HTSeqGenie: a software package to analyse high-throughput sequencing experiments

Gregoire Pau, Cory Barr, Jens Reeder, Michael Lawrence
Jeremiah Degenhardt, Tom Wu, Melanie Huntley, Matt Brauer

August 26, 2014

Contents

1 Introduction

The *HTSeqGenie* package is a robust and efficient software to analyze high-throughput sequencing experiments in a reproducible manner. It supports the RNA-Seq and Exome-Seq protocols and provides: quality control reporting (using the *ShortRead* package), detection of adapter contamination, read alignment versus a reference genome (using the *gmapR* package), counting reads in genomic regions (using the *GenomicRanges* package), and read-depth coverage computation.

2 Example analysis of two RNA-Seq experiments of lung cell lines

2.1 Analysis on a genomic region centered on TP53

In this section, we are analysing two RNA-Seq experiments of human lung cell lines (H1993 and H2073) on a 2 Mb genomic region centered on the TP53 gene. This region, not the full human genome, has been chosen to provide a vignette that can be run in a reasonable amount of time.

We first load the package *HTSeqGenie*. We then load the package *LungCancerLines* to get the FASTQ files of the lung cancer cell lines we are going to analyze. These files only cover the TP53 genomic region and have been subsampled to 2500 reads, due to vignette time constraints.

```
> library("HTSeqGenie")
> library("LungCancerLines")
```

To analyze the reads, the package needs a `GmapGenome` object, which contains the information needed to align the reads to a reference genome. The package also needs a genomic features object, which is a named list of `GRanges` objects containing the genomic features (gene, exon, transcript...) of the corresponding genome. For convenience, this vignette provides the function `TP53Genome`, generating a 2 Mb subset of the UCSC hg19 human genome, centered on the gene TP53, and the `TP53GenomicFeatures`, producing a RData file containing the associated genomic features. The next section explains how to build custom `GmapGenome` and genomic features objects.

```
> tp53Genome <- TP53Genome()
> tp53GenomicFeatures <- TP53GenomicFeatures()
```

The samples are analysed with a common set of configuration parameters, specific to RNA-Seq paired-end experiments. Details about these parameters are provided in the Configuration section.

```
> rtp53 <- list(
+               ## aligner
+               path.gsnap_genomes=path(directory(tp53Genome)),
+               alignReads.genome=genome(tp53Genome),
+
+               ## gene model
+               path.genomic_features=dirname(tp53GenomicFeatures),
+               countGenomicFeatures.gfeatures=basename(tp53GenomicFeatures)
+               )
```

The function `runPipeline` runs the full analysis pipeline, using a list of configuration parameters. The following commands analyse the H1993 and H2073 samples. The function returns the path to a directory that contains the full analysis: bams, QC reports and a results directory which include coverage and count data.

```
> H1993dir <- do.call(runPipeline,
+                     c(## RNASeq TP53genome parameters
+                       rtp53,
+
+                       ## input
+                       input_file=LungCancerFastqFiles()[["H1993.first"]],
+                       input_file2=LungCancerFastqFiles()[["H1993.last"]],
+                       paired_ends=TRUE,
+                       quality_encoding="illumina1.8",
+
+                       ## output
+                       save_dir="H1993",
+                       prepend_str="H1993",
+                       overwrite_save_dir="erase"
+                       ))
> H2073dir <- do.call(runPipeline,
+                     c(## RNASeq TP53genome parameters
+                       rtp53,
+
+                       ## input
+                       input_file=LungCancerFastqFiles()[["H2073.first"]],
+                       input_file2=LungCancerFastqFiles()[["H2073.last"]],
+                       paired_ends=TRUE,
+                       quality_encoding="illumina1.8",
+
+                       ## output
+                       save_dir="H2073",
+                       prepend_str="H2073",
+                       overwrite_save_dir="erase"
+                       ))
```

Several quality control reports (short reads, alignment, counting) have been generated in the `report/` directories of the paths pointed by `H1993dir` and `H2073dir`. The following example uses the function `getTabDataFromFile` to get the read counts per gene in both cell lines. Further analysis such as differential expression using replicates can be performed using the package *DESeq*.

```
> library("org.Hs.eg.db")
> gc1 <- getTabDataFromFile(H1993dir, "counts_gene")
> gc2 <- getTabDataFromFile(H2073dir, "counts_gene")
> entrez <- as.character(gc1$name)
> hgnc <- unlist(as.list(org.Hs.egSYMBOL)[entrez])
> hgnc <- hgnc[entrez]
> data.frame(entrez=entrez, hgnc=hgnc, H1993.count=gc1$count, H2073.count=gc2$count, row.names=NULL)
```

```
       entrez            hgnc H1993.count H2073.count
1   100128288    LOC100128288           0           0
2   100422983         MIR4314           0           0
3   100500892            <NA>           0           0
4   100506713    LOC100506713           3           1
5   100506755        MIR497HG          16          16
6   100529209 RNASEK-C17orf49          42          72
7   100529211  TMEM256-PLSCR3          37          33
```

8	100533955	SENP3-EIF4A1	471	479
9	100616406	MIR4521	0	0
10	10462	CLEC10A	0	0
11	1107	CHD3	132	112
12	112483	SAT2	6	12
13	11337	GABARAP	41	56
14	1140	CHRNB1	0	2
15	116840	CNTROB	7	27
16	118432	RPL29P2	0	0
17	124637	CYB5D1	25	17
18	124751	KRBA2	3	2
19	1366	CLDN7	89	30
20	146754	DNAH2	3	2
21	146852	ODF4	0	0
22	147040	KCTD11	7	3
23	162515	SLC16A11	0	0
24	162517	FBXO39	0	0
25	1742	DLG4	1	4
26	1856	DVL2	25	28
27	1949	EFNB3	0	4
28	1984	EIF5A	301	288
29	201232	SLC16A13	4	5
30	201243	C17orf74	0	0
31	2256	FGF11	0	7
32	22899	ARHGEF15	0	0
33	23135	KDM6B	17	8
34	23399	CTDNEP1	49	71
35	23587	ELP5	23	23
36	239	ALOX12	2	0
37	242	ALOX12B	0	0
38	245	ALOX12P2	1	2
39	247	ALOX15B	0	0
40	254863	TMEM256	26	17
41	255877	BCL6B	2	1
42	26168	SENP3	79	47
43	26781	SNORA67	1	2
44	284023	LOC284023	0	5
45	284029	LINC00324	1	0
46	284111	SLC13A5	0	1
47	284114	TMEM102	4	2
48	2874	GPS2	43	52
49	29098	RANGRF	5	16
50	3000	GUCY2D	0	0
51	339168	TMEM95	0	0
52	37	ACADVL	148	149
53	374768	SPEM1	0	0
54	399512	SLC25A35	5	19
55	407977	TNFSF12-TNFSF13	3	22
56	432	ASGR1	0	0
57	433	ASGR2	0	0
58	442898	MIR324	0	0

59	4628	MYH10	127	79
60	482	ATP1B2	0	0
61	51087	YBX2	12	13
62	5187	PER1	3	18
63	5198	PFAS	18	39
64	5430	POLR2A	154	160
65	54739	XAF1	0	0
66	54785	C17orf59	2	1
67	55135	WRAP53	18	15
68	57048	PLSCR3	11	14
69	574456	MIR497	0	0
70	57555	NLGN2	0	11
71	57659	ZBTB4	29	47
72	58485	TRAPPC1	36	50
73	59344	ALOXE3	0	1
74	6154	RPL26	206	140
75	643664	SLC35G6	0	0
76	643904	RNF222	0	0
77	6462	SHBG	7	13
78	6517	SLC2A4	0	1
79	652965	SNORA48	0	0
80	652966	SNORD10	0	0
81	6665	SOX15	0	0
82	677763	SCARNA21	0	0
83	6844	VAMP2	6	14
84	7157	TP53	82	69
85	79142	PHF23	16	19
86	80169	CTC1	25	15
87	81565	NDEL1	30	32
88	83659	TEKT1	0	0
89	84314	TMEM107	17	8
90	84316	NAA38	12	8
91	84461	NEURL4	53	67
92	84667	HES7	1	0
93	8711	TNK1	12	4
94	8741	TNFSF13	1	21
95	8742	TNFSF12	2	1
96	9196	KCNAB3	0	0
97	9212	AURKB	23	23
98	92162	TMEM88	0	0
99	9513	FXR2	37	33
100	9526	MPDU1	58	51
101	968	CD68	63	92
102	9744	ACAP1	0	1

2.2 Analysis on the full human genome

This section will be completed later.

3 Analysis steps

3.1 General

The pipeline consists in a set of high-level modules that are configured with a configuration file. See the Configuration section for a list of all parameters.

The modules are:

- preprocessing (preprocessReads)
- read alignment (alignReads)
- read counting (countGenomicFeatures)
- computation of coverage (calculateCoverage)

The following diagram shows a high-level overview of the performed analysis steps:
The pipeline produces the directory logs/, which contains log files about the analysis:

- logs/audit.txt: the environment used to run the analysis (pipeline version number, host, R session, PATH, gsnap version, samtools version)
- logs/config.txt: the expanded configuration file used to run the analysis
- logs/progress.log: the progress log

3.2 Preprocessing

The preprocessing module applies a sequence of optional steps before read alignment. These optional operations are:

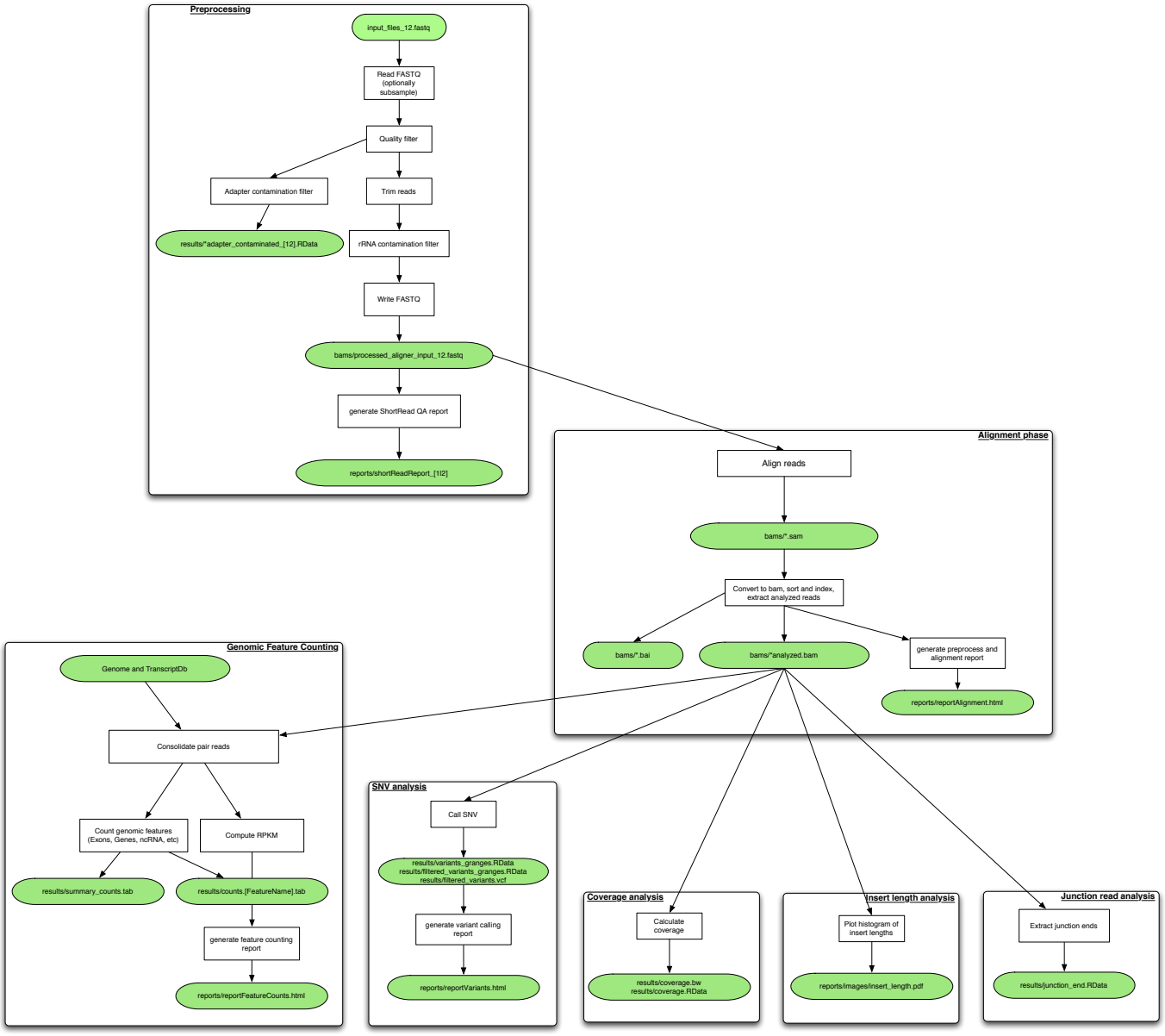
- read subsampling (randomly sample a subset of reads from input data)
- trimming reads end to a fixed size
- filtering reads based on nucleotide qualities
- detection of (but not filter) adapter-contaminated reads
- creation of ShortRead reports (on a random subset of reads)

Output

- bams/processed.aligner_input.fastq: processed FASTQ file(s) given to the aligner
- results/adapter_contaminated.RData: RData files containing the names of the detected adapter-contaminated reads
- results/summary_preprocess.tab: a tab-separated file containing a summary of the preprocessing step
- reports/shortReadReport: HTML ShortRead quality reports

3.3 Alignment

The alignment module uses the gsnap aligner to align reads, and creates the file analyzed.bam, containing reads that are analyzed by the downstream modules. This file currently contains all uniquely mapping reads (i.e. this is the concatenation of the *uniq.bam files).

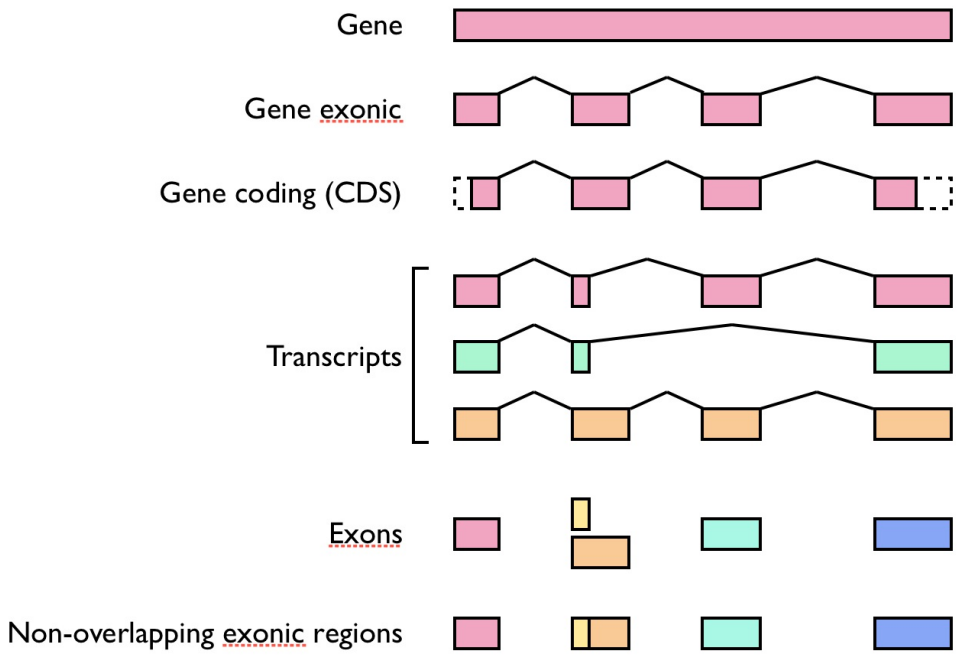


Output

- bams/*.bam: output aligned bam files
- bams/*.bai: corresponding bam indexes
- results/summary_alignment.tab: a tab-separated file containing summary alignment information
- report/reportAlignment.html: an HTML summary report about alignment

3.4 Count genomic features

This module counts how many reads overlaps with the genomic intervals defined in RData file pointed by the "countGenomicFeatures.gfeatures" parameter. On each interval, RPKM is computed using: (1e9*count) / (nbreads * width), where count is the number of reads overlapping with the interval, nbreads is the total number of reads being analysed and width is the width of the interval.



The genomic features are defined as:

- A gene region is an interval (i.e. connected) and includes introns, 5' and 3' UTRs.
- A gene exonic region is an interval union of all exons of a particular gene. It includes 5' and 3' UTRs, but not introns.
- A gene coding region is a gene exonic region without the 5' and 3' UTRs.
- A transcript is an interval union of the exons of a particular transcript. It includes 5' and 3' UTRs, but not introns.
- Non-overlapping exonic regions (DEXSeq) are exons that have been split to remove overlapping regions.

Output

- results/counts_feature_type.tab: a tab-separated file containing counts, widths and RPKMs
- results/summary_counts.tab: a tab-separated file containing summary count information
- report/reportFeatureCounts.html: a HTML summary report about count genomic features

3.5 Coverage

This module computes coverage (read-depth) data.

Output

- results/coverage.RData: a RData file containing coverage data
- results/coverage.bw: bigwig file for IGB

4 Configuration

4.1 General

Configuration files are written in a DCF format, a sequence of lines containing "<parameter>:<value>". The format supports comments starting with a sharp ('#') sign and is whitespace tolerant. The HTSeqGenie pipeline uses a templated configuration framework where configuration files can inherit parameters from other configuration files, through the parameter "template_config". Parameters defined after "template_config" override parameters from the inherited templates.

Default parameters depend on the configuration templates.

4.2 Configuration parameters

Configuration files ultimately derive from the default-config.txt configuration file, which enumerates all available parameters. The following sections enumerate all available configuration parameters.

Template

- template_config: An optional file name containing a template configuration file. To locate it, the software first looks in the directory defined by the environment variable HTSEQGENIE_CONFIG and, if not found, in the installed package HTSeqGenie. No default value.

Input

- input_file: The path of a FASTQ or gzipped FASTQ file containing the input reads. No default value.
- input_file2: An optional path of a FASTQ or gzipped FASTQ file containing the input paired reads in paired-end sequencing. If present, the parameter paired_ends must be set to TRUE. No default value.
- paired_ends: A logical indicating whether the reads are coming from paired-end sequencing. Default is TRUE.
- quality_encoding: An optional string indicating which quality encoding is used. Possible values are "sanger", "solexa", "illumina1.3", "illumina1.5" and "illumina1.8". If absent, a non-robust strategy is tried to guess it. No default value.
- subsample_nbreads: An optional integer. If present, the preprocess module randomly subsamples the indicated number of reads from the input files before analysis. This feature is disabled by default.
- chunk_size: An integer indicating how many reads are present in a chunk. This parameter should not be changed. Default is 1e6.
- max_nbchunks: An optional integer. If present, it sets the maximum number of chunks to be processed. This parameter is mostly present for debug purposes. This feature is disabled by default.

Output

- save_dir: A directory path indicating where the output results should be stored. No default value.
- overwrite_save_dir: A string indicating how to overwrite/save data in the save_dir directory. Possible values are "never", "erase" and "overwrite". If "never", the pipeline will stop if the indicated save_dir is present. If "erase", the save_dir directory is removed before starting the analysis. Default is "never".
- prepend_str: A string that will be preprended to output filenames. No default value.
- remove_processedfastq: A logical indicating whether the preprocessed FASTQ files before alignment should be removed. Default is TRUE.
- remove_chunkdir: A logical indicating whether the chunks/ subdirectory should be removed. Default is TRUE.
- tmp_dir: A temporary directory where intermediate files will be written to. If set, this should point to a non-replicated, non-snapshoted directory such as /gne/research/data/bioinfo/ngs_analysis/CGP_3.0/cgptmp. If not set, temporary files will be written into the save_dir.

System

- num_cores: An integer indicating how many cores should be used on this machine. This value can be overriden by the ngs_pipeline calling script. Default value is 1.

Path

- path.genomic_features: A path to the directory that contains the genomic features RData files used for counting. No default.
- path.gsnap_genomes: A path to the directory that contains Gsnap's genomic indices. No default.

Debug

- debug.level: A string indicating which maximal level of debug should be used in the log file. Possible values are "ERROR", "WARN", "INFO" and "DEBUG". Default is "DEBUG".
- debug.tracemem: A logical indicating whether periodic memory information should be included in the log file. Default is TRUE.

Trim reads

- trimReads.do: A logical indicating whether the reads should be end-trimmed. Default is FALSE.
- trimReads.length: An integer indicating to what size (in nucleotides) the reads should be trimmed to. No default value.

Filter quality

- filterQuality.do: A logical indicating whether the reads should be filtered out based on read nucleotide qualities. A read is kept if the fraction "filterQuality.minFrac" of nucleotides have a quality higher than "filterQuality.minQuality". Default is TRUE.
- filterQuality.minQuality: An integer. See filterQuality.do. Default is 23.
- filterQuality.minFrac: A numeric ranging from 0.0 to 1.0. See filterQuality.do. Default is 0.7.

Detect adapter contamination

- detectAdapterContam.do: A logical indicating whether reads that look contaminated by adapter sequences should be reported (but NOT filtered). Default is TRUE.
- detectAdapterContam.force_paired_end_adapter: A logical indicating whether paired end adapter chemistry was used for single-end sequencing. Default is FALSE.

Detect ribosomal RNA

- detectRRNA.do: A logical indicating whether ribosomal RNAs should be filtered out. Default is TRUE.
- detectRRNA.rrna_genome: A string indicating which gsnap genome index to use (-d flag) to detect ribosomal RNAs. No default value.

ShortRead report

- shortReadReport.do: A logical indicating whether a ShortRead report should be generated from the preprocessed reads. Default is TRUE.
- shortReadReport.subsample_nbreads: An optional integer indicating how many reads should be subsampled from the preprocessed reads for generating the report. No value indicates to take all of them (can be memory and time consuming). Default is 20e6.

Aligner

- alignReads.genome: A string indicating which gsnap genome index to use (-d flag). No default value.
- alignReads.max_mismatches: An optional integer indicating how many maximal mismatches gsnap should use (-m flag). If absent, gsnap uses an automatic value. No default value.
- alignReads.sam_id: A string (–read-group-id flag). No default value.
- alignReads.snp_index: An optional string containing gsnap SNP database index (-v flag). No default value.
- alignReads.splice_index: An optional string containing gsnap splice index (-s flag). No default value.
- alignReads.static_parameters: An optional string containing extra gsnap parameters. No default value.
- alignReads.nbthreads_perchunk: An optional integer indicating how many threads should be used to process one chunk (-t flag). If unspecified, the value is set to min(num_cores, 4). This parameter is mostly given for debug purposes. No default value.

Count genomic features

- countGenomicFeatures.do: A logical indicating whether counts per genomic feature should be computed. Default is TRUE.
- countGenomicFeatures.gfeatures: A filename containing the RData file used by the counting features module. The full path is "path.genomic_features"/ "countGenomicFeatures.gfeatures". No default value.

Variants

- analyzeVariants.do: A logical indicating whether variant calling should be performed. Default is TRUE.
- analyzeVariants.use_read_length: FALSE
- analyzeVariants.with_qual: A logical indicating whether variant calling should take mapping quality into account. Default is TRUE.
- analyzeVariants.bqual: Default is 23.
- analyzeVariants.bin_fraction: Default is 0.1.

Coverage

- coverage.extendReads: A logical whether reads should be extended in the coverage vector. Default is FALSE
- coverage.fragmentLength: Amount by which single ended reads should be extended. Usually defined by the fragment length. No default value.

5 Session Information

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

- R version 3.1.1 (2014-07-10), `x86_64-unknown-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: AnnotationDbi 1.26.0, BSgenome 1.32.0, Biobase 2.24.0, BiocGenerics 0.10.0, BiocParallel 0.6.1, Biostrings 2.32.1, DBI 0.2-7, GenomeInfoDb 1.0.2, GenomicAlignments 1.0.6, GenomicRanges 1.16.4, HTSeqGenie 3.14.1, IRanges 1.22.10, LungCancerLines 0.2.0, RSQLite 0.11.4, Rsamtools 1.16.1, ShortRead 1.22.0, VariantAnnotation 1.10.5, XVector 0.4.0, gmapR 1.6.6, org.Hs.eg.db 2.14.0
- Loaded via a namespace (and not attached): BBmisc 1.7, BatchJobs 1.3, Cairo 1.5-6, GenomicFeatures 1.16.2, Matrix 1.1-4, RColorBrewer 1.0-5, RCurl 1.95-4.3, VariantTools 1.6.1, XML 3.98-1.1, biomaRt 2.20.0, bitops 1.0-6, brew 1.0-6, checkmate 1.3, chipseq 1.14.0, codetools 0.2-9, digest 0.6.4, fail 1.2, foreach 1.4.2, grid 3.1.1, hwriter 1.3.1, iterators 1.0.7, lattice 0.20-29, latticeExtra 0.6-26, rtracklayer 1.24.2, sendmailR 1.1-2, stats4 3.1.1, stringr 0.6.2, tools 3.1.1, zlibbioc 1.10.0