fundiversity: a modular R package to compute functional diversity indices

Matthias Grenié[a,b,*], Hugo Gruson[b,c,**]

[a] *German Centre for Integrative Biodiversity Research (iDiv) Halle-Jena-Leipzig, Puschstraße 4, 04103 Leipzig, Germany*
[b] *CEFE Univ Montpellier …*
[c] *Tropical Medecine London*

Abstract

(max 350 words)

1. set the context for and purpose of the work;
2. indicate the approach and methods;
3. outline the main results;
4. identify the conclusions and the wider implications.

Keywords: biodiversity; diversity facet; R package; functional biogeography; functional ecology; community ecology

Running title (max 45 char.): fundiversity: functional diversity in R
Word count (3000-4000 incl. refs & captions): 1945

Introduction

Functional diversity, the diversity of traits across scales, is a major facet of biodiversity (Pavoine & Bonsall, 2011). It has been shown to relate to ecosystem functioning (Díaz & Cabido, 2001; Leps et al., 2006). Since its definition it has been widely used across ecological contexts (Cadotte et al., 2011). Many indices exist to characterize it across its three dimensions: richness, evenness, divergence (Pavoine & Bonsall, 2011). To compute these indices in a standardized ways ecologists rely on computational tools able to crunch the numbers for thousands of species and thousands of sites. In the last few years, R has been the programming language of choice for ecologists (Lai et al., 2019; R Core Team, 2021). The main tool available to compute functional diversity indices has been the `FD` package which has now accumulated more than 1200 citations (Laliberté et al., 2014). The `FD` package has not been updated since 2015 and it does not offer the flexibility needed by some users. Ecology increasingly use larger datasets which require more efficient computations (Farley et al., 2018; Wüest et al., 2020). The tools to compute functional diversity indices are thus in great need of improvement.

Need for modularity in computation. The main function of the `FD` package `dbFD()` lets the users compute a dozen functional diversity indices in a single call from raw trait data (Laliberté et al., 2014). While great for exploratory analyses this can increase computation time when only a single index is needed. Furthermore, it does not enforce good practice in choosing beforehand the appropriate functional diversity index for the question(s) asked (Legras et al., 2018; Mason et al., 2013; Schleuter et al., 2010). It encourages the user to fish the functional diversity index that matches their predicted relationships. This can lead the users to report all computed functional diversity indices even when there no clear expectations on different functional diversity facets and/or to report correlated indices (Legras et al., 2018; Mason et al.,

*Corresponding Author
**ORCID-ID <0000-0002-4659-7522> ORCID-ID <0000-0002-4094-1476>
Email addresses: `matthias.grenie@idiv.de` (Matthias Grenié), `hugo.gruson@normalsup.org` (Hugo Gruson)

2013; McPherson et al., 2018; Schleuter et al., 2010). Building the tool so that indices can be computed independently of one another has the added benefit to make maintenance and the addition of new indices easier.

Need for faster computation (with three solutions: algorithmic efficiency, parallelization, memoization). The average size of datasets analyzed in ecology increased several folds in the last years (Wüest et al., 2020). Considering that most analyses on functional diversity rely on null models that increase the data size by two or three orders of magnitude (Gotelli & Graves, 1996), the need for efficient computation is paramount. First, any improvement of the algorithmic efficiency to compute functional diversity indices could save sensible amount of time as its repeated many times. For example we noted that many R packages that compute functional diversity indices do not leverage on matrix algebra with its libraries available that can cut the number of operations by orders of magnitude. Second, functional diversity indices are generally computed over many mathematically independent sites. With the rise of multi-core computers, parallelization, i.e. splitting independent computations between independent Computing Processor Units (CPUs), is becoming the norm. Very few functional diversity R packages propose parallelization which leaves the burden of implementing it to the user. Furthermore, when they do, they do not rely on recently released `future` framework (Bengtsson, 2020) that allows the user to seamlessly parallelize her computations on multiple cores on a single machine, or across several machine, or even on a remote cluster. Third, computation on exact same input can be cached through a process called memoization (Wickham et al., 2021). This avoid wasting computing power on previously seen inputs. Several functional diversity indices rely on the computation of convex hulls across a multi-dimensional space (Cornwell et al., 2006; Villéger et al., 2008). Caching the results of this costly computation could save time and computing power when measuring the diversity across a similar sets, such as sites across a given region.

Need for reliable software. Increasing discussion are held regarding scientific software robustness and reliability in ecology (Mislan et al., 2016; Poisot, 2015; White, 2015; Wilson et al., 2017). Mainly because most ecologists are self-trained in programming, these virtuous practices are rarely applied in ecology (Barraquand et al., 2014). For example, unit tests use predefined inputs to compare the software's outputs to expectations (Poisot, 2015). To our knowledge very few R functional diversity packages provide unit tests to assess that the functions behave in the expected manner. Automatic tests of one's code are crucial when developing a tool for the wider audience as it may be used in many different contexts.

What we do here. We here propose a modern alternative to `FD` called `fundiversity` that benefit from modern development practices, needed features for large sized dataset (modularity, parallelization, and memoization), and greater flexibility. The package can be easily extended to accommodate additional diversity indices not covered by following a clear design pattern detailed in the next section.

Main features of `fundiversity`

To ensure the consistency of its functions and to make it easy to use `fundiversity` follows clear design principles. We expose its distinctive features and principles in this section.

Modular. To give a maximum flexibility to the users, we tried to make `fundiversity` as modular as possible. Each function in `fundiversity` computes a single functional diversity index and doesn't require change of arguments to compute more. So that if the user is interested in computing a single index, she only needs to use a single function. All functions in `fundiversity` are prefixed with `fd_` to avoid conflict with similar named functions in other packages, as its becoming standard practice in newer R packages (rOpenSci et al., 2021). In line with its modularity, we focused on making the inputs and outputs of functions coherent. The functions compute functional diversity indices using two main information: a species by traits matrix and a site-species matrix, as such all functions accept these two objects as first arguments. Because the function outputs one diversity value per site the outputs is always structured similarly: one `site` column that contains the name of its site and one column names in function of the computed index (such as `FRic` when computing functional richness). As such, the shape of the output is predictable and easy to be combined with other information.

Parallelization. Parallelization can be an easy way to vastly decrease computation speed why leveraging on the architecture of modern computers. By default almost all functions in `fundiversity` can be

parallelized. `fundiversity` provides parallelization through the `future` backend (Bengtsson, 2020). Parallelization is toggled through a single function call using `future::plan()` before using fundiversity functions. Thanks to the flexibility given by the future backend, the code to use won't change whether parallelizing across several cores on a single computer, across multiple computers, or on a remote high performance cluster. The user has only to update the call to `future::plan()` to distribute her computations on another infrastructure. Furthermore the future backend provides load balancing so that no cores/units stays idle for too long and the parallelized tasks are split evenly.

Memoization. Because functional diversity indices can be computed repeatedly on the same data subset, such as in null models, we can leverage these repeated computations to reuse already computed indices. For example to compute functional richness (FRic) the first step is to compute the convex hull of the input data then the program needs to compute the volume of this convex hull. The first step takes the most time and as such, storing the results of each computed convex hull across a subset can cut the computation time. Avoid recomputing what has already been computed. fundiversity uses this for complex computations such as convex hulls. Memoization means a trade of a little of computer memory (keeping the convex hulls stored) for more computation speed. This default behavior can be overridden through a change in the option `fundiversity.memoise`.

Minimal External Dependencies. To ease the risk of breaking in the long term it is important to minimize the number of dependencies in the package. Identified as a major risk in software and especially scientific software development (REF). Not a real issue for FD but some packages that wrap around FD actual depends on many other package which render them quite brittle for the users after years of not being actively developed.

Agnostic of used input. The functions are all accepting data.frame, matrices, or sparse matrices as input to increase the flexibility given to the user. Big Matrix sparse matrices when can be useful.

Main indices. Because both hillR and betapart packages are great tools to compute hill indices and functional beta-diversity indices, respectively. To avoid reinventing the wheel when it works well, fundiversity mostly contains indices that are not provided by these two packages. fundiversity contains mostly indices of functional alpha diversity (and 1 index of functional beta-diversity). We focused on indices available through the `dbFD()` function in the FD package and on indices that could leverage on faster implementation. fundiversity contains the following alpha functional diversity indices: FRic functional richness, FDis functional dispersion, FDiv functional divergence, FEve functional evenness, and Q Rao's quadratic entropy. Fundiversity also contains a beta-diversity index as it can be quite used to compare functional richness between sites, and no implementation was available at the time it was developed (but see `betapart` newest function).

Case Study

How would fundiversity sit in your analyses? Classical functional diversity workflow:

1. Get traits for your sampling units (individuals, species, communities, etc.)
2. Compute dissimilarity for this. (fundiversity computes euclidean distance by default if all traits are continuous but there are more approriate dissimilarity metrics that can combine both continuous and categorical traits) cite Gower et al. 1981 and Pavoine et al. 2007
3. Actually use fundiversity functions to compute functional diversity indices
4. Analyze results which are always in similar format → tidy format

Show case study with data in fundiversity? Through default data available in the package.

Performance Comparison

In order to track to what extent the additional functionalities provided fundiversity actually helped in cutting the computation time we compared computation time on standardized datasets across similar functions in other packages. We only compared to "original" packages that provide actual functions and

not wrappers that depends on other packages to provide computation of functional diversity indices. We identified Y similar packages. Most indices are included `FD::dbFD()` function but the comparison would be unfair as the function computes many indices in a single call while functions in fundiversity only compute single indices. An updated version of this section can be found through the performance comparison vignette within the fundiversity package with the `vignette("performance", package = "fundiversity")`. Only for Rao's quadratic entropy (and for beta functional richness intersection?).

For test purposes we used datasets of increasing complexity (increasing number of species, increasing number of traits, increasing number of sites, draw performance in a 3D space?).

Plots of comparison between packages

Note on the efficiency of different functions (within fundiversity) with dataset of increasing complexity.

Note on the effect of parallelization on computing speed.

Note on the effect of memoization on computing speed.

Acknowledgements

MG gratefully acknowledges the support of iDiv funded by the German Research Foundation (DFG–FZT 118, 202548816).

Authors' Contributions

MG and HG both conceived the package. MG led the writing of the manuscript. Both authors contributed critically to the drafts and gave final approval for publication.

Data Availability

fundiversity is available on CRAN through `install.packages("fundiversity")` as well as on GitHub at https://github.com/Bisaloo/fundiversity, for archival all releases are available on Zenodo at https://doi.org/10.5281/zenodo The data used in this article are available from the package, through `data(package = "fundiversity")` call.

References

Barraquand, F., Ezard, T. H. G., Jørgensen, P. S., Zimmerman, N., Chamberlain, S., Salguero-Gómez, R., Curran, T. J., & Poisot, T. (2014). Lack of quantitative training among early-career ecologists: A survey of the problem and potential solutions. *PeerJ*, *2*, e285. https://doi.org/10.7717/peerj.285

Bengtsson, H. (2020). *A unifying framework for parallel and distributed processing in r using futures*. https://arxiv.org/abs/2008.00553

Cadotte, M. W., Carscadden, K., & Mirotchnick, N. (2011). Beyond species: Functional diversity and the maintenance of ecological processes and services. *Journal of Applied Ecology*, *48*(5), 1079–1087. https://doi.org/10.1111/j.1365-2664.2011.02048.x

Cornwell, W. K., Schwilk, D. W., & Ackerly, D. D. (2006). A Trait-Based Test for Habitat Filtering: Convex Hull Volume. *Ecology*, *87*(6), 1465–1471. https://doi.org/10.1890/0012-9658(2006)87%5B1465:ATTFHF%5D2.0.CO;2

Díaz, S., & Cabido, M. (2001). Vive la diffã©rence: Plant functional diversity matters to ecosystem processes. *Trends in Ecology & Evolution*, *16*(11), 646–655. https://doi.org/10.1016/S0169-5347(01)02283-2

Farley, S. S., Dawson, A., Goring, S. J., & Williams, J. W. (2018). Situating Ecology as a Big-Data Science: Current Advances, Challenges, and Solutions. *BioScience*, *68*(8), 563–576. https://doi.org/10.1093/biosci/biy068

Gotelli, N. J., & Graves, G. R. (1996). *Null models in ecology*. Smithsonian Institution Press.

Lai, J., Lortie, C. J., Muenchen, R. A., Yang, J., & Ma, K. (2019). Evaluating the popularity of R in ecology. *Ecosphere*, *10*(1), e02567. http://esajournals-onlinelibrary-wiley-com/doi/abs/10.1002/ecs2.2567

Laliberté, E., Legendre, P., & Shipley, B. (2014). *FD: Measuring functional diversity from multiple traits, and other tools for functional ecology.*

Legras, G., Loiseau, N., & Gaertner, J. -C. (2018). Functional richness: Overview of indices and underlying concepts. *Acta Oecologica*, *87*, 34–44. https://doi.org/10.1016/j.actao.2018.02.007

Leps, J., Bello, F., Lavorel, S., & Berman, S. (2006). Quantifying and interpreting functional diversity of natural communities: Practical considerations matter. *Preslia*, *78*, 481–501.

Mason, N. W. H., de Bello, F., Mouillot, D., Pavoine, S., & Dray, S. (2013). A guide for using functional diversity indices to reveal changes in assembly processes along ecological gradients. *Journal of Vegetation Science*, *24*(5), 794–806. https://doi.org/10.1111/jvs.12013

McPherson, J. M., Yeager, L. A., & Baum, J. K. (2018). A simulation tool to scrutinise the behaviour of functional diversity metrics. *Methods in Ecology and Evolution*, *9*(1), 200–206. https://doi.org/10.1111/2041-210X.12855

Mislan, K. A. S., Heer, J. M., & White, E. P. (2016). Elevating The Status of Code in Ecology. *Trends in Ecology & Evolution*, *31*(1), 4–7. https://doi.org/10.1016/j.tree.2015.11.006

Pavoine, S., & Bonsall, M. B. (2011). Measuring biodiversity to explain community assembly: A unified approach. *Biological Reviews*, *86*(4), 792–812. https://doi.org/10.1111/j.1469-185X.2010.00171.x

Poisot, T. (2015). Best publishing practices to improve user confidence in scientific software. *Ideas in Ecology and Evolution*, *8*. https://doi.org/10.4033/iee.2015.8.8.f

R Core Team. (2021). *R: A language and environment for statistical computing* [Manual]. R Foundation for Statistical Computing. https://www.R-project.org/

rOpenSci, Anderson, B., Chamberlain, S., DeCicco, L., Gustavsen, J., Krystalli, A., Lepore, M., Mullen, L., Ram, K., Ross, N., Salmon, M., & Vidoni, M. (2021). *rOpenSci packages: Development, maintenance, and peer review* (Version 0.6.0) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.4554776

Schleuter, D., Daufresne, M., Massol, F., & Argillier, C. (2010). A user's guide to functional diversity indices. *Ecological Monographs*, *80*(3), 469–484. https://doi.org/10.1890/08-2225.1

Villéger, S., Mason, N. W. H., & Mouillot, D. (2008). New Multidimensional Functional Diversity Indices for a Multifaceted Framework in Functional Ecology. *Ecology*, *89*(8), 2290–2301. https://doi.org/10.1890/07-1206.1

White, E. (2015). Some thoughts on best publishing practices for scientific software. *Ideas in Ecology and Evolution*, *8*. https://doi.org/10.4033/iee.2015.8.9.c

Wickham, H., Hester, J., Chang, W., Müller, K., & Cook, D. (2021). *Memoise: Memoisation of functions* [Manual]. https://CRAN.R-project.org/package=memoise

Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L., & Teal, T. K. (2017). Good enough practices in scientific computing. *PLOS Computational Biology*, *13*(6), e1005510. https://doi.org/10.1371/journal.pcbi.1005510

Wüest, R. O., Zimmermann, N. E., Zurell, D., Alexander, J. M., Fritz, S. A., Hof, C., Kreft, H., Normand, S., Cabral, J. S., Szekely, E., Thuiller, W., Wikelski, M., & Karger, D. N. (2020). Macroecology in the age of Big Data – Where to go from here? *Journal of Biogeography*, *47*(1), 1–12. https://doi.org/10.1111/jbi.13633