

# Package ‘DNABarcodeCompatibility’

June 12, 2018

**Type** Package

**Title** A tool for optimizing combinations of DNA barcodes used in multiplexed experiments on Next Generation Sequencing platforms.

**Version** 1.0.0

**Author** Céline Trébeau, Jacques Boutet de Monvel, Fabienne Wong Jun Tai, Raphaël Etournay

**Maintainer** Céline Trébeau <celine.trebeau@pasteur.fr>

**Description** The package allows one to obtain optimized combinations of DNA barcodes to be used for multiplex sequencing. In each barcode combination, barcodes are pooled with respect to Illumina chemistry constraints. Combinations can be filtered to keep those that are robust against substitution and insertion/deletion errors thereby facilitating the demultiplexing step. In addition, the package provides an optimizer function to further favor the selection of barcode combinations with least redundancy of DNA barcodes.

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.0.1

**Imports** dplyr, tidyr, numbers, purrr, DNABarcodes, stringr, rJava

**Suggests** knitr, rmarkdown, testthat

**VignetteBuilder** knitr

## R topics documented:

DNABarcodeCompatibility-package . . . . .	2
distance_filter . . . . .	2
experiment_design . . . . .	3
file_loading_and_checking . . . . .	4
get_all_combinations . . . . .	5
get_random_combinations . . . . .	6
IlluminaIndexes . . . . .	6
IlluminaIndexesRaw . . . . .	7
optimize_combinations . . . . .	7

Index	9
-------	---

---

DNABarcodeCompatibility-package

*DNABarcodeCompatibility: to find least-redundant sets of compatible barcodes for multiplex experiments performed on next generation sequencing platforms.*

---

## Description

The DNABarcodeCompatibility package provides six functions to load DNA barcodes, and to generate, filter and optimize sets of barcode combinations for multiplex sequencing experiments. In particular, barcode combinations are selected to be compatible with respect to Illumina chemistry constraints, and can be filtered to keep those that are robust against substitution and insertion/deletion errors thereby facilitating the demultiplexing step. In addition, the package provides an optimizer function to further favor the selection of compatible barcode combinations with least redundancy of DNA barcodes.

---

distance\_filter

*Select barcode combinations with error correction properties*

---

## Description

Filters a list of barcode combinations for a given distance metric (hamming or seqlev) and threshold in order to produce a list of barcodes satisfying the distance constraints.

## Usage

```
distance_filter(index_df, combinations_m, metric, d)
```

## Arguments

index_df	A dataframe containing barcodes identifiers, corresponding DNA sequences along with GC content and presence of homopolymers.
combinations_m	A matrix of compatible barcode combinations.
metric	The type of distance (hamming or seqlev).
d	The minimum value of the distance.

## Details

The "hamming" distance is suitable for correcting substitution errors. The "seqlev" distance is suitable for correcting both substitution and insertion/deletion errors.

## Value

A filtered matrix containing the identifiers of the barcodes satisfying the distance constraints.

## References

Buschmann, T. 2015. The Systematic Design and Application of Robust DNA Barcodes.  
 Buschmann, T. 2017. DNABarcodes: an R package for the systematic construction of DNA sample tags. *Bioinformatics* 33, 920–922.

**See Also**

[get\\_all\\_combinations](#), [get\\_random\\_combinations](#)

**Examples**

```
barcodes <- DNABarcodeCompatibility::IlluminaIndexes
m <- get_all_combinations(barcodes, 2, 4)
distance_filter(barcodes, m, "hamming", 3)
```

---

experiment_design	<i>Find a set of barcode combinations with least redundant barcodes for single and dual indexing</i>
-------------------	------------------------------------------------------------------------------------------------------

---

**Description**

This function uses the Shannon Entropy to identify a set of compatible barcode combinations with least redundancy between DNA barcodes, in the context of single and dual indexing. It performs either an exhaustive or a random-greedy search of compatible DNA-barcode combinations depending on the size of the DNA-barcode population and of the number of samples to be multiplexed.

**Usage**

```
experiment_design(file1, sample_number, mplex_level, chemistry = 4,
  file2 = NULL, export = NULL, metric = NULL, d = 3)
```

**Arguments**

file1	The input data file that contains 2 columns separated by a space or a tabulation, namely the sequence identifiers and corresponding DNA sequence
sample_number	Number of libraries to be sequenced.
mplex_level	The number at which the barcodes will be multiplexed.
chemistry	An integer representing the number of channels (1, 2, 4) of the desired Illumina platform.
file2	The input data file that contains 2 columns separated by a space or a tabulation, namely the sequence identifiers and corresponding DNA sequence; used for dual-indexing.
export	If not NULL, results are saved in a csv file at the specified path.
metric	The type of distance (hamming or seqlev).
d	The minimum value of the distance.

**Details**

By specifying the total number of libraries and the number of libraries to be multiplexed, this function returns an optimal set of DNA-barcode combinations to be used for sequencing.

The inputs of the algorithm are a list of  $n$  distinct barcodes, the number  $N$  of required libraries, and the multiplex level  $k$ ;  $N = ak$ , where  $a$  is the number of lanes of the flow cells to be used for the experiment.

- Step 1:

This step consists of identifying a set of compatible barcode combinations. Given the number of barcodes and the multiplex level, the total number of barcode combinations (compatible or not) reads:

$$\binom{n}{k}$$

If this number is not too large, the algorithm will perform an exhaustive search and output all compatible combinations of k barcodes. Otherwise, it will proceed by picking up combinations at random, in order to identify a large enough set of compatible barcode combinations.

- Step 2:

Finds an optimized set of barcode combinations in which barcode redundancy is minimized (see details in [optimize\\_combinations](#))

### Value

A dataframe containing compatible DNA-barcode combinations organized by lanes of the flow cell.

### Examples

```
write.table(DNABarcodeCompatibility::IlluminaIndexesRaw,
  txtfile <- tempfile(), row.names = FALSE, col.names = FALSE, quote=FALSE)
experiment_design(file1=txtfile, sample_number=18, mplex_level=3, chemistry=4)
```

---

file\_loading\_and\_checking

*Loading and checking DNA barcodes.*

---

### Description

Loads the file containing DNA barcodes and analyze barcode content.

### Usage

```
file_loading_and_checking(file)
```

### Arguments

file	The input data file that contains 2 columns separated by a space or a tabulation, namely the sequence identifiers and corresponding DNA sequence.
------	---------------------------------------------------------------------------------------------------------------------------------------------------

### Details

This function loads the DNA barcodes from the input file and checks barcodes for unicity (identifier and sequence), DNA content, and equal size. It also calculates the fraction of G and C relative to A and T, as referred to as "GC content", and it detects the presence of homopolymers of length  $\geq 3$ .

**Value**

A dataframe containing sequence identifiers, nucleotide sequence, GC content, presence of homopolymers.

**Examples**

```
write.table(DNABarcodeCompatibility::IlluminaIndexesRaw,
  txtfile <- tempfile(), row.names = FALSE, col.names = FALSE, quote=FALSE)
file_loading_and_checking(txtfile)
```

---

get_all_combinations	<i>Get all compatible combinations.</i>
----------------------	-----------------------------------------

---

**Description**

Finds the exhaustive set of compatible barcode combinations.

**Usage**

```
get_all_combinations(index_df, mplex_level, chemistry)
```

**Arguments**

index_df	A dataframe containing barcodes identifiers, corresponding DNA sequences along with GC content and presence of homopolymers.
mplex_level	The number at which the barcodes will be multiplexed.
chemistry	An integer representing the number of channels (1, 2, 4) of the Illumina platform to be used.

**Details**

Be aware that the total number of combinations may become prohibitively large for large barcode sets and large multiplexing numbers.

**Value**

A matrix containing the identifiers of compatible barcode combinations.

**See Also**

[get\\_random\\_combinations](#)

**Examples**

```
get_all_combinations(DNABarcodeCompatibility::IlluminaIndexes, 3, 4)
```

---

```
get_random_combinations
```

*Get a large set of compatible combinations.*

---

### Description

Finds a randomly generated set of at most 1000 combinations of compatible barcodes.

### Usage

```
get_random_combinations(index_df, mplex_level, chemistry)
```

### Arguments

<code>index_df</code>	A dataframe containing barcodes identifiers, corresponding DNA sequences along with GC content and presence of homopolymers.
<code>mplex_level</code>	The number at which the barcodes will be multiplexed.
<code>chemistry</code>	An integer representing the number of channels (1, 2, 4) of the Illumina platform to be used.

### Details

This function is suited if the total number of possible combinations is too high for an exhaustive search to be possible in a reasonable amount of time.

### Value

A matrix containing the identifiers of compatible barcode combinations.

### See Also

[get\\_all\\_combinations](#)

### Examples

```
get_random_combinations(DNABarcodeCompatibility::IlluminaIndexes, 3, 4)
```

---

```
IlluminaIndexes
```

*Barcode dataset from Illumina with features.*

---

### Description

48 barcodes from Illumina TruSeq Small RNA kits along with percentage in CG content and presence of homopolymers of size  $\geq 3$

### Usage

```
IlluminaIndexes
```

**Format**

A data frame with 48 rows and 4 variables:

**Id** barcode identifier

**sequence** DNA sequence

**GC\_content** percentage of G and C relative to A and T

**homopolymer** presence of nucleotide repetitions ( $\geq 3$ ) ...

---

IlluminaIndexesRaw	<i>Barcode dataset from Illumina.</i>
--------------------	---------------------------------------

---

**Description**

48 barcodes from Illumina TruSeq Small RNA kits

**Usage**

```
IlluminaIndexesRaw
```

**Format**

A data frame with 48 rows and 2 variables:

**V1** barcode identifier

**V2** DNA sequence ...

---

optimize_combinations	<i>Find a set of barcode combinations with least redundant barcodes</i>
-----------------------	-------------------------------------------------------------------------

---

**Description**

This function uses the Shannon Entropy to identify a set of compatible barcode combinations with least redundancy between DNA barcodes.

**Usage**

```
optimize_combinations(combination_m, nb_lane, index_number)
```

**Arguments**

**combination\_m** A matrix of compatible barcode combinations.

**nb\_lane** The number of lanes to be use for sequencing (i.e. the number of libraries divided by the multiplex level).

**index\_number** The total number of distinct DNA barcodes in the dataset.

### Details

$N/k$  compatible combinations are then selected using a Shannon entropy maximization approach. It can be shown that the maximum value of the entropy that can be attained for a selection of  $N$  barcodes among  $n$ , with possible repetitions, reads:

$$S_{max} = -(n-r) \frac{\lfloor N/n \rfloor}{N} \log\left(\frac{\lfloor N/n \rfloor}{N}\right) - r \frac{\lceil N/n \rceil}{N} \log\left(\frac{\lceil N/n \rceil}{N}\right)$$

where  $r$  denotes the rest of the division of  $N$  by  $n$ , while

$$\lfloor N/n \rfloor$$

and

$$\lceil N/n \rceil$$

denote the lower and upper integer parts of  $N/n$ , respectively.

+ Case 1: number of lanes < number of compatible DNA-barcode combinations

This function seeks for compatible DNA-barcode combinations of highest entropy. In brief this function uses a randomized greedy descent algorithm to find an optimized selection. Note that the resulting optimized selection may not be globally optimal. It is actually close to optimal and much improved in terms of non-redundancy of DNA barcodes used, compared to a randomly chosen set of combinations of compatible barcodes.

+ Case 2: number of lanes >= number of compatible DNA-barcode combinations

In such a case, there are not enough compatible DNA-barcode combinations and redundancy is inevitable.

### Value

A matrix containing an optimized set of combinations of compatible barcodes.

### See Also

[get\\_all\\_combinations](#), [get\\_random\\_combinations](#), [experiment\\_design](#)

### Examples

```
m <- get_random_combinations(DNABarcodeCompatibility::IlluminaIndexes, 3, 4)
optimize_combinations(m, 12, 48)
```



# Index

## \*Topic **datasets**

    IlluminaIndexes, [6](#)

    IlluminaIndexesRaw, [7](#)

distance\_filter, [2](#)

DNABarcodeCompatibility-package, [2](#)

experiment\_design, [3](#), [8](#)

file\_loading\_and\_checking, [4](#)

get\_all\_combinations, [3](#), [5](#), [6](#), [8](#)

get\_random\_combinations, [3](#), [5](#), [6](#), [8](#)

IlluminaIndexes, [6](#)

IlluminaIndexesRaw, [7](#)

optimize\_combinations, [4](#), [7](#)