NATDB: An R package that downloads species trait data, but is Not A Trait DataBase

Order TBD: William D. Pearse, Konrad Hafen, Mallory Hagadorn, Marley Haupt, Spencer B. Hudson, Sylvia Kinosian, Ryan McCleary, Alexandre Rego, & Katie Welgarz

March 7, 2017

1 Abstract

1. Ecologists and evolutionary biologists often wish to make use of species trait data, either as ancillary data, such as in community ecology, or as the primary focus of a study, such as macro-evolutionary modelling.
2. Such biologists are often hampered by the difficulties of collecting sufficient trait data from published sources. There are very few open access databases of species' traits.
3. We present `NATDB`, an `R` package—not a trait database—that automatically downloads species trait data from existing sources.
4. NATDB collates trait data from over XXX publications across XXX species, and at the time of writing downloads over XXX individual trait measurements.
5. NATDB is emphatically *not* a trait database: it circumvents issues over intellectual ownership of species trait data by not distributing data, but rather giving users automated tools to build their own data from existing, published datasets. Our hope is to establish a user community around this package, adding both additional data sources and cleaning routines for the data itself.
6. Upon acceptance, `NATDB` will be uploaded to CRAN, but is currently available for download on `GitHub`. It can be installed by typing `library(devtools);install_github("willpearse/natdb")` at an `R` console.

2 Introduction

Ecologists and evolutionary biologists have long recognised the importance of (functional) traits in their work (Dıaz & Cabido 2001). Large datasets of plants (Kattge *et al.* 2011), mammals (Jones *et al.* 2009), and birds (Wilman *et al.* 2014) have opened the door to analyses of the evolution (Harmon & Glor 2010) and global distribution (Kattge *et al.* 2011) of trait diversity. Species' traits help us better predict how species will respond to land use (Mayfield *et al.* 2010) and climate change (Estrada *et al.* 2016), allowing us to generalise and compare across species to find general biodiversity patterns.

Yet, despite its importance, it is often difficult to find data on species' functional traits. We suggest there are three main reasons for this: (1) it is difficult to obtain trait data, (2) it is difficult to collate trait data, and (3) concerns have been raised about intellectual property and the distribution of trait data. (1) Often the most functionally important species traits are the most difficult to measure (Cornelissen *et al.* 2003; Violle *et al.* 2007), and even when measuring a trait is simple finding a specimen is often not. Usefully measuring and defining species' traits is not an easy thing. (2) Creating and maintaining large databases is difficult: the nomenclature for species and traits is not universal (Kattge *et al.* 2011; Hudson *et al.* 2017), and unifying concepts across different datasets takes detailed knowledge of species and their traits. (3) Unlike other other kinds of data such as DNA sequences (Benson *et al.* 2013), the publication of species trait data has been controversial (*e.g.*, Poisot *et al.* 2014; Moles *et al.* 2013). The reasons for this are complex and numerous, but perhaps the most compelling argument is a concern that releasing data will lead to it being 'hoovered up' into a database where the creator of the database will get credit but the original collectors of the data none.

We present here NATDB, an R package that releases over XXX pieces of trait data for over XXX species, making existing species trait data more widely available for use by ecologists and evolutionary biologists. We argue that NATDB is a prototype for a new way of making data more accessible that avoids concerns about data ownership: NATDB is *not a trait database*. NATDB is a software package that simplifies the process of collating data the user already had access to, and so obviates any concerns over 'hoovering up' data because it simply retrieves data the authors have already publicly released. NATDB contains no data, and so users must cite the sources of data when using the package. This model both liberates the vast trait-based knowledge that already exists in the literature, and protects the intellectual contributions of those who collected the data in the first place.

3 Description

NATDB consists of a series of internal functions, each of which downloads species trait data from a single paper. Typically, a user will download a set of data and then subset that down to only the species or traits that they require. Note that, by default, NATDB waits ten seconds between downloading datasets to minimise impact on journals' servers. For example, the following would download all the data in NATDB, and then subset that down to only two kinds of traits for two species:

```
library(natdb)
data <- natdb(taxon)
species <- c("Quercus_robur", "Pinus_sylvestris")
traits <- c("SLA", "height")
subset.data <- data[species, traits]
```

By default, NATDB caches whatever it has been asked to download during an R session. So, for example, if the user were to realise that they wanted data on an additional species or trait after executing the code above, they could run the entire script again and NATDB would not download any more data. The cache option allows the user to override this behaviour if they desire. A user can also specify a directory when invoking NATDB so that they can save their searches between sessions. The following code, for example, would cache results between sessions, and would add additional data to that cache as new versions of NATDB were released. This is the recommended way to use NATDB, as it saves the user time and reduces server load.

```
data <- natdb(cache="~/.natdb_cache/")
subset.data <- data[c("Phocoena_phocoena","Tursiops_truncatus"),]
```

NATDB has a single class for dealing with trait data, called (unimaginatively) NATDB. This class has head, print, summary, rbind, cbind, and as.data.frame methods to make it easier for the user to work with their data. Internally, NATDB distinguishes between, and convert all data into, numeric and character types, and *melts* (*sensu* Wickham 2007) all data within these types. This makes it easy to add new data to an existing NATDB object, while keeping reducing memory load. If NATDB were to store data in a data.frame-like format, it would require XXX cells (XXX species, XXX traits) to store all its data, XXX% of which would be missing.

Taxonomic group	# Species	# Traits	% Complete	Citations
Plants			Wright *et al.* (2004)	
Mammals			Jones *et al.* (2009) and Wilman *et al.* (2014)	
Birds			Wilman *et al.* (2014)	
...TBC...				

Table 1: Overview of data available for download within NATDB. Overall, the package downloads XXX data points, covering XXX species and XXX separate functional traits. XXX% of these trait values have some form of meta-data associated with them.

Ready access to meta-data is important in any database. The databases NATDB can build are complex, in that the meta-data that different source datasets provide can vary a great deal. We follow the general approach of *FigShare* (https://figshare.com/) and *DataDryad* (http://datadryad.org/) in not enforcing rigid meta-data requirements, but placing the meta-data of each source dataset within a comparable framework so as to allow users to interrogate the meta-data in their own way. Thus while we do standardise some aspects of the data (*e.g.*, ensuring all latitude and longitude measurements, where present, are termed `latitude` and `longitude`), users must check whatever subset of data they have to see what meta-data are available. For example:

```
head(metadata(subset.data))
table(metadata(subset.data))
plot(subset.data$SLA ~ metadata(subset.data)$latitude)
```

We make no guarantee that the taxonomy or units of the data within NATDB are internally compatible: users are responsible for checking the validity of the data they have collated. However, we have written wrappers for common taxonomic services for cleaning problems with species nomenclature using existing R packages (Cayuela *et al.* 2012, Taxonstand and others?).

```
data <- natdb(taxon)[c("Quercus_robur", "Pinus_sylvestris"), c("SLA", "height")]
subset.data <- data[units(data) == "mm", "height"]
subset.data <- tpl.clean(subset.data)
```

Finally, it is important that those who generated the data NATDB downloads are appropriately cited. It is simple to generate BibTeX, RIS, and plaintext files to help with citations:

```
citation(subset.data, "bibtex")
```

4 Comparison with existing tools

As table 2 shows, NATDB downloads more data than a set of comparable databases, although its data is, perhaps by nature of its wide taxonomic coverage, less complete per species. The most important way in which NATDB differs from the other tools and datasets in table 2 is that it has been designed, from the ground-up, to be easy to extend. Adding a publication's data to the package requires no knowledge other than the basic structure of data to be added. The average length of the functions that load a data structure into NATDB is XXX lines of R code, in part because as part of this project we developed code for the R package fulltext (Chamberlain 2015) to automate the download of data from published papers. NATDB uses reflective programming to query itself to determine what datasets are available for download, and as such extension is trivial. This represents

Dataset	R native?	Taxonomic scope	# Species	# Traits	% Complete	Citation
TRY	✗	Plants			Kattge *et al.* (2011)	
D3	✗	Plants			Hintze *et al.* (2013)	
TR8	✓	Plants			Gionata (2015)	
NATDB	✓	Organisms				

Table 2: Comparison of NATDB to existing packages or databases. As described in the text, we only compare NATDB with open access databases and packages.

a major advantage to NATDB: it is a living package that will, we hope, grow as authors add their own publications to it. We provide detailed instructions on how to contribute data sources to NATDB in the package's vignette.

The flexibility and scope of NATDB, however, means it has not been as carefully cleaned and checked as datasets typically are. This is by design: NATDB is fundamentally different, and we use TRY (Kattge *et al.* 2011) to illustrate this. TRY is a carefully-collated dataset that has required thousand of person-hours to create, and to reflect this and ensure that the data is used correctly, its authors require that many data contributors and the two lead authors of the database are offered co-authorship on any publication making use of TRY data. We consider this a reasonable request given the amount of effort involved in producing a database like TRY, and the feedback and data-validation that these additional co-authors provide to a finished manuscript. NATDB is not a database and does not follow this model: the data are provided 'as-is' and neither we, nor the original data publishers, require co-authorship for use of the package. Basic taxonomic and data

5 Future directions

We actively encourage code contributions, and the package's online vignette contains a detailed set of instructions on how to contribute functions that download data from new sources. Our intention is to make the process as simple as possible, in the hope that authors who release the data underlying their publications will contribute to the package, in the process making NATDB better and making it easier for others to use (and cite) their work. This, in part, is the reason we have few formal checks on meta-data and units within NATDB: we consider the first hurdle to be getting more data into a useable format within R, and all other issues can be handled later. We hope that, using NATDB as a base, others will develop cleaning and checking routines that can be applied to the package. Whether these will be incorporated into NATDB itself, or released as separate companion package(s), remains to be seen.

NATDB is both an experiment in a new way of making data more accessible, and a useful resource that we are already making use of in our daily working lives. It is our hope that NATDB, and resources like it, will continue to grow.