https://doi.org/10.51408/issi2025_134

Biblium: An Advanced Python Library for Bibliometric and Scientometric Analysis

Lan Umek[1], Dejan Ravšelj[2]

[1]*lan.umek@fu.uni-lj.si*, [2]*dejan.ravselj@fu.uni-lj.si*
University of Ljubljana, Faculty of Public Administration, Gosarjeva 5, SI-1000 Ljubljana (Slovenia)

Abstract

This paper introduces Biblium, a Python library designed to perform comprehensive bibliometric and scientometric analysis. Biblium replicates the core functionalities of the widely-used R package, Bibliometrix, while expanding its capabilities with several innovative features that enhance group analysis and visualization. This new tool aims to address the growing need for advanced, flexible, and reproducible bibliometric workflows within the Python ecosystem. A key distinguishing feature of Biblium is its ability to conduct group analysis, a functionality critical for analyzing bibliographic data that naturally divides into subsets based on factors such as publication periods, scientific disciplines, or geographic regions. Biblium implements multiple algorithms for comparative group analysis, enabling users to investigate associations between authors, keywords, sources, and other bibliometric elements across these subgroups. This group-level granularity facilitates insights into patterns of collaboration, thematic evolution, and differential impact across fields or timeframes. In addition to descriptive analysis, Biblium includes predictive algorithms that can forecast group membership based on key bibliometric indicators such as keywords, references, or citation patterns. This predictive modeling offers a forward-looking perspective on emerging research clusters and thematic areas. Biblium also empowers users to define custom concepts by leveraging keywords, abstracts, and other textual data, allowing for flexible and targeted analyses. The library features advanced visualizations, such as scatterplots that integrate multiple performance indicators (e.g., total number of citations, H-index, average year of publication, etc.) for authors, sources, and countries. This level of visualization extends the interpretative depth of bibliometric data and aids in more intuitive exploration and presentation of results. To support robust reporting and dissemination of findings, Biblium offers extensive options for exporting analysis outputs in formats including docx, html, tex, xlsx, and pptx. This versatility ensures seamless integration with academic publishing workflows and diverse dissemination platforms. Biblium also expands on traditional bibliometric indices by incorporating a variety of metrics beyond the H-index, providing a more nuanced evaluation of academic performance and influence. Biblium is expected to be publicly available on GitHub by June 2025, fostering an open-source community around bibliometric analysis in Python. This paper will detail the technical architecture, core algorithms, and key use cases of Biblium, demonstrating its application across different bibliometric scenarios. By providing a powerful, user-friendly alternative to existing tools, Biblium aims to accelerate bibliometric research across disciplines.

Introduction

In recent years, bibliometric and scientometric analyses have gained significant importance as essential tools for understanding trends, impact, and collaborations within scientific research. The increasing volume of scholarly publications and the need to assess the quality and influence of research outputs have made these analyses useful for researchers, policymakers, and institutions alike. For researchers, especially those who are new to a field, bibliometric tools provide a valuable overview of key trends, influential works, and potential collaborators, serving as a

crucial starting point for gaining insights and navigating the landscape of scientific literature.
As the demand for bibliometric insights grows, so does the need for robust tools that can process and analyze bibliographic data efficiently. Many software solutions and libraries have been developed to address this demand, with a notable example being the Bibliometrix library in R (Aria & Cuccurullo, 2017a). Bibliometrix provides a comprehensive suite of tools for bibliometric analysis, including descriptive statistics, network analysis, and visualization, making it a widely adopted resource for researchers across disciplines.
Parallel to the evolution of bibliometric tools, Python has emerged as a dominant programming language for data analysis and scientific computing. Its versatility, extensive library ecosystem, and ease of use have made Python the preferred choice for researchers and developers in various fields. Despite its widespread adoption, Python still lacks a comprehensive library that mirrors the functionalities offered by Bibliometrix. Moreover, existing software solutions often fall short in effectively analyzing (sub)groups of bibliographic data and leveraging data mining techniques for deeper insights. This gap underscores the need for innovative Python-based solutions to cater to the growing demand for bibliometric tools. This paper addresses that gap by introducing Biblium, a powerful Python library that brings advanced bibliometric analysis capabilities, replicates key functionalities of Bibliometrix, and introduces novel features—particularly for (sub)group analysis and data mining—to meet the evolving needs of the research community.
The paper is structured as follows: it begins with an overview of existing software solutions for bibliometric analysis, highlighting their capabilities and limitations, with a particular focus on Python-based libraries. This is followed by the introduction of Biblium. The final section discusses future directions and potential enhancements for Biblium, aiming to address current limitations and expand its utility for the bibliometric research community.

Software for bibliometric research

Bibliometric analysis relies on various software tools to process and visualize research data. This chapter provides an overview of key solutions, structured in two parts. The first section briefly introduces the most significant general-purpose bibliometric software, focusing on widely used and impactful tools rather than an exhaustive list. The second section delves into Python-based solutions, offering a detailed exploration of their functionalities, advantages, and implementation for bibliometric studies.

General-purpose bibliometric software

Various tools are available for bibliometric analyses, each with distinct strengths in processing, visualizing, and exploring data. VOSviewer (van Eck & Waltman, 2010) is widely used for its intuitive interface and visualization features. It creates bibliometric maps based on co-citation, co-authorship, and co-occurrence data. Compatible with databases like Web of Science, Scopus, and PubMed, it is particularly effective for visualizing large-scale networks and identifying research

clusters. The same authors implemented CitNetExplorer (van Eck & Waltman, 2017), a software solution which specializes in analyzing and visualizing citation networks. It allows interactive exploration of citation relationships and integrates with VOSviewer.
CiteSpace (Chen, 2006), focuses on trend analysis and detecting emerging topics. It identifies influential papers and key turning points in research fields using citation burst detection and network analysis, providing insights into the evolution of scientific domains.
Bibliometrix and its web-based interface, Biblioshiny (Aria & Cuccurullo, 2017b) offer comprehensive bibliometric capabilities. As an R package, Bibliometrix integrates science mapping, statistical analysis, and network exploration, while Biblioshiny provides an accessible interface for users without programming skills.
The Sci2 Tool (Team, 2009) supports advanced network and temporal analysis. It visualizes citation networks, collaboration patterns, and temporal trends. SciMat (Cobo et al., 2012) is tailored for longitudinal analysis and thematic evolution. It identifies research trends over time, focusing on knowledge progression and its influence across different periods. More detailed overview of best software solutions for bibliometric analysis can be found in (Moral-Muñoz et al., 2020).

Python libraries for bibliometric analysis

Metaknowledge (McLevey & McIlroy-Young, 2017) is one of the earliest Python-based tools for bibliometric analysis. The package offers various analytical capabilities, including longitudinal analysis, standard and multi-reference publication year spectroscopy, computational text analysis (e.g., topic modeling and burst analysis), and network analysis. One notable feature is its ability to estimate researcher gender by retrieving the Global Name Dataset from Open Gender Tracker's GitHub repository (*OpenGenderTracking*, 2013) and matching author and co-author names with probable genders.
Tethne (Peirson, 2016) is a Python-based tools for bibliometric analysis, developed to facilitate computational research in network science and bibliometrics. It was designed with a focus on co-citation, bibliographic coupling, and co-authorship analysis. Tethne provides functionalities for handling bibliometric data sourced from Web of Science (WoS) and Scopus. A key strength of Tethne is its integration with NetworkX, which allows users to analyze citation and collaboration networks effectively. However, its development has slowed down, and it lacks support for advanced natural language processing.
Pybliometrics (Rose & Kitchin, 2019) is a powerful Python library designed for bibliometric research with data sourced exclusively from Scopus. Unlike earlier tools, it offers direct access to Scopus API, allowing for large-scale data retrieval and analysis. Pybliometrics provides functions for citation counts, author productivity analysis, and institutional impact metrics. While it lacks built-in machine learning or NLP functionalities, it is used due to its efficient and programmatic approach to bibliometric research.
Scientopy (Ruiz-Rosero et al., 2019) is a relatively recent addition to the bibliometric analysis landscape. It offers comprehensive tools for analyzing bibliometric data

from WoS and Scopus, including citation analysis, co-authorship networks, and keyword trends. Scientopy is recognized for its ease of use and ability to generate detailed descriptive statistics and visualizations of scientific output over time. While it provides solid bibliometric functionalities, it does not incorporate sophisticated NLP techniques.

Litstudy (Heldens et al., 2022) was developed as an efficient Python package to assist researchers in conducting literature reviews. It supports the retrieval and processing of scientific metadata from multiple sources, including Scopus and other repositories. Its primary strengths lie in text mining and citation analysis, allowing users to extract key terms, identify trends, and map research landscapes.

TechMiner (Velasquez, 2023) is a Python-based graphical application tool that is useful for analyzing Scopus data by cleaning, renaming, and extracting relevant information while standardizing text formats. It offers analytical modules, including descriptive statistics, citation and co-word analysis, collaboration and conceptual mapping, term clustering, growth indicators, and impact assessments (H and M-index). Advanced tools like factor, correlation, latent semantic, and main path analysis enhance bibliometric insights. Additional features include thematic analysis, time-based tracking, top document ranking, and global visualization via a world map.

PyBibX (Pereira et al., 2025) is the most advanced Python library for bibliometric and scientometric analysis, incorporating cutting-edge artificial intelligence tools. It supports data from Scopus, WoS, and PubMed, providing comprehensive exploratory data analysis (EDA), citation, collaboration, and similarity networks. A major innovation of PyBibX is its AI-driven capabilities, including embedding vectors, topic modeling, and text summarization. It integrates models such as Sentence-BERT, BERTopic, BERT, chatGPT, and PEGASUS to enhance bibliometric insights. PyBibX stands out as the first bibliometric tool to feature AI-driven conversational analytics, allowing researchers to interact with bibliometric results dynamically.

None of these Python libraries can analyze groups of documents from bibliographic datasets, a key limitation that differentiates them from more advanced bibliometric tools. While some, like Metaknowledge, Tethne, and Pybliometrics, provide functionalities for network analysis and citation metrics, their capabilities are restricted to specific tasks such as co-citation analysis or institutional impact assessment. Many of these tools, except for TechMiner and PyBibX, lack advanced NLP (and AI-driven analytics), making them significantly less comprehensive than Bibliometrix in R.

Bibliometric analysis with Biblium

Biblium is a tool designed for the analysis and visualization of bibliographic data. In this section, we will describe its basic functionalities, showing how it can assist in exploring and interpreting scientific publications. To demonstrate its capabilities, we will use a sample dataset comprising the 500 most-cited documents related to bibliometric, scientometric, and informetric research. This dataset serves as an illustrative example to highlight Biblium's functionalities.

Currently, Biblium is hosted on a private GitHub repository, accessible to selected collaborators for review and testing. Plans are in place to make the repository publicly available by June 2025 on the site: https://github.com/lan-umek/biblium

Python Libraries for Biblium

In Biblium, a variety of Python libraries were employed to handle bibliographic data, perform statistical analyses, and generate visualizations. The primary data structure used was the Pandas DataFrame, which served as the main object for storing bibliographic data and output dataframes, such as performance measurements and scientific production metrics (Mckinney, 2010). Pandas was extensively utilized for data transformations, including sorting, filtering, and computing new features, while NumPy provided foundational support for numerical operations (Harris et al., 2020). For statistical analysis (statistical tests, clustering, entropy calculation), Scipy.stats was employed (Virtanen et al., 2020), and predictive modeling tasks, such as logistic regression, were conducted using Scikit-learn (Pedregosa Fabian et al., 2011) and Statsmodels (Seabold & Perktold, 2010).
Visualization is important aspect of Biblium, with Matplotlib (Hunter, 2007) and Seaborn (Waskom, 2021) being used for creating plots, Plotly for k-field plots and geographical maps (Inc., 2015), Squarify for treemaps (Laserson, 2009), and upsetplot (Nothman, 2023).
Network analysis and visualization were achieved using NetworkX (Hagberg et al., 2008), complemented by CDlib (Community Detection Library) for partitioning algorithms (Rossetti et al., 2019). Additionally, Pyvenn (Tctianchi, 2014) was used for generating Venn diagrams, while adjustText (Flyamer, 2012) ensured clear labeling on graphs by preventing overlap and properly positioning labels.
Text processing tasks, such as lemmatization, stop words removal, and n-gram analysis, were handled by Natural Language Toolkit NLTK (Bird et al., 2009), while Gensim (Rehurek & Sojka, 2011) was employed for topic modelling. For image manipulation, the PIL library was utilized (Umesh, 2012), and Wordcloud (Mueller, 2010) was used to generate word clouds.

Initialization

The provided code snippet demonstrates the initialization process of Biblium's core BibliumAnalysis class.

```
>>> import biblium as bb
>>> ba = bb.BiblioAnalysis(f_name="data.csv", db="scopus")
```

The BiblioAnalysis class in Biblium can be initialized with either a file name (e.g., "data.csv") or a Pandas DataFrame. The db argument, specifying the database source, is currently limited to "wos" (Web of Science) and "scopus", with support for additional databases expected to be added soon. Biblium supports input files in csv, xlsx, and txt formats, and plans are in place to accommodate more file types in the future.

Although db is a keyword argument and not strictly required during initialization, failing to provide this information will result in a BiblioAnalysis instance with almost no meaningful or interesting results computed. Providing the correct database information is essential to unlock the full potential of the tool. Biblium adopts the terminology used in the Scopus database. If data from a different database are provided, Biblium automatically adjusts column names to align with the closest matching Scopus terms.
In the initialization process user can change several additional keyword arguments, the most important being the pre_compute argument and res_folder. The pre_compute parameter controls the level of preprocessing and descriptive statistics performed during initialization. Setting pre_compute=1 computes basic descriptive statistics, while increasing the value up to pre_compute=4 adds progressively more advanced computations, including lemmatization of abstracts, the computation of new derived features, and other detailed analyses.
When the res_folder parameter is provided, a directory is created with structured subfolders: networks, plots, tables, and reports, where results are saved in various formats. This ensures an organized structure of the results.
In addition to these core parameters, various other optional parameters can be customized during initialization. These include settings for the default color scheme, the resolution of png plots to be saved, the language of the output, and more. While these parameters are not described in the paper, they will be fully detailed in the Biblium tutorial and documentation available on GitHub. This ensures users have the flexibility to adapt Biblium to their specific needs and preferences.

Main Information
By calling

```
>>> ba.get_main_info()
```

two dataframes are computed: main_info_df and production_df. The main_info_df includes basic publication metrics such as the total number of documents, sources, and citations, as well as details like the number of documents per source and the proportion of cited documents. Temporal aspects include ranging from the first and last years of publication, the period length, and most productive year, to other statistics like the average year, standard deviation of publication year, and quartiles of publication years (Q1, median, Q3). Growth trends are quantified through metrics such as overall document growth and growth rates over the past year, five years, and ten years. Citation analysis includes the highest number of citations for a document, the H-index, and G-index, alongside average citation metrics for all and cited documents specifically. The analysis also highlights productivity by most frequent sources, countries, and keywords.
Authorship and collaboration metrics enrich the analysis with insights into the number of unique authors, documents per author, and co-authorship patterns, including single-author and multi-author documents. These are further detailed through collaboration indices and international collaboration trends. Reference data

provides an understanding of citation behavior, including references per document, correlation between publication year and average reference year, and the distribution of unique references. Language and accessibility metrics describe the most frequent document languages, the number of multi-language publications, and open-access availability.

The production_df data frame captures both annual and cumulative statistics related to scientific output (number of documents, total number of citations). It details the number of documents published each year, allowing researchers to observe patterns of growth or decline in productivity over time.

The scientific production is plotted using the

```
>>> ba.plot_production(cut_year=2000)
```

which includes a cutoff at the year 2000 to reduce the wide spread of data and create a more visually representative graph. The plot shows the yearly number of documents as bars, alongside a line representing the cumulative number of citations over time (Figure 1).

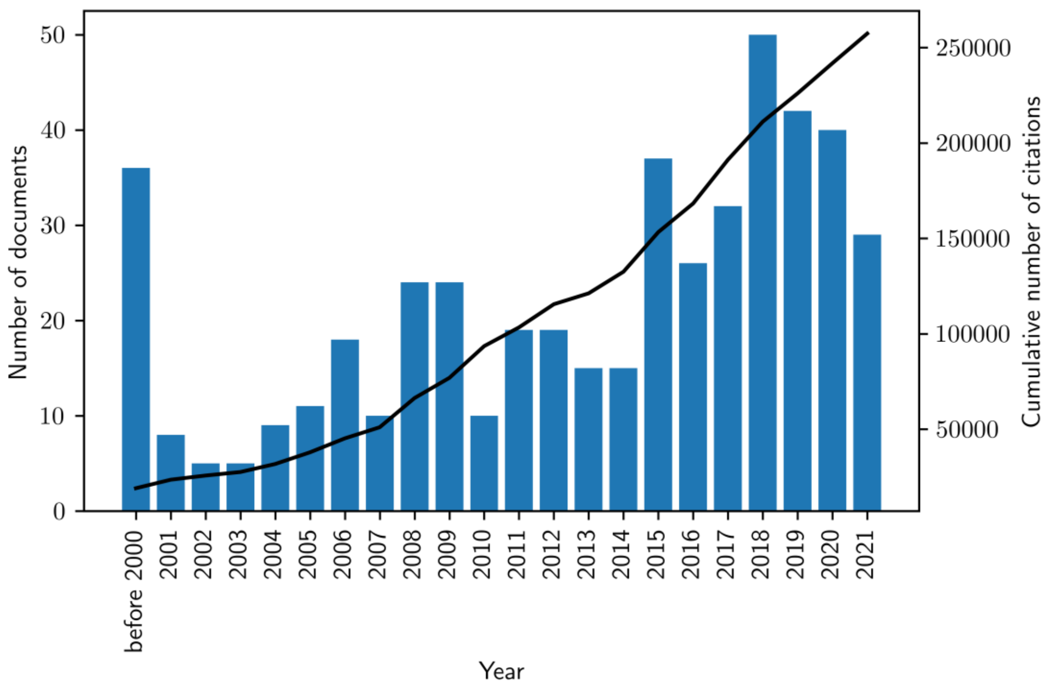

Figure 1. Scientific production plot from Biblium: annual number of documents and cumulative citations over time, with a cut-off year applied for improved visualization.

Measuring Performance

In Biblium, performance indicators are implemented to analyze scientific production across various units of observation, such as sources, authors, countries of corresponding authors, references, keywords, scientific fields, etc. These functionalities allow users to explore patterns and trends in scholarly output by focusing on specific dimensions, like the distribution of contributions by country, the prominence of particular keywords, or the impact of sources and references.

The process involves two stages. First, Biblium counts the number of occurrences for each unit of observation. Next, these counts are used to compute performance indicators for a user defined subset of units. The following snippet is implemented to count the occurrences of sources and compute additional statistics for the top 20 sources based on the number of documents (Users can define subsets based on specific criteria, utilize subsets from other dataframes, or even employ regular expressions for the selection.). This function takes additional parameter "level", ranging from 0 to 4, which determines the extent of the computed statistics. The selection of items for statistical analysis in Biblium is impressively versatile. At level 0, only the source counts are returned. Higher levels progressively compute more detailed metrics, level 4 includes all implemented statistics, offering a comprehensive analysis of source performance. This flexible approach allows users to tailor the depth of analysis to their specific needs.

```
>>> ba.count_sources()
>>> ba.get_sources_stats(top=20, level=1)
```

At level 1, the code computes essential statistics, including total citations, the average publication year, and the H-index, providing a solid foundation for evaluating source performance. Level 2 expands the analysis with additional metrics, such as the G-index, cumulative citation counts (C5, C10, ..., C100), and publication year distribution statistics like the first quartile (Q1), median, and third quartile (Q3). At level 3, the code computes the interdisciplinarity by calculating the normalized entropy of counts related to different scientific disciplines, offering insights into the diversity of a source's contributions. Level 4 focuses on advanced or specialized indices, such as the HG-index (Alonso et al., 2010), m quotient (Hirsch, 2005), Tapered H-index (Anderson et al., 2008), A-index and R-index (Jin et al., 2007), and q^2-index (Cabrerizo et al., 2010). These metrics cater to nuanced evaluation needs and are ideal for more sophisticated analyses. Many other advanced indices will be introduced in future updates, enhancing the scope of Level 4 statistics and providing users with more comprehensive analytical tools. The code is designed to be general and adaptable, extending beyond the evaluation of sources to other units of observation, such as authors, countries of corresponding authors, references, keywords, and scientific fields.

The computed statistics are stored in a pandas DataFrames named sources_counts_df and sources_stats_df, which provides a structured format for further analysis and visualization. This DataFrame can be easily leveraged for various plotting purposes, enabling users to explore the data visually. One of the most insightful visualizations is a scatterplot, as it can incorporate up to four indicators simultaneously. In Biblium it can be plotted using the default parameters by calling

```
>>> ba.scatter_plot_top_sources()
```

By default, the scatter_plot_top_sources function visualizes the number of documents on the x-axis and total citations on the y-axis, both on logarithmic scales. The H-index determines marker size, while the average publication year defines marker color, and abbreviated source titles serve as labels. Additionally, one categorical property can be shown with different shapes of the dots. The plot includes the top 20 sources based on the number of documents. Optional features like mean lines (dashed line indicating the means of variables on both axes), mean values, and regression lines are disabled by default. Users can further customize annotations with arrow properties, ensuring the visualization is both insightful and adaptable to different datasets. The scatterplot is saved in the “plots” folder in three different formats: png, pdf, and svg. For illustration, a scatterplot produced on the dataset used in the paper is shown in Figure 2.

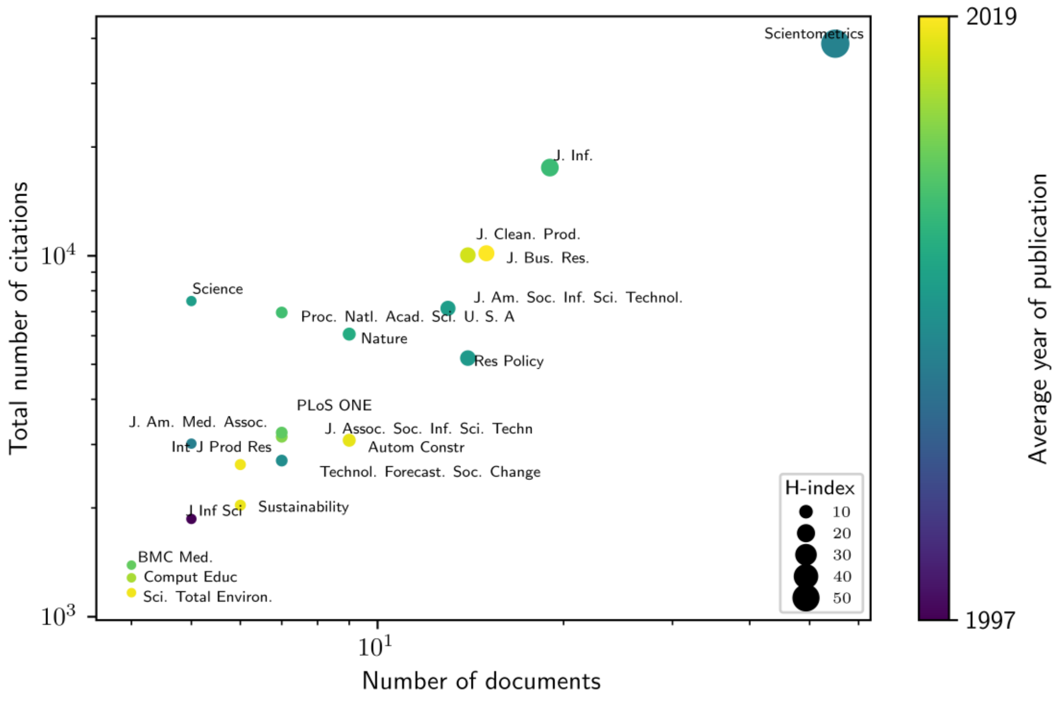

Figure 2. Scatter plot of top 20 sources from Biblium: total number of documents versus total citations on logarithmic scales, with marker size indicating H-index and color representing the average publication year.

Networks

Biblium includes several types of bibliographic networks to facilitate the analysis of scientific literature. These networks encompass keyword co-occurrence, co-authorship, co-citation, co-occurrence of custom-made concepts, and bibliographic coupling. However, a citation network between documents is missing. For illustration, a keyword co-occurrence network will be presented, based on authors' keywords, to demonstrate thematic structures within the dataset. The keyword co-occurrence network can be computed using the snippet

```
>>> ba.get_keyword_co_net()
```

First, the network is computed and saved as a .net file in Pajek (Batagelj & Mrvar, 2004) format (in network subfoder of the results folder). Then, partitioning (community detection, by default using the Louvain algorithm (Blondel et al., 2008)) is applied to the nodes. The partition is saved in a .clu file (Pajek format). Additionally, several vectors (numerical properties of the nodes) are computed, including the number of documents, total number of citations, average year of publication, and H-index. These are saved in .vec files (Pajek format). The network can then be plotted in Pajek or any other software that supports the Pajek format. Biblium can also plot it. By calling

```
>>> ba.plot_keyword_co_net()
```

the network and overlay representations are plotted: in the network view, color represents the partition, and size corresponds to the number of documents; in the overlay view, color indicates the average year of publication, and size represents the number of documents. Notice that the user can specify the number of keywords to include, the partitioning algorithm, and the network layout (default: spring_layout from NetworkX). The plots are saved in png, pdf and svg format.
Besides the classical keyword co-occurrence network, other visualization methods are available. A heatmap can be generated, which provides a matrix representation of keyword co-occurrence frequencies, highlighting patterns and relationships in a structured way. Additionally, a thematic map can be generated to represent clusters of keywords within a two-dimensional space. By default, the x-axis corresponds to centrality, while the y-axis represents density. This visualization facilitates the identification of thematic structures, including motor themes, highly developed themes, as well as emerging or declining themes and basic themes, based on their positioning within the quadrants. The Figure 3 represents a “classical” keyword co-occurrence network of the top 20 keywords.

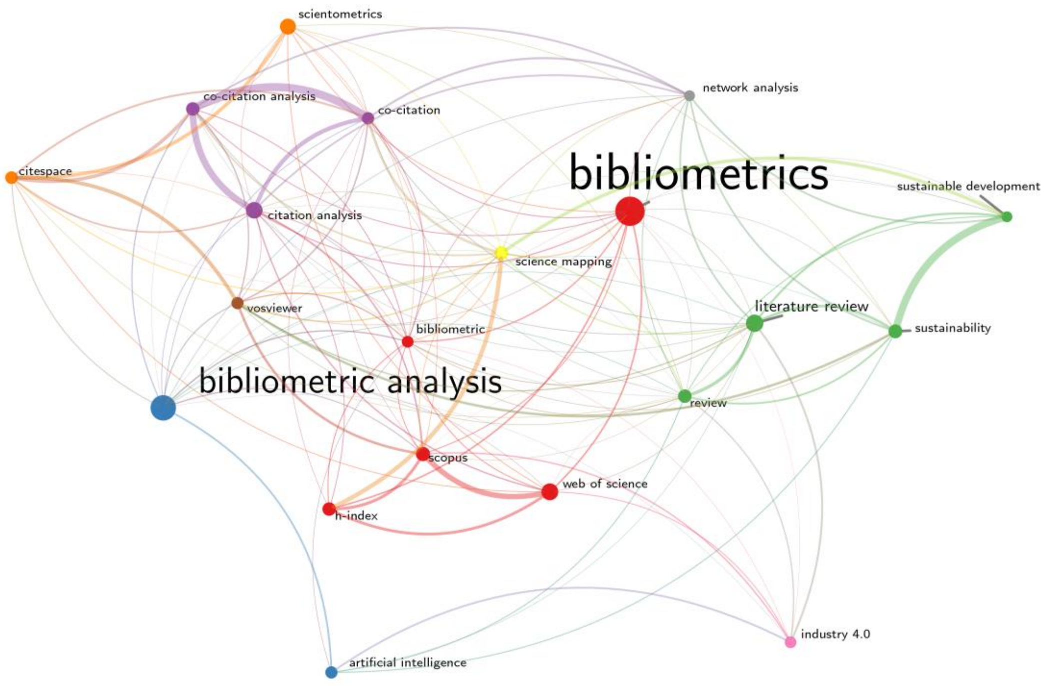

Figure 3. Co-occurrence network of author's keyword visualization from Biblium: direct output displaying relationships between key terms, with no synonym cleaning applied.

In the Figure 3, some keywords share the same meaning, which can create redundancy. For example, "bibliometric analysis" and "bibliometrics" refer to the same concept, just as "citation analysis" can be simplified to "citation," and "systematic review" to "review." To address this, Biblium supports keyword cleaning by merging synonyms and removing irrelevant terms.

Users can provide a dictionary mapping of synonyms to a preferred term and a list of words to be removed. Alternatively, they can prepare two Excel files (.xlsx) containing this information in a structured format. The structure of the synonym file can follow one of these formats: Column-based: The column name represents the term that should be kept, while all words listed below it will be replaced by this term; or Row-based: The first word in each row is the preferred term, and all other words in the same row will be replaced by it. Biblium also offers automatic cleaning by standardizing keywords, such as converting plural forms to singular where appropriate. This feature ensures a cleaner, more consistent keyword representation in the network. The example of keyword co-occurrence network after the cleaning is shown in Figure 4.

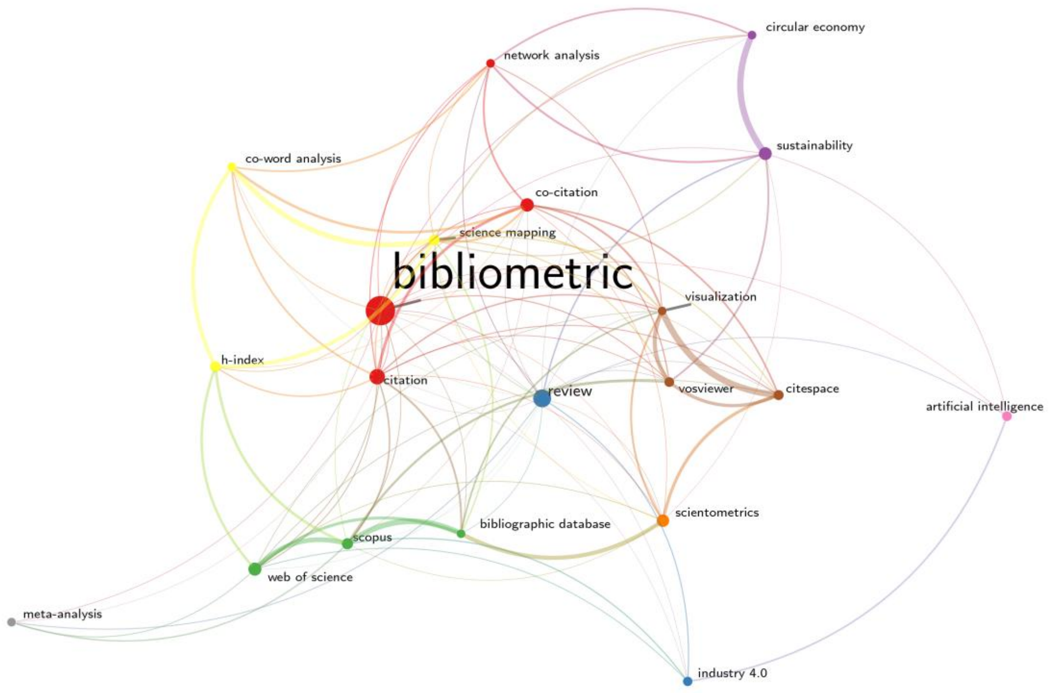

Figure 4. Co-occurrence network of author's keyword visualization from Biblium: cleaned output with synonyms consolidated.

Reports

Biblium supports comprehensive export capabilities that allow users to generate professional bibliometric reports in multiple formats. Reports can be exported as Excel (.xlsx) files with styled tables, as well as Word (.docx) and PowerPoint (.pptx) documents, following a user-defined structure stored in Excel templates. For Word exports, Biblium uses the python-docx library, enabling detailed styling, structured headings, captions, and dynamic table and figure generation. PowerPoint reports are built using python-pptx, allowing for custom slide layouts, embedded figures, and narrative content aligned with analytical outputs. These flexible export options ensure seamless integration with dissemination workflows and enhance the clarity of bibliometric insights. Reports can be generated by calling

```
>>> ba.save_reports(formats=["docx", "xlsx", “pptx”, "tex"], f_name="report")
```

where the user specifies in which format(s) the report should be saved and under what file name.

Other functionalities

In addition to the core functionalities discussed in detail throughout this section, Biblium offers several other useful features that enhance bibliometric analysis. These functionalities provide additional flexibility and depth for users interested in refining their research. For instance, the tool supports the computation of new variables by integrating information from abstracts, titles, and keywords. It also enables users to define custom concepts using keywords and regular expressions, facilitating the identification of more complex thematic structures. Preprocessing tools such as lemmatization, stopword removal, and user-defined word filtering further improve text clarity for subsequent analysis.

Beyond text processing, Biblium includes a range of scientific metrics that capture interdisciplinary connections, reference statistics (sources, authors, age distributions), and top cited documents and references. Users can segment data into custom time periods for group analyses, track the dynamics of sources, authors, and keywords, and explore trending topics over time. Additionally, various statistical techniques allow for association analysis between general concepts (such as keywords, authors, and sources) and user-defined categories, offering deeper insights into the analysed dataset.

Biblium includes several clustering algorithms for grouping the documents and other units, such as sources and authors, etc. These algorithms rely on keywords, references, or any user-defined (dis)similarity measure. The main clustering methods implemented are hierarchical clustering, k-means, and bibliographic coupling. Once cluster membership is determined, statistical evaluation—such as comparing groups using descriptive statistics and statistical tests—can be performed. Additionally, data mining approaches like logistic regression can be applied for further analysis. However, group membership does not necessarily have to result from a clustering approach, as statistical comparison and data mining techniques can also be used independently to analyze predefined groups.

For more advanced exploration, Biblium includes topic modeling (Latent Dirichlet allocation, LDA (Blei et al., 2003)), factor analysis (correspondence analysis, hierarchical clustering, multidimensional scaling, MDS), sentiment analysis, and extended visualizations. The k-field plot, an extension of the traditional three-field Sankey diagram, provides an effective way to relate multiple concepts within a dataset. Visualization tools such as bar plots, lollipop plots, violin plots, heatmaps, Venn diagrams, word clouds, and treemaps make it easier to interpret results. Additionally, classic bibliometric models, including Lotka's law (author productivity) and Bradford's law (source dispersion), are implemented to provide theoretical context.

Biblium ensures the preservation and reusability of computed data through pickling a BiblioAnalysis object and the storage of indicator dataframes, making it easy to retrieve results for further analysis. While these functionalities are not explored in full detail in this section, they provide additional opportunities for users to deepen their bibliometric investigations and tailor analyses to their specific research needs.

Bibliometric group analysis with Biblium

Group analysis plays an important role in bibliometric studies by enabling the identification of patterns and trends within specific segments of a dataset. Groups can be defined based on predefined criteria such as time periods, geographic regions, or scientific disciplines, but they can also be formed using clustering algorithms,

which categorize documents based on keywords, references, or abstracts (e.g., bibliographic coupling). The group analysis offers a different and more detailed bibliometric analysis than analyzing the whole dataset. For example, a temporal group analysis can reveal how scientific priorities have changed over decades, while a geographic analysis can highlight the research strengths of different countries or institutions. Similarly, grouping by scientific disciplines allows us to examine how different fields contribute to the overall scientific landscape and interact with one another, which topic are more associated with particular domains, etc.
In this chapter, we will illustrate the functionalities of Biblium on group analysis based on scientific disciplines, using classification data extracted from the Scopus webpage. Each document is assigned to one or more categories—Social Sciences, Physical Sciences, Health Sciences, or Life Sciences (excluding Multidisciplinary for this example)—allowing us to explore discipline-specific trends and contributions. We will demonstrate how to initialize and conduct this analysis within Biblium.
In Biblium, group-based analysis is performed by initializing an object of the class BiblioGroup. This initialization builds on the standard setup used in BiblioAnalysis, with the key addition of a flexible parameter called group_desc, which defines how documents are grouped for analysis. The group_desc can take multiple forms, offering extensive flexibility: (1) the name of a column in the bibliometric DataFrame, where each value indicates the group membership of a document; (2) a binary indicator DataFrame, with rows representing documents and columns representing groups; (3) the name of a multi-valued column (e.g., keywords or authors), where items are separated by a delimiter; or (4) a dictionary of regular expressions, used to classify documents based on text in a specified column (e.g., abstracts). When time-based grouping is required, group_desc="Year" can be combined with cutpoints or n_periods to define custom or evenly spaced time periods. Biblium automatically detects the appropriate grouping logic and constructs a binary group matrix that supports both disjoint and overlapping group structures. This design ensures accurate and scalable bibliometric analysis across a wide range of use cases.

```
>>> bg = bb.BiblioGroupAnalysis(f_name="data.csv", db="scopus", group_desc=”Sciences”)
```

Notice that the initialization process is similar to that of the BiblioAnalysis object, with the key difference being the inclusion of the group_desc parameter. This parameter corresponds to a column in the dataset (data.csv) that contains information about the scientific classification of each document. If a document is associated with multiple scientific fields, they are separated by a semicolon (;). This classification column is not included in the dataset directly downloaded from Scopus and must be generated by the user. However, for datasets originating from Scopus, Biblium provides specialized functions to assist in adding this classification column. These functions require additional metadata about sources, which is accessible on the Scopus webpage for registered users.

Main Information

BiblioGroupAnalysis extends the functionalities of bibliometric analysis to grouped data, ensuring that most features available for individual datasets are also implemented for groups. Key analyses such as main information, scientific production, and performance indicators are all accessible within this framework, maintaining a consistent output structure. Typically, results are stored in pandas DataFrames, which can be conveniently saved in xlsx format for further use. However, in BiblioGroupAnalysis, outputs are separated by group, allowing for comparative insights across different categories. The following snippet

```
>>> bg.get_main_info()
```

computes the main information for grouped data, and the selected rows of the output are displayed in the table 1.

Table 1. Biblium summary statistics (part of the computed dataframe): overview of document distribution, sources, citations, and collaboration trends across the groups (scientific disciplines).

	Social Sciences	*Physical Sciences*	*Health Sciences*	*Life Sciences*
Number of documents	295	265	80	39
Number of sources	121	107	65	35
Timespan	1983: 2022	1976: 2022	1996: 2022	1997: 2022
Total citations	164,203	143,355	35,392	17,933
H-index	248	239	80	39
Average year of publication	2012.61	2012.95	2010.91	2014.59
Top 5 sources	Scientometrics, Journal of Informetrics, Journal of Business Research, Journal of Cleaner Production, Research Policy	Scientometrics, Journal of Informetrics, Journal of Cleaner Production, Journal of the American Society for Information Science and Technology, Automation in Construction	JAMA, BMC Medicine, BMJ (Online), Environmental Research Letters, PLoS Medicine	PLoS Biology, Expert Opinion on Biological Therapy, PeerJ, Trends in Ecology and Evolution, Agronomy
Top 5 keywords	bibliometrics, bibliometric analysis, literature review, web of science, citation analysis	bibliometrics, bibliometric analysis, web of science, literature review, citation analysis	bibliometrics, bibliometric analysis, open access, scientific publishing, citation analysis	bibliometric analysis, scientometrics, citation analysis, bibliometrics, regenerative medicine
Collaboration index	3.51	4.21	5.42	5.12

These kinds of tables allow for comparisons across the analyzed groups in terms of research output, impact, collaboration as well as the content. Differences in the number of documents and sources highlight variations in publication density and dispersion, while the timespan and average publication year indicate dynamics of the research trends. Citation metrics, including total citations and H-index, reveal differences in research influence and scholarly impact. The top keywords and source illustrate publishing trends and possible overlap between groups. Note that the presented statistics represent only a subset of the possible results and are provided here for illustrative purposes. To visually represent the top sources for each group (in this case, scientific disciplines), we can use the following snippet:

```
>>> bg.plot_top_sources_barh()
```

The output is shown in Figure 5. The plot shows the distribution of documents across sources, categorized into four groups based on scientific discipline: Social Sciences, Physical Sciences, Health Sciences, and Life Sciences. It is evident that some sources contribute to multiple scientific disciplines. Similar graphs can be generated for authors, countries of the corresponding author, keywords, and other categories.

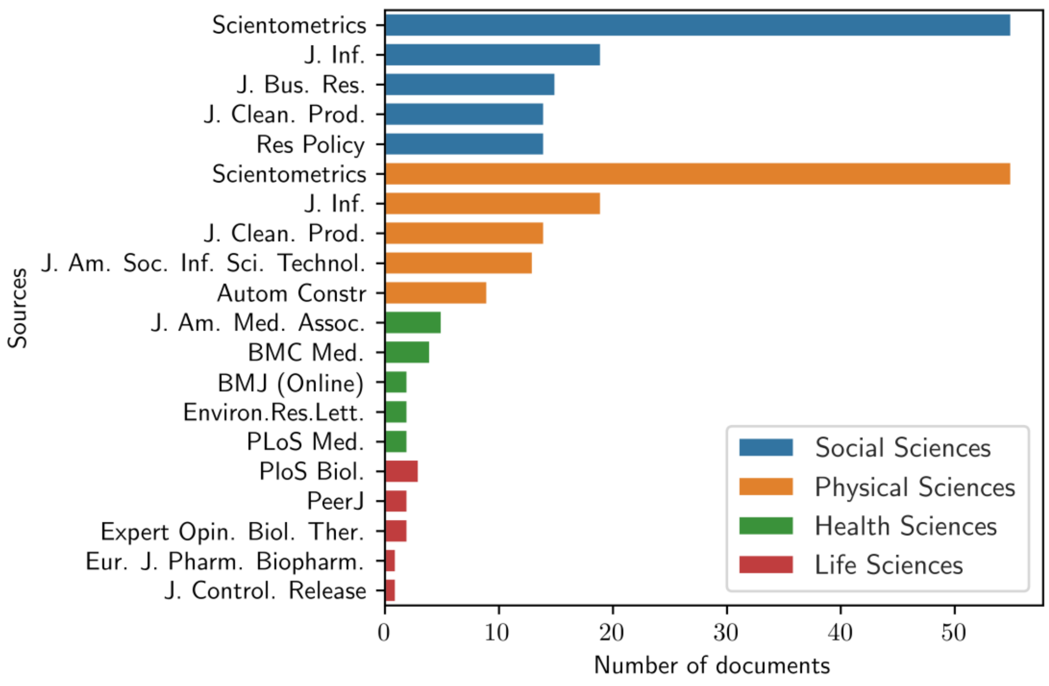

Figure 5. Bar chart from Biblium: top sources contributing to documents in analyzed groups: Social Sciences, Physical Sciences, Health Sciences, and Life Sciences.

Analysis of group overlapping

When analyzing groups, a document can belong to multiple groups (such as in our case, where it belongs to multiple scientific disciplines), meaning that some groups may overlap. This overlap is implemented in Biblium through various visualizations, including upsetplot, heatmaps, clustermaps, network graphs, and Venn diagrams. Heatmaps and clustermaps depict the total number of overlapping documents or their normalized values, with Jaccard index as the default normalization method, though other indices are available. The overlap can be plotted using a snippet

```
>>> bg.plot_overlapping(kind="venn")
```

that generates a Venn diagram. The final result of this visualization is shown in Figure 6. If the groups are completely disjoint, these visualizations are meaningless, and Biblium does not provide them.

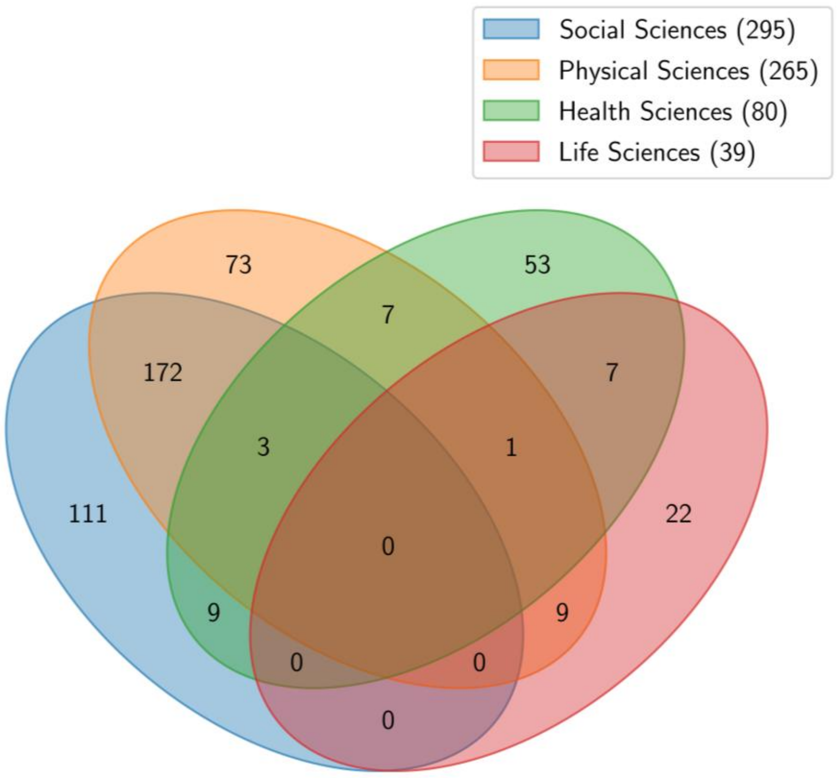

Figure 6. Venn diagram from Biblium: distribution of documents across four scientific disciplines (Social Sciences, Physical Sciences, Health Sciences, and Life Sciences). Overlaps indicate documents categorized under multiple disciplines.

This type of visualization, as shown in Figure 6, effectively captures overlapping and non-overlapping regions for up to four groups. While it is technically possible to represent up to six groups in a Venn diagram, the plot becomes increasingly complex and difficult to interpret as the number of groups exceeds four. In a Venn diagram, the sizes of the groups are included by default in the legend for reference, helping to understand the overall sizes of the groups.

For datasets with more than four groups, a heatmap may be a better choice for visualizing pairwise overlaps, as it is not limited by the number of groups and provides a clearer representation of relationships between categories. However, more complex overlaps involving more than two groups cannot be shown in a heatmap.

Scientific production

The scientific production over time for all groups can be computed in a manner similar to computation in BiblioAnalysis. Here, the resulting plot provides insights into the distribution of published documents across all groups and overall cumulative citations. The script snippet

```
>>> bg.get_production()
>>> bg.plot_production(cut_year=2000)
```

produced the graph shown in Figure 7.

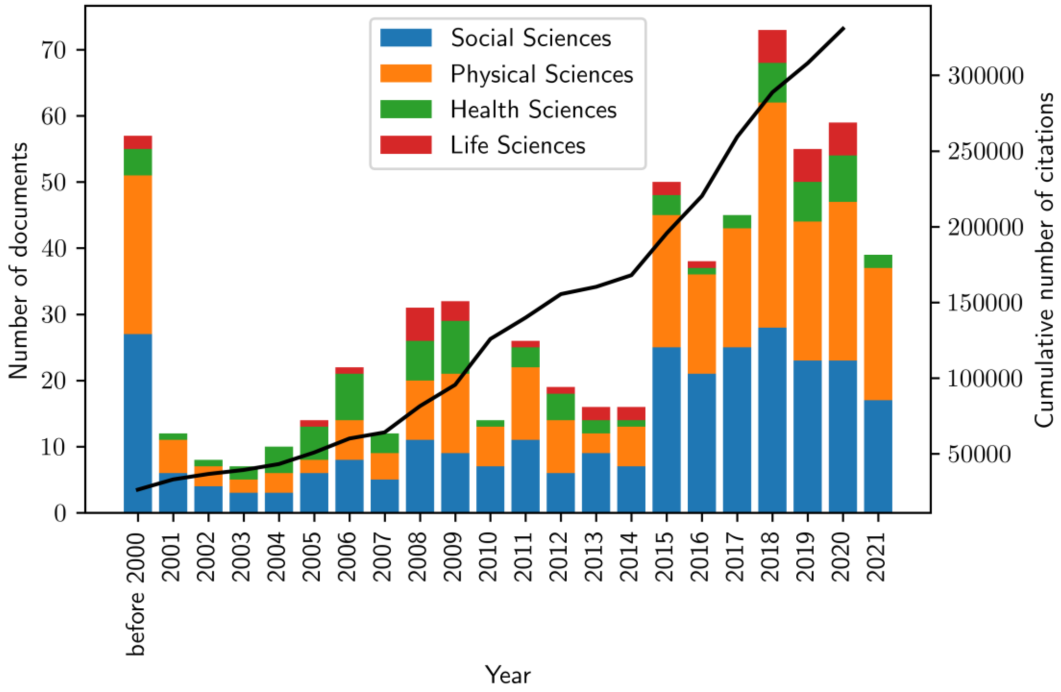

Figure 7. Stacked bar chart from Biblium: annual number of documents categorized by scientific disciplines, alongside cumulative citations over time.

The figure presents a stacked bar chart with a cutoff year, categorizing documents into overlapping disciplinary groups. The black line represents the cumulative number of citations. Since the groups are not pairwise disjoint, the total number of documents in any given year can exceed the actual count due to overlaps, requiring careful interpretation. This visualization provides an aggregated view of the temporal trends in scientific output and its impact.

Measuring Associations

In bibliometric group analysis, various concepts such as sources, authors, and keywords can be associated with divisions into groups, enabling a structured examination of their relationships. The general approach involves computing a 2×2 contingency table for each association, where documents are classified based on whether they belong to a particular group (yes/no) and whether they represent the given concept (yes/no). This results in four possible combinations, forming the basis for statistical analysis. From these tables, multiple measures can be derived to quantify the strength and significance of associations. By default, Biblium computes counts, marginal proportions, the Jaccard index, Yule's Q, and the odds ratio, along with Fisher's exact test to assess statistical significance. While many additional measures are implemented in Biblium, they remain disabled by default.

Let us illustrate the computation of association on a concrete pair (group, keyword). The table 2 shows the relationship between group membership (Physical Sciences or not) and keyword assignment ("industry 4.0" or not). The count a = 10 represents the number of documents in the Physical Sciences group having assigned keyword "industry 4.0," while b = 145 refers to those in the same group but without this keyword. Similarly, c = 0 indicates no documents outside the Physical Sciences group are linked to "industry 4.0," whereas d = 107 represents those outside this group and without the keyword. In total, 262 documents were analyzed, a number lower than the total available (under 500) since those without author keywords were automatically excluded.

Table 2. Contingency table behind calculations from Biblium for illustration: distribution of documents categorized by their membership to "Industry 4.0" and Physical Sciences. This table is a starting point for statistical calculations.

	"industry 4.0"	*"not industry 4.0"*	*Total*
Physical Sciences	a=10	b=145	155
not Physical Sciences	c=0	d=107	107
Total	10	252	262

From this table Jaccard index can be computed using the formula

$$J = \frac{a}{a+b+c} = \frac{10}{10+145+0} = 0.0645$$

and Yule's Q using formula

$$Q = \frac{ad - bc}{ad + bd} = \frac{10 * 107 - 145 * 0}{10 * 107 + 145 * 0} = 1$$

A Yule's Q value of 1 indicates a perfect positive association between the two variables. These measures provide insight into the strength and overlap of the association between group membership and keyword usage. In addition to descriptive measures like Jaccard and Yule's Q, a statistical test such as Fisher's exact test can be performed on this contingency table to assess whether the observed

association is statistically significant. This test is implemented in Biblium by default, providing a rigorous method to evaluate the strength and significance of such associations. Alongside the p-value, Biblium also computes the odds ratio (OR) to quantify the strength of the association. Since multiple hypotheses are tested simultaneously, p-values are adjusted to control for false discovery rates. By default, Biblium applies the Benjamini-Hochberg FDR (Benjamini & Hochberg, 1995) method to ensure reliable statistical inference.
The snippet

```
>>> bg.associate_keywords()
```

computes associations between all groups in the dataset and a selected set of keywords, which by default includes the top 20 keywords across the entire dataset. These associations can be calculated for various units, such as sources, authors, countries of corresponding authors, and more, depending on the analysis focus. For illustration, the table 3 displays only a subset of group-keyword pairs and selected measures. While the analysis generates a range of metrics (including raw counts a, b, c and d from the contingency table), the table highlights key measures such as the Jaccard index, Yule's Q, and the p-value from Fisher's exact test. For illustration, only associations with an unadjusted p-value less than or equal to 0.05 are included in the table 3.

Table 3. Part of Biblium keyword association analysis (computed keywords_assoc_df dataframe): keyword-group pairs with Jaccard index, Yule's Q, and p-values less than 0.05.

group	*keyword*	*Jaccard*	*Yule Q*	*p-value*
Physical Sciences	industry 4.0	0.065	1.000	0.006
Social Sciences	co-citation	0.142	0.698	0.008
Physical Sciences	web of science	0.127	0.596	0.009
Life Sciences	scientometrics	0.129	0.720	0.014
Social Sciences	network analysis	0.077	1.000	0.014
Social Sciences	sustainability	0.087	0.733	0.049
Social Sciences	co-citation analysis	0.082	0.717	0.050

In Biblium, the association of keywords with groups can also be visually explored through word clouds. For each group one word cloud is plotted. In each plot, the size of a word in the word cloud reflects its frequency in the dataset, while the color represents its strength of association with the group, based on a selected measure of association (e.g., Jaccard index, Yule's Q). This visualization provides an intuitive way to interpret both the prevalence and the relevance of keywords within specific groups, enhancing the understanding of the relationships in the dataset.

Other functionalities

In Biblium, bibliometric group analysis is further enhanced by the PredictBiblioGroup class, which is inherited from the BiblioGroupAnalysis class. This implementation integrates data mining prediction methods, where group membership serves as the dependent variable. Predictive modeling in PredictBiblioGroup supports all statistical prediction models available in Scikit-learn (Pedregosa Fabian et al., 2011), along with logistic regression from Statsmodels (Seabold & Perktold, 2010). These models facilitate the classification of bibliometric entities based on various predictive features. To ensure the reliability of predictions, model evaluation is performed using 5-fold cross-validation by default. Performance assessment includes standard classification metrics such as classification accuracy and area under the curve (AUC), among others.

Another class inherited from BiblioGroupAnalysis focuses on the identification of research related to Sustainable Development Goals (SDGs). This method relies on predefined queries from Scopus, where selected keywords and rules determine whether a document (based on its title, abstract, and keywords) is associated with a particular SDG (SDG1–SDG16). This classification approach provides insights into the alignment of scientific output with global sustainability objectives.

This is just one example of a specific application we utilized in our previous research ((Umek et al., 2023), (Umek et al., 2024)). More broadly, this method can be adapted for different thematic groupings. By defining a customized list of keywords relevant to a particular concept, the same approach can be applied to classify documents into other specific research domains.

Conclusion and Future Work

The Biblium project represents an advancement in bibliometric and scientometric analysis within the Python ecosystem, offering a comprehensive and flexible alternative to existing tools. By replicating and expanding upon the core functionalities of Bibliometrix, Biblium introduces key innovations such as (sub)group analysis, predictive modeling, and advanced visualization techniques.

The Biblium project aims for a public release through GitHub, making the source code accessible to the research community and open-source contributors. This step will allow for collaboration, issue tracking, and community-driven improvements. Alongside the GitHub release, Biblium will be packaged as a Python library available via pip, enabling easy installation and integration into research workflows. The pip package will ensure streamlined updates and compatibility with various Python environments.

To enhance Biblium's analytical capabilities, data mining techniques will be incorporated, enabling users to uncover hidden patterns, trends, and relationships within bibliographic datasets. These methods will support text mining, clustering, and classification, facilitating deeper insights into scientific production and citation networks. By integrating advanced algorithms, Biblium will become a powerful tool for researchers looking to analyze large bibliometric datasets with minimal effort.

Biblium will expand its visualization capabilities using Bokeh (Bokeh Development Team, 2018), a powerful library for interactive and high-performance graphics. This will allow users to create dynamic plots, interactive dashboards, and detailed visual representations of bibliographic data. Compared to static plots, Bokeh will enable users to explore their data more intuitively, with zooming, filtering, and hover tools enhancing interpretability. These improvements will be particularly beneficial for analyzing citation networks, keyword co-occurrences, and publication trends.

Future versions of Biblium will incorporate large language models (LLMs) to enhance automated document analysis, clustering, and topic identification. LLMs can be leveraged for the automatic description of document clusters, identification of emerging topics, and a more automated approach to textual and bibliometric analysis. By integrating AI-based techniques, Biblium will provide deeper insights beyond traditional statistical methods, offering richer contextual understanding. One aspect of future research will involve comparing traditional clustering algorithms—such as hierarchical clustering, partitioning methods like k-means, and bibliographic coupling—with AI-driven approaches to evaluate their effectiveness and applicability in structuring scientific knowledge.

A Tkinter-based application (Lundh, 1999) is planned to provide a graphical user interface (GUI) for Biblium, making it accessible to users unfamiliar with coding. The app will include menus, scrollable canvases, and tabbed interfaces for organizing various bibliometric tasks. Users will be able to load, filter, and visualize data through an intuitive interface, with buttons for executing core functions. The Tkinter app will serve as a lightweight, standalone solution for researchers seeking an easy-to-use bibliometric tool.

To further broaden its usability, Biblium will be developed as an add-on for Orange (Demšar et al., 2013), a popular open-source data visualization and machine learning tool. This integration will enable users to apply Biblium's bibliometric and text-mining functionalities within Orange's visual programming environment. Researchers will be able to drag and drop Biblium components into their workflows, combining them with Orange's built-in machine learning and data processing features. This will make Biblium accessible to a wider audience and facilitate exploratory bibliometric analysis without requiring extensive coding knowledge.

Acknowledgments

The authors acknowledge the financial support from the Slovenian Research and Innovation Agency (research programme No. P5-0093 and project No. J5-50183). Finally, in the preparation of this manuscript, the authors utilized ChatGPT, version 4o, developed by OpenAI, for limited and supplementary purposes. Specifically, ChatGPT was employed to assist with checking the grammar, enhancing clarity, and

polishing the language. ChatGPT did not contribute to the intellectual content or scientific insights of the manuscript.

References

Alonso, S., Cabrerizo, F. J., Herrera-Viedma, E., & Herrera, F. (2010). hg-index: A new index to characterize the scientific output of researchers based on the h- and g-indices. *Scientometrics, 82*(2), 391–400. https://doi.org/10.1007/S11192-009-0047-5

Anderson, T. R., Hankin, R. K. S., & Killworth, P. D. (2008). Beyond the Durfee square: Enhancing the h-index to score total publication output. *Scientometrics, 76*(3), 577–588. https://doi.org/10.1007/S11192-007-2071-2/METRICS

Aria, M., & Cuccurullo, C. (2017a). bibliometrix: An R-tool for comprehensive science mapping analysis. *Journal of Informetrics, 11*(4), 959–975. https://doi.org/10.1016/J.JOI.2017.08.007

Aria, M., & Cuccurullo, C. (2017b). bibliometrix: An R-tool for comprehensive science mapping analysis. *Journal of Informetrics, 11*(4), 959–975. https://doi.org/10.1016/J.JOI.2017.08.007

Batagelj, V., & Mrvar, A. (2004). *Pajek — Analysis and Visualization of Large Networks*. 77–103. https://doi.org/10.1007/978-3-642-18638-7_4

Benjamini, Y., & Hochberg, Y. (1995). Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society. Series B (Methodological), 57*(1), 289–300. https://doi.org/10.2307/2346101

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. "O'Reilly Media, Inc."

Blei, D. M., Ng, A. Y., & Edu, J. B. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research, 3*, 993–1022. https://doi.org/10.5555/944919.944937

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment, 2008*(10), P10008. https://doi.org/10.1088/1742-5468/2008/10/P10008

Bokeh Development Team. (2018). *Bokeh: Python library for interactive visualization*.

Cabrerizo, F. J., Alonso, S., Herrera-Viedma, E., & Herrera, F. (2010). q2-Index: Quantitative and qualitative evaluation based on the number and impact of papers in the Hirsch core. *Journal of Informetrics, 4*(1), 23–28. https://doi.org/10.1016/J.JOI.2009.06.005

Chen, C. (2006). CiteSpace II: Detecting and visualizing emerging trends and transient patterns in scientific literature. *Journal of the American Society for Information Science and Technology, 57*(3), 359–377. https://doi.org/10.1002/ASI.20317

Cobo, M. J., Lõpez-Herrera, A. G., Herrera-Viedma, E., & Herrera, F. (2012). SciMAT: A new science mapping analysis software tool. *Journal of the American Society for Information Science and Technology, 63*(8), 1609–1630. https://doi.org/10.1002/ASI.22688

Demšar, J., Curk, T., Erjavec, A., Gorup, C., Hočevar, T., Milutinovič, M., Možina, M., Polajnar, M., Toplak, M., Starič, A., Štajdohar, M., Umek, L., Žagar, L., Žbontar, J., Žitnik, M., & Zupan, B. (2013). Orange: Data mining toolbox in python. *Journal of Machine Learning Research, 14*.

Flyamer, I. (2012). adjustText. In *GitHub repository*. GitHub.

Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). *Exploring Network Structure, Dynamics, and Function using NetworkX*.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van

Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature 2020 585:7825*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Heldens, S., Sclocco, A., Dreuning, H., van Werkhoven, B., Hijma, P., Maassen, J., & van Nieuwpoort, R. V. (2022). litstudy: A Python package for literature reviews. *SoftwareX*, *20*, 101207. https://doi.org/10.1016/J.SOFTX.2022.101207

Hirsch, J. E. (2005). An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences of the United States of America*, *102*(46), 16569–16572. https://doi.org/10.1073/PNAS.0507655102

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science and Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Inc., P. T. (2015). *Collaborative data science*. Plotly Technologies Inc.

Jin, B. H., Liang, L. M., Rousseau, R., & Egghe, L. (2007). The R- and AR-indices: Complementing the h-index. *Chinese Science Bulletin*, *52*(6), 855–863. https://doi.org/10.1007/S11434-007-0145-9/METRICS

Laserson, U. (2009). Sqaurify. In *GitHub repository*. GitHub.

Lundh, F. (1999). An introduction to tkinter. *URL: Www. Pythonware. Com/Library/Tkinter/Introduction/Index. Htm*.

Mckinney, W. (2010). *Data Structures for Statistical Computing in Python*.

McLevey, J., & McIlroy-Young, R. (2017). Introducing metaknowledge: Software for computational research in information science, network analysis, and science of science. *Journal of Informetrics*, *11*(1), 176–197. https://doi.org/10.1016/J.JOI.2016.12.005

Moral-Muñoz, J. A., Herrera-Viedma, E., Santisteban-Espejo, A., & Cobo, M. J. (2020). Software tools for conducting bibliometric analysis in science: An up-to-date review. *Profesional de La Información*, *29*(1), 1699–2407. https://doi.org/10.3145/EPI.2020.ENE.03

Mueller, A. (2010). Word cloud. In *GitHub repository*. GitHub.

Nothman, J. (2023). *UpSetPlot*. https://github.com/jnothman/UpSetPlot

OpenGenderTracking. (2013). https://github.com/OpenGenderTracking/GenderTracker

Pedregosa Fabian, Michel, V., Grisel OLIVIERGRISEL, O., Blondel, M., Prettenhofer, P., Weiss, R., Vanderplas, J., Cournapeau, D., Pedregosa, F., Varoquaux, G., Gramfort, A., Thirion, B., Grisel, O., Dubourg, V., Passos, A., Brucher, M., Perrot andÉdouardand, M., Duchesnay, andÉdouard, & Duchesnay Y, Fré. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825–2830.

Peirson, B. R. E. (2016). *Tethne v0.7. http://diging.github.io/tethne/*. Et Al.

Pereira, V., Pereira Basilio, M., Henrique, C., & Santos, T. (2025). PyBibX-a Python library for bibliometric and scientometric analysis powered with artificial intelligence tools. *Data Technologies and Applications*. https://doi.org/10.1108/DTA-08-2023-0461

Rehurek, R., & Sojka, P. (2011). Gensim--python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, *3*(2).

Rose, M. E., & Kitchin, J. R. (2019). pybliometrics: Scriptable bibliometrics using a Python interface to Scopus. *SoftwareX*, *10*, 100263. https://doi.org/10.1016/J.SOFTX.2019.100263

Rossetti, G., Milli, L., & Cazabet, R. (2019). CDLIB: a python library to extract, compare and evaluate communities from complex networks. *Applied Network Science*, *4*(1), 1–26. https://doi.org/10.1007/S41109-019-0165-9/TABLES/5

Ruiz-Rosero, J., Ramirez-Gonzalez, G., & Viveros-Delgado, J. (2019). Software survey: ScientoPy, a scientometric tool for topics trend analysis in scientific publications. *Scientometrics*, *121*(2), 1165–1188. https://doi.org/10.1007/S11192-019-03213-W

Seabold, S., & Perktold, J. (2010). statsmodels: Econometric and statistical modeling with python. *9th Python in Science Conference*.

Tctianchi. (2014). Pyvenn. In *GitHub repository*. GitHub.

Team, S. (2009). *Science of Science (Sci2) Tool*. https://sci2.cns.iu.edu

Umek, L., Ravšelj, D., & Aristovnik, A. (2023). Public sector reforms and sustainable development: evidence from bibliometric analys. *IASIA 2023 Conference*.

Umek, L., Takahiro, M., Aristovnik, A., & Ravšelj, D. (2024). Collaborative governance and sustainable development: evidence from bibliometric analysis. *IAS Mombasa Conference 2024*.

Umesh, P. (2012). Image Processing in Python. *CSI Communications*, *23*.

van Eck, N. J., & Waltman, L. (2010). Software survey: VOSviewer, a computer program for bibliometric mapping. *Scientometrics*, *84*(2), 523–538. https://doi.org/10.1007/S11192-009-0146-3

van Eck, N. J., & Waltman, L. (2017). Citation-based clustering of publications using CitNetExplorer and VOSviewer. *Scientometrics*, *111*(2), 1053–1070. https://doi.org/10.1007/S11192-017-2300-7/TABLES/4

Velasquez, J. D. (2023). TechMiner: Analysis of bibliographic datasets using Python. *SoftwareX*, *23*. https://doi.org/10.1016/J.SOFTX.2023.101457

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … Vázquez-Baeza, Y. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods 2020 17:3*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2

Waskom, M. L. (2021). seaborn: statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021