Seminar III: R/Bioconductor: Shortread and chipseq

Alejandro Reyes
areyes@lcg.unam.mx
Bachelor in Genomic Sciences
www.lcg.unam.mx/~lcollado/B

Universidad Nacional Autonoma de Mexico

August - December, 2009

ShortRead and chipseq

Exploring data with Shortread package

Aligned shortreads

Chipseq data analysis

Libraries

- Packages we are going to use in this section

```
> library(ShortRead)
> library(chipseq)
> library(GenomicFeatures)
> library(BSgenome.Mmusculus.UCSC.mm9)
```

What is ShortRead?

- It was developed by Martin Morgan
- "The ShortRead package aims to provide key functionality for input, qual ity assurance, and basic manipulation of short read DNA sequences such as those produced by Solexa, 454, Helicos, SOLiD, and related technologies"
- This first part is a lab session made by Cei Abreu

Starting with ShortRead

- Basic functions of ShortRead

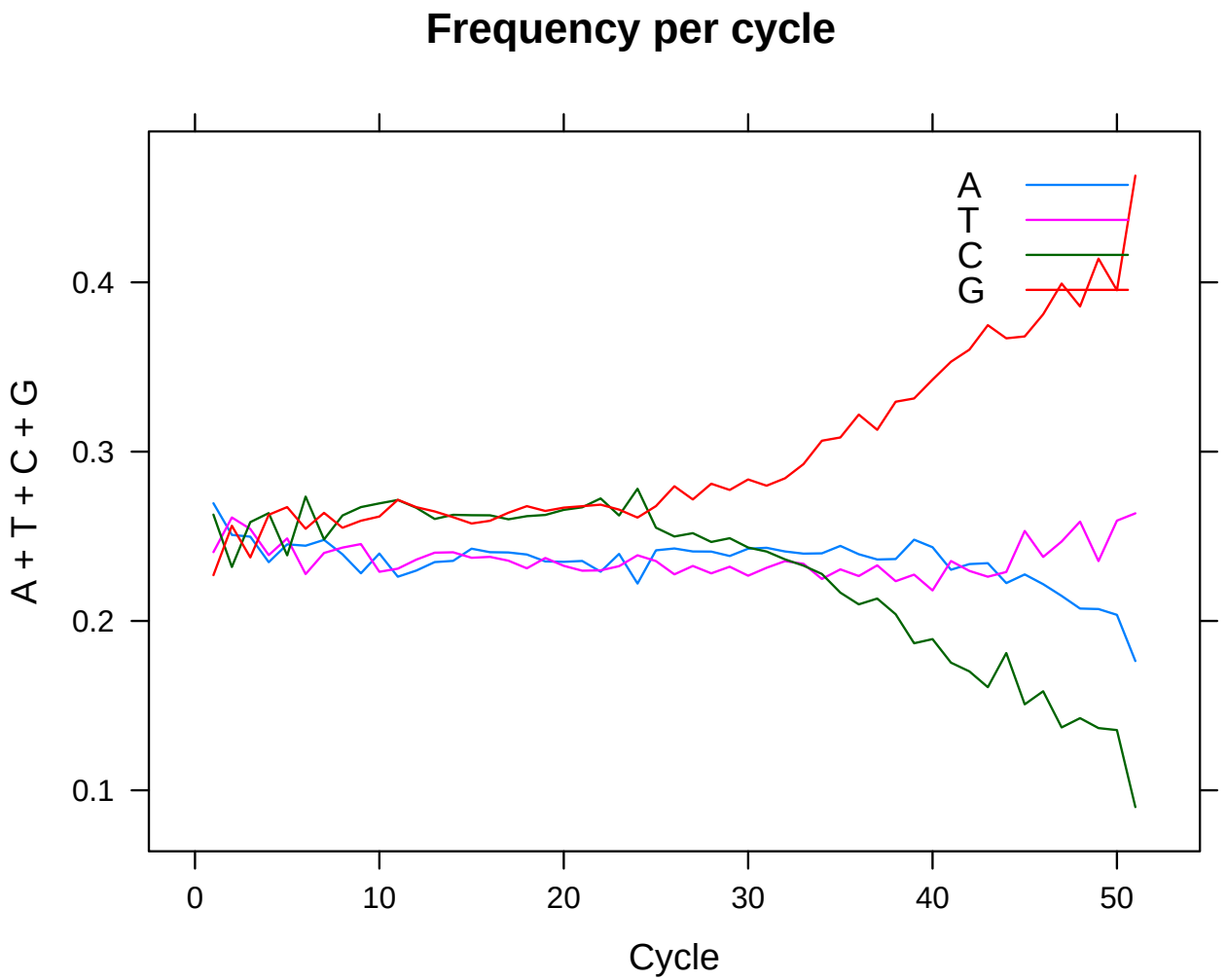
```
> reads <- readFastq(".", pattern = "Typhi_solexa.fastq.aa")
> head(reads, 1)

class: ShortReadQ
length: 1 reads; width: 51 cycles

> head(id(reads), 1)

  A BStringSet instance of length 1
    width seq
[1]    18 IL2_40_5_1_654_768

> head(sread(reads), 1)

  A DNAStringSet instance of length 1
    width seq
[1]    51 AACGCGTTTTGGCG...AAGTAAAAAAGAA
```

Length of the reads

- Why is it important to consider alphabet frequency per cycle in solexa reads?

```
> abc <- alphabetByCycle(sread(reads),
+     alphabet = c("A", "T", "G",
+         "C", "N"))
> abc <- abc/colSums(abc)
> dataabc <- as.data.frame(abc)
```

Alphabet frequency per cycle



Working with qualities

- ShortRead also allows you to work with the qualities given by solexa reads.
- This is a very important thing to consider, and you can filter your reads to have just what you need.

Qualities

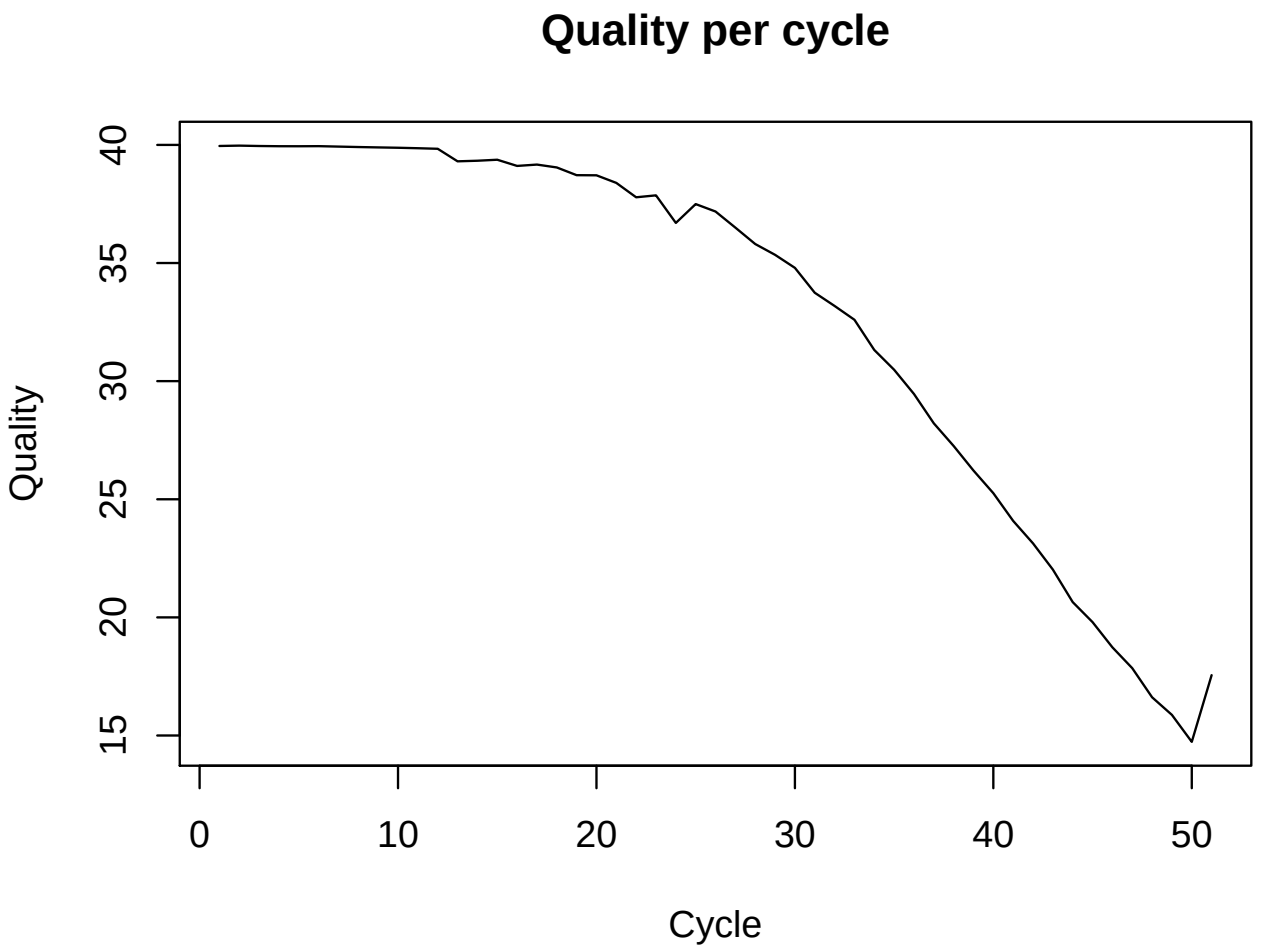
- We can also plot the qualities by cycle in order to cut the sequences when quality falls down.

```
> qualitymatrix <- as(quality(reads),
+     "matrix")
> head(qualitymatrix[, 1:7])
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]   40   40   40   40   40   40   40
[2,]   40   40   40   40   40   40   40
[3,]   40   40   40   40   40   40   40
[4,]   40   40   40   40   40   40   40
[5,]   40   40   40   40   40   40   40
[6,]   40   40   40   40   40   40   40
```

```
> meanquality <- apply(qualitymatrix,
+     2, mean)
```

Quality per cycle



Cutting sequences

- The two plots indicate us that we need to cut the sequences in order to have shorter but with a better quality secuences
- The function narrow allow us to do this easily

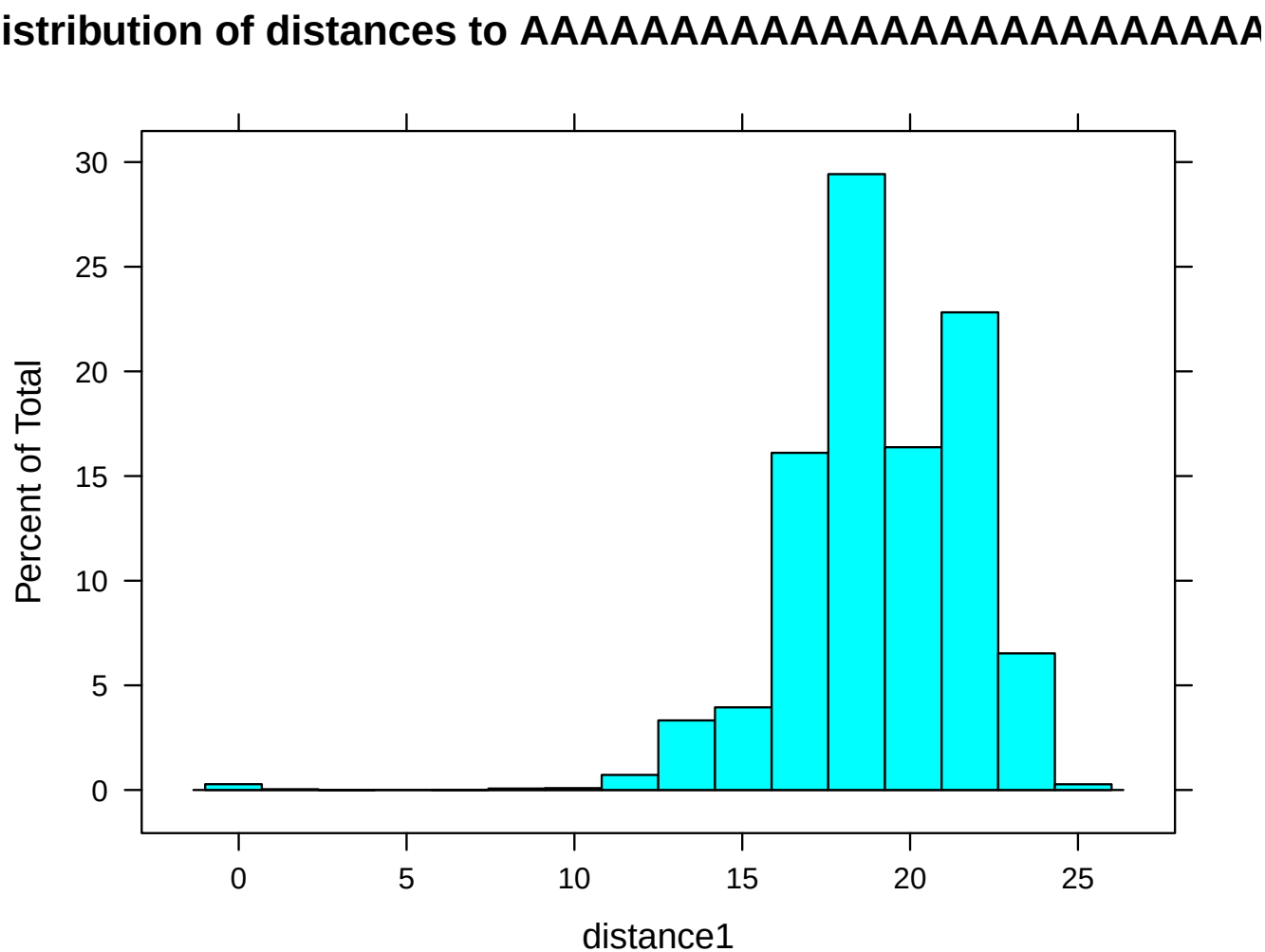
```
> reads

class: ShortReadQ
length: 25000 reads; width: 51 cycles

> shortreads <- narrow(reads, start = 1,
+     end = 25)
> shortreads

class: ShortReadQ
length: 25000 reads; width: 25 cycles
```

More quality

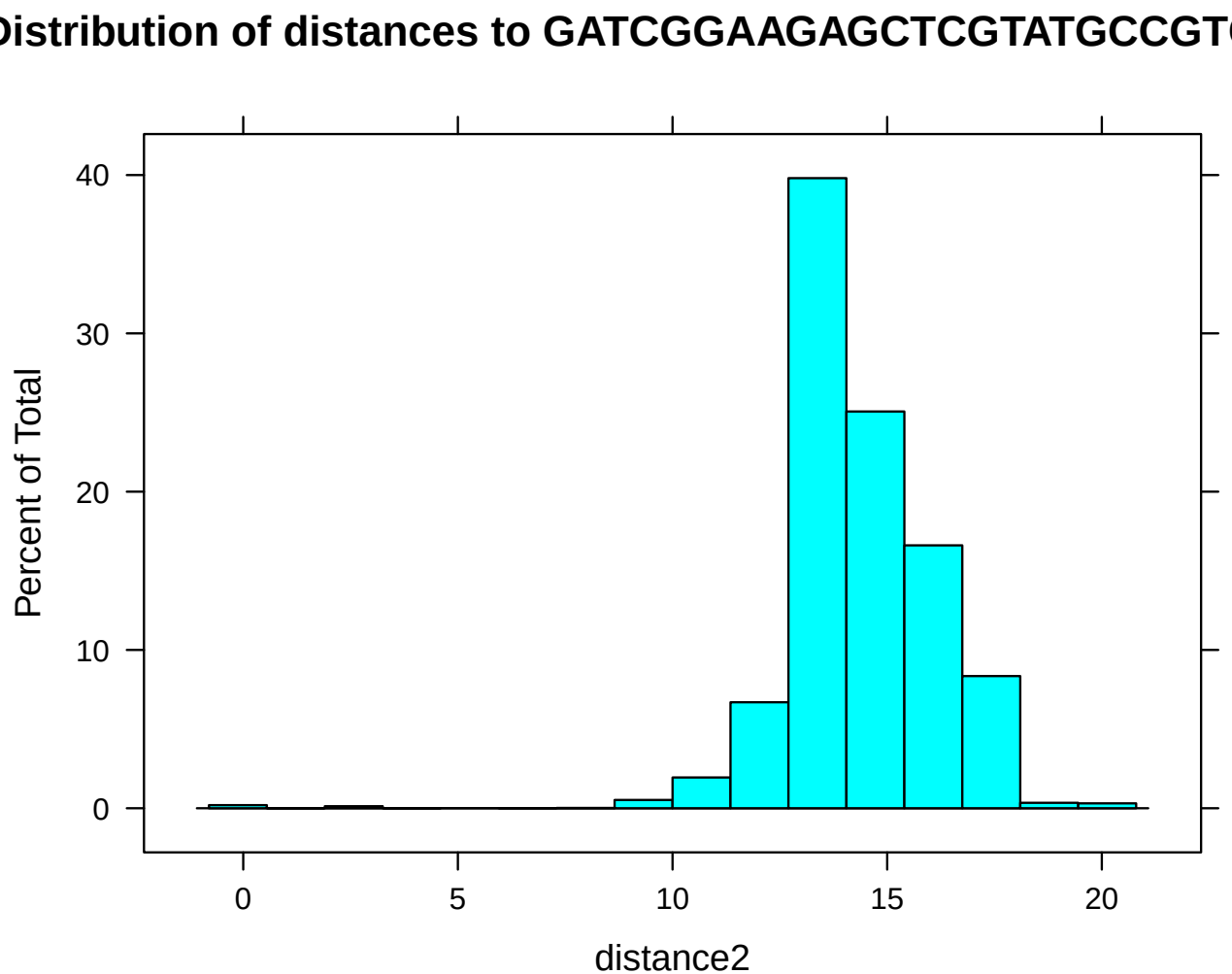
- Ok, now we have just the first 25 cycles!
- In solexa reads we have 2 sequences that are very common... Any idea?
- 1. AAAAAAAAAAAAAAAAAAAAAAAAA
- 2. GATCGGAAGAGCTCGTATGCCGTCT
- The function srdistance calculates de distance between two sequences, therefore it is useful to eliminate this sequences.

```
> distance1 <- srdistance(shortreads,
+     "AAAAAAAAAAAAAAAAAAAAAAAAA")[[1]]
> distance2 <- srdistance(shortreads,
+     "GATCGGAAGAGCTCGTATGCCGTCT")[[1]]
```

Distance


Distribution of distances to AAAAAAAAAAAAAAAAAAAAAAAA
Percent of Total
0
5
10
15
20
25
30
distance1

Distance


Distribution of distances to GATCGGAAGAGCTCGTATGCCGTCT
Percent of Total
0
10
20
30
40
0
5
10
15
20
distance2

Clean sequences

- We have two vectors containing the distance to a respective sequence, how can we remove this sequences from our reads??

```
> length(shortreads)
```

```
[1] 25000
```

```
> cleanreads <- reads[distance1 >
+     5 & distance2 > 5]
> length(cleanreads)
```

```
[1] 24838
```

- Then, we can write our clean sequences into a fastq file!

```
> writeFastq(cleanreads, "cleanthypi.fastq")
```

Aligned shortreads

- ShortRead also contains function to work with aligned reads
- It is focused on aligned reads produced by Solexa Genome Analyzer ELAND Software, but there are also ways to read alignments from MAQ or Bowtie. The name of the funcion is readAligned.

```
> exptPath <- system.file("extdata",
+     package = "ShortRead")
> sp <- SolexaPath(exptPath)
```

- Note that there al NA values in strand and position. This means that those sequences could not be aligned by the software.

Functions

- It is focused on aligned reads produced by Solexa Genome Analyzer ELAND Software, but there are also ways to read alignments from MAQ or Bowtie. The name of the funcion is readAligned.

```
> aln <- readAligned(sp, "s_2_export.txt")
> aln

class: AlignedRead
length: 1000 reads; width: 35 cycles
chromosome: NM NM ... chr5.fa 29:255:255
position: NA NA ... 71805980 NA
strand: NA NA ... + NA
alignQuality: NumericQuality
alignData varLabels: run lane ... filtering contig
```

Functions

- There are some functions to analyze the data such as position, strand

```
> head(position(aln), 3)
[1] NA NA NA
> table(strand(aln))

  -   +   *
203 203   0
```

Qualities, again!

- We can also play with the qualities of both the sequences and of the alignment!
- The qualities are string-coded by Solexa establishment, the letter A corresponds to the log10 of 1.
- The qualities of the alignment are a little bit different, being 0 a failure in the alignment.

```
> head(quality(alignQuality(aln)))
```

```
[1] 0 0 0 0 0 0
```

Filter data

- And with this qualities we can filter our data!!!!
- Lets suppose we want just the sequences that are aligned and filtered by Solexa.

```
> filtered <- alignData(aln)[["filtering"]] ==
+     "Y"
> mapped <- !is.na(position(aln))
> filteredmapped <- aln[filtered &
+     mapped]
> filteredmapped
```

```
class: AlignedRead
length: 364 reads; width: 35 cycles
chromosome: chr17.fa chr18.fa ... chr8.fa chr5.fa
position: 69345321 54982866 ... 19708804 71805980
strand: - + ... - +
alignQuality: NumericQuality
alignData varLabels: run lane ... filtering contig
```

And in case you are a little more biologist than bioinformatician...

- If you want a default analysis or you do not know how to do graphs (hope is not your case)... ShortRead can do this for you!

```
> qual <- qa(sp)
> rpt <- report(qual, dest = ".")
```

Chipseq

- Chipseq is a useful tool for analyzing reads!

```
> data(cstest)
> cstest

GenomeDataList: 2 elements
names(2): ctcf gfp

> str(cstest$ctcf$chr10)

List of 2
 $ -: int [1:72371] 3012999 3013096 3013098 3013135 3032735 3040511 304052
 $ +: int [1:73179] 3012936 3012941 3012944 3012955 3012963 3012969 301297
```

Chipseq

- Chipseq allows you to extend your reads up to what you want.

```
> bc <- basesCovered(cstest$ctcf$chr10,
+     shift = 1:250, seqLen = 24)
> ext <- extendReads(cstest$ctcf$chr10,
+     seqLen = 150)
> head(ext)

IRanges instance:
      start     end width
[1] 3012936 3013085   150
[2] 3012941 3013090   150
[3] 3012944 3013093   150
[4] 3012955 3013104   150
[5] 3012963 3013112   150
[6] 3012969 3013118   150
```

Chipseq

- It also allow us to work with coverages.
- Coverage is the number of times each base of the genome is covered by the extended reads.
- Therefore we can identify islands and peaks.

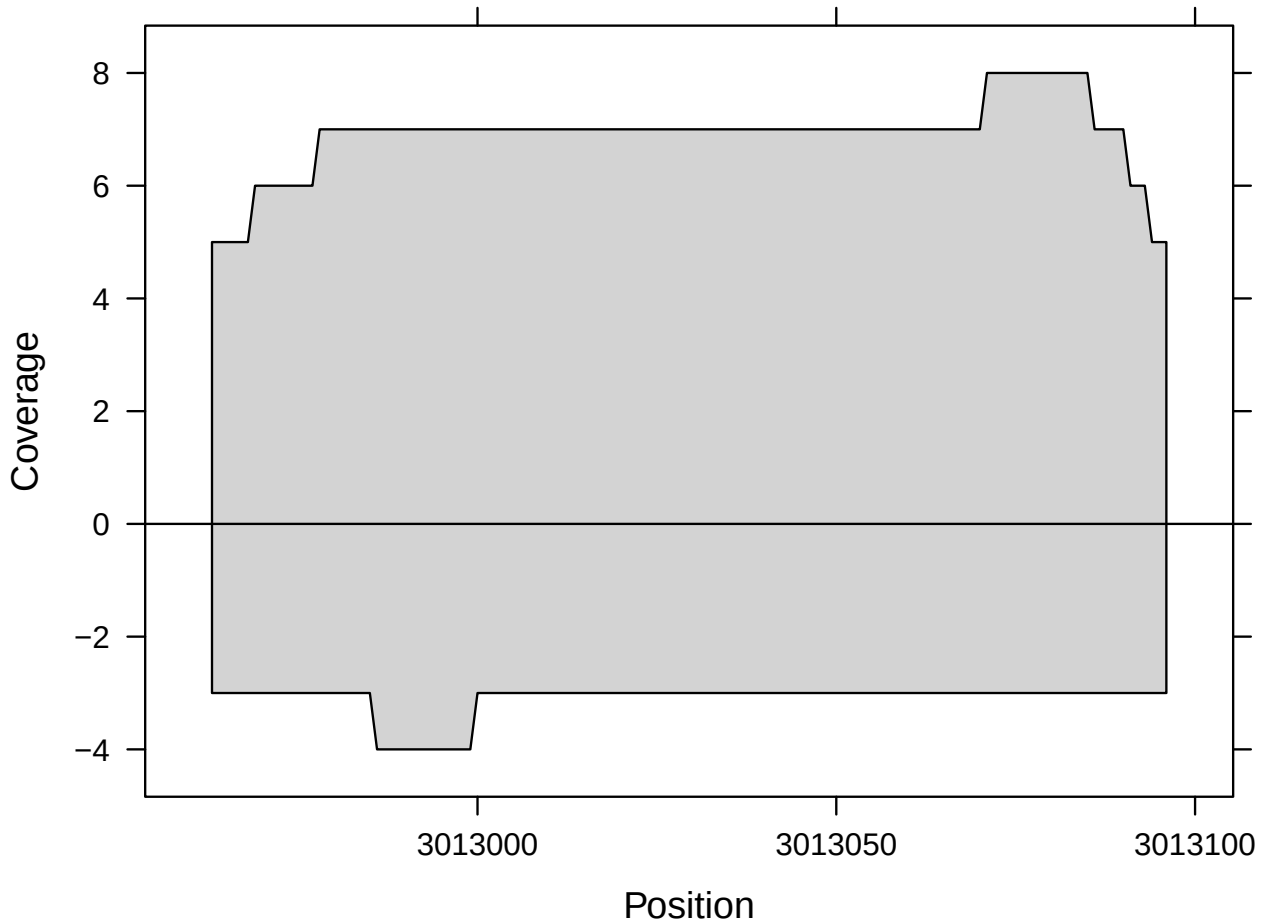
```
> musculuschrlen <- seqlengths(Mmusculus)
> cov <- coverage(ext, width = musculuschrlen["chr10"])
> islands <- slice(cov, lower = 1)
```

Peaks

- If we want to see how a peak is distributed between the strands we can do this...

```
> peaks <- slice(cov, lower = 8)
> cov.pos <- coverage(extendReads(cstest$ctcf$chr10,
+     strand = "+", seqLen = 150),
+     width = musculuschrlen["chr10"])
> cov.neg <- coverage(extendReads(cstest$ctcf$chr10,
+     strand = "-", seqLen = 150),
+     width = musculuschrlen["chr10"])
> peaks.pos <- copyIRanges(peaks,
+     cov.pos)
> peaks.neg <- copyIRanges(peaks,
+     cov.neg)
```

Alphabet frequency per cycle



Chipseq

- Chipseq allows you to work with many lanes!
- The function gdapply helps you with this!

Thanks!

- Class is over

Chipseq

```
> sessionInfo()

R version 2.10.0 Under development (unstable) (2009-09-
i686-pc-linux-gnu

locale:
 [1] LC_CTYPE=en_US.UTF-8
 [2] LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=C
 [6] LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8
 [8] LC_NAME=C
```

Chipseq

```
 [9] LC_ADDRESS=C
[10] LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8
[12] LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices
[4] utils     datasets  methods
[7] base

other attached packages:
 [1] BSgenome.Mmusculus.UCSC.mm9_1.3.11
 [2] GenomicFeatures_0.1.1
 [3] rtracklayer_1.5.13
```

Chipseq

```
 [4] RCurl_1.2-0
 [5] bitops_1.0-4.1
 [6] chipseq_0.1.27
 [7] ShortRead_1.3.36
 [8] lattice_0.17-25
 [9] BSgenome_1.13.14
[10] Biostrings_2.13.46
[11] IRanges_1.3.87

loaded via a namespace (and not attached):
[1] Biobase_2.5.6 DBI_0.2-4
[3] grid_2.10.0   hwriter_1.1
[5] RSQLite_0.7-2 tools_2.10.0
[7] XML_2.6-0
```