

# revtools: An R package to support article screening for evidence synthesis

Martin J. Westgate 

Fenner School of Environment & Society,  
The Australian National University,  
Acton, ACT, Australia

## Correspondence

Martin J. Westgate, Fenner School of  
Environment & Society, The Australian  
National University, Acton, ACT 2601,  
Australia.  
Email: martin.westgate@anu.edu.au

The field of evidence synthesis is growing rapidly, with a corresponding increase in the number of software tools and workflows to support the construction of systematic reviews, systematic maps, and meta-analyses. Despite much progress, however, a number of problems remain, including slow integration of new statistical or methodological approaches into user-friendly software, low prevalence of open-source software, and poor integration among distinct software tools. These issues hinder the utility and transparency of new methods to the research community. Here, I present revtools, an R package to support article screening during evidence synthesis projects. It provides tools for the import and deduplication of bibliographic data, screening of articles by title or abstract, and visualization of article content using topic models. The software is entirely open-source and combines command-line scripting for experienced programmers with custom-built user interfaces for casual users, with further methods to support article screening to be added over time. revtools provides free access to novel methods in an open-source environment and represents a valuable step in expanding the capacity of R to support evidence synthesis projects.

## KEYWORDS

data visualization, meta-analysis, natural language processing, systematic review, topic models

## 1 | INTRODUCTION

Evidence synthesis is increasingly used to help inform policy responses to critical societal issues.<sup>1</sup> Further, the growth of evidence synthesis methodologies—predominantly systematic reviews (SRs), but also including systematic maps and meta-analyses—has been concurrent with a massive increase in the amount of published scientific material, estimated at 50 million articles in 2010<sup>2</sup> and now growing at over 2.5 million articles a year.<sup>3</sup> In combination, these dual pressures have focused attention on new methods to support SR, with a vast number of software tools and workflows now available for researchers.<sup>4,5</sup> Despite much progress, however,

several problems have emerged in the SR software landscape during this rapid expansion. In particular, many software tools have limited interoperability with one another, or hide the details of the algorithms they provide to assist with article screening and related tasks, problems that are incentivized in commercial environments. Further, new statistical or computational methods are not always made available to users in a timely or easily accessible way, hindering their application to real problems.<sup>6</sup> Addressing these issues would greatly assist progress in the application of novel techniques to evidence synthesis projects.<sup>7</sup>

In recent years, an alternative software development model has emerged that draws on community-maintained

programming languages such as R or Python to support bespoke, open-source software tools. These tools have the advantage that they can be provided for free, and often more quickly than they can be integrated into commercial software.<sup>8</sup> Further, the open-source nature of these tools means they are able to build on software libraries built by other developers, allowing rapid adaptation and specialization. While these changes have had some impact on evidence synthesis workflows—and particularly on meta-analysis, where many software tools are now available<sup>9</sup>—there has been more limited development of tools to support data search and screening. This is problematic because those tasks form the foundation of effective SRs.

In this paper, I present *revtools*, an R package for importing, manipulating and screening bibliographic data. The package serves a dual function of providing robust infrastructure to support current SRs and meta-analyses, while also supporting the development of more advanced software tools, including novel data visualization tools to facilitate article screening (see Table 1 for a description of key functions). Although developed entirely within a command-line-based environment, *revtools* has been designed to be as user-friendly as possible by including interfaces that launch in the users' web browser. These interfaces are built using the R packages *Shiny*<sup>10</sup> and *shinydashboard*,<sup>11</sup> removing any dependencies on external software. Functions are also provided to help experienced programmers to explore the underlying models and data in more detail. Users can download the stable version of *revtools* from CRAN (<https://cran.r-project.org/package=revtools>); view, download, or contribute to the development version on GitHub (<https://github.com/mjwestgate/revtools>); or view further documentation and examples on a dedicated website (<https://revtools.net>). Below, I describe the functionality provided by *revtools* in order of a typical SR workflow.

## 2 | FUNCTIONALITY

### 2.1 | Data import and export

When conducting an SR, it is common to have bibliographic data stored in files generated by a bulk download of some kind, typically from an academic database such as Web of Science or Scopus. Bibliographic data from these sources can be stored in a range of formats (with BibTeX and RIS formats among the most common), and while there are R packages for importing BibTeX data (eg, RefManager<sup>12</sup>), there is no function to identify the format of a bibliographic data file and import it in a consistent manner. This is problematic for tagged styles for which no standard formatting exists, for example where

### Highlights

- Article screening is one of the most time-consuming stages of a systematic review.
- While many screening tools are available, few provide visual summaries of article content to facilitate rapid removal of irrelevant content.
- Here, I present *revtools*, an R package for data import, deduplication and screening in evidence synthesis projects.
- *revtools* allows key tasks to be performed interactively in the browser, meaning programming knowledge is not required.
- This software provides a framework for expanding the availability of efficient tools for article screening in the R ecosystem.

two files have identical content but different filename suffixes (such as *.ciw* and *.ris*) or have the same suffix but different formats (common with *.ris*, for example). In *revtools*, the function *read\_bibliography* autodetects tag data and article delimiters and converts them into a *data.frame* (the standard format for R datasets) with human-readable column names. If the dataset is already stored in a spreadsheet-like *csv* file, then *read\_bibliography* imports the file as is, with a small number of changes to ensure consistency with other formats. Finally, users can export bibliographic data from R using the function *write\_bibliography*.

Although *read\_bibliography* returns a *data.frame* by default, it does so by means of an intermediate object stored in the new class *bibliography*. Accessing this data class is optional and will not be necessary for most users, but is included because its list-based structure allows easy storage of nested information structures, as is common in bibliographic datasets. Specifically, each *bibliography* object is a list in which each entry is an article, consisting of another list storing standard bibliographic data such as the title, author, or keywords for that reference. A priority for future development is to facilitate deeper levels of nestedness where required, such as data on author affiliations or ORCID ID numbers. That functionality would be very useful for integrating bibliographic datasets with (co)citation data, in the context of “research weaving” projects, for example.<sup>13</sup> In anticipation of this added functionality, class *bibliography* is fully supported as an S3 class with its own *print*, *summary*, and subset (“[” and “[[”) functions and can be converted to a *data.frame* using *as.data.frame* or back again using *as.bibliography*.

**TABLE 1** Key functions available in *revtools*

Category	Function	Description
Data manipulation	<i>read_bibliography</i>	Import bibliographic data from a range of formats into an R <i>data.frame</i>
	<i>write_bibliography</i>	Export bibliographic data from R into .ris or .bib formats
	<i>merge_columns</i>	Combine ( <i>rbind</i> ) two <i>data.frames</i> with different columns into a single <i>data.frame</i>
Duplicate detection	<i>find_duplicates</i>	Identify potentially duplicated references within a <i>data.frame</i>
	<i>extract_unique_references</i>	Return a <i>data.frame</i> that contains only “unique” references
	<i>screen_duplicates</i>	Manually screen potential duplicates identified via <i>find_duplicates</i>
Text mining and visualization	<i>make_dtm</i>	Take a vector of strings and return a document-term matrix (DTM)
	<i>run_topic_model</i>	Run a topic model with specified properties
	<i>screen_topics</i>	Use topic models to visualize and screen articles
Distributing tasks among a team	<i>allocate_effort</i>	Assign proportions of articles to members of a team
	<i>distribute_tasks</i>	Use results from <i>allocate_effort</i> to split a <i>data.frame</i> , returning a .csv file for each reviewer
	<i>aggregate_tasks</i>	Combined screened references back into a single <i>data.frame</i>
Manual screening	<i>screen_titles</i>	Screen articles by title
	<i>screen_abstracts</i>	Screen articles by abstract

## 2.2 | Identification and removal of duplicates

A key component of SR protocols is that researchers should search a number of databases to ensure that all available literature is detected.<sup>14</sup> This approach is very thorough but has the side effect of introducing duplicates into the dataset that must be removed prior to article screening. Rather than building an automated function that makes many decisions on behalf of the user (which is difficult to optimize<sup>15</sup>), *revtools* implements a fuzzy-matching algorithm for duplicate detection. This approach is more general and flexible but requires some care to return sensible results.

The main function for identifying duplicates in *revtools* is *screen\_duplicates*, which runs a graphical user interface (GUI) in the browser that allows calculation and manual screening of duplicates. Alternatively, the user can call the underlying function—called *find\_duplicates*—directly from the workspace, as both functions use the same arguments. First, the user selects a variable from their dataset

that they want to investigate for potential duplicates, such as the article titles or DOIs. Second, they select any “grouping” variables that defines a subset of the data where duplicates should be sought. For example, the user may wish to only seek matching titles within the same journal or the same publication year. Leaving this argument blank leads to more duplicates being located but takes longer to run. Third, the user specifies a method for matching potential duplicates, either an exact match, or fuzzy matching using either *stringdist*<sup>16</sup> or the new function *fuzzdist*, which implements functions from the *fuzzywuzzy* Python library (<https://github.com/seatgeek/fuzzywuzzy>). Finally, the user can set the threshold for deciding whether two strings (which may contain titles, DOIs, etc) are identical and choose whether to convert strings to lower case or to remove punctuation. If the user has called *screen\_duplicates*, setting these arguments and hitting the “calculate duplicates” button will lead to a display of potential duplicated entries, allowing the user to interactively choose between them. If instead they have

called *find\_duplicates*, the function will return a vector where duplicated articles share the same value. The user can then pass this vector to *screen\_duplicates* for manual screening or use *extract\_unique\_references* to attempt the deduplication for them. Future versions of *revtools* will supplement these methods with further options for automation of article deduplication.

### 2.3 | Visualizing search results using text mining

A common problem in SRs is that search results from academic databases nearly always contain information that is not relevant to the target question. While tools to improve the sensitivity and specificity of article searches exist, this problem is difficult to avoid entirely because many fields lack well-defined tagging ontologies to improve search efficiency (with Medical Subject Headings [MeSH] being the obvious exception), while semantic properties of natural languages ensure that any given keyword is likely to be used in a range of contexts.<sup>17</sup> What is needed, therefore, is a method that allows the user to gain an a priori impression of the dominant themes within their search results<sup>18</sup> and to filter out irrelevant content prior to manual screening.

In *revtools*, the new function *screen\_topics* attempts to address this gap by visualizing the results of a topic model<sup>19</sup> run on user-selected bibliographic data. This allows the user to decide how many themes (or “topics”) they would like to identify within the corpus and which data should be used to parameterize the model (typically some combination of titles, keywords, and abstracts). Once the model has been calculated, the main window shows a scatterplot in which each point represents one article, with points colored according to the highest-weighted topic for that article (Figure 2). Also shown is a bar chart showing the number of articles within each topic, which also serves as a key to the words that characterise those topics. If article abstracts are available in the dataset, these are shown below the main panel. Finally, an expandable sidebar gives access to user options, including changing between article or word displays, changes to the color scheme using the viridis family of palettes,<sup>20</sup> or saving progress in a range of formats. By default, the *screen\_topics* uses articles as its unit of analysis; but the user can change this behavior, for example, to look for themes among journals or years.

The main value of *screen\_topics* is interacting with bibliographic data in an intuitive way to exclude irrelevant material, using diagrams drawn using the *plotly* R package.<sup>21</sup> Points or bars that are selected bring up further information that can be used to select or exclude a given article,

word, or topic. One critical feature is that all data are linked, meaning that the user can exclude uninformative content and rerun the topic model at any time, unlike most related tools that require topic models to be precalculated (eg, *LDavis*<sup>22</sup>). While useful, this approach can lead to slow performance when models are recalculated, even accounting for default settings that prioritize speed over accuracy (ie, five topics, 10 000 iterations). These settings are adequate for a cursory investigation, but users that wish to exclude groups of articles based on their assigned topic, or that require robust classification of the whole corpus, will need to increase these values. More broadly, using topics as a basis for article inclusion or exclusion is subjective and may introduce bias. This approach should only be used for classifying articles in a very broad sense—such as whether they are within or outside of the target discipline—and should not be considered a replacement for careful screening of individual articles during SR.

If users wish to explore topic models in more detail, they can run their own models using *run\_topic\_model*. This is currently a wrapper function to the *topicmodels* library,<sup>23</sup> but other methods of topic identification will be added in future. The main input to *run\_topic\_model* is a document-term matrix, which can be constructed using the new function *make\_dtm*. This function takes the user-specified text data and applies several standard text processing functions from the *tm* package,<sup>24</sup> namely, conversion to lower case; removal of punctuation, numbers, and “stop” words (such as “the” and “and”); stemming; and the removal of short words (less than three letters) or rare terms (present in less than 1% of articles). After collating all words that contain the same stem, this function returns whichever of the original terms was most common in the dataset, rather than the stem itself (ie, “manag” might be returned as “management”). These functions allow users to duplicate the methods used by *screen\_topics* in the R workspace and facilitate increased flexibility to explore model performance, for example, by comparing the fit of different model types, or models containing different numbers of topics. This is useful if the topic model and its results are themselves the focus of the investigation.<sup>25</sup>

### 2.4 | Dividing screening tasks among a team

SRs are often conducted by teams of researchers who work together to ensure high standards within the screening process. A common approach is “double-screening,” allocating each article to two separate reviewers and comparing the results, which helps to ensure consistency in the application of screening criteria and that disagreements about article relevance are discussed openly, leading to reduced



bias.<sup>14</sup> In large reviews, however, or where time is limited, teams may instead choose to use partial double-screening, whereby only a subset of articles are double-screened and the remainder screened once only. Either way, it is important that teams of reviewers have a means of dividing a dataset of articles among participating researchers.

In *revtools*, users can decide how much work they will each perform using the function *allocate\_effort*, including specifying the size of their team, the proportion of articles to be checked by more than one reviewer, and the maximum number of times each article should be checked. It is also possible—though optional—to specify reviewer names and to allocate different proportions of articles to each reviewer. Users can either call *allocate\_effort* directly, or via the higher-level function *distribute\_tasks*, which also requires a bibliographic dataset as input and (by default) outputs a set of *.csv* files with the required properties. Although the user sets the proportion of articles to be reviewed by each person, the identity of the articles in each subset is chosen at random and so is sensitive to external settings such as *set.seed*. Once the reviews have been completed (using *screen\_abstracts*, for example), they can be recombined into a single *data.frame* using the *aggregate\_tasks* function.

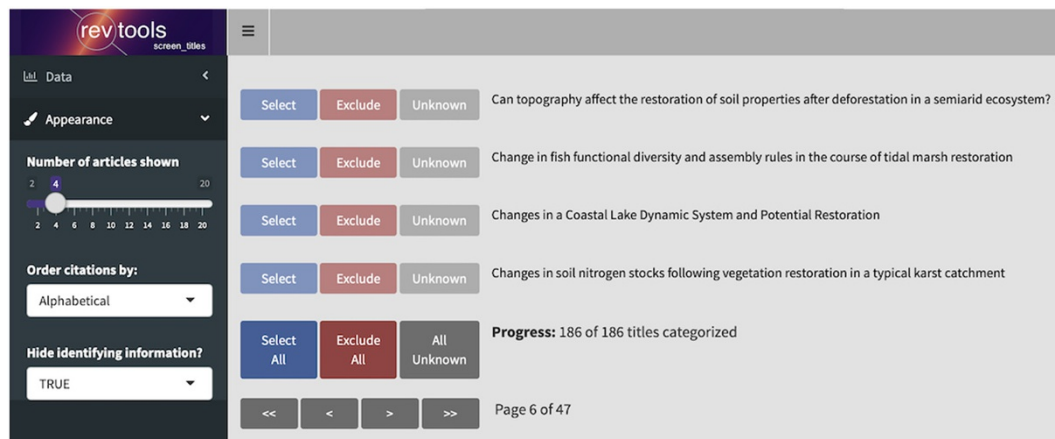
## 2.5 | Manual article screening

The final stage of the SR process that can be completed using *revtools* is manual screening of articles to determine their relevance to the review. Users can choose to screen by title only using *screen\_titles* or to screen article titles and abstracts simultaneously using *screen\_abstracts*. The latter function is superficially similar to related tools in the *metagear* package,<sup>26</sup> with the exception that the *revtools* equivalent is presented using *Shiny* for consistency with *screen\_duplicates* and *screen\_topics*, and to

reduce dependence on external software. Both interfaces have broadly similar content: they both allow the user to drag-and-drop a dataset into the app for screening; the user can choose to show or hide information that identifies the author and journal for consistency with SR guidelines; and they can choose the order in which articles are presented. The text color changes to reflect the users' decisions as articles are selected or excluded, and both apps allow the user to export their decisions to an external file or back to the workspace for further processing in R. The only major difference between them is in their displays; *screen\_abstracts* shows only one article at a time, whereas by default *screen\_titles* shows eight titles at once (Figure 1).

## 3 | EXAMPLE: SCREENING DIVERSE DATASETS

To demonstrate the capabilities available in *revtools*, here, I provide a worked example of how the tools described above may be integrated to solve real problems in evidence synthesis projects. To generate an example dataset, I searched Web of Science, Scopus, and PubMed on 30 January 2019 for all articles containing the term “restoration” in the title that had been published so far that year. These searches returned 247, 286, and 104 articles, respectively, or 637 articles in total. The term “restoration” was chosen as it was given as an example of homonymy in a related paper on article classification,<sup>27</sup> while the time limits were chosen as a simple means of reducing the size of the dataset for demonstration purposes. A real SR search would include a more sophisticated combination of search terms and would not be restricted to such a narrow time window.



**FIGURE 1** Screenshot of the *screen\_titles* interface. Article titles are taken from the example “restoration” dataset (see text) [Colour figure can be viewed at [wileyonlinelibrary.com](http://wileyonlinelibrary.com)]

The analysis begins by installing *revtools*, and then reading the three datasets into R using *read\_bibliography*, as follows:

```
# install and load package
install.packages("revtools")
library(revtools)

# import data from working directory
data_list <- read_bibliography(list.files())

# optional: retain only columns with few missing values
keep_cols <- which(
  apply(data_unique, 2, function(a) {
    length(which(!is.na(a)))
  }) > 300
)
data <- data[, keep_cols]
```

As there is a high likelihood that these three databases will contain some overlapping references, the next logical step is to search for and remove those duplicates. We could achieve this interactively using *screen\_duplicates* or in the command line as follows:

```
# find duplicated DOIs within the dataset
doi_match <- find_duplicates(data,
  match_variable = "doi",
  group_variables = NULL,
  match_function = "exact"
)

# automatically extract one row per duplicate
data_unique <- extract_unique_references(data,
  doi_match)
```

```
nrow(data_unique) # n = 491, i.e. duplication rate
= 23%
```

```
# an alternative is to try fuzzy title matching
title_match <- find_duplicates(data,
  match_variable = "title",
  group_variables = NULL,
  match_function = "stringdist",
  method = "osa",
  threshold = 5
)

length(unique(title_match)) # n = 492; one less
than DOI
# but also slower than exact matching, especially for
large datasets
```

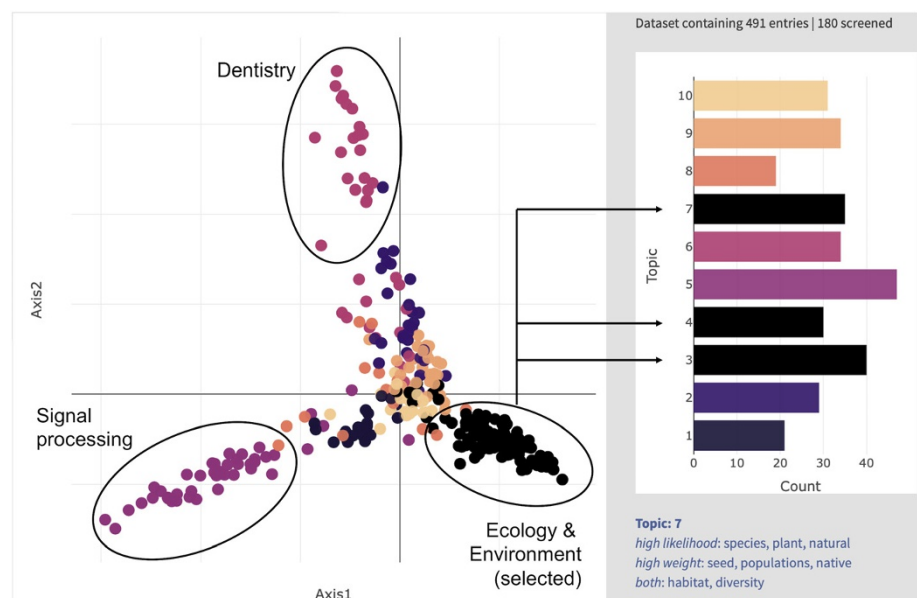
The final stage is to use *screen\_topics* to look for themes in the corpus. Importantly, if we specify an object for this function to return to, then we can save and reload our progress from within R, as follows:

```
result <- screen_topics(data_unique)
```

In this case, we conduct a journal-level analysis and include article titles, abstracts, and keywords in our topic model. Figure 2 shows the resulting ordination after calculation 10 topics and removing a set of additional stopwords, as well as the original search term (“restoration”).

This analysis shows that a number of discrete disciplines are represented—including engineering, medicine, signal processing, and dentistry—while three separate topics in this ordination refer to different uses of the term “restoration” in ecological or environmental contexts. Using the “topics” barplot in *screen\_topics* (right-hand side of Figure 2), we can “select” these topics as being of

**FIGURE 2** Annotated screenshot of *screen\_topics* on the “restoration” dataset, showing one point per journal and 10 topics. Topics containing environmental keywords have been selected for further screening and are shown in black [Colour figure can be viewed at [wileyonlinelibrary.com](http://wileyonlinelibrary.com)]



further interest, while leaving the remaining topics uncategorized. Given the high degree of overlap among topics in the center of the scatterplot, however, a more risk-averse alternative might be to exclude clearly irrelevant topics (such as signal processing and dentistry, topics 5 and 6, respectively) and manually screen the remaining articles. By using the “exit app” button in the “data” tab of the sidebar, these selections are passed back to the workspace. The resulting object (of class “*screen\_topics\_progress*”) contains our selections as an additional column added on to our source dataset and are stored in the section named “raw”:

```
articles_data <- result$raw
env_articles <- articles_data[which (articles_data
  $selected), ]
nrow (env_articles) # n = 186
```

It is then a simple process to screen these articles manually using *screen\_titles* or *screen\_abstracts* or to investigate the themes within this subset of articles in more detail using *screen\_topics* for a second time.

## 4 | DISCUSSION

In this paper, I have presented *revtools*, a new R package to support article screening for evidence synthesis projects. As well as providing basic infrastructure such as support for a range of bibliographic data formats and tools for de-duplication and article screening, *revtools* supports tools that draw on text mining to explore patterns in bibliographic data. These tools can rapidly summarize topics and locate relevant information within any set of article abstracts, irrespective of academic discipline.

While tools for visualizing text data are not yet common in SRs, expanding their use will be critical if SRs are to remain viable given projected increases in the size of the academic literature. Research from the environmental sciences shows that the average SR currently requires 164 person-days of work.<sup>28</sup> While comparable data are not available for medical SRs, current data suggest that these take an average of five people 67 weeks to complete.<sup>29</sup> Further increases in publication rates will make these approaches unviable for most researchers or teams; indeed, some authors have argued that this has already happened, specifically for understanding impacts of rare interventions for biodiversity conservation.<sup>30</sup> Without access to robust time-saving software tools, evidence synthesis researchers risk facing a marked increase in the so-called “synthesis gap,” an incapacity of the scientific establishment to synthesize its’ own research for application to societal problems.<sup>7</sup>

While *revtools* is already a well-developed software package, there are a number of ways in which its functionality could be expanded to improve efficiencies in evidence synthesis. One priority is to allow users to test the effect of fitting covariates to topic models as implemented in the *stm* R package<sup>31</sup>; another is to explore different methods of displaying articles in 2D space, such as the t-SNE algorithm.<sup>32</sup> Both of these approaches could address one shortcoming of the current topic model infrastructure in *revtools*, namely, that these models take a long time to run, particularly for large datasets. A more advanced feature would be to implement some form of dynamic article recommendation, whereby user choices are used to inform recommendations about potentially relevant information. This approach is widely accepted<sup>33</sup> and a range of implementations exist,<sup>27,34</sup> although few of them have been rigorously tested.<sup>6</sup> Besides making these tools more widely available, therefore, one advantage of incorporating them into R would be to facilitate comparisons among these distinct but related approaches. Finally, access to the tools provided by *revtools* would be greatly improved if they were available outside of R, a feature made more achievable by their construction as *Shiny* apps that are relatively easy to host online. This development would greatly increase access by users unwilling or unable to work in a scripting environment and will be a priority for future software releases.

Despite plans to expand the functionality of *revtools*, it is not intended that it should become a standalone package capable of supporting the entire evidence synthesis pathway. Instead, the features provided by *revtools* are likely to be particularly powerful when used in combination with those of related packages. Currently, for example, it is possible to plan a search strategy with *litsearchr*,<sup>35</sup> screen your articles with *revtools*, download articles using *fulltext*,<sup>36</sup> analyze the resulting data in *metafor*<sup>37</sup> or one of the many other meta-analysis packages available in R,<sup>9</sup> and present a database of study results using *EviAtlas*.<sup>38</sup> While this is encouraging, many stages of the SR workflow remain unsupported within R, with data extraction being a key gap. To address this, several of the developers of the packages listed above are working together to construct an integrated package named *metaverse* to guide users through a standard SR workflow. This project will also focus on building new software to integrate existing tools and fill the gaps between them. It is intended, therefore, that *revtools* will continue to grow but retain a focus on article screening and visualization of textual data sources.

## ACKNOWLEDGEMENTS

N. Haddaway contributed substantial discussion and insight regarding the design of *revtools*. Those discussions

were supported by an International Outgoing Visiting Fellowship to MW, awarded by the Australian Research Councils' Centre of Excellence for Environmental Decisions. Comments from P. Barton and three anonymous reviewers greatly improved this manuscript.


## CONFLICT OF INTEREST

The author reported no conflict of interest.

## DATA AVAILABILITY STATEMENT

The data used in the "example" section of this paper are openly available on GitHub at <https://github.com/mjwestgate/revtools/tree/master/examples>.

## ORCID

Martin J. Westgate  <https://orcid.org/0000-0003-0854-2034>

## REFERENCES

- Donnelly CA, Boyd I, Campbell P, et al. Four principles to make evidence synthesis more useful for policy. *Nature*. 2018; 558(7710):361-364. <https://doi.org/10.1038/d41586-018-05414-4>
- Jinha AE. Article 50 million: an estimate of the number of scholarly articles in existence. *Learn Publ*. 2010;23(3):258-263. <https://doi.org/10.1087/20100308>
- Bornmann L, Mutz R. Growth rates of modern science: a bibliometric analysis based on the number of publications and cited references. *J Assoc Inf Sci Technol*. 2015;66(11):2215-2222. <https://doi.org/10.1002/asi.23329>
- Kohl C, McIntosh EJ, Unger S, et al. Online tools supporting the conduct and reporting of systematic reviews and systematic maps: a case study on CADIMA and review of existing tools. *Environ Evid*. 2018;7(1):8. <https://doi.org/10.1186/s13750-018-0115-5>
- Marshall C, Brereton P. Systematic review toolbox: a catalogue of tools to support systematic reviews. In *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*. 2015. ACM
- O'Mara-Eves A, Thomas J, McNaught J, Miwa M, Ananiadou S. Using text mining for study identification in systematic reviews: a systematic review of current approaches. *Syst Rev*. 2015;4(1):5. <https://doi.org/10.1186/2046-4053-4-5>
- Westgate MJ, Haddaway NR, Cheng SH, McIntosh EJ, Marshall C, Lindenmayer DB. Software support for environmental evidence synthesis. *Nat Ecol Evol*. 2018;2(4):588-590. <https://doi.org/10.1038/s41559-018-0502-x>
- Wallace BC, Lajeunesse MJ, Dietz G, et al. OpenMEE: intuitive, open-source software for meta-analysis in ecology and evolutionary biology. *Methods Ecol Evol*. 2017;8(8):941-947. <https://doi.org/10.1111/2041-210X.12708>
- Polanin JR, Hennessy EA, Tanner-Smith EE. A review of meta-analysis packages in R. *J Educ Behav Stat*. 2017;42(2):206-242. <https://doi.org/10.3102/1076998616674315>
- Chang W, Cheng J, Allaire J, Xie Y, McPherson J (2016) Shiny: web application framework for R. R Package version 0.14.2. <https://CRAN.R-project.org/package=shiny>
- Chang W & Ribeiro B Borges (2017) shinydashboard: create dashboards with 'Shiny'. R package version 0.6.1 <https://CRAN.R-project.org/package=shinydashboard>
- McLean MW. RefManagerR: import and manage BibTeX and BibLaTeX references in R. *J Open Source Softw*. 2017;2:338. <https://doi.org/10.21105/joss.00338>
- Nakagawa S, Samarasinhe G, Haddaway NR, et al. Research weaving: visualizing the future of research synthesis. *Trends Ecol Evol*. 2018;34(3):224-238. <https://doi.org/10.1016/j.tree.2018.11.007>
- Collaboration for Environmental Evidence (2018) *Guidelines for Systematic Review and Evidence Synthesis in Environmental Management*. Version 5.0., A.S. Pullin, G.K. Frampton, B. Livoreil, and G. Petrokofsky, Editors. Environmental Evidence: Bangor, United Kingdom.
- Rathbone J, Carter M, Hoffmann T, Glasziou P. Better duplicate detection for systematic reviewers: evaluation of systematic review assistant-deduplication module. *Syst Rev*. 2015;4(1):6. <https://doi.org/10.1186/2046-4053-4-6>
- Van der Loo MP. The stringdist package for approximate string matching. *The R Journal*. 2014;6(1):111-122.
- Westgate MJ, Lindenmayer DB. The difficulties of systematic reviews. *Conserv Biol*. 2017;31(5):1002-1007. <https://doi.org/10.1111/cobi.12890>
- Crisan A, Munzner T, Gardy JL. Adjutant: an R-based tool to support topic discovery for systematic and literature reviews. *Bioinformatics*. 2018;35(6):1070-1072. <https://doi.org/10.1093/bioinformatics/bty722>
- Blei DM. Probabilistic topic models. *Commun ACM*. 2012;55(4):77-84. <https://doi.org/10.1145/2133806.2133826>
- Garnier S (2017). viridis: default color maps from 'matplotlib'. R package version 0.4.0 <https://CRAN.R-project.org/package=viridis>
- Sievert C (2018). plotly for R. <https://plotly-r.com>
- Sievert C & Shirley K (2015) LDavis: interactive visualization of topic models. R package version 0.3.2. <https://CRAN.R-project.org/package=LDavis>
- Grün B, Hornik K. topicmodels: an R package for fitting topic models. *J Stat Softw*. 2011;40(13):1-30. <https://doi.org/10.18637/jss.v040.i13>
- Meyer D, Hornik K, Feinerer I. Text mining infrastructure in R. *J Stat Softw*. 2008;25(5):1-54. <https://doi.org/10.18637/jss.v025.i05>
- Westgate MJ, Barton PS, Pierson JC, Lindenmayer DB. Text analysis tools for identification of emerging topics and research gaps in conservation science. *Conserv Biol*. 2015;29(6):1606-1614. <https://doi.org/10.1111/cobi.12605>
- Lajeunesse MJ. Facilitating systematic reviews, data extraction and meta-analysis with the metagear package for R. *Methods Ecol Evol*. 2016;7(3):323-330. <https://doi.org/10.1111/2041-210X.12472>
- Roll U, Correia RA, Berger-Tal O. Using machine learning to disentangle homonyms in large text corpora. *Conserv Biol*. 2017;32(3):716-724. <https://doi.org/10.1111/cobi.13044>



28. Haddaway NR, Westgate MJ. Predicting the time needed for environmental systematic reviews and systematic maps. *Conserv Biol.* 2019;33(2):434-443. <https://doi.org/10.1111/cobi.13231>
29. Borah R, Brown AW, Capers PL, Kaiser KA. Analysis of the time and workers needed to conduct systematic reviews of medical interventions using data from the PROSPERO registry. *BMJ Open.* 2017;7(2):e012545. <https://doi.org/10.1136/bmjopen-2016-012545>
30. Sutherland WJ, Wordley CF. A fresh approach to evidence synthesis. *Nature.* 2018;558(7710):364-366. <https://doi.org/10.1038/d41586-018-05472-8>
31. Roberts ME, Stewart BM & Tingley D (2018). stm: R package for structural topic models. R package version 1.3.3 <http://www.structuraltopicmodel.com>
32. Donaldson J (2016). tsne: T-distributed stochastic neighbor embedding for R (t-SNE). R package version 0.1-3. <https://cran.r-project.org/package=tsne>
33. Hemens BJ, Iorio A. Computer-aided systematic review screening comes of age. *Ann Intern Med.* 2017;167(3):210-211. <https://doi.org/10.7326/M17-1295>
34. Cheng SH, Augustin C, Bethel A, et al. Using machine learning to advance synthesis and use of conservation and environmental evidence. *Conserv Biol.* 2018;32(4):762-764. <https://doi.org/10.1111/cobi.13117>
35. Grames E, Stillman A, Tingley M & Elphick C (2019). litsearchr: automated search term selection and search strategy for systematic reviews. R package version 0.1.0. <https://github.com/elizagrames/litsearchr>
36. Chamberlain S (2018) fulltext: full text of 'scholarly' articles across many data sources. R package version 1.0.1. <https://CRAN.R-project.org/package=fulltext>
37. Viechtbauer W. Conducting meta-analyses in R with the metafor package. *J Stat Softw.* 2010;36(3):1-48.
38. Haddaway NR, Feierman A, Grainger MJ, et al. EviAtlas: a tool for visualising systematic map databases. *Environ Evid.* 2019;8(22). <https://doi.org/10.1186/s13750-019-0167-1>

**How to cite this article:** Westgate MJ. revtools: An R package to support article screening for evidence synthesis. *Res Syn Meth.* 2019;10:606–614. <https://doi.org/10.1002/jrsm.1374>