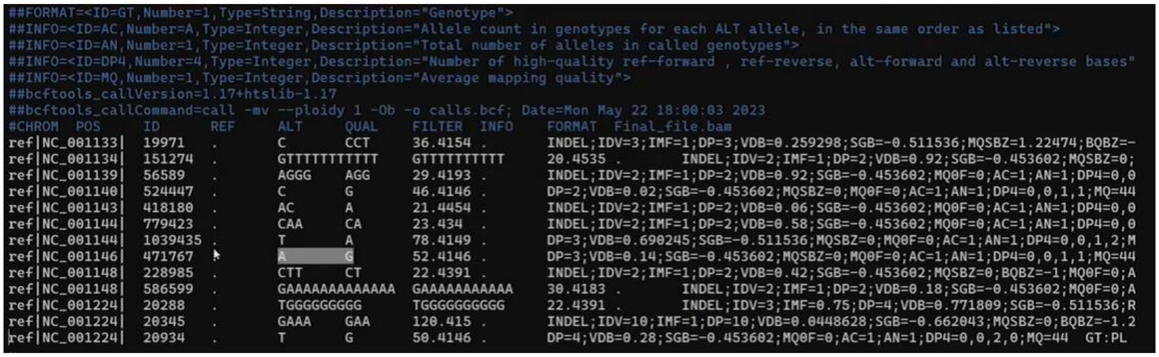
So, again, all these commands come with their own options, and you have to go through these options and see which one actually applies in your case. Again, doing that in the class is not possible. So, what I did was just collect all those commands, and we did the hands-on, but I encourage you to explore this bit more. So, what is remaining now is actually looking at the data and also doing a visualization and a manual check to see whether the variants that are called are correct or not. So, for that, we will use this tool called the integrated genome browser, or igb. It is a free tool you can download and install on your system, and you can load and visualize these results.

So, that is what we are going to do now, and then we will follow up on our discussion on these VCF file formats. So, we have created these VCF files, and along the way, you also solved one VCF file. So, vcf and vcf, what do these files look like, and what is the specific format? We will discuss that after this, ok? So, let us now go into the visualization part. Let us run the integrated genome browser and also let us look at the vcf file that we generated. So, let us move on to the terminal, and ah.

So, I have just listed the files that we created last time. So, using this command, ls minus l

So, you will see all these files are around, right? You see the sorted dot bam, sorted to out dot aviare dot bam, etcetera. All these files are here. The final bcf file would also be here, right? So,let us see, right, we have this if we go up, we see this filtered calls dot vcf ok.

So, this is a file that contains this variant information. So, let us have a look at this right here if filtered calls dot vcf and I just opened the file right and we again use this command colon se nowrap and we will come to the format in a moment right for this information is right ah. So, you have some header lines here, which again contain something called format information and also VCF tools, etcetera. We will discuss this in much more detail. The interesting part is that here you can see these are the variants.



```
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##INFO=<ID=AC,Number=A,Type=Integer,Description="Allele count in genotypes for each ALT allele, in the same order as listed">
##INFO=<ID=AN,Number=1,Type=Integer,Description="Total number of alleles in called genotypes">
##INFO=<ID=DP4,Number=4,Type=Integer,Description="Number of high-quality ref-forward , ref-reverse, alt-forward and alt-reverse bases"
##INFO=<ID=MQ,Number=1,Type=Integer,Description="Average mapping quality">
##bcftools_callVersion=1.17+htslib-1.17
##bcftools_callCommand=call -mv --ploidy 1 -Ob -o calls.bcf; Date=Mon May 22 18:00:03 2023
#CHROM  POS      ID      REF     ALT     QUAL     FILTER  INFO    FORMAT  Final_file.bam
ref|NC_001133|  19971    .       C       CCT      36.4154 .       INDEL;IDV=3;IMF=1;DP=3;VDB=0.259298;SGB=-0.511536;MQSBZ=1.22474;BQBZ=-
ref|NC_001134|  151274   .       GTTTTTTTTTTT     GTTTTTTTTTT     20.4535 .       INDEL;IDV=2;IMF=1;DP=2;VDB=0.92;SGB=-0.453602;MQSBZ=0;
ref|NC_001139|  56589    .       AGGG    AGG      29.4193 .       INDEL;IDV=2;IMF=1;DP=2;VDB=0.92;SGB=-0.453602;MQ0F=0;AC=1;AN=1;DP4=0,0
ref|NC_001140|  524447   .       C       G        46.4146 .       DP=2;VDB=0.02;SGB=-0.453602;MQSBZ=0;MQ0F=0;AC=1;AN=1;DP4=0,0,1,1;MQ=44
ref|NC_001143|  418180   .       AC      A        21.4454 .       INDEL;IDV=2;IMF=1;DP=2;VDB=0.06;SGB=-0.453602;MQ0F=0;AC=1;AN=1;DP4=0,0
ref|NC_001144|  779423   .       CAA     CA       23.434  .       INDEL;IDV=2;IMF=1;DP=2;VDB=0.58;SGB=-0.453602;MQ0F=0;AC=1;AN=1;DP4=0,0
ref|NC_001144|  1039435  .       T       A        78.4149 .       DP=3;VDB=0.690245;SGB=-0.511536;MQSBZ=0;MQ0F=0;AC=1;AN=1;DP4=0,0,1,2;M
ref|NC_001146|  471767   .       A       G        52.4146 .       DP=3;VDB=0.14;SGB=-0.453602;MQSBZ=0;MQ0F=0;AC=1;AN=1;DP4=0,0,1,1;MQ=44
ref|NC_001148|  228985   .       CTT     CT       22.4391 .       INDEL;IDV=2;IMF=1;DP=2;VDB=0.42;SGB=-0.453602;MQSBZ=0;BQBZ=-1;MQ0F=0;A
ref|NC_001148|  586599   .       GAAAAAAAAAAAAA   GAAAAAAAAAAA    30.4183 .       INDEL;IDV=2;IMF=1;DP=2;VDB=0.18;SGB=-0.453602;MQ0F=0;A
ref|NC_001224|  20288    .       TGGGGGGGGG       TGGGGGGGGGG     22.4391 .       INDEL;IDV=3;IMF=0.75;DP=4;VDB=0.771809;SGB=-0.511536;R
ref|NC_001224|  20345    .       GAAA    GAA      120.415 .       INDEL;IDV=10;IMF=1;DP=10;VDB=0.0448628;SGB=-0.662043;MQSBZ=0;BQBZ=-1.2
ref|NC_001224|  20934    .       T       G        50.4146 .       DP=4;VDB=0.28;SGB=-0.453602;MQ0F=0;AC=1;AN=1;DP4=0,0,2,0;MQ=44  GT:PL
```

So, again, this is the chromosome ID; right again, this corresponds to some chromosomes 1, 2 3 etcetera. Then you have this position, right? This is where in this chromosome you see this variant, the ID part, which belongs to ID. This dot means there is no information. Then the reference is c, and alternate means this is the variant that is observed compared to the reference sequence. So, alternate is observed is CCT ok and the quality score is 36.4 and they should be above 20 right we filtered with the 20 cut off and it says it is an indel ok.

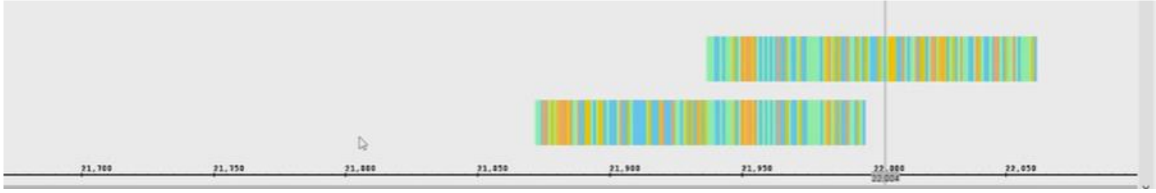
So, it is an indel, which means it is an insertion, like right. So, because in the reference we have c and in alternate you have cc t, you have this insertion here C 1 C and 1 T, and we should be able to see this in our data, ok? So, let us have a look at this, and then so on. You have all these interesting variants around here, and there are a lot of indels, and then there are some SNVs, right? You can see this SNV here: T to A or A to G, and again, their positions and the chromosomes are given OK. So, what we will do is go to the integrated genome browser, open up these alignments, and see whether we can actually see this variation and whether it is an artifact of alignment or not.

So, let us open an integrated genome browser. So, I will just show you right here that it is right. This is the site where you can download this integrated genome browser, and you can install it very easily on your system. You can just say install IGP. So, this will install, and you can download and install it on your system, ok? So, I have installed this here, ok, and we can open it. So, if you go on top right on the left side, let us see if we can increase the size a bit. Okay, anyway, I do not see this option here.

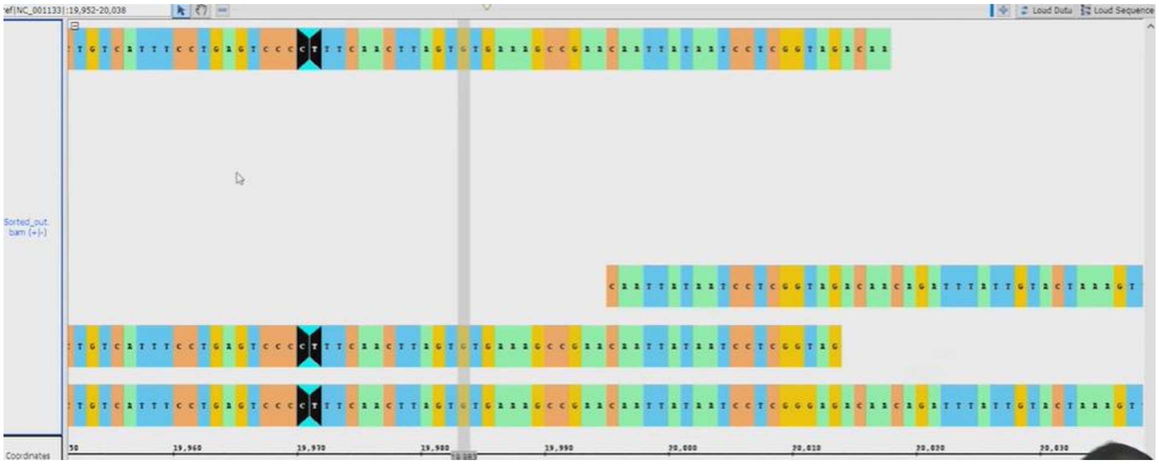
So, what you can do is I will just open here. If you click on the top, you will find multiple options. So, the one option that we need is the file option. So, with this file option, we have an open file and you can go to the directory where you have created this file. This align alignment and all the sorted alignment, etcetera, are right, and we will load this. Let us say this sorted out dot BAM, ok. This is a file we created during the variable calling process, and why are you choosing this because IGP requires an index file?

So, for any BAM file that you use, you also need to have an index file created. So, you can choose any BAM file, but for that, you will have to first create an index using the Sam Tools index command, and then you can load it here, ok? So, we have already created an index file for this one. So, let us load this, ok, and it has loaded, and it says you have to zoom in, and then you have to click on load data, ok. So, how do you zoom in here? Is this zooming right?

So, you can see where, like, going into this position, right? You can see this position, right?, and you can go to any position and load data here on this side. So, what you see now is very interesting, ok? So, here you see the read mappings, ok? So, each of these bars is right here. So, this is one read, and you can have multiple reads mapping to one position here.



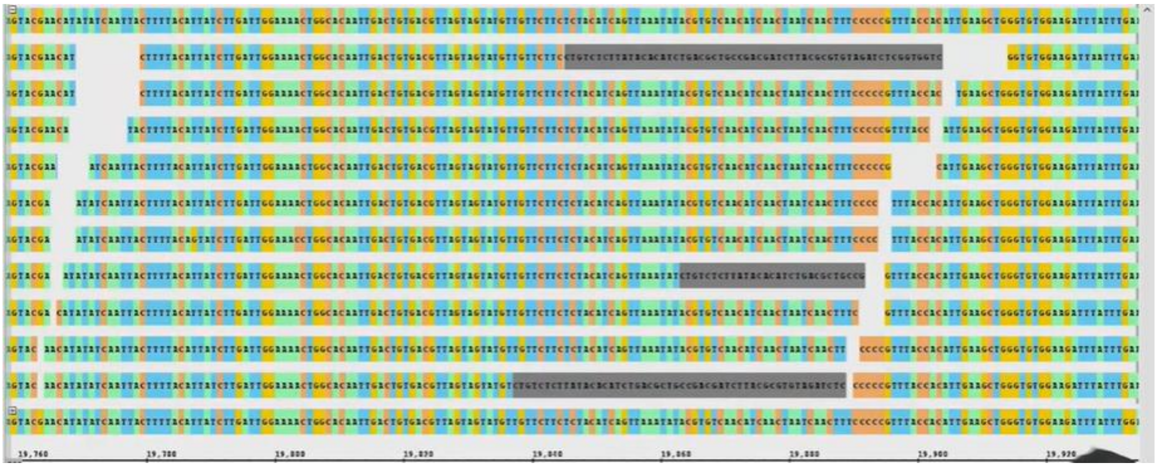
So, we can also zoom in further, and you can now see these reads in much more detail, ok, and there you see different colors because these are the bases that have been colored differently, ok, and you might be wondering why this is so sparse for some regions; there is no read mapping at all, right? You can see this for some regions. So, here is one region where you can see these two reads, but here on the left side there is no read mapping. So, remember, we are working with that small data set, where we only took about 25000 paired entries. So, it is kind of expected that 25000 reads will not be able to cover most of these positions. So, what we are going to do is let us look at this position where we saw this indel right.



So, what is the position again going back to 19971 in this reference, NC 001133? So, if you go back to IGB, right on the left-hand side, you can see these IDs, and we can expand this a bit, and you see the first one is this NC 001133, which also gives the length of that reference. So, we are on the right chromosome, but we need to find the right position. So, which is the 1999, right? So, let us scroll a bit right and see that 19900 and 91 are probably or 970 ok.

So, yeah, 971, right? So, this is the right position. So, as you can see here, right if we zoom in, yeah. So, you can zoom in, and you can see these bases now, right? So, in this position, right, you can see 19971. You have these 3 reads mapping, and each of them is showing these insertions. You can see these signs right, so you can go there, put your mouse, and it will show you these are the types of insertions. You have CT base insertions, ok? So, all 3 reads are showing this insertion, ok?

So, this is how we can actually verify if this is correct and if the alignment is also okay, etcetera. You can also load the reference sequence. You can also load the VCF file here. So, one thing I will show you is right. So, we can zoom out a bit and see the read mapping for these positions, and we can actually do what we can do. We can actually load the full mapping that I have actually done, and I can show you what this full mapping would look like. So, it will look much more complete in these positions, ok?

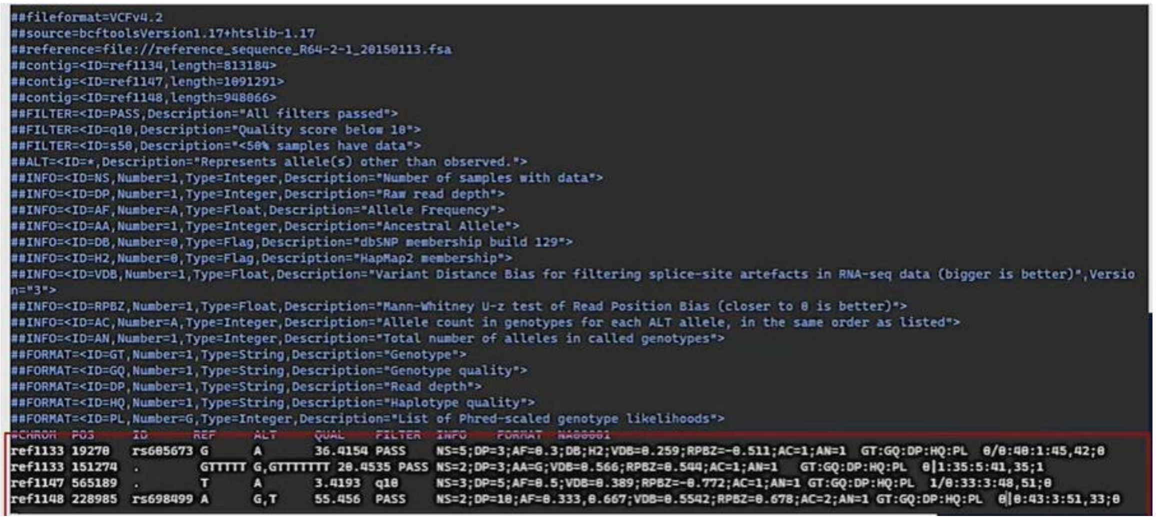


So, let us load that. I will load that file for you, and this is the full output I have generated. I have also sorted and created an index dot m dot bai. So, I will simply load this, and now that it has loaded, it is on top. So, I will just image this one, ok, and I will zoom in. Probably we can zoom out this one, and I can zoom in and load it, ok. So, this is the full file with full output data, and as you can see, it will look much more complete here, ok? So, here is this file, which was only part of it. Right here is the one with all the reads. OK, that I have already aligned, etcetera, and generated the map, and you can see it looks much more complete. Ok, I will probably load a bit more data, and you can see this clearly. Right right you have these reads now mapping to almost all positions with multiple reads.

And again in this one, if you look at the same position right here in 1900-971, you see this insertion CT insertion is there. You can see this clearly; this is also marked for you. So, I will zoom in a bit more, and you can see this CT insertion again, ok? So, this is how we can verify at least some of these AH variances that we see in the vcf file and be sure that, ok, we are calling them the right variance and that these are not artifacts of alignment or something else. So, if this is clear now, we can move on to the VCF file format and discuss what we have in there, how we can read it, and what kind of information we get in the VCF file. So, I will first go into the AH presentation, and then we will again have a look at the VCF file.

So, we have seen two file formats: one is vcf, which is called the variant call format OK. So, this is a format that we can read ourselves, and it also has a binary version, which is called the VCF format. So, this is the machine-readable format, but we can read it or view the contents using these VCF tools. So, we have done we have done this in the hands on. So, let us now talk about the vcf files and their contents.

So that we can read and understand what we have in there, So, this is important for the subsequent analysis. So, a vcf file is a text file that contains information about variants and has SNPs, small indels, and SPs. So, again, depending on the analysis and the sample that you are working with, you will have different types of information.



```
##fileformat=VCFv4.2
##source=bcftoolsVersion1.17+htslib-1.17
##reference=file://reference_sequence_R64-2-1_20150113.fsa
##contig=<ID=ref1134,length=813184>
##contig=<ID=ref1147,length=1091291>
##contig=<ID=ref1148,length=948066>
##FILTER=<ID=PASS,Description="All filters passed">
##FILTER=<ID=q10,Description="Quality score below 10">
##FILTER=<ID=s50,Description="<50% samples have data">
##ALT=<ID=*,Description="Represents allele(s) other than observed.">
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of samples with data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Raw read depth">
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=Integer,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##INFO=<ID=VDB,Number=1,Type=Float,Description="Variant Distance Bias for filtering splice-site artefacts in RNA-seq data (bigger is better)",Versio
n="3">
##INFO=<ID=RPBZ,Number=1,Type=Float,Description="Mann-Whitney U-z test of Read Position Bias (closer to 0 is better)">
##INFO=<ID=AC,Number=A,Type=Integer,Description="Allele count in genotypes for each ALT allele, in the same order as listed">
##INFO=<ID=AN,Number=1,Type=Integer,Description="Total number of alleles in called genotypes">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=String,Description="Genotype quality">
##FORMAT=<ID=DP,Number=1,Type=String,Description="Read depth">
##FORMAT=<ID=HQ,Number=1,Type=String,Description="Haplotype quality">
##FORMAT=<ID=PL,Number=G,Type=Integer,Description="List of Phred-scaled genotype likelihoods">
#CHROM  POS     ID        REF    ALT       QUAL     FILTER  INFO     FORMAT   NA00001
ref1133 19270   rs605673  G      A         36.4154  PASS    NS=5;DP=3;AF=0.3;DB;H2;VDB=0.259;RPBZ=-0.511;AC=1;AN=1  GT:GQ:DP:HQ:PL  0/0:40:1:45,42;0
ref1133 151274  .         GTTTTT G,GTTTTTTT 20.4535 PASS   NS=2;DP=3;AA=G;VDB=0.566;RPBZ=0.544;AC=1;AN=1   GT:GQ:DP:HQ:PL  0|1:35:5:41,35;1
ref1147 565189  .         T      A         3.4193   q10     NS=3;DP=5;AF=0.5;VDB=0.389;RPBZ=-0.772;AC=1;AN=1 GT:GQ:DP:HQ:PL  1/0:33:3:48,51;0
ref1148 228985  rs698499  A      G,T       55.456   PASS    NS=2;DP=10;AF=0.333,0.667;VDB=0.5542;RPBZ=0.678;AC=2;AN=1 GT:GQ:DP:HQ:PL  0|0:43:3:51,33;0
```

It will contain ah three parts mostly. So, you have one part, which is called the meta information lines, then you have one header line and then you have the data lines. So, you can imagine right, this is like a SAM file we have discussed. For example, you have header lines, then the alignment lines, and similarly, a VCF file is also formatted like this: you have meta-information lines, you have one header line, and then you have the data lines. So, here is an example of a vcf file, and in the first part, you can see the meta information line in blue. So, this is what I have highlighted: this is a meta-information line. Ah, you have a lot of information. Again, we will discuss what this meta-information line actually contains. Then you have the data lines. Okay, here is the data about the variants. Again, we will discuss what these mean. You see a lot of things and a lot of information and we will discuss what it means.

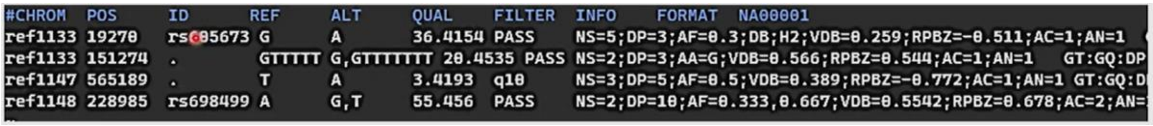
And finally, you have just one line. This is the header line. This actually contains the header for the data lines. Okay, again, we will discuss this header line. So, data lines are tab-delimited. So, they are tab-separated text lines, and each line contains 8 mandatory fields, OK, and the header line mentions the names of these 8 mandatory fields. So, it is just saying, OK, what do you have in the data lines?

So, what are these headers that you see? They are chromosomes. So, this is the chromosome number you see in this hash, if you notice the example. So, it is a chromosome number; this is the first field; the second is the position; this is the reference position where the variant is present. Then you have the third one, which is the ID or an identifier. So, this is something like looking at some databases and identifying whether this variant is present in another database. It has been given an ID, and this ID is then linked here. Then, fourth, you have the reference space, the base that is present in the reference sequence, right?

So, this is the reference sequence that is mentioned. The fifth field is the ALT, or alternate field, that contains the alternate bases. This is actually the variants that we see in our data. Then 6 is the quality, right? So, this is the PHRED scale quality score for ALT right or the alternate base, and this is related to the probability that the call in the alternate base is wrong. So, again, very similar to the PHRED score, as we have talked about, then you have the seventh position, which is the filter, which is the filtering status, and this will mention pass if the variant passes all filters, right?

So, we have talked about these different types of filters that you can set when we are doing variant calling. So, this field will describe whether the variant passes all filters that have been set or if there is some filter where the variant is not clear. So, the variant does not clear the filter that will be mentioned. Then you have the info fields. Ok, again, we will discuss what these info fields mean, and this will also be described in the meta information lines.

In the data lines, you have some optional fields as well. So, these optional fields include format fields. Again, we will discuss what these format fields are in detail, and this information is contained in the meta information line. So, that is why I thought maybe it is better to first discuss the data lines and describe what you have, and then go to the meta information lines otherwise, it might not be clear. In those data lines, in the optional fields, you can also have the genotype fields OK, and you will see these two signs: one for unfazed genomes and another for phased genomes OK. We have talked about phasing in when you are talking about the variant calling analysis pipeline, and this is again applicable for deployed genomes, right? So, for example, if you are working with human genome data, you will have this phasing information there, ok?



```
#CHROM  POS     ID        REF    ALT        QUAL    FILTER  INFO     FORMAT  NA00001
ref1133 19270   rs605673  G      A          36.4154 PASS    NS=5;DP=3;AF=0.3;DB;H2;VDB=0.259;RPBZ=-0.511;AC=1;AN=1
ref1133 151274  .         GTTTTT G,GTTTTTTT 20.4535 PASS    NS=2;DP=3;AA=G;VDB=0.566;RPBZ=0.544;AC=1;AN=1   GT:GQ:DP
ref1147 565189  .         T      A          3.4193  q10     NS=3;DP=5;AF=0.5;VDB=0.389;RPBZ=-0.772;AC=1;AN=1 GT:GQ:D
ref1148 228985  rs698499  A      G,T        55.456  PASS    NS=2;DP=10;AF=0.333,0.667;VDB=0.5542;RPBZ=0.678;AC=2;AN=
```

Then you can also have sample information, right? So, you can describe some of these characteristics of this format for each of these samples that you have. This is again going to be there, ok? So, here is an example of a data line, as you can see. So, first, you have the chromosome number. So, instead of a chromosome number, you have some reference ID, again referring to certain contigs or chromosomes. If you have positions where the variants are seen, then you have ID OK, and this ID is related to some of the database information that is present.

Then you have the references reference space, and then you have the alternate base. This actually contains information about the variant that is present. Then you have the quality score, and you can see these quality values that are here. Then you have the filter information again. Pass means the variant passes all filters that are present, but in one example here, you can see this Q-10 value right. So, Q 10 probably means the quality score of the variant should be above 10, but this variant does not pass.

So, you can see the quality score is below 10, right? So, it is something like 3.4 or so. Then you have some info fields ok. You can see in this NS, DP, AF, etcetera, different characters and then you have different numbers, ok, and you might be wondering what these actually mean.

So, all these terms will be explained in the meta-information line, okay? This information will be present and explained in the meta-information lines, and once you combine this meta-information line with this data line, this will be clear. So, we will talk about these meta-information lines in the subsequent classes, and then you will see that this part will be clear. Then you have something called the format field that you can probably only see certain parts for certain variants, right? Again, these format fields are defined in the meta-information lines. So, this information is present there, and we will use this information to understand what it means.

And finally, you have the sample name. Here, I am showing just one example sample name. You can have multiple of them, and for each of them, you will have this format followed. So, this format, GT, GQ etcetera, that you can see, will actually be repeated for each sample that you have. So, if you have multiple samples, this information will be given for multiple samples, ok? So, then we will have the meta-information lines we will talk about in the next class, and then you will understand what the data lines actually contain.

So, here are the references that we have used for this class. To summarize, we have talked about variant calling using samples and VCF tools in multiple steps. So, we have summarized this in this class. We have talked about the hands-on work that we have done, and we have mentioned the steps and the process commands that we utilized. So, it might sound very complex, but once you do it yourself, you will see that each of these steps actually does something very important. And there are only a few main steps right now that we have seen; for example, you have the local realignment and you have the duplicate marking, but some of the steps that we have done in the hands-on are actually preparing the data for those realignments and the final variant calls, etcetera.

Then we also visualized using the integrated genome browser, or IGP. So, we can verify the results that we obtain through the integrated genome browser. So, this is a good way to see that our pipeline is working fine; there is no problem or issue with the pipeline. Then we talked about the VCF files that are generated by the variant calling, and we talked about the three parts that are present. So, we have meta-information lines, header lines, and data lines, and in this class, we have described the data lines OK and also what the header lines contain. So, the header line simply mentions or describes the fields that are present in the data lines.

We have not talked about the meta-information lines, and in the next class we will discuss the meta-information lines. By combining the meta information lines with the data lines, you will be able to understand the VCF file completely. Thank you, and thank you.