Métodos Estadísticos y Analíticos de Datos Genómicos: ShortRead, Biostrings y Genominator

Alejandro Reyes
areyes@lcg.unam.mx
Licenciatura en Ciencias Genómicas

Universidad Nacional Autonoma de Mexico

January 20, 2010

ShortRead and Biostrings

Libraries

- Packages we are going to use in this section

```
> source("http://bioconductor.org/biocLite.R")
> biocLite(c("ShortRead", "Genominator"))

> library(ShortRead)
```

What is Biostrings?

- It provides containers for representing large biological sequences
- Provides utilities for basic computations on sequences (alphabetfrequency, translate, reverseComplement)
- Tools for matching and pairwise alignments

Alignment tools

- matchPDict is fast, find all ocurrences with a given number of mismatches, supports masked regions but does not support indels.
- vmatchPattern is similar to matchPDict, but it supports indels and uses edit distance penalty scheme.
- pairwiseAlignment is not useful for large sequences, returns only the best score, cannot handle masked genomes but includes a quality-based scoring.
-

Little example

- We want to find the "TATAAT" -10 boxes in a region of the Ecoli genome

```
> library(BSgenome.Ecoli.NCBI.20080805)
> Ecoli

E. coli genome
|
| organism: Escherichia coli (E. coli)
| provider: NCBI
| provider version: 2008/08/05
| release date: NA
| release name: NA
|
| sequences (see '?seqnames'):
|   NC_008253  NC_008563  NC_010468
```

Little example

```
|   NC_004431  NC_009801  NC_009800
|   NC_002655  NC_002695  NC_010498
|   NC_007946  NC_010473  NC_000913
|   AC_000091
|
| (use the '$' or '[[' operator to
| access a given sequence)
```

```
> matchPattern("GAAC", Ecoli[["NC_008253"]])

  Views on a 4938920-letter DNAString subject
subject: AGCTTTTCATTCTG...AGTAAGTGATTTTC
views:
         start     end width
    [1]    75      78     4 [GAAC]
    [2]   378     381     4 [GAAC]
```

Little example

```
        [3]     537     540      4 [GAAC]
        [4]     552     555      4 [GAAC]
        [5]    1446    1449      4 [GAAC]
        [6]    1476    1479      4 [GAAC]
        [7]    1515    1518      4 [GAAC]
        [8]    1641    1644      4 [GAAC]
        [9]    1905    1908      4 [GAAC]
        ...     ...     ...    ... ...
[18985] 4936662 4936665      4 [GAAC]
[18986] 4937077 4937080      4 [GAAC]
[18987] 4937126 4937129      4 [GAAC]
[18988] 4937158 4937161      4 [GAAC]
[18989] 4937260 4937263      4 [GAAC]
[18990] 4937338 4937341      4 [GAAC]
[18991] 4938297 4938300      4 [GAAC]
```

Little example

```
[18992] 4938486 4938489      4 [GAAC]
[18993] 4938744 4938747      4 [GAAC]
```

What is ShortRead?

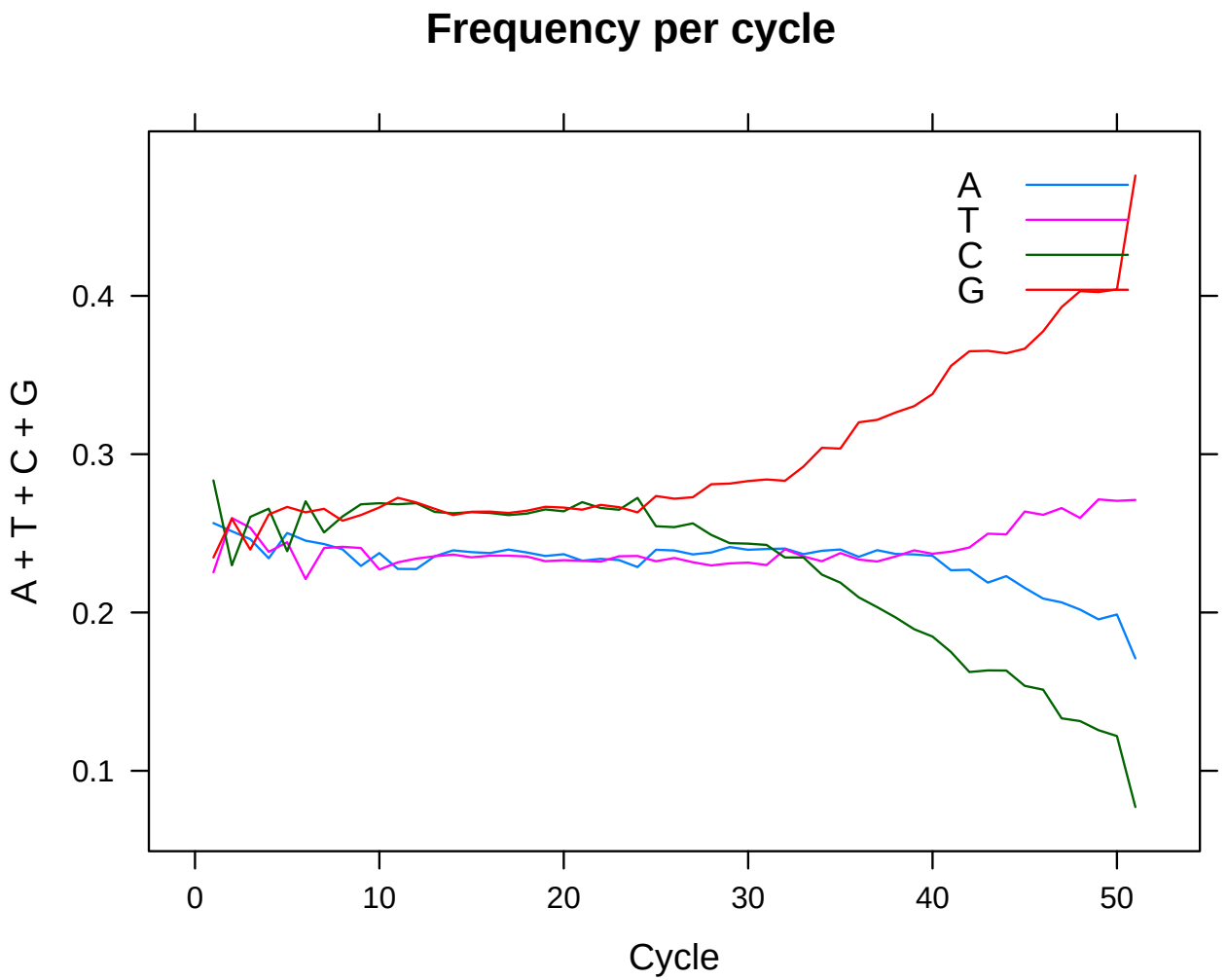
- It was developed by Martin Morgan
- "The ShortRead package aims to provide key functionality for input, qual ity assurance, and basic manipulation of short read DNA sequences such as those produced by Solexa, 454, Helicos, SOLiD, and related technologies"

Length of the reads

- Why is it important to consider alphabet frequency per cycle in solexa reads?
- According to solexa pipeline, in sequencing a genome we should find this frequencies similar to the GC content of the organism

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

Alphabet frequency per cycle



Finding overrepresented sequences

- With very simple and fast code we can find overrepresented sequences!

```
> seq <- tables(reads, n = 15)
> topReads <- data.frame(read = names(seq[["top"]]),
+     count = unname(seq[["top"]]))
> topReads
```

```
                                                 read
1  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
2  GATCGGAAGAGCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAAAAA
3  AGATCGGAAGAGCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAAAA
4  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACA
5  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAA
6  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAAA
7  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAAAAAA
8  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAAAA
9  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAAAAA
10 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAAAAAAA
11 GATCGGAAGAGCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAATAA
12 AGATCGGAAGAGCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAAGA
13 GATCGGAAGAGCTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAAATA
```

Finding overrepresented sequences

```
14 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAAAAAAAA
15 GATCGGAAGACTCGTATGCCGTCTTCTGCTTGAAAAAAAAAAAAAAAAA
   count
1   2891
2    533
3    429
4    109
5     95
6     71
7     63
8     58
9     56
10    40
11    36
12    35
13    34
14    32
15    32
```

Working with the sequencing 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

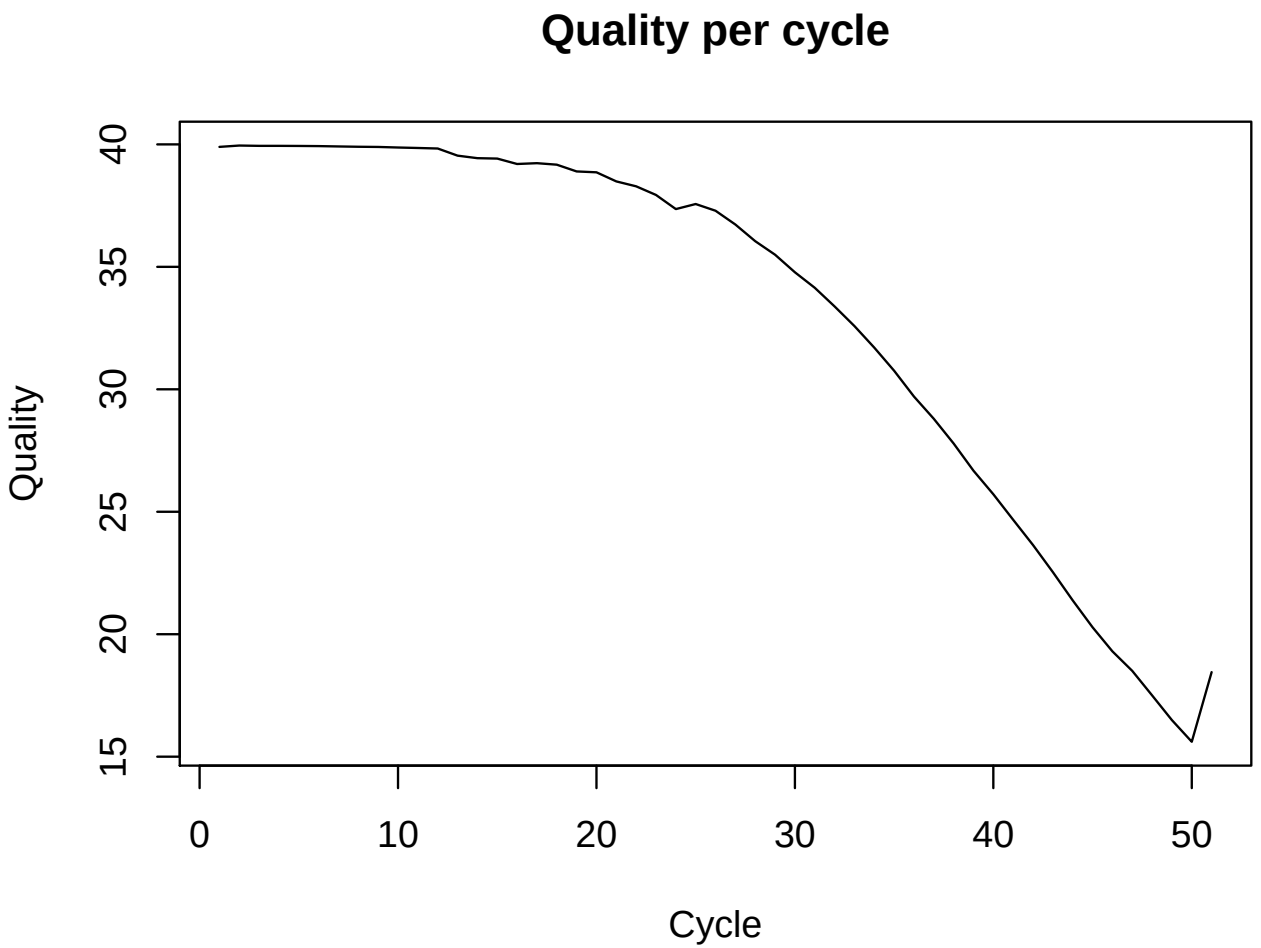
- We can also plot the qualities by cycle in order to cut the sequences when quality falls down.

```
> 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 last plot 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

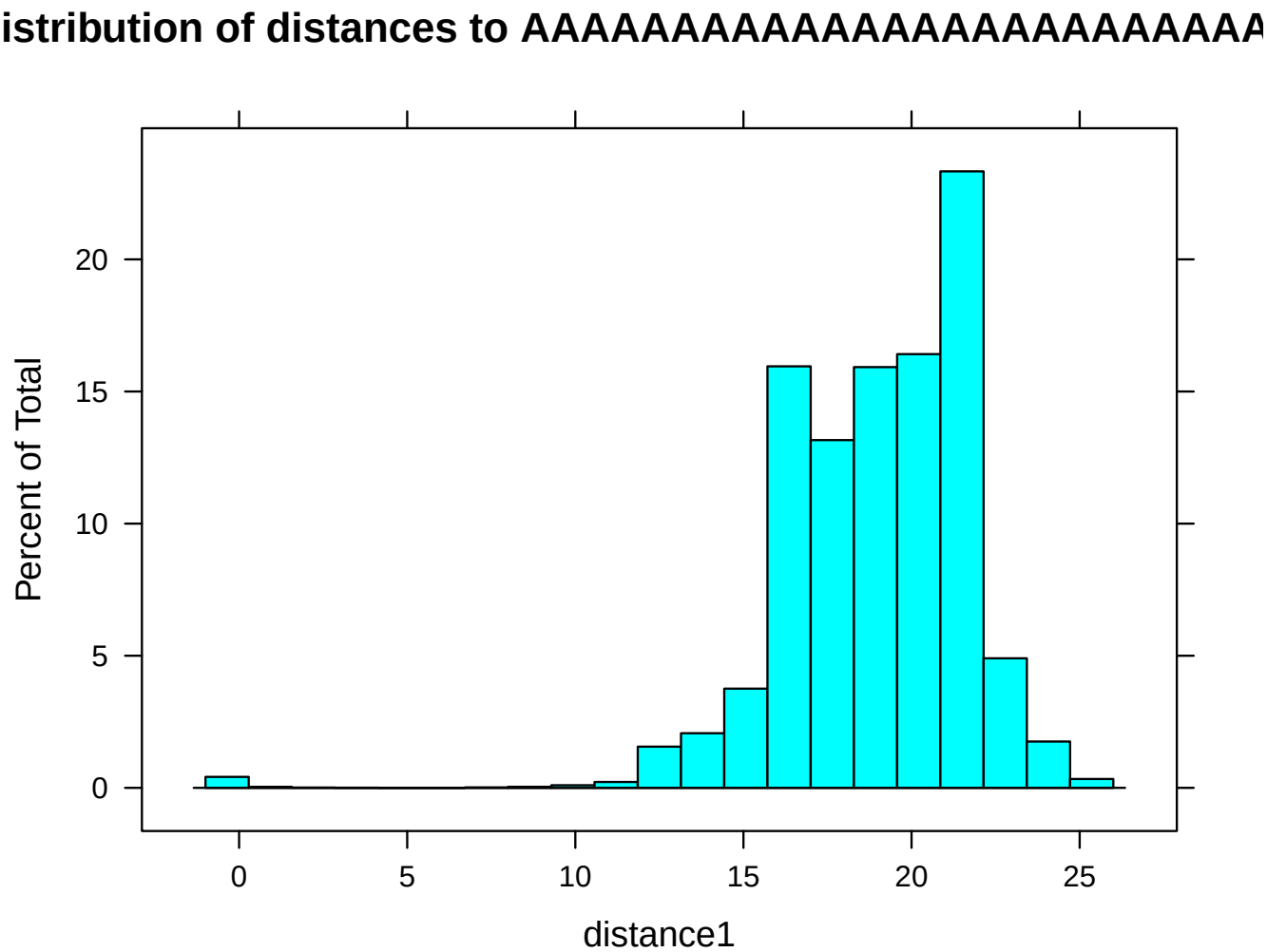
```
> reads

class: ShortReadQ
length: 1045208 reads; width: 51 cycles

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

class: ShortReadQ
length: 1045208 reads; width: 25 cycles
```

More quality

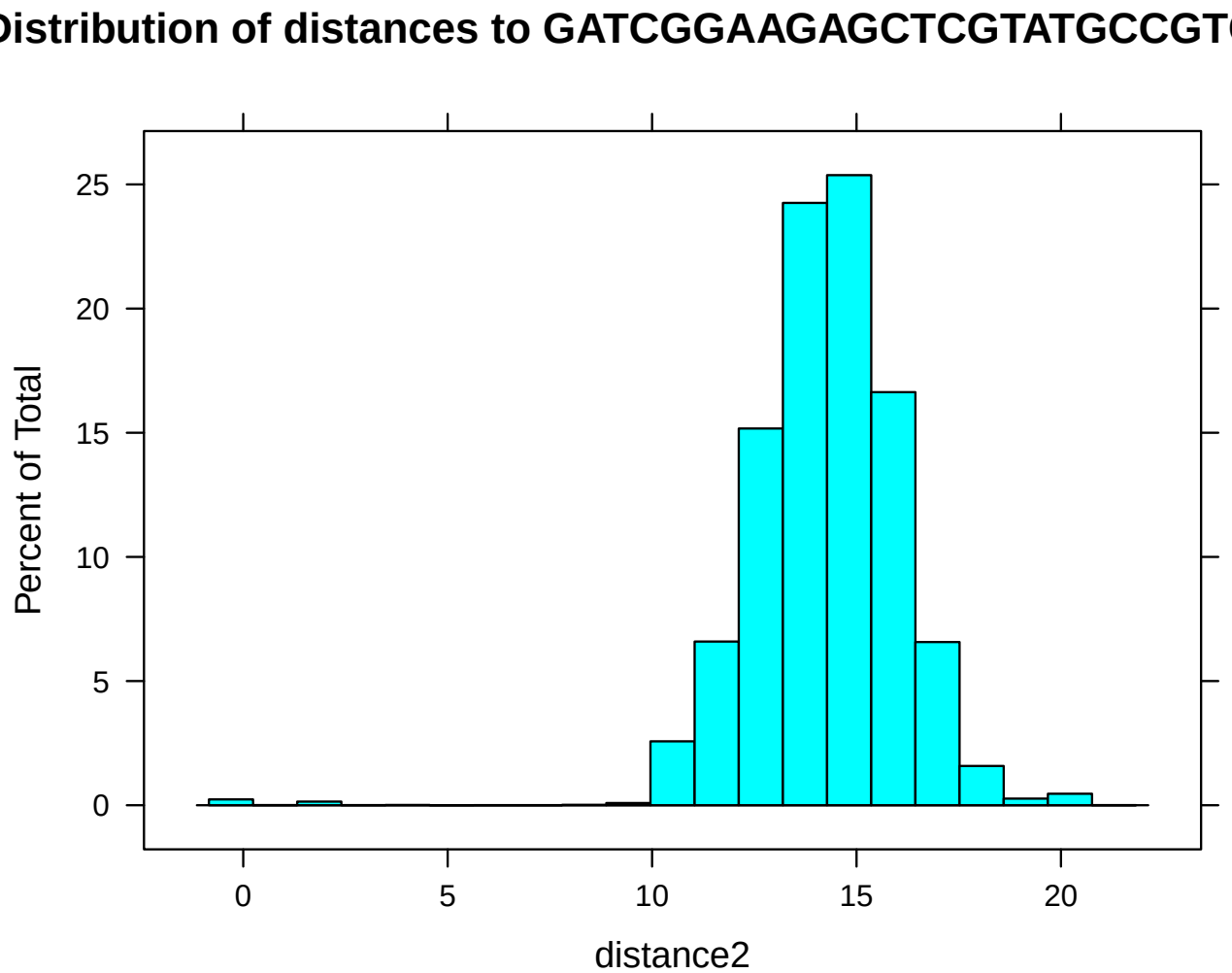
- Ok, now we have just the first 25 cycles!
- In solexa reads we have 2 sequences that are very common
- 1. AAAAAAAAAAAAAAAAAAAAAAAAA
- 2. GATCGGAAGAGCTCGTATGCCGTCT
- The function srdistance calculates de distance between two sequences, therefore it is useful to eliminate this sequences.

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

Distance


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

Distance


Distribution of distances to GATCGGAAGAGCTCGTATGCCGTCT
Percent of Total
0
5
10
15
20
25
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] 1045208

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

[1] 1036040
```

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

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

Create our own filters

- We can also create our own filters with srFilters

```
> filter <- srFilter(function(x) {
+     apply(as(quality(x), "matrix"),
+         1, sum) > 1000
+ }, name = "GoodQualityBases")
> reads[filter(reads)]

class: ShortReadQ
length: 1030888 reads; width: 51 cycles
```

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 = ".")
```

Genominator

- The Genominator package provides an interface to storing and retrieving genomic data, together with some additional functionality aimed at high-throughput sequence data.
- There are 3 broad classes of functions within Genominator: functions that import and transform data, functions that retrieve and summarize data and finally functions that operate on retrieved data (focused on analysis of next generation sequencing data).
- This package is very new and is still under development.

Import

- It uses SQLite!

```
> library(Genominator)
> aln <- readAligned("..", pattern = "andale.txt",
+     type = "Bowtie")
> aln2 <- readAligned("..", pattern = "andale.txt",
+     type = "Bowtie")
> chrMap <- levels(chromosome(aln))
> lista <- NULL
> lista <- list(Ecoli = aln, otro = aln2)
> eData <- importFromAlignedReads(lista,
+     chrMap = chrMap, dbFilename = "my.db",
+     tablename = "raw", overwrite = TRUE,
+     deleteIntermediates = FALSE)
```

Import

```
Writing table: 0.42 sec
Creating index: 0.688 sec
Creating table: __tmp_7567: 0.005 sec
inserting: 0.109 sec
droping original table: 0.019 sec
renaming table: 0.005 sec
creating index: 0.043 sec
Writing table: 0.408 sec
Creating index: 0.685 sec
Creating table: __tmp_8926: 0.004 sec
inserting: 0.109 sec
droping original table: 0.019 sec
renaming table: 0.005 sec
creating index: 0.043 sec
```

```
> head(eData)
```

Import

	chr	location	strand	Ecoli	otro
1	1	148	1	2	2
2	1	1175	-1	3	3
3	1	2580	1	1	1
4	1	2650	1	1	1
5	1	3646	1	1	1
6	1	8103	1	4	4

Import

```
> getRegion(eData, chr = 1, strand = 0,
+     start = 10000, end = 12000)

  chr location strand Ecoli otro
1   1    12000     -1     1    1

> laneCounts <- summarizeExpData(eData)
> laneCounts

Ecoli  otro
92885 92885
```

References

- http://www.lcg.unam.mx/ compu2/cei/
- http://www.bioconductor.org/workshops/2009/SeattleNov09

Session info

```
> sessionInfo()

R version 2.11.0 Under development (unstable) (2009-10-31 r50269)
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
 [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
```

Session info

```
[4] utils     datasets  methods
[7] base

other attached packages:
[1] Genominator_1.1.3
[2] RSQLite_0.7-3
[3] DBI_0.2-5
[4] BSgenome.Ecoli.NCBI.20080805_1.3.16
[5] ShortRead_1.5.10
[6] lattice_0.17-26
[7] BSgenome_1.15.2
[8] Biostrings_2.15.2
[9] IRanges_1.5.12

loaded via a namespace (and not attached):
[1] Biobase_2.7.0 grid_2.11.0
[3] hwriter_1.1
```