

# Package ‘qckitfastq’

October 18, 2024

**Type** Package

**Title** FASTQ Quality Control

**Version** 1.21.0

**Description** Assessment of FASTQ file format with multiple metrics including quality score, sequence content, overrepresented sequence and Kmers.

**License** Artistic-2.0

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 6.1.1

**SystemRequirements** GNU make

**biocViews** Software,QualityControl,Sequencing

**LinkingTo** Rcpp, RSeqAn

**Imports** magrittr, ggplot2, dplyr, seqTools, zlibbioc, data.table, reshape2, grDevices, graphics, stats, utils, Rcpp, rlang, RSeqAn

**Biarch** True

**Suggests** knitr, rmarkdown, kableExtra, testthat

**VignetteBuilder** knitr

**git\_url** <https://git.bioconductor.org/packages/qckitfastq>

**git\_branch** devel

**git\_last\_commit** 1e671c4

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.20

**Date/Publication** 2024-10-17

**Author** Wenyue Xing [aut],  
August Guang [aut, cre]

**Maintainer** August Guang <[august.guang@gmail.com](mailto:august.guang@gmail.com)>

## Contents

adapter_content . . . . .	2
calc_adapter_content . . . . .	3
calc_format_score . . . . .	4
calc_over_rep_seq . . . . .	4
dimensions . . . . .	5
find_format . . . . .	5
GC_content . . . . .	6
gc_per_read . . . . .	6
kmer_count . . . . .	7
overrep_kmer . . . . .	7
overrep_reads . . . . .	8
per_base_quality . . . . .	9
per_read_quality . . . . .	9
plot_adapter_content . . . . .	10
plot_GC_content . . . . .	11
plot_outliers . . . . .	11
plot_overrep_kmer . . . . .	12
plot_overrep_reads . . . . .	13
plot_per_base_quality . . . . .	13
plot_per_read_quality . . . . .	14
plot_read_content . . . . .	14
plot_read_length . . . . .	15
qual_score_per_read . . . . .	16
read_base_content . . . . .	16
read_content . . . . .	17
read_length . . . . .	17
run_all . . . . .	18

<b>Index</b>	<b>19</b>
--------------	-----------

---

adapter_content	<i>Creates a sorted from most frequent to least frequent abundance table of adapters that are found to be present in the reads at greater than 0.1% of the reads. If output_file is selected then will save the entire set of adapters and counts. Only available for macOS/Linux due to dependency on C++14.</i>
-----------------	---

---

### Description

Creates a sorted from most frequent to least frequent abundance table of adapters that are found to be present in the reads at greater than 0.1% of the reads. If output\_file is selected then will save the entire set of adapters and counts. Only available for macOS/Linux due to dependency on C++14.

### Usage

```
adapter_content(infile, adapter_file = system.file("extdata",
  "adapters.txt", package = "qckitfastq"), output_file = NA)
```

**Arguments**

infile            the path to a gzipped FASTQ file  
adapter\_file    Path to adapters.txt file. Default from package.  
output\_file     File to save data frame to. Default NA.

**Value**

Sorted table of adapters and counts.

**Examples**

```
if(.Platform$OS.type != "windows") {  
  infile <- system.file("extdata", "test.fq.gz",  
    package = "qckitfastq")  
  adapter_content(infile)[1:5]  
}
```

---

calc\_adapter\_content    *Compute adapter content in reads. This function is only available for macOS/Linux.*

---

**Description**

Compute adapter content in reads. This function is only available for macOS/Linux.

**Usage**

```
calc_adapter_content(infile, adapters)
```

**Arguments**

infile            filepath to fastq sequence  
adapters          filepath to adapters

**Value**

map object with adapter names as the key and the number of times the adapters appears in the reads as the value

**Examples**

```
if(.Platform$OS.type != "windows") {  
  adapter_file <- system.file("extdata", "adapters.txt", package = "qckitfastq")  
  infile <- system.file("extdata", "test.fq.gz", package = "qckitfastq")  
  content <- calc_adapter_content(infile, adapter_file)  
}
```

---

calc\_format\_score      *Calculate score based on Illumina format*

---

### Description

Calculate score based on Illumina format

### Usage

```
calc_format_score(score, score_format)
```

### Arguments

score                    An ascii quality score from the fastq  
score\_format            The illumina format

### Value

a string as with the best guess as to the illumina format

### Examples

```
calc_format_score("A", "Sanger")
```

---

calc\_over\_rep\_seq      *Calculate sequece counts for each unique sequence and create a table with unique sequences and corresponding counts*

---

### Description

Calculate sequece counts for each unique sequence and create a table with unique sequences and corresponding counts

### Usage

```
calc_over_rep_seq(infile, min_size = 5L, buffer_size = 1000000L)
```

### Arguments

infile                    A string giving the path for the fastqfile  
min\_size                  An int for ththresholding over representation  
buffer\_size                An int for the number of lines to keep in memory

### Value

calculate overrepresented sequence count

### Examples

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
calc_over_rep_seq(infile)[seq_len(5)]
```

---

dimensions	<i>Extract the number of columns and rows for a FASTQ file using seqTools.</i>
------------	--

---

**Description**

Extract the number of columns and rows for a FASTQ file using seqTools.

**Usage**

```
dimensions(fseq, sel)
```

**Arguments**

fseq	an object that is the read result of the seq.read function
sel	'reads' for #reads/rows, 'positions' for #positions/columns

**Value**

a numeric value of the number of reads or the number of positions

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz",
  package = "qckitfastq")
fseq <- seqTools::fastqq(infile, k=6)
dimensions(fseq, "reads")
```

---

find_format	<i>Gets quality score encoding format from the FASTQ file. Return possibilities are Sanger(Illumina1.8), Solexa(Illumina1.0), Illumina1.3, and Illumina1.5. This encoding is heuristic based and may not be 100 since there is overlap in the encodings used, so it is best if you already know the format.</i>
-------------	---

---

**Description**

Gets quality score encoding format from the FASTQ file. Return possibilities are Sanger(Illumina1.8), Solexa(Illumina1.0), Illumina1.3, and Illumina1.5. This encoding is heuristic based and may not be 100 since there is overlap in the encodings used, so it is best if you already know the format.

**Usage**

```
find_format(infile, reads_used)
```

**Arguments**

infile	A string giving the path for the fastq file
reads_used	int, the number of reads to use to determine the encoding format.

**Value**

A string denoting the read format. Possibilities are Sanger, Solexa, Illumina1.3, and Illumina1.5.

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
find_format(infile,100)
```

---

GC_content	<i>Calculates GC content percentage for each read in the dataset.</i>
------------	---

---

**Description**

Calculates GC content percentage for each read in the dataset.

**Usage**

```
GC_content(infile, output_file = NA)
```

**Arguments**

infile	the object that is the path to the FASTQ file
output_file	File to write results to. Default NA.

**Value**

Data frame with read ID and GC content of each read.

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz",
  package = "qckitfastq")
head(GC_content(infile))
```

---

gc_per_read	<i>Calculate GC nucleotide sequence content per read of the FASTQ gzipped file</i>
-------------	--

---

**Description**

Calculate GC nucleotide sequence content per read of the FASTQ gzipped file

**Usage**

```
gc_per_read(infile)
```

**Arguments**

infile	A string giving the path for the fastqfile
--------	--

**Value**

GC content percentage per read

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
gc_per_read(infile)[1:10]
```

---

kmer\_count

*Return kmer count per sequence for the length of kmer desired*

---

**Description**

Return kmer count per sequence for the length of kmer desired

**Usage**

```
kmer_count(infile, k, output_file = NA)
```

**Arguments**

infile            the object that is the path to gzipped FASTQ file  
k                    the length of kmer  
output\_file        File to save plot to. Default NA.

**Value**

kmers counts per sequence

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz",
  package = "qckitfastq")
km<-kmer_count(infile,k=4)
km[1:20,1:10]
```

---

overrep\_kmer

*Generate overrepresented kmers of length k based on their observed to expected ratio at each position across all sequences in the dataset. The expected proportion of a length k kmer assumes site independence and is computed as the sum of the count of each base pair in the kmer times the probability of observing that base pair in the data set, i.e.  $P(A)count\_in\_kmer(A)+P(C)count\_in\_kmer(C)+...$  The observed to expected ratio is computed as  $\log_2(obs/exp)$ . Those with  $obsexp\_ratio > 2$  are considered to be overrepresented and appear in the returned data frame along with their position in the sequence.*

---

**Description**

Generate overrepresented kmers of length  $k$  based on their observed to expected ratio at each position across all sequences in the dataset. The expected proportion of a length  $k$  kmer assumes site independence and is computed as the sum of the count of each base pair in the kmer times the probability of observing that base pair in the data set, i.e.  $P(A)\text{count\_in\_kmer}(A)+P(C)\text{count\_in\_kmer}(C)+\dots$ . The observed to expected ratio is computed as  $\log_2(\text{obs}/\text{exp})$ . Those with  $\text{obsexp\_ratio} > 2$  are considered to be overrepresented and appear in the returned data frame along with their position in the sequence.

**Usage**

```
overrep_kmer(infile, k, output_file = NA)
```

**Arguments**

```
infile      path to gzipped FASTQ file
k           the kmer length
output_file File to save plot to. Default NA.
```

**Value**

Data frame with columns: Position (in read), Obsexp\_ratio, & Kmer

**Examples**

```
infile <-system.file("extdata", "test.fq.gz",
  package = "qckitfastq")
overrep_kmer(infile,k=4)
```

---

overrep_reads	<i>Sort all sequences per read by count.</i>
---------------	--

---

**Description**

Sort all sequences per read by count.

**Usage**

```
overrep_reads(infile, output_file = NA)
```

**Arguments**

```
infile      Path to gzipped FASTQ file.
output_file File to save data frame to. Default NA.
```

**Value**

Table of sequences sorted by count.



**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz",
  package = "qckitfastq")
overrep_reads(infile)[1:5,]
```

---

per_base_quality	<i>Compute the mean, median, and percentiles of quality score per base. This is returned as a data frame.</i>
------------------	---

---

**Description**

Compute the mean, median, and percentiles of quality score per base. This is returned as a data frame.

**Usage**

```
per_base_quality(infile, output_file = NA)
```

**Arguments**

infile	Path to a gzipped FASTQ file
output_file	File to write results in CSV format to. Default NA.

**Value**

A dataframe of the mean, median and quantiles of the FASTQ file

**Author(s)**

Wenyue Xing, <wenyue\_xing@brown.edu>  
August Guang, <august\_guang@brown.edu>

**Examples**

```
per_base_quality(system.file("extdata", "10^5_reads_test.fq.gz",
  package = "qckitfastq"))
```

---

per_read_quality	<i>Compute the mean quality score per read. per_read_quality</i>
------------------	--

---

**Description**

Compute the mean quality score per read. per\_read\_quality

**Usage**

```
per_read_quality(infile, output_file = NA)
```

**Arguments**

infile            Path to FASTQ file  
output\_file      File to write plot to. Will not write to file if NA. Default NA.

**Value**

Data frame of mean quality score per read

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")  
prq <- per_read_quality(infile)
```

---

plot\_adapter\_content    *Creates a bar plot of the top 5 most present adapter sequences.*

---

**Description**

Creates a bar plot of the top 5 most present adapter sequences.

**Usage**

```
plot_adapter_content(ac_sorted, output_file = NA)
```

**Arguments**

ac\_sorted        Sorted table of adapters and counts.  
output\_file      File to save data frame to. Default NA.

**Value**

Barplot of top 5 most frequent adapter sequences.

**Examples**

```
if(.Platform$OS.type != "windows") {  
  infile <- system.file("extdata", "test.fq.gz", package = "qckitfastq")  
  ac_sorted <- adapter_content(infile)  
  plot_adapter_content(ac_sorted)  
}
```

---

plot_GC_content	<i>Generate mean GC content histogram.</i>
-----------------	--

---

**Description**

Generate mean GC content histogram.

**Usage**

```
plot_GC_content(gc_df, output_file = NA)
```

**Arguments**

gc_df	the object that is the GC content vectors generated from GC content function
output_file	File to write plot to. Will not write to file if NA. Default NA.

**Value**

A histogram of mean GC content.

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
gc_df <- GC_content(infile)
plot_GC_content(gc_df)
```

---

plot_outliers	<i>Determine how to plot outliers. Heuristic used is whether their obsexp_ratio differs by more than 1 and whether they fall into the same bin or not. If for 2 outliers, obsexp_ratio differs by less than .4 and they are in the same bin, then combine into a single plotting point. NOT FULLY FUNCTIONAL</i>
---------------	--

---

**Description**

Determine how to plot outliers. Heuristic used is whether their obsexp\_ratio differs by more than 1 and whether they fall into the same bin or not. If for 2 outliers, obsexp\_ratio differs by less than .4 and they are in the same bin, then combine into a single plotting point. NOT FULLY FUNCTIONAL

**Usage**

```
plot_outliers(overkm, top_num)
```

**Arguments**

overkm	data frame with columns pos, obsexp_ratio, and kmer that has already been reordered by descending obsexp_ratio
top_num	number of most overrepresented kmers to plot. Default is 5.

**Value**

currently 0 as function is not fully working.

---

plot_overrep_kmer	<i>Create a box plot of the <math>\log_2(\text{observed/expected})</math> ratio across the length of the sequence as well as top overrepresented kmers. Only ratios greater than 2 are included in the box plot. Default is 20 bins across the length of the sequence and the top 2 overrepresented kmers, but this can be changed by the user.</i>
-------------------	---

---

**Description**

Create a box plot of the  $\log_2(\text{observed/expected})$  ratio across the length of the sequence as well as top overrepresented kmers. Only ratios greater than 2 are included in the box plot. Default is 20 bins across the length of the sequence and the top 2 overrepresented kmers, but this can be changed by the user.

**Usage**

```
plot_overrep_kmer(overkm, bins = 20, top_num = 2, output_file = NA)
```

**Arguments**

overkm	data frame with columns pos, obsexp_ratio, and kmer
bins	number of intervals across the length of the sequence
top_num	number of most overrepresented kmers to plot
output_file	File to write plot to. Will not write to file if NA. Default NA.

**Value**

A box plot of the  $\log_2(\text{observed/expected})$  ratio across the length of the sequence

**Examples**

```
infile <- system.file("extdata", "test.fq.gz",
  package = "qckitfastq")
over_km <- overrep_kmer(infile,k=4)
plot_overrep_kmer(over_km)
```

---

plot\_overrep\_reads     *Plot the top 5 sequences*

---

**Description**

Plot the top 5 sequences

**Usage**

```
plot_overrep_reads(overrep_reads, output_file = NA)
```

**Arguments**

overrep\_reads     the table that sorts the sequence content and corresponding counts in descending order

output\_file       File to save plot to. Will not write to file if NA. Default NA.

**Value**

plot of the top 5 overrepresented sequences

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
overrep_df <- overrep_reads(infile)
plot_overrep_reads(overrep_df)
```

---

plot\_per\_base\_quality     *Generate a boxplot of the per position quality score.*

---

**Description**

Generate a boxplot of the per position quality score.

**Usage**

```
plot_per_base_quality(per_base_quality, output_file = NA)
```

**Arguments**

per\_base\_quality     a data frame of the mean, median and quantiles of sequence quality per base. Most likely generated with the 'per\_base\_quality' function.

output\_file       File to save plot to. Will not write to file if NA. Default NA.

**Value**

A boxplot of per position quality score distribution.

**Examples**

```
pbq <- per_base_quality(system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq"))
plot_per_base_quality(pbq)
```

---

plot\_per\_read\_quality *Plot the mean quality score per sequence as a histogram. High quality sequences are those mostly distributed over 30. Low quality sequences are those mostly under 30.* plot\_per\_read\_quality

---

**Description**

Plot the mean quality score per sequence as a histogram. High quality sequences are those mostly distributed over 30. Low quality sequences are those mostly under 30. plot\_per\_read\_quality

**Usage**

```
plot_per_read_quality(prq, output_file = NA)
```

**Arguments**

prq                    Data frame from per\_read\_quality function  
output\_file            File to write plot to. Will not write to file if NA. Default NA.

**Value**

Plot of mean quality score per read

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
prq <- per_read_quality(infile)
plot_per_read_quality(prq)
```

---

plot\_read\_content      *Plot the per position nucleotide content.*

---

**Description**

Plot the per position nucleotide content.

**Usage**

```
plot_read_content(read_content, output_file = NA)
```

**Arguments**

read\_content          Data frame produced by read\_content function.  
output\_file            File to save plot to. Will not write to file if NA. Default NA.

**Value**

ggplot line plot of all nucleotide content including A, T, G, C and N

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
fseq <- seqTools::fastqq(infile,k=6)
read_content <- read_content(fseq)
plot_read_content(read_content)
```

---

plot\_read\_length      *Plot a histogram of the number of reads with each read length.*

---

**Description**

Plot a histogram of the number of reads with each read length.

**Usage**

```
plot_read_length(read_len, output_file = NA)
```

**Arguments**

read\_len      Data frame of read lengths and number of reads with that length.  
output\_file    File to save plot to. Default is NA, i.e. do not write to file.

**Value**

A histogram of the read length distribution.

**Author(s)**

Wenyue Xing, <wenyue\_xing@brown.edu>, August Guang, <august\_guang@brown.edu>

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
fseq <- seqTools::fastqq(infile,k=6)
read_len <- read_length(fseq)
plot_read_length(read_len)
```

---

qual\_score\_per\_read     *Calculate the mean quality score per read of the FASTQ gzipped file*

---

**Description**

Calculate the mean quality score per read of the FASTQ gzipped file

**Usage**

```
qual_score_per_read(infile)
```

**Arguments**

infile                    A string giving the path for the fastqfile

**Value**

mean quality per read

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
qual_score_per_read(infile)$q50_per_position[1:10]
```

---

read\_base\_content     *Compute nucleotide content per position for a single base pair. Wrapper function around seqTools.*

---

**Description**

Compute nucleotide content per position for a single base pair. Wrapper function around seqTools.

**Usage**

```
read_base_content(fseq, content)
```

**Arguments**

fseq                    a seqTools::fastqq object  
content                nucleotide. Options are "A","T","G","C","N"(either capital or lower case)

**Value**

Nucleotide sequence content per position.

**Author(s)**

Wenyue Xing, <wenyue\_xing@brown.edu>, August Guang <august\_guang@brown.edu>



**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
fseq <- seqTools::fastqq(infile,k=6)
read_base_content(fseq,"A")
```

---

read_content	<i>Compute nucleotide content per position. Wrapper function around seqTools.</i>
--------------	---

---

**Description**

Compute nucleotide content per position. Wrapper function around seqTools.

**Usage**

```
read_content(fseq, output_file = NA)
```

**Arguments**

fseq	a seqTools::fastqq object
output_file	File to write results in CSV format to. Will not write to file if NA. Default NA.

**Value**

Data frame of nucleotide sequence content per position

**Examples**

```
infile <- system.file("extdata", "10^5_reads_test.fq.gz", package = "qckitfastq")
fseq <- seqTools::fastqq(infile,k=6)
read_content(fseq)
```

---

read_length	<i>Creates a data frame of read lengths and the number of reads with that read length.</i>
-------------	--

---

**Description**

Creates a data frame of read lengths and the number of reads with that read length.

**Usage**

```
read_length(fseq, output_file = NA)
```

**Arguments**

fseq	a seqTools object produced by seqTools::fastqq on the raw FASTQ file
output_file	File to save data frame to. Default NA.

**Value**

Data frame of read lengths and number of reads with that length.

**Examples**

```
infile <- system.file("extdata", "test.fq.gz",
  package = "qckitfastq")
fseq <- seqTools::fastqq(infile, k=6)
read_len <- read_length(fseq)
```

---

run_all	<i>Will run all functions in the qckitfastq suite and save the data frames and plots to a user-provided directory. Plot names are supplied by default.</i>
---------	--

---

**Description**

Will run all functions in the qckitfastq suite and save the data frames and plots to a user-provided directory. Plot names are supplied by default.

**Usage**

```
run_all(infile, dir)
```

**Arguments**

infile	Path to gzipped FASTQ file
dir	Directory to save results to

**Value**

Generate files from all functions

**Examples**

```
infile <- system.file("extdata", "test.fq.gz",
  package = "qckitfastq")
testfolder <- tempdir()
run_all(infile, testfolder)
```

# Index

adapter\_content, 2

calc\_adapter\_content, 3  
calc\_format\_score, 4  
calc\_over\_rep\_seq, 4

dimensions, 5

find\_format, 5

GC\_content, 6  
gc\_per\_read, 6

kmer\_count, 7

overrep\_kmer, 7  
overrep\_reads, 8

per\_base\_quality, 9  
per\_read\_quality, 9  
plot\_adapter\_content, 10  
plot\_GC\_content, 11  
plot\_outliers, 11  
plot\_overrep\_kmer, 12  
plot\_overrep\_reads, 13  
plot\_per\_base\_quality, 13  
plot\_per\_read\_quality, 14  
plot\_read\_content, 14  
plot\_read\_length, 15

qual\_score\_per\_read, 16

read\_base\_content, 16  
read\_content, 17  
read\_length, 17  
run\_all, 18