Basic4Cseq: an R/Bioconductor package for the analysis of 4C-seq data

Carolin Walter

June 30, 2014

Contents

1 Introduction

Chromosome conformation capture combined with high-throughput sequencing (4C-seq) is a method that allows the identification of chromosomal interactions between one potential interaction partner, called viewpoint, and virtually any other part of the genome, without prior knowledge of potential interaction partners [van de Werken *et al.*, 2012]. Unlike regular NGS data, the raw short reads from a 4C-seq experiment can only originate from precisely defined points in the genome. Special care has therefore be taken to filter non-valid reads and to avoid forms of bias due to differences in fragment features.

A typical workflow for the analysis of 4C-seq data consists of the following steps [van de Werken *et al.*, 2012, Gheldof *et al.*, 2012]:

1. Virtual fragment library creation
2. Mapping from reads to fragments
3. Fragment-based analysis and processing (filtering, normalization, quality controls...)
4. Visualization

Due to the high coverage around the experiment's viewpoint and sparse remote data, different routines for near-cis and far-cis / trans visualization are necessary. Statistical enrichment approaches are advisable for the analysis of genomic areas remote from the viewpoint [Splinter *et al.*, 2012].

Basic4Cseq can create virtual fragment libraries for BSGenome packages, map provided reads to fragments, and offers functions for basic filtering and normalization of the 4C-seq data. Near-cis visualization routines are included, as well as a function for the visualization of imported trans-interaction intervals. Basic quality controls (cp.[van de Werken *et al.*, 2012]) offer further information on the read distribution of the 4C-seq data.

1.1 Loading the package

After installation, the package can be loaded into R by typing

```
> library(Basic4Cseq)
```

into the R console.

1.2 Provided functionality

Basic4Cseq requires the R-packages *ShortRead*, *Biostrings*, *caTools*, and *GenomicRanges*. While the package *BSgenome.Ecoli.NCBI.20080805* is used for example purposes, the package *BSgenome.Hsapiens.UCSC.hg19* or any respective corresponding BSgenome package (e.g. for Mus musculus) is suggested for the analysis of real 4C-seq data. Optional visualization of imported long-range interaction intervals requires the package *RCircos*.

This package provides the following basic functions for the analysis of 4C-seq data:

- `createVirtualFragmentLibrary`: Creation of virtual fragment libraries for BSgenome packages, including filtering options on fragment level (size of fragments and fragment ends, presence of second restriction enzyme cutting site)
- `readsToFragments`: Mapping functionality from reads to 4C-seq fragments
- `getReadDistribution`: Quality control based on the read distribution in fragment ends
- `normalizeFragmentData`: Normalization of fragment read count (RPM)
- `visualizeViewpoint`: Near-cis visualization (coverage plot with running median / running mean smoothing and quantile visualization)
- `drawHeatmap`: Near-cis domainogram visualization (heatmap-like multi-scale contact profiles)

Optional functions include:

- `printWigFile`: Export of fragment data as wig-files

- `plotTransInteractions`: Import and visualization of trans interaction intervals
- `prepare4CseqData`: Wrapper for the alignment of FASTQ files and filtering of the resulting SAM / BAM files, if external tools (BWA [Li *et al.*, 2009], SAMtools [Li *et al.*, 2009], BEDtools [Quinlan *et al.*, 2010]) are available
- `checkRestrictionEnzymeSequence`: Filtering option for valid 4C-seq reads in BAM-files
- `simulateDigestion`: Simulated digestion of a genome with two restriction enzymes, assuming full enzyme efficiency, and without consideration of fragment types or filtering options

In addition to the examples presented in this vignette, more detailed information on the functions' parameters and additional examples are presented in the corresponding manual pages.

2 Data preparation and bioinformatics software

The main input format for *Basic4Cseq*'s main functions is the binary alignment / map (BAM) format. Virtually any alignment software can be used to map the raw experimental data to the corresponding reference genome. However, BWA is suggested for both its speed and its direct control over the number of allowed mismatches. Optional filtering steps use SAMtools and BEDtools to remove invalid reads in the BAM file.

A virtual fragment library is required for the analysis of 4C-seq data. This library contains information on 4C-seq fragments and can be created for virtually any genome present as BSgenome package. Information stored consists of the position of the fragment, presence of a second restriction enzyme site, length of the fragment, length of both fragment ends, and validity of the fragment ends (i.e. a fragment end has to be unique and may not exceed a specified minimum and maximum length, if such thresholds are given). While the processing of the data takes time and hard disk space, the library can be preprocessed and applied to any experiment with the same underlying genome, restriction enzyme combination, and read length.

3 Analysis of 4C-seq data: From aligned reads to near-cis plots

Basic4Cseq offers functions for basic filtering, analysis and subsequent visualization of 4C-seq data. Fetal liver data of [Stadhouders *et al.*, 2012] is included in this package to demonstrate its functionality. Only a fraction of the original reads are included due to space limits and to speed up the analysis.

3.1 Creation of a fragment library

While the virtual fragment library needed for the analysis of Stadhouders et al's fetal liver data is already included in *Basic4Cseq*, the creation of the corresponding library is usually the first step for the analysis of 4C-seq data. We demonstrate the process on a short example string:

```
> testGenome = DNAString("ATCATGAAGTACTACATGGCACCATGT")
> fragmentData = createVirtualFragmentLibrary(chosenGenome = testGenome, firstCutter = "catg",
        secondCutter = "gtac", readLength = 2,  chromosomeName = "test", libraryName = "")
> head(fragmentData)

  chromosomeName fragmentStart fragmentEnd fragmentCentre isNonBlind fragmentLength
2           test             7          14             11       TRUE              8
  leftFragEndLength rightFragEndLength leftFragEndValid rightFragEndValid
2                 2                  2             TRUE              TRUE
```

The function splits a given genome (or chromosome) at the provided primary restriction sites. The resulting fragments are scanned for the presence of a secondary restriction site, and the ends of each fragment are checked for uniqueness and their length. The created virtual fragment library is then written to hard disk. Per default, only nonblind fragments are kept, but it is possible to include any blind fragments as well. Since blind fragments are a source of bias, the data has to be interpreted with caution [van de Werken *et al.*, 2012]. `createVirtualFragmentLibrary` can also mark fragments or fragment ends with a length above or below certain thresholds as invalid, thus providing the option to remove fragments or fragment ends which are extremely long or short.

The virtual fragment library for the fetal liver data can be created in a similar way. The time needed for the creation of the library is proportional to the length of the genome, so computing a library for longer genomes takes time.

```
> library(BSgenome.Mmusculus.UCSC.mm9)
> createVirtualFragmentLibrary(chosenGenome = Mmusculus, firstCutter = "aagctt", secondCutter =
        "catg", readLength = 54,  libraryName = "myFullFetalLiverVFL.csv")
```

For the following analysis, we therefore import both a prepared virtual fragment library and the raw read data:

```
>   library(ShortRead)
>   libraryFile <- system.file("extdata", "vfl_aagctt_catg_mm9_54_vp.csv", package="Basic4Cseq")
>   bamFile <- system.file("extdata", "fetalLiverShort.bam", package="Basic4Cseq")
>   liverReads <- readAligned(bamFile, type = "BAM")
>   liverReads
```

```
class: AlignedRead
length: 2175 reads; width: 54 cycles
chromosome: 10 10 ... 10 10
position: 20534548 20534608 ... 21912517 21961217
strand: - + ... - +
alignQuality: NumericQuality
alignData varLabels: flag
```

The virtual fragment library contains the 4C-seq fragment data for the enzyme combination HindIII ("AAGCTT") and NlaIII ("CATG"), the corresponding genome MM9 and a read length of 54 bp. Only the fragments around the experiment's viewpoint on chromosome 10 are included due to space constrains.

3.2 Initialization of a Data4Cseq object

The fragment library is central for the analysis of 4C-seq data, but some meta data is useful as well. Since we wish to mark some points of interest in the near-cis visualizations (e.g. the experiment's viewpoint), we load and add the information that is stored in a provided BED-file (with an additional column to store colour information):

```
>   pointsOfInterestFile <- system.file("extdata", "fetalLiverVP.bed", package="Basic4Cseq")
>   liverPoints<-readPointsOfInterestFile(pointsOfInterestFile)
>   liverPoints
```

```
  chr    start      end name colour
1  10 20880662 20880775   VP  black
2  10 20970000 20970000 peak    red
```

With a virtual fragment library, reads and meta data at our disposal, we can now create a Data4Cseq object.

```
> liverData = Data4Cseq(viewpointChromosome = "10", viewpointInterval = c(20879870, 20882209),
        readLength = 54, pointsOfInterest = liverPoints, rawReads = liverReads)
> liverData

4C-seq experiment data
Type: Data4Cseq
Viewpoint:  10 : 20879870 - 20882209
Read length:  54
Number of reads:  2175
Number of total fragments:  0
Number of near-cis fragments:  0
Points of interest:  2
```

The reads are then mapped to the predefined fragment library:

```
>   rawFragments(liverData)<-readsToFragments(liverData, libraryFile)
>   liverData

4C-seq experiment data
Type: Data4Cseq
Viewpoint:  10 : 20879870 - 20882209
Read length:  54
Number of reads:  2175
Number of total fragments:  453
Number of near-cis fragments:  0
Points of interest:  2
```

The function expects the provided reads and the provided fragment library to match. If just a partial fragment library is used, it is recommended to remove reads outside the chosen regions with external tools like the BEDTools program suite.

3.3 Filtering and normalization

For near-cis visualizations, all data on chromosomes that do not contain the viewpoint are irrelevant. While it is possible to visualize the whole viewpoint chromosome as "near-cis" plot, the visibility of peaks around the viewpoint is increased dramatically if the visualization region is restricted to this area. Since *Basic4Cseq* focuses on the creation of near-cis plots (in contrast to far-cis interactions), we pick a region around the experiment's viewpoint for visualization. Fragments adjacent to the viewpoint are per default removed to prevent bias through overrepresented sequences caused by self-ligation.

```
>   # pick near-cis fragments
>   nearCisFragments(liverData)<-chooseNearCisFragments(liverData,
        regionCoordinates = c(20800000, 21100000))
>   head(nearCisFragments(liverData))

    chromosomeName fragmentStart fragmentEnd fragmentCentre isNonBlind fragmentLength
104             10      20803166    20804714       20803924       TRUE           1548
105             10      20804721    20808310       20806515       TRUE           3589
106             10      20808317    20808848       20808517       TRUE            531
107             10      20808855    20810907       20809888       TRUE           2052
108             10      20811250    20814707       20813114       TRUE           3457
109             10      20814714    20814861       20814774       TRUE            147
    leftFragEndLength rightFragEndLength leftFragEndValid rightFragEndValid
```

```
104               229                260             TRUE             TRUE
105                19                 19            FALSE            FALSE
106               102                232             TRUE             TRUE
107               136                121             TRUE             TRUE
108               547                276             TRUE             TRUE
109                59                 85             TRUE             TRUE
    leftFragEndReads rightFragEndReads fragEndReadsAverage
104                2                 0                   1
105                0                 0                   0
106                1                 0                   0
107                5                 0                   2
108                0                 0                   0
109                1                10                   6
```

The raw read count can then be RPM-normalized to allow for better comparability between different plots.

```
>   # normalization of near-cis data
>   library("caTools")
>   nearCisFragments(liverData)<-normalizeFragmentData(liverData)
>   head(nearCisFragments(liverData))

    chromosomeName fragmentStart fragmentEnd fragmentCentre isNonBlind fragmentLength
104             10      20803166    20804714       20803924       TRUE           1548
105             10      20804721    20808310       20806515       TRUE           3589
106             10      20808317    20808848       20808517       TRUE            531
107             10      20808855    20810907       20809888       TRUE           2052
108             10      20811250    20814707       20813114       TRUE           3457
109             10      20814714    20814861       20814774       TRUE            147
    leftFragEndLength rightFragEndLength leftFragEndValid rightFragEndValid
104               229                260             TRUE             TRUE
105                19                 19            FALSE            FALSE
106               102                232             TRUE             TRUE
107               136                121             TRUE             TRUE
108               547                276             TRUE             TRUE
109                59                 85             TRUE             TRUE
    leftFragEndReads rightFragEndReads fragEndReadsAverage
104         919.5402             0.000            459.7701
105           0.0000             0.000              0.0000
106         459.7701             0.000              0.0000
107        2298.8506             0.000            919.5402
108           0.0000             0.000              0.0000
109         459.7701          4597.701           2758.6207
```

3.4 Quality control

The distribution of reads throughout the genome provides information on the quality of the 4C-seq experiment data. *Basic4Cseq* provides the following quality statistics: the number of total reads, cis to overall ratio of reads, and the percentage of covered fragment ends within a certain distance around the experiment's viewpoint. Reference values for high-quality experiments, as provided by van de Werken et al [van de Werken *et al.*, 2012], are more than one million reads total, a cis to overall ratio of more than 40% and a large fraction of covered fragment ends in the viewpoint's vicinity. The region around

the viewpoint which is to be checked can be customized; van de Werken et al recommend 2 Mb total for 6 bp cutters and 0.2 Mb total for 4 bp cutters. Since this value can be considered to be quite small, an alternative recommendation would be to use at least a window of 0.5 Mb in total for 4 bp cutters. The parameter "distanceFromVP" allows the use of custom regions.

```
>   getReadDistribution(liverData, useFragEnds = TRUE, outputName = "")

[1] "total reads: 2175"
[1] "reads on the viewpoint chromosome: 2175 (100% of total reads)"
[1] "reads in the viewpoint region: 1602 (73.66% of total reads)"
[1] "covered fragment ends in the viewpoint region: 62.22%"
```

Since the example data is is taken from the experiment's viewpoint chromosome, the cis to overall ratio is unrealistic. A relatively low number of fragment ends in the viewpoint's vicinity is covered for the example data; this number rises significantly if the complete data set of Stadhouders et al is used.

The statistics can be exported as a standard text file, or printed on screen if no output file name is provided.

3.5 Near-cis visualization

Basic4Cseq offers two visualization routines for near-cis interactions, a coverage plot and an additional multi-scale contact intensity profile in a domainogram- or heatmap-like format. Both plots can be exported as PDF or TIFF file, the format is automatically chosen according to the file name. If no file name is provided, the plot is printed on screen. Pre-defined regions of interest can be marked in both plots. Fragment end data can be used directly, or the values can be interpolated on fragment level.

The visualization strategy for the coverage plot is similar to van de Werken et al [van de Werken *et al.*, 2012]: Fragment data is smoothed via a running median or running mean approach with a specified window size. Quantiles (per default 20%, 50% and 80%) are further smoothed and interpolated with R's loess function. The interpolation effect offered by the loess function helps to make the data profile visible.

The viewpoint region itself can be excluded from the visualization to stop overrepresented sequences (most likely caused by self-ligation) from distorting the picture. `visualizeViewpoint` expects a *Data4Cseq* instance to visualize, but an alternative input of a custom data frame ("chrom", "start", "end", "reads") that contains both position and (normalized) read count of the fragment data to visualize is possible as well. The results of the example are shown in figure 1.

```
>   # This command creates a near-cis plot of the fetal liver's viewpoint data
>   visualizeViewpoint(liverData, plotFileName = "", mainColour = "blue",
        plotTitle = "Fetal Liver Near-Cis Plot", loessSpan = 0.1, maxY = 6000,
        xAxisIntervalLength = 50000, yAxisIntervalLength = 1000)
```

For the heatmap-like intensity profile, read counts per fragment are normalized to [0, 1] and the resulting intensity values are expressed on a log2 scale to allow for better visibility of distant interactions. R's heat colours are chosen for the intensity visualization; discarded data (either repetitive fragment ends or removed blind fragments, if this option is taken) is represented in black to allow for easy visibility of regions with few valid data points. The results of the example are shown in figure 2.

```
>   # This command creates a near-cis heatmap plot of the fetal liver data
>   drawHeatmap(liverData, plotFileName = "", xAxisIntervalLength = 50000, bands = 5)
```

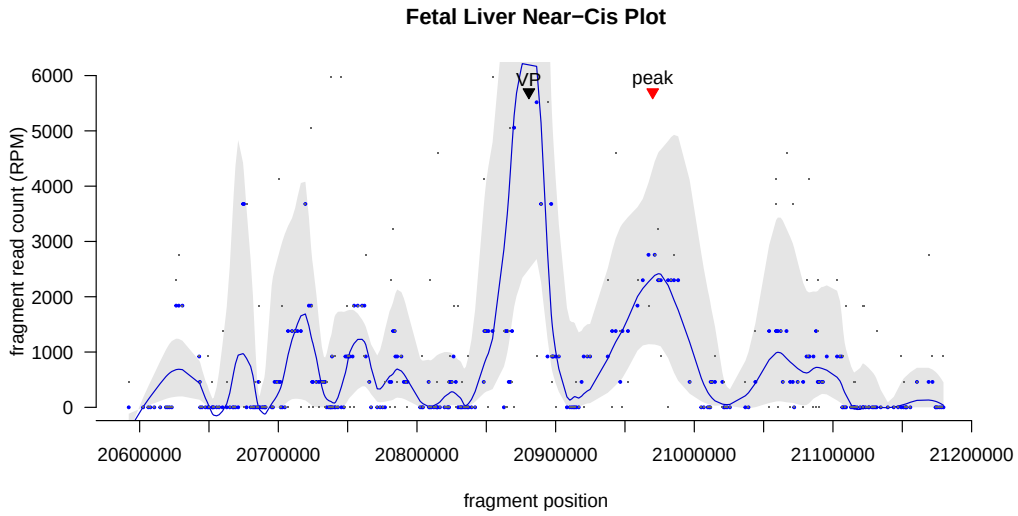

Figure 1: Near-cis coverage plot of fetal liver data taken from Stadhouders et al [Stadhouders *et al.*, 2012].

4 Far-cis and trans interactions

For regions located either more remote from the viewpoint or on other chromosomes, it is advisable to use a statistical enrichment approach for the analysis of the 4C-seq fragment data, e.g. [Splinter *et al.*, 2012]. *Basic4Cseq* can export fragment-based 4C-seq data as WIG-files for use with external algorithms. Per default, no header is added to the wig data. While this is sufficient for visualizations with some tools and convenient for further downstream analysis, other tools (e.g. the UCSC Genome Browser) require a header line. `printWigFile` allows input of custom header lines through the parameter `headerUCSC`.

```
>   printWigFile(liverData, wigFileName = "fetalLiver.wig")
```

Splinter et al's spider plots already offer a way to create far-cis visualizations. For trans interactions, *Basic4Cseq* provides a wrapper for *RCircos* to produce basic Circos-plots of inter-chromosomal interactions.

```
>   library("RCircos")
>   transInteractions <- system.file("extdata", "transInteractionData.txt", package="Basic4Cseq")
>   ideogramData <- system.file("extdata", "RCircos_GRCm38_ideogram.csv", package="Basic4Cseq")
>   plotTransInteractions(transInteractions, "10", c(20000100, 20001000), ideogramData,
        PlotColor = "blue", expandBands = TRUE, expansionValue = 1000000, plotFileName = "")
```

5 Optional functionality

5.1 Controlling the restriction enzyme sequence of reads in SAM files

Basic4Cseq offers further filter functions for a SAM file with reads where the first restriction enzyme sequence has not been trimmed. The first bases of a read (i.e. the position of the first restriction enzyme sequence in a standard 4C-seq experiment; normally 4 or 6 bp long) are checked for mismatches. Reads with mismatches in the restriction enzyme sequence are deleted to remove reads that did not match

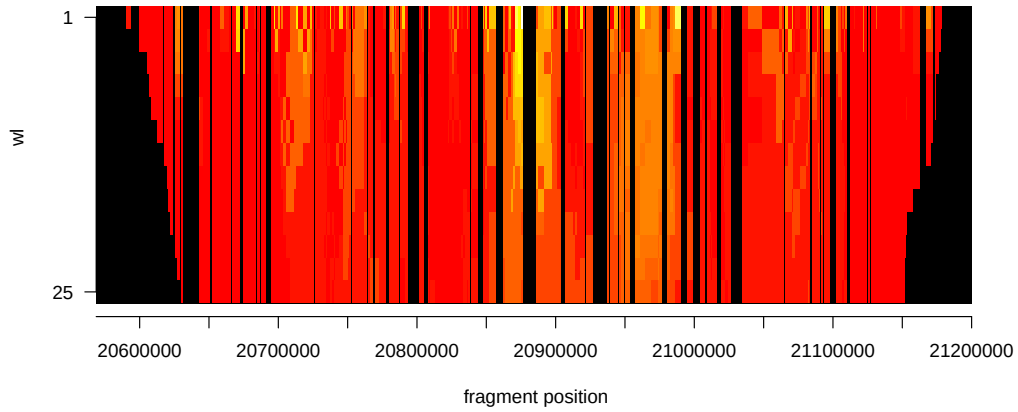

Figure 2: Near-cis multi-scale contact intensity profile of fetal liver data taken from Stadhouders et al [Stadhouders *et al.*, 2012]. Export as pdf and tiff is supported; the dimensions of the plot can be customized.

perfectly to fragment ends, but overlap and distort the true signal. SAM and BAM files can easily be converted with the help of SAMtools.

```
>   # The demonstration reads are taken from Stadhouders et al's data,
>   # but additional cutter sequences have been added manually for demonstration purposes
>   fetalLiverCutterFile <- system.file("extdata", "fetalLiverCutter.sam", package="Basic4Cseq")
>   checkRestrictionEnzymeSequence("aagctt", fetalLiverCutterFile, "fetalLiverCutter_filtered.sam")
```

5.2 Extraction of BED files

BED files can be extracted from the fragment library for direct use with tools like the Integrative Genomics Viewer (IGV).

```
>   printBEDFragmentLibrary(libraryFile, "BEDLibrary_FL_vp.bed")
```

5.3 Restriction enzyme database

For convenience, a small database of restriction enzyme names and sequences has been added. The data was taken from van de Werken et al's database of restriction enzyme combinations [van de Werken *et al.*, 2012]. The function giveEnzymeSequence loads this database (a tab-separated file that can be expanded or replaced as necessary) and returns the corresponding enzyme sequence for a given enzyme name.

```
>   enzymeData <- system.file("extdata", "enzymeData.csv", package="Basic4Cseq")
>   giveEnzymeSequence(enzymeData, "NlaIII")
```

```
[1] "CATG"
```

5.4 Digestion simulation

Basic4Cseq can simulate the digestion process with two restriction enzymes for a dna sequence or a BSgenome package and simply count the resulting fragments. The resulting virtual library of fragment parts does not provide information on blind or non-blind fragments, but provides information on the fragment length distribution of the real (i.e. biological) 4C-seq library. In contrast to the regular virtual fragment library for 4C-seq data, fragments between two adjacent secondary restriction enzyme sites are counted as well. This information can then be used for quality controls of the biological fragment library.

The results of the example are shown in figure 3.

```
>   shortTestGenome = "ATCCATGTAGGCTAAGTACACATGTTAAGGTACAGTACAATTGCACGATCAT"
>   fragments = simulateDigestion("catg", "gtac", shortTestGenome)
>   head(fragments)

  length frequency
1      1         2
2      3         1
3      5         1
4      8         1
5     14         1

>   # This command creates a histogram plot of virtual library fragment length and frequencies
>   drawDigestionFragmentHistogram(fragments)
```

5.5 BAM file filtering

Basic4Cseq offers a wrapper function for the alignment of FASTQ files and filtering of the resulting SAM / BAM files. `prepare4CseqData` relies on the availability of the external tools BWA, SAMtools, and BEDtools. A provided 4C-seq fastq file is read from hard disk, and the reads are aligned with BWA. The function `checkRestrictionEnzymeSequence` is used for optional filtering. Samtools and bedtools provide the necessary functionality for intersecting the filtered reads with a given 4C-seq fragment library for visualization purposes (e.g. with the Integrative Genomics Viewer, IGV).

```
>     # BWA, samtools and bedtools must be installed
>     # It is assumed that the example data files (from the package) are in the active directory
>     prepare4CseqData("veryShortExample.fastq", "CATG", "veryShortLib.csv",
         referenceGenome = "veryShortReference.fasta")
```

References

[Stadhouders *et al.*, 2012] Stadhouders, R., Thongjuea, S., *et al* (2012) Dynamic long-range chromatin interactions control Myb proto-oncogene transcription during erythroid development., *EMBO*, **31**, 986-999.

[van de Werken *et al.*, 2012] van de Werken, H., Landan, G., Holwerda, S., et al. (2012) Robust 4C-seq data analysis to screen for regulatory DNA interactions, *Nature Methods*, **9**, 969-971.

[Gheldof *et al.*, 2012] Gheldof, N., Leleu, M., Noordermeer, D., et al. (2012) Detecting Long-Range Chromatin Interactions Using the Chromosome Conformation Capture Sequencing (4C-seq) Method, *Methods in Molecular Biology*, **786**, 212-225.

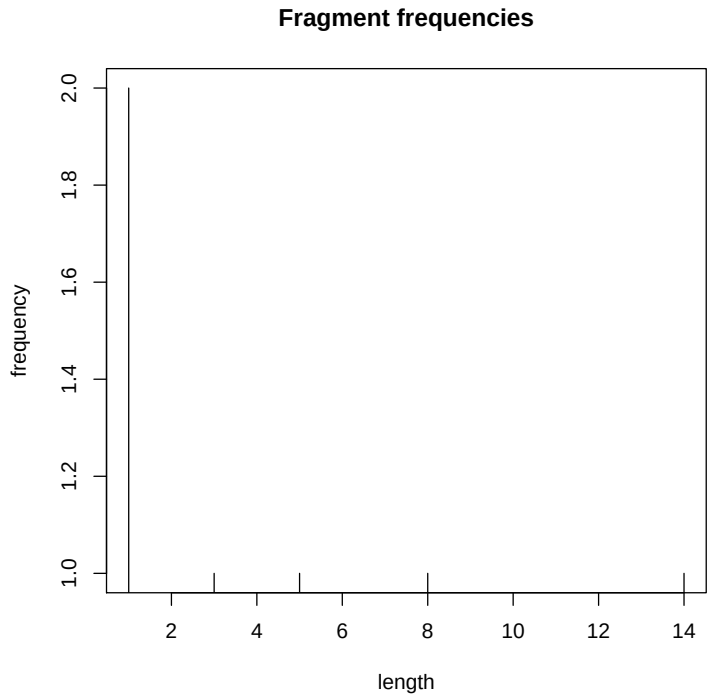

Figure 3: Histogram plot of virtual library fragment length and frequencies. The underlying genome is a short example string.

[van de Werken *et al.*, 2012] van de Werken, H., de Vree, P., Splinter, E., et al. (2012) 4C technology: protocols and data analysis, *Methods Enzymology*, **513**, 89-112.

[Splinter *et al.*, 2012] Splinter, E., de Wit, E., van de Werken, H., et al. (2012) Determining long-range chromatin interactions for selected genomic sites using 4C-seq technology: From fixation to computation, *Methods*, **58**, 221-230.

[Li *et al.*, 2009] Li, H. and Durbin, R. (2009) Fast and accurate short read alignment with Burrows-Wheeler Transform, *Bioinformatics*, **25**, 1754-60.

[Li *et al.*, 2009] Li, H., Handssaker, B, Wysoker, A. et al. (2009) The Sequence alignment/map (SAM) format and SAMtools, *Bioinformatics*, **25**, 2078-9.

[Quinlan *et al.*, 2010] Quinlan, A. and Hall, I. (2010) BEDTools: a flexible suite of utilities for comparing genomic features, *Bioinformatics*, **26**, 841-2.

6 Session Information

```
R version 3.1.0 (2014-04-10)
Platform: x86_64-unknown-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=C               LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
```

```
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
[10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

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

other attached packages:
 [1] Basic4Cseq_1.0.2        caTools_1.17            ShortRead_1.22.0
 [4] GenomicAlignments_1.0.1 BSgenome_1.32.0         Rsamtools_1.16.1
 [7] GenomicRanges_1.16.3    GenomeInfoDb_1.0.2      BiocParallel_0.6.1
[10] Biostrings_2.32.0       XVector_0.4.0           IRanges_1.22.9
[13] BiocGenerics_0.10.0

loaded via a namespace (and not attached):
 [1] BBmisc_1.7                           BSgenome.Ecoli.NCBI.20080805_1.3.1000
 [3] BatchJobs_1.2                        Biobase_2.24.0
 [5] DBI_0.2-7                            RCircos_1.1.2
 [7] RColorBrewer_1.0-5                   RSQLite_0.11.4
 [9] Rcpp_0.11.2                          bitops_1.0-6
[11] brew_1.0-6                           checkmate_1.1
[13] codetools_0.2-8                      digest_0.6.4
[15] fail_1.2                             foreach_1.4.2
[17] grid_3.1.0                           hwriter_1.3
[19] iterators_1.0.7                      lattice_0.20-29
[21] latticeExtra_0.6-26                  plyr_1.8.1
[23] sendmailR_1.1-2                      stats4_3.1.0
[25] stringr_0.6.2                        tools_3.1.0
[27] zlibbioc_1.10.0
```