Implementation of the SInAS workflow

The individual steps of the workflow have been implemented in the open statistical software R, version 3.6.1 (R Core Team 2019), using the `rgbif` (Chamberlain et al. 2019), `openxlsx` (Schauberger and Walker 2019) and `data.table` packages (Dowle and Srinivasan 2019). In the following, the requirements to run the workflow and the implementation of the workflow are described in detail.

All parts of the workflow are open source and can be modified by the user, but we strongly recommend to report all modifications of the provided codes and tables together with the versions of the individual databases in the publication of the respective results to ensure transparency and reproducibility.

Requirements:

Applying the workflow requires that: 1) R software and required packages are installed; 2) the workflow scripts and folder structure are obtained; 3) the configuration file is adjusted; and 4) the original databases to be merged are available. A stable internet connection is required to run the taxonomic standardisation.

The workflow scripts can be run on Linux, Windows and macOS systems.

1. The R environment:

R needs to be installed on the computer and can be freely obtained from https://cran.r-project.org. Running the workflow requires three R packages (`rgbif`, `openxlsx` and `data.table`) and their dependencies, which need to be installed as well. This can be done by executing (copy->paste->enter) the command `install.packages(c('rgbif','openxlsx','data.table'))` into the R terminal.

2. Workflow scripts and tables

The R scripts with the implemented workflow can be obtained from "https://github.com/hseebens/sTwist_Workflow/archive/1.2.zip" (or later versions). The provided zip file comprises the required folder structure with the subfolders `R/`, `Config/` and `Inputfiles/`. Running the workflow for the first time will generate an additional subfolder `Output/`, where all output files are stored. The R scripts in the subfolder `R/` do not need to be modified by the user. The subfolder `Inputfiles/` contains the individual databases, which are going to be merged. The subfolder `Config/` includes nine tables in Office Open XML-based spreadsheet format (.xlsx), which can be read by R.

The subfolder `Config/` includes a configuration file ("DatabaseInfo.xlsx"). This file contains the basic information about the characteristics of the individual databases such as file name, and names of columns where to find the respective information necessary to run the workflow. For each database to be merged the user has to fill in the required information (see below).

The location table ("AllLocations.xlsx") contains a list of location names and is used as a reference for the standardisation of location names. A proposed list of locations is included in this file but any list of region names can be added as long as the basic information of four columns including region names, country IDs, 2-digit ISO codes and keywords is provided (in the columns "RegionID", "Region", "keywords" and "ISO2", respectively). The keywords represent a selection of alternative spellings of location names or geographic sub-units, which will be used to identify the correct match. Sub-units will be aggregated to the provided higher level. Multiple keywords within the same cell of the spreadsheet have to be separated by "; " as these can be identified as a separator by the workflow. Keywords should be selected carefully to avoid duplicated matches (e.g., "United" would match several countries). The selection of locations is based on the 2-digit ISO code (ISO 3166-1 alpha-2), extended by a few locations to meet the demands of the studies in the field of biological invasions. In addition, the list contains information about continents, WGSRPD regions on level one and four, Global Administrative Areas (GADM) names, GloNAF regions, island/mainland distinction and 3-digit ISO codes. The list of region names can be changed by the user by modifying, adding or removing entries. Note that the entries in the column "RegionID" have to be complete and adjusted in case of modification. Otherwise, the changes will not be recognised.

The table "UserDefinedTaxonNames.xlsx" provides taxonomic information alternative to the one obtained from the GBIF backbone taxonomy. Here, information about unresolved taxa or deviating taxonomic concepts can be included. By modifying this table, entries obtained from GBIF will be overwritten. All replacements for location and taxon names are stored in the translation tables as an output file of the standardisation steps (see below).

Rules for treating first record entries can be defined in the table "Guidelines_eventDates.xlsx". In this table, the user can enter the term to be processed and the respective replacement. The replacement is exact, which means that the term in column "Entry" will be replaced exactly by the entry in column "Replacement". An empty field in the column "Replacement" will remove the entry in the column "Entry" without any substitution. Examples based on Dyer et al. (2017b) are provided in the table.

In addition, five translation tables are provided, one each for five variables of the Darwin Core terminology related to alien species. Four have been proposed and refined by Groom et al. (2019), namely establishmentMeans, occurrenceStatus, degreeOfEstablishment and pathway. The fifth, habitat, has already been part of the standard Darwin Core terminology. For all five variables, the translation tables provide two columns, one with the term used in the original database and one with its replacement. However, deviations of the definitions of individual terms cannot be standardised within this workflow and thus the user should pay attention to varying definitions. If possible, standard definitions as provided by Darwin Core should be used.

3. Adjusting the configuration file

As a minimum requirement to run the scripts, the user has to adjust the configuration file "DatabaseInfo.xlsx", as only those databases will be considered in the workflow, which are added to this file. Information stored in the configuration file consists of three categories: required, optional and additional. The headers of the columns, which contained the information, are called "variables".

- ❏ *Required variables*: These variables are required to run the workflow. These are short names of databases, file names stored on the disk, columns of taxon names and location names in each original database. This is the minimum amount of information that is required for the standardisation.
- ❏ *Optional variables*: This information will be standardised or used within the workflow if provided, but is not required to successfully execute the workflow. For instance, information about the taxonomic group covered in the database can be used to check multiple entries in GBIF, which would be otherwise exported as missing matches of species names. Optional columns contain establishmentMeans, occurrenceStatus, degreeOfEstablishment, pathway, habitat, taxonomic group (currently included: "All", "Amphibians", "Reptiles", "Birds", "Insects", "Mammals", "Vascular plants"), taxonomic authority, kingdom and event date (start and end, or only start; numeric).
- ❏ *Additional variables*: These columns are kept as they are provided in the original database without affecting the workflow. Multiple column names can be provided in a single column separated by semicolon. Allowing for additional variables is a convenient way to include information in the final database, which can then be used for purposes other than the integration of databases.

4. Databases of alien species

Databases of alien species occurrences have to be provided as spreadsheet files (only *.xlsx), with all required information included on the first sheet. All information has to be strictly formatted as a table (first row: column headers; following rows: database entries; each row corresponds to one entry; no empty lines; only the first sheet will be read in). All databases have to be stored in the subfolder `Inputfiles/`. Databases have to be provided as .xlsx files. Note that currently only .xlsx (no .xls) files are allowed, as the R package `openxlsx` can only handle these formats.

Implementation of the workflow in R

The workflow can be executed by running the R script `runWorkflow.r` in the `R/` subfolder, which calls the full sequence of the workflow. This is most easily done by first opening the R project "sTWISTworkflow" (double click on `sTWISTworkflow.Rproj` in the parent directory), which sets the path to the working directory. Afterwards, start the workflow by e.g. pasting the command `source('R/runWorkflow.r')` into the R terminal.

In the following, the implementation of the workflow is described in detail, although this knowledge is not required to apply the workflow and run the scripts.

Step 0. Start workflow (R script `runWorkflow.r`)

At this very first step, the required R packages and functions of the workflow are loaded and executed in sequence. Messages of current status of the workflow progress and warnings if applicable are printed to the R terminal. At this point, the user can define the name of the final output and a version number. In addition, the user can select whether intermediate output should be stored or deleted after a successful run of the scripts.

Step 1. Load and prepare individual databases (R script `PrepareDatasets.r`)

This represents the first step of the actual workflow. A new subfolder called `Output/` with the subfolder `Intermediate/` and `Check/` are created, if these do not exist already. After successfully running the workflow, the parent folder `Output/` will contain the final merged database, a full list of taxa and a list of translated location names. Results of each intermediate step will be stored in `Intermediate/`, while information useful for cross-checking such as unresolved terms, missing location and taxon names will be saved in `Check/`. At this step, the original databases are loaded and prepared for further processing. This includes a formal check if required information provided in the configuration files is provided, and an error is printed if this is not the case. Following the information in the configuration file, columns of the respective content are identified and the column headers are renamed in a consistent way. Column entries of separated taxon names (i.e., genus, species and authors in different columns) are merged into a single column. All databases with now standardised column headers are exported.

Step 2a. Standardisation of terminologies (R script `StandardiseTerms.r`)

At this step, terms of the variables establishmentMeans, occurrenceStatus, degreeOfEstablishment, pathway and habitat are standardised by following the Darwin Core terminology and the proposed refinements by Groom et al. (2019). Standardisation is done by replacing the original terms with the standardised ones as provided in the respective translation table. In case terms of the original databases could not be found in the translation tables, a list of unresolved terms (i.e., mis-matches) are exported for reference and cross-checking.

Step 2b. Standardisation of location names (R script `StandardiseLocationNames.r`)

In the second step of the workflow, location names are standardised based on the list of provided location names (“AllLocations.xlsx”). This will replace the original location names, which is done by first identifying exact matches and subsequently by checking keywords if provided. To ease matching of location names and keywords, some basic cleaning is performed such as removal of “the” or trimming spaces. Matching is done in lowercase letters to avoid conflicts due to varying usages of cases. A translation table of original and new region names is stored and non-matching regions are exported in a table of missing region names.

Step 2c. Standardisation of taxon names (R script `StandardiseTaxonNames.r`)

In the second step of the workflow, taxon names are compared to the GBIF backbone taxonomy using the R function `name_backbone()` from the package `rgbif` (Chamberlain et al. 2019). If the name could be found in the GBIF backbone, taxonomic information is obtained including scientific name, status (accepted or synonym) and higher classification. The actual taxonomic check is performed in a sub-function called `CheckGBIFTax()`, which is part of the workflow implementation. Only taxon names listed as accepted in GBIF are considered. For each match of the provided taxon name and the GBIF taxonomy, the following information is kept: the scientific name, taxonomic tree, type of matching (exact or fuzzy, see below) and the rank of the taxon.

The following rules are applied to check for a taxon match:

- Exact matches: The match of the provided taxon name with the GBIF entry has to be exact (i.e., matchType in the GBIF output is set to “EXACT”). If an exact match could be found, the accepted taxon name and associated information is reported back (Status=”ACCEPTED” in the R function `name_backbone()`).
- Synonyms: If the taxon name could not be exactly matched, there is a check for entries as synonyms (i.e., status=”SYNONYM”, matchType=”EXACT” in the R function `name_backbone()`). As scientific names including the authority are not provided for synonym records by GBIF, the supplementary information called ‘alternatives’ is checked for exact matches of accepted names. If this does not retrieve a result, a new request is sent to GBIF now including the accepted taxon name to obtain the correct taxonomic information.
- Fuzzy matches: If neither an exact match nor a synonym is available for a taxon, fuzzy matches are allowed to avoid mismatched due to minor spelling errors. Fuzzy matches are provided by GBIF and we accept a fuzzy match with a high level of confidence of 100, which removes minor spelling errors.
- Homonyms: Homonyms represent taxon names, which are used identically for different taxa. The chance of obtaining false results due to homonyms can be reduced by providing full scientific names including the authority, which is, however, not always the case. Without author names, differentiation can only be done by accessing additional information provided by the user in the configuration file. The workflow allows for incorporating additional taxonomic information such as the kingdom. This can be done by providing information about the kingdom in the original database and information about the taxonomic group of which currently the following are accepted: “Vascular plants”, “Reptiles”, “Amphibians”, “Birds” and “All”. The respective column headers need to be indicated in the configuration file under “Column_kingdom” and “Taxon_group”, respectively. If an exact match of a taxon name is found multiple times in the GBIF output “alternatives”, the kingdom and/or family names are compared to ensure that both describe the same species. Homonyms are flagged as such in the file “Taxa_FullList.csv” column “GBIFnote” for further checking.
- Alternative names: In few cases, GBIF provides synonyms only in the output “alternatives”. In this case, taxon names are obtained from this file if a single record providing an accepted taxon name could be identified. If multiple taxon names are provided, the original taxon name is kept and this instance is flagged as having multiple taxon names in the file “Taxa_FullList.csv”.
- If none of the above matches the requested species name, the original taxon name is kept and exported as a taxon name missing in GBIF for further checking. Note that missing species names are kept in the database, but are flagged in the final database as “Missing” in the column “GBIFStatus”.

After cross-checking with the GBIF taxonomy, the user can specify taxonomic information in a spreadsheet file (“UserDefinedTaxonNames.xlsx”), which will replace the information included so far. Taxonomic information about higher taxonomic levels and/or authors can be added by entering the same taxon name for old and new versions plus additional taxonomic information. In this case, the taxon name is kept, while the new taxonomic information is added.

At the end of this step, all databases with now standardised taxon names are exported together with a full list of original and new taxon names with taxonomic information and a list of missing names. Note that unresolved taxon names are kept in the final database.

Step 2d. Standardisation of event dates (R script: `GeteventDate.r`)

In the fourth step of the workflow, years of first record, here called event dates, are standardised. Event dates provided as single years are retained. In cases of a provided time ranges with two dates, their mean is used as the event date. In cases deviating from these formats, the respective entry needs to be translated into a numeric value. These translations have to be explicitly formulated by the user in table "Guidelines_eventDates.xlsx", which provides examples mostly following the rules determined in Dyer et al. (2017b). Non-matching entries will be exported and treated as missing values in the final database.

Step 3. Merging all standardised databases (R script: `MergeDatabases.r`)

In the final step, all databases are merged based on the standardised location names and taxon names. This is done by first merging the column entries of the standardised databases. In cases of conflicting entries, deviating entries are concatenated and stored in one cell of the final database. After merging all columns of standardised databases, duplicated entries are removed. A duplicated entry is defined as two rows in the final database with exactly the same taxon and location name. Other entries of these rows may deviate. If this is the case, duplicated entries are merged by concatenating their entries. For example, if a species is reported as "introduced" in one database and "uncertain" in another, the final output for this cell would be "introduced; uncertain". A warning will inform the user of existing conflicts, why may be solved by modifying the content of the original databases. All duplicated entries will be removed in this way. The resulting merged database contains unique entries of standardised location and taxon names, event dates and terminologies for Darwin Core variables if provided.