SOFTWARE

prepR4pcm: An R Package for Preparing Data and Trees for Phylogenetic Comparative Methods

Shinichi Nakagawa[1,2] | Santiago Ortega[1] | Ayumi Mizuno[1] | Eduardo S.A. Santos[1] | Malgorzata Lagisz[1,2] | Bhavya Jain[1] | Jimuel Jr Celeste[1] | Sergio Poo Hernandez[1]

[1]COSSEE, Department of Biological Sciences, University of Alberta, Edmonton, AB, Canada; [2]Evolution & Ecology Research Centre, School of Biological, Earth and Environmental Sciences, University of New South Wales, Sydney, NSW, Australia

Correspondence: Shinichi Nakagawa, snakagaw@ualberta.ca / s.nakagawa@unsw.edu.au

Running headline: Preparing data and trees for PCMs

Abstract

1. Phylogenetic comparative methods require species names in a trait dataset to match tip labels in a phylogenetic tree. Yet this apparently simple prerequisite is often one of the most fragile steps in a comparative workflow. Names may differ because of, for example, formatting, taxonomic revisions, synonyms, or spelling errors. If these differences are resolved informally, species can be lost from analyses, and the reasons for their loss can be difficult to reconstruct.
2. Here, we present `prepR4pcm`, an R package for preparing data and trees for phylogenetic comparative methods. The package reconciles species names through a staged procedure: exact matching, normalised matching, synonym lookup with local taxonomic databases, and optional fuzzy matching for likely spelling errors. Each decision is stored in a reconciliation object with the original name, matched name, match type, confidence score, and a short explanation. This object turns name matching from a hidden preprocessing step into an auditable part of the analysis.
3. `prepR4pcm` also supports the points where comparative workflows need human judgement. Users can inspect unresolved names, accept or reject suggested matches, add manual corrections, apply taxonomy crosswalks (which link names across taxonomic systems), compare reconciliation runs, and generate reports. The package then returns a matched data frame and pruned tree with the same species set, ready for phylogenetic generalised least squares, phylogenetic mixed models, phylogenetic meta-analysis, and related workflows. If users do not yet have a tree, `prepR4pcm` can retrieve trees from several sources, date trees when suitable information is available, and format tree-source citations.
4. We illustrate the workflow using bundled datasets with realistic name mismatches. `prepR4pcm` is available at `https://github.com/itchyshin/prepR4pcm` with documentation and vignettes covering data and tree reconciliation, tree retrieval, multi-tree workflows, and phylogenetic meta-analysis.

Keywords: *phylogenetic comparative analysis, species names, taxonomic harmonisation, data–tree reconciliation, provenance*

1 Introduction

Phylogenetic comparative methods (PCMs) allow researchers to study trait variation while accounting for shared evolutionary history (Felsenstein, 1985; Grafen, 1989). These methods range from phylogenetic generalised least squares (PGLS; Freckleton et al., 2002) to phylogenetic generalised linear mixed models or meta-analyses (e.g. Hadfield and Nakagawa, 2010; Nakagawa and Santos, 2012; Mizuno et al., 2025). Before fitting such models, researchers need two matched objects: a dataset with species names and a phylogenetic tree with tip labels representing the same species names. Yet this apparently simple prerequisite, matching species names, is often a stumbling block in a comparative workflow, because using scientific species names (binomial strings) as keys causes mismatches that can remove species from the analysis.

The name-matching problem often becomes visible only when the data and the tree are brought together. A species may be present in both sources, but fails to match because one source writes the name as "*Parus major*", whereas another uses the tip label "*Parus_major*". In other cases, the two sources may use different names for the same or corresponding taxon, for example, because of an older genus name, a synonym, or a different taxonomic authority. Also, species may be genuinely absent from the tree. These cases could have different biological and practical implications, yet they may have similar effects if simple filtering is applied to remove mismatched species, thereby reducing the number of species in the analysis. For PCMs, this loss matters not only because the retained species set has less information and power, but also because the data analysed and the phylogenetic structure used to analyse them, especially if the removed species are not randomly distributed.

Several R (R Core Team, 2025) packages address parts of this problem. Taxonomic tools, such as `taxize` (Chamberlain and Szöcs, 2013), `taxadb` (Norman et al., 2020), `Taxonstand` (Cayuela et al., 2012), and `bdc` (Ribeiro et al., 2022), help users validate, standardise, or clean species names; the underlying Global Names Architecture (Mozzherin et al., 2017) provides language-agnostic name-parsing and verification services that several of these tools draw on. Tree-retrieval tools, such as `rotl` (Michonneau et al., 2016), `rtrees` (Li, 2023), `clootl` (Miller et al., 2025), `fishtree` (Rabosky et al., 2018; Chang et al., 2019), and `datelife` (Sanchez Reyes et al., 2024), help users obtain phylogenies. PCM helper functions, such as `geiger::treedata()` (Harmon et al., 2008; Pennell et al., 2014) and `caper::comparative.data()` (Orme et al., 2025), help align data and trees for analysis. However, these tools are not designed to provide a single workflow that takes raw species names, from several data objects and produces aligned comparative objects together with an explicit, auditable record of name-matching decisions. The gap is therefore not the absence of useful tools, but the lack of a transparent conduit connecting between name reconciliation, tree preparation, and downstream PCM analysis (Table 1).

Here, we present `prepR4pcm`, an R package for preparing species-level data and phylogenetic trees for PCMs. The package connects three steps that are often handled separately: name reconciliation, tree preparation, and final data-tree alignment. First, it reconciles species names through a staged procedure that combines exact matching, name normalisation, synonym lookup, and optional fuzzy matching. Second, it records each decision in a reconciliation object, allowing users to inspect which names matched, changed, left unresolved, or corrected manually. Third, it applies the reconciliation to return a data frame and a pruned tree containing the same species set, ready for common comparative workflows, including PGLS, phylogenetic mixed models, and phylogenetic meta-analysis. When users do not yet have a suitable tree, `prepR4pcm`

can also retrieve trees from several sources, record name substitutions, heck the alignment between the tree and species-level data, and format tree-source citations to credit the original references.

Table 1. Existing R tools for taxonomic name handling and phylogenetic tree retrieval relevant to phylogenetic comparative methods. The first three columns identify each tool and place it in context. Tool = R package or function; Role = the main task the tool was designed to perform; Coverage = the main names, taxa, tree sources, databases, or input objects covered by the tool. The remaining columns summarise features relevant to preparing data and trees for PCMs. Tree = accepts or works directly with a user-supplied `ape::phylo` tree; Prov. = records provenance or an audit trail for name-matching, tree retrieval, or alignment decisions; Fuz. = supports fuzzy matching for likely spelling errors; Syn. = supports synonym lookup or accepted-name resolution; Map = supports taxonomy mapping tables or crosswalks between naming systems; Fetch = retrieves a tree from an external source; Multi = can return, handle, or compare more than one tree, such as posterior samples or alternative phylogenies. ✓ = full support; ∼ = partial or context-dependent support; – = absent.

Tool	Role	Coverage	Tree	Prov.	Fuz.	Syn.	Map	Fetch	Multi
`taxize`	Taxonomic name resolution	Multiple online taxonomic databases	–	–	∼	✓	–	–	–
`rgnparser`[a]	Name parsing and verification	Global Names services and linked name sources	–	∼	✓	✓	∼	–	–
`taxadb`	Local taxonomic name resolution	Local copies of multiple taxonomic databases	–	–	–	✓	–	–	–
`Taxonstand`	Plant name standardisation	Plant names, originally against The Plant List	–	∼	∼	✓	–	–	–
`bdc`	Biodiversity data cleaning	Occurrence records and associated taxonomic names	–	∼	∼	✓	–	–	–
`rotl`	Taxonomy and tree retrieval	Open Tree of Life taxonomy and synthesis tree	∼	–	–	✓	–	✓	–
`rtrees`	Megatree assembly	Taxon-specific megatrees, including birds, mammals, fishes, plants, and others	∼	∼	–	–	–	✓	✓
`clootl`	Avian tree retrieval	Bird trees mapped to eBird/-Clements taxonomy	∼	–	–	–	–	✓	∼
`fishtree`	Fish tree retrieval	Ray-finned fish phylogenies from the Fish Tree of Life	∼	–	–	–	–	✓	✓
`datelife`	Dated-tree synthesis	Published dated phylogenies across taxa	∼	∼	–	∼	–	✓	✓
`geiger::treedata`	Data–tree alignment	User-supplied comparative dataset and tree	✓	–	–	–	–	–	–
`caper::comparative.data`	PCM data-object creation	User-supplied comparative dataset and tree	✓	–	–	–	–	–	–
`prepR4pcm`	**Audited PCM preparation pipeline**	**User datasets and trees, with optional tree retrieval**	✓	✓	✓	✓	✓	✓	✓

[a]R wrapper for Global Names Architecture services (Mozzherin et al., 2017); `taxize` also uses the Global Names Verifier internally.
This table compares broad workflow features rather than the quality, completeness, or taxonomic authority of each tool.

2 Package description

2.1 Overview and design

prepR4pcm was designed around three principles: conservative matching, transparent records, and reproducible outputs. These principles apply to both main parts of the package: reconciling species names among data objects and trees, and preparing trees for downstream PCMs.

First, prepR4pcm is conservative. It gives priority to high-confidence matches and does not replace them with weaker matches later in the workflow. Exact and normalised matches are attempted before synonym lookup and optional fuzzy matching. Names are not changed without being recorded, and uncertain matches are left for user review.

Second, the package is transparent. Each reconciliation returns a `reconciliation` object that records what happened to each name. This object stores the original and matched names, the match type, a score, a short explanation, the settings used in the run, and any manual overrides. Users can print the object for a summary or extract the full mapping table for inspection and reporting.

Third, the package is reproducible. The reconciliation object stores the settings used in the run, together with the package, database, and tree-source information where relevant. This design helps users repeat a workflow, compare alternative reconciliation runs, and report how the final data and tree objects were produced.

The package uses ape (Paradis and Schliep, 2019) for phylogenetic tree handling, cli for user messages, rlang for tidy evaluation, and tibble for data frames. Synonym matching uses taxadb (Norman et al., 2020) when this suggested dependency is available.

2.2 Matching cascade

The core of prepR4pcm is a "four-stage matching cascade" (Figure 1). The cascade starts with the safest matches (exact name matching), then moves to those that require more interpretation. At each stage, only names that remain unmatched advance to the next. Thus, an exact match is never replaced by a weaker match later in the pipeline.

Stage 1: Exact matching. The package first compares names exactly as written. These matches receive a score of 1.0 (ranging from 0 to 1).

Stage 2: Normalised matching. The package then standardises names before matching them again. This step deals with common formatting differences, such as spaces versus underscores, inconsistent capitalisation, repeated spaces, authority strings, Open Tree of Life suffixes, and infra-specific rank abbreviations. These matches also receive a score of 1.0, but they are labelled as "normalised" because the original strings differed.

Stage 3: Synonym matching. Names that remain unmatched can be checked against local taxonomic databases through taxadb. This step is useful when the dataset uses one accepted name and the tree uses a synonym or an older name. Where possible, the package records which source was used. Synonym matches receive a lower score than exact or normalised

matches because they depend on the taxonomic authority used. Synonyms also get a score of 1.0.

Stage 4: Fuzzy matching. Finally, the package can suggest near matches for likely spelling errors. Fuzzy matching compares the genus and species epithet using a component-based string distance. It is useful for finding typographical errors, but it can also suggest wrong matches. For this reason, fuzzy matching is optional, and uncertain matches are flagged for review rather than silently accepted. Scores below 1.0 are used for fuzzy matches.

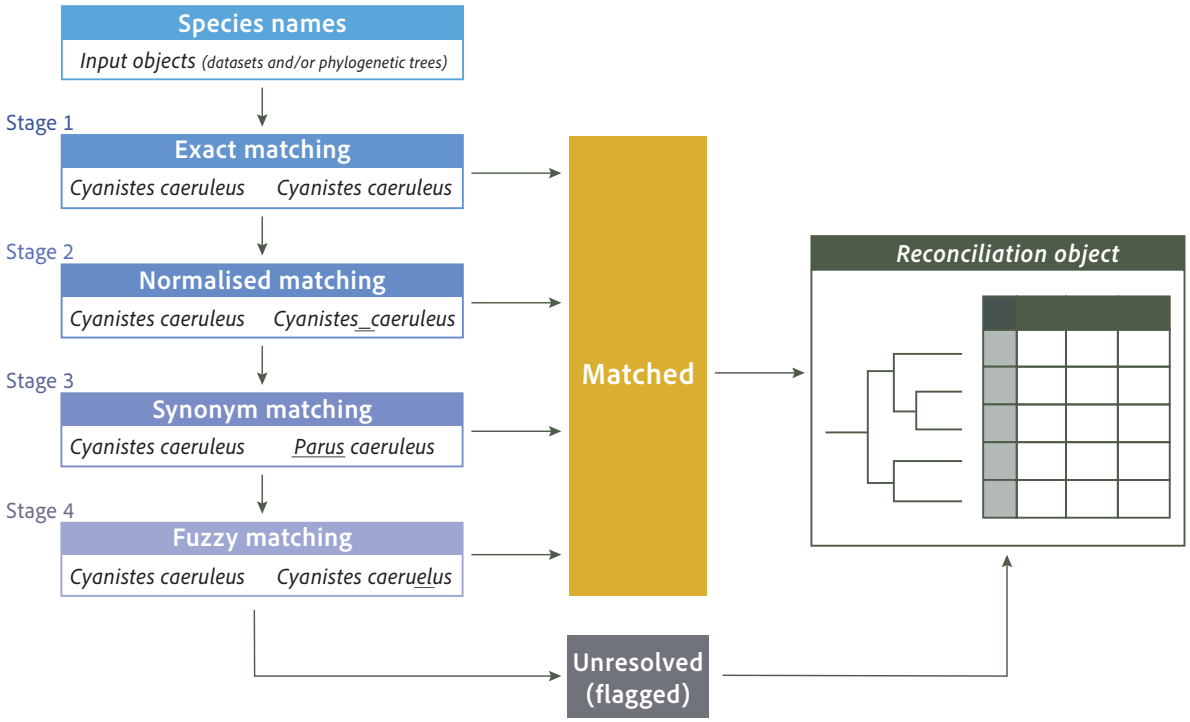


Figure 1. The four-stage matching cascade in `prepR4pcm`. Species names from input objects, such as datasets and phylogenetic trees, are passed through a conservative matching cascade. Exact matches are identified first, followed by matches after name normalisation, synonym-based matches, and optional fuzzy matches for likely spelling errors. The example species names illustrate the intended distinction among matching stages: identical spelling for exact matching (*Cyanistes caeruleus*), formatting differences for normalised matching (*Cyanistes_caeruleus*), synonymy or older nomenclature for synonym matching (*Parus caeruleus*), and a likely typographical error for fuzzy matching (*Cyanistes caeruelus*). Names matched at any stage, as well as names retained as unresolved or flagged cases, are stored in the reconciliation object so that the process can be inspected, reported, and reproduced.

2.3 The reconciliation object

The central output of `prepR4pcm` is a `reconciliation` object. This object records how names were matched, changed, left unresolved, or corrected by the user. It is produced not only when a dataset is reconciled with a tree, but also when users reconcile two datasets, two trees, several datasets against one tree, or one dataset against several trees. Thus, the same object structure is used across data–data, data–tree, tree–tree, multi-dataset, and multi-tree workflows.

A `reconciliation` object has four main components. The `$mapping` table records the names from the input objects and the outcome for each name. The exact columns depend on the workflow, but the table includes information such as original names, matched names, match status, match type, score, source, and notes. The `$meta` list stores the settings used in the run, together with timestamps and version information. The `$counts` table summarises how many names were matched, changed, or left unresolved. The `$overrides` table records manual changes made by the user.

The object separates reconciliation from application. Users can first inspect the name-matching decisions, then decide whether to accept them, add corrections, or rerun the reconciliation with different settings. The object has `print()`, `summary()`, and `format()` methods. Printing gives a short summary of match coverage. The function `reconcile_mapping()` returns the full mapping table, which users can inspect, save, or include as supplementary material.

This object is the main difference between a simple alignment and an auditable reconciliation. Standard PCM helper functions are useful for making data and trees ready for analysis, but they are not designed to record all name-matching decisions. `prepR4pcm` keeps this record, so users can see which names matched, which names changed, which names remained unresolved, and which names were corrected by hand before the final data and tree objects are created.

2.4 Curation, augmentation, and multi-input workflows

Because the reconciliation object keeps the matching decisions separate from the final aligned objects, users can curate the result before applying it. Automated matching should not be the final word in every case. Taxonomy changes, spelling errors, and species splits and lumps often require human judgement. `prepR4pcm` therefore provides tools for curation after the first reconciliation.

Users can correct individual matches with `reconcile_override()` or apply many corrections with `reconcile_override_batch()`. They can also use `reconcile_crosswalk()` to apply a taxonomic crosswalk: a table that links names in one taxonomy to names in another. For example, a crosswalk between BirdLife-based names and Jetz et al. / BirdTree tip labels can show which BirdLife names correspond to tip labels in Jetz et al. bird phylogeny (Jetz et al., 2012; Tobias et al., 2022).

Some studies need to deal with species that are present in the data but absent from the tree. `reconcile_augment()` can add such species to a tree for exploratory analyses, for example, by placing an unresolved species with a close relative. Internally, `prepR4pcm` can also delegate grafting to the `V.PhyloMaker`/`U.PhyloMaker` suits when the required backbone and genus-family information are available (Jin and Qian, 2019, 2022, 2023). This step should be treated as a modelling assumption rather than an observed result. Users should therefore report which species were added and repeat the analysis with and without augmented tips.

The package also supports studies with more than one dataset or more than one tree. `reconcile_data()` reconciles two data frames, `reconcile_merge()` joins reconciled datasets, `reconcile_multi()` reconciles several datasets against one tree, and `reconcile_to_trees()` reconciles one dataset against several trees. These functions are useful when users want to compare alternative trees or combine trait datasets from different sources.

2.5 Tree retrieval and dating

Many users begin with a trait dataset without a suitable phylogenetic tree. `pr_get_tree()` helps in this situation by retrieving a candidate phylogeny from public sources. At present, the package supports five backends: `rotl`, `rtrees`, `clootl`, `fishtree`, and `datelife`. These backends differ in taxonomic coverage, tree sources, and whether they return a single tree or a set of trees.

Before a backend is called, `prepR4pcm` standardises the input species names with `pr_normalize_names()`. This step helps avoid failures caused by simple formatting differences. When taxonomic name resolution changes an input name, the package records the replacement, linking the user's original name to the resolved name. The package also checks that the reported matched and unmatched names remain tied to the user's input list.

Multi-tree support depends on the backend. For example, some sources can return posterior samples of trees, whereas others return a single tree. The function `pr_date_tree()` can add time calibration to a topology using `datelife` where suitable information is available. The function `pr_cite_tree()` formats citations for tree sources in plain text, Markdown, or BibTeX.

2.6 Reporting, comparison, and export

A reconciliation is most useful when it can be inspected, compared, and reported. `reconcile_summary()` prints a detailed summary to the console, and `reconcile_report()` writes a self-contained HTML report. `reconcile_suggest()` proposes near matches for unresolved names, `reconcile_diff()` compares two reconciliation runs, and `reconcile_splits_lumps()` flags cases where one taxonomy treats a name as one species, but another taxonomy treats it as several species, or *vice versa*.

Once users are satisfied with the reconciliation, `reconcile_apply()` returns the two objects required for most PCMs: a data frame and a pruned tree with the same set of species. `reconcile_export()` writes the aligned data, tree, and mapping table to disk. For tree comparisons, `pr_tree_compare()` summarises overlap in tip sets and differences in topology. For phylogenetic meta-analysis, `pr_phylo_cor()` builds a phylogenetic correlation matrix that can be used in packages such as `metafor` (Viechtbauer, 2010). Together, these functions move users from raw species names to auditable, aligned comparative objects (Table 2).

Table 2. The 31 available functions in `prepR4pcm` package, organised by functionalities. Functions prefixed `reconcile_` enable the matching and alignment pipeline; functions prefixed `pr_` enable tree retrieval, dating, comparison, and supporting utilities.

Function	Family	Description
`reconcile_tree`	Reconciliation	Reconcile a data frame against a phylogenetic tree
`reconcile_data`	Reconciliation	Reconcile two data frames
`reconcile_trees`	Reconciliation	Reconcile two phylogenetic trees
`reconcile_to_trees`	Reconciliation	Reconcile one dataset against a list of trees
`reconcile_multi`	Reconciliation	Reconcile multiple datasets against one tree
`reconcile_apply`	Alignment	Produce matched data frame and pruned tree
`reconcile_merge`	Alignment	Join two reconciled datasets
`reconcile_override`	Curation	Manually correct an individual match
`reconcile_override_batch`	Curation	Apply batch corrections from CSV or data frame
`reconcile_crosswalk`	Curation	Convert a taxonomy crosswalk to overrides
`reconcile_augment`	Augmentation	Graft missing species onto a phylogenetic tree for exploratory analyses
`reconcile_mapping`	Inspection	Extract the full mapping tibble
`reconcile_summary`	Inspection	Generate a detailed summary report
`reconcile_plot`	Inspection	Plot match composition (bar or pie chart)
`reconcile_suggest`	Review	Suggest fuzzy candidates for unresolved names
`reconcile_review`	Review	Interactively accept/reject flagged matches
`reconcile_diff`	Review	Compare two reconciliation runs
`reconcile_splits_lumps`	Review	Detect taxonomic splits and lumps
`reconcile_report`	Reporting	Produce self-contained HTML report
`reconcile_export`	Reporting	Write aligned data, tree, and mapping to disk
`pr_get_tree`	Tree retrieval	Fetch a tree from external sources / backends: `rotl`/`rtrees`/`clootl`/`fishtree`/`datelife` (manual selection or auto)
`pr_get_tree_status`	Tree retrieval	Probe which backends are installed and reachable
`pr_date_tree`	Dating	Add DateLife calibration to an existing topology
`pr_cite_tree`	Citations	Format citations for the sources of phylogenies used (select formats: text/Markdown/BibTeX)
`pr_tree_compare`	Comparison	Pairwise Jaccard, Robinson–Foulds, and bipartition-matched branch-length tree agreement
`pr_phylo_cor`	Correlation	Build the phylogenetic correlation matrix from a tree for `metafor`/`MCMCglmm`
`pr_tree_cache_dir`	Cache	Get or set the cache directory
`pr_tree_cache_status`	Cache	List cache entries
`pr_tree_cache_clear`	Cache	Wipe the cache
`pr_normalize_names`	Utility	Standardise scientific-name formatting
`pr_extract_tips`	Utility	Extract tip labels from a phylogenetic tree

3 Worked examples

3.1 Example 1: Reconciling a dataset with an existing tree

We first demonstrate the core workflow using a subset of AVONET bird morphological data (Tobias et al., 2022) and a subset of the Jetz et al. bird phylogeny (Jetz et al., 2012), both bundled with the package. This example is deliberately simple, but it illustrates a common problem in comparative workflows. AVONET uses spaces in species names, whereas the tree uses underscores. Exact matching therefore fails, even when the biological species are the same.

Core workflow: data–tree reconciliation

```
library(prepR4pcm)
library(caper)

# Load the example trait data and phylogenetic tree
data(avonet_subset)
data(tree_jetz)

# Step 1: Reconcile species names between the data and tree
# fuzzy = TRUE allows fuzzy suggestions, but resolve = "flag"
# keeps uncertain matches for review rather than accepting them silently
rec <- reconcile_tree(
x = avonet_subset,
tree = tree_jetz,
x_species = "Species1",
fuzzy = TRUE,
resolve = "flag")

# Step 2: Inspect how names were matched
# This table shows whether names were matched exactly, after
# normalisation, by synonym lookup, by fuzzy matching, or left unresolved
mapping <- reconcile_mapping(rec)
table(mapping$match_type)

# Step 3: Apply the reconciliation
# This returns an aligned data frame and a pruned tree with the
# same species set
aligned <- reconcile_apply(
rec,
data = avonet_subset,
tree = tree_jetz,
species_col = "Species1",
drop_unresolved = TRUE)

# Step 4: Use the aligned objects in a comparative analysis
cd <- comparative.data(
phy = aligned$tree,
data = aligned$data,
names.col = "Species1",
vcv = TRUE)
```

```
model <- pgls(log(Mass) ~ log(Wing.Length),
data = cd)
```

The output of `table(mapping$match_type)` shows that all retained species were matched through normalisation:

```
normalised   unresolved_x
657          262
```

In this example, `reconcile_tree()` compares 919 data names with 657 tree tips. It matches 657 data names to tree tips, all through normalised matching, because the main difference is formatting. The remaining 262 species names from the AVONET data object are unresolved because they are present in the AVONET subset but not in the tree subset. Although fuzzy matching is enabled in the code, no retained species are matched by fuzzy matching in this example. After `reconcile_apply()` is run, both the data frame and the tree contain the same 657 species.

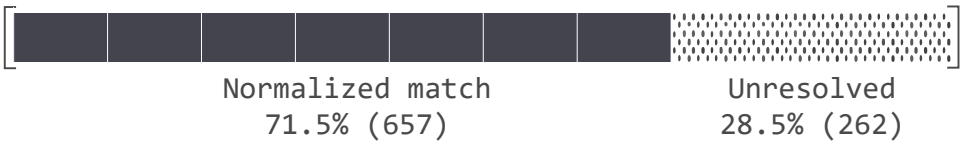
3.2 Inspecting the audit trail

This example illustrates why an audit is useful. A simple final count would tell us that 657 species were retained. The reconciliation object tells us more: it shows that the retained species matched after a formatting change, that no exact matches occurred, and that 262 data rows were removed because they had no matching tree tip in this example.

The same audit trail becomes more important in less tidy examples. The `reconciliation` object can also record synonym matches, fuzzy suggestions, manual overrides, unresolved names, taxonomic sources, database versions, and user-supplied corrections. Users can inspect this information with `reconcile_mapping()`, summarise it with `reconcile_summary()`, or generate an HTML report with `reconcile_report()`.

Figure 2 shows two outputs from this workflow: the match composition and a schematic preview of the HTML report. Together, these outputs show both the numerical result of reconciliation and the record of decisions that led to the final data–tree pair.

a. Match composition



b. Reconcilation report preview

Reconcilation Report

Generated by prepR4pcm

Type	data vs tree
Source x	avonet_subset
Source y	phylo (657 tips)
Authority	col
Reconcilation date	year-month-day hh:mm:ss

Match summary

Match type	Count	% of x
Exact	0	0.0%
Normalized	657	71.5%
Synonym	0	0.0%
Fuzzy	0	0.0%
Manual	0	0.0%
Flagged	0	0.0%
Unresolved (x only)	262	28.5%
Unresolved (y only)	0	
Total matched	**657**	**71.5%**

Normalized matches (657)

Formatting differences resolved automatically.

Name (x)	Name (y)	Notes
Acanthiza apicalis	Acanthiza_apicalis	'Acanthiza apicalis' normalised to 'Acanthiza apicalis'
Acanthiza chrysorrhoa	Acanthiza_chrysorrhoa	'Acanthiza chrysorrhoa' normalised to 'Acanthiza chrysorrhoa'
⋮		

Figure 2. Example outputs from a `prepR4pcm` reconciliation workflow. (a) Match composition for the worked example using an AVONET data subset containing 919 species names and a subset of the Jetz et al. (2012) phylogeny containing 657 tip labels. In this example, 657 data names were matched to the 657 tree tips through normalised matching, mainly because AVONET uses spaces in binomials, whereas the tree uses underscores. The remaining 262 species names in the AVONET data subset were unresolved data-only names, meaning that they were present in the data subset but absent from the tree subset. (b) Schematic preview of the HTML reconciliation report generated by `prepR4pcm`. The report summarises run metadata, source objects, taxonomic authority, match-type counts, unresolved names, and examples of names resolved through normalised matching.

3.3 Example 2: Retrieving trees before reconciliation

The first example assumes that the user already has a phylogenetic tree. In many projects, however, users begin with a trait dataset but without a suitable tree. In this case, `prepR4pcm` can retrieve a candidate tree before the reconciliation step. The code below retrieves a posterior sample of fish trees and formats the relevant citations.

```
# Retrieve 50 trees for three fish species from the fishtree backend
trees <- pr_get_tree(
c("Salmo salar", "Esox lucius",
"Oncorhynchus mykiss"),
source = "fishtree",
n_tree = 50)

# Check that the result is a posterior sample of trees
class(trees$tree)    # "multiPhylo"
length(trees$tree)   # 50

# Format citations for the tree source
cat(pr_cite_tree(trees, format = "markdown"))
```

The retrieved tree object can then be used in the same way as a user-supplied tree. For example, users can pass the retrieved tree to `reconcile_tree()`, inspect the resulting `reconciliation` object, and then use `reconcile_apply()` to create aligned data and tree objects.

The package code, documentation, and worked vignettes are available in the anonymised repository at https://anonymous.4open.science/r/prepR4pcm-2A33/. The vignettes demonstrate data–tree reconciliation, tree retrieval, multi-tree workflows, tree comparison, bird and mammal trait workflows, and phylogenetic meta-analysis.

4 Bundled data

Importantly, `prepR4pcm` includes small example datasets for vignettes, worked examples, and tests. These datasets are not intended to replace the full published datasets. Rather, they provide realistic examples in which species names do not always match perfectly.

The bird data examples include an AVONET data subset (Tobias et al., 2022), a NestTrait subset (Chia et al., 2023), a plumage lightness subset (Delhey et al., 2019), a Jetz et al. tree subset (Jetz et al., 2012), a subset of the complete avian phylogeny mapped to the eBird/Clements Checklist v2025 (McTavish et al., 2025; Clements et al., 2025), and a BirdLife to BirdTree taxonomy crosswalk distributed with AVONET (Tobias et al., 2022; Jetz et al., 2012). These objects support the worked example above and the bird-trait vignette.

The mammal examples include subsets from the Amniote life-history database (Myhrvold et al., 2015), PanTHERIA (Jones et al., 2009), and TetrapodTraits (Moura et al., 2024), along with a mammal tree used for demonstration (Upham et al., 2019) . Users who rely upon these example datasets in their published work should cite the original data providers and, where needed, replace the example objects with the full original datasets.

The examples are useful because they are not perfectly clean. Some species names differ in formatting, some datasets follow different taxonomies, and some species are present in one source but not another. They therefore show the kinds of problems that users often face in real PCM workflows.

5 Discussion

Preparing data for PCMs often begins with an important yet underreported step: matching species names between a dataset and a phylogenetic tree. Researchers usually report the final number of species analysed, but they may not report how many species were lost during matching, which names failed to match, or how ambiguous names were resolved. These missing details make it difficult to reproduce the analysis or to assess whether the final species set could affect the results.

`prepR4pcm` makes this step visible. The package reconciles names, records the decisions, and returns aligned objects for downstream analysis. The key outputs are not only a pruned tree and a matched data frame but also a `reconciliation` object that can be inspected, saved, shared, and reported. This object records exact matches, normalised matches, synonym matches, fuzzy suggestions, unresolved names, and manual corrections. It also allows users to report how many names were matched at each stage, which names remained unresolved, which decisions were corrected manually, and which taxonomic authority or database version was used. The full mapping table can be archived as supplementary material so that readers can trace how the final species set was obtained.

The same idea applies to tree retrieval. When users retrieve trees using `prepR4pcm`, the package records the tree source and the name substitutions made along the way. This record helps users write a clearer methods section and provides a more complete audit trail for the analysis. In this sense, `prepR4pcm` treats name matching, tree preparation, and final data alignment as parts of one workflow rather than as separate preprocessing steps.

Several limitations remain. Fuzzy matching can produce incorrect matches, so users should review fuzzy suggestions before accepting them. Tree augmentation also requires care. Here, tree augmentation means adding species that are present in the dataset but absent from the phylogenetic tree. Such additions may be based on genus-level placement, a close relative, a taxonomic crosswalk, or information from another more detailed published tree. These choices are assumptions, not observed phylogenetic relationships. Users should therefore report which species were added, how they were added, and what source or rule was used. They should also inspect the resulting tree visually, because plotting the augmented tree is often a useful sanity check. Where possible, analyses should be repeated with and without augmented species, and sometimes with alternative augmentation strategies.

Synonym matching also depends on the coverage and version of the taxonomic database used. Users should always report the authority, database version, and any manual changes made during reconciliation. Future versions will expose the Global Names Architecture (Mozzherin et al., 2017) as alternative backends, including `rgnparser` for the parsing step in `pr_normalize_names()` and the Global Names Verifier for synonym resolution against a larger set of sources, alongside the existing `taxadb`-based path.

By making reconciliation and related decisions explicit, `prepR4pcm` helps users move from

raw species names to analysis-ready comparative objects. More importantly, it helps users show how they got there. This audit trail should make PCM workflows easier to verify, repeat, and report.

Author contributions

Author Statement

Conceptualization: SN, BJ, SPH. Data curation: SN, AM, SO. Formal analysis: SN, AM, SO. Funding Acquisition: SN, ML. Investigation: All. Methodology: SN. Project administration: SN, ESAS, SPH. Supervision: SN, ESAS, ML. Validation: All. Visualization: AM. Writing - Original Draft: SN. Writing - Review & Editing: All.

Acknowledgements

We thank the developers of the upstream R packages that `prepR4pcm` wraps or builds on.

Statements

Conflict of interest: The authors declare no conflict of interest.

Peer review: The peer review history for this article is available at [URL].

Data availability: All data and code are available at https://github.com/itchyshin/prepR4pcm. A Zenodo DOI will be added at acceptance.

Use of generative AI

Generative AI tools, including Claude Code and Codex, were extensively used during software development to assist with code drafting, refactoring, documentation, and test development. These tools were not used as independent sources of scientific evidence. AI-assisted code was checked by the authors through unit tests, worked examples, vignettes, manual inspection, and comparisons with expected outputs. The authors take full responsibility for the accuracy of the package, analyses, and manuscript.

ORCID

Shinichi Nakagawa 0000-0002-7765-5182; Malgorzata Lagisz 0000-0002-3993-6127; Santiago Ortega 0000-0002-3518-276X; Bhayvay Jain 0009-0000-1021-4326 Jimuel Jr Celeste 0000-0003-0224-4124; Ayumi Mizuno 0000-0003-0822-5637; Eduardo S. A. Santos 0000-0002-0434-3655 Sergio Poo Hernandez 0000-0003-0155-645X

References

Luis Cayuela, Íñigo Granzow-de la Cerda, Fabio S. Albuquerque, and Duncan J. Golicher. Taxonstand: An R package for species names standardisation in vegetation databases. *Methods in Ecology and Evolution*, 3:1078–1083, 2012. doi: 10.1111/j.2041-210X.2012.00232.x.

Scott A. Chamberlain and Eduard Szöcs. taxize: taxonomic search and retrieval in R. *F1000Research*, 2:191, 2013. doi: 10.12688/f1000research.2-191.v2.

Jonathan Chang, Daniel L Rabosky, Stephen A Smith, and Michael E Alfaro. An r package and online resource for macroevolutionary studies using the ray-finned fish tree of life. *Methods in Ecology and Evolution*, 10(7):1118–1124, 2019.

Stephanie Yuan Chia, Yi-Ting Fang, Yi-Ting Su, Pei-Yu Tsai, Chia Hsieh, Shu-Han Tsao, Jia-Yang Juang, Chih-Ming Hung, and Mao-Ning Tuanmu. A global database of bird nest traits. *Scientific Data*, 10:923, 2023. doi: 10.1038/s41597-023-02837-1.

J. F. Clements, P. C. Rasmussen, T. S. Schulenberg, M. J. Iliff, J. A. Gerbracht, D. Lepage, A. Spencer, S. M. Billerman, B. L. Sullivan, M. Smith, and C. L. Wood. The ebird/clements checklist of birds of the world: v2025. https://www.birds.cornell.edu/clementschecklist/download/, 2025. Downloaded [insert date].

Kaspar Delhey, James Dale, Mihai Valcu, and Bart Kempenaers. Reconciling ecogeographical rules: rainfall and temperature predict global colour variation in the largest bird radiation. *Ecology Letters*, 22(4):726–736, 2019. doi: 10.1111/ele.13233.

Joseph Felsenstein. Phylogenies and the comparative method. *American Naturalist*, 125:1–15, 1985. doi: 10.1086/284325.

Robert P. Freckleton, Paul H. Harvey, and Mark Pagel. Phylogenetic analysis and comparative data: a test and review of evidence. *American Naturalist*, 160:712–726, 2002. doi: 10.1086/343873.

Alan Grafen. The phylogenetic regression. *Philosophical Transactions of the Royal Society of London. Series B*, 326:119–157, 1989. doi: 10.1098/rstb.1989.0106.

Jarrod D. Hadfield and Shinichi Nakagawa. General quantitative genetic methods for comparative biology: phylogenies, taxonomies and multi-trait models for continuous and categorical characters. *Journal of Evolutionary Biology*, 23:494–508, 2010. doi: 10.1111/j.1420-9101.2009.01915.x.

Luke J. Harmon, Jason T. Weir, Chad D. Brock, Richard E. Glor, and Wendell Challenger. GEIGER: investigating evolutionary radiations. *Bioinformatics*, 24:129–131, 2008. doi: 10.1093/bioinformatics/btm538.

Walter Jetz, G. H. Thomas, J. B. Joy, K. Hartmann, and A. O. Mooers. The global diversity of birds in space and time. *Nature*, 491:444–448, 2012. doi: 10.1038/nature11631.

Yi Jin and Hong Qian. V.PhyloMaker: an R package that can generate very large phylogenies for vascular plants. *Ecography*, 42(8):1353–1359, 2019. doi: 10.1111/ecog.04434.

Yi Jin and Hong Qian. V.PhyloMaker2: An updated and enlarged R package that can generate very large phylogenies for vascular plants. *Plant Diversity*, 44(4):335–339, 2022. doi: 10.1016/j.pld.2022.05.005.

Yi Jin and Hong Qian. U.PhyloMaker: An R package that can generate large phylogenetic trees for plants and animals. *Plant Diversity*, 45(3):347–352, 2023. doi: 10.1016/j.pld.2022.12.007.

Kate E. Jones, Jon Bielby, Marcel Cardillo, Susanne A. Fritz, Justin O'Dell, C. David L. Orme, Kamran Safi, Wes Sechrest, Elizabeth H. Boakes, Chris Carbone, Christina Connolly, Michael J. Cutts, Janine K. Foster, Richard Grenyer, Michael Habib, Christopher A. Plaster, Samantha A. Price, Elizabeth A. Rigby, Janna Rist, Amber Teacher, Olaf R. P. Bininda-Emonds, John L. Gittleman, Georgina M. Mace, and Andy Purvis. PanTHERIA: a species-level database of life history, ecology, and geography of extant and recently extinct mammals. *Ecology*, 90(9):2648, 2009. doi: 10.1890/08-1494.1.

Daijiang Li. rtrees: an r package to assemble phylogenetic trees from megatrees. *Ecography*, 2023(7):e06643, 2023. doi: 10.1111/ecog.06643. URL https://github.com/daijiang/rtrees.

Emily Jane McTavish, Jeff A. Gerbracht, Mark T. Holder, Marshall J. Iliff, Denis Lepage, Pam Rasmussen, Benjamin Redelings, Luna L. Sanchez-Reyes, and Eliot T. Miller. A complete and dynamic tree of birds. *Proceedings of the National Academy of Sciences*, 122(18): e2409658122, 2025. doi: 10.1073/pnas.2409658122.

François Michonneau, Joseph W. Brown, and David J. Winter. rotl: an R package to interact with the Open Tree of Life data. *Methods in Ecology and Evolution*, 7:1476–1481, 2016. doi: 10.1111/2041-210X.12593.

Eliot Miller, Emily Jane McTavish, and Luna L. Sanchez-Reyes. clootl: Fetch and explore the cornell lab of ornithology open tree of life avian phylogeny, 2025. URL https://github.com/eliotmiller/clootl.

Ayumi Mizuno, Szymon M. Drobniak, Coralie Williams, Malgorzata Lagisz, Shinichi Nakagawa, Xiang-Yi Li Richter, and Carolin Kosiol. Promoting the use of phylogenetic multinomial generalised mixed-effects model to understand the evolution of discrete traits. *Journal of Evolutionary Biology*, 38:1699–1715, 2025. doi: 10.1093/jeb/voaf116.

Mario R. Moura, Karoline Ceron, Jhonny J. M. Guedes, Rosana Chen-Zhao, Yanina V. Sica, Julie Hart, Wendy Dorman, Julia M. Portmann, Pamela González-del Pliego, Ajay Ranipeta, Alessandro Catenazzi, Fernanda P. Werneck, Luís Felipe Toledo, Nathan S. Upham, João F. R. Tonini, Timothy J. Colston, Robert Guralnick, Rauri C. K. Bowie, R. Alexander Pyron, and Walter Jetz. TetrapodTraits: A comprehensive trait database with imputed values for global tetrapods. *PLOS Biology*, 22(7):e3002658, 2024. doi: 10.1371/journal.pbio.3002658.

Dmitry Y. Mozzherin, Alexander A. Myltsev, and David J. Patterson. "gnparser": a powerful parser for scientific names based on parsing expression grammar. *BMC Bioinformatics*, 18: 279, 2017. doi: 10.1186/s12859-017-1663-3.

Nathan P. Myhrvold, Elita Baldridge, Benjamin Chan, Dhileep Sivam, Daniel L. Freeman, and S. K. Morgan Ernest. An amniote life-history database to perform comparative analyses with birds, mammals, and reptiles. *Ecology*, 96(11):3109, 2015. doi: 10.1890/15-0846R.1.

Shinichi Nakagawa and Eduardo SA Santos. Methodological issues and advances in biological meta-analysis. *Evolutionary Ecology*, 26(5):1253–1274, 2012.

Kari E. A. Norman, Scott Chamberlain, and Carl Boettiger. taxadb: a high-performance local taxonomic database interface. *Methods in Ecology and Evolution*, 11:1153–1159, 2020. doi: 10.1111/2041-210X.13440.

David Orme, Rob Freckleton, Gavin Thomas, Thomas Petzoldt, Susanne Fritz, Nick Isaac, and Will Pearse. *caper: Comparative Analyses of Phylogenetics and Evolution in R*, 2025. R package version 1.0.4.

Emmanuel Paradis and Klaus Schliep. ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R. *Bioinformatics*, 35:526–528, 2019. doi: 10.1093/bioinformatics/bty633.

Matthew W. Pennell, Jonathan M. Eastman, Graham J. Slater, Joseph W. Brown, Josef C. Uyeda, Richard G. FitzJohn, Michael E. Alfaro, and Luke J. Harmon. geiger v2.0: an expanded suite of methods for fitting macroevolutionary models to phylogenetic trees. *Bioinformatics*, 30: 2114–2116, 2014. doi: 10.1093/bioinformatics/btu181.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2025. URL https://www.R-project.org/.

Daniel L. Rabosky, Jonathan Chang, Pascal O. Title, Peter F. Cowman, Lauren Sallan, Matt Friedman, Kristin Kaschner, Cristina Garilao, Thomas J. Near, Marta Coll, and Michael E. Alfaro. An inverse latitudinal gradient in speciation rate for marine fishes. *Nature*, 559:392–395, 2018. doi: 10.1038/s41586-018-0273-1.

Bruno R. Ribeiro, Santiago José Elías Velazco, Karlo Guidoni-Martins, Geiziane Tessarolo, Lucas Jardim, Steven P. Bachman, and Rafael Loyola. bdc: a toolkit for standardizing, integrating and cleaning biodiversity data. *Methods in Ecology and Evolution*, 13:1421–1428, 2022. doi: 10.1111/2041-210X.13868.

Luna L. Sanchez Reyes, Brian C. O'Meara, Joseph W. Brown, and Emily Jane McTavish. DateLife: Leveraging databases and analytical tools to reveal the dated tree of life. *Systematic Biology*, 73(2):470–485, 2024. doi: 10.1093/sysbio/syae015.

Joseph A. Tobias, Catherine Sheard, Alex L. Pigot, et al. AVONET: morphological, ecological and geographical data for all birds. *Ecology Letters*, 25:581–597, 2022. doi: 10.1111/ele.13898.

Nathan S. Upham, Jacob A. Esselstyn, and Walter Jetz. Inferring the mammal tree: Species-level sets of phylogenies for questions in ecology, evolution, and conservation. *PLOS Biology*, 17 (12):e3000494, 2019. doi: 10.1371/journal.pbio.3000494.

Wolfgang Viechtbauer. Conducting meta-analyses in r with the metafor package. *Journal of statistical software*, 36:1–48, 2010.