High-throughput sequencing with R: Mapping, Biostrings and ShortRead

Kasper Daniel Hansen
Margaret Taub
based on slides developed by
Jim Bullard

University of Copenhagen
August 17-21, 2009

Introduction

These slides will discuss mapping of sequence data as well as the Biostrings and ShortRead packages.

- Alignment and tools (mostly external to R).
- Biostrings overview.
- Alignment tools in Biostrings.
- Mapping data and reading it into R.

A comment

Analysis of high-throughput sequencing data and especially RNA-Seq is still in its infancy.

It is unclear what is the *best* way to think about and analyze these data. It is also unclear are the right entities to compute on.

We focus on some tools and some computations we have found useful.

Alignment input: FASTQ files

FASTQ files represent a common "end-point" from the various sequencing platforms (i.e. NCBI short read archive, `http://www.ncbi.nlm.nih.gov/Traces/sra/sra.cgi`). Qualities are encoded in ASCII and depending on the platform have slightly different meanings/ranges and encoding. More details can be found at: `http://en.wikipedia.org/wiki/FASTQ_format`

```
@GA-EAS46_2_209DG:6:1:890:752
TTCTCTTAAGTCTTCTAGTTCTCTTCTTTCTCCACT
+GA-EAS46_2_209DG:6:1:890:752
hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh
@GA-EAS46_2_209DG:6:1:905:558
TCTGGCTTAACTTCTTCTTTTTTTTCTTCTTCTTCT
+GA-EAS46_2_209DG:6:1:905:558
hhhfhhhhhhhhhhhhhhhhhhhhhhehhGhhJh
```

Mapping reads to the transcriptome

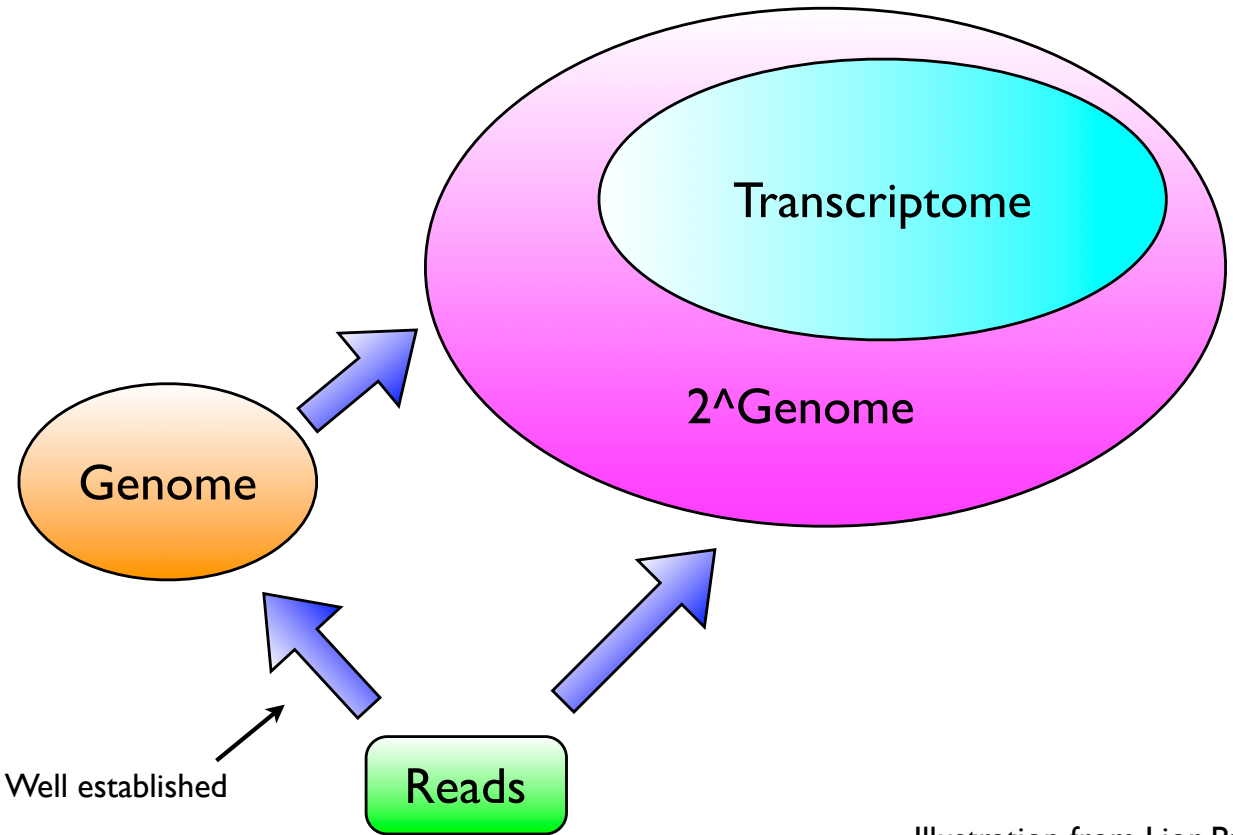
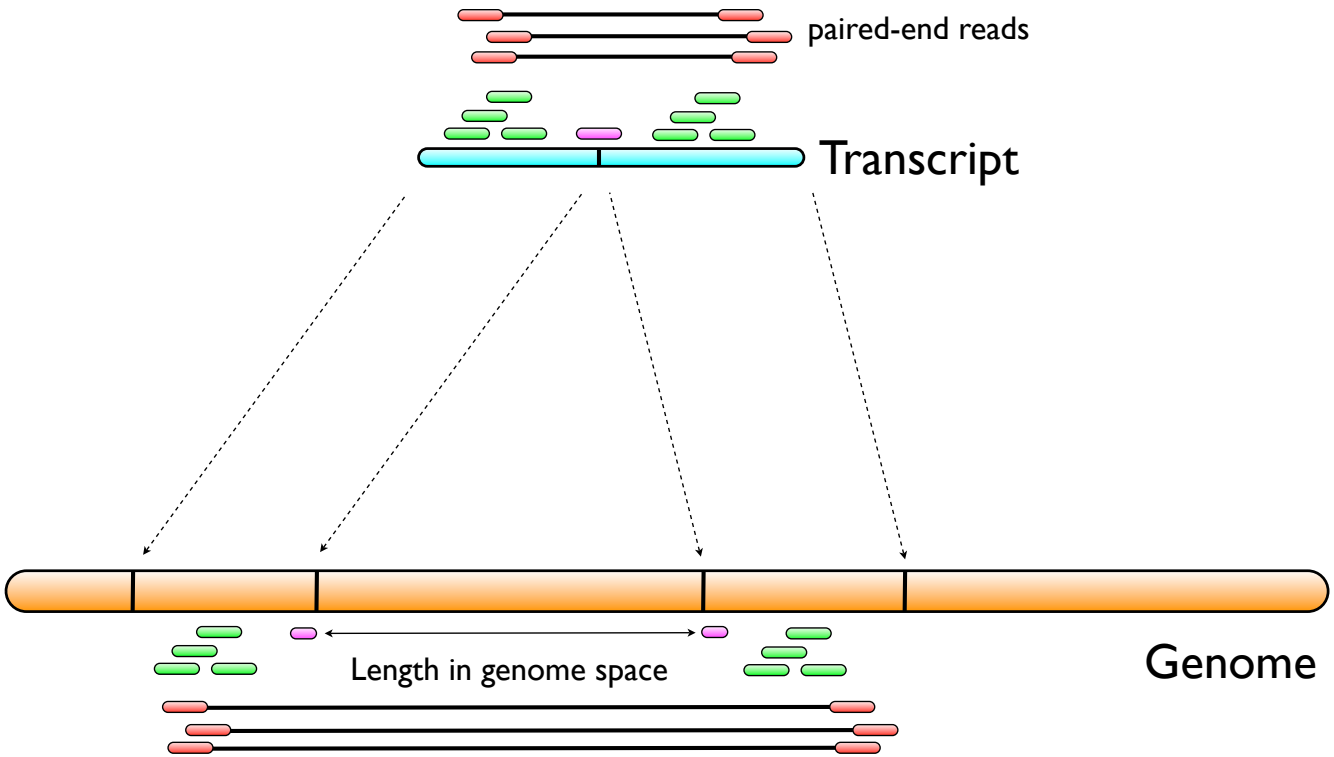


Illustration from Lior Patcher

Mapping reads to the transcriptome 2



Mapping reads to the transcriptome 3

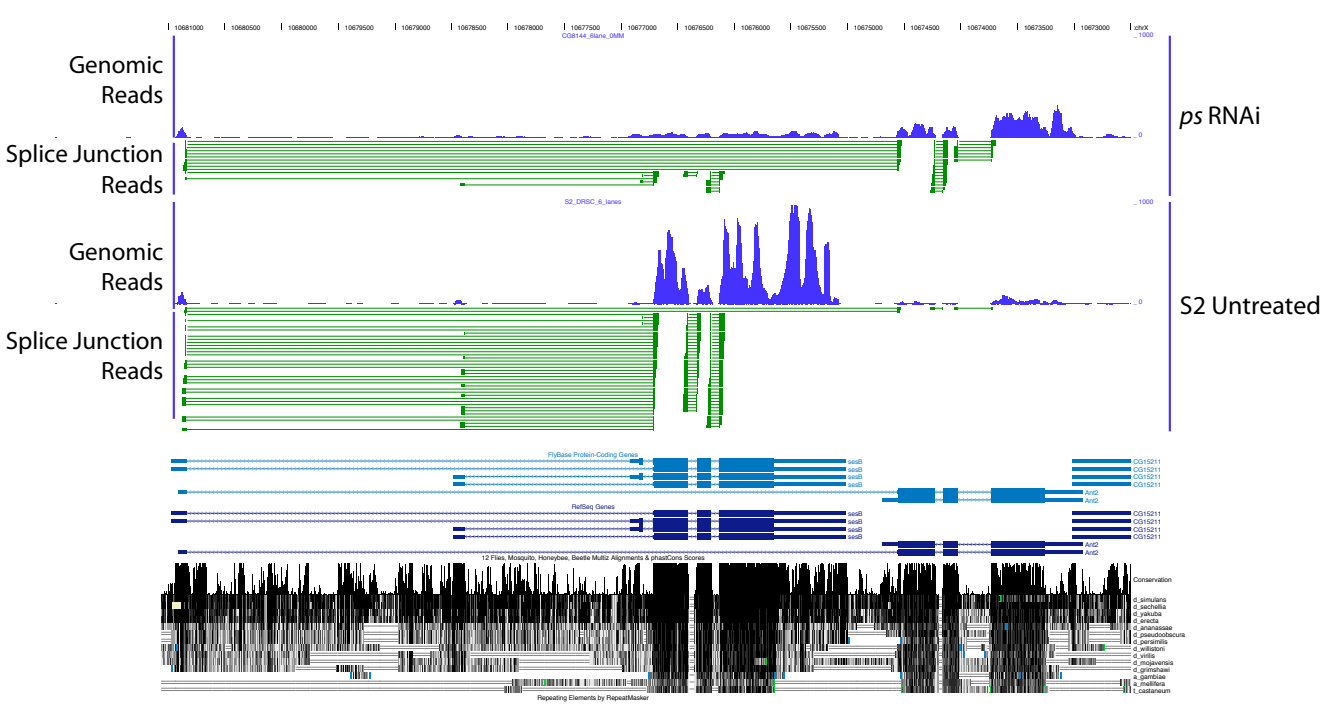


Image courtesy of Brenton Graveley.

Mapping reads to the transcriptome 4

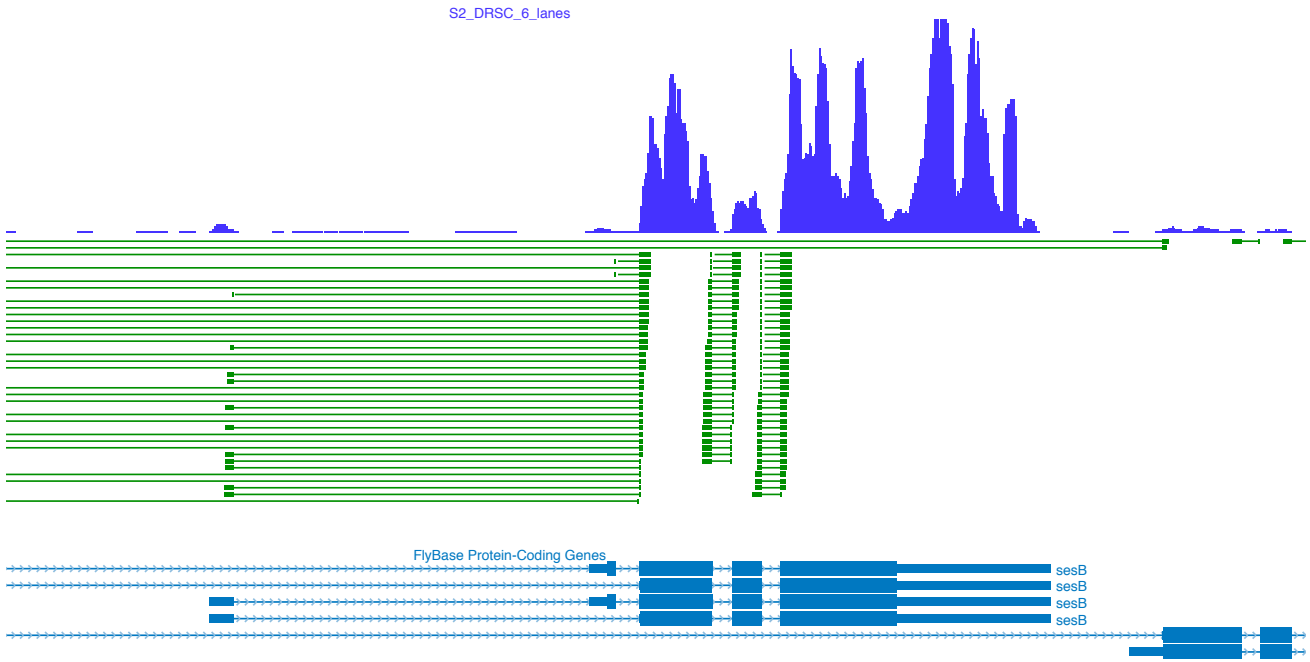
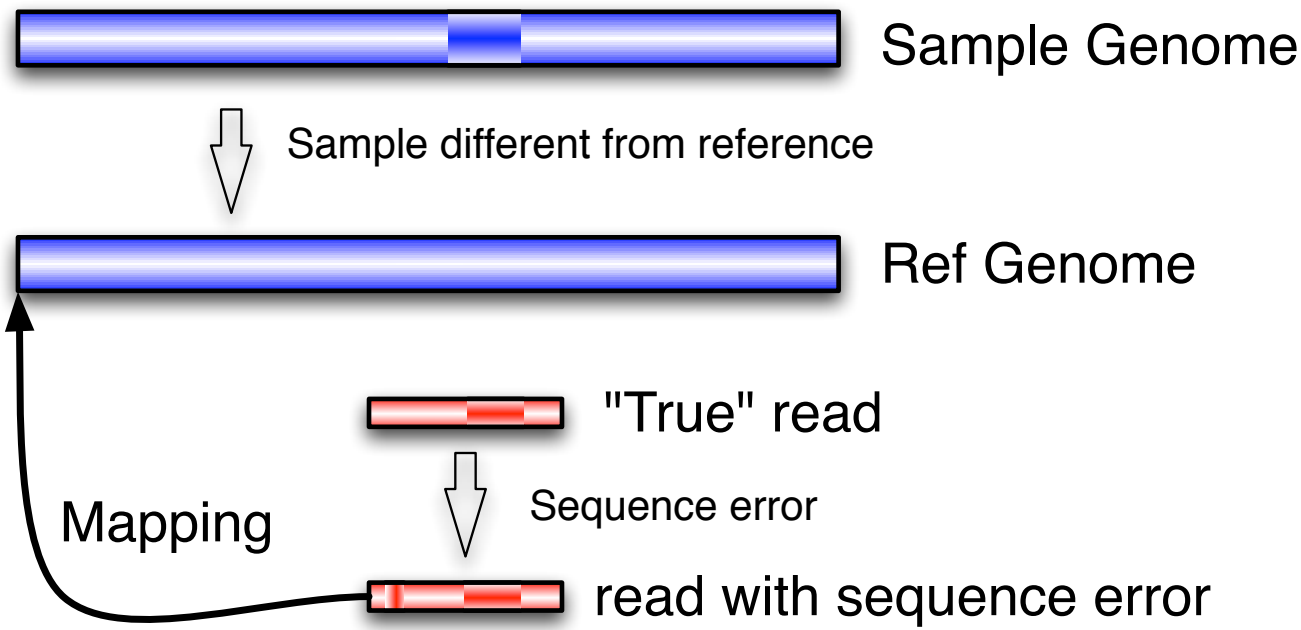


Image courtesy of Brenton Graveley.

Sequencing errors / Genome differences



Evidence *suggests* that Illumina sequencing does not introduce indels.

Alignment Tools

The number of short read aligners have exploded, but a couple tools have emerged as the *de facto* standards.

- Eland: Illumina's aligner, quality aware, fast, paired end capable
- MAQ: Good SNP caller, quality aware, paired end capable
- Bowtie: Super fast, offers different alignment strategies, paired end capable
- BWA: Fast, indel support, paired ends, qualities
- NovoAlign: MAQ like speed, many features.

A superior overview of the different aligners is available at: `http://www.sanger.ac.uk/Users/lh3/NGSalign.shtml` Additionally, a comparison of two of the best aligners can be found here: `http://www.massgenomics.org/2009/07/maq-bwa-and-bowtie-compared.html`

Common Alignment Strategies

- Use qualities (default for Bowtie and MAQ)
- Perfect match, no repeats (Strict / Lenient)
- Mismatches, no repeats
- Paired end data (PET) (harder for RNA-Seq)

The standard Illumina protocol yields unstranded reads.

Most aligners are evaluated in terms of "how many reads are mapped". Is this the right objective?

Watch out for the output of the program; there are many different conventions (0-based, what happens to read hitting the reverse strand, etc.)

SAM/BAM Formats

SAM (BAM) is a new general format for storing mapped reads. It is developed as part of the 1000 Genomes project and is quickly becoming a kind of standard. Some alignment tools output this format directly, otherwise there are scripts in `samtools` for doing it for most popular aligners. Details at `http://samtools.sourceforge.net/index.shtml`

A comment

There are no great comprehensive tools for analyzing deep-sequence data.
It will involve a fair amount of coding and gluing together various tools.

We will introduce some tools from Bioconductor that can be useful.

I use a lot of shell scripting.

Biostrings overview

- A package for working with large (biological) strings.
- Two main types of objects: A really long single string (think chromosome) or a set of shorter strings (think reads or genes). BString vs. BStringSet.
- These classes are implemented *efficiently* minimizing copy and memory loading/unloading.
- The BSgenome contains some infrastructure for whole genomes.
- Methods for dealing with biological data, including basic manipulation (complementation, translation, etc.), string searching (exact/inexact matching, Smith-Waterman, PWMs).
- Fairly complicated class structure.

> *I'm sorry to say that, at least for me, this has become hopelessly confusing, and I imagine that many other users feel the same. – Simon Anders on bioc-sig-sequencing*

Computing on genomes is not trivial. The approach to a given computation is important.

Strings in Biostrings

- BString (general), DNAString, RNAString AAString (Amino Acid), all examples of XString.
- complement, reverse, reverseComplement, translate, for the classes where "it makes sense".
- Convert to and from a standard R character string.
- Constructor: `DNAString("ACGGGGG")`.
- Support for IUPAC alphabet.
- Subsetting subseq, using the SEW format (two out of the three start/end/width). Efficient.
- StringSets are collections of Strings, like DNAStringSet.

BSgenomes

As examples, we will use whole genomes. Use available.genomes to get available genomes. Long package names, but always a shorter object name.

```
> library(BSgenome.Scerevisiae.UCSC.sacCer1)
> Scerevisiae
> Scerevisiae[[1]]
> Scerevisiae[["chr1"]]
```

A BSgenome may also have *masks*. We will ignore this for now.

Biostrings, more

- alphabetFrequency, oligonucleotideFrequency and others.

```
> alphabetFrequency(Scerevisiae[["chr1"]])
> oligonucleotideFrequency(Scerevisiae[["chr1"]],
+     width = 3)
```

- chartr for character translation ("make all As into Cs").
- Various IO functions (also ShortRead).

Views

A view is a set of substrings of an XString, stored and manipulated very efficiently. Example: to store exon sequences, one can just store the genomic location of the exons.

```
> Views(Scerevisiae[["chr1"]], start = c(300,
+     400, 500), width = 50)

  Views on a 230208-letter DNAString subject
subject: CCACACCACACCCA...GTGGTGTGTGTGGG
views:
    start end width
[1]   300 349    50 [CTGTTCTT...AAATAAC]
[2]   400 449    50 [CCCTCACT...AGTATAT]
[3]   500 549    50 [TCTCTCAC...CGGCACT]
```

A view is associated with a *subject*. Internally, they are essentially IRanges, so they are very fast to compute with. Views can in many cases be treated exactly as other strings. There are methods such as narrow, trim, gaps, restrict.

Matching in Biostrings

There are various ways of *matching* or *aligning* strings to each other. We use matching to denote searching for an exact match or possibly a match with a certain number of mismatches. Alignment denotes a more general strategy, e.g. Smith-Waterman.

- matchPattern / countPattern: match 1 sequence to 1 sequence.
- vmatchPattern / vcountPattern: match 1 sequence to many sequences.
- pairwiseAlignment: align many sequences to 1 sequence.
- matchPDict / countPDict: match many sequences to 1 sequence. ("dict" indicates that the many sequences are preprocessed into a dictionary).
- matchPWM, trimLRpattern

The different functions are optimized for different situations. They also use different algorithms, which has a big impact. Especially pairwiseAlignment is flexible and therefore has a complicated syntax.

Example: mapping probes to a genome

Get a list of Scerevisiae probes from the "yeast2" Affymetrix array.

```
> library(yeast2probe)
> ids <- scan("s_pombe.msk", skip = 2,
+     what = list(probeset = character(0),
+         junk = character(0)))$probeset
> probes <- yeast2probe$sequence[yeast2probe$Probe.Set.Name %in%
+     ids]
> probes <- DNAStringSet(probes)
```

Mapping

```
> require(BSgenome.Scerevisiae.UCSC.sacCer1)
> dict0 <- PDict(probes)
> dict0.r <- PDict(reverseComplement(probes))
> hits <- matchPDict(dict0, Scerevisiae[[1]])
> table(countIndex(hits))
> table(countPDict(dict0, Scerevisiae[[1]]))
```

Example, cont'd

Over all chromosomes

```
> allhits <- lapply(1:16, function(i) {
+     countPDict(dict0, Scerevisiae[[i]]) +
+         countPDict(dict0.r, Scerevisiae[[i]])
+ })
> table(rowSums(do.call(cbind, allhits)))
```

ShortRead

ShortRead contains tools for

- Work with GERALD/BUSTARD output.
- Generate QA report on BUSTARD files.
- Read a variety of short read data formats.

Data

We will use data from (Lee, Hansen, Bullard, Dudoit, and Sherlock (2008) PLoS Genetics).

We are considering a wild-type and a mutant strain of yeast, both grown in rich media. The two strains were sequenced using an Illumina Genome Analyzer.

We are using a subset of the data: 1M reads from each of two lanes of the two strains.

Reading the unaligned data

```
> require(ShortRead)
> fq <- readFastq("seqdata", "mut_1_f.fastq$")
> fq
```

```
class: ShortReadQ
length: 1000000 reads; width: 36 cycles
```

There are various simple accessor functions for this object, especially sread and quality.

A brief look at the unaligned data

Using alphabetByCycle and as(, "matrix") (which creates a big read times cycle matrix), we can do

```
> alp <- alphabetByCycle(sread(fq))
> matplot(t(prop.table(alp[DNA_BASES,
+     ], margin = 2)), type = "l")
> qaMat.raw <- as(quality(fq), "matrix")
> plot(colMeans(qaMat.raw))
```

Aligning the data

We now align the data using Bowtie.

```
#!/bin/bash

BOWTIE_OPTS="-m 1 -v 2 --all -p 2 -q --quiet -3 10"
BOWTIE_OPTS="-m 1 -v 0 --all -p 2 -q --quiet -3 10"
BOWTIE_OPTS="-v 2 -k 2 -p 2 -q --quiet -3 10"
BOWTIE_OPTS="-v 0 -k 1 -p 2 -q --quiet -3 10"

for f in `ls seqdata/*.fastq`
do
 bowtie s_cerevisiae $BOWTIE_OPTS $f > $f.bowtie
done
```

How many hits do we get?

Reading in the aligned data

We read the data into R as a 4-component list (one lane per component).

```
> files <- list.files("seqdata",
+     pattern = "\\.bowtie")
> aligned <- sapply(files, function(f) {
+     readAligned("seqdata", pattern = f,
+         type = "Bowtie")
+ })
> names(aligned) <- gsub("_f.fastq.bowtie",
+     "", names(aligned))
> aligned[[1]]
> save(aligned, file = "aligned.rda")
```

Some exercises

1. Determine the number of times the motif "TATAA" occurs in the yeast genome. How often does it occur with one mismatch?
2. The `seqdata` directory has an object called `ste12` which is a position weight matrix for the transcription factor "Ste12" (obtained from SCPD). Where does it occur in the genome?
3. Compute the average quality for each cycle for the aligned reads and compare to the qualities for the unaligned reads. What is the difference?

Solution 1

```
> sum(sapply(1:16, function(i) {
+     motif <- DNAString("TATAA")
+     countPattern(motif, Scerevisiae[[i]]) +
+         countPattern(reverseComplement(motif),
+             Scerevisiae[[i]])
+ }))
```

Solution 2

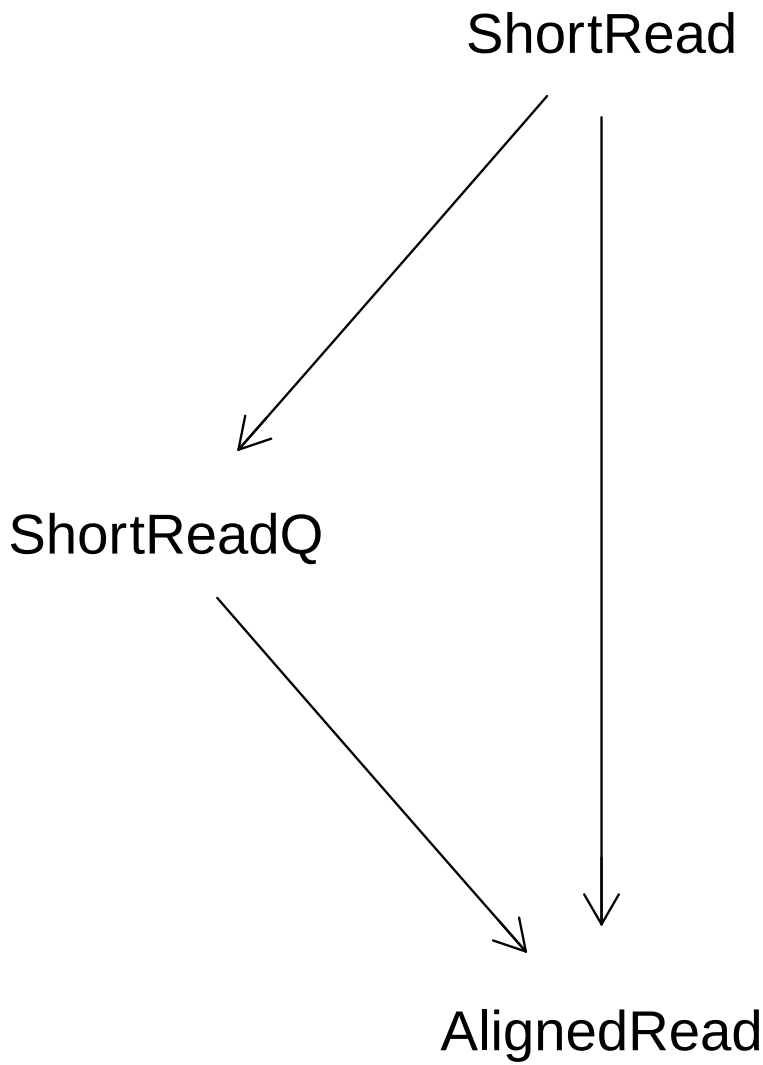
Solution 3

SessionInfo

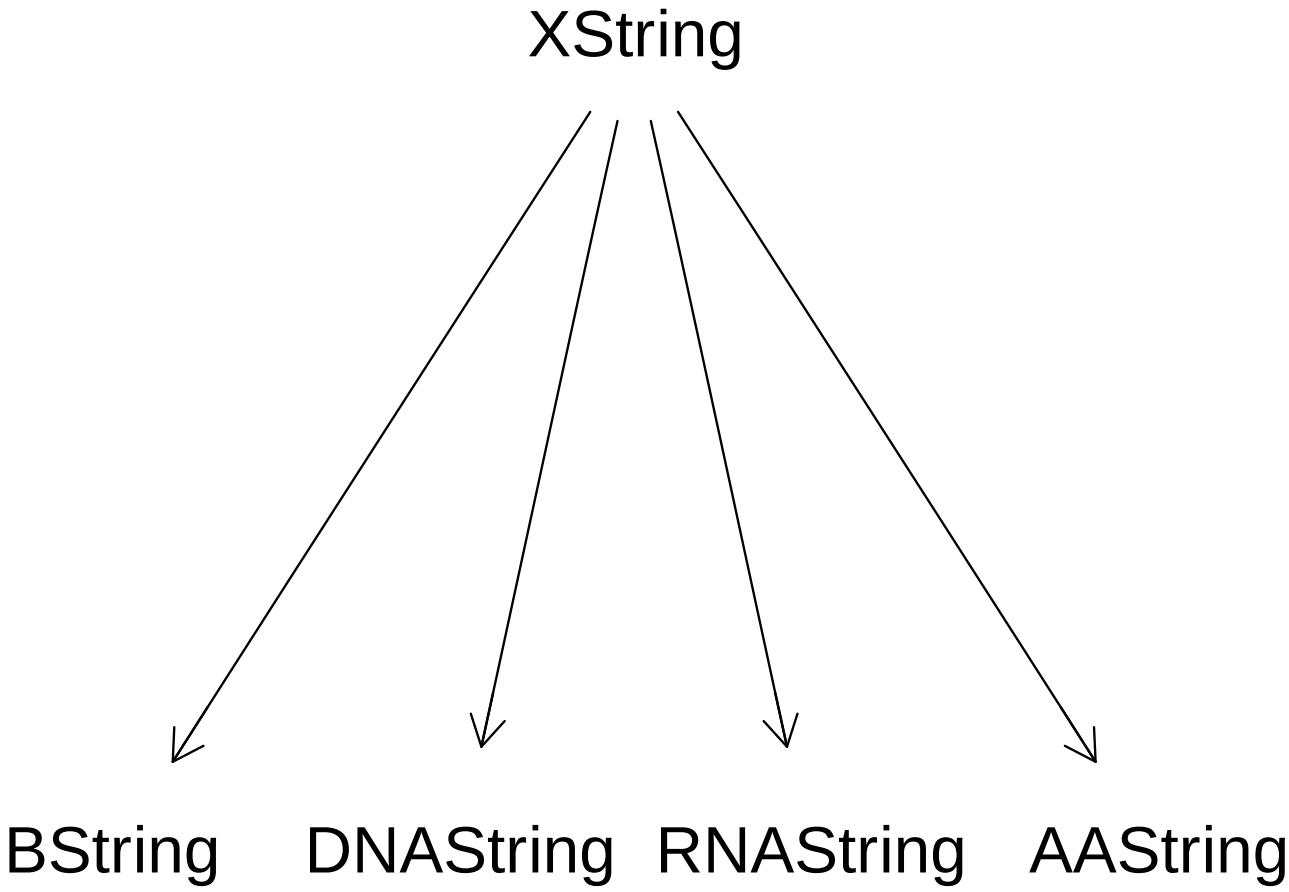
```
> toLatex(sessionInfo())
```

- R version 2.9.2 RC (2009-08-17 r49312), `i386-apple-darwin9.8.0`
- Locale: `en_US.UTF-8/en_US.UTF-8/C/C/en_US.UTF-8/en_US.UTF-8`
- Base packages: base, datasets, graphics, grDevices, grid, methods, stats, utils
- Other packages: Biostrings 2.12.8, BSgenome 1.12.3, BSgenome.Scerevisiae.UCSC.sacCer1 1.3.13, classGraph 0.7-2, graph 1.22.2, IRanges 1.2.3, lattice 0.17-25, Rgraphviz 1.23.4, ShortRead 1.2.1
- Loaded via a namespace (and not attached): Biobase 2.4.1, hwriter 1.1, tools 2.9.2

Some ShortRead classes



Some Biostrings classes (single strings)



Some Biostrings classes (sets of strings)