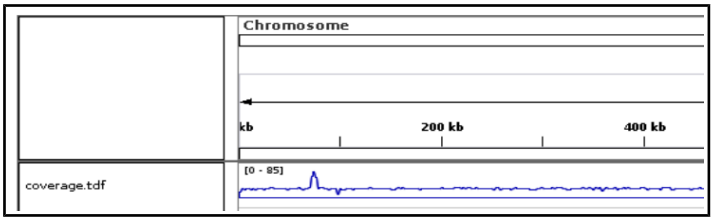
Producing a wig file with the trace of physical coverage

Preliminaries and programs installation

Here we explain how to create and visualize a track of quantitative data on the IGV genome browser. The tracks will be defined as text files, using the "wiggle" specifications (http://genome.ucsc.edu/goldenPath/help/wiggle.html). Then the text file will be transformed in a tdf file, with IGVtoos. The track will be finally visualized with the IGV genome browser, as shown in the figure.



IGV Genome browser

To find and install the IGV genome browser please refer to http://software.broadinstitute.org/software/igv/userguide . Our goal is to visualize some tracks with quantitative data, but before we must inform IGV about the genome that we are dealing with. IGV needs to know at least the name and size of the chromosomes and possibly also their sequence. Several options are available to input the genomic data. Since we are dealing with a relatively small genome, the simplest way is to use the fasta file.

It is essential that the genome contains the “chromosomes” indicated in the wig file. Therefore, if the chromosome of the fasta file are called chromosome_1, chromosome_2, etc, then also the wiggle file must refer to the same names. It is important that the name matches perfectly, in a case-sensitive manner.

In our practical we will use the genome of the bacterium *Lactobacillus casei*, that is constituted by a single chromosome. The name of the chromosome is defined in the fisrt line of the fasta file:

```
>genome 3079196bp
```

Therefore our cromosome is called “genome” and in the wiggle file anything mapping on it should be declared as “chrom=genome” rather than “chrom=chr1” as shown in the examples below, or “chromosome” shown in the figure.

In a fasta file with multiple sequences, the lines starting with “>” will indicate the beginning of a new region of DNA (cheomosomes, contigs, genes, etc.) and the name associated to the region.

Wiggle specifications

There are two modes: "fixedStep" and "variableStep". Even within a single track the two modes can be intermixed as same parts may be better defined in one or the other mode. In any case, all the definitions must be ordered according to their position on the chromosomes. The following example shows the two alternative ways to achieve the same output:

```
...
fixedStep chrom=chr1 start=9990 step=10 span=10
8
9
10
11
13
14
fixedStep chrom=chr1 start=12036 step=10 span=10
34
36
37
13
14
...
```

```
...
variableStep  chrom=chr1 span=10
9990   8
10000  9
10010  10
10020  11
10030  12
10040  13
10050  14
12036  34
12046  36
12056  37
12066  13
12076  14
...
```

The above examples define two blocks: the first starting at position 9990 of chr1, defines values that must be plotted every step bases (10 in the example), each value defining a range of span bases (10 in the example). In the above examples "variableStep" requires more memory, but in other cases it may be better. A nice thing is that you can mix the two formats within the same wiggle file. You should use a .wig extension for wiggle files.

IGV_Tools and tdf format

Although wig files can be directly loaded on IGV, to achieve a faster loading it is better to "binarize" and index the wig file. This is particularly important for large files. It is essential at this point to download and install both the IGV genome browser and the accompanying IGV_Tools.

Once the IGV_Tools have been installed you can open a shell and, assuming that you are in the folder with the binaries of IGV_Tools, type the following:

```
./igvtools toTDF nomefile.wig nomefile.tdf file.chrom.sizes
```

The file file.chrom.sizes defines the chromosome names and sizes. In the package IGV_Tools are defined many different genomes including the human genome (hg19).

Please, refer to https://software.broadinstitute.org/software/igv/igvtools for more details on IGVTools.