checkMyIndex: a web-based R/Shiny interface for choosing compatible sequencing indexes

Hugo Varet, Jean-Yves Coppée

Bioinformatics, 35(5), 2019, 901–902
doi: 10.1093/bioinformatics/bty706
Advance Access Publication Date: 25 August 2018
Applications Note

Data and text mining

checkMyIndex: a web-based R/Shiny interface for choosing compatible sequencing indexes

Hugo Varet[1,2,*] and Jean-Yves Coppée[2]

[1]Center of Bioinformatics, Biostatistics and Integrative Biology, Institut Pasteur – Bioinformatics and Biostatistics Hub – C3BI, USR 3756 IP CNRS, F-75015 Paris, France and [2]Center for Technological Resources and Research, Institut Pasteur – Transcriptome and Epigenome Platform – Biomics Pole – C2RT, F-75015 Paris, France

*To whom correspondence should be addressed.

Associate Editor: Jonathan Wren

Received on May 15, 2018; revised on August 11, 2018; editorial decision on August 13, 2018; accepted on August 23, 2018

Abstract

Summary: When sequencing several libraries simultaneously, the selection of compatible combinations of indexes is critical for ensuring that the sequencer will be able to decipher the short, sample-specific barcodes added to each fragment. However, researchers have few tools to help them choose optimal indexes. Here, we present checkMyIndex, an online R/Shiny application that facilitates the selection of the right indexes as a function of the experimental constraints.

Availability and implementation: checkMyIndex is available free of charge at https://checkmyindex.pasteur.fr as an online, web-based R/Shiny application. The source code is available on GitHub at https://github.com/PF2-pasteur-fr/checkMyIndex.

Contact: hugo.varet@pasteur.fr.

1 Introduction

The sequencing of RNA and DNA libraries has become a standard experimental tool in the life sciences. It can notably be used to characterize and understand biological mechanisms (Goodwin *et al.*, 2016). For instance, knowledge of DNA sequences can help researchers to discover disease-specific single nucleotide polymorphisms, whereas transcriptome analyses can reveal differentially expressed genes or provide information for genome annotation. As sequencing capacities increase, some biological applications do not require all the reads produced by a single sequencing lane. Hence, in order to optimize costs, researchers often choose to sequence several samples simultaneously by adding short (typically 6- to 8-base-pair) sample-specific barcodes to their libraries. The indexes are then read by the sequencer, and each read is assigned to its original sample via an a posteriori demultiplexing procedure. In this context, the selection of compatible indexes is a crucial step; if the indexes are not compatible, the sequencer will not be able to read each index entirely. Indeed, the compatibility of a set of indexes depends on (i) the indexes' sequence composition at each position, and (ii) the sequencer's chemistry (e.g. Illumina HiSeq or NextSeq) [Illumina Index Adapters Pooling Guide, Document # 1000000041074 v02 (2018)]. By way of an example, two- or four-channel chemistry requires each color-transformed nucleotide to be present at least once at each position. Using incompatible indexes would cause the demultiplexing procedure to fail, and so the sequencing reads would not be assigned to the corresponding biological samples. Accordingly, the reads could not be used as input for any future bioinformatics analyses. Although simultaneously sequencing a pool of libraries can lower costs, this strategy must therefore be carefully planned. Several tools have been developed to check index compatibility. For instance, Illumina's Experiment Manager (IEM) software checks the compatibility of indexes provided by the user. However, IEM is available only for computers running Windows operating systems, and cannot build combinations that fulfil complex criteria (such as the use of single indexes in several lanes). sicTool (http://amaryllisnucleics.com/sicTool/) also provides a web interface with advanced options but does not handle the different Illumina chemistries. Here, we describe the checkMyIndex web-based application. When given a set of available indexes, checkMyIndex generates compatible combinations that meet the idiosyncratic experimental conditions imposed by both the research objective and the user's preferences. Moreover, checkMyIndex can help staff in core facilities to generate good-quality, ready-to-analyze sequencing data.

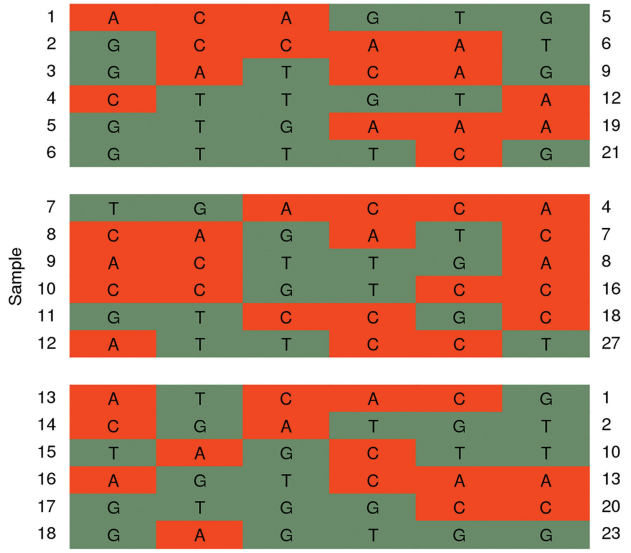

Fig. 1. A heat-map-like plot of a proposed flow cell design for sequencing 18 samples using 3 lanes and unique indexes. Samples (in rows) are grouped by pool/lane, and each nucleotide in each index is color-coded as a function of the chosen Illumina chemistry. Sample IDs are printed on the left, and index IDs are printed on the right

2 Results

We used the popular R package shiny (Chang *et al.*, 2017) to develop a user-friendly application (available free of charge at checkmyindex.pasteur.fr) dedicated to searching for compatible combinations of indexes. The corresponding R source code is hosted on GitHub (https://github.com/PF2-pasteur-fr/checkMyIndex) so that researchers can also run the application on their own computer. Note that checkMyIndex uses only native R code, and shiny is the only additional package required. An R script is also available for advanced users willing to run checkMyIndex from the command line.

Our application supports single- and dual-indexing in order to achieve higher multiplexing rates and decrease error rates (Kircher *et al.*, 2012). checkMyIndex is also compatible with the various chemistries used by the Illumina HiSeq, MiSeq, NextSeq and iSeq devices. With the four-channel chemistry (HiSeq and MiSeq), for instance, a red laser detects A/C bases and a green laser detects G/T bases; the indexes are compatible if there is at least one red light and one green light at each position. We refer the reader to both the checkMyIndex help page and the Illumina documentation for more detailed information on the different chemistries.

In practice, the user only needs to provide his/her available indexes as a simple, two-column, tab-separated text format file. The first column contains the index identifiers, and the second contains the corresponding short sequences (see either the GitHub repository or the checkMyIndex help page for examples). When the indexes are loaded, a dissimilarity score is attributed to each one. If the N indexes are labelled from I_1 to I_N, the score for index k is defined as $\min_{j \neq k} d_H(I_k, I_j)$, where d_H is the Hamming distance (the number of mismatches). This enables the user to check that each index differs sufficiently from all the others, so that the demultiplexing procedure can tolerate sequencing errors. Next, several constraints have to be defined via the interface: the total number of samples, the multiplexing rate (i.e. the number of samples per pool/lane) and whether the same index or the same combination of indexes can be used several times. The latter constraint has been designed to ensure a certain level of diversity and thus avoid the same indexes from being used all the time. Moreover, using unique indexes allows one to perform a small sequencing test on a single lane by pooling all the libraries.

From a technical standpoint, it is sometimes impossible or excessively length to find a solution that fulfils all the constraints. In fact, the number of index combinations to be tested can increase rapidly as the multiplexing rate rises. If n is the number of available indexes and m is the multiplexing rate, the number of possible combinations is defined by $n!/(m!(n-m)!)$. For instance, when looking for combinations of $m = 12$ indexes in a list of $n = 48$, the program may have to generate and test the compatibility of almost 7×10^{10} combinations. Since adding an index to an already compatible set does not revoke compatibility, our trick in this situation is to find a partial solution with a smaller number of samples per pool/lane and then to complete it to obtain the desired multiplexing rate. In practice, the algorithm decreases the desired multiplexing rate step by step until the number of possible combinations falls to below 2 00 000. This quickly provides an intermediate, partial solution that can then be completed with some of the remaining indexes. Using the figures above, a partial solution can be easily found with $m = 4$ (i.e. 1 94 580 possible combinations) and completed to reach $m = 12$. In most situations, checkMyIndex therefore returns a solution in a few seconds. It should be noted that a new score is computed for each index in the proposed solution, in order to assess its dissimilarity vs. the other indexes in the pool/lane.

Lastly, the user can visualize a heat-map-like plot of the solution in the corresponding tab (Fig. 1). Furthermore, the table containing the proposed flow cell design can be downloaded and saved for future experiments or processing.

3 Conclusion

checkMyIndex is a user-friendly, easy-to-use web application that facilitates searching for compatible indexes in sequencing experiments. Solutions are returned quickly (in a few seconds) by the underlying algorithm, and satisfy the constraints imposed by the user (such as not using the same index several times). Moreover, the structure of the R code will allow new features to be added (e.g. when new chemistries are developed).

Acknowledgements

We thank Odile Sismeiro, Caroline Proux, Juliana Pipoli da Fonseca, Christiane Bouchier, Sean Kennedy and Thomas Cokelaer for assistance and helpful suggestions. We also acknowledge Céline Trébeau and Raphaël Etournay for their constructive discussions, and Emmanuel Guichard and Nicolas Maillet for their help during the virtual machine set-up.

Funding

This work was supported by the France Génomique consortium [ANR10-INBS-09-08 and ANR-10-INBS-09-10].

Conflict of Interest: none declared.

References

Chang,W. *et al.* (2017) *Shiny: Web Application Framework for R. R Package Version 1.0.5*. doi: 10.1093/bioinformatics/btx308.

Goodwin,S. *et al.* (2016) Coming of age: ten years of next-generation sequencing technologies. *Nat. Rev. Genet.*, **17**, 333–351.

Kircher,M. *et al.* (2012) Double indexing overcomes inaccuracies in multiplex sequencing on the Illumina platform. *Nucleic Acids Res.*, **40**, e3.