Using seqTools package

Wolfgang Kaisers, CBiBs HHU Dusseldorf

June 30, 2024

1 `seqTools` package

The `seqTools` package provides functionality for collecting and analyzing quality measures from FASTQ files.

1.1 FASTQ file format

FASTQ is the *de facto* standard file format for DNA second generation sequencers. A description of the format has been published in 2009 [1]. Fastq are is a plain text format. The content is structured in reads. Each read consists of four elements:

Nr	Segment	First character
1	Header	'@'
2	Sequence	
3	'+'	'+'
4	Quality	

A FASTQ read might look like the following:

```
@identifier
TATCGATCAAGGTTCAAAGCAGTATCGATTATCGATCAACATTTGTTCATATCGATCAAG
+
<.>x.x:;:;:+.+.+.:;:;.+.+.+.:;:;:;*:*:*:;:;:+-+-+-.,.,.,.,.
```

There is no explicit limitation for line length. Also, multiple sequence and quality segments may be present.

Considerations for parsing. As the header signifying '@' (ASCII 64) characters may also be present in of quality segments and therefore even occur as first character in quality lines, the '@' may not be used as read delimiter. Therefore an essential convention is that sequence and quality strings have equal length. So, the begin of the following read is not indicated by the '@' but (implicitly) by the position of the '+'. The '+' segment may contain a copy of the text in the header segment.

Sequence Segment The sequence segment contains the essential read information. Normally it is a sequence which contains the characters {A,C,G,T,N}, but some other letters are also allowed [2].

Quality Segment The quality segment contains a string which has the same length as the sequence string. For each character in the sequence segment, a quality value is given in the quality segment. The quality values are encoded in the ASCII values of the contained characters. The encoding values can be looked-up with the `phredTable` function:

```
> library(seqTools)
> head(phredTable())
```

```
  ascii phred char
1    33     0    !
2    34     1    "
3    35     2    #
4    36     3    $
5    37     4    %
6    38     5    &
```

ASCII and Phred values are related by an offset of 33

$$ASCII = Phred + 33 \tag{1}$$

$$\tag{2}$$

because the ASCII values 0-32 encode non-printable characters. The maximum ASCII for printable charachters is 126, so the range for Phred values is 0 to 93. Phred qualities encode the estimated probability (p) that the corresponding nucleotide in the sequence is correct by

$$Phred = -10\ log_{10}(p). \tag{3}$$

$$\tag{4}$$

Quality measures on FASTQ files Standard quality measures span descriptive (position wise) statistics on counted on sequence and quality data. The `seqTools` package provides functions for analysis Nucleotide (single Nucleotides, N's, GC and any other combination), DNA k-mer and Phred abundances.

2 Analysis of FASTQ files with `seqTools`

The processing of FASTQ files is divided in two steps: Data collection and data analysis. The data collection step is intended to run unsupervised without user interaction because there may be large data volumes to be analyzed (e.g. several hundred GByte compressed files). The data collection is intended to be done batchwise (e.g. one batch for one flowcell). Each batch is collected by one call of `fastqq` which produces one `fastqq` object. For later analysis, these objects can be merged together.

In the second part, directed analysis can be done by merging objects, data extracting, creation of summarizing tables or plotting figures.

3 Collecting data from FASTQ files

The basic function for collecting data from FASTQ files is the `fastqq` function. In the following example we construct the `fastqq` object which will be used later on:

```
> basedir <- system.file("extdata", package="seqTools")
> filenames <- file.path(basedir, c("g4_l101_n100.fq.gz", "g5_l101_n100.fq.gz"))
> fq <- fastqq(filenames, k=6, probeLabel=c("g4", "g5"))
```

The function reads compressed and uncompressed FASTQ files. `fastqq` takes a vector of FASTQ file names, k (the length of the DNA k-mers) and `probeLabel`'s. Sensible values for k are 6 or 9,
The first argument is a character vector with FASTQ-file names. The second argument `k` denotes the length of the DNA k-mers that are to be counted. The third arguments gives the labels for the collected probes which are later on used in tables and figures (so they should be handy). When filenames and probeLabel have different length, the given `probeLabel`'s are discarded.

In our experience, `fastqq` reads at a rate of ~220.000 reads per second on a desktop computer (for k=9). The memory consumption is proportional to 4^k, because the k-mers are counted in static arrays. After counting, the returned objects should be stored on the hard disc (using `save`) for later analysis.

When the FASTQ files from a whole Illumina flowcell are located in a separate directory, the `fastqq` call could be:

```
> filenames <- dir(path="fastqDir", pattern="*.fastq.gz")
> fq <- fastqq(file.path("fastqDir", filenames), k=6)
```

Printing a fastqq object For objects of class `fastqq` there exists a specialized `show` function which displays some summarizing information from the given object:

```
> fq
```

```
Class       :               Fastqq
nFiles      :                    2
maxSeqLen   :                  101
k (Kmer len):                    6

nReads      :                  200
nr  N   nuc :                    2
Min seq len :                  101
Max seq len :                  101
```

The `fastqq` object contains data from two FASTQ files with a k-mer length of 6. The two files altogether contain 200 reads and 2 'N' characters are counted (a low number relates to high quality). All counted reads contain 101 nucleotide entries.

3.1 Structure of fastqq objects

The function `fastqq` returns an `fastqq` object. `fastqq` objects are S4 types which contain 8 tables:

Name	nrows	ncols	Content
nFiles	1	1	Number of FASTQ files
Kmer	4^k	nFiles	k-mer counts
seqLenCount	maxSeqLen	nFiles	Sequence length counts
seqLen	2	nFiles	Minimal and maximal sequence length
nReads	1	nFiles	Number of reads per file
nN	1	nFiles	Number of N Nucleotides per file
gcContent	101	nFiles	GC content per Read (%) counts
nac	nIupac	maxSeqLen	Position wise count of nucleotides
phred	94	maxSeqLen	Positoin wise phred counts

Table 1: `fastqq` tables. nFiles=Number of FASTQ files, nIupac=Number of IUPAC characters (=19). maxSeqLen=Maximal read length in any counted FASTQ file.

4 Analysis of collected data

Standard accessor functions There is a set of accessor functions which return a subset of the contained data:

Name	Return type	Value
filenames	character	Fastq file names
collectTime	list	Start and end time of data collection
collectDur	numeric	Collection time (sec)
getK	numeric	Length of DNA k-mers
nFiles	numeric	Number of FASTQ files
nNnucs	numeric	Number of 'N's per FASTQ file
nReads	numeric	Number of reads per FASTQ file
maxSeqLen	numeric	Maximal read length in all files
seqLenCount	matrix	Sequence length counts per file
nucFreq	matrix	Position wise nucleotide count
gcContent	numeric	GC-content distribution for file
gcContentMatrix	matrix	GC content of all files
seqLen	matrix	Sequence length counts
kmerCount	matrix	File wise k-mer counts
phred	matrix	Position wise count values for phred
phredQuantiles	matrix	Position wise phred value for quantiles
mergedPhred	matrix	Position wise phred value for all files

Table 2: `fastqq` accessor functions. nFiles=Number of FASTQ files, nIupac=Number of IUPAC characters (=19). maxSeqLen=Maximal read length in any counted FASTQ file.

5 Merging and melting `fastqq` objects

Melting Melting a `fastqq` object is done to produce a result with lower k value. As explanatory example we melt down to k=2 and print out the resulting k-mer counts:

```
> fqm <- meltDownK(fq, newK=2)
> kmerCount(fqm)[, 1]
```

```
 AA  AC  AG  AT  CA  CC  CG  CT  GA  GC  GG  GT  TA  TC  TG
725 363 833 532 628 664 133 660 699 534 733 484 408 508 743
 TT
953
```

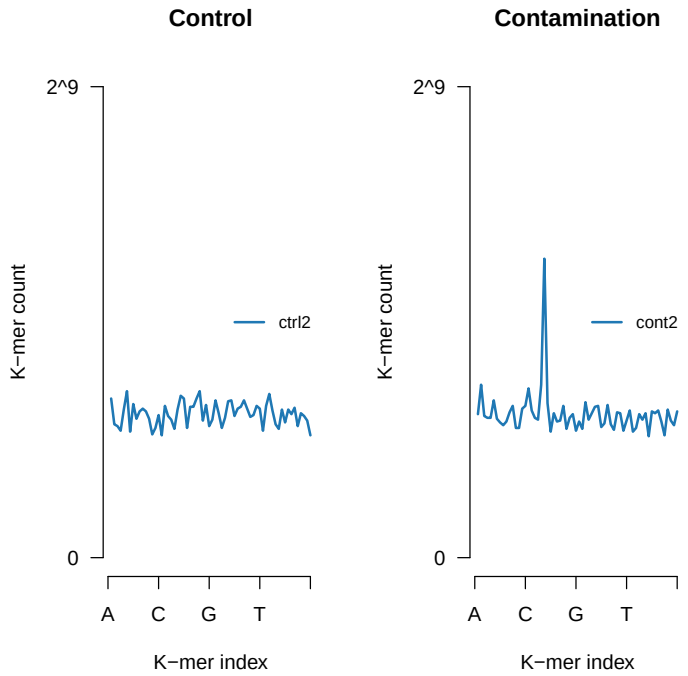
The melt function is internally used in the `mergeFastqq` function.

Merging Two `fastqq` objects can be merged together. In essence, the merging procedure creates an object which has the same structure as if the `fastqq` function would have been called on all given FASTQ file-names together. When both `fastqq` objects have been created with different k values, the count values from the larger k are melted down (by summing up the appropriate k-mer counts) so that the result contains count values for the smaller k-value.

```
> files1 <- file.path(basedir, c("sfq1_ctrl.fq.gz", "sfq2_ctrl.fq.gz"))
> files2 <- file.path(basedir, c("sfq1_cont.fq.gz", "sfq2_cont.fq.gz"))
> fq1 <- fastqq(files1, k=3, probeLabel=c("ctrl1","ctrl2"))
> fq2 <- fastqq(files2 , k=3, probeLabel=c("cont1", "cont2"))
```

All four files contain reads which are created by simulation (`sim_fq`). The 'cont' files are additinally contaminated with deterministic reads which produces a spike in the k-mer count plots.

```
> op <- par(mfrow = c(1, 2))
> plotKmerCount(fq1, 2, mxey=9, main="Control")
> plotKmerCount(fq2, 2, mxey=9, main="Contamination")
> par(op)
```



These two files are now merged together

```
> mrg <- mergeFastqq(fq1, fq2)
> mrg
```

```
Class       :              Fastqq
nFiles      :                   4
maxSeqLen   :                 102
k (Kmer len):                   3

nReads      :                 400
nr  N   nuc :                   0
Min seq len :                 102
Max seq len :                 102
```

Designated usage of the mergeFastqq function. The `mergeFastqq` function is intended to be used for the preparation for hierarchical clustering based on DNA k-mer counts. The FASTQ data can be collected separately from e.g. different sequencing runs which are then merged together to one object. From there, a distance matrix can be retrieved which serves as input fo r hierarchical clustering algorithms.

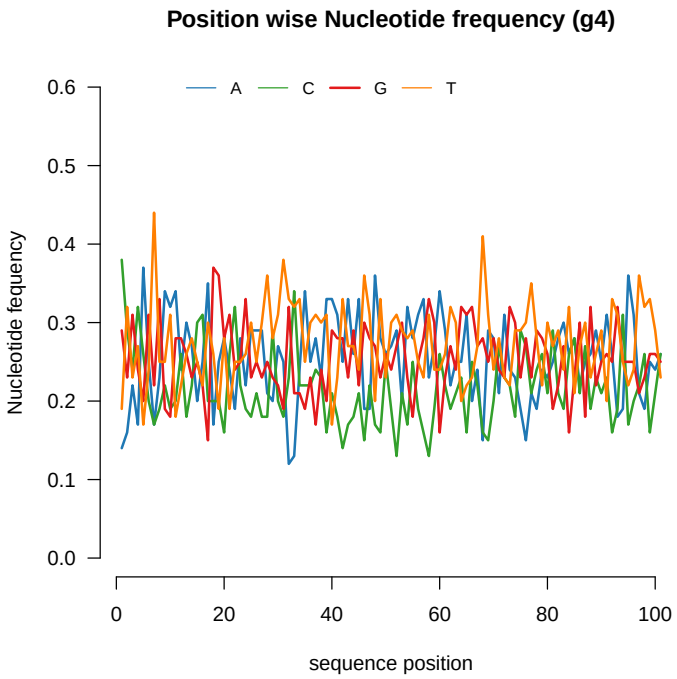
6 Plots for count values on single FASTQ files

These plot comprise functions which produce summarizing figures on nucleotide and phred score statistics.

6.1 Plots for nucleotide counts

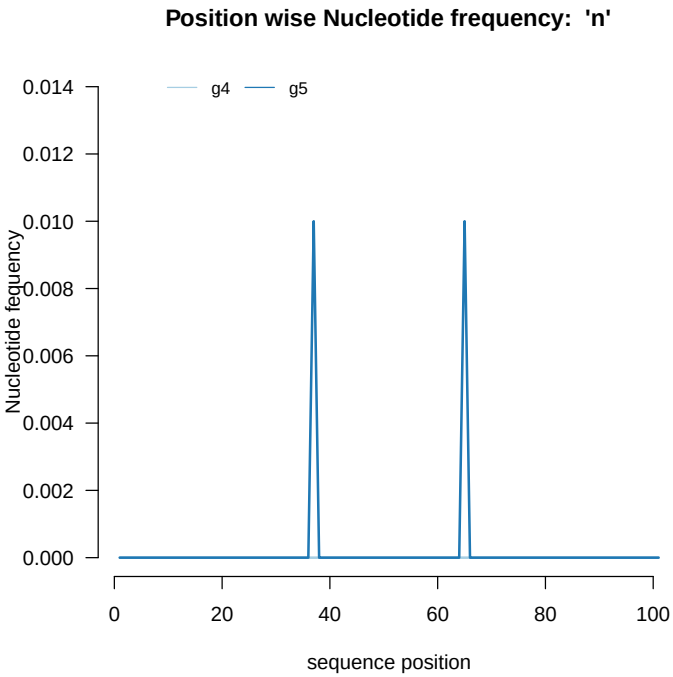
Position wise counts for Nucleotide frequencies The `plotNucFreq` function produces a plot of the position wise nucleotide frequencies from one FASTQ file. The file is depicted by the second argument (given as index):

```
> plotNucFreq(fq,1)
```



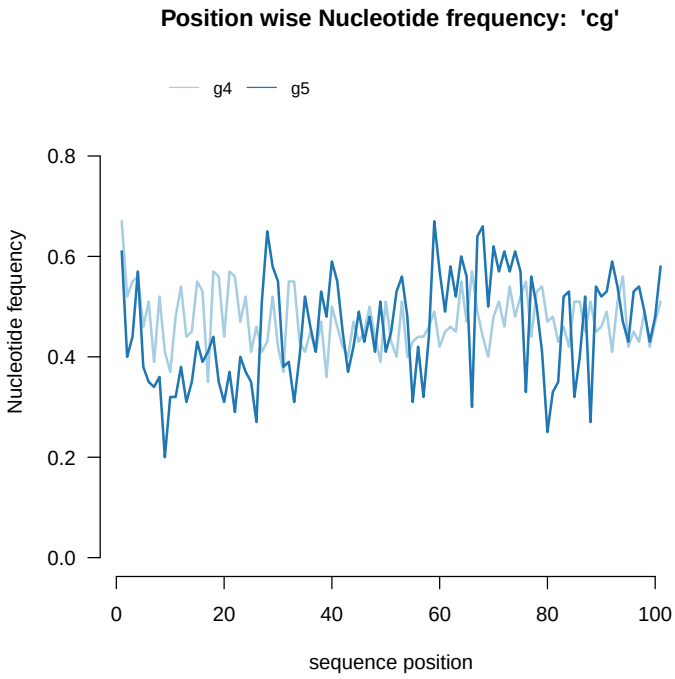
The `plotNucCount` function allows for plotting the position wise percentual nucleotide content for any combination of allowed IUPAC characters (given as `nucs` argument). There will be a plot generated which covers data from all FASTQ files. The standard value for `nucs` is 16 from which Count values for 'N' are displayed:

```
> plotNucCount(fq)
```



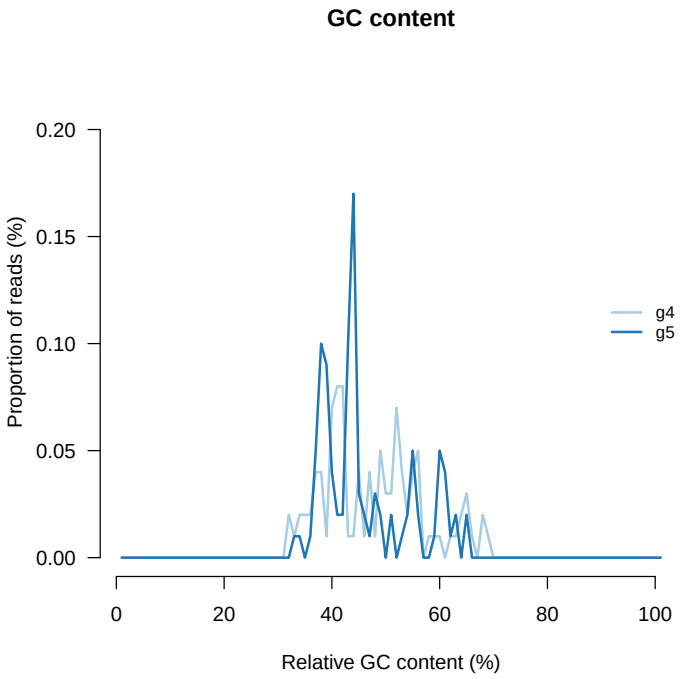
For position wise plot of GC content the index values for G and C can be combined:

```
> plotNucCount(fq, c(2, 3))
```



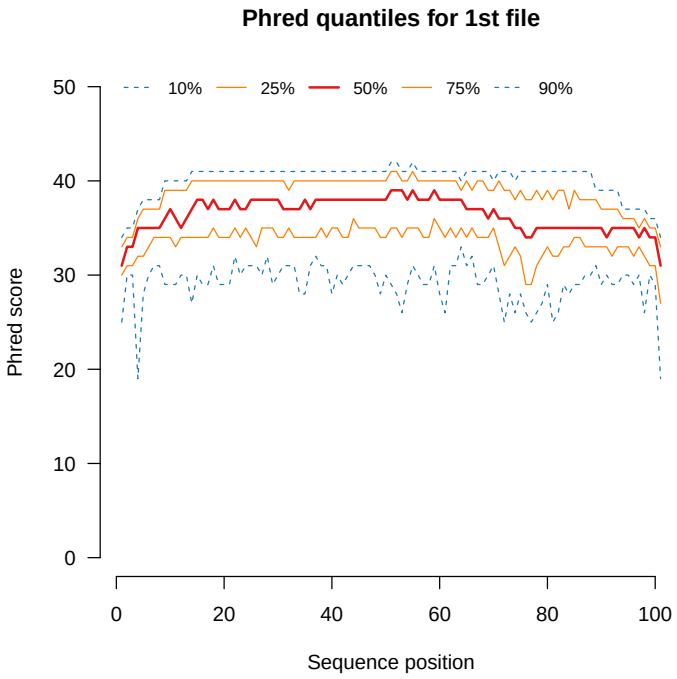
Distribution of GC content on reads The `plotGCcontent` function produces one plot for the entire fastqq object. For each contained file, the distribution of the percentual GC content is given. The percentual representation ensures that the area under all shown lines sums up to 1:

```
> plotGCcontent(fq)
```



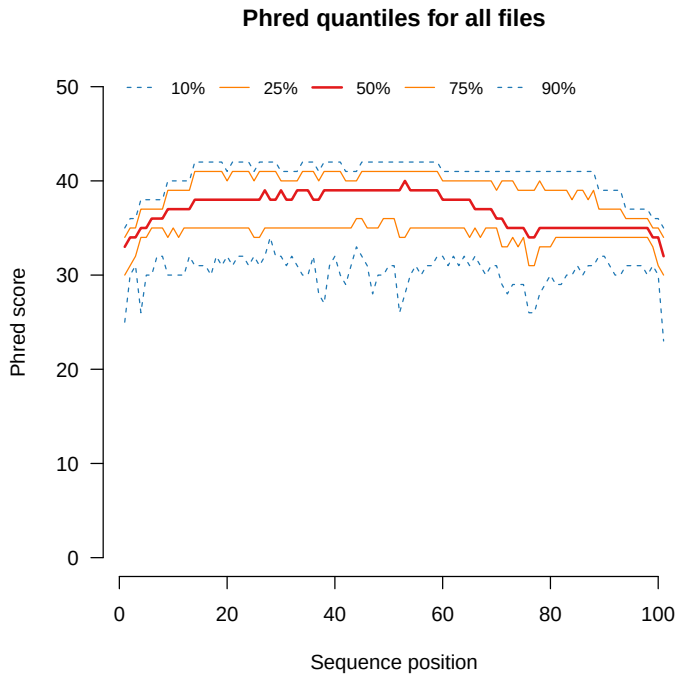
Distribution of phred values The `plotPhredQuant` function plots the position wise phred quantiles for the quantiles 10%, 25%, 50%, 75% and 90% for one FASTQ file. The second argument 'i' indicates the index of the FASTQ file for which the values are plotted:

```
> plotPhredQuant(fq, 1, "Phred quantiles for 1st file")
```



The quantile lines provide a boxplot-like information. When for example the lower dashed blue line (the 10 % quantile) is positioned at the value 20 for sequence position 5, then for read position 5, 10% of the nucleotides were scored with 20 or lower.

```
> plotMergedPhredQuant(fq, main = "Phred quantiles for all files")
```



From the figure, we know that roughly 80 % of the phred scores fall into the range between 30 and 40 and that median values (red lined) vary in the same range. The nucleotide qualites are best in the middle region of the reads and decrease towards the read ends.

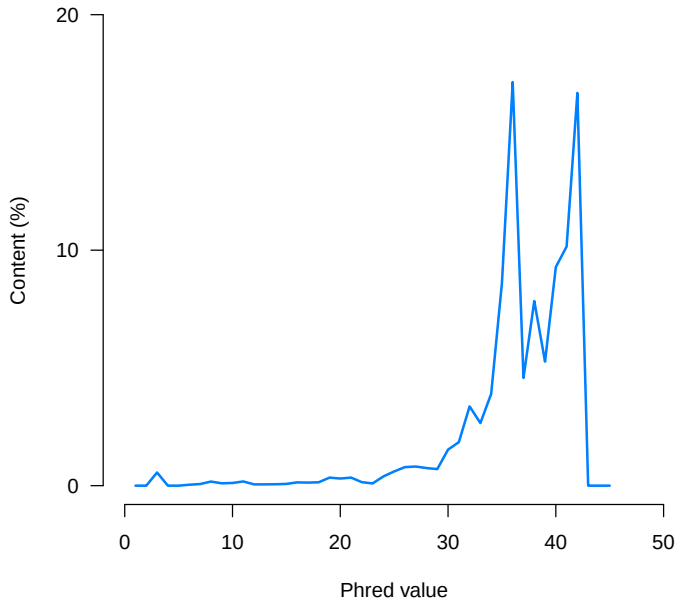
Returning global Phred counts The `phredDist` function returns a named vector with relative content of Phred values from a subset of FASTQ-files in a `fastqq` object. When no subset index is given, the Phred values are counted for the whole object.

```
> phred<-phredDist(fq, 1)
> phred<-phredDist(fq)
> head(phred)
```

```
           0            1            2            3
0.0000000000 0.0000000000 0.0055940594 0.0000000000
           4            5
0.0000000000 0.0003960396
```

Plotting global Phred counts The values returned by `phredDist` can be directly plotted. The `plotPhredDist` function produced a plot of global Phred value counts from a whole `fastqq` object or a subset (given by the index i). The plot can be used to assess the relative content of low and high quality Phred values. When for example the `fastqq` object contains data from one flowcell and the plot reveals a high proportion of low quality Nucleotides (e.g. $>$ 20 % lower than 10), then the data quality from the whole flowcell may be questionable.

```
> plotPhredDist(fq)
```



7 Analysis of DNA k-mer profile on FASTQ files

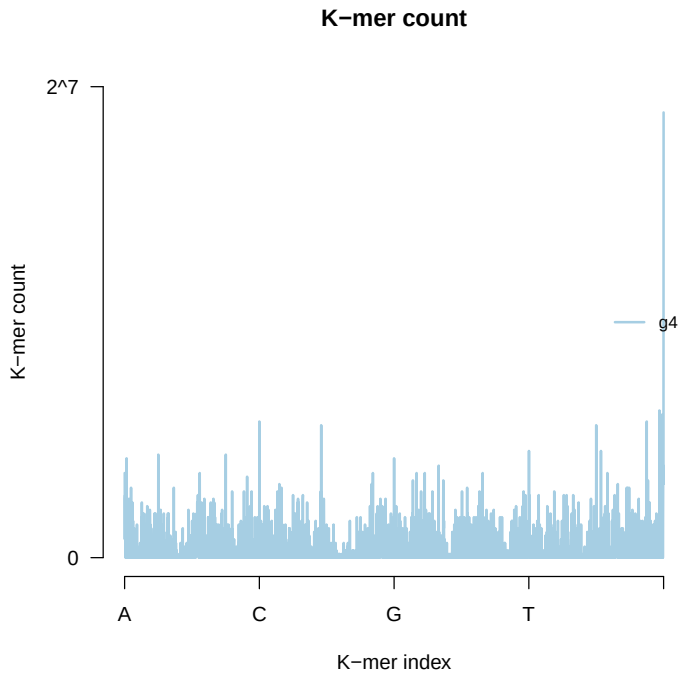
The function `fastqq` counts the occurence of all DNA k-mers within FASTQ files. For a given k, there exist 4^k DNA k-mers. Internally, the DNA k-mers are converted into a (k-dependent) index which is used for counting. There are functions which convert k-mer indexes into k-mers and vice versa:

```
> kMerIndex(c("CCC", "GGG"))
```

```
[1] 22 43
```

Count for DNA k-mers The function `plotKmerCount` plots count values for contained DNA k-mers whithin one FASTQ file. The plotted profiles provide information about the regularity of the count values. Possibly strong over represented k-mers can be seen here.

```
> plotKmerCount(fq, 1)
```



7.1 Distance measures based on DNA k-mer counts

From the DNA k-mer counts, for every FASTQ file there is a vector of length 4^k with non-negative integer values. These vectors can be used to define distances between two files using standard distance measured. Implemented in this package is the Canberra distance. Let $x, y \in \mathbb{R}^n$, so $x = \{x_1, \ldots, x_n\}$ and $y = \{y_1, \ldots, y_n\}$. The Canberra distance between x_i and y_i is defined as

$$d_c(x, y) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|}$$

In essence, the Canberra distance relies on the absolute difference normed by the mean value of the compared values. The total read numbers are scaled to a common value (the maximal read number in all lanes) in order to compensate a systematic offset from align depth.

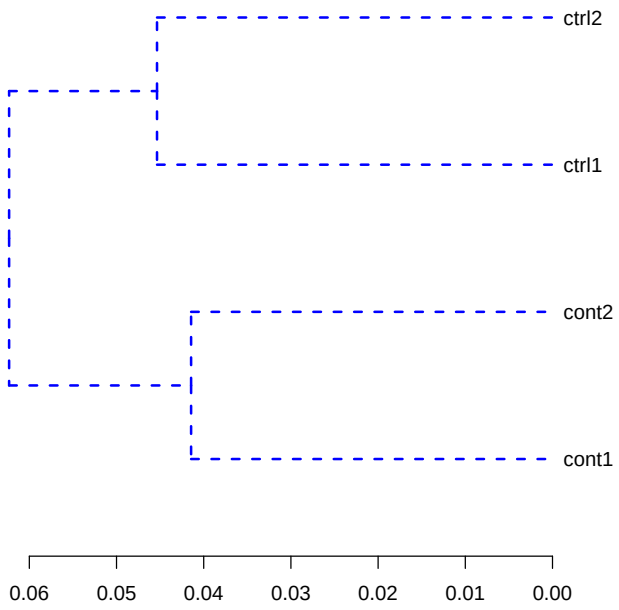
Calculation of Canberra distances on `fastqq` objects The `cbDistMatrix` calculates Canberra distance matrices on FASTQ files in a `fastqq`:

```
> mtx <- cbDistMatrix(mrg)
> mtx
```

```
           ctrl1      ctrl2      cont1 cont2
ctrl1 0.00000000 0.00000000 0.00000000     0
ctrl2 0.04535505 0.00000000 0.00000000     0
cont1 0.05353266 0.06231715 0.00000000     0
cont2 0.05564388 0.05069881 0.04144771     0
```

Hierarchical clustering (HC) on fastqq The distance matrix then can be used for HC. The figure produced by the following code shows that HC discriminates the control from the contaminant group.

```
> hc <- hclust(as.dist(mtx))
> hcd <- as.dendrogram(hc, lty=2, lwd=2)
> op <- par(mar = c(3, 1, 1, 5))
> plot(hcd, horiz=TRUE, las=1, edgePar=list(lwd=2, lty=2, col="blue"))
> par(op)
```



8 Miscellaneous

8.1 ASCII related functions

There are some functions which perform conversions between characters and ASCII values which are simple wrappers around R-functions:

```
> char2ascii("a")

[1] 97

> ascii2char(97:99)

[1] "abc"
```

The `phredTable` will return a table with phred, ASCII and characters:

```
> phredTable()
> phredTable(20:30)
```

References

[1] P.J.A. Cock, C.J. Fields, N. Goto, M.L. Heuer, and Rice P.M. The sanger fastq file format for sequences with quality scores and the solexa/illumina fastq variants. *Nucleic Acids Research*, 38:1767–1771, 2010. `http://dx.doi.org/10.1093/nar/gkp1137`.

[2] NCBI. Ncbi letter codes. `http://www.ncbi.nlm.nih.gov/staff/tao/tools/tool_lettercode.html`.