Received: 28 April 2023 | Revised: 23 November 2023 | Accepted: 5 February 2024

DOI: 10.1111/ddi.13820

Diversity and Distributions WILEY

METHOD

funspace: An R package to build, analyse and plot functional trait spaces

Carlos P. Carmona[1] | Nicola Pavanetto[2] | Giacomo Puglielli[3]

[1]Institute of Ecology and Earth Sciences, University of Tartu, Tartu, Estonia

[2]Institute of Agricultural and Environmental Sciences, Estonian University of Life Sciences, Tartu, Estonia

[3]Departamento de Biología Vegetal y Ecología, Facultad de Biología, Universidad de Sevilla, Sevilla, Spain

Correspondence
Giacomo Puglielli, Departamento de Biología Vegetal y Ecología, Facultad de Biología, Universidad de Sevilla, Apartado 1095, 41080 Sevilla, Spain.
Email: gpuglielli@us.es

Funding information
MCIN/AEI/10.13039/501100011033; FEDER "ESF Investing in your future", Grant/Award Number: PID2021-122214NA-I00; European Regional Development Fund, Grant/Award Number: MOBERC40 and MOBERC100; Eesti Teadusagentuur, Grant/Award Number: PSG293 and PRG2142; MCIN/AEI/10.13039/501100011033, Grant/Award Number: IJC2020-043331-I

Editor: Rob Cooke

Abstract

Aim: Functional trait space analyses are pivotal to describe and compare organisms' functional diversity across the tree of life. Yet, there is no single application that streamlines the many sometimes-troublesome steps needed to build and analyse functional trait spaces.

Innovation: To fill this gap, we propose `funspace`, an R package to easily handle bivariate and multivariate functional trait space analyses. The six functions that constitute the package can be grouped in three modules: 'Building and exploring', 'Mapping' and 'Plotting'. The building and exploring module defines the main features of a functional trait space (e.g. functional diversity metrics) by leveraging kernel density-based methods. The mapping module uses general additive models to map how a target variable distributes within a trait space. The plotting module provides many options for creating flexible and publication-ready figures representing the outputs obtained from previous modules. We provide a worked example to demonstrate a complete `funspace` workflow.

Main Conclusions: `funspace` will provide researchers working with functional traits across the tree of life with a new tool to easily explore: (i) the main features of any functional trait space, (ii) the relationship between a functional trait space and any other biological or non-biological factor that might contribute to shaping species' functional diversity.

KEYWORDS
data imputation, functional diversity, functional traits, general additive models, kernel density, principal component analysis, trait space

1 | INTRODUCTION

Traits are defined as characteristics of individual organisms that can be measured at any relevant level of biological organization (Dawson et al., 2021). They reflect species adaptations to the main selective forces in their natural habitats, and therefore have a (more or less direct) link with species' performance – that is, functional traits – and/or ecosystem processes (de Bello et al., 2021). Thus, functional traits represent an essential tool for ecologists and ecophysiologists worldwide to describe the diversity of organisms' form and function in relation to any aspect that might influence a species' biology.

Classical studies have hypothesized that species adaptations to the environment might have evolved around few theoretical strategic axes (Greenslade, 1983; Grime, 1977; Pianka, 1970; Westoby, 1998).

Subsequent research has attempted to characterize such axes using functional traits for fish (Winemiller & Rose, 1992), corals (Darling et al., 2012), and vascular plants (e.g. Diaz et al., 2004), among other groups of organisms. The implication of having a handful of axes describing species' strategies implies that the diversity of organisms' form and function must be constrained by trade-offs among traits (Pianka, 1970 for animals; Greenslade, 1983 for insects; Westoby et al., 2002 for vascular plants). Nowadays, we know that pervasive trade-offs between traits constrain organisms' form and function along (relatively) few trait dimensions that are often independent, thus limiting possible trait combinations within two – or highly dimensional spaces, also called functional trait spaces, across the tree of life (Bueno et al., 2023; Carmona, Bueno, et al., 2021; Carmona, Tamme, et al., 2021; Chelli et al., 2024; Díaz et al., 2016; Junker et al., 2023; Mouillot et al., 2021; Westoby et al., 2021; Winemiller et al., 2015). Expressing the diversity of organisms' form and function within functional trait spaces of a reduced number of dimensions allows for robust generalization and quantification of species' strategic axes of trait variation (see Mouillot et al., 2021). Thus, functional trait spaces represent indispensable tools to describe organisms' functional diversity across the tree of life.

Different approaches allow building multivariate functional trait spaces. The most common quantitative tool is Principal Component Analysis (PCA), as it allows reducing a trait dataset to few independent trait dimensions defined by the inherent relationships between traits. However, other approaches for dimensionality reduction that allow considering both quantitative and categorical trait data, such as Principal Coordinates Analysis (PCoA) or nonmetric multidimensional scaling (NMDS) are also widely used. Recent methodological advances have permitted to explore additional features of functional trait spaces besides the trait dimensions that define them. In particular, kernel density analyses (Carmona et al., 2016, 2019; Duong, 2007) have revealed that species occupy functional trait spaces differentially, resulting in species clumping around some trait combinations that are much more frequent than others (Carmona, Bueno, et al., 2021; Díaz et al., 2016; Puglielli et al., 2021 for vascular plants; Carmona, Tamme, et al., 2021; Cox et al., 2021; Toussaint et al., 2021 for different animal groups). As a result, kernel density estimates are nowadays widely employed to estimate different properties about the way in which different species and groups of species occupy a functional trait space, including aspects such as the amount of space occupied (functional richness) or the degree to which species in a group have different trait combinations (functional divergence), among other indices of functional diversity (see Mammola et al., 2021 for a review).

A first pitfall arising when using PCA (but also distance-based ordinations approaches depending on the dissimilarity metric considered) is that missing values are not allowed in the dataset. This is especially the case of large-scale analyses, where the unprecedented trait availability provided by global databases contrasts with the use of disparate trait frameworks across studies and disciplines, resulting in large databases including many traits but with missing information for many of the species (e.g. the TRY plant trait database, Kattge et al., 2020). This has pushed research towards developing imputation methods (e.g. Penone et al., 2014; Schrodt et al., 2015; Stekhoven & Bühlmann, 2012) that mostly use machine-learning techniques to impute missing trait values based on trait–trait correlations sometimes improved by accounting for species' phylogenetic information (Penone et al., 2014). A key point to consider is that missing trait information is not distributed randomly within datasets. For example, across mammals, big species with large range areas are generally better informed than small species or species with small ranges (González-Suárez et al., 2012); similar biases have been described for plants (Carmona, Bueno, et al., 2021; Sandel et al., 2015). Recent research is increasingly suggesting that imputation can largely correct these biases, so that functional diversity patterns inferred from imputed datasets are much closer to the real ones than those that would be estimated using only species with complete trait information (Penone et al., 2014; Stewart et al., 2023).

Even when there are no missing data, another pitfall is deciding how many dimensions to retain to maximize the information contained in the dataset while minimizing information redundancy. In the case of PCA, this is usually done by retaining the first two Principal Components (PCs) since they capture most of the variance in a dataset. However, if the intrinsic dimensionality of the considered trait dataset is higher, this approach might lead to the loss of biologically relevant information (Laughlin, 2014). Several methods have been proposed to determine the number dimensions (Mouillot et al., 2021; Peres-Neto et al., 2005). In the context of PCA, the parallel analysis method proposed by Horn (1965) is one of the most precise methods in identifying the correct number of PCs to retain (Peres-Neto et al., 2005), and one of the most widely used. This method contrasts eigenvalues produced through a PCA on *n* random data sets of uncorrelated variables with the same dimension of the original dataset to produce eigenvalues for components that are adjusted for the sample error-induced inflation. Apart from some large cross-species studies (e.g. Carmona, Bueno, et al., 2021; Carmona, Tamme, et al., 2021; Díaz et al., 2016; Guillemot et al., 2022; Toussaint et al., 2021), the two described pitfalls of using PCA are hardly considered when building functional trait spaces. We argue that this might mostly depend on lack of knowledge of the existence of specific procedures, and likely, on the potential difficulties when implementing them. However, the most common practices when building functional trait spaces remain choosing the first two PCs and/or dropping observations with missing values, with the inevitable loss of biological information.

Another aspect that deserves attention is how to use functional trait spaces as a ground to test the relationship between species' trait strategies and additional variables. This includes analyses such as exploring the relationship between species' ecological strategies (i.e. their position in the functional trait space) and climatic features at a species' habitat, or to explore how extinction risk (Carmona, Tamme, et al., 2021), as well as any other adaptive syndrome (Chelli et al., 2024; Pavanetto et al., 2023; Puglielli et al., 2022), is related to species traits. However, analysing these multidimensional relationships is inherently complex (Villéger et al., 2011). One layer of

complexity is provided by the need to use the multiple dimensions defining the functional trait space as predictors in models attempting to link multidimensional species strategies to other ecological dimensions (e.g. climate). Another problem is that more than often we do not have any previous knowledge on the functional form (e.g. linearity) of the relationship linking the target dimension to species' strategies. In addition, visualizing such relationships is not straightforward, and we often rely on separately analysing relationships between response variables and single trait dimensions (e.g. a single PC). These difficulties can be overcome by considering functional trait spaces as a set of coordinates that we can use to build maps of any response variable of interest. General additive models (GAMs), used to estimate smooth functional relationships between predictor variables and a response (Pedersen et al., 2019), provide a solution to do that. Accordingly, they have been consistently used for fitting and mapping, for instance, species distribution models' predictions in spaces defined by geographical coordinates (e.g. Naimi & Araújo, 2016). GAMs allow expressing the models' predictor as a multidimensional smoother, allowing to use the functional trait space dimensions as a single predictor (see Carmona, Tamme, et al., 2021; Chelli et al., 2024; Pavanetto et al., 2023; Puglielli et al., 2022 for examples). In the case of functional trait spaces, GAMs provide a sound and flexible modelling solution because they operate using piecewise functions adapting to the local conditions within the functional trait space, so that GAMs behaviour in a particular portion of the data point cloud does not overall alter the global behaviour of the model (Pedersen et al., 2019; Venables & Dichmont, 2004). While this comes at the cost of the interpretability of the coefficients, spline regressions are more easily interpreted from a graphical point of view rather than through the values of their coefficients (Venables & Dichmont, 2004). Thus, GAMs become particularly useful to visualize and test patterns of how target variables vary within functional trait spaces.

Despite functional trait spaces are widely used in disparate research fields spanning ecology, plant science, animal ecology, evolutionary ecology – for example, searching the term 'trait space' in Web of Science returned 11,440 documents – there is no single R package for building, exploring, mapping and plotting functional trait spaces. To fill this gap, we propose **funspace**, an R package to handle functional trait space analyses using bivariate or multivariate trait data. Due to the very little difference between using **funspace** with pairs of traits or considering a multivariate ordination as input (Figure 1), here we present the package functionalities in the context of PCA-based analyses, and we refer to **funspace** documentation and examples for other cases, including raw trait data or other ordination methods (e.g. NMDS). The package consists of three main interconnected modules (Figure 1; Table 1):

1. *Building and exploring module*: it consists of multiple functions to build the functional trait space from bivariate or multivariate input data using multivariate kernel density methods, and to analyse the main features of a functional trait space (e.g. functional diversity indexes, testing against null models). If specified, all these analyses can be iterated across levels of a grouping variable (e.g. within groups such as families or populations, etc.).
2. *Mapping module*: it consists of one function that uses GAMs to statistically test for the link between a target variable at the species level (such as extinction risk) and the position of species within the functional trait space. The function handles multiple groups as well.
3. *Plotting module*: it consists of a generic plot function that can receive as input the objects built in either of the previous modules. In case the input is the object built in module 1 (building and exploring), the function prints a bivariate (or pairs of dimensions in case of multivariate) functional trait space displaying bivariate kernel density estimates (for single or multiple groups). If the input is an object built in module 2 (mapping module), then the function prints a heatmap depicting how a target variable is distributed within the functional trait space (for single or multiple groups). All plots can be customized and new features can readily be added after plotting.

2 | MODULE 1: BUILDING AND EXPLORING A FUNCTIONAL TRAIT SPACE

funspace includes three functions to build and explore a functional trait space: funspaceDim(), impute(), and funspace() (Table 1; Figure 1). Here we only illustrate the main features of the functions when the input is a PCA object and refer to the **funspace** documentation for full details on the functions' arguments and input data. However, note that the same exact procedures can be applied to any bivariate trait data (i.e. pairs of raw traits, or scores of species on a different type of ordination). An example of how to use the functions constituting module 1 is reported in Box 1.

`funspaceDim()` allows the user to identify the number of dimensions that are needed to build a functional trait space. The identified dimensions are those that minimize redundancy while maximizing the information contained in the trait data based on the method proposed by Horn (1965) (see Introduction). Briefly, this method contrasts the eigenvalues produced through PCAs run on (30*(number of variables)) random datasets with the same number of variables and observations as the input dataset. Eigenvalues >1 are retained in the adjustment. `funspaceDim()` implements the Horn test via the `paran()` function of the R package `paran` (Dinno, 2018). A trait matrix is the only input needed to run the function. `funspaceDim()` returns the number of dimensions to be retained from the subsequent PCA (Figure 1) and prints this information in the R console. Note that `funspaceDim()` can be used only when the user wants to pass a PCA object as argument of `funspace()`. In the case of PCoA as the input object, the dimensions are iteratively identified by the analysis and depend on the choice of the distance measure used (e.g. Euclidean distance) together with the number of input variables. For NMDS, the user can specify the number of dimensions (`k` argument of `vegan::monoMDS()`, see

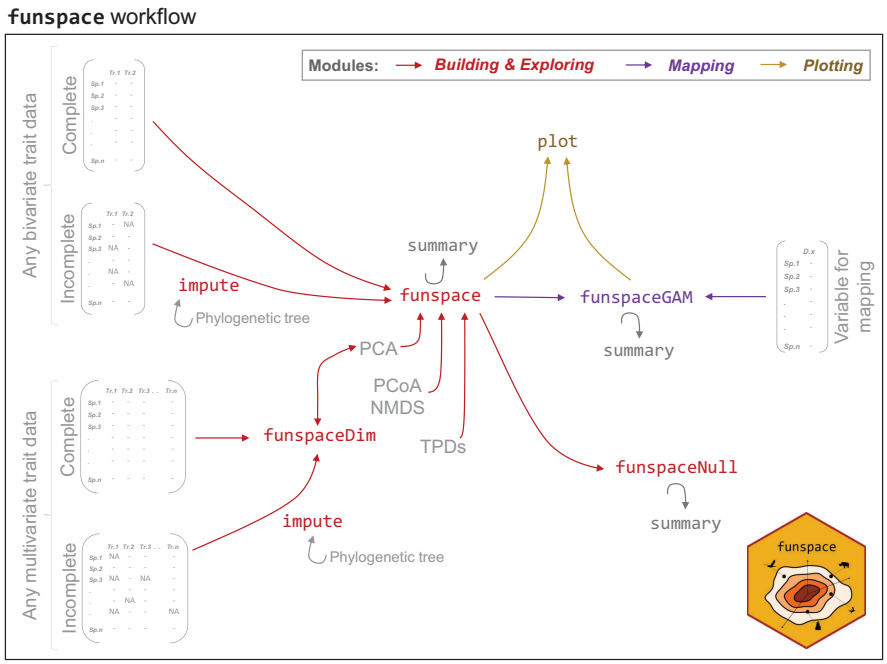

FIGURE 1 `funspace` workflow by package module. Functions classification according to package modules is highlighted by colour coding. Input data are shown in grey. In case the input trait matrix has missing trait data, the user might want to gap-fill the bivariate or multivariate trait matrix using the `impute()` function, with the option to account for phylogenetic information in the imputation process. The second step is running an ordination (PCA, PCoA, NMDS) before using the other **`funspace`** functions. In the case of a PCA object, the user can select the number of principal components to be retained in the analysis, by using a the multivariate trait matrix to `funspaceDim()`. The double-headed arrows indicate that the dimensions retained using `fuspaceDim()` are those that should be extracted from the subsequent PCA as well as those used to build the trait space using `funspace()`. In cases where the user decides to identify the trait space dimensions using `funspaceDim()`, and to pass the target PCs to `funspace()` instead of the PCA object, at this point the workflow follows the same path as that of any bivariate trait matrix. That is, the user can directly use a bivariate trait matrix (including coordinates obtained from any ordination method) to run `funspace()`. Because **`funspace`** is strongly focused in providing graphical representations of functional spaces, in cases when the functional space has higher dimensionalities, two dimensions at a time have to be selected within the `funspace()` function. Finally, an object `TPD::TPDs()` or `TPD::TPDMeans()` can be passed as argument of the `funspace()` function, and we refer to Carmona et al. (2016, 2019) for building TPDs objects. Once the `funspace()` object has been defined, the user can plot the output (i.e. bivariate functional trait space with kernel density estimates) using the `plot()` function. In case the user wants to test the observed functional trait space against a multivariate normal or a uniform distribution, the `funspace()` object can be passed to `funspaceNull()`. `funspaceGAM()` allows assessing how an additional target variable (e.g. temperature, extinction risk, an additional trait non included in the trait space, etc.) is distributed within the functional trait space previously obtained with `funspace()`. The resulting map of the target variable within the functional space can then be represented graphically as a heatmap using `plot()`. All the modules can handle multiple groups. The package also includes a generic `summary()` function that can be used to print the output of `funspace()`, `funspaceNull()`, and `funspaceGAM()`. Function descriptions are reported in Table 1. A full worked example is shown in Boxes 1–3.

below) before running the analysis, and the best value of k needs to be determined using the model stress, a measure of the model's goodness of fit obtained by comparing distances in the original dissimilarity matrix to the fitted one in the ordination space.

If the input trait matrix (either bivariate or multivariate) contains missing data, the user can fill the gaps using the `impute()` function. By default, `impute()` fills the gaps in the dataset using information on trait–trait relationships using random forest models as implemented in the `missForest` R package (Stekhoven & Bühlmann, 2012). Optionally, a phylogenetic tree (an object of class `phylo`) can be included to improve the imputation procedure by accounting for species phylogenetic relatedness together with the information on trait–trait relationships (Penone et al., 2014). Briefly, a given number of phylogenetic eigenvectors derived from the tree (specified by the argument `nEigen`, default is 10) are added to the original trait matrix, and the resulting dataset is then used in the above-mentioned random forest-based procedure. The `addingSpecies` argument allows the user to add to the phylogeny those species that are only present in the trait matrix. If `TRUE` (default is `FALSE`), the `phytools::add.species.to.genus()` function (Revell, 2012) is used to add species to the root of the genus if there are congeneric species in the tree. Those species that cannot be added to the phylogeny are imputed without taking phylogenetic information into account. Users who want to make use of the multiple options available in `add.species.to.genus()` should modify their phylogenetic tree beforehand. The `impute()` output is a list containing the imputed version of the trait matrix and its non-imputed counterpart.

After having defined the number of dimensions that define the functional trait space, and having ran a PCA, PCoA or NMDS, the user can build the functional trait space using `funspace()`, the

TABLE 1 `funspace` functions by module.

Module	Function	Description
Building & exploring	`funspaceDim()`	Identifies the number of dimensions that are needed to build a functional trait space.
	`impute()`	Imputes missing data in a trait matrix using trait–trait relationships and, if specified, phylogenetic information.
	`funspace()`	Builds a functional trait space including multivariate kernel density at different quantiles. Multi-group option available.
	`funspaceNull()`	Tests for statistical difference between the observed functional trait space area versus theoretical areas generated using either a multivariate normal or a uniform distribution.
Mapping	`funspaceGAM()`	Fits GAMs using a target variable as the response variable and functional trait space axes as a two-dimensional smoother predictor. Returns GAM model predictions within a functional trait space. Multi-group option available.
Plotting	`plot()`	If a `funspace()` object is passed as argument, the function prints a functional trait space with multivariate kernel density estimates. If a `funspaceGAM()` object is passed as argument, the function returns a GAM map (i.e. heatmap) of how a target variable distributes within the functional trait space. Multi-group option available.
Building & exploring and mapping	`summary()`	Gives a summary of the main features and results of objects created with the `funspace()`, `funspaceNull()`, and `funspaceGAM()` functions.

core function of the package. By default, the function needs only one object to run: either an ordination object (either a PCA obtained using the `base::princomp()` function, or a PCoA obtained using the `vegan::capscale()` function, or a NMDS obtained using the `vegan::monoMDS()` or `vegan::metaMDS()` functions), or a TPDs object obtained with the `TPD::TPDs()` or the `TPD::TPDsMean()` functions, or a data frame including species coordinates in any other type of ordination (as well as any multivariate trait data) (Figure 1). By default, `funspace()` uses the first two dimensions of the provided space, but the user can plot any pairs of dimensions by modifying the `PCs` argument. We recommend users to set the pairs of PCs based on the output of `funspaceDim()`. With this input, `funspace()` estimates the probability of occurrence of trait combinations within the space defined by pairs of dimensions using kernel density estimation with unconstrained bandwidth selectors combining the functionalities available in the R packages ks (Duong, 2007) and TPD (Carmona et al., 2019). The grid size for computing multivariate kernel density estimates is defined by the `n_divisions` argument that sets the total number of divisions of each dimension or trait; since the resulting functional space is bidimensional, the total number of cells in which the functional trait space is divided is equal to `n_divisions`. Note that increasing `n_divisions` results in longer time for computing the functional trait space (this can be an issue particularly when the `funspaceNull()` is used), but it increases the smoothness of the space edges in subsequent plotting (see Module 3). When the input is an ordination object, `funspace()` uses by default the range of the selected axes to build the grid limits in which estimating the multivariate kernel density estimates, but user-defined ranges can be used for calculations as well. The probability threshold used for multivariate kernel density estimates (i.e. the boundaries of the trait probability density function) can be set using the `threshold` argument (default = 0.999 corresponding 99.9% of the trait probability density function being retained). If the dataset includes a categorical variable (for example containing the family to which each species belongs), the same procedure can be applied to each level of the grouping variable by specifying it in the `group.vec` argument. In this case, by default, `funspace()` constraints group bandwidths for plotting to the global bandwidth, to avoid kernel density estimates of each group to exceed the borders of the functional trait space defined using the whole dataset. However, group-specific bandwidths can be generated by specifying `fixed.bw = FALSE`.

Other than the information that is later used for plotting, and when the input data is a PCA object, `funspace()` also returns a table reporting the loadings of the traits in the provided PCA (eigenvectors multiplied by the square root of eigenvalues) (Table 2). In this sense, it is important to note that the 'loadings' returned by `base::princomp()` are actually eigenvectors, which do not provide information about the amount of variance contained within each PC. The squared loadings are used to estimate the proportion of the original variance of each trait that is explained by each selected component (specified in the `PCs` argument) as well as by the two selected components (Table 2, Box 1). This information can help understanding how well each trait is represented by the selected functional space. In addition, and independently of the data input (ordination object, TPDs object, or matrix), `funspace()` always returns a table reporting the functional richness and functional divergence of the dataset (Mason et al., 2005) (Table 3, Box 1). Functional richness represents the amount of functional trait space (i.e. the area in a two-dimensional context) occupied by the set of species. Functional divergence quantifies how much the TPD function is distributed towards the edges of the functional trait space occupied by an assemblage. These indexes are calculated using the trait probability density approach (Carmona et al., 2016, 2019) across the whole dataset, and, if specified, per each level of a grouping variable. The user can retrieve all the `funspace()` output(s) using the `summary()` function (Box 1).

Finally, the function `funspaceNull()` allows the user to test how the area (i.e. functional richness) of the observed functional trait space differs from that of a null model. Two null models are implemented. (i)

BOX 1 Using `funspace` module 1

We use one of the **`funspace`** example dataset to illustrate the package workflow. The dataset is a subset of Carmona, Bueno, et al. (2021) dataset, containing trait information for the traits defining the Global Spectrum of Plant Form and Function (GSPFF, Díaz et al., 2016, namely plant height, seed mass, specific stem density, leaf area, leaf nitrogen content on a mass basis and specific leaf area) for >10,000 vascular plant species. Trait data was compiled from the TRY database (Kattge et al., 2020).

```
# Load the package
library(funspace)
# Load example data (traits already log10-transformed and scaled)
data.aux <- funspace::GSPFF_missing
```

The dataset contains missing trait information, so we use the `impute()` function, coupled with phylogenetic information to impute missing trait data. The phylogenetic tree was retrieved using the `V.Phylomaler` R package (Jin & Qian, 2019) and is contained in the `phylo` object that is loaded with **`funspace`**. Note that, when using user-specified data, the trait dataset provided must include species names as row names; the same naming convention should be used in the phylogenetic tree (if provided).

```
# Load the example tree (it must be an object of class phylo)
phylo.tree <- funspace::phylo
# Imputing missing data (Because there are >10k species, it takes long to run!)
GSPFF_imputed <- impute(traits=data.aux, phylo=phylo.tree, addingSpecies=TRUE) # Output is a list
# Save the imputed trait data in a separate object:
imputed.traits <- GSPFF_imputed$imputed
# Testing the number of dimensions defining the functional trait space
funspaceDim(imputed.traits) # two dimensions retained
# Run PCA
pca.trait <- princomp(imputed.traits, cor=TRUE)
# Building the functional trait space (using the first two PCs)
trait_space_global <- funspace(x=pca.trait, PCs=c(1,2), n_divisions=300)
```

`funspace` example datasets also include a data frame with taxonomic information for the species in our GSPFF subset. We will use this information to illustrate how to deal with groups when using **`funspace`** Module 1.

```
# Loading the taxonomic information
tax_inf <- funspace::GSPFF_missing_tax
# Defining a group to include four major families
selFam <- c("Pinaceae", "Poaceae", "Fabaceae", "Lauraceae")
selRows <- which(tax_inf$family %in% selFam)
tax_subset <- droplevels(tax_inf[selRows,])
# Creating a subset of the GSPFF to retain target groups
GSPFF_subset <- imputed.traits[selRows, ]
# Run PCA on the subset
PCA_subset <- princomp(GSPFF_subset)
```

A grouping variable can be specified in `funspace()` by passing a vector including the variable to the `group.vec` argument.

```
# Building the functional trait space (using the first two PCs) including groups
trait_space_families <- funspace(x=PCA_subset, PCs=c(1,2), group.vec=tax_subset$family,
n_divisions=300)
# We can print the outputs for both spaces:
```

BOX 1 (Continued)

```
summary(trait_space_global)

summary(trait_space_families)
```

In this example, the `summary()` function applied to the object `trait_space_global` returns the proportion of variance explained for each trait by each of the selected components of the PCA (Table 2) and the functional diversity indexes describing the global space (Table 3). In the multi-group case (i.e. `trait_space_families` object), functional indexes are returned per each level of the grouping variable as well (Table 3). Note that functional diversity indexes (for single or multiple groups) are returned independently of the `funspace()` input.

TABLE 2 PCA output returned by `funspace()`.

Trait	Comp.1	Comp.2	Overall_ explained
Leaf area	25.21	41.49	66.70
Leaf nitrogen content	8.92	60.42	69.34
Plant height	79.94	5.72	85.66
Specific leaf area	25.24	59.93	79.17
Specific stem density	78.51	2.46	80.97
Seed mass	66.26	6.20	72.46

Note: A table summarizing the percentage of variance of each trait that is explained by each principal component (Comp.1 and Comp.2 in this example) and across components (Overall_explained). The table in this example is returned as part of the output of `summary(trait_space_global)`, corresponding to the global spectrum of plant form and function using imputed trait information (see Module 1 and Box 1). From this output we can see that the first principal component is mainly explaining plant height, specific stem density and seed mass, whereas the second component is more strongly related to leaf traits (leaf area, leaf nitrogen content and specific leaf area). Finally, the `Overall_explained` column reports the percentage of variance explained by the considered space (i.e. the sum of the variance explained by individual components). In this functional space, the quality of the representation of plant height, specific stem density and specific leaf area is better than that of leaf area and leaf nitrogen content. Note that the function output is slightly different since the table has been reorganized for presentation purposes.

TABLE 3 Functional diversity indexes returned by `funspace()`.

Set of species	Threshold	Functional richness	Functional divergence
Global	99.90%	45.55	0.56
Fabaceae	99.90%	39.92	0.58
Lauraceae	99.90%	17.13	0.46
Pinaceae	99.90%	13.19	0.39
Poaceae	99.90%	28.50	0.40

Note: A table summarizing two functional indexes (functional richness and divergence, see Module 1) for the global set of species and for each group contained in the `trait_space_families` example (see Module 1 and Box 1). The quantile threshold at which the indexes are calculated is also returned in the output. The table is returned as part of the `summary()` output. In this example we can see that functional richness and functional divergence are larger for the Fabaceae family than for the other families, reflecting the fact that legumes include both herbaceous and tree species (see Figure 3), whereas species within the other families are generally more similar between them. Note that the function output is slightly different since the table has been reorganized for presentation purposes.

The first null model generates data following a bivariate normal distribution (`null.distribution = 'multnorm'`) with the same mean and variance–covariance structure as the original data. This null model corresponds to the expectation that some trait combinations, specifically those towards the centre of the space, are more likely than others, and the resulting space has the approximate shape of an ellipse (Carmona, Bueno, et al., 2021). (ii) Another null model generates a dataset with variables following a uniform distribution (`null.distribution = 'uniform'`), creating a functional trait space where all trait combinations within the range of the observed functional trait space axes are equally possible, and the space assumes the approximate shape of a rectangle (Díaz et al., 2016). Given a `funspace()` object as input, `funspaceNull()` generates a vector of null model areas across a user-defined number of iterations. Then, using the function `as.randtest` from the `ade4` R package (Dray & Dufour, 2007), it tests for the statistical difference between the observed value of functional richness and the null expectation (considering in both cases the probability threshold value used in the `funspace()` object input). Hypothesis testing options follow the `as.randtest()` specifications. To get meaningful comparisons with the original functional trait space, `funspaceNull()` builds the trait probability density functions at each iteration using the same grid size and bandwidth for computing kernel density estimates as the `funspace()` object. `funspaceNull()` returns: the observed functional richness, the average functional richness of the null model across iterations, and the *p*-value and standardized effect size associated to the hypothesis test. This information can be retrieved using the `summary()` function. Note that `funspaceNull()` is meant to test the global space against null models, so it does not handle multiple groups.

3 | MODULE 2: MAPPING A FUNCTIONAL TRAIT SPACE

funspace includes the function `funspaceGAM()` that automatizes GAM modelling steps needed for mapping patterns of a target

dimension within the functional trait space (Table 1, Figure 1). An example of how to use `funspaceGAM()` is reported in Box 2.

`funspaceGAM()` fits a GAM with a single response variable (defined by the `y` argument) and a two-dimensional smoother defined by the functional trait space axes as the bivariate predictor. GAM specifications are set to the default ones (see `mgcv::gam()` function, Wood, 2017), only the `family` argument can be specified by the user. Functional trait space axes are inherited by the `funspace()` object specified in the `funspace` argument of `funspaceGAM()`. Given a `funspace()` object, the function automatically defines the functional trait space boundaries and grid size for generating GAM model predictions by inheriting the `threshold` and the grid size information from the `funspace()` object. Note that, the greater the grid size is, the longer the GAM computational time will be.

If a grouping variable is specified when running `funspace()` (see Module 1, Box 1), GAMs are also fitted per each level of the grouping variable and only within the portion of the space occupied by the data points of each subgroup (see details of `fixed_bw` argument of `funspace()` for different ways of estimating the bandwidth for each group). To avoid fitting models with not enough data, if there are fewer observations in a particular group than specified in the argument `minObs` (defaults to 30), the model is not fitted for that group, and a warning message is returned to inform the user. GAM summary statistics for all data pooled or, if specified, for each level of the specified grouping variable, can be retrieved using the `summary()` function. The output is not shown in the example below because the summary is equivalent to that of the function `mgcv::summary()`.

BOX 2 Using `funspace` mapping module

For this example, we will consider the GSPFF dataset that contains information on six traits for the set of species with complete trait information. We will create a response variable that increases as we move further from the origin of coordinates of the space and add some random normally distributed noise to avoid a perfect fitting.

```
# Creating the GSPFF functional space:
pca.gspff <- princomp(GSPFF, cor=TRUE)
trait_space_gspff <- funspace(x=pca.gspff, PCs=c(1,2), n_divisions=300)
# Response variable we want to map
y <- abs(pca.gspff$scores[, 1]*pca.gspff$scores[, 2])+rnorm(nrow(GSPFF), 0, 1)
# Fitting GAM for all data pooled (It takes few seconds to run!)
fit.gam<- funspaceGAM(y=y, funspace=trait_space_gspff)
# To fit GAMs for each level of the grouping variable (not shown), we just have to run funspaceGAM() by imputing a funspace() object that was built by specifying the group.vec argument (e.g. the object 'trait_space_families' in Box 1).
```

4 | MODULE 3: PLOTTING A FUNCTIONAL TRAIT SPACE

funspace includes a generic `plot()` function that allows the user to plot objects of the funspace class created with the `funspace()` and `funspaceGAM()` functions (see Box 3 for a worked example).

By default, the only argument needed for plotting is an object of class `funspace` that is assigned to the `x` argument of the `plot()` function. If the plotting input was created with the `funspace()` function, the plotting function returns a functional trait space where coloured areas represent the density of occupation of the functional space by the considered species (i.e. the trait probability density function as described in Carmona et al., 2016, 2019; Figure 2a). When the input is the result of the `funspaceGAM()` function, the `plot()` function prints a heatmap in the functional space depicting the predicted values for the response variable within the external boundaries of the trait probability density function (Figure 2b). Note that, in both cases, the resulting plot(s) might appear more or less smooth depending on the number of `n_divisions` set when building the functional trait space using the `funspace()` function (Box 1). Larger values of `n_divisions` will result in smoother plots, but also in higher computational times. An important argument of the `plot()` function is the `type` argument. By default, this argument is set to `'global'`, and it allows the user to plot the functional trait space for all data pooled (Figure 2a,b). However, if `type` is set to `'groups'`, and given that a grouping vector has been already specified when building the `funspace()` object (Box 1), the `plot()` function creates a plot for each level of the grouping variable (Figure 3, see the `fixed_bw` argument of `funspace()`).

Many graphical parameters can be set for plotting. For example, if the input object comes from `funspace()`, when the argument `quant.plot` is set to `TRUE` (default is `FALSE`), contour lines are drawn at quantiles of the trait probability density function specified by the argument `quant` (by default these are the `threshold` argument that was used in `funspace()` and the 0.50 and 0.25 quantiles). If `quant.plot` is set to `TRUE` when the input comes from the `funspaceGAM()` function, then the contour lines indicate the quantiles of the values of the response variable predicted by the GAM model. When quantile lines are displayed, their features (e.g. type, colour) can be set using specific graphical parameters (e.g. `quant.lty`, `quant.col`). Independently of the input object, the gradient palette (with a number of colours specified by the `ncolors` argument, which defaults to 100) can be easily modified by specifying a vector including the two or more colours in the `colours` argument. Plot axes limits can be adjusted by modifying the base plot graphical parameters `xlim` and `ylim`. When the input used to create the `funspace()` object is a PCA, the user can also decide whether to plot arrows representing the trait loadings in the PCA by setting the argument `arrows` to `TRUE`

BOX 3 Using `funspace` plotting module

For this example, we use the `trait_space_families` and `fit.gam` objects defined in Boxes 1 and 2 and generated using the `funspace()` and `funspaceGAM()` function, respectively.

```
# 1 – Plotting a functional space including a global (all data pooled) trait probability distribution. In this case, we will plot the trait_space_families object.
plot(x=trait_space_families, # funspace object
type="global", # plot the global TPD
quant.plot=TRUE, # add quantile lines
arrows=TRUE, # add arrows for PCA loadings
arrows.length=0.9) # make arrows a bit shorter than the default.
```

The output is shown in Figure 2a.

```
# Plotting an object created with the funspaceGAM() function: we will use the fit.gam object built in Box 2.
plot(x=fit.gam,
type="global",
quant.plot=TRUE,
quant.col="grey80") # a lighter tone for the quantiles
```

The output is shown in Figure 2b.

```
# To display the use of the 'type' argument, we also plot the functional trait space with an individual trait probability density function per each level of the grouping variable (Fabaceae, Lauraceae, Pinceae, Poaceae). The GAM output per each group is not shown, but it could be obtained replacing the trait_space_families object with the fit.gam one in the following code.
par(mfrow=c(2, 2), mar=c(2,2,1,1), mgp=c(1, 0.1, 0))
plot(x=trait_space_families,
type="groups", # a plot for each group (family)
quant.plot=TRUE,
globalContour=T, # The contour of the global TPD
pnt=T, # add points for species of each family
pnt.cex=0.1, # points should be small
pnt.col=rgb(0.2, 0.8, 0.1, alpha=0.2), # colour for points
axis.title.line=1)
```

The output is shown in Figure 3.

(default is `FALSE`). Full details on plotting options (e.g. points display, etc.) can be found in the **`funspace`** documentation.

5 | DISCUSSION

funspace provides users with an important tool for conducting functional trait-space analyses from raw trait data and a way to increase the reproducibility of such analyses. The interconnected modules of the package provide in fact all the necessary steps to streamline sometimes troublesome operations, including imputation of missing data (Stewart et al., 2023) or the definition and analysis of functional spaces using multivariate kernel density (Mammola et al., 2021). **`funspace`** is explicitly based on the TPD framework (Carmona et al., 2016, 2019), which is not optimized for plotting. Thus, **`funspace`** extends the TPD functionalities in terms of plotting, and if the functions are defined within the same trait range (by setting the trait_range argument), the results of the two packages are fully comparable. This allows integrating funspace() outputs with additional analyses not included in the package, such as TPD-based dissimilarity, redundancy, or uniqueness (Carmona et al., 2016). In particular, the main features of **`funspace`** include: (i) it requires minimum input from users, increasing its scope of application; (ii) it is optimized for graphical representation and visual exploration of functional trait spaces. In the following, we discuss how **`funspace`** integrates and expands the current landscape of functional diversity R packages.

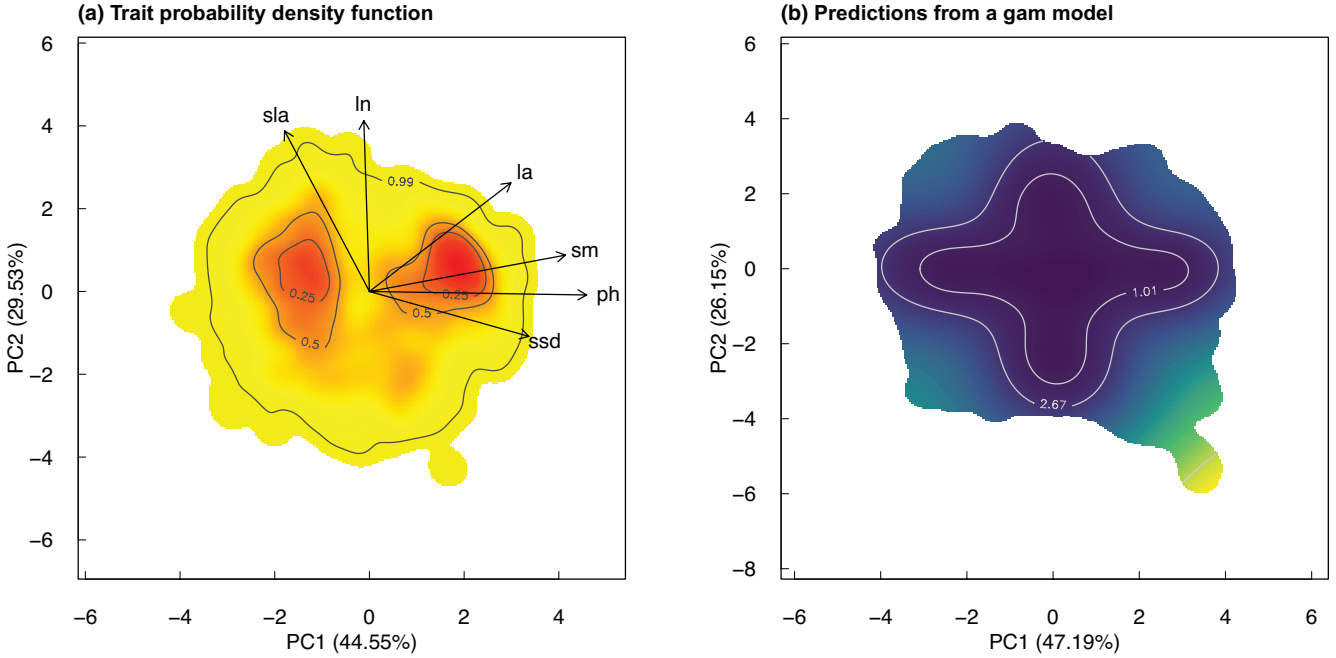

FIGURE 2 `plot()` output by input object. (a) Plot of an object created with the function `funspace()` obtained using all data points in the `GSPFF` dataset. Colours indicate the probabilistic distribution of trait combinations in the functional trait space defined by a PCA (red = high probability; yellow = low probability). Contour lines indicate 0.99, 0. 50, and 0.25 quantiles of the probability distribution. The output shows that there are two hotspots (corresponding, from left to right, to herbaceous plants and angiosperm trees, see Díaz et al., 2016). The variance explained by each component and the loadings of the original traits are also shown. (b) Plot showing the GAM predicted values for a variable (see Box 3 for explanations) within the same functional space; the input was created using `funspaceGAM()`. The heatmap shows the predicted values for a variable with increasing values as we move away from the centre of coordinates of the functional space. Contour lines are the quantiles of the GAM predictions expressed in the same unit as the response variable. The code to reproduce each panel is available in Boxes 1–3.

One advantage of **funspace** for users is its ability to generate functional spaces using only a matrix of functional features, eliminating the need for additional input data (i.e. abundance or presence/absence matrices) required by other packages that are mainly focused on community-level diversity indices. For example, both the BAT and betapart R packages (Baselga & Orme, 2012; Cardoso et al., 2015) require a community matrix to estimate the functional space. However, it should be noted that both packages are more focused on estimating alpha and beta diversity indices, including taxonomic, phylogenetic, and functional aspects, so their focus is different from that of our package. Moreover, and importantly, they do not include specific functions for visualizing or exploring functional spaces, nor any function to map external variables in such spaces.

Mapping different variables in a functional space is one of the main features of **funspace**, and this is done using the `funspaceGAM()` function. `funspaceGAM()` requires as input an object created by the core `funspace()` function and the target response variable, allowing for mapping target variables in any trait space in a seamless and intuitive manner. Finally, the resulting plots can be easily customized, and, by default, they are publication ready. To the best of our knowledge, only two R functions perform similar operations. These are the `envfit()` and `ordisurf()` functions included in the vegan R package (Oksanen et al., 2022) and they allow for fitting multiple regression or GAM with a similar purpose as `funspaceGAM()`. However, `envfit()` and `ordisurf()` need the ordination axes as input. Thus, ordination axes need to be separately extracted from objects of ordination functions, requiring an additional step from the users. Moreover, those alternative functions, being more focused on modelling, provide default plots that are generally barebone, often requiring additional tuning from the users as well as additional packages to improve plotting (e.g. ggordiplots, Quensen et al., 2023). `funspaceGAM()` solves all these limitations by automatically retrieving ordination axes for modelling, thus providing a single-step solution for generating publication-ready plots, and associated statistical tests, of how target variables distribute within ordination analyses outputs.

Finally, the mFD R package (Magneville et al., 2022), a recently released extension of the previous FD package (Laliberté et al., 2014), offers a wide range of functions to calculate trait-based distances, construct multidimensional functional spaces, and compute various alpha and beta functional diversity indices. In addition, mFD allows for the construction of functional spaces using only a trait matrix. However, and this is the main difference with **funspace**, mFD relies on convex hull estimation to build functional spaces, an approach that may have limitations with specific types of data. For example, convex hulls are extremely sensitive to outliers and are not able to account for the differential distribution of data points within trait spaces (i.e. multivariate

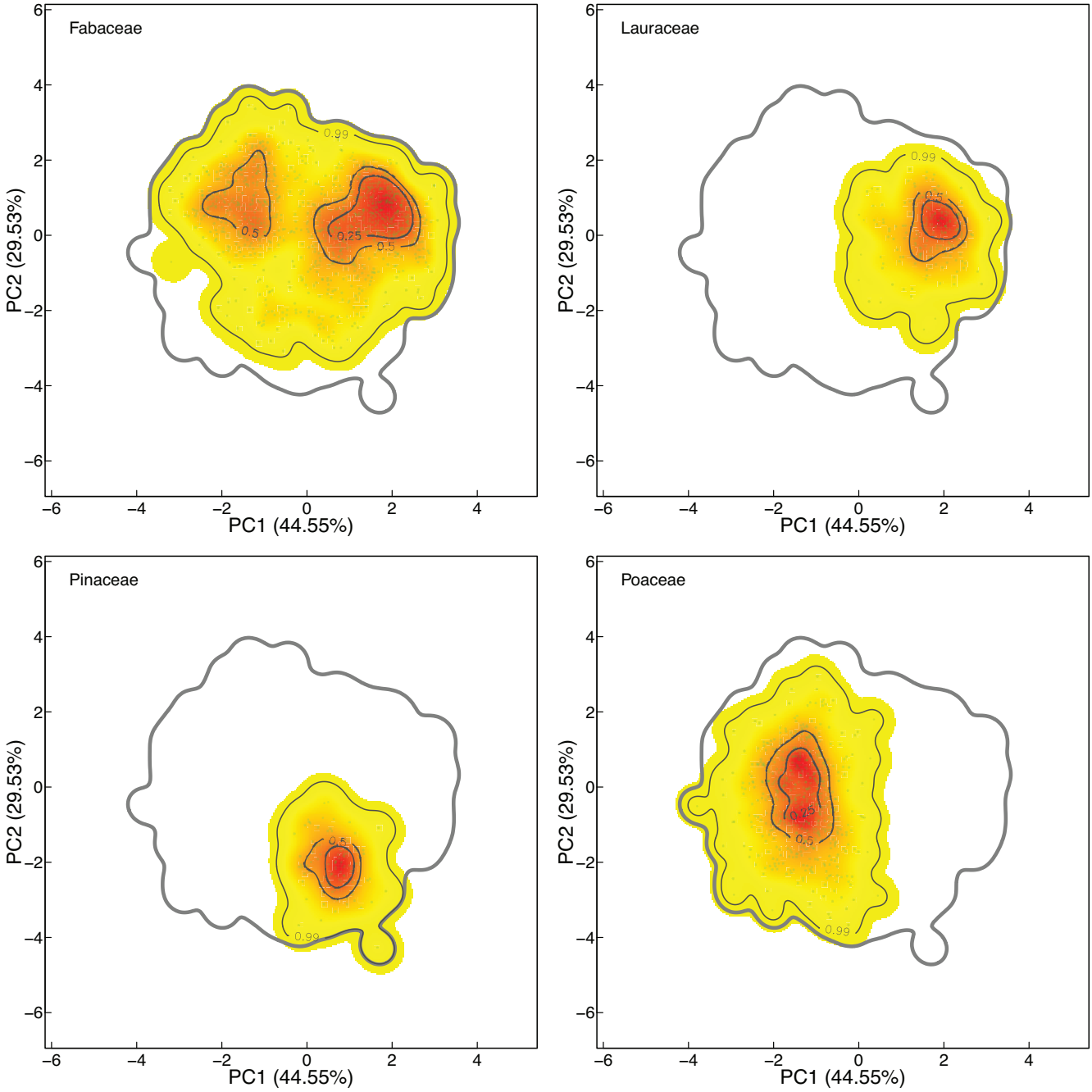

FIGURE 3 Plot of a `funspace` object created with the function `funspace()` including groups for four major plant families (Fabaceae, Lauraceae, Pinaceae, Poaceae). For each family, its corresponding trait probability distribution is represented within the functional space; the 99.9% probability quantile of the global trait probability distribution (i.e. the one including all species from the four families together) is shown to provide a common reference and make comparisons easier. Colours, interpretation of contour lines and explanation of axes is the same as in Figure 2a. The code to reproduce this figure is available in Boxes 1 and 3.

density) (Mammola et al., 2021). This limitation of convex hulls becomes especially important in cases when missing data need to be imputed (Stewart et al., 2023). Additionally, mFD lacks support for external variable mapping within the functional spaces, although this falls outside the package's intended scope. In sum, mFD and **funspace** are complementary packages to broaden the set of tools for users to explore the diversity of organismal form and function across the tree of life.

6 | CONCLUSIONS

funspace streamlines all the necessary steps to build, explore, map, and plot functional trait spaces, making such analyses readily accessible to any user interested in bivariate or multivariate functional trait analyses. Importantly, **funspace** automatizes most of the procedures needed to run such analyses, and for this reason can be of use to inexperienced R users as well. This package provides a

standardized and reproducible set of procedures that can be used to increase comparability among studies involving functional trait space analyses at different scales and across disciplines. Moreover, this package, due to its easy usage, has the potential to increase the number of studies addressing research questions that require simultaneous consideration of multiple traits and their relationship with multiple ecological variables, such as climate (i.e. functional trait space-environment relationships). Lastly, **funspace** provides key outputs to interpret the main features of any functional trait space and publication-ready plots, increasing the usefulness and potential applicability of this package. We hope that **funspace** will provide the basic tool to build species' functional traits spaces across the tree of life, to explore the main features of such spaces, and to link them to any biological and non-biological factor that can influence species' functional diversity.

ACKNOWLEDGEMENTS

We thank the reviewers and the editor for their insightful comments.

FUNDING INFORMATION

CPC was supported by the Estonian Research Council (PSG293 and PRG2142) and the European Regional Development Fund via the Mobilitas Pluss programme (MOBERC40 and MOBERC100). GP was supported by the grant IJC2020-043331-I funded by MCIN/AEI/10.13039/501100011033, and by the grant PID2021-122214NA-I00 funded by MCIN/AEI/10.13039/501100011033 and by FEDER 'ESF Investing in your future'.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

PEER REVIEW

The peer review history for this article is available at https://www.webofscience.com/api/gateway/wos/peer-review/10.1111/ddi.13820.

DATA AVAILABILITY STATEMENT

The package is available at CRAN (https://cran.r-project.org/package=funspace) and it contains all the example datasets used in the main text. The package and the datasets are also available in Dryad: Carmona et al. (2024). https://doi.org/10.5061/dryad.4tmpg4fg6.

ORCID

Carlos P. Carmona https://orcid.org/0000-0001-6935-4913
Nicola Pavanetto https://orcid.org/0000-0002-9441-863X
Giacomo Puglielli https://orcid.org/0000-0003-0085-4535

REFERENCES

Baselga, A., & Orme, C. D. L. (2012). betapart: An R package for the study of beta diversity. *Methods in Ecology and Evolution*, *3*, 808–812. https://doi.org/10.1111/j.2041-210X.2012.00224.x

Bueno, C. G., Toussaint, A., Träger, S., Díaz, S., Moora, M., Munson, A. D., Pärtel, M., Zobel, M., Tamme, R., & Carmona, C. P. (2023). Reply to: The importance of trait selection in ecology. *Nature*, *618*, E31–E34.

Cardoso, P., Rigal, F., & Carvalho, J. C. (2015). BAT – Biodiversity assessment tools, an R package for the measurement and estimation of alpha and beta taxon, phylogenetic and functional diversity. *Methods in Ecology and Evolution*, *6*, 232–236. https://doi.org/10.1111/2041-210X.12310

Carmona, C. P., Bueno, C. G., Toussaint, A., Träger, S., Díaz, S., Moora, M., Munson, A. D., Pärtel, M., Zobel, M., & Tamme, R. (2021). Fine-root traits in the global spectrum of plant form and function. *Nature*, *597*(7878), 7878. https://doi.org/10.1038/s41586-021-03871-y

Carmona, C. P., de Bello, F., Mason, N. W. H., & Lepš, J. (2016). Traits without borders: Integrating functional diversity across scales. *Trends in Ecology & Evolution*, *31*(5), 382–394. https://doi.org/10.1016/j.tree.2016.02.003

Carmona, C. P., de Bello, F., Mason, N. W. H., & Lepš, J. (2019). Trait probability density (TPD): Measuring functional diversity across scales based on TPD with R. *Ecology*, *100*(12), e02876. https://doi.org/10.1002/ecy.2876

Carmona, C. P., Pavanetto, N., & Puglielli, G. (2024). funspace: An R package to build, analyze and plot functional trait spaces [dataset]. Dryad. https://doi.org/10.5061/dryad.4tmpg4fg6

Carmona, C. P., Tamme, R., Pärtel, M., de Bello, F., Brosse, S., Capdevila, P., González, M. R., González-Suárez, M., Salguero-Gómez, R., Vásquez-Valderrama, M., & Toussaint, A. (2021). Erosion of global functional diversity across the tree of life. *Science Advances*, *7*(13), eabf2675. https://doi.org/10.1126/sciadv.abf2675

Chelli, S., Klimešová, J., Tsalakos, J. L., & Puglielli, G. (2024). Unraveling the clonal trait space: Beyond aboveground and fine-root traits. *Journal of Ecology*. https://doi.org/10.1111/1365-2745.14265

Cox, D. T. C., Gardner, A. S., & Gaston, K. J. (2021). Diel niche variation in mammals associated with expanded trait space. *Nature Communications*, *12*(1), 1. https://doi.org/10.1038/s41467-021-22023-4

Darling, E. S., Alvarez-Filip, L., Oliver, T. A., McClanahan, T. R., & Côté, I. M. (2012). Evaluating life-history strategies of reef corals from species traits. *Ecology Letters*, *15*(12), 1378–1386. https://doi.org/10.1111/j.1461-0248.2012.01861.x

Dawson, S. K., Carmona, C. P., González-Suárez, M., Jönsson, M., Chichorro, F., Mallen-Cooper, M., Melero, Y., Moor, H., Simaika, J. P., & Duthie, A. B. (2021). The traits of "trait ecologists": An analysis of the use of trait and functional trait terminology. *Ecology and Evolution*, *11*(23), 16434–16445. https://doi.org/10.1002/ece3.8321

de Bello, F., Carmona, C. P., Dias, A. T. C., Götzenberger, L., Moretti, M., & Berg, M. P. (2021). *Handbook of trait-based ecology. From theory to R tools*. https://doi.org/10.1017/9781108628426

Diaz, S., Hodgson, J. G., Thompson, K., Cabido, M., Cornelissen, J. H. C., Jalili, A., Montserrat-Martí, G., Grime, J. P., Zarrinkamar, F., Asri, Y., Band, S. R., Basconcelo, S., Castro-Díez, P., Funes, G., Hamzehee, B., Khoshnevi, M., Pérez-Harguindeguy, N., Pérez-Rontomé, M. C., Shirvany, F. A., … Zak, M. R. (2004). The plant traits that drive ecosystems: Evidence from three continents. *Journal of Vegetation Science*, *15*(3), 295–304. https://doi.org/10.1111/j.1654-1103.2004.tb02266.x

Díaz, S., Kattge, J., Cornelissen, J. H. C., Wright, I. J., Lavorel, S., Dray, S., Reu, B., Kleyer, M., Wirth, C., Colin Prentice, I., Garnier, E., Bönisch, G., Westoby, M., Poorter, H., Reich, P. B., Moles, A. T., Dickie, J., Gillison, A. N., Zanne, A. E., … Gorné, L. D. (2016). The global spectrum of plant form and function. *Nature*, *529*(7585), 7585. https://doi.org/10.1038/nature16489

Dinno, A. (2018). *paran: Horn's test of principal components/factors (1.5.2)*. https://CRAN.R-project.org/package=paran

Dray, S., & Dufour, A.-B. (2007). The ade4 package: Implementing the duality diagram for ecologists. *Journal of Statistical Software*, *22*, 1–20. https://doi.org/10.18637/jss.v022.i04

Duong, T. (2007). ks: Kernel density estimation and kernel discriminant analysis for multivariate data in R. *Journal of Statistical Software*, *21*, 1–16. https://doi.org/10.18637/jss.v021.i07

González-Suárez, M., Lucas, P. M., & Revilla, E. (2012). Biases in comparative analyses of extinction risk: Mind the gap. *Journal of Animal Ecology*, *81*(6), 1211–1222.

Greenslade, P. J. M. (1983). Adversity selection and the habitat templet. *The American Naturalist*, *122*(3), 352–365.

Grime, J. P. (1977). Evidence for the existence of three primary strategies in plants and its relevance to ecological and evolutionary theory. *The American Naturalist*, *111*(982), 1169–1194.

Guillemot, J., Martin-StPaul, N. K., Bulascoschi, L., Poorter, L., Morin, X., Pinho, B. X., Le Maire, G. R. L., Bittencourt, P., Oliveira, R. S., Bongers, F., Brouwer, R., Pereira, L., Gonzalez Melo, G. A., Boonman, C. C. F., Brown, K. A., Cerabolini, B. E. L., Niinemets, Ü., Onoda, Y., Schneider, J. V., … Brancalion, P. H. S. (2022). Small and slow is safe: On the drought tolerance of tropical tree species. *Global Change Biology*, *28*(8), 2622–2638. https://doi.org/10.1111/gcb.16082

Horn, J. L. (1965). A rationale and test for the number of factors in factor analysis. *Psychometrika*, *30*(2), 179–185. https://doi.org/10.1007/BF02289447

Jin, Y., & Qian, H. (2019). V.PhyloMaker: An R package that can generate very large phylogenies for vascular plants. *Ecography*, *42*(8), 1353–1359. https://doi.org/10.1111/ecog.04434

Junker, R. R., Albrecht, J., Becker, M., Keuth, R., Farwig, N., & Schleuning, M. (2023). Towards an animal economics spectrum for ecosystem research. *Functional Ecology*, *37*(1), 57–72. https://doi.org/10.1111/1365-2435.14051

Kattge, J., Bönisch, G., Díaz, S., Lavorel, S., Prentice, I. C., Leadley, P., Tautenhahn, S., Werner, G. D. A., Aakala, T., Abedi, M., Acosta, A. T. R., Adamidis, G. C., Adamson, K., Aiba, M., Albert, C. H., Alcántara, J. M., Alcázar, C. C., Aleixo, I., Ali, H., … Wirth, C. (2020). TRY plant trait database – Enhanced coverage and open access. *Global Change Biology*, *26*(1), 119–188. https://doi.org/10.1111/gcb.14904

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

Laughlin, D. C. (2014). The intrinsic dimensionality of plant traits and its relevance to community assembly. *Journal of Ecology*, *102*(1), 186–193. https://doi.org/10.1111/1365-2745.12187

Magneville, C., Loiseau, N., Albouy, C., Casajus, N., Claverie, T., Escalas, A., Leprieur, F., Maire, E., Mouillot, D., & Villéger, S. (2022). mFD: An R package to compute and illustrate the multiple facets of functional diversity. *Ecography*, *2022*, e05904. https://doi.org/10.1111/ecog.05904

Mammola, S., Carmona, C. P., Guillerme, T., & Cardoso, P. (2021). Concepts and applications in functional diversity. *Functional Ecology*, *35*(9), 1869–1885. https://doi.org/10.1111/1365-2435.13882

Mason, N. W. H., Mouillot, D., Lee, W. G., & Wilson, J. B. (2005). Functional richness, functional evenness and functional divergence: The primary components of functional diversity. *Oikos*, *111*(1), 112–118. https://doi.org/10.1111/j.0030-1299.2005.13886.x

Mouillot, D., Loiseau, N., Grenié, M., Algar, A. C., Allegra, M., Cadotte, M. W., Casajus, N., Denelle, P., Guéguen, M., Maire, A., Maitner, B., McGill, B. J., McLean, M., Mouquet, N., Munoz, F., Thuiller, W., Villéger, S., Violle, C., & Auber, A. (2021). The dimensionality and structure of species trait spaces. *Ecology Letters*, *24*(9), 1988–2009. https://doi.org/10.1111/ele.13778

Naimi, B., & Araújo, M. B. (2016). sdm: A reproducible and extensible R platform for species distribution modelling. *Ecography*, *39*(4), 368–375. https://doi.org/10.1111/ecog.01881

Oksanen, J., Simpson, G. L., Blanchet, F. G., Kindt, R., Legendre, P., Minchin, P. R., O'Hara, R. B., Solymos, P., Stevens, M. H. H., Szoecs, E., Wagner, H., Barbour, M., Bedward, M., Bolker, B., Borcard, D., Carvalho, G., Chirico, M., Caceres, M. D., Durand, S., … Weedon, J. (2022). *vegan: Community ecology package*.

Pavanetto, N., Carmona, C. P., Laanisto, L., Niinemets, Ü., & Puglielli, G. (2024). Trait dimensions of abiotic stress tolerance in woody plants of the northern hemisphere. *Global Ecology and Biogeography*, *33*, 272–285. https://doi.org/10.1111/geb.13788

Pedersen, E. J., Miller, D. L., Simpson, G. L., & Ross, N. (2019). Hierarchical generalized additive models in ecology: An introduction with mgcv. *PeerJ*, *7*, e6876. https://doi.org/10.7717/peerj.6876

Penone, C., Davidson, A. D., Shoemaker, K. T., Di Marco, M., Rondinini, C., Brooks, T. M., Young, B. E., Graham, C. H., & Costa, G. C. (2014). Imputation of missing data in life-history trait datasets: Which approach performs the best? *Methods in Ecology and Evolution*, *5*(9), 961–970. https://doi.org/10.1111/2041-210X.12232

Peres-Neto, P. R., Jackson, D. A., & Somers, K. M. (2005). How many principal components? Stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis*, *49*(4), 974–997. https://doi.org/10.1016/j.csda.2004.06.015

Pianka, E. R. (1970). On r- and K-selection. *The American Naturalist*, *104*(940), 592–597.

Puglielli, G., Hutchings, M. J., & Laanisto, L. (2021). The triangular space of abiotic stress tolerance in woody species: A unified trade-off model. *New Phytologist*, *229*(3), 1354–1362. https://doi.org/10.1111/nph.16952

Puglielli, G., Pavanetto, N., & Laanisto, L. (2022). Towards a "periodic table" of abiotic stress tolerance strategies of woody plants. *Flora*, *292*, 152089. https://doi.org/10.1016/j.flora.2022.152089

Quensen, J., Simpson, G., & Oksanen, J. (2023). *ggordiplots: Make "ggplot2" versions of Vegan's Ordiplots*.

Revell, L. J. (2012). phytools: An R package for phylogenetic comparative biology (and other things). *Methods in Ecology and Evolution*, *3*(2), 217–223. https://doi.org/10.1111/j.2041-210X.2011.00169.x

Sandel, B., Gutiérrez, A. G., Reich, P. B., Schrodt, F., Dickie, J., & Kattge, J. (2015). Estimating the missing species bias in plant trait measurements. *Journal of Vegetation Science*, *26*(5), 828–838. https://doi.org/10.1111/jvs.12292

Schrodt, F., Kattge, J., Shan, H., Fazayeli, F., Joswig, J., Banerjee, A., Reichstein, M., Bönisch, G., Díaz, S., Dickie, J., Gillison, A., Karpatne, A., Lavorel, S., Leadley, P., Wirth, C. B., Wright, I. J., Wright, S. J., & Reich, P. B. (2015). BHPMF – A hierarchical Bayesian approach to gap-filling and trait prediction for macroecology and functional biogeography. *Global Ecology and Biogeography*, *24*(12), 1510–1521. https://doi.org/10.1111/geb.12335

Stekhoven, D. J., & Bühlmann, P. (2012). MissForest—Non-parametric missing value imputation for mixed-type data. *Bioinformatics*, *28*(1), 112–118. https://doi.org/10.1093/bioinformatics/btr597

Stewart, K., Carmona, C. P., Clements, C., Venditti, C., Tobias, J. A., & González-Suárez, M. (2023). Functional diversity metrics can perform well with highly incomplete data sets. *Methods in Ecology and Evolution*, *14*, 2856–2872. https://doi.org/10.1111/2041-210X.14202

Toussaint, A., Brosse, S., Bueno, C. G., Pärtel, M., Tamme, R., & Carmona, C. P. (2021). Extinction of threatened vertebrates will lead to idiosyncratic changes in functional diversity across the world. *Nature Communications*, *12*(1), 1. https://doi.org/10.1038/s41467-021-25293-0

Venables, W. N., & Dichmont, C. M. (2004). GLMs, GAMs and GLMMs: An overview of theory for applications in fisheries research. *Fisheries Research*, *70*(2), 319–337. https://doi.org/10.1016/j.fishres.2004.08.011

Villéger, S., Novack-Gottshall, P. M., & Mouillot, D. (2011). The multidimensionality of the niche reveals functional diversity changes in benthic marine biotas across geological time. *Ecology Letters*, *14*(6), 561–568. https://doi.org/10.1111/j.1461-0248.2011.01618.x

Westoby, M. (1998). A leaf-height-seed (LHS) plant ecology strategy scheme. *Plant and Soil*, *199*(2), 213–227. https://doi.org/10.1023/A:1004327224729

Westoby, M., Falster, D. S., Moles, A. T., Vesk, P. A., & Wright, I. J. (2002). Plant ecological strategies: Some leading dimensions of variation between species. *Annual Review of Ecology and Systematics*, *33*(1), 125–159. https://doi.org/10.1146/annurev.ecolsys.33.010802.150452

Westoby, M., Nielsen, D. A., Gillings, M. R., Gumerov, V. M., Madin, J. S., Paulsen, I. T., & Tetu, S. G. (2021). Strategic traits of bacteria and archaea vary widely within substrate-use groups. *FEMS Microbiology Ecology*, *97*(11), fiab142. https://doi.org/10.1093/femsec/fiab142

Winemiller, K. O., Fitzgerald, D. B., Bower, L. M., & Pianka, E. R. (2015). Functional traits, convergent evolution, and periodic tables of niches. *Ecology Letters*, *18*(8), 737–751. https://doi.org/10.1111/ele.12462

Winemiller, K. O., & Rose, K. A. (1992). Patterns of life-history diversification in north American fishes: Implications for population regulation. *Canadian Journal of Fisheries and Aquatic Sciences*, *49*(10), 2196–2218. https://doi.org/10.1139/f92-242

Wood, S. N. (2017). *Generalized additive models: An introduction with R, Second Edition* (2nd ed.). Chapman and Hall/CRC. https://doi.org/10.1201/9781315370279

BIOSKETCHES

CPC is assistant professor of macroecology at the University of Tartu (Estonia). NP is a PhD student in plant ecology at the Estonian University of Life Sciences (Tartu). GP is a post-doctoral researcher in plant ecology and ecophysiology at the University of Seville (Spain). They share interest in trait-based ecology at multiple scales, with special reference to defining general patterns of how functional traits integrate to define the adaptive syndromes employed by organisms (mainly vascular plants) to cope with multiple abiotic and biotic limitations.

Author contributions: CPC and GP conceived the idea. All authors equally contributed to the package development. CPC produced the final version of the package. GP and NP wrote the first draft of the manuscript. All authors contributed to reach the final version.

How to cite this article: Carmona, C. P., Pavanetto, N., & Puglielli, G. (2024). `funspace`: An R package to build, analyse and plot functional trait spaces. *Diversity and Distributions*, *30*, e13820. https://doi.org/10.1111/ddi.13820