Next Generation Sequencing Technologies: Data Analysis and Applications

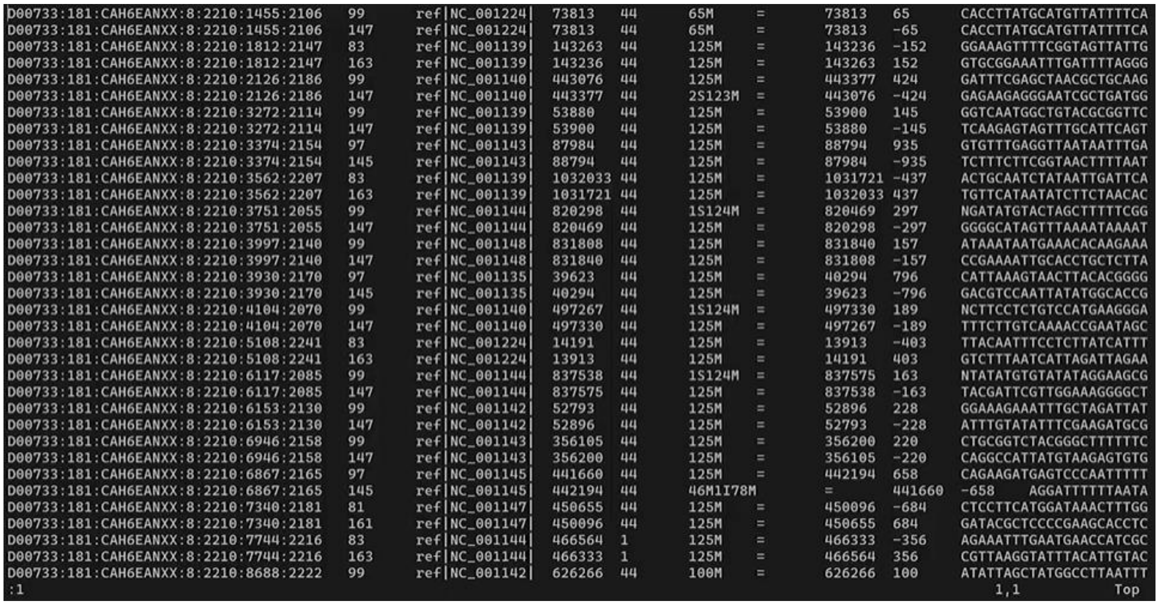
SAM and BAM format

Dr. Riddhiman Dhar, Department of Biotechnology

Indian Institute of Technology Kharagpur

Good day, everyone. Welcome to the course on Next Generation Sequencing Technologies, Data Analysis, and Applications. This is a machine-readable file, and there are two sections: the header section and the alignment section. Now, in the case of Bowtie 2, by default, Bowtie 2 will print the same header @ hd, @ sq, and @ pg. So, this information will be in the header section. Now, sometimes, if you have a lot of information in the header section, it might be difficult to actually look at the alignment section. So, what you can do is we can remove these header sections when we are running Bowtie 2, ok?

So, Bowtie 2 gives us that option as well, and you can do this by running these options. Minus-minus no hat minus-minus no sq when you are running Bowtie 2, ok. And we can do that right before we actually start going into the alignment section because the header section could be a bit distracting. So, let us do that. Let us run this, and then we can see right away how this will change the output file, ok? So, we are going back to the original command, right? We have this; we are supplying this iteration of one-trim data that reads one-trimmed and reads two-trimmed data, right?



```
D00733:181:CAH6EANXX:8:2210:1455:2106   99      ref|NC_001224|  73813   44      65M     =       73813   65      CACCTTATGCATGTTATTTTCA
D00733:181:CAH6EANXX:8:2210:1455:2106   147     ref|NC_001224|  73813   44      65M     =       73813   -65     CACCTTATGCATGTTATTTTCA
D00733:181:CAH6EANXX:8:2210:1812:2147   83      ref|NC_001139|  143263  44      125M    =       143236  -152    GGAAAGTTTTCGGTAGTTATTG
D00733:181:CAH6EANXX:8:2210:1812:2147   163     ref|NC_001139|  143236  44      125M    =       143263  152     GTGCGGAAATTTGATTTTAGGG
D00733:181:CAH6EANXX:8:2210:2126:2186   99      ref|NC_001140|  443076  44      125M    =       443377  424     GATTTCGAGCTAACGCTGCAAG
D00733:181:CAH6EANXX:8:2210:2126:2186   147     ref|NC_001140|  443377  44      2S123M  =       443076  -424    GAGAAGAGGGAATCGCTGATGG
D00733:181:CAH6EANXX:8:2210:3272:2114   99      ref|NC_001139|  53880   44      125M    =       53900   145     GGTCAATGGCTGTACGCGGTTC
D00733:181:CAH6EANXX:8:2210:3272:2114   147     ref|NC_001139|  53900   44      125M    =       53880   -145    TCAAGAGTAGTTTGCATTCAGT
D00733:181:CAH6EANXX:8:2210:3374:2154   97      ref|NC_001143|  87984   44      125M    =       88794   935     GTGTTTGAGGTTAATAATTTGA
D00733:181:CAH6EANXX:8:2210:3374:2154   145     ref|NC_001143|  88794   44      125M    =       87984   -935    TCTTTCTTCGGTAACTTTTAAT
D00733:181:CAH6EANXX:8:2210:3562:2207   83      ref|NC_001139|  1032033 44      125M    =       1031721 -437    ACTGCAATCTATAATTGATTCA
D00733:181:CAH6EANXX:8:2210:3562:2207   163     ref|NC_001139|  1031721 44      125M    =       1032033 437     TGTTCATAATATCTTCTAACAC
D00733:181:CAH6EANXX:8:2210:3751:2055   99      ref|NC_001144|  820298  44      1S124M  =       820469  297     NGATATGTACTAGCTTTTTCGG
D00733:181:CAH6EANXX:8:2210:3751:2055   147     ref|NC_001144|  820469  44      125M    =       820298  -297    GGGGCATAGTTTAAAATAAAAT
D00733:181:CAH6EANXX:8:2210:3997:2140   99      ref|NC_001148|  831808  44      125M    =       831840  157     ATAAATAATGAAACACAAGAAA
D00733:181:CAH6EANXX:8:2210:3997:2140   147     ref|NC_001148|  831840  44      125M    =       831808  -157    CCGAAAATTGCACCTGCTCTTA
D00733:181:CAH6EANXX:8:2210:3930:2170   97      ref|NC_001135|  39623   44      125M    =       40294   796     CATTAAAGTAACTTACACGGGG
D00733:181:CAH6EANXX:8:2210:3930:2170   145     ref|NC_001135|  40294   44      125M    =       39623   -796    GACGTCCAATTATATGGCACCG
D00733:181:CAH6EANXX:8:2210:4104:2070   99      ref|NC_001140|  497267  44      1S124M  =       497330  189     NCTTCCTCTGTCCATGAAGGGA
D00733:181:CAH6EANXX:8:2210:4104:2070   147     ref|NC_001140|  497330  44      125M    =       497267  -189    TTTCTTGTCAAAACCGAATAGC
D00733:181:CAH6EANXX:8:2210:5108:2241   83      ref|NC_001224|  14191   44      125M    =       13913   -403    TTACAATTTCCTCTTATCATTT
D00733:181:CAH6EANXX:8:2210:5108:2241   163     ref|NC_001224|  13913   44      125M    =       14191   403     GTCTTTAATCATTAGATTAGAA
D00733:181:CAH6EANXX:8:2210:6117:2085   99      ref|NC_001144|  837538  44      1S124M  =       837575  163     NTATATGTGTATATAGGAAGCG
D00733:181:CAH6EANXX:8:2210:6117:2085   147     ref|NC_001144|  837575  44      125M    =       837538  -163    TACGATTCGTTGGAAAGGGGCT
D00733:181:CAH6EANXX:8:2210:6153:2130   99      ref|NC_001142|  52793   44      125M    =       52896   228     GGAAAGAAATTTGCTAGATTAT
D00733:181:CAH6EANXX:8:2210:6153:2130   147     ref|NC_001142|  52896   44      125M    =       52793   -228    ATTTGTATATTTCGAAGATGCG
D00733:181:CAH6EANXX:8:2210:6946:2158   99      ref|NC_001143|  356105  44      125M    =       356200  220     CTGCGGTCTACGGGCTTTTTTC
D00733:181:CAH6EANXX:8:2210:6946:2158   147     ref|NC_001143|  356200  44      125M    =       356105  -220    CAGGCCATTATGTAAGAGTGTG
D00733:181:CAH6EANXX:8:2210:6867:2165   97      ref|NC_001145|  441660  44      125M    =       442194  658     CAGAAGATGAGTCCCAATTTTT
D00733:181:CAH6EANXX:8:2210:6867:2165   145     ref|NC_001145|  442194  44      46M1I78M        =       441660  -658    AGGATTTTTTAATA
D00733:181:CAH6EANXX:8:2210:7340:2181   81      ref|NC_001147|  450655  44      125M    =       450096  -684    CTCCTTCATGGATAAACTTTGG
D00733:181:CAH6EANXX:8:2210:7340:2181   161     ref|NC_001147|  450096  44      125M    =       450655  684     GATACGCTCCCCGAAGCACCTC
D00733:181:CAH6EANXX:8:2210:7744:2216   83      ref|NC_001144|  466564  1       125M    =       466333  -356    AGAAATTTGAATGAACCATCGC
D00733:181:CAH6EANXX:8:2210:7744:2216   163     ref|NC_001144|  466333  1       125M    =       466564  356     CGTTAAGGTATTTACATTGTAC
D00733:181:CAH6EANXX:8:2210:8688:2222   99      ref|NC_001142|  626266  44      100M    =       626266  100     ATATTAGCTATGGCCTTAATTT
:1                                                                                                            1,1           Top
```

And then we will give this option. We can have this minus-minus local option, and then we can give these options: minus-minus no hat and minus-minus no sq, ok? So, we do not want this header or cube lines. So, we can simply use these commands to remove those, ok? So, it will run, and we will see the output file. You will see there is no header line, and it looks clean, right? So, in some cases, this might be useful, ok?

If you are doing the alignment yourself right now, you know what this reference genome is, what the length of this reference sequence is, which program you are using, etcetera. These are known to you. So, you are not getting any new information, right? So, maybe having this header line does not make sense in your case, ok? So, let us look at this output file now once we have removed this header, and you can see we are in the alignment section right away, ok? We do not have any headers, right?

So, you can go to the first line on this file using this colon-one command, vi, and you can see there is no header, ok? We are looking at the alignments, ok? And we will now try to interpret this data, ok? You see a lot of information in this file, and we will kind of interpret what this information means, ok? This is very important if you want to write the codes yourself to process these files.

So, we will go back to the presentation, and I will introduce these terms, and then we can come back to this file and have a look, ok? What kind of information do we have in the actual results, and how do you interpret that result? So, the alignment section is right. So, in the alignment section, you have these multiple lines that you have just seen, ok? And each line describes an alignment for a read, or it will tell whether there is a failure, ok?

So, if a read did not align, it will also say that in that alignment file or alignment section, ok. One thing we should remember is that the alignment of all map reads is represented on the forward genomic strand, right? So, this is by convention, and if you have read that map to reverse strand, then in this alignment section, they will be provided as reverse complemented, ok? So, when you are interpreting the data, you should keep that in mind, ok? So, as I said, each line contains the alignment of one read, and it is a collection of 11 or more fields.

And these are all separated by tabs, ok? So, this information is important because if you are trying to process this file yourself, you will use this tab separation to get those fields, ok? Now why did I say that we have 11 or more? Right, why not a fixed number? Because in SAM format, you need to have mandatory fields, right? So, these will always be there, and these are the first 11 fields, ok? So, you will have at least 11 fields.

These mandatory fields will always be there. And on top of that, you have something called optional fields, right? So, these may or may not be there in the file, and in some cases, you will probably see them. So, in our data, you will see these optional fields. These are there because they are also giving us some important information about the mapping, ok?

So, what are these 11 fields? So, I will go one by one and we will also interpret what these fields actually convey, ok? So, some of them are really straightforward. So, you probably understand very easily. Some of them are not, right? So, I will elaborate on those fields a bit more, and you will probably understand what those actually mean, ok?

So, let us move into the mandatory fields first, right? So, then we can discuss the optional fields later, ok? So, field 1 in the compulsory or mandatory fields is something called

QNAME, right? So, this is how we denote this by the capital Q name, right? So, all in capital, right?

And this actually mentions the name of the aligned field, right? So, this name actually comes from the FASTQ file, right? So, you have seen that we have some headers, and we have the name of the read, ok? And that is copied in here, ok? So, we can actually now tell you, right, between the FASTQ file and see which read actually mapped where, ok?

So, this information is there, and this information is also important, right? If you are dealing with paired-end data, ok. So, reads with identical QNAMEs are assumed to come from the same template or fragment, right? So, for paired-end data, they are coming from the same fragment, right? We are sequencing from both directions. So, the QNAME would be identical, as we have seen, right? When you are dealing with this, read 1, read 2 files, right?

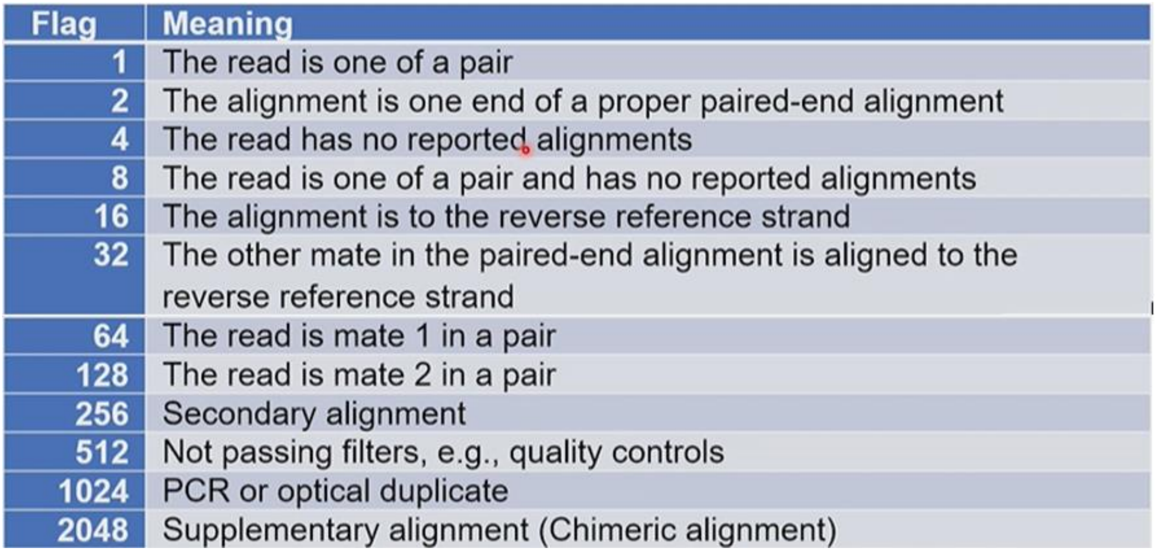
There are two different files: read 1 dot FASTQ and read 2 dot FASTQ. If you look at the header names, the names are identical except for a second field, right, where the read number is given: right, 1 or 2, right. So, that is not part of QNAME, ok? So, that means that the second part of the header in the FASTQ file is not part of the name, ok? As you will see in a moment, the QNAME consists of the first part of the header in the FASTQ file.

So, it simply tells us about the name of the aligned read. Then you have the second field, which is called FLAG. So, this is a combination of FLAGs, which is like a sum. We usually take the sum of the FLAGs that are present. You might ask what these FLAGs are, ok?

So, they are actually describing certain characteristics of the book that we are dealing with, ok? So, it is not just the characteristics of the read, but in addition, it also describes the characteristics of the meet in the case that we are dealing with paired-end data. So, let's look at these numbers. So, they are usually denoted by numbers, and we take the sum. The final FLAG that we report in the SAM file or in the program reports is the sum of these numbers, ok?

So, what we will simply do is look at these numbers, and you can identify certain cases where these numbers would be applicable for your read and you will simply take those numbers and sum them up. We will also take some examples for better understanding, ok? So, the first FLAG, right, which is denoted by this number 1, is read as 1 of a pair, ok? So, we are kind of trying to interpret this in the context of Illumina data. Of course, there is a slightly different interpretation or a slightly more general interpretation of these FLAGs.

So, for example, you will see that 1 does not always mean paired-end data. It means you have templates matching multiple regions of the reference, ok? So, we will kind of interpret this in terms of Illumina data because we are kind of going through examples of Illumina data, ok? So, FLAG 1 will be applied when a read is one of a pair, right? So, now you can imagine that if you are dealing with single-end sequencing data, you will not apply this FLAG to your read.



Flag	Meaning
1	The read is one of a pair
2	The alignment is one end of a proper paired-end alignment
4	The read has no reported alignments
8	The read is one of a pair and has no reported alignments
16	The alignment is to the reverse reference strand
32	The other mate in the paired-end alignment is aligned to the reverse reference strand
64	The read is mate 1 in a pair
128	The read is mate 2 in a pair
256	Secondary alignment
512	Not passing filters, e.g., quality controls
1024	PCR or optical duplicate
2048	Supplementary alignment (Chimeric alignment)

Then you have FLAG value 2, which is the alignment of one end of a proper paired-end alignment, again applying to paired-end data, right? So, this will not apply if you are dealing with single-end data. Then you have FLAG 4, not 3, ok, FLAG 4, and it is applied when the read has no reported alignments, ok. This is something that will apply to single-end as well as paired-end data.

Then you have FLAG 8, right? So, not 5, 6, or 7 in between those numbers are there. You have FLAG 8, and the read is 1 of a pair, and it has no reported alignments, ok? So, FLAG 4 and FLAG 8 are slightly different, but as you can see, FLAG 4 will apply to both single-end and paired-end data, whereas FLAG 8 will apply only to paired-end data. Now FLAG 16 is the alignment to the reverse of the reference sequence, right? So, reverse the reference strand, ok?

And then you have 32, which actually describes the alignment of the mate. So, 16 is the alignment of the read that you are working with, or the read for which you have the QNAME. This value of the FLAG is for the mate, ok? So, the mate or the pair in the paired-end element is aligned to the reference strand, ok? So, it kind of gives you the orientation, right? Which one is occurring upstream of the other one?

So, this information is kind of given here. Then we have FLAG 64, which means the read is mate 1 in a pair. So, it is giving us information about these mates, right? So, mate 1, mate 2. If we have FLAG 128, this means the read is mate 2 in a pair, right?

So, we are dealing with paired-end data. We will get these FLAGs. Then we have 256. So, it will tell us whether this is a primary alignment or a secondary alignment. So, if you see this FLAG, this means this is a secondary alignment, and there must be some primary alignment that has been described.

Then you have 512. There is some sort of quality filtering, or this read is not passing this quality control check. Then you have 1024, which means this is a duplicate PCR or optical duplicate. We have described what a PCR duplicate is and what an optical duplicate is. Then you have 2048. It means there is a supplementary alignment or chimeric alignment.

Now, these chimeric alignments happen primarily due to structural variation in gene fusions. So, if you can imagine that, let us say two chromosomes are fused, right? So, one part of the read will map to, let us say, chromosome 1. The other part of the read will map to chromosome 10, right? So, if those kinds of situations happen, then you will get this FLAG 2048.

Now one thing you would notice is that these FLAGs, right, one in the earlier slide as well as here. So, they are actually kind of separated, right? So, you have these 2 to the power 0, 2 to the power 1, 2 to the power 2, and So, on, right? This is done to actually generate unique sums, right? So, as I have mentioned, this FLAG that we get in field 2 is actually the sum of applicable FLAGs.

So, the sum should be unique So, that you can figure out which FLAGs were applied to a read. These numbers are right, and they are not in sequences. So, why is that, right? Because when you take the sum of each of these unique combinations, you would get a unique sum value, okay? So, that is why these are followed in this way, okay? And let us take some examples So, that we understand how we calculate these FLAG values, okay?

So, the first one is, let's say I give you a scenario and then you have to identify what the FLAG value is. So, example 1 is an unpaired read that aligns with the reverse reference strand, okay? So, I have mentioned this unpaired read, which aligns with the reverse reference strand, okay? So, you do not have to, of course, remember the FLAG values. Just looking at the table, you have to kind of now deduce which kind of FLAG you would apply to this read, okay?

So, this is unpaired, right? So, you will not apply any of these paired-end data FLAGs, right? So, the only thing I can see is, right, that is, right, So, it aligns with the reverse reference strand. The only thing you can see is 16, okay? So, that is actually what you would apply for. This will apply to single-end data as well, and the FLAG is 16, okay?

So, you can try this yourself. You can go back and see the table, right? I am not seeing this in front of me. So, slightly difficult to go back and forth, but you will see that once you look at the table, you can see which kind of tag is applicable to your data, then you can apply that and just take the sums of all the FLAGs that are applicable, okay? So, let us take a second example where you can apply multiple FLAGs, and then you will be clear, right, what is the final FLAG, okay?

So, this is a more interesting situation, right? We have a paired-end read that aligns with the reverse reference strand and is the first made in the pair, okay? So, there is a lot of information. It is paired-end data, which means there would be a lot of FLAGs that could be applicable, right? It also gives the information that this read aligns with the reverse reference strand, okay? So, if you go back to the table, I am not going to do that, but you can check the table and apply the appropriate FLAGs, and you will see the FLAG will be 83, which will be 64 plus 16 plus 2 plus 1, okay?

So, all these FLAGs are 64, 16, 2, and 1. These FLAGs are applicable in this situation for this read, and you just take the sum of these and get this FLAG value of 83, okay? So, you can now actually do that yourself, right? And you can interpret the data, and the sum will be unique depending on the combinations that you use, okay? So, what I will do is look at the same file, and we will look at these first few fields that we have just described, okay, and especially the FLAG field. We will see what kind of value we get from the actual data, okay? So, let us go to the terminal, right, and let us look at the output file, okay? We open that output file using this VI, right, and we use colon se nowrap, right?

We removed the wrapping because we can kind of see this very nicely; the data is organized very nicely. In the first column, you have the queue name, right? This is the read name, right, and another thing you probably would notice here is that these first two, right, have the same queue name. That is because they are coming from the paired-end data, right? So, they read one and read two of the same fragments, and So, you can see this data is organized in pairs, right?

So, you have these first two, then the next two, and So, on. Okay, because we are dealing with paired-end data, this is what is expected. The second field, as I have mentioned, is the FLAG, right? Here you have this 99, 147, and you see this 83, right? We just described how you get this FLAG 83.

So, you see this FLAG 83, then you have 163, etc. So, again, depending on the situation, you can calculate this FLAG value, which would be the sum of all the applicable FLAGs for the read that you are working with, okay? So, this kind of gives you a feel for the real FLAG values that you have, okay? You can go down and you will see you have only a few combinations around, right, and that is generated by the method that we just described, okay? So, we can now go back; we will keep this open, but we can go back now to the presentation and look at the other fields after this FLAG, okay?

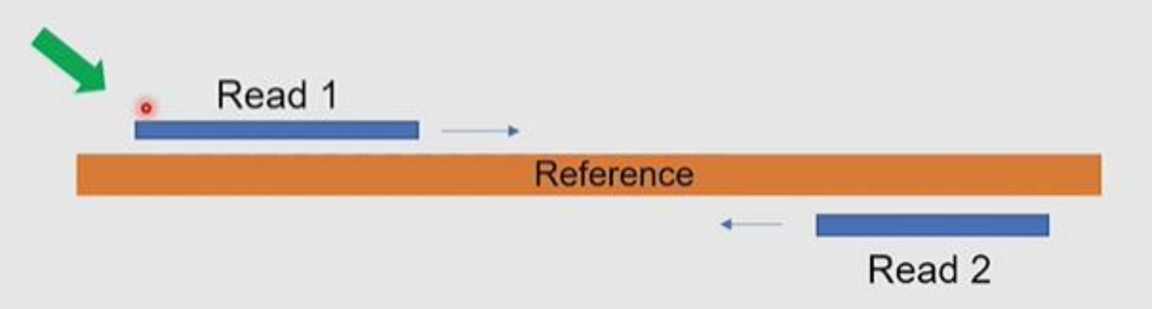
So, we have described fields 1 and 2. Now we can go back to field 3, right? So, field 3 actually describes something called Rname, okay? So, this is the name of the reference sequence where alignment occurs, okay? So, reference sequence means, depending on the data, whether this is actually stored as chromosomal data, right?

So, it will give you the chromosome name, right? So, in the first format file that we used in the mapping process, we have the chromosome name, chromosome number, or chromosome ID—some sort of ID. So, here in the R name, it will just give that chromosome ID, and then you will also have a field that will give you the position, right? So, what you will see is that if you have this @ Sq header line, we have removed that now, but as you have seen, this R name will be one of the values, okay? It is there, So, this is the sequence name. So, @ Sq Sn, it will be mentioned, and this R name value will be mentioned.

So, if it is, let us say, chromosome 1, you will have a header line saying @ of Sq Sn chromosome 1, right? You have seen this example in that we have just shown you, right, with the header, okay. So, in addition, you will also have this @ of Sq ln, right? It was there, and that was giving the chromosome size, right? So, that will give you the chromosome size, length of the chromosome, etc.

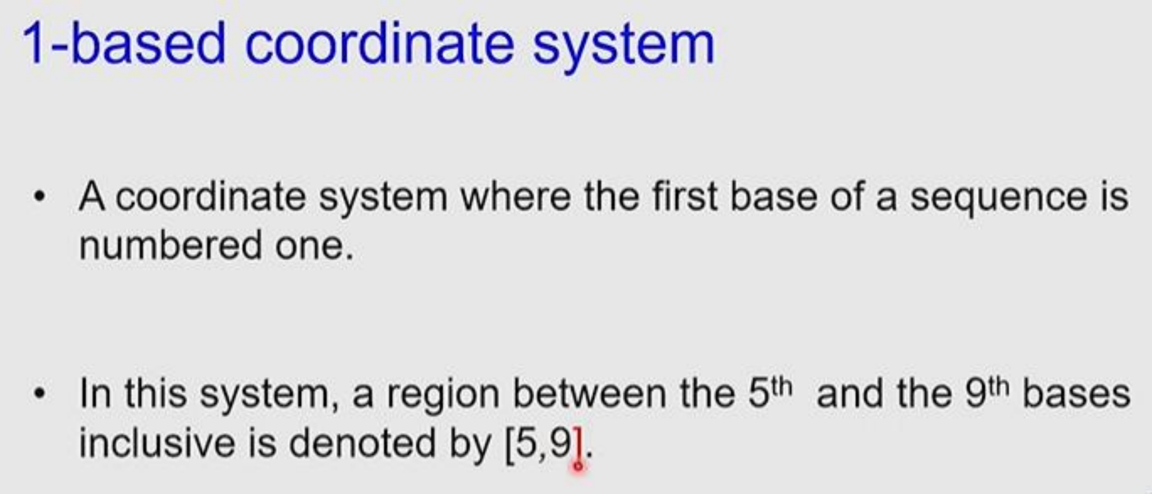
, okay. So, for unmapped reads, you will not see an R name, right? So, because there is no mapping, the Rname will have a value of a star, okay? So, if you see this term, it means this read did not align against the reference sequence, okay? So, after this reference, what is expected is the position, right? So, you want to know the position in the reference sequence, and that is what field 4 gives you, okay? So, this is something called POS, or position, and this is a one-based positioning system in the forward reference strand.

So, as I mentioned when you started with the alignment section, I just mentioned all alignments are given with respect to the forward reference strand, right? And this is a point-based positioning system. So, I will explain in the next slide what this one-based position system is, okay? And it gives the location where the leftmost character of the alignment occurs, So, in the reference sequence, the leftmost character of the read occurs in the reference sequence.



That is the position that we get in this POS, okay? And this is set to 0 if you have unmapped reads, okay? So, if your read did not align, you will get this to 0, okay? So, just to clarify, right, this is what you will get. This is the position. So, if you have this read 1, read 2, if you are describing the results for read 1, it will give you this position, right, So, the leftmost position here, okay?

All right, So, here is the position, right? This position will be given in the POS field, okay? So, this is the fourth tree. Now you might ask, what is this one-based positioning system or one-based coordinate system? So, this is a coordinate system where the first base of a sequence is numbered 1, right? So, you might be kind of wondering why we have to explain why one-based and so, on because there is something called a zero-based coordinate system that we use in many cases, okay?

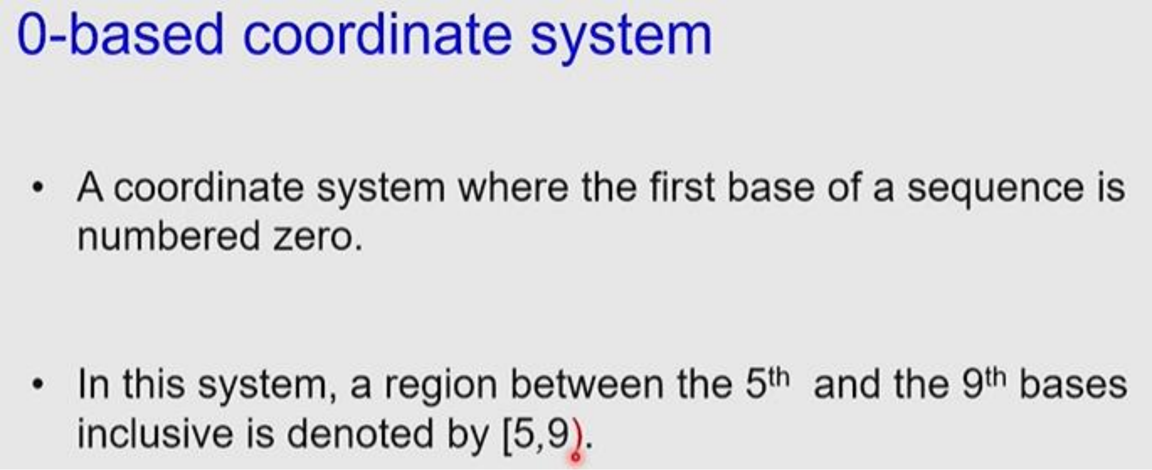


1-based coordinate system

- A coordinate system where the first base of a sequence is numbered one.
- In this system, a region between the 5th and the 9th bases inclusive is denoted by [5,9].

So, this is something that we have to explicitly clarify because otherwise, you might have problems interpreting the exact location of this tree. So, you might actually make some mistakes there. So, this is a coordinate system where the first base will be numbered 1, okay? And this is denoted by these closed brackets, okay? So, for example, if you have a region between 5 and 9, this will be denoted by these 5, 9 closed brackets.

So, notice this one here because you will see that in the case of the zero-coordinate system, this will not be closed; this will be a different sign, okay? And the same file formats: VCF and GFF; So, these are VCF and GFF; these are other formats. So, for example, we will come to VCF format later on and GFF format when you talk about gene ontology. So, they use this one-based coordinate system, okay? So, it is good to know that these files use a one-based coordinate system because, when you are using these files, you need to keep that in mind, okay?



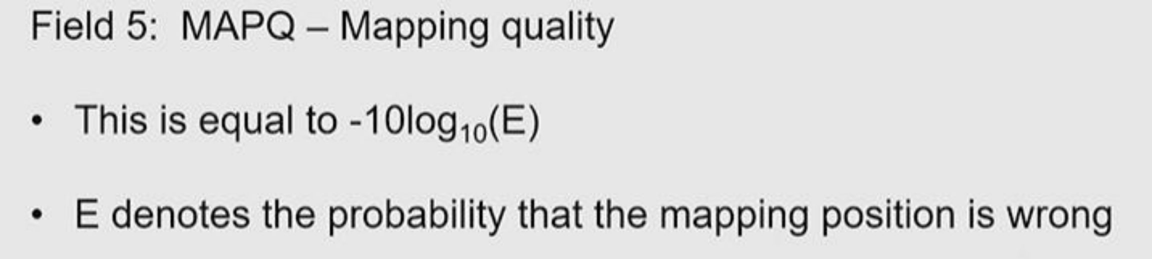
0-based coordinate system

- A coordinate system where the first base of a sequence is numbered zero.
- In this system, a region between the 5th and the 9th bases inclusive is denoted by [5,9).

As I mentioned, there is a zero-based coordinate system where the first base of a sequence is numbered 0, okay? Instead of starting from 1, you start from 0, and if you are familiar with programming, we see that in some cases we actually start counting with 0, not with 1, okay? So, this is where this system uses the same kind of idea you have with a zero-based system, okay? And if you have this right in your system, you have this between the same example where between 5th and 9th base inclusive in a zero-based system, this will be denoted by 5, 9 with this open bracket, okay, not the close bracket or the third bracket, okay. Then BAM and BED formats, as well as other ones, use this zero-based coordinate system.

Again, this is important to know if you are working with these files and you are writing codes to extract data from them. So, in field 5, right now we have this query name, and we have the FLAG, which kind of describes some of the properties of the alignment, right? Then we have the reference sequence name, we have the position information, and then in field 5, it gives you something called MAPQ, or mapping quality, okay? So, it is actually comparable to what we have described before for base quality, which is a very comparable idea. So, you get a number in the mapping quality value, and this is equal to minus 10 log 10 E, okay?

So, it is kind of a very similar idea, like when we calculate the probability of error for a base, given the quality score it was $10^{-(Q/10)}$. It is a very similar idea here. So, here E denotes the probability of error in the mapping process; right instead of base call, this is a probability of error in the mapping process. So, this denotes the probability that the mapping position is wrong, okay? So, this quality score given the quality score, right we can calculate what would be the probability of error, right?



Field 5: MAPQ – Mapping quality

- This is equal to $-10\log_{10}(E)$
- E denotes the probability that the mapping position is wrong

So, higher quality, you know, right? Okay, we can be very confident about the alignment, and it is probably not the wrong mapping. The value is set to 255 if the mapping quality score is not available. Right, the program does not calculate this mapping quality score, okay? So, what I do actually calculates this mapping quality score map Q value, and we will see that, okay? So, before we go into field 6, what I wanted to show you is the data. Right, for actual data, how about these fields that we have just described? Okay, and then we can come back to field 6, which actually stands for CIGAR. So, this is a CIGAR string representation, and again, it is quite complex information, I will describe what this actually means, okay?

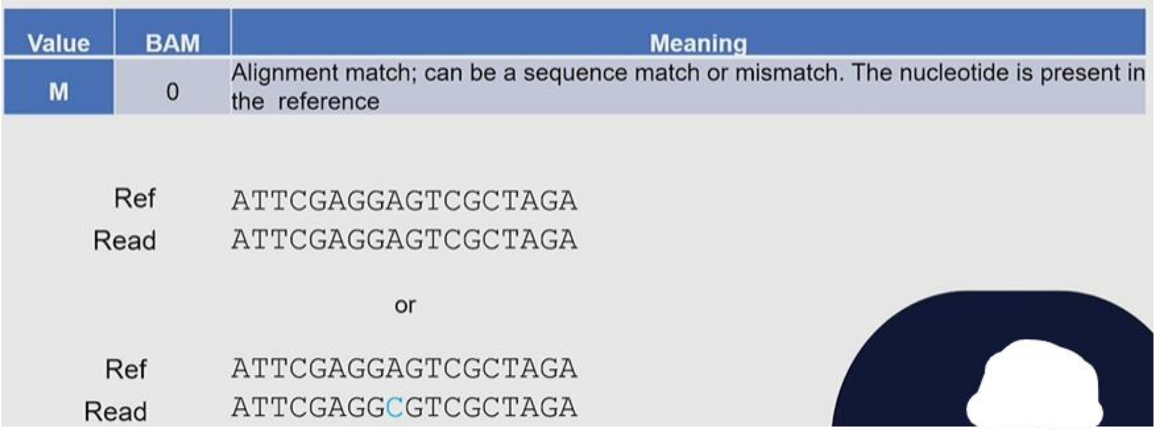
So, let us go back to the terminal and see these fields that we have just described. So, in the first two fields we have, I have already mentioned that you have quality Q, name query name, and then the FLAG. The third one is the name field, right? So, here is this reference number: This is actually pointing to a chromosome ID, okay? So, this is the ID that is given in the reference sequence, and that chromosome ID is given here, okay?

So, we can actually interpret, right? We can actually map it back to the chromosome, whether it is chromosome 1 or chromosome 6 in that organism. We can find that out using some mapping, okay? So, this is the R name, and then you have the fourth field, okay? So, the fourth field gives you the position on that chromosome, all right?

So, here you see, right in the fourth field. So, again, for pairs, you can find these locations. So, here is this: 831, 838. So, in this chromosome in this position, the leftmost character aligns for this field, okay and you can see these numbers vary, right for reads and their mapping to different chromosomes, but the pair usually would align to the same chromosome, right? This is kind of expected because most of these alignments will be concordant mapping. Okay, all right.

So, these first four fields are done. The next one is the map Q. This is the mapping quality score which is 44, right? So, this is quite reliable, right? So, as you have seen, we can calculate the probability of error in mapping by, say, calculating this minus 10 log 10 E. So, if you can calculate, you will see that a higher value is better, right?

So, here this is good, but for some alignment, you see this value is 1, right? So, maybe these alignments are not that great, and perhaps you may have to quantify them, or maybe you have to filter them out when you are processing them, okay? So, that is for later. If you are processing this file, you may have to be aware of this quality score, okay? So, the next field I will just mention here is the CIGAR string, okay? We will now try to interpret what this actually means. This is a combination of numbers and letters, and we will see how we actually interpret these results, okay?



Value	BAM	Meaning
M	0	Alignment match; can be a sequence match or mismatch. The nucleotide is present in the reference

```
Ref     ATTCGAGGAGTCGCTAGA
Read    ATTCGAGGAGTCGCTAGA

                or

Ref     ATTCGAGGAGTCGCTAGA
Read    ATTCGAGGCGTCGCTAGA
```

So, here is the thing: let us go back to the presentation and let us see what this CIGAR string actually means, okay? So, the signature string consists of some numbers and some letters. So, in the SAM file, this will be denoted by letters.

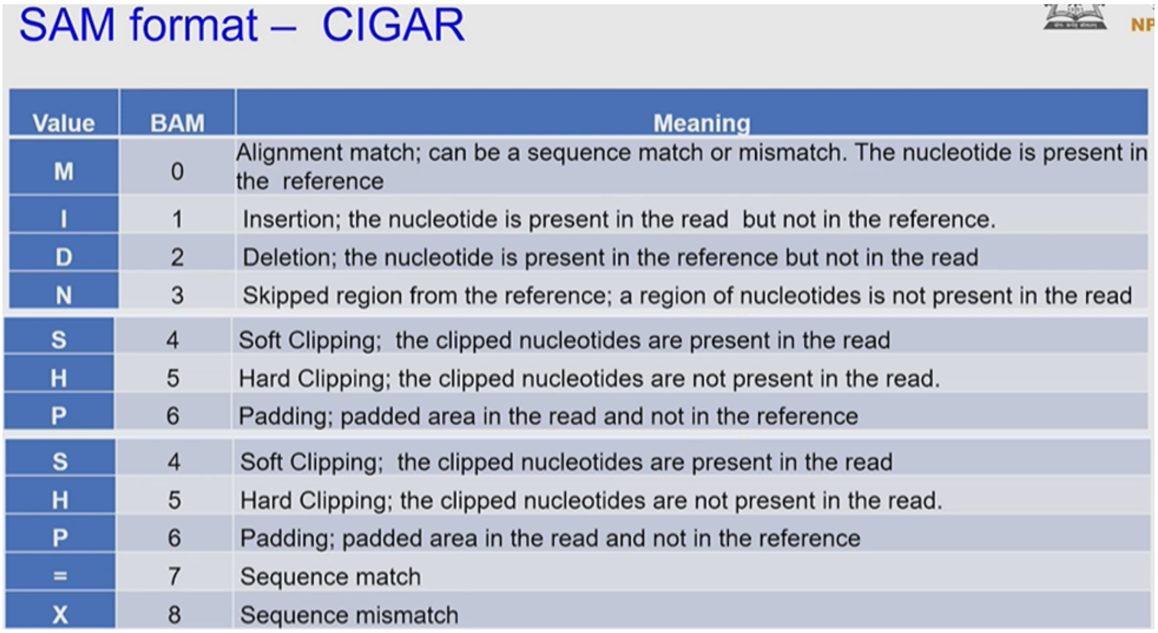
In the BAM file, these are represented in numbers. So, M stands for alignment match, okay? So, do not confuse these with matches, sequence matches, or mismatches. This means that a nucleotide is present in the reference sequence. So, for a read, you have a reference sequence space, okay? So, this means there is a match or mismatch, and there is no indel, okay?

There is no insertion or deletion, okay? So, here is the situation, right? You can see this will be described by match and mismatch, okay? So, you do not have to; you cannot interpret, right, whether there is a base mismatch or a base match. It simply tells, okay, that there is a base corresponding to the read in the reference sequence and there is no insertion or deletion. Then you have the sign I in SAM format. This is 1 in BAM, and this is an insertion, which means the nucleotide is present in the read but not in the reference sequence. This is a situation, right?

So, this is an insertion in the read. Then you have D which is the deletion. The nucleotide is present in the reference but not in the read, right? So, this has been deleted in the read. So, that is why you call deletion and this is an example, right?

So, here you see this G present in the reference but not present in the read. So, this would be a deletion. Then you have something called N, okay, which is the skip region from the reference, okay. So, this is a region that is not present in the read, okay? In the case of RNA sequencing data, you will see this situation where we can interpret this N means there is an intron, right? So, you can have this kind of gapped alignment, right, or split alignment where you have these two parts, right, two exons aligning to slightly different positions, right, slightly apart in the reference sequence, right, and you can have this N or this skip region, okay.

So, in that case, you will have these N values. Then you have this concept of soft clipping or S, right? So, what is this soft clipping, okay? So, this will appear when you are doing local alignment and these bases are not used for alignment, okay, and this will appear mostly at the end of the read most of the time, okay. So, this is present in the read sequence, but also, they are also shown in the SAM file, right? They are part of the read sequence and also shown in the SAM file, okay.



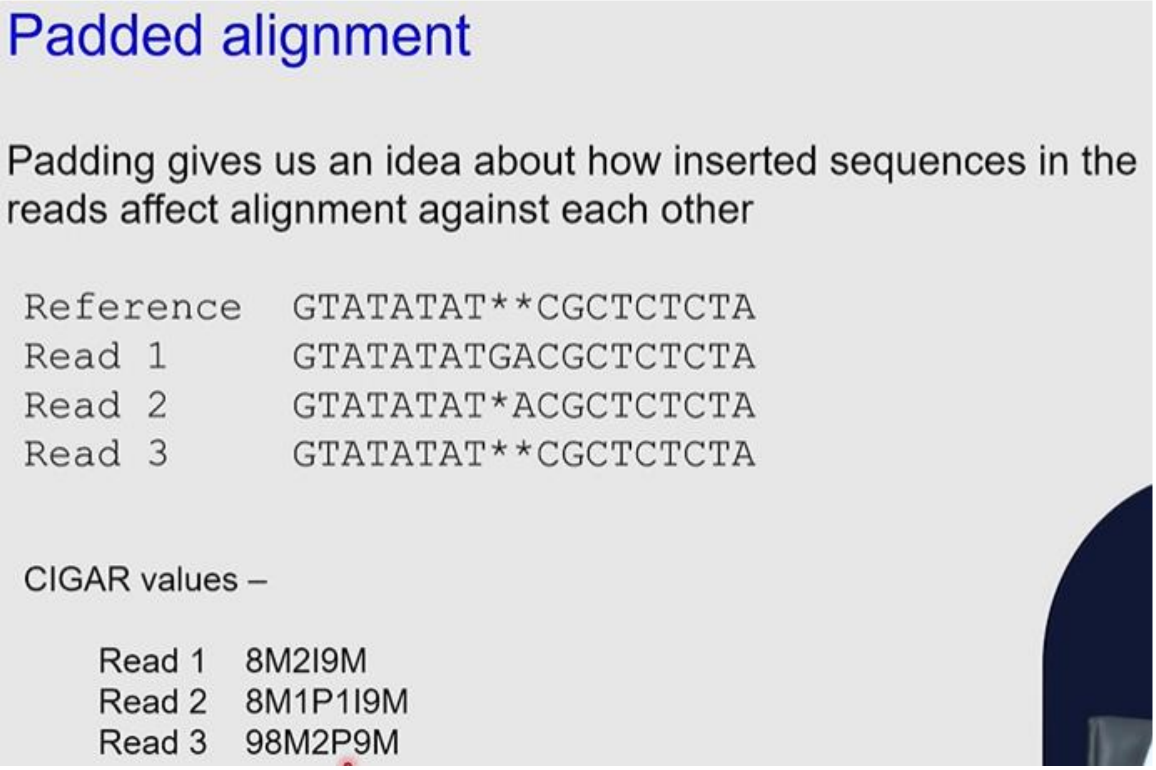
SAM format – CIGAR

Value	BAM	Meaning
M	0	Alignment match; can be a sequence match or mismatch. The nucleotide is present in the reference
I	1	Insertion; the nucleotide is present in the read but not in the reference.
D	2	Deletion; the nucleotide is present in the reference but not in the read
N	3	Skipped region from the reference; a region of nucleotides is not present in the read
S	4	Soft Clipping; the clipped nucleotides are present in the read
H	5	Hard Clipping; the clipped nucleotides are not present in the read.
P	6	Padding; padded area in the read and not in the reference
S	4	Soft Clipping; the clipped nucleotides are present in the read
H	5	Hard Clipping; the clipped nucleotides are not present in the read.
P	6	Padding; padded area in the read and not in the reference
=	7	Sequence match
X	8	Sequence mismatch

Why is, why do we worry about this? Because there is something called hard clipping which is denoted by H, okay. So, again this applies to local alignment and these bases are not used for alignment, but then these sequences are also removed from the read sequence that is shown in the SAM file, okay. There is a final concept which is called the padded alignment, right. So, this gives us an idea about how inserted sequences in the reads affect alignment against each other, okay. So, this will take a bit of explanation, right? What is this padding, or what do I mean by padded alignment.

So, padding gives us an idea about how inserted sequence in the reads affect alignment against each other, okay. So, it is not comparing in the reference sequence because in the reference sequence these bases are not present at all, right. So, let us take this example and then it would be clear. So, imagine this example where you have this reference sequence.

So, star means gaps, right? So, there is no gap, there is no base in there. In read 1 you have this insertion, right, GA. But in read 2 we see only one base A, we do not see G, right? So, we have this star. And in read 3 we do not see any of these inserted bases, So, we just see stars, right? So, if in SIGAR what we will show is 8M, 2Y, 9M and So, this means in the first read we have two insertions corresponding to the reference.



Padded alignment

Padding gives us an idea about how inserted sequences in the reads affect alignment against each other

```
Reference   GTATATAT**CGCTCTCTA
Read 1      GTATATATGACGCTCTCTA
Read 2      GTATATAT*ACGCTCTCTA
Read 3      GTATATAT**CGCTCTCTA
```

CIGAR values –

Read 1	8M2I9M
Read 2	8M1P1I9M
Read 3	98M2P9M

In read 2, read 3, right, there is padding, right. So, the first base here in read 2 you will see is that is not present, right? So, we can use something called 1P. This is a one-base padding, then you have this one-base insertion, right? In the read 3 examples, there is no insertion, So, this will be used as two base padding, right, So, 2P, okay. So, we can actually kind of also look at this bit carefully when you actually look at the data, right, how this actually is interpreted, okay?

So, we have again the CIGAR sign which means sequence match and finally, we have X which is called the sequence mismatch, okay. So, well in the next class, we will actually look at some of the actual CIGAR strings. So, I have just very briefly or very quickly showed you, okay, this is the column that shows the SIGAR string but we will try to interpret in the real data in the next class, okay. So, these are reference that I used for this class and then to summarize we have now learned about the SAM 5 format.

We looked into the header section in the last class, and we have started looking at the alignment section. We have not completed it because there are four more fields that are present, and we will describe those in the next class. And in the alignment section, as I mentioned, there are 11 mandatory fields that will always be present in a SAM 5, and then you can also have some optional fields, okay? And these mandatory fields describe the mapping of a read and also how it is made in the case of paired-end sequencing, and we will actually look at the second part in the next class. Thank you.