# bibmanager Documentation

***Release 1.4.3***

**Patricio Cubillos**

**Aug 10, 2022**

# Contents


**1 Features** **3**

**2 Contributors** **5**

**3 Documentation** **7**
- 3.1 Getting Started . . . 7
- 3.2 BibTeX Management . . . 9
- 3.3 LaTeX Management . . . 22
- 3.4 ADS Management . . . 24
- 3.5 PDF Management . . . 31
- 3.6 FAQs and Resources . . . 35
- 3.7 API . . . 38
- 3.8 Contributing . . . 71
- 3.9 License . . . 72

**4 Be Kind** **75**

**5 Featured Articles** **77**

**Python Module Index** **79**

**Index** **81**

**The Next Standard in BibTeX Management**

**Author** Patricio Cubillos and contributors (see *Contributors*)

**Contact** pcubillos[at]fulbrightmail.org

**Organizations** Space Research Institute (IWF)

**Web Site** https://github.com/pcubillos/bibmanager

**Date** Aug 10, 2022

# CHAPTER 1

# Features

`bibmanager` is a command-line based application to facilitate the management of BibTeX entries, allowing the user to:

- Unify all BibTeX entries into a single database
- Automate .bib file generation when compiling a LaTeX project
- Automate duplicate detection and updates from arXiv to peer-reviewed
- Clean up (remove duplicates, ADS update) any external bibfile (since version 1.1.2)
- Keep a database of the entries' PDFs and fetch PDFs from ADS (since version 1.2)
- Browse interactively through the database (since version 1.3)
- Keep track of the more relevant entries using custom-set tags (since version 1.4)

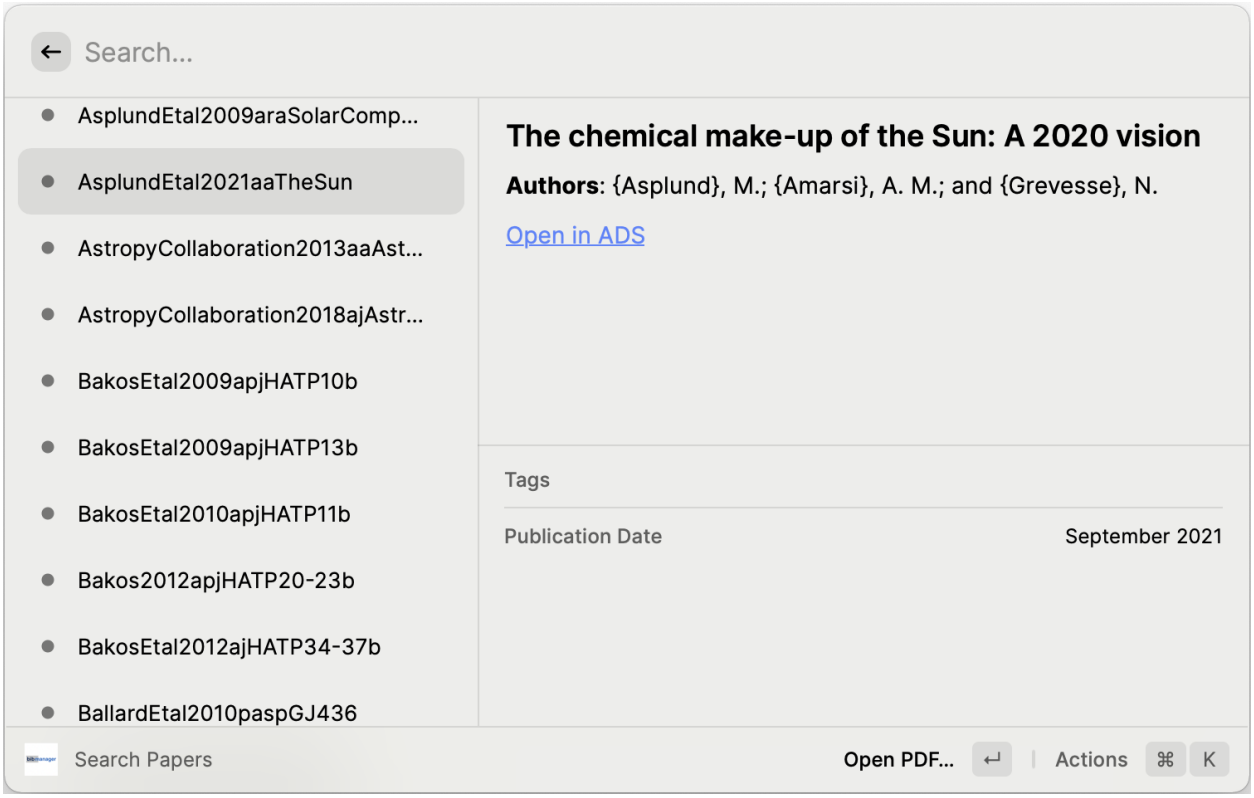
`bibmanager` also simplifies many other BibTeX-related tasks:

- Add or modify entries into the `bibmanager` database:
  - Merging user's .bib files
  - Manually adding or editing entries
  - Add entries from ADS bibcodes
- entry adding via your default text editor
- Query entries in the `bibmanager` database by author, year, or title keywords
- Generate .bib files built from your .tex files
- Compile LaTeX projects with the `latex` or `pdflatex` directives
- Perform queries into ADS and add entries by bibcode
- Fetch PDF files from ADS (via their bibcode, new since version 1.2)

**Note:** Did you know that Aaron David Schneider built this totally amazing bibmanager graphic interface?

This extension lets you quickly browse through your database, retrieve metadata (title, date, tags), open in ADS or PDF (download if needed), or just copy things to the clipboard. **I've tried it and I can only recommend to checking it out!**

This is implemented via Raycast, which is available for Mac OS X users. To install Raycast and bibmanager extension check *these simple instructions*.




Check out this video tutorial to get started with `bibmanager`:

And this one covering some other features:

CHAPTER 2

# Contributors

`bibmanager` was created and is maintained by Patricio Cubillos (pcubillos[at]fulbrightmail.org).

These people have directly contributed to make the software better:

- K.-Michael Aye
- Ellert van der Velden
- Aaron David Schneider

CHAPTER 3

# Documentation

## 3.1 Getting Started

`bibmanager` offers command-line tools to automate the management of BibTeX entries for LaTeX projects.

`bibmanager` places all of the user's bibtex entries in a centralized database, which is beneficial because it allows `bibmanager` to automate duplicates detection, arxiv-to-peer review updates, and generate bibfiles with only the entries required for a specific LaTeX file.

There are four main categories for the `bibmanager` tools:

- *BibTeX Management* tools help to create, edit, browse, and query from a `bibmanager` database, containing all BibTeX entries that a user may need.
- *LaTeX Management* tools help to generate (automatically) a bib file for specific LaTeX files, and compile LaTeX files without worring for maintaining/updating its bib file.
- *ADS Management* tools help to makes queries into ADS, add entries from ADS, and cross-check the `bibmanager` database against ADS, to update arXiv-to-peer reviewed entries.
- *PDF Management* tools help to maintain a database of the PDF files associated to the BibTex entries: Fetch from ADS, set manually, and open in a PDF viewer.

Once installed (see below), take a look at the `bibmanager` main menu by executing the following command:

```
# Display bibmanager main help menu:
bibm -h
```

From there, take a look at the sub-command helps or the rest of these docs for further details, or see the *Quick Example* for an introductory worked example.

### 3.1.1 System Requirements

`bibmanager` is compatible with Python3.6+ and has been tested to work in both Linux and OS X, with the following software:

- numpy (version 1.15.1+)
- requests (version 2.19.1+)
- packaging (version 17.1+)
- prompt_toolkit (version 3.0.5+)
- pygments (version 2.2.0+)

### 3.1.2 Install

To install `bibmanager` run the following command from the terminal:

```
pip install bibmanager
```

Or if you prefer conda:

```
conda install -c conda-forge bibmanager
```

Alternatively (e.g., for developers), clone the repository to your local machine with the following terminal commands:

```
git clone https://github.com/pcubillos/bibmanager
cd bibmanager
python setup.py develop
```

---

**Note:** To enable the ADS functionality, first you need to obtain an ADS token, and set it into the `ads_tokend` config parameter. To do this:

1. Create an account and login into the new ADS system.
2. Get your token (or generate a new one) from here.
3. Set the `ads_token` bibmanager parameter:

```
# Set ads_token to 'my_ads_token':
bibm config ads_token my_ads_token
```

---

### 3.1.3 Quick Example

Adding your BibTeX file into `bibmanager` is as simple as one command:

```
# Add this sample bibfile into the bibmanager database:
bibm merge ~/.bibmanager/examples/sample.bib
```

Compiling a LaTeX file that uses those BibTeX entries is equally simple:

```
# Compile your LaTeX project:
bibm latex ~/.bibmanager/examples/sample.tex
```

This command produced a BibTeX file according to the citations in sample.tex; then executed latex, bibtex, latex, latex; and finally produced a pdf file out of it. You can see the results in *~/.bibmanager/examples/sample.pdf*.

As long as the citation keys are in the `bibmanager` database, you won't need to worry about maintaining a bibfile anymore. The next sections will show all of the capabilities that `bibmanager` offers.

## 3.2 BibTeX Management

### 3.2.1 reset

Reset the bibmanager database.

**Usage**

```
bibm reset [-h] [-d | -c] [bibfile]
```

This command resets the bibmanager database from scratch. It creates a .bibmanager/ folder in the user folder (if it does not exists already), and it resets the bibmanager configuration to its default values.

If the user provides the `bibfile` argument, this command will populate the database with the entries from that file; otherwise, it will set an empty database.

Note that this will overwrite any pre-existing database. In principle the user should not execute this command more than once in a given CPU.

**Options**

**bibfile**
: Path to an existing BibTeX file.

**-d, –database**
: Reset only the bibmanager database.

**-c, –config**
: Reset only the bibmanager config parameters.

**-h, –help**
: Show this help message and exit.

**Examples**

```
# Reset bibmanager database from scratch:
bibm reset

# Reset, including entries from a BibTeX file:
bibm reset my_file.bib

# Reset only the database (keep config parameters):
bibm reset my_file.bib -d

# Reset only the config parameters (keep database):
bibm reset -c
```

---

### 3.2.2 merge

Merge a BibTeX file into the bibmanager database.

**Usage**

```
bibm merge [-h] bibfile [take]
```

**Description**

This command merges the content from an input BibTeX file with the bibmanager database.

The optional 'take' arguments defines the protocol for possible- duplicate entries. Either take the 'old' entry (database), take the 'new' entry (bibfile), or 'ask' the user through the prompt (displaying the alternatives). bibmanager considers four fields to check for duplicates: doi, isbn, bibcode, and eprint.

Additionally, bibmanager considers two more cases (always asking):
(1) new entry has duplicate key but different content, and
(2) new entry has duplicate title but different key.

**Options**

**bibfile**
Path to an existing BibTeX file.

**take**
Decision protocol for duplicates (choose: {old, new, ask}, default: old)

**-h, –help**
Show this help message and exit.

**Examples**

```
# Merge BibTeX file ignoring duplicates (unless they update from arXiv to peer-
↪reviewed):
bibm merge my_file.bib

# Merge BibTeX file ovewriting entries if they are duplicates:
bibm merge my_file.bib new

# Merge BibTeX file asking the user which to take for each duplicate:
bibm merge my_file.bib ask
```

---

### 3.2.3 edit

Edit the bibmanager database in a text editor.

**Usage**

```
bibm edit [-h]
```

**Description**

This command let's you manually edit the bibmanager database, in your pre-defined text editor. Once finished editing, save and close the text editor, and press ENTER in the terminal to incorporate the edits (edits after continuing on the terminal won't count).

bibmanager selects the OS default text editor. But the user can set a preferred editor, see 'bibm config -h' for more information.

**Options**

**-h, –help**
Show this help message and exit.

**Examples**

```
# Launch text editor on the bibmanager BibTeX database:
bibm edit
```

### Meta-Information

*(New since Version 1.2)*
`bibmanager` allows the user to add meta-information to the entries (info that is not contained in the BibTex itself). This meta-info can be set while editing the database with the `bibm edit` command, by writting it before an entry. There are currently two meta-parameters:

- The *freeze* meta-parameter is a flag that freezes an entry, preventing it to be modified when running *ads-update*.
- The *pdf* meta-parameter links a PDF file to the entry. To do this, type '*pdf:*' followed by the path to a PDF file. If the PDF file is already in the *home/pdf* folder (see *config*), there's no need to specify the path to the file. Alternatively, see the commands in *PDF Management*.
- The *tags* meta-parameter enable setting user-defined tags for grouping and searching entries *(New since Version 1.4)*

Below there's an example to freeze and link a PDF file to an entry:

```
This file was created by bibmanager
https://pcubillos.github.io/bibmanager/

...

freeze
pdf: /home/user/Downloads/Rubin1980.pdf
@ARTICLE{1980ApJ...238..471R,
       author = {{Rubin}, V.~C. and {Ford}, W.~K., Jr. and {Thonnard}, N.},
        title = "{Rotational properties of 21 SC galaxies with a large range of␣
↪luminosities and radii, from NGC 4605 (R=4kpc) to UGC 2885 (R=122kpc).}",
      journal = {\apj},
         year = "1980",
        month = "Jun",
       volume = {238},
        pages = {471-487},
          doi = {10.1086/158003},
```

(continues on next page)

(continued from previous page)

```
       adsurl = {https://ui.adsabs.harvard.edu/abs/1980ApJ...238..471R},
      adsnote = {Provided by the SAO/NASA Astrophysics Data System}
}

...
```

### 3.2.4 add

Add entries into the bibmanager database.

**Usage**

```
bibm add [-h] [take]
```

**Description**

This command allows the user to manually add BibTeX entries into the bibmanager database through the terminal prompt.

The optional 'take' argument defines the protocol for possible-duplicate entries. Either take the 'old' entry (database), take the 'new' entry (bibfile), or 'ask' the user through the prompt (displaying the alternatives). bibmanager considers four fields to check for duplicates: doi, isbn, bibcode, and eprint.

Additionally, bibmanager considers two more cases (always asking):
(1) new entry has duplicate key but different content, and
(2) new entry has duplicate title but different key.

**Options**

**take**
: Decision protocol for duplicates (choose: {old, new, ask}, default: new)

**-h, –help**
: Show this help message and exit.

**Examples**

```
# Start multi-line prompt session to enter one or more BibTeX entries:
bibm add
```

### 3.2.5 tag

Add or remove tags to entries in the database.

**Usage**

```
bibm tag [-h] [-d] [-v VERB]
```

**Description**

This command adds or removes user-defined tags to specified entries in the Bibmanager database, which can then be used for grouping and searches. The tags are case sensitive and should not contain blank spaces.
*(New since version 1.4)*

Additionally, if the user only sets tags (but no entries), this command will display the existing entries that contain those tags.
There are five levels of verbosity:
verb < 0: Display only the keys of the entries
verb = 0: Display the title, year, first author, and key
verb = 1: Display additionally the ADS/arXiv urls and meta info
verb = 2: Display additionally the full list of authors
verb > 2: Display the full BibTeX entries

**Options**

**-h, –help**
Show this help message and exit.
**-d, –delete**
Delete tags instead of add.
**-v VERB, –verb VERB**
Verbosity level if used to display entries.

**Examples**

```
# Add a tag to an entry:
bibm tag
(Syntax is: KEY_OR_BIBCODE KEY_OR_BIBCODE2 ... tags: TAG TAG2 ...)
Hunter2007ieeeMatplotlib tag: python

# Add multiple tags to multiple entries:
bibm tag
(Syntax is: KEY_OR_BIBCODE KEY_OR_BIBCODE2 ... tags: TAG TAG2 ...)
1913LowOB...2...56S 1918ApJ....48..154S tags: galaxies history

# Remove tags:
bibm tag -d
(Syntax is: KEY_OR_BIBCODE KEY_OR_BIBCODE2 ... tags: TAG TAG2 ...)
Slipher1913lobAndromedaRarialVelocity tags: galaxies


# Display all entries that contain the 'galaxies' tag:
bibm tag
(Syntax is: KEY_OR_BIBCODE KEY_OR_BIBCODE2 ... tags: TAG TAG2 ...)
tags: galaxies
```

### 3.2.6 search

Search entries in the bibmanager database.

**Usage**

```
bibm search [-h] [-v VERB]
```

**Description**

This command will trigger a prompt where the user can search for entries in the bibmanager database by authors, years, title keywords, BibTeX key, or ADS bibcode. The matching results are displayed on screen according to the specified verbosity. Search syntax is similar to ADS searches (including tab completion).

Multiple author, title keyword, and year queries act with AND logic; whereas multiple-key queries and multiple-bibcode queries act with OR logic (see examples below).

There are five levels of verbosity:
verb < 0: Display only the keys of the entries
verb = 0: Display the title, year, first author, and key
verb = 1: Display additionally the ADS/arXiv urls and meta info
verb = 2: Display additionally the full list of authors
verb > 2: Display the full BibTeX entries

**Note:**

(1) There's no need to worry about case in author names, unless they conflict with the BibTeX format rules: http://mirror.easyname.at/ctan/info/bibtex/tamethebeast/ttb_en.pdf, p.23. For example, *author:"oliphant, t"* will match *'Travis Oliphant'* (because there is no ambiguity in first-von-last names), but *author:"travis oliphant"* wont match, because the lowercase *'travis'* will be interpreted as the von part of the last name.

(2) Title words/phrase searches are case-insensitive.

**Options**

**-v VERB, –verb VERB**
Set output verbosity.

**-h, –help**
Show this help message and exit.

**Examples**

**Note:** These examples below assume that you merged the sample bibfile already, i.e.: `bibm merge ~/.bibmanager/examples/sample.bib`

Searches follow the ADS search syntax. Pressing *tab* displays the search fields:

Fig. 1: The tab-completion also displays extra information at the bottom when navigating through some options.

Name examples:

```
# Search by last name (press tab to prompt the autocompleter):
bibm search
(Press 'tab' for autocomplete)
author:"oliphant"

Title: Array programming with NumPy, 2020
Authors: {Harris}, Charles R.; et al.
key: HarrisEtal2020natNumpy

Title: SciPy 1.0: fundamental algorithms for scientific computing in Python,
    2020
Authors: {Virtanen}, Pauli; et al.
key: VirtanenEtal2020natmeScipy
```

```
# Search by last name and initials (note blanks require one to use quotes):
bibm search
(Press 'tab' for autocomplete)
author:"oliphant, t"

Title: Array programming with NumPy, 2020
Authors: {Harris}, Charles R.; et al.
key: HarrisEtal2020natNumpy

Title: SciPy 1.0: fundamental algorithms for scientific computing in Python,
    2020
Authors: {Virtanen}, Pauli; et al.
key: VirtanenEtal2020natmeScipy
```

```
# Search by first-author only:
bibm search
author:"^Harris"

Title: Array programming with NumPy, 2020
Authors: {Harris}, Charles R.; et al.
key: HarrisEtal2020natNumpy
```

```
# Search multiple authors (using AND logic):
bibm search
(Press 'tab' for autocomplete)
author:"harris" author:"virtanen"

Title: Array programming with NumPy, 2020
Authors: {Harris}, Charles R.; et al.
key: HarrisEtal2020natNumpy

Title: SciPy 1.0: fundamental algorithms for scientific computing in Python,
    2020
Authors: {Virtanen}, Pauli; et al.
key: VirtanenEtal2020natmeScipy
```

Combine search fields:

```
# Search by author, year, and title words/phrases (using AND logic):
bibm search
(Press 'tab' for autocomplete)
author:"oliphant, t"  title:"numpy"

Title: Array programming with NumPy, 2020
Authors: {Harris}, Charles R.; et al.
key: HarrisEtal2020natNumpy
```

```
# Search multiple words/phrases in title (using AND logic):
bibm search
(Press 'tab' for autocomplete)
title:"HD 209458b" title:"atmospheric circulation"

Title: Atmospheric Circulation of Hot Jupiters: Coupled Radiative-Dynamical
       General Circulation Model Simulations of HD 189733b and HD 209458b,
       2009
Authors: {Showman}, Adam P.; et al.
key: ShowmanEtal2009apjRadGCM
```

Year examples:

```
# Search on specific year:
bibm search
(Press 'tab' for autocomplete)
year: 1913

Title: The radial velocity of the Andromeda Nebula, 1913
Authors: {Slipher}, V. M.
key: Slipher1913lobAndromedaRarialVelocity
```

```
# Search anything between the specified years (inclusive):
bibm search
(Press 'tab' for autocomplete)
year:2013-2016

Title: Novae in the Spiral Nebulae and the Island Universe Theory, 1917
Authors: {Curtis}, H. D.
key: Curtis1917paspIslandUniverseTheory

Title: The radial velocity of the Andromeda Nebula, 1913
Authors: {Slipher}, V. M.
key: Slipher1913lobAndromedaRarialVelocity
```

```
# Search anything up to the specified year (note this syntax is not available on ADS):
bibm search
(Press 'tab' for autocomplete)
year: -1917

Title: Novae in the Spiral Nebulae and the Island Universe Theory, 1917
Authors: {Curtis}, H. D.
key: Curtis1917paspIslandUniverseTheory

Title: The radial velocity of the Andromeda Nebula, 1913
Authors: {Slipher}, V. M.
key: Slipher1913lobAndromedaRarialVelocity
```

```
# Search anything since the specified year:
bibm search
(Press 'tab' for autocomplete)
author:"oliphant, t" year: 2020-

Title: Array programming with NumPy, 2020
Authors: {Harris}, Charles R.; et al.
key: HarrisEtal2020natNumpy

Title: SciPy 1.0: fundamental algorithms for scientific computing in Python,
    2020
Authors: {Virtanen}, Pauli; et al.
key: VirtanenEtal2020natmeScipy
```

ADS bibcode examples (same applies to searches by key):

```
# Search by bibcode:
bibm search
(Press 'tab' for autocomplete)
bibcode:2013A&A...558A..33A

Title: Astropy: A community Python package for astronomy, 2013
Authors: {Astropy Collaboration}; et al.
key: Astropycollab2013aaAstropy

# UTF-8 encoding also works just fine:
bibm search
(Press 'tab' for autocomplete)
bibcode:2013A%26A...558A..33A

Title: Astropy: A community Python package for astronomy, 2013
Authors: {Astropy Collaboration}; et al.
key: Astropycollab2013aaAstropy
```

Search multiple keys (same applies to multiple-bibcodes searches):

```
# Search multiple keys at once (using OR logic):
bibm search
(Press 'tab' for autocomplete)
key:Curtis1917paspIslandUniverseTheory key:Shapley1918apjDistanceGlobularClusters

Title: Novae in the Spiral Nebulae and the Island Universe Theory, 1917
Authors: {Curtis}, H. D.
key: Curtis1917paspIslandUniverseTheory

Title: Studies based on the colors and magnitudes in stellar clusters. VII.
       The distances, distribution in space, and dimensions of 69 globular
       clusters., 1918
Authors: {Shapley}, H.
key: Shapley1918apjDistanceGlobularClusters
```

Use the `-v VERB` command to set the verbosity:

```
# Display only the keys:
bibm search -v -1
(Press 'tab' for autocomplete)
year: 1910-1920
```

(continues on next page)

(continued from previous page)

```
Keys:
Curtis1917paspIslandUniverseTheory
Shapley1918apjDistanceGlobularClusters
Slipher1913lobAndromedaRarialVelocity
```

```
# Display title, year, first author, and all keys/urls:
bibm search -v 1
(Press 'tab' for autocomplete)
author:"Burbidge, E"

Title: Synthesis of the Elements in Stars, 1957
Authors: {Burbidge}, E. Margaret; et al.
bibcode:   1957RvMP...29..547B
ADS url:   https://ui.adsabs.harvard.edu/abs/1957RvMP...29..547B
key: BurbidgeEtal1957rvmpStellarElementSynthesis
```

```
# Display title, year, full author list, URLs, and meta info:
bibm search -v 2
(Press 'tab' for autocomplete)
author:"Burbidge, E"

Title: Synthesis of the Elements in Stars, 1957
Authors: {Burbidge}, E. Margaret; {Burbidge}, G. R.; {Fowler}, William A.; and
         {Hoyle}, F.
bibcode:   1957RvMP...29..547B
ADS url:   https://ui.adsabs.harvard.edu/abs/1957RvMP...29..547B
key: BurbidgeEtal1957rvmpStellarElementSynthesis
```

```
# Display full BibTeX entry:
bibm search -v 3
(Press 'tab' for autocomplete)
author:"Burbidge, E"

@ARTICLE{BurbidgeEtal1957rvmpStellarElementSynthesis,
       author = {{Burbidge}, E. Margaret and {Burbidge}, G.~R. and {Fowler}, William␣
↪A.
        and {Hoyle}, F.},
        title = "{Synthesis of the Elements in Stars}",
      journal = {Reviews of Modern Physics},
         year = 1957,
        month = Jan,
       volume = {29},
        pages = {547-650},
          doi = {10.1103/RevModPhys.29.547},
       adsurl = {https://ui.adsabs.harvard.edu/abs/1957RvMP...29..547B},
      adsnote = {Provided by the SAO/NASA Astrophysics Data System}
}
```

### 3.2.7 browse

Browse through the bibmanager database.
*(New since version 1.3)*

**Usage**

```
bibm browse [-h]
```

**Description**

Display the entire bibmanager database in an interactive full-screen application that lets you:
- Navigate through or search for specific entries
- Visualize the entries' full BibTeX content
- Select entries for printing to screen or to file
- Open the entries' PDF files
- Open the entries in ADS through the web browser
- Select sub-group of entries by tags *(New since version 1.4)*

**Options**

**-h, –help**
Show this help message and exit.

**Examples**

```
bibm browse
```

---

### 3.2.8 export

Export the bibmanager database into a bib file.

**Usage**

```
bibm export [-h] bibfile
```

**Description**

Export the entire bibmanager database into a bibliography file to a .bib or .bbl format according to the file extension of the 'bibfile' argument.

> **Caution:** For the moment, only export to .bib.

**Options**

**bibfile**
Path to an output BibTeX file.

**-h, –help**
Show this help message and exit.

**-meta**
Also include meta-information in output file.

**Examples**

```
bibm export my_file.bib
```

---

### 3.2.9 cleanup

Clean up a bibtex file of duplicates and outdated entries.

**Usage**

```
bibm cleanup [-h] [-ads] bibfile
```

**Description**

Clean up a BibTeX file by removing duplicates, sorting the entries, and (if requested) updating the entries by cross-checking against the ADS database. All of this is done independently of the `bibmanager` database. The original file will be preserved by prepending the string '*orig_yyyy_mm_dd_*' with the corresponding date.
*(New since version 1.1.2)*

**Options**

**bibfile**
Path to an existing BibTeX file.
**-ads**
Update the bibfile entries cross-checking against the ADS database.
**-h, –help**
Show this help message and exit.

**Examples**

```
# Remove duplicates and sort:
bibm cleanup file.bib

# Remove duplicates, update ADS entries, and sort:
bibm cleanup file.bib -ads
```

---

### 3.2.10 config

Manage the bibmanager configuration parameters.

**Usage**

```
bibm config [-h] [param] [value]
```

**Description**

This command displays or sets the value of bibmanager config parameters. These are the parameters that can be set by the user:

- The `style` parameter sets the color-syntax style of displayed BibTeX entries. The default style is 'autumn'. See http://pygments.org/demo/6780986/ for a demo of the style options. The available options are:

  default, emacs, friendly, colorful, autumn, murphy, manni, monokai, perldoc, pastie, borland, trac, native, fruity, bw, vim, vs, tango, rrt, xcode, igor, paraiso-light, paraiso-dark, lovelace, algol, algol_nu, arduino, rainbow_dash, abap

- The `text_editor` sets the text editor to use when editing the bibmanager manually (i.e., a call to: bibm edit). By default, bibmanager uses the OS-default text editor. Typical text editors are: emacs, vim, gedit. To set the OS-default editor, set text_editor to *'default'*. Note that aliases defined in the .bash file are not accessible.
- The `paper` parameter sets the default paper format for latex compilation outputs (not for pdflatex, which is automatic). Typical options are 'letter' (e.g., for ApJ articles) or 'A4' (e.g., for A&A).
- The `ads_token` parameter sets the ADS token required for ADS requests. To obtain a token, follow the steps described here: https://github.com/adsabs/adsabs-dev-api#access
- The `ads_display` parameter sets the number of entries to show at a time, for an ADS search query. The default number of entries to display is 20.
- The `home` parameter sets the `bibmanager` home directory (this could be very handy, e.g., by placing the database in a Dropbox folder to share the same database across multiple machines).

The number of arguments determines the action of this command (see examples below):

- with no arguments, display all available parameters and values.
- with the 'param' argument, display detailed info on the specified parameter and its current value.
- with both 'param' and 'value' arguments, set the value of the parameter.

**Options**

**param**
: A bibmanager config parameter.

**value**
: Value for a bibmanager config parameter.

**-h, –help**
: Show this help message and exit.

**Examples**

```
# Display all config parameters and values:
bibm config

bibmanager configuration file:
PARAMETER    VALUE
-----------  -----
style        autumn
text_editor  default
paper        letter
ads_token    None
ads_display  20
home         /home/user/.bibmanager/
```

```
# Display value and help for the ads_token parameter:
bibm config ads_token

The 'ads_token' parameter sets the ADS token required for ADS requests.
To obtain a token follow the two steps described here:
  https://github.com/adsabs/adsabs-dev-api#access

The current ADS token is 'None'
```

```
# Set the value of the BibTeX color-syntax:
bibm config style autumn

style updated to: autumn.
```

## 3.3 LaTeX Management

### 3.3.1 bibtex

Generate a bibtex file from a tex file.

**Usage**

```
bibm bibtex [-h] texfile [bibfile]
```

**Description**

This command generates a BibTeX file by searching for the citation keys in the input LaTeX file, and stores the output into BibTeX file, named after the argument in the *\bibliography{bib_file}* call in the LaTeX file. Alternatively, the user can specify the name of the output BibTeX file with the `bibfile` argument.

Any citation key not found in the bibmanager database, will be shown on the screen prompt.

**Options**

**texfile**
: Path to an existing LaTeX file.

**bibfile**
: Path to an output BibTeX file.

**-h, –help**
Show this help message and exit.

**Examples**

```
# Generate a BibTeX file with references cited in my_file.tex:
bibm bibtex my_file.tex

# Generate a BibTeX file with references cited in my_file.tex,
# naming the output file 'this_file.bib':
bibm bibtex my_file.tex this_file.bib
```

---

### 3.3.2 latex

Compile a LaTeX file with the latex command.

**Usage**

```
bibm latex [-h] texfile [paper]
```

**Description**

This command compiles a LaTeX file using the latex command, executing the following calls:

- Compute a BibTex file out of the citation calls in the .tex file.
- Remove all outputs from previous compilations.
- Call latex, bibtex, latex, latex to produce a .dvi file.
- Call dvi2ps and ps2pdf to produce the final .pdf file.

Prefer this command over the `bibm pdflatex` command when the LaTeX file contains .ps or .eps figures (as opposed to .pdf, .png, or .jpeg).

Note that the user does not necessarily need to be in the dir where the LaTeX files are.

**Options**

**texfile**
Path to an existing LaTeX file.

**paper**
Paper format, e.g., letter or A4 (default=letter).

**-h, –help**
Show this help message and exit.

**Examples**

```
# Compile a LaTeX project:
bibm latex my_file.tex

# File extension can be ommited:
bibm latex my_file

# Compile, setting explicitely the output paper format:
bibm latex my_file A4
```

### 3.3.3 pdflatex

Compile a LaTeX file with the pdflatex command.

**Usage**

```
bibm pdflatex [-h] texfile
```

**Description**

This command compiles a LaTeX file using the pdflatex command, executing the following calls:

- Compute a BibTeX file out of the citation calls in the LaTeX file.
- Remove all outputs from previous compilations.
- Call pdflatex, bibtex, pdflatex, pdflatex to produce a .pdf file.

Prefer this command over the `bibm latex` command when the LaTeX file contains .pdf, .png, or .jpeg figures (as opposed to .ps or .eps).

Note that the user does not necessarily need to be in the dir where the LaTeX files are.

**Options**

**texfile**
: Path to an existing LaTeX file.

**-h, –help**
: Show this help message and exit.

**Examples**

```
# Compile a LaTeX project:
bibm pdflatex my_file.tex

# File extension can be ommited:
bibm pdflatex my_file
```

## 3.4 ADS Management

**Note:** To enable the ADS functionality, first you need to obtain an ADS token[1], and set it into the `ads_tokend` config parameter. To do this:

1. Create an account and login into the new ADS system.
2. Get your token (or generate a new one) from here.
3. Set the `ads_token` bibmanager parameter:

```
# Set ads_token to 'my_ads_token':
bibm config ads_token my_ads_token
```

### 3.4.1 ads-search

Do a query on ADS.

**Usage**

```
bibm ads-search [-h] [-n] [-a] [-f] [-o]
```

**Description**

This command enables ADS queries. The query syntax is identical to a query in the new ADS's one-box search engine: https://ui.adsabs.harvard.edu. Here there is a detailed documentations for ADS searches: https://adsabs.github.io/help/search/search-syntax See below for typical query examples.

If you set the `-a/--add` flag, the code will prompt to add entries to the database right after showing the ADS search results. Similarly, set the `-f/--fetch` or `-o/--open` flags to prompt to fetch or open PDF files right after showing the ADS search results. Note that you can combine these to add and fetch/open at the same time (e.g., `bibm ads-search -a -o`), or you can fetch/open PDFs that are not in the database (e.g., `bibm ads-search -o`). *(New since version 1.2.7)*

**Note:** Note that a query will display at most 'ads_display' entries on screen at once (see `bibm config ads_display`). If a query matches more entries, the user can execute `bibm ads-search -n` to display the next set of entries.

**Caution:** When making an ADS query, note that ADS requires the field values (when necessary) to use *double* quotes. For example: *author:"^Fortney, J"*.

**Options**

**-n, –next**
Display next set of entries that matched the previous query.

[1] https://github.com/adsabs/adsabs-dev-api#access

**-a, –add**

Query to add an entry after displaying the search results.
*(New since version 1.2.7)*

**-f, –fetch**

Query to fetch a PDF after displaying the search results.
*(New since version 1.2.7)*

**-o, –open**

Query to fetch/open a PDF after displaying the search results.
*(New since version 1.2.7)*

**-h, –help**

Show this help message and exit.

### Examples

```
# Search entries for given author (press tab to prompt the autocompleter):
bibm ads-search
(Press 'tab' for autocomplete)
author:"^Fortney, J"

Title: Exploring A Photospheric Radius Correction to Model Secondary Eclipse
       Spectra for Transiting Exoplanets
Authors: Fortney, Jonathan J.; et al.
adsurl:  https://ui.adsabs.harvard.edu/abs/2019arXiv190400025F
bibcode: 2019arXiv190400025F

Title: Laboratory Needs for Exoplanet Climate Modeling
Authors: Fortney, J. J.; et al.
adsurl:  https://ui.adsabs.harvard.edu/abs/2018LPICo2065.2068F
bibcode: 2018LPICo2065.2068F

...

Showing entries 1--20 out of 74 matches.  To show the next set, execute:
bibm ads-search -n
```

Basic author search examples:

```
# Search by author in article:
bibm ads-search
(Press 'tab' for autocomplete)
author:"Fortney, J"

# Search by first author:
bibm ads-search
(Press 'tab' for autocomplete)
author:"^Fortney, J"

# Search multiple authors:
bibm ads-search
(Press 'tab' for autocomplete)
author:("Fortney, J" AND "Showman, A")
```

Search combining multiple fields:

```
# Search by author AND year:
bibm ads-search
(Press 'tab' for autocomplete)
author:"Fortney, J" year:2010

# Search by author AND year range:
bibm ads-search
(Press 'tab' for autocomplete)
author:"Fortney, J" year:2010-2019

# Search by author AND words/phrases in title:
bibm ads-search
(Press 'tab' for autocomplete)
author:"Fortney, J" title:Spitzer

# Search by author AND words/phrases in abstract:
bibm ads-search
(Press 'tab' for autocomplete)
author:"Fortney, J" abs:"HD 209458b"
```

Restrict searches to articles or peer-reviewed articles:

```
# Search by author AND request only articles:
bibm ads-search
(Press 'tab' for autocomplete)
author:"Fortney, J" property:article

# Search by author AND request only peer-reviewed articles:
bibm ads-search
(Press 'tab' for autocomplete)
author:"Fortney, J" property:refereed
```

Add entries and fetch/open PDFs right after the ADS search:

```
# Search and prompt to open a PDF right after (fetched PDF is not stored in database):
bibm ads-search -o
(Press 'tab' for autocomplete)
author:"^Fortney, J" property:refereed year:2015-2019

Title: Exploring a Photospheric Radius Correction to Model Secondary Eclipse
       Spectra for Transiting Exoplanets
Authors: Fortney, Jonathan J.; et al.
adsurl:  https://ui.adsabs.harvard.edu/abs/2019ApJ...880L..16F
bibcode: 2019ApJ...880L..16F
...

Fetch/open entry from ADS:
Syntax is:  key: KEY_VALUE FILENAME
     or:  bibcode: BIBCODE_VALUE FILENAME
bibcode: 2019ApJ...880L..16F Fortney2019.pdf
```

```
# Search and prompt to add entry to database right after:
bibm ads-search -a
(Press 'tab' for autocomplete)
author:"^Fortney, J" property:refereed year:2015-2019
```

(continues on next page)

(continued from previous page)

```
Title: Exploring a Photospheric Radius Correction to Model Secondary Eclipse
       Spectra for Transiting Exoplanets
Authors: Fortney, Jonathan J.; et al.
adsurl:  https://ui.adsabs.harvard.edu/abs/2019ApJ...880L..16F
bibcode: 2019ApJ...880L..16F
...

Add entry from ADS:
Enter pairs of ADS bibcodes and BibTeX keys, one pair per line
separated by blanks (press META+ENTER or ESCAPE ENTER when done):
2019ApJ...880L..16F FortneyEtal2019apjPhotosphericRadius
```

```
# Search and prompt to add entry and fetch/open its PDF right after:
bibm ads-search -a -f
(Press 'tab' for autocomplete)
author:"^Fortney, J" property:refereed year:2015-2019

Title: Exploring a Photospheric Radius Correction to Model Secondary Eclipse
       Spectra for Transiting Exoplanets
Authors: Fortney, Jonathan J.; et al.
adsurl:  https://ui.adsabs.harvard.edu/abs/2019ApJ...880L..16F
bibcode: 2019ApJ...880L..16F
...

Add entry from ADS:
Enter pairs of ADS bibcodes and BibTeX keys, one pair per line
separated by blanks (press META+ENTER or ESCAPE ENTER when done):
2019ApJ...880L..16F FortneyEtal2019apjPhotosphericRadius
```

### 3.4.2 ads-add

Add entries from ADS by bibcode into the bibmanager database.

**Usage**

```
bibm ads-add [-h] [-f] [-o] [bibcode key] [tag1 [tag2 ...]]
```

**Description**

This command add BibTeX entries from ADS by specifying pairs of ADS bibcodes and BibTeX keys.

Executing this command without arguments (i.e., `bibm ads-add`) launches an interactive prompt session allowing the user to enter multiple bibcode, key pairs.

By default, added entries replace previously existent entries in the bibmanager database.

With the optional arguments `-f/--fetch` or `-o/--open`, the code will attempt to fetch and fetch/open (respectively) the associated PDF files of the added entries.
*(New since version 1.2.7)*

Either at `bibm ads-add` or later via the prompt you can specify tags for the entries to be add.
*(New since version 1.4)*

#### Options

**bibcode**
The ADS bibcode of an entry.

**key**
BibTeX key to assign to the entry.

**tags**
Optional BibTeX tags to assign to the entries.
*(New since version 1.4)*

**-f, –fetch**
Fetch the PDF of the added entries.
*(New since version 1.2.7)*

**-o, –open**
Fetch and open the PDF of the added entries.
*(New since version 1.2.7)*

**-h, –help**
Show this help message and exit.

#### Examples

```
# Let's search and add the greatest astronomy PhD thesis of all times:
bibm ads-search
(Press 'tab' for autocomplete)
author:"^payne, cecilia" doctype:phdthesis

Title: Stellar Atmospheres; a Contribution to the Observational Study of High
       Temperature in the Reversing Layers of Stars.
Authors: Payne, Cecilia Helena
adsurl:  https://ui.adsabs.harvard.edu/abs/1925PhDT.........1P
bibcode: 1925PhDT.........1P
```

```
# Add the entry to the bibmanager database:
bibm ads-add 1925PhDT.........1P Payne1925phdStellarAtmospheres
```

The user can optionally assign tags or request to fetch/open PDFs:

```
# Add the entry and assign a 'stars' tag to it:
bibm ads-add 1925PhDT.........1P Payne1925phdStellarAtmospheres stars


# Add the entry and fetch its PDF:
bibm ads-add -f 1925PhDT.........1P Payne1925phdStellarAtmospheres

# Add the entry and fetch/open its PDF:
bibm ads-add -o 1925PhDT.........1P Payne1925phdStellarAtmospheres
```

Alternatively, the call can be done without arguments, which allow the user to request multiple entries at once (and as above, set tags to each entry as desired):

```
# A call without bibcode,key arguments (interactive prompt):
bibm ads-add
Enter pairs of ADS bibcodes and BibTeX keys (plus optional tags)
Use one line for each BibTeX entry, separate fields with blank spaces.
(press META+ENTER or ESCAPE ENTER when done):
1925PhDT.........1P Payne1925phdStellarAtmospheres stars

# Multiple entries at once, assigning tags (interactive prompt):
bibm ads-add
Enter pairs of ADS bibcodes and BibTeX keys (plus optional tags)
Use one line for each BibTeX entry, separate fields with blank spaces.
(press META+ENTER or ESCAPE ENTER when done):
1925PhDT.........1P Payne1925phdStellarAtmospheres stars
1957RvMP...29..547B BurbidgeEtal1957rvmpStellarSynthesis stars nucleosynthesis
```

### 3.4.3 ads-update

Update bibmanager database cross-checking entries with ADS.

**Usage**

```
bibm ads-update [-h] [update_keys]
```

**Description**

This command triggers an ADS search of all entries in the `bibmanager` database that have a `bibcode`. Replacing these entries with the output from ADS. The main utility of this command is to auto-update entries that were added as arXiv version, with their published version.

For arXiv updates, this command updates automatically the year and journal of the key (where possible). This is done by searching for the year and the string *'arxiv'* in the key, using the bibcode info. For example, an entry with key *'NameEtal2010arxivGJ436b'* whose bibcode changed from *'2010arXiv1007.0324B'* to *'2011ApJ...731...16B'*, will have a new key *'NameEtal2011apjGJ436b'*. To disable this feature, set the `update_keys` optional argument to *'no'*.

**Options**

**update_keys**
: Update the keys of the entries. (choose from: {no, arxiv}, default: arxiv).

**-h, –help**
: Show this help message and exit.

**Examples**

**Note:** These example outputs assume that you merged the sample bibfile already, i.e.: `bibm merge ~/.bibmanager/examples/sample.bib`

```
# Look at this entry with old info from arXiv:
bibm search -v
author:"^Beaulieu"

Title: Methane in the Atmosphere of the Transiting Hot Neptune GJ436b?, 2010
Authors: {Beaulieu}, J.-P.; et al.
bibcode:   2010arXiv1007.0324B
ADS url:   http://adsabs.harvard.edu/abs/2010arXiv1007.0324B
arXiv url: http://arxiv.org/abs/arXiv:1007.0324
key: BeaulieuEtal2010arxivGJ436b


# Update bibmanager entries that are in ADS:
bibm ads-update

Merged 0 new entries.
(Not counting updated references)
There were 1 entries updated from ArXiv to their peer-reviewed version.
These ones changed their key:
BeaulieuEtal2010arxivGJ436b -> BeaulieuEtal2011apjGJ436b


# Let's take a look at this entry again:
bibm search -v
author:"^Beaulieu"

Title: Methane in the Atmosphere of the Transiting Hot Neptune GJ436B?, 2011
Authors: {Beaulieu}, J. -P.; et al.
bibcode:   2011ApJ...731...16B
ADS url:   https://ui.adsabs.harvard.edu/abs/2011ApJ...731...16B
arXiv url: http://arxiv.org/abs/1007.0324
key: BeaulieuEtal2011apjGJ436b
```

**Note:** There might be cases when one does not want to ADS-update an entry. To prevent this to happen, the user can set the *freeze* meta-parameter through the `bibm edit` command (see *edit*).

**References**

## 3.5 PDF Management

Since version 1.2, `bibmanager` also doubles as a PDF database. The following commands describe how to fetch PDF entries from ADS, or manually link and open the PDF files associated to the `bibmanager` database. All PDF files are stored in the `home`/pdf folder (see *config*, for more info to set `home`).

PDF files can also be manually linked to the database entries via the `bibm edit` command (see *Meta-Information*).

### 3.5.1 fetch

Fetch a PDF file from ADS.

**Usage**

```
bibm fetch [-h] [-o] [keycode] [filename]
```

**Description**

This command attempts to fetch from ADS the PDF file associated to a Bibtex entry in the `bibmanager` database. The request is made to the Journal, then the ADS server, and lastly to ArXiv until one succeeds. The entry is specified by either the BibTex key or ADS bibcode, these can be specified on the initial command, or will be queried after through the prompt (see examples).

If the output PDF filename is not specified, the routine will guess a name with this syntax: LastnameYYYY_Journal_vol_page.pdf

Requests for entries not in the database can be made only by ADS bibcode (and auto-completion wont be able to predict their bibcode IDs).
*(New since version 1.2)*

**Options**

**keycode**
Either a BibTex key or an ADS bibcode identifier.

**filename**
Name for fetched PDF file.

**-h, –help**
Show this help message and exit

**-o, –open**
Open the fetched PDF if the request succeeded.

**Examples**

**Note:** These examples assume that you have this entry into the database: Rubin, V. C. et al. (1980), ApJ, 238, 471. E.g., with: `bibm ads-add 1980ApJ...238..471R RubinEtal1980apjGalaxiesRotation`

A `bibm fetch` call without arguments triggers a prompt search with auto-complete help:

Fig. 2: Note that as you navigate through the options, the display shows info about the entries at the bottom. Also, as long as the user provides a valid bibcode, you can fetch any PDF (no need to be an entry in the database).

```
# Fetch PDF for entry by BibTex key:
bibm fetch RubinEtal1980apjGalaxiesRotation

Fetching PDF file from Journal website:
Request failed with status code 404: NOT FOUND
Fetching PDF file from ADS website:
```

(continues on next page)

(continued from previous page)

```
Saved PDF to: '/home/user/.bibmanager/pdf/Rubin1980_ApJ_238_471.pdf'.

To open the PDF file, execute:
bibm open RubinEtal1980apjGalaxiesRotation

# Fetch PDF fir entry by ADS bibcode:
bibm fetch 1980ApJ...238..471R
...
Fetching PDF file from ADS website:
Saved PDF to: '/home/user/.bibmanager/pdf/Rubin1980_ApJ_238_471.pdf'.

To open the PDF file, execute:
bibm open RubinEtal1980apjGalaxiesRotation


# Fetch and set the output filename:
bibm fetch 1980ApJ...238..471R  Rubin1980_gals_rotation.pdf
...
Fetching PDF file from ADS website:
Saved PDF to: '/home/user/.bibmanager/pdf/Rubin1980_gals_rotation.pdf'.

To open the PDF file, execute:
bibm open RubinEtal1980apjGalaxiesRotation
```

A `bibm fetch` call with the `-o`/`--open` flag automatically opens the PDF file after a successful fetch:

```
# Use prompt to find the BibTex entry (and open the PDF right after fetching):
bibm fetch RubinEtal1980apjGalaxiesRotation -o

Fetching PDF file from Journal website:
Request failed with status code 404: NOT FOUND
Fetching PDF file from ADS website:
Saved PDF to: '/home/user/.bibmanager/pdf/Rubin1980_ApJ_238_471.pdf'.
```

---

### 3.5.2 open

Open the PDF file of a BibTex entry in the database.

**Usage**

```
bibm open [-h] [keycode]
```

**Description**

This command opens the PDF file associated to a Bibtex entry in the `bibmanager` database. The entry is specified by either its BibTex key, its ADS bibcode, or its PDF filename. These can be specified on the initial command, or will be queried through the prompt (with auto-complete help).

If the user requests a PDF for an entry without a PDF file but with an ADS bibcode, `bibmanager` will ask if the user wants to fetch the PDF from ADS.
*(New since version 1.2)*

**Options**

**keycode**
Either a key or an ADS bibcode identifier.

**-h, –help**
Show this help message and exit

**Examples**

```
# Open setting the BibTex key:
bibm open RubinEtal1980apjGalaxiesRotation

# Open setting the ADS bibcode:
bibm open 1980ApJ...238..471R

# Open setting the PDF filename:
bibm open Rubin1980_ApJ_238_471.pdf
```

```
# Use the prompt to find the BibTex entry:
bibm open
Syntax is:  key: KEY_VALUE
       or:  bibcode: BIBCODE_VALUE
       or:  pdf: PDF_VALUE
(Press 'tab' for autocomplete)
key: RubinEtal1980apjGalaxiesRotation
```

---

### 3.5.3 pdf

Link a PDF file to a BibTex entry in the database.

**Usage**

```
bibm pdf [-h] [keycode pdf] [name]
```

**Description**

This command manually links an existing PDF file to a Bibtex entry in the `bibmanager` database. The PDF file is moved to the *'home/pdf'* folder (see *config*). The entry is specified by either the BibTex key or ADS bibcode, these can be specified on the initial command, or will be queried after through the prompt (see examples).

If the output PDF filename is not specified, the code will preserve the file name. If the user sets *'guess'* as filename, the code will guess a name based on the BibTex information.
*(New since version 1.2)*

**Options**

**keycode**

Either a key or an ADS bibcode identifier.

**pdf**
Path to PDF file to link to entry.

**filename**
New name for the linked PDF file.

**-h, –help**
Show this help message and exit

**Examples**

Say you already have an article's PDF file here: *~/Downloads/Rubin1980.pdf*

```
# Link a downloaded PDF file to an entry:
bibm pdf 1980ApJ...238..471R ~/Downloads/Rubin1980.pdf
Saved PDF to: '/home/user/.bibmanager/pdf/Rubin1980.pdf'.

# Link a downloaded PDF file (guessing the name from BibTex):
bibm pdf 1980ApJ...238..471R ~/Downloads/Rubin1980.pdf guess
Saved PDF to: '/home/user/.bibmanager/pdf/Rubin1980_ApJ_238_471.pdf'.

# Link a downloaded PDF file (renaming the file):
bibm pdf 1980ApJ...238..471R ~/Downloads/Burbidge1957.pdf RubinEtal_1980.pdf
Saved PDF to: '/home/user/.bibmanager/pdf/RubinEtal_1980.pdf'.
```

```
# Use the prompt to find the BibTex entry:
bibm pdf
Syntax is:  key: KEY_VALUE PDF_FILE FILENAME
       or:  bibcode: BIBCODE_VALUE PDF_FILE FILENAME
(output FILENAME is optional, set it to guess for automated naming)

key: RubinEtal1980apjGalaxiesRotation ~/Downloads/Rubin1980.pdf
Saved PDF to: '/home/user/.bibmanager/pdf/Rubin1980.pdf'.
```

## 3.6 FAQs and Resources

### 3.6.1 Frequently Asked Questions

#### Why should I use `bibmanager`? I have already my working ecosystem.

`bibmanager` simply makes your life easier, keeping all of your references at the tip of your fingers:

- No need to wonder whether to start a new BibTeX file from scratch or reuse an old one (probably a massive file), nor to think which was the most current.
- Easily add new entries: manually, from your existing BibTeX files, or from ADS, without risking having duplicates.
- Generate BibTeX files and compile a LaTeX project with a single command.
- You can stay up to date with ADS with a single command.

### I use several machines to work, can I use a single database across all of them?

Yes!, since vesion 1.2 `bibmanager` has a `home` config parameter which sets the location of the database. By default `home` points at *~/.bibmanager*; however, you can set the `home` parameter into a folder in a Dropbox-type of system. The only nuance is that you'll need to install and configure `bibmanager` in each machine, but now all of them will be pointing to the same database.

Note that the folder containing the associated PDF files (i.e., `home`/pdf) will also be moved into the new location.

---

### I compiled my LaTeX file before merging its bibfile, did I just overwite my own BibTeX file?

No, if `bibmanager` has to overwrite a bibfile edited by the user (say, *'myrefs.bib'*), it saves the old file (and date) as *'orig_yyyy-mm-dd_myrefs.bib'*.

---

### I meged the BibTeX file for my LaTeX project, but it says there are missing references when I compile. What's going on?

Probably, there were duplicate entries with previous entries in the `bibmanager` database, but they had different keys. Simply, do a search of your missing reference, to check it's key, something like:

```
# Surely, first author and year have not changed:
bibm search
author:"^Author" year:the_year
```

Now, you can update the key in the LaTeX file (and as a bonus, you wont run into having duplicate entries in the future).

---

### That Raycast extension looks sweet! How do I install it?

Right, Raycast rocks. To install Raycast, simply go to their homepage (https://www.raycast.com/), click on the `Download` tab in the upper right corner and follow the instruction of the installer.

To install the `bibmanager` extension, click on the `Store` tab (from Raycast home's page), and search for bibmanager. Once redirected, you'll see a `Install Extension` tab, click it and follow the instructions.

---

### I installed `bibmanager` while being in a virtual environment. But I don't want to start the virtual env every time I want to use `bibm`.

(This is not a question!, please state your FAQ in the form of a question) Anyway, no worries, the `bibm` executable entry point is safe to use even if you are not in the virtual environment. What you can do is to add the path to the entry point into your bash:

```
# first, search for the entry-point executable (while in the virtual environment):
which bibm

/home/username/py36/bin/bibm
```

Then, add an alias with that path into your bash, e.g.: `alias bibm='/home/username/py36/bin/bibm'`. Now, you can access `bibm` at any time.

---

#### A unique database? Does it mean I need to have better keys to differentiate my entries?

Certainly, as a database grows, short BibTeX keys like *'LastnameYYYY'* are sub-optimal, since they may conflict with other entries, and are not descriptive enough. A good practice is to adopt a longer, more descriptive format. I personally suggests this one:

| Authors | Format | Example |
|---|---|---|
| 1 | LastYYYYjournalDescription | Shapley1918apjDistanceGClusters |
| 2 | Last1Last2YYYYjournalDescription | PerezGranger2007cseIPython |
| 3 | LastEtalYYYYjournalDescription | AstropycollabEtal2013aaAstropy |

That is:

- the first-author last name (capitalized)
- either nothing, the second-author last name (capitalized), or *'Etal'*
- the publication year
- the journal initials if any (and lower-cased)
- a couple words from the title that describe the article (capitalized or best format at user's discretion).

These long keys will keep you from running into issues, and will make the citations in your LaTeX documents nearly unambiguous at sight.

---

#### The code breaks with `UnicodeEncodeError` when running over ssh. What's going on?

As correctly guessed in this Stack Overflow post, Python cannot determine the terminal encoding, and falls back to ASCII. You can fix this by setting the following environment variable, e.g., into your bash:

```
export PYTHONIOENCODING=utf-8
```

---

### 3.6.2 Resources

Docs for queries in the new ADS:
http://adsabs.github.io/help/search/search-syntax

The ADS API:
https://github.com/adsabs/adsabs-dev-api

BibTeX author format:
http://mirror.easyname.at/ctan/info/bibtex/tamethebeast/ttb_en.pdf
http://texdoc.net/texmf-dist/doc/bibtex/base/btxdoc.pdf

Pygment style BibTeX options:
http://pygments.org/demo/6693571/

Set up conda:
https://github.com/conda-forge/staged-recipes

Testing:
https://docs.pytest.org/
http://pythontesting.net/framework/pytest/pytest-fixtures-nuts-bolts/
https://blog.dbrgn.ch/2016/2/18/overriding_default_arguments_in_pytest/
https://www.patricksoftwareblog.com/monkeypatching-with-pytest/
https://requests-mock.readthedocs.io/en/

Useful info from stackoverflow:
https://stackoverflow.com/questions/17317219
https://stackoverflow.com/questions/18011902
https://stackoverflow.com/questions/26899001
https://stackoverflow.com/questions/2241348
https://stackoverflow.com/questions/1158076
https://stackoverflow.com/questions/17374526
https://stackoverflow.com/questions/43165341

## 3.7 API

### 3.7.1 bibmanager

### 3.7.2 bibmanager.bib_manager

**class** bibmanager.bib_manager.**Bib**(*entry*, *pdf=None*, *freeze=None*, *tags=[]*)

```
  Bibliographic-entry object.

.. code-block:: pycon

  Create a Bib() object from given entry.

  Parameters
  ----------
  entry: String
      A bibliographic entry text.
  pdf: String
```

(continues on next page)

(continued from previous page)

```
    Name of PDF file associated with this entry.
  freeze: Bool
    Flag that, if True, prevents the entry to be ADS-updated.

  Examples
  --------
  >>> import bibmanager.bib_manager as bm
  >>> entry = '''@Misc{JonesEtal2001scipy,
            author = {Eric Jones and Travis Oliphant and Pearu Peterson},
            title  = {{SciPy}: Open source scientific tools for {Python}},
            year   = {2001},
          }'''
  >>> bib = bm.Bib(entry)
  >>> print(bib.title)
  SciPy: Open source scientific tools for Python
  >>> for author in bib.authors:
  >>>    print(author)
  Author(last='Jones', first='Eric', von='', jr='')
  Author(last='Oliphant', first='Travis', von='', jr='')
  Author(last='Peterson', first='Pearu', von='', jr='')
  >>> print(bib.sort_author)
  Sort_author(last='jones', first='e', von='', jr='', year=2001, month=13)
```

bibmanager.bib_manager.**display_bibs**(*labels*, *bibs*, *meta=False*)

```
Display a list of bib entries on screen with flying colors.

Parameters
----------
labels: List of Strings
    Header labels to show above each Bib() entry.
bibs: List of Bib() objects
    BibTeX entries to display.
meta: Bool
    If True, also display the meta-information.

Examples
--------
>>> import bibmanager.bib_manager as bm
>>> e1 = '''@Misc{JonesEtal2001scipy,
       author = {Eric Jones and Travis Oliphant and Pearu Peterson},
       title  = {{SciPy}: Open source scientific tools for {Python}},
       year   = {2001},
     }'''
>>> e2 = '''@Misc{Jones2001,
       author = {Eric Jones and Travis Oliphant and Pearu Peterson},
       title  = {SciPy: Open source scientific tools for Python},
       year   = {2001},
     }'''
>>> bibs = [bm.Bib(e1), bm.Bib(e2)]
>>> bm.display_bibs(["DATABASE:\n", "NEW:\n"], bibs)
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
DATABASE:
@Misc{JonesEtal2001scipy,
       author = {Eric Jones and Travis Oliphant and Pearu Peterson},
       title  = {{SciPy}: Open source scientific tools for {Python}},
       year   = {2001},
```

(continues on next page)

(continued from previous page)

```
    }

NEW:
@Misc{Jones2001,
       author = {Eric Jones and Travis Oliphant and Pearu Peterson},
       title  = {SciPy: Open source scientific tools for Python},
       year   = {2001},
    }
```

bibmanager.bib_manager.**display_list**(*bibs*, *verb=-1*)

```
Display a list of BibTeX entries with different verbosity levels.

Although this might seem a duplication of display_bibs(), this
function is meant to provide multiple levels of verbosity and
generally to display longer lists of entries.

Parameters
----------
bibs: List of Bib() objects
    BibTeX entries to display.
verb: Integer
    The desired verbosity level:
    verb < 0: Display only the keys.
    verb = 0: Display the title, year, first author, and key.
    verb = 1: Display additionally the ADS and arXiv urls.
    verb = 2: Display additionally the full list of authors.
    verb > 2: Display the full BibTeX entry.
```

bibmanager.bib_manager.**remove_duplicates**(*bibs*, *field*)

```
Look for duplicates (within a same list of entries) by field and
remove them (in place).

Parameters
----------
bibs: List of Bib() objects
    Entries to filter.
field: String
    Field to use for filtering ('doi', 'isbn', 'bibcode', or 'eprint').
```

bibmanager.bib_manager.**filter_field**(*bibs*, *new*, *field*, *take*)

```
Filter duplicate entries by field between new and bibs.
This routine modifies new removing the duplicates, and may modify
bibs (depending on take argument).

Parameters
----------
bibs: List of Bib() objects
    Database entries.
new: List of Bib() objects
    New entries to add.
field: String
    Field to use for filtering.
take: String
    Decision-making protocol to resolve conflicts when there are
```

(continues on next page)

(continued from previous page)

```
    duplicated entries:
    'old': Take the database entry over new.
    'new': Take the new entry over the database.
    'ask': Ask user to decide (interactively).
```

bibmanager.bib_manager.**read_file**(*bibfile=None*, *text=None*)

```
Create a list of Bib() objects from a BibTeX file (.bib file).

Parameters
----------
bibfile: String
    Path to an existing .bib file.
text: String
    Content of a .bib file (ignored if bibfile is not None).

Returns
-------
bibs: List of Bib() objects
    List of Bib() objects of BibTeX entries in bibfile, sorted by
    Sort_author() fields.

Examples
--------
>>> import bibmanager.bib_manager as bm
>>> text = (
>>>     "@misc{AASteamHendrickson2018aastex62,\n"
>>>     "author = {{AAS Journals Team} and {Hendrickson}, Amy},\n"
>>>     "title  = {{AASJournals/AASTeX60: Version 6.2 official release}},\n"
>>>     "year   = 2018\n"
>>>     "}")
>>> bibs = bm.read_file(text=text)
```

bibmanager.bib_manager.**save**(*entries*)

```
Save list of Bib() entries into bibmanager pickle database.

Parameters
----------
entries: List of Bib() objects
    bib files to store.

Examples
--------
>>> import bibmanager.bib_manager as bm
>>> # TBD: Load some entries
>>> bm.save(entries)
```

bibmanager.bib_manager.**load**(*bm_database=None*)

```
Load a Bibmanager database of BibTeX entries.

Parameters
----------
bm_database: String
    A Bibmanager pickle database file.  If None, default's the
    database in system.
```

(continues on next page)

(continued from previous page)

```
Returns
-------
bibs: List Bib() instances
    Return an empty list if there is no database file.

Examples
--------
>>> import bibmanager.bib_manager as bm
>>> bibs = bm.load()
```

bibmanager.bib_manager.**find**(*key=None*, *bibcode=None*, *bibs=None*)

```
Find an specific entry in the database.

Parameters
----------
key: String
    Key of entry to find.
bibcode: String
    Bibcode of entry to find (ignored if key is not None).
bibs: List of Bib() instances
    Database where to search.  If None, load the Bibmanager database.

Returns
-------
bib: a Bib() instance
    BibTex matching either key or bibcode.
```

bibmanager.bib_manager.**get_version**(*bm_database=None*)

```
Get version of pickled database file.
If database does not exists, return current bibmanager version.
If database does not contain version, return '0.0.0'.

Parameters
----------
bm_database: String
    A Bibmanager pickle database file.  If None, default's the
    database in system.

Returns
-------
version: String
    bibmanager version of pickled objects.

Examples
--------
>>> import bibmanager.bib_manager as bm
>>> bibs = bm.get_version()
```

bibmanager.bib_manager.**export**(*entries*, *bibfile=None*, *meta=False*)

```
Export list of Bib() entries into a .bib file.

Parameters
----------
```

(continues on next page)

(continued from previous page)

```
entries: List of Bib() objects
    Entries to export.
bibfile: String
    Output .bib file name.  If None, export into home directory.
meta: Bool
    If True, include meta information before the entries on the
    output bib file.
```

bibmanager.bib_manager.**merge**(*bibfile=None*, *new=None*, *take='old'*, *base=None*)

```
Merge entries from a new bibfile into the bibmanager database
(or into an input database).

Parameters
----------
bibfile: String
    New .bib file to merge into the bibmanager database.
new: List of Bib() objects
    List of new BibTeX entries (ignored if bibfile is not None).
take: String
    Decision-making protocol to resolve conflicts when there are
    partially duplicated entries.
    'old': Take the database entry over new.
    'new': Take the new entry over the database.
    'ask': Ask user to decide (interactively).
base: List of Bib() objects
    If None, merge new entries into the bibmanager database.
    If not None, merge new entries into base.

Returns
-------
bibs: List of Bib() objects
    Merged list of BibTeX entries.

Examples
--------
>>> import bibmanager.bib_manager as bm
>>> import os
>>> # TBD: Need to add sample2.bib into package.
>>> newbib = os.path.expanduser("~") + "/.bibmanager/examples/sample2.bib"
>>> # Merge newbib into database:
>>> bm.merge(newbib, take='old')
```

bibmanager.bib_manager.**init**(*bibfile=None*, *reset_db=True*, *reset_config=False*)

```
Initialize bibmanager, reset database entries and config parameters.

Parameters
----------
bibfile: String
    A bibfile to include as the new bibmanager database.
    If None, reset the bibmanager database with a clean slate.
reset_db: Bool
    If True, reset the bibmanager database.
reset_config: Bool
    If True, reset the config file.
```

(continues on next page)

(continued from previous page)

```
Examples
--------
>>> import bibmanager.bib_manager as bm
>>> import os
>>> bibfile = os.path.expanduser("~") + "/.bibmanager/examples/sample.bib"
>>> bm.init(bibfile)
```

bibmanager.bib_manager.**add_entries**(*take='ask'*)

```
Manually add BibTeX entries through the prompt.

Parameters
----------
take: String
    Decision-making protocol to resolve conflicts when there are
    partially duplicated entries.
    'old': Take the database entry over new.
    'new': Take the new entry over the database.
    'ask': Ask user to decide (interactively).
```

bibmanager.bib_manager.**edit**()

```
Manually edit the bibfile database in text editor.

Resources
---------
https://stackoverflow.com/questions/17317219/
https://docs.python.org/3.6/library/subprocess.html
```

bibmanager.bib_manager.**search**(*authors=None*, *year=None*, *title=None*, *key=None*, *bibcode=None*, *tags=None*)

```
Search in bibmanager database by authors, year, or title keywords.

Parameters
----------
authors: String or List of strings
    An author name (or list of names) with BibTeX format (see parse_name()
    docstring).  To restrict search to a first author, prepend the
    '^' character to a name.
year: Integer or two-element integer tuple
    If integer, match against year; if tuple, minimum and maximum
    matching years (including).
title: String or iterable (list, tuple, or ndarray of strings)
    Match entries that contain all input strings in the title (ignore case).
key: String or list of strings
    Match any entry whose key is in the input key.
bibcode: String or list of strings
    Match any entry whose bibcode is in the input bibcode.
tags: String or list of strings
    Match entries containing all specified tags.

Returns
-------
matches: List of Bib() objects
    Entries that match all input criteria.
```

(continues on next page)

(continued from previous page)

```
Examples
--------
>>> import bibmanager.bib_manager as bm
>>> # Search by last name:
>>> matches = bm.search(authors="Cubillos")
>>> # Search by last name and initial:
>>> matches = bm.search(authors="Cubillos, P")
>>> # Search by author in given year:
>>> matches = bm.search(authors="Cubillos, P", year=2017)
>>> # Search by first author and co-author (using AND logic):
>>> matches = bm.search(authors=["^Cubillos", "Blecic"])
>>> # Search by keyword in title:
>>> matches = bm.search(title="Spitzer")
>>> # Search by keywords in title (using AND logic):
>>> matches = bm.search(title=["HD 189", "HD 209"])
>>> # Search by key (note that unlike the other fields, key and
>>> # bibcode use OR logic, so you can get many items at once):
>>> matches = bm.search(key="Astropycollab2013aaAstropy")
>>> # Search by bibcode (note no need to worry about UTF-8 encoding):
>>> matches = bm.search(bibcode=["2013A%26A...558A..33A",
>>>                              "1957RvMP...29..547B",
>>>                              "2017AJ....153....3C"])
```

bibmanager.bib_manager.**prompt_search**(*keywords*, *field*, *prompt_text*)

```
Do an interactive prompt search in the Bibmanager database by
the given keywords, with auto-complete and auto-suggest only
offering non-None values of the given field.
Only one keyword must be set in the prompt.
A bottom toolbar dynamically shows additional info.

Parameters
----------
keywords: List of strings
    BibTex keywords to search by.
field: String
    Filtering BibTex field for auto-complete and auto-suggest.
prompt_text: String
    Text to display when launching the prompt.

Returns
-------
kw_input: List of strings
    List of the parsed input (same order as keywords).
    Items are None for the keywords not defined.
extra: List of strings
    Any further word written in the prompt.

Examples
--------
>>> import bibmanager.bib_manager as bm
>>> # Search by key or bibcode, of entries with non-None bibcode:
>>> keywords = ['key', 'bibcode']
>>> field = 'bibcode'
>>> prompt_text = ("Sample search  (Press 'tab' for autocomplete):\n")
>>> prompt_input = bm.prompt_search(keywords, field, prompt_text)
Sample search  (Press 'tab' for autocomplete):
```

(continues on next page)

(continued from previous page)

```
key: Astropy2013aaAstroPy
>>> # Look at the results (list corresponds to [key, bibcode]):
>>> print(prompt_input[0])
['Astropy2013aaAstroPy', None]
>>> print(f'extra = {prompt_input[1]}')
extra = [None]

>>> # Repeat search, now by bibcode:
>>> prompt_input = u.prompt_search(keywords, field, prompt_text)
Sample search  (Press 'tab' for autocomplete):
bibcode: 2013A&A...558A..33A
>>> print(prompt_input[0])
[None, '2013A&A...558A..33A']
```

bibmanager.bib_manager.**prompt_search_tags**(*prompt_text*)

```
Do an interactive prompt search in the Bibmanager database by
the given keywords, with auto-complete and auto-suggest only
offering non-None values of the given field.
Only one keyword must be set in the prompt.
A bottom toolbar dynamically shows additional info.

Parameters
----------
prompt_text: String
    Text to display when launching the prompt.

Returns
-------
kw_input: List of strings
    List of the parsed input (same order as keywords).
    Items are None for the keywords not defined.
```

bibmanager.bib_manager.**browse**()

```
A browser for the bibmanager database.
```

### 3.7.3 bibmanager.config_manager

bibmanager.config_manager.**help**(*key*)

```
Display help information.

Parameters
----------
key: String
    A bibmanager config parameter.
```

bibmanager.config_manager.**display**(*key=None*)

```
Display the value(s) of the bibmanager config file on the prompt.

Parameters
----------
key: String
```

(continues on next page)

(continued from previous page)

```
    bibmanager config parameter to display.  Leave as None to display
    the values from all parameters.

Examples
--------
>>> import bibmanager.config_manager as cm
>>> # Show all parameters and values:
>>> cm.display()
bibmanager configuration file:
PARAMETER    VALUE
-----------  -----
style        autumn
text_editor  default
paper        letter
ads_token    None
ads_display  20
home         /home/user/.bibmanager/

>>> # Show an specific parameter:
>>> cm.display('text_editor')
text_editor: default
```

bibmanager.config_manager.**get**(*key*)

```
Get the value of a parameter in the bibmanager config file.

Parameters
----------
key: String
    The requested parameter name.

Returns
-------
value: String
    Value of the requested parameter.

Examples
--------
>>> import bibmanager.config_manager as cm
>>> cm.get('paper')
'letter'
>>> cm.get('style')
'autumn'
```

bibmanager.config_manager.**set**(*key*, *value*)

```
Set the value of a bibmanager config parameter.

Parameters
----------
key: String
    bibmanager config parameter to set.
value: String
    Value to set for input parameter.

Examples
--------
```

(continues on next page)

(continued from previous page)

```
>>> import bibmanager.config_manager as cm
>>> # Update text editor:
>>> cm.set('text_editor', 'vim')
text_editor updated to: vim.

>>> # Invalid bibmanager parameter:
>>> cm.set('styles', 'arduino')
ValueError: 'styles' is not a valid bibmanager config parameter.
The available parameters are:
  ['style', 'text_editor', 'paper', 'ads_token', 'ads_display', 'home']

>>> # Attempt to set an invalid style:
>>> cm.set('style', 'fake_style')
ValueError: 'fake_style' is not a valid style option.  Available options are:
  default, emacs, friendly, colorful, autumn, murphy, manni, monokai, perldoc,
  pastie, borland, trac, native, fruity, bw, vim, vs, tango, rrt, xcode, igor,
  paraiso-light, paraiso-dark, lovelace, algol, algol_nu, arduino,
  rainbow_dash, abap

>>> # Attempt to set an invalid command for text_editor:
>>> cm.set('text_editor', 'my_own_editor')
ValueError: 'my_own_editor' is not a valid text editor.

>>> # Beware, one can still set a valid command that doesn't edit text:
>>> cm.set('text_editor', 'less')
text_editor updated to: less.
```

bibmanager.config_manager.**update_keys**()

```
Update config in HOME with keys from ROOT, without overwriting values.
```

### 3.7.4 bibmanager.latex_manager

bibmanager.latex_manager.**no_comments**(*text*)

```
Remove comments from tex file, partially inspired by this:
https://stackoverflow.com/questions/2319019

Parameters
----------
text: String
    Content from a latex file.

Returns
-------
no_comments_text: String
    Input text with removed comments (as defined by latex format).

Examples
--------
>>> import bibmanager.latex_manager as lm
>>> text = r'''
Hello, this is dog.
% This is a comment line.
This line ends with a comment. % A comment
```

(continues on next page)

(continued from previous page)

```
However, this is a percentage \%, not a comment.
OK, bye.'''
>>> print(lm.no_comments(text))
Hello, this is dog.
This line ends with a comment.
However, this is a percentage \%, not a comment.
OK, bye.
```

bibmanager.latex_manager.**citations**(*text*)

```
Generator to find citations in a tex text.  Partially inspired
by this: https://stackoverflow.com/questions/29976397

Notes
-----
Act recursively in case there are references inside the square
brackets of the cite call.  Only failing case I can think so far
is if there are nested square brackets.

Parameters
----------
text: String
    String where to search for the latex citations.

Yields
------
citation: String
    The citation key.

Examples
--------
>>> import bibmanager.latex_manager as lm
>>> import os
>>> # Syntax matches any of these calls:
>>> tex = r'''
\citep{AuthorA}.
\citep[pre]{AuthorB}.
\citep[pre][post]{AuthorC}.
\citep [pre] [post] {AuthorD}.
\citep[{\pre},][post]{AuthorE, AuthorF}.
\citep[pre][post]{AuthorG} and \citep[pre][post]{AuthorH}.
\citep{
 AuthorI}.
\citep
[][]{AuthorJ}.
\citep[pre
 ][post] {AuthorK, AuthorL}
\citep[see also \citealp{AuthorM}][]{AuthorN}'''
>>> for citation in lm.citations(tex):
>>>     print(citation, end=" ")
AuthorA AuthorB AuthorC AuthorD AuthorE AuthorF AuthorG AuthorH AuthorI AuthorJ␣
↪AuthorK AuthorL AuthorM AuthorN

>>> # Match all of these cite calls:
>>> tex = r'''
\cite{AuthorA}, \nocite{AuthorB}, \defcitealias{AuthorC}.
\citet{AuthorD}, \citet*{AuthorE}, \Citet{AuthorF}, \Citet*{AuthorG}.
```

(continues on next page)

(continued from previous page)

```
\citep{AuthorH}, \citep*{AuthorI}, \Citep{AuthorJ}, \Citep*{AuthorK}.
\citealt{AuthorL},     \citealt*{AuthorM},
\Citealt{AuthorN},     \Citealt*{AuthorO}.
\citealp{AuthorP},     \citealp*{AuthorQ},
\Citealp{AuthorR},     \Citealp*{AuthorS}.
\citeauthor{AuthorT},  \citeauthor*{AuthorU}.
\Citeauthor{AuthorV},  \Citeauthor*{AuthorW}.
\citeyear{AuthorX},    \citeyear*{AuthorY}.
\citeyearpar{AuthorZ}, \citeyearpar*{AuthorAA}.'''
>>> for citation in lm.citations(tex):
>>>     print(citation, end=" ")
AuthorA AuthorB AuthorC AuthorD AuthorE AuthorF AuthorG AuthorH AuthorI AuthorJ␣
↪AuthorK AuthorL AuthorM AuthorN AuthorO AuthorP AuthorQ AuthorR AuthorS AuthorT␣
↪AuthorU AuthorV AuthorW AuthorX AuthorY AuthorZ AuthorAA

>>> texfile = os.path.expanduser('~')+"/.bibmanager/examples/sample.tex"
>>> with open(texfile, encoding='utf-8') as f:
>>>     tex = f.read()
>>> tex = lm.no_comments(tex)
>>> cites = [citation for citation in lm.citations(tex)]
>>> for key in np.unique(cites):
>>>     print(key)
AASteamHendrickson2018aastex62
Astropycollab2013aaAstropy
Hunter2007ieeeMatplotlib
JonesEtal2001scipy
MeurerEtal2017pjcsSYMPY
PerezGranger2007cseIPython
vanderWaltEtal2011numpy
```

bibmanager.latex_manager.**parse_subtex_files**(*tex*)

```
Recursively search for subfiles included in tex. Append their
content at the end of tex and return.

Parameters
----------
tex: String
    String to parse.

Returns
-------
tex: String
    String with appended content from any subfile.
```

bibmanager.latex_manager.**build_bib**(*texfile*, *bibfile=None*)

```
Generate a .bib file from a given tex file.

Parameters
----------
texfile: String
    Name of an input tex file.
bibfile: String
    Name of an output bib file.  If None, get bibfile name from
    bibliography call inside the tex file.
```

(continues on next page)

(continued from previous page)

```
Returns
-------
missing: List of strings
    List of the bibkeys not found in the bibmanager database.
```

bibmanager.latex_manager.**clear_latex**(*texfile*)

```
Remove by-products of previous latex compilations.

Parameters
----------
texfile: String
    Path to an existing .tex file.

Notes
-----
For an input argument texfile='filename.tex', this function deletes
the files that begin with 'filename' followed by:
  .bbl, .blg, .out, .dvi,
  .log, .aux, .lof, .lot,
  .toc, .ps,  .pdf, Notes.bib
```

bibmanager.latex_manager.**compile_latex**(*texfile*, *paper=None*)

```
Compile a .tex file into a .pdf file using latex calls.

Parameters
----------
texfile: String
    Path to an existing .tex file.
paper: String
    Paper size for output.  For example, ApJ articles use letter
    format, whereas A&A articles use A4 format.

Notes
-----
This function executes the following calls:
- compute a bibfile out of the citation calls in the .tex file.
- removes all outputs from previous compilations (see clear_latex())
- calls latex, bibtex, latex, latex to produce a .dvi file
- calls dvips to produce a .ps file, redirecting the output to
  ps2pdf to produce the final .pdf file.
```

bibmanager.latex_manager.**compile_pdflatex**(*texfile*)

```
Compile a .tex file into a .pdf file using pdflatex calls.

Parameters
----------
texfile: String
    Path to an existing .tex file.

Notes
-----
This function executes the following calls:
- compute a bibfile out of the citation calls in the .tex file.
```

(continues on next page)

(continued from previous page)

```
- removes all outputs from previous compilations (see clear_latex())
- calls pdflatex, bibtex, pdflatex, pdflatex to produce a .pdf file
```

### 3.7.5 bibmanager.ads_manager

bibmanager.ads_manager.**manager**(*query=None*)

```
A manager, it doesn't really do anything, it just delegates.
```

bibmanager.ads_manager.**search**(*query*, *start=0*, *cache_rows=200*, *sort='pubdate+desc'*)

```
Make a query from ADS.

Parameters
----------
query: String
    A query string like an entry in the new ADS interface:
    https://ui.adsabs.harvard.edu/
start: Integer
    Starting index of entry to return.
cache_rows: Integer
    Maximum number of entries to return.
sort: String
    Sorting field and direction to use.

Returns
-------
results: List of dicts
    Query outputs between indices start and start+rows.
nmatch: Integer
    Total number of entries matched by the query.

Resources
---------
A comprehensive description of the query format:
- http://adsabs.github.io/help/search/
Description of the query parameters:
- https://github.com/adsabs/adsabs-dev-api/blob/master/Search_API.ipynb

Examples
--------
>>> import bibmanager.ads_manager as am
>>> # Search entries by author (note the need for double quotes,
>>> # otherwise, the search might produce bogus results):
>>> query = 'author:"cubillos, p"'
>>> results, nmatch = am.search(query)
>>> # Search entries by first author:
>>> query = 'author:"^cubillos, p"'
>>> # Combine search by first author and year:
>>> query = 'author:"^cubillos, p" year:2017'
>>> # Restrict search to article-type entries:
>>> query = 'author:"^cubillos, p" property:article'
>>> # Restrict search to peer-reviewed articles:
>>> query = 'author:"^cubillos, p" property:refereed'
```

(continues on next page)

(continued from previous page)

```
>>> # Attempt with invalid token:
>>> results, nmatch = am.search(query)
ValueError: Invalid ADS request: Unauthorized, check you have a valid ADS token.
>>> # Attempt with invalid query ('properties' instead of 'property'):
>>> results, nmatch = am.search('author:"^cubillos, p" properties:refereed')
ValueError: Invalid ADS request:
org.apache.solr.search.SyntaxError: org.apache.solr.common.SolrException: undefined␣
↪field properties
```

bibmanager.ads_manager.**display**(*results*, *start*, *index*, *rows*, *nmatch*, *short=True*)

```
Show on the prompt a list of entries from an ADS search.

Parameters
----------
results: List of dicts
    Subset of entries returned by a query.
start: Integer
    Index assigned to first entry in results.
index: Integer
    First index to display.
rows: Integer
    Number of entries to display.
nmatch: Integer
    Total number of entries corresponding to query (not necessarily
    the number of entries in results).
short: Bool
    Format for author list. If True, truncate with 'et al' after
    the second author.

Examples
--------
>>> import bibmanager.ads_manager as am
>>> start = index = 0
>>> query = 'author:"^cubillos, p" property:refereed'
>>> results, nmatch = am.search(query, start=start)
>>> display(results, start, index, rows, nmatch)
```

bibmanager.ads_manager.**add_bibtex**(*input_bibcodes*, *input_keys*, *eprints=[]*, *dois=[]*, *update_keys=True*, *base=None*, *tags=None*)

```
Add bibtex entries from a list of ADS bibcodes, with specified keys.
New entries will replace old ones without asking if they are
duplicates.

Parameters
----------
input_bibcodes: List of strings
    A list of ADS bibcodes.
input_keys: List of strings
    BibTeX keys to assign to each bibcode.
eprints: List of strings
    List of ArXiv IDs corresponding to the input bibcodes.
dois: List of strings
    List of DOIs corresponding to the input bibcodes.
update_keys: Bool
    If True, attempt to update keys of entries that were updated
```

(continues on next page)

(continued from previous page)

```
    from arxiv to published versions.
base: List of Bib() objects
    If None, merge new entries into the bibmanager database.
    If not None, merge new entries into base.
tags: Nested list of strings
    The list of tags for each input bibcode.

Returns
-------
bibs: List of Bib() objects
    Updated list of BibTeX entries.

Examples
--------
>>> import bibmanager.ads_manager as am
>>> # A successful add call:
>>> bibcodes = ['1925PhDT.........1P']
>>> keys = ['Payne1925phdStellarAtmospheres']
>>> am.add_bibtex(bibcodes, keys)
>>> # A failing add call:
>>> bibcodes = ['1925PhDT....X....1P']
>>> am.add_bibtex(bibcodes, keys)
Error: There were no entries found for the input bibcodes.

>>> # A successful add call with multiple entries:
>>> bibcodes = ['1925PhDT.........1P', '2018MNRAS.481.5286F']
>>> keys = ['Payne1925phdStellarAtmospheres', 'FolsomEtal2018mnrasHD219134']
>>> am.add_bibtex(bibcodes, keys)
>>> # A partially failing call will still add those that succeed:
>>> bibcodes = ['1925PhDT.....X...1P', '2018MNRAS.481.5286F']
>>> am.add_bibtex(bibcodes, keys)
Warning: bibcode '1925PhDT.....X...1P' not found.
```

bibmanager.ads_manager.**update**(*update_keys=True*, *base=None*)

```
Do an ADS query by bibcode for all entries that have an ADS bibcode.
Replacing old entries with the new ones.  The main use of
this function is to update arxiv version of articles.

Parameters
----------
update_keys: Bool
    If True, attempt to update keys of entries that were updated
    from arxiv to published versions.
base: List of Bib() objects
    The bibfile entries to update.  If None, use the entries from
    the bibmanager database as base.
```

bibmanager.ads_manager.**key_update**(*key*, *bibcode*, *alternate_bibcode*)

```
Update key with year and journal of arxiv version of a key.

This function will search and update the year in a key,
and the journal if the key contains the word 'arxiv' (case
insensitive).

The function extracts the info from the old and new bibcodes.
```

(continues on next page)

(continued from previous page)

```
ADS bibcode format: http://adsabs.github.io/help/actions/bibcode

Examples
--------
>>> import bibmanager.ads_manager as am
>>> key = 'BeaulieuEtal2010arxivGJ436b'
>>> bibcode           = '2011ApJ...731...16B'
>>> alternate_bibcode = '2010arXiv1007.0324B'
>>> new_key = am.key_update(key, bibcode, alternate_bibcode)
>>> print(f'{key}\n{new_key}')
BeaulieuEtal2010arxivGJ436b
BeaulieuEtal2011apjGJ436b

>>> key = 'CubillosEtal2018arXivRetrievals'
>>> bibcode           = '2019A&A...550A.100B'
>>> alternate_bibcode = '2018arXiv123401234B'
>>> new_key = am.key_update(key, bibcode, alternate_bibcode)
>>> print(f'{key}\n{new_key}')
CubillosEtal2018arXivRetrievals
CubillosEtal2019aaRetrievals
```

### 3.7.6 bibmanager.pdf_manager

bibmanager.pdf_manager.**guess_name**(*bib*, *arxiv=False*)

```
Guess a PDF filename for a BibTex entry.  Include at least author
and year.  If entry has a bibtex, include journal info.

Parameters
----------
bib: A Bib() instance
    BibTex entry to generate a PDF filename for.
arxiv: Bool
    True if this PDF comes from ArXiv.  If so, prepend 'arxiv_' into
    the output name.

Returns
-------
guess_filename: String
    Suggested name for a PDF file of the entry.

Examples
--------
>>> import bibmanager.bib_manager as bm
>>> import bibmanager.pdf_manager as pm
>>> bibs = bm.load()
>>> # Entry without bibcode:
>>> bib = bm.Bib('''@misc{AASteam2016aastex61,
>>>     author       = {{AAS Journals Team} and {Hendrickson}, A.},
>>>     title        = {AASJournals/AASTeX60: Version 6.1},
>>>     year         = 2016,
>>> }''')
>>> print(pm.guess_name(bib))
AASJournalsTeam2016.pdf
```

(continues on next page)

(continued from previous page)

```
>>> # Entry with bibcode:
>>> bib = bm.Bib('''@ARTICLE{HuangEtal2014jqsrtCO2,
>>>   author = {{Huang ()}, Xinchuan and {Gamache}, Robert R.},
>>>    title = "{Reliable infrared line lists for 13 CO$_{2}$}",
>>>     year = "2014",
>>>   adsurl = {https://ui.adsabs.harvard.edu/abs/2014JQSRT.147..134H},
>>> }''')
>>> print(pm.guess_name(bib))
>>> Huang2014_JQSRT_147_134.pdf

>>> # Say, we are querying from ArXiv:
>>> print(pm.guess_name(bib, arxiv=True))
Huang2014_arxiv_JQSRT_147_134.pdf
```

bibmanager.pdf_manager.**open**(*pdf=None*, *key=None*, *bibcode=None*, *pdf_file=None*)

```
Open the PDF file associated to the entry matching the input key
or bibcode argument.

Parameters
----------
pdf: String
    PDF file to open.  This refers to a filename located in
    home/pdf/.  Thus, it should not contain the file path.
key: String
    Key of Bibtex entry to open it's PDF (ignored if pdf is not None).
bibcode: String
    Bibcode of Bibtex entry to open it's PDF (ignored if pdf or key
    is not None).
pdf_file: String
    Absolute path to PDF file to open.  If not None, this argument
    takes precedence over pdf, key, and bibcode.
```

bibmanager.pdf_manager.**set_pdf**(*bib*, *pdf=None*, *bin_pdf=None*, *filename=None*, *arxiv=False*, *replace=False*)

```
Update the PDF file of the given BibTex entry in database
If pdf is not None, move the file into the database pdf folder.

Parameters
----------
bibcode: String or Bib() instance
    Entry to be updated (must exist in the Bibmanager database).
    If string, the ADS bibcode of key ID of the entry.
pdf: String
    Path to an existing PDF file.
    Only one of pdf and bin_pdf must be not None.
bin_pdf: String
    PDF content in binary format (e.g., as in req.content).
    Only one of pdf and bin_pdf must be not None.
arxiv: Bool
    Flag indicating the source of the PDF.  If True, insert
    'arxiv' into a guessed name.
filename: String
    Filename to assign to the PDF file.  If None, take name from
    pdf input argument, or else from guess_name().
replace: Bool
```

(continues on next page)

(continued from previous page)

```
    Replace without asking if the entry already has a PDF assigned;
    else, ask the user.

Returns
-------
filename: String
    If bib.pdf is not None at the end of this operation,
    return the absolute path to the bib.pdf file (even if this points
    to a pre-existing file).
    Else, return None.
```

bibmanager.pdf_manager.**request_ads**(*bibcode*, *source='journal'*)

```
Request a PDF from ADS.

Parameters
----------
bibcode: String
    ADS bibcode of entry to request PDF.
source: String
    Flag to indicate from which source make the request.
    Choose between: 'journal', 'ads', or 'arxiv'.

Returns
-------
req: requests.Response instance
    The server's response to the HTTP request.
    Return None if it failed to establish a connection.

Note
----
If the request succeeded, but the response content is not a PDF,
this function modifies the value of req.status_code (in a desperate
attempt to give a meaningful answer).

Examples
--------
>>> import bibmanager.pdf_manager as pm
>>> bibcode = '2017AJ....153....3C'
>>> req = pm.request_ads(bibcode)

>>> # On successful request, you can save the PDF file as, e.g.:
>>> with open('fetched_file.pdf', 'wb') as f:
>>>     f.write(r.content)

>>> # Nature articles are not directly accessible from Journal:
>>> bibcode = '2018NatAs...2..220D'
>>> req = pm.request_ads(bibcode)
Request failed with status code 404: NOT FOUND
>>> # Get ArXiv instead:
>>> req = pm.request_ads(bibcode, source='arxiv')
```

bibmanager.pdf_manager.**fetch**(*bibcode*, *filename=None*, *replace=None*)

```
Attempt to fetch a PDF file from ADS.  If successful, then
add it into the database.  If the fetch succeeds but the bibcode is
not in the database, download file to current folder.
```

(continues on next page)

(continued from previous page)

```
Parameters
----------
bibcode: String
    ADS bibcode of entry to update.
filename: String
    Filename to assign to the PDF file.  If None, get from
    guess_name() function.
Replace: Bool
    If True, enforce replacing a PDF regardless of a pre-existing one.
    If None (default), only ask when fetched PDF comes from arxiv.

Returns
-------
filename: String
    If successful, return the full path of the file name.
    If not, return None.
```

### 3.7.7 bibmanager.utils

bibmanager.utils.**HOME**

```
os.path.expanduser('~') + '/.bibmanager/'
```

bibmanager.utils.**ROOT**

```
os.path.realpath(os.path.dirname(__file__) + '/..') + '/'
```

bibmanager.utils.**BOLD**

```
'\x1b[1m'
```

bibmanager.utils.**END**

```
'\x1b[0m'
```

bibmanager.utils.**BANNER**

```
'\n::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::\n'
```

bibmanager.utils.**ads_keywords**

```
['author:"^"', 'author:""', 'year:', 'title:""', 'abstract:""', 'property:refereed',
↪'property:article', 'abs:""', 'ack:""', 'aff:""', 'arXiv:', 'arxiv_class:""',
↪'bibcode:', 'bibgroup:""', 'bibstem:', 'body:""', 'citations()', 'copyright:',
↪'data:""', 'database:astronomy', 'database:physics', 'doctype:abstract',
↪'doctype:article', 'doctype:book', 'doctype:bookreview', 'doctype:catalog',
↪'doctype:circular', 'doctype:eprint', 'doctype:erratum', 'doctype:inproceedings',
↪'doctype:inbook', 'doctype:mastersthesis', 'doctype:misc', 'doctype:newsletter',
↪'doctype:obituary', 'doctype:phdthesis', 'doctype:pressrelease',
↪'doctype:proceedings', 'doctype:proposal', 'doctype:software', 'doctype:talk',
↪'doctype:techreport', 'doi:', 'full:""', 'grant:', 'identifier:""', 'issue:',
↪'keyword:""', 'lang:""', 'object:""', 'orcid:', 'page:', 'property:ads_openaccess',
↪'property:eprint', 'property:eprint_openaccess', 'property:inproceedings',
↪'property:non_article', 'property:notrefereed', 'property:ocrabstract',
↪'property:openaccess', 'property:pub_openaccess', 'property:software', 'references()
↪', 'reviews()', 'similar()', 'topn()', 'trending()', 'useful()', 'vizier:""',
↪'volume:']
```

(continues on next page)

(continued from previous page)

bibmanager.utils.**BM_DATABASE**()

```
The database of BibTex entries
```

bibmanager.utils.**BM_BIBFILE**()

```
Bibfile representation of the database
```

bibmanager.utils.**BM_TMP_BIB**()

```
Temporary bibfile database for editing
```

bibmanager.utils.**BM_CACHE**()

```
ADS queries cache
```

bibmanager.utils.**BM_HISTORY_SEARCH**()

```
Search history
```

bibmanager.utils.**BM_HISTORY_ADS**()

```
ADS search history
```

bibmanager.utils.**BM_HISTORY_PDF**()

```
PDF search history
```

bibmanager.utils.**BM_HISTORY_TAGS**()

```
PDF search history
```

bibmanager.utils.**BM_PDF**()

```
Folder for PDF files of the BibTex entries
```

**class** bibmanager.utils.**Author**(*last*, *first*, *von*, *jr*)

```
  Author(last, first, von, jr)

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

**class** bibmanager.utils.**Sort_author**(*last*, *first*, *von*, *jr*, *year*, *month*)

```
  Sort_author(last, first, von, jr, year, month)

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

bibmanager.utils.**ignored**(**exceptions*)

```
Context manager to ignore exceptions. Taken from here:
https://www.youtube.com/watch?v=anrOzOapJ2E
```

bibmanager.utils.**cd**(*newdir*)

```
Context manager for changing the current working directory.
Taken from here: https://stackoverflow.com/questions/431684/
```

bibmanager.utils.**ordinal**(*number*)

```
Get ordinal string representation for input number(s).

Parameters
----------
number: Integer or 1D integer ndarray
    An integer or array of integers.

Returns
-------
ord: String or List of strings
    Ordinal representation of input number(s).  Return a string if
    input is int; else, return a list of strings.

Examples
--------
>>> from bibmanager.utils import ordinal
>>> print(ordinal(1))
1st
>>> print(ordinal(2))
2nd
>>> print(ordinal(11))
11th
>>> print(ordinal(111))
111th
>>> print(ordinal(121))
121st
>>> print(ordinal(np.arange(1,6)))
['1st', '2nd', '3rd', '4th', '5th']
```

bibmanager.utils.**count**(*text*)

```
Count net number of braces in text (add 1 for each opening brace,
subtract one for each closing brace).

Parameters
----------
text: String
    A string.

Returns
-------
counts: Integer
    Net number of braces.

Examples
--------
>>> from bibmanager.utils import count
```

(continues on next page)

(continued from previous page)

```
>>> count('{Hello} world')
0
```

bibmanager.utils.**nest**(*text*)

```
Get braces nesting level for each character in text.

Parameters
----------
text: String
    String to inspect.

Returns
-------
counts: 1D integer list
    Braces nesting level for each character.

Examples
--------
>>> from bibmanager.utils import nest
>>> s = "{{P\\'erez}, F. and {Granger}, B.~E.},"
>>> n = nest(s)
>>> print(f"{s}\n{''.join([str(v) for v in n])}")
{{P\'erez}, F. and {Granger}, B.~E.},
01222222221111111111222222221111111110
```

bibmanager.utils.**cond_split**(*text*, *pattern*, *nested=None*, *nlev=-1*, *ret_nests=False*)

```
Conditional find and split strings in a text delimited by all
occurrences of pattern where the brace-nested level is nlev.

Parameters
----------
text: String
    String where to search for pattern.
pattern: String
    A regex pattern to search.
nested: 1D integer iterable
    Braces nesting level of characters in text.
nlev: Integer
    Required nested level to accept pattern match.
ret_nests: Bool
    If True, return a list with the arrays of nested level for each
    of the returned substrings.

Returns
-------
substrings: List of strings
    List of strings delimited by the accepted pattern matches.
nests: List of integer ndarrays [optional]
    nested level for substrings.

Examples
--------
>>> from bibmanager.utils import cond_split
>>> # Split an author list string delimited by ' and ' pattern:
>>> cond_split("{P\\'erez}, F. and {Granger}, B.~E.", " and ")
```

(continues on next page)

(continued from previous page)

```
["{P\\'erez}, F.", '{Granger}, B.~E.']
>>> # Protected instances (within braces) won't count:
>>> cond_split("{AAS and Astropy Teams} and {Hendrickson}, A.", " and ")
['{AAS and Astropy Teams}', '{Hendrickson}, A.']
>>> # Matches at the beginning or end do not count for split:
>>> cond_split(",Jones, Oliphant, Peterson,", ",")
['Jones', ' Oliphant', ' Peterson']
>>> # But two consecutive matches do return an empty string:
>>> cond_split("Jones,, Peterson", ",")
['Jones', '', ' Peterson']
```

bibmanager.utils.**cond_next**(*text*, *pattern*, *nested*, *nlev=1*)

```
Find next instance of pattern in text where nested is nlev.

Parameters
----------
text: String
    Text where to search for regex.
pattern: String
    Regular expression to search for.
nested: 1D integer iterable
    Braces-nesting level of characters in text.
nlev: Integer
    Requested nested level.

Returns
-------
    Index integer of pattern in text.  If not found, return the
    index of the last character in text.

Examples
--------
>>> from bibmanager.utils import nest, cond_next
>>> text = '"{{HITEMP}, the high-temperature molecular database}",'
>>> nested = nest(text)
>>> # Ignore comma within braces:
>>> cond_next(text, ",", nested, nlev=0)
53
```

bibmanager.utils.**find_closing_bracket**(*text*, *start_pos=0*, *get_open=False*)

```
Find the closing bracket that matches the nearest opening bracket in
text starting from start_pos.

Parameters
----------
text: String
    Text to search through.
start_pos: Integer
    Starting position where to start looking for the brackets.
get_opening: Bool
    If True, return a tuple with the position of both
    opening and closing brackets.

Returns
-------
```

(continues on next page)

(continued from previous page)

```
end_pos: Integer
    The absolute position to the cursor position at closing bracket.
    Returns None if there are no matching brackets.

Examples
--------
>>> import bibmanager.utils as u
>>> text = '@ARTICLE{key, author={last_name}, title={The Title}}'
>>> end_pos = u.find_closing_bracket(text)
>>> print(text[:end_pos+1])
@ARTICLE{key, author={last_name}, title={The Title}}

>>> start_pos = 14
>>> end_pos = find_closing_bracket(text, start_pos=start_pos)
>>> print(text[start_pos:end_pos+1])
author={last_name}
```

bibmanager.utils.**parse_name**(*name*, *nested=None*, *key=None*)

```
Parse first, last, von, and jr parts from a name, following these rules:
http://mirror.easyname.at/ctan/info/bibtex/tamethebeast/ttb_en.pdf
Page 23.

Parameters
----------
name: String
    A name following the BibTeX format.
nested: 1D integer ndarray
    Nested level of characters in name.
key: Sting
    The entry that contains this author name (to display in case of
    a warning).

Returns
-------
author: Author namedtuple
    Four element tuple with the parsed name.

Examples
--------
>>> from bibmanager.utils import parse_name
>>> names = ['{Hendrickson}, A.',
>>>          'Eric Jones',
>>>          '{AAS Journals Team}',
>>>          "St{\\'{e}}fan van der Walt"]
>>> for name in names:
>>>     print(f'{repr(name)}:\n{parse_name(name)}\n')
'{Hendrickson}, A.':
Author(last='{Hendrickson}', first='A.', von='', jr='')

'Eric Jones':
Author(last='Jones', first='Eric', von='', jr='')

'{AAS Journals Team}':
Author(last='{AAS Journals Team}', first='', von='', jr='')

"St{\\'{e}}fan van der Walt":
```

(continues on next page)

(continued from previous page)

```
Author(last='Walt', first="St{\\'{e}}fan", von='van der', jr='')
```

bibmanager.utils.**repr_author**(*Author*)

```
Get string representation of an Author namedtuple in the format:
von Last, jr., First.

Parameters
----------
Author: An Author() namedtuple
    An author name.

Examples
--------
>>> from bibmanager.utils import repr_author, parse_name
>>> names = ['Last', 'First Last', 'First von Last', 'von Last, First',
>>>          'von Last, sr., First']
>>> for name in names:
>>>     print(f"{name!r:22}: {repr_author(parse_name(name))}")
'Last'                 : Last
'First Last'           : Last, First
'First von Last'       : von Last, First
'von Last, First'      : von Last, First
'von Last, sr., First': von Last, sr., First
```

bibmanager.utils.**purify**(*name*, *german=False*)

```
Replace accented characters closely following these rules:
https://tex.stackexchange.com/questions/57743/
For a more complete list of special characters, see Table 2.2 of
'The Not so Short Introduction to LaTeX2e' by Oetiker et al. (2008).

Parameters
----------
name: String
    Name to be 'purified'.
german: Bool
    Replace umlaut with german style (append 'e' after).

Returns
-------
Lower-cased name without accent characters.

Examples
--------
>>> from bibmanager.utils import purify
>>> names = ["St{\\'{e}}fan",
             "{{\\v S}ime{\\v c}kov{\\'a}}",
             "{AAS Journals Team}",
             "Kov{\\'a}{\\v r}{\\'i}k",
             "Jarom{\\'i}r Kov{\\'a\\v r\\'i}k",
             "{\\.I}volgin",
             "Gon{\\c c}alez Nu{\~n}ez",
             "Knausg{\\aa}rd Sm{\\o}rrebr{\\o}d",
             'Schr{\\"o}dinger Be{\\ss}er']

>>> for name in names:
```

(continues on next page)

(continued from previous page)

```
>>>     print(f"{name!r:35}: {purify(name)}")
"St{\\'{e}}fan"                     : stefan
"{{\\v S}ime{\\v c}kov{\\'a}}"      : simeckova
'{AAS Journals Team}'               : aas journals team
"Kov{\\'a}{\\v r}{\\'i}k"           : kovarik
"Jarom{\\'i}r Kov{\\'a\\v r\\'i}k"  : jaromir kovarik
'{\\.I}volgin'                      : ivolgin
'Gon{\\c c}alez Nu{\\~n}ez'         : goncalez nunez
'Knausg{\\aa}rd Sm{\\o}rrebr{\\o}d' : knausgaard smorrebrod
'Schr{\\"o}dinger Be{\\ss}er'       : schrodinger besser
```

bibmanager.utils.**initials**(*name*)

```
Get initials from a name.

Parameters
----------
name: String
    A name.

Returns
-------
initials: String
    Name initials (lower cased).

Examples
--------
>>> from bibmanager.utils import initials
>>> names = ["", "D.", "D. W.", "G.O.", '{\\"O}. H.', "J. Y.-K.",
>>>          "Phil", "Phill Henry Scott"]
>>> for name in names:
>>>     print(f"{name!r:20}: {initials(name)!r}")
''                  : ''
'D.'                : 'd'
'D. W.'             : 'dw'
'G.O.'              : 'g'
'{\\"O}. H.'        : 'oh'
'J. Y.-K.'          : 'jyk'
'Phil'              : 'p'
'Phill Henry Scott' : 'phs'
>>> # 'G.O.' is a typo by the user, should have had a blank in between.
```

bibmanager.utils.**get_authors**(*authors*, *format='long'*)

```
Get string representation for the author list.

Parameters
----------
authors: List of Author() nametuple
format: String
    If format='ushort', display only the first author's last name,
        followed by a '+' if there are more authors.
    If format='short', display at most the first two authors followed
        by 'et al.' if corresponds.
    Else, display the full list of authors.

Returns
```

(continues on next page)

(continued from previous page)

```
-------
author_list: String
    String representation of the author list in the requested format.

Examples
--------
>>> from bibmanager.utils import get_authors, parse_name
>>> author_lists = [
>>>     [parse_name('{Hunter}, J. D.')],
>>>     [parse_name('{AAS Journals Team}'), parse_name('{Hendrickson}, A.')],
>>>     [parse_name('Eric Jones'), parse_name('Travis Oliphant'),
>>>      parse_name('Pearu Peterson')]
>>>    ]
>>> # Ultra-short format:
>>> for i,authors in enumerate(author_lists):
>>>     print(f"{i+1} author(s): {get_authors(authors, format='ushort')}")
1 author(s): Hunter
2 author(s): AAS Journals Team+
3 author(s): Jones+

>>> # Short format:
>>> for i,authors in enumerate(author_lists):
>>>     print(f"{i+1} author(s): {get_authors(authors, format='short')}")
1 author(s): {Hunter}, J. D.
2 author(s): {AAS Journals Team} and {Hendrickson}, A.
3 author(s): Jones, Eric; et al.

>>> # Long format:
>>> for i,authors in enumerate(author_lists):
>>>     print(f"{i+1} author(s): {get_authors(authors)}")
1 author(s): {Hunter}, J. D.
2 author(s): {AAS Journals Team} and {Hendrickson}, A.
3 author(s): Jones, Eric; Oliphant, Travis; and Peterson, Pearu
```

bibmanager.utils.**next_char**(*text*)

```
Get index of next non-blank character in string text.
Return zero if all characters are blanks.

Parameters
----------
text: String
    A string, duh!.

Examples
--------
>>> from bibmanager.utils import next_char
>>> texts = ["Hello", "  Hello", "  Hello ", "", "\n Hello", "  "]
>>> for text in texts:
>>>     print(f"{text!r:11}: {next_char(text)}")
'Hello'    : 0
'  Hello'  : 2
'  Hello ' : 2
''         : 0
'\n Hello' : 2
'  '       : 0
```

bibmanager.utils.**last_char**(*text*)

```
Get index of last non-blank character in string text.

Parameters
----------
text: String
    Any string.

Returns
-------
index: Integer
    Index of last non-blank character.

Examples
--------
>>> from bibmanager.utils import last_char
>>> texts = ["Hello", "  Hello", "  Hello  ", "", "\n Hello", "  "]
>>> for text in texts:
>>>     print(f"{text!r:12}: {last_char(text)}")
'Hello'     : 5
'  Hello'   : 7
'  Hello  ' : 7
''          : 0
'\n Hello'  : 7
'  '        : 0
```

bibmanager.utils.**get_fields**(*entry*)

```
Generator to parse entries of a bibliographic entry.

Parameters
----------
entry: String
    A bibliographic entry text.

Yields
------
The first yield is the entry's key.  All following yields are
three-element tuples containing a field name, field value, and
nested level of the field value.

Notes
-----
Global quotations or braces on a value are removed before yielding.

Example
-------
>>> from bibmanager.utils import get_fields
>>> entry = '''
@Article{Hunter2007ieeeMatplotlib,
  Author    = {{Hunter}, J. D.},
  Title     = {Matplotlib: A 2D graphics environment},
  Journal   = {Computing In Science \& Engineering},
  Volume    = {9},
  Number    = {3},
  Pages     = {90--95},
  publisher = {IEEE COMPUTER SOC},
```

(continues on next page)

(continued from previous page)

```
  doi       = {10.1109/MCSE.2007.55},
  year      = 2007
}'''
>>> fields = get_fields(entry)
>>> # Get the entry's key:
>>> print(next(fields))
Hunter2007ieeeMatplotlib

>>> # Now get the fields, values, and nested level:
>>> for key, value, nested in fields:
>>>   print(f"{key:9}: {value}\n{'':11}{''.join([str(v) for v in nested])}")
author   : {Hunter}, J. D.
           233333332222222
title    : Matplotlib: A 2D graphics environment
           22222222222222222222222222222222222
journal  : Computing In Science \& Engineering
           22222222222222222222222222222222222
volume   : 9
           2
number   : 3
           2
pages    : 90--95
           222222
publisher: IEEE COMPUTER SOC
           22222222222222222
doi      : 10.1109/MCSE.2007.55
           2222222222222222222
year     : 2007
           1111
```

bibmanager.utils.**req_input**(*prompt*, *options*)

```
Query for an answer to prompt message until the user provides a
valid input (i.e., answer is in options).

Parameters
----------
prompt: String
    Prompt text for input()'s argument.
options: List
    List of options to accept.  Elements in list are cast into strings.

Returns
-------
answer: String
    The user's input.

Examples
--------
>>> from bibmanager.utils import req_input
>>> req_input('Enter number between 0 and 9: ', options=np.arange(10))
>>> # Enter the number 10:
Enter number between 0 and 9: 10
>>> # Now enter the number 5:
Not a valid input.  Try again: 5
'5'
```

bibmanager.utils.**warnings_format**(*message*, *category*, *filename*, *lineno*, *file=None*, *line=None*)

```
Custom format for warnings.
```

bibmanager.utils.**tokenizer**(*attribute*, *value*, *value_token=Token.Literal.String*)

```
    Shortcut to generate formatted-text tokens for attribute-value texts.

    The attribute is set in a Token.Name.Attribute style, followed
    by a colon (Token.Punctuation style), and followed by the value
    (in value_token style).

    Parameters
    ----------
    attribute: String
        Name of the attribute.
    value: String
        The attribute's value.
    value_token: a pygments.token object
        The style for the attribute's value.

    Returns
    -------
    tokens: List of (style, text) tuples.
        Tuples that can lated be fed into a FormattedText() or
        other prompt_toolkit text formatting calls.

    Examples
    --------
    >>> import bibmanager.utils as u

    >>> tokens = u.tokenizer('Title', 'Synthesis of the Elements in Stars')
    >>> print(tokens)
    [(Token.Name.Attribute, 'Title'),
     (Token.Punctuation, ': '),
     (Token.Literal.String, 'Synthesis of the Elements in Stars'),
     (Token.Text, '
')]

    >>> # Pretty printing:
    >>> import prompt_toolkit
    >>> from prompt_toolkit.formatted_text import PygmentsTokens
    >>> from pygments.styles import get_style_by_name

    >>> style = prompt_toolkit.styles.style_from_pygments_cls(
    >>>     get_style_by_name('autumn'))
    >>> prompt_toolkit.print_formatted_text(
    >>>     PygmentsTokens(tokens), style=style)
    Title: Synthesis of the Elements in Stars
```

**class** bibmanager.utils.**DynamicKeywordCompleter**(*key_words*)

```
  Provide tab-completion for keys and words in corresponding key.

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

**class** bibmanager.utils.**DynamicKeywordSuggester**

```
  Give dynamic suggestions as in DynamicKeywordCompleter.

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

**class** bibmanager.utils.**KeyWordCompleter**(*words*, *bibs*)

```
  Simple autocompletion on a list of words.

  :param words: List of words or callable that returns a list of words.
  :param ignore_case: If True, case-insensitive completion.
  :param meta_dict: Optional dict mapping words to their meta-text. (This
      should map strings to strings or formatted text.)
  :param WORD: When True, use WORD characters.
  :param sentence: When True, don't complete by comparing the word before the
      cursor, but by comparing all the text before the cursor. In this case,
      the list of words is just a list of strings, where each string can
      contain spaces. (Can not be used together with the WORD option.)
  :param match_middle: When True, match not only the start, but also in the
                       middle of the word.
  :param pattern: Optional compiled regex for finding the word before
      the cursor to complete. When given, use this regex pattern instead of
      default one (see document._FIND_WORD_RE)

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

**class** bibmanager.utils.**AutoSuggestCompleter**

```
  Give suggestions based on the words in WordCompleter.

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

**class** bibmanager.utils.**AutoSuggestKeyCompleter**

```
  Give suggestions based on the words in WordCompleter.

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

**class** bibmanager.utils.**LastKeyCompleter**(*key_words*)

```
  Give completer options according to last key found in input.

.. code-block:: pycon

  Parameters
  ----------
  key_words: Dict
      Dictionary containing the available keys and the
      set of words corresponding to each key.
      An empty-string key denotes the default set of words to
      show when no key is found in the input text.
```

**class** bibmanager.utils.**LastKeySuggestCompleter**

```
  Give suggestions based on the keys and words in LastKeyCompleter.

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

**class** bibmanager.utils.**KeyPathCompleter**(*words*, *bibs*)

```
  Simple autocompletion on a list of words.

  :param words: List of words or callable that returns a list of words.
  :param ignore_case: If True, case-insensitive completion.
  :param meta_dict: Optional dict mapping words to their meta-text. (This
      should map strings to strings or formatted text.)
  :param WORD: When True, use WORD characters.
  :param sentence: When True, don't complete by comparing the word before the
      cursor, but by comparing all the text before the cursor. In this case,
      the list of words is just a list of strings, where each string can
      contain spaces. (Can not be used together with the WORD option.)
  :param match_middle: When True, match not only the start, but also in the
                       middle of the word.
  :param pattern: Optional compiled regex for finding the word before
      the cursor to complete. When given, use this regex pattern instead of
      default one (see document._FIND_WORD_RE)

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

**class** bibmanager.utils.**AlwaysPassValidator**(*bibs*, *toolbar_text=''*)

```
  Validator that always passes (using actually for bottom toolbar).

.. code-block:: pycon

  Initialize self.  See help(type(self)) for accurate signature.
```

## 3.8 Contributing

Feel free to contribute to this repository by submitting code pull requests, raising issues, or emailing the administrator directly.

### 3.8.1 Raising Issues

Whenever you want to raise a new issue, make sure that it has not already been mentioned in the issues list. If an issue exists, consider adding a comment if you have extra information that further describes the issue or may help to solve it.

If you are reporting a bug, make sure to be fully descriptive of the bug, including steps to reproduce the bug, error output logs, etc.

Make sure to designate appropriate tags to your issue.

An issue asking for a new functionality must include the `wish list` tag. These issues must include an explanation as to why is such feature necessary. Note that if you also provide ideas, literature references, etc. that contribute to the implementation of the requested functionality, there will be more chances of the issue being solved.

### 3.8.2 Programming Style

Everyone has his/her own programming style, I respect that. However, some people have terrible style. Following good coding practices (see PEP 8, PEP 20, and PEP 257) makes everyone happier: it will increase the chances of your code being added to the main repo, and will make me work less. I strongly recommend the following programming guidelines:

- Always keep it simple.
- Lines are strictly 80 character long, no more.
- **Never ever! use tabs (for any reason, just don't).**
- Avoid hard-coding values at all cost.
- Avoid excessively short variable names (such as `x` or `a`).
- Avoid excessively long variable names as well (just try to write a meaningful name).
- Indent with 4 spaces.
- Put whitespace around operators and after commas.
- Separate blocks of code with 1 empty line.
- Separate classes and functions with 2 empty lines.
- Separate methods with 1 empty line.
- Contraptions require meaningful comments.
- Prefer commenting an entire block before the code than using in-line comments.
- Always, always write docstrings.
- Use `is` to compare with `None`, `True`, and `False`.
- Limit try–except clauses to the bare minimum.
- If you added a new functionality, make sure to also add its repective tests.
- Make sure that your modifications pass the automated tests (travis).

Good pieces of code that do not follow these principles will still be gratefully accepted, but with a frowny face.

### 3.8.3 Pull Requests

To submit a pull request you will need to first (only once) fork the repository into your account. Edit the changes in your repository. When making a commit, always include a descriptive message of what changed. Then, click on the pull request button.

## 3.9 License

The MIT License (MIT)

Copyright (c) 2018-2022 Patricio Cubillos

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# CHAPTER 4

# Be Kind

If `bibmanager` was useful for your research, please be kind and acknowledge the effort of the developers of this project. Here's a BibTeX entry for that:

```
@MISC{Cubillos2019zndoBibmanager,
       author = {{Cubillos}, Patricio E.},
        title = "{bibmanager: A BibTeX manager for LaTeX projects}",
         year = 2019,
        month = Apr,
          eid = {10.5281/zenodo.2547042},
          doi = {10.5281/zenodo.2547042},
    publisher = {Zenodo},
       url    = {https://doi.org/10.5281/zenodo.2547042},
      adsurl = {https://ui.adsabs.harvard.edu/abs/2019zndo...2547042C},
     adsnote = {Provided by the SAO/NASA Astrophysics Data System},
}
```

# CHAPTER 5

## Featured Articles

ADS Blog: **User-Developed Tools for ADS**
*(30 Jul 2019)*
http://adsabs.github.io/blog/3rd-party-tools

AstroBetter: **Bibmanager: A BibTex Manager Designed for Astronomers**
*(17 Feb 2020)*
https://www.astrobetter.com/blog/2020/02/17/bibmanager-a-bibtex-manager-designed-for-astronomers/

Please send any feedback or inquiries to:

Patricio Cubillos (pcubillos[at]fulbrightmail.org)

# Python Module Index

## b

bibmanager, 38
bibmanager.ads_manager, 52
bibmanager.bib_manager, 38
bibmanager.config_manager, 46
bibmanager.latex_manager, 48
bibmanager.pdf_manager, 55
bibmanager.utils, 58

# Index


## A

add_bibtex() (*in module bibmanager.ads_manager*), 53
add_entries() (*in module bibmanager.bib_manager*), 44
ads_keywords (*in module bibmanager.utils*), 58
AlwaysPassValidator (*class in bibmanager.utils*), 71
Author (*class in bibmanager.utils*), 59
AutoSuggestCompleter (*class in bibmanager.utils*), 70
AutoSuggestKeyCompleter (*class in bibmanager.utils*), 70

## B

BANNER (*in module bibmanager.utils*), 58
Bib (*class in bibmanager.bib_manager*), 38
bibmanager (*module*), 38
bibmanager.ads_manager (*module*), 52
bibmanager.bib_manager (*module*), 38
bibmanager.config_manager (*module*), 46
bibmanager.latex_manager (*module*), 48
bibmanager.pdf_manager (*module*), 55
bibmanager.utils (*module*), 58
BM_BIBFILE() (*in module bibmanager.utils*), 59
BM_CACHE() (*in module bibmanager.utils*), 59
BM_DATABASE() (*in module bibmanager.utils*), 59
BM_HISTORY_ADS() (*in module bibmanager.utils*), 59
BM_HISTORY_PDF() (*in module bibmanager.utils*), 59
BM_HISTORY_SEARCH() (*in module bibmanager.utils*), 59
BM_HISTORY_TAGS() (*in module bibmanager.utils*), 59
BM_PDF() (*in module bibmanager.utils*), 59
BM_TMP_BIB() (*in module bibmanager.utils*), 59
BOLD (*in module bibmanager.utils*), 58
browse() (*in module bibmanager.bib_manager*), 46
build_bib() (*in module bibmanager.latex_manager*), 50

## C

cd() (*in module bibmanager.utils*), 60
citations() (*in module bibmanager.latex_manager*), 49
clear_latex() (*in module bibmanager.latex_manager*), 51
compile_latex() (*in module bibmanager.latex_manager*), 51
compile_pdflatex() (*in module bibmanager.latex_manager*), 51
cond_next() (*in module bibmanager.utils*), 62
cond_split() (*in module bibmanager.utils*), 61
count() (*in module bibmanager.utils*), 60

## D

display() (*in module bibmanager.ads_manager*), 53
display() (*in module bibmanager.config_manager*), 46
display_bibs() (*in module bibmanager.bib_manager*), 39
display_list() (*in module bibmanager.bib_manager*), 40
DynamicKeywordCompleter (*class in bibmanager.utils*), 69
DynamicKeywordSuggester (*class in bibmanager.utils*), 69

## E

edit() (*in module bibmanager.bib_manager*), 44
END (*in module bibmanager.utils*), 58
export() (*in module bibmanager.bib_manager*), 42

## F

fetch() (*in module bibmanager.pdf_manager*), 57
filter_field() (*in module bibmanager.bib_manager*), 40
find() (*in module bibmanager.bib_manager*), 42
find_closing_bracket() (*in module bibmanager.utils*), 62

## G

get() (*in module bibmanager.config_manager*), 47
get_authors() (*in module bibmanager.utils*), 65
get_fields() (*in module bibmanager.utils*), 67
get_version() (*in module bibmanager.bib_manager*), 42
guess_name() (*in module bibmanager.pdf_manager*), 55

## H

help() (*in module bibmanager.config_manager*), 46
HOME (*in module bibmanager.utils*), 58

## I

ignored() (*in module bibmanager.utils*), 59
init() (*in module bibmanager.bib_manager*), 43
initials() (*in module bibmanager.utils*), 65

## K

key_update() (*in module bibmanager.ads_manager*), 54
KeyPathCompleter (*class in bibmanager.utils*), 71
KeyWordCompleter (*class in bibmanager.utils*), 70

## L

last_char() (*in module bibmanager.utils*), 66
LastKeyCompleter (*class in bibmanager.utils*), 70
LastKeySuggestCompleter (*class in bibmanager.utils*), 70
load() (*in module bibmanager.bib_manager*), 41

## M

manager() (*in module bibmanager.ads_manager*), 52
merge() (*in module bibmanager.bib_manager*), 43

## N

nest() (*in module bibmanager.utils*), 61
next_char() (*in module bibmanager.utils*), 66
no_comments() (*in module bibmanager.latex_manager*), 48

## O

open() (*in module bibmanager.pdf_manager*), 56
ordinal() (*in module bibmanager.utils*), 60

## P

parse_name() (*in module bibmanager.utils*), 63
parse_subtex_files() (*in module bibmanager.latex_manager*), 50
prompt_search() (*in module bibmanager.bib_manager*), 45
prompt_search_tags() (*in module bibmanager.bib_manager*), 46
purify() (*in module bibmanager.utils*), 64

## R

read_file() (*in module bibmanager.bib_manager*), 41
remove_duplicates() (*in module bibmanager.bib_manager*), 40
repr_author() (*in module bibmanager.utils*), 64
req_input() (*in module bibmanager.utils*), 68
request_ads() (*in module bibmanager.pdf_manager*), 57
ROOT (*in module bibmanager.utils*), 58

## S

save() (*in module bibmanager.bib_manager*), 41
search() (*in module bibmanager.ads_manager*), 52
search() (*in module bibmanager.bib_manager*), 44
set() (*in module bibmanager.config_manager*), 47
set_pdf() (*in module bibmanager.pdf_manager*), 56
Sort_author (*class in bibmanager.utils*), 59

## T

tokenizer() (*in module bibmanager.utils*), 69

## U

update() (*in module bibmanager.ads_manager*), 54
update_keys() (*in module bibmanager.config_manager*), 48

## W

warnings_format() (*in module bibmanager.utils*), 68