pmparser and PMDB: resources for large-scale, open studies of the biomedical literature

Joshua L Schoenbachler [1] , **Jacob J Hughey** Corresp. [1, 2]

1 Department of Biomedical Informatics, Vanderbilt University Medical Center, Nashville, Tennessee, United States

2 Department of Biological Sciences, Vanderbilt University, Nashville, Tennessee, United States

Corresponding Author: Jacob J Hughey
Email address: jakejhughey@gmail.com

PubMed is an invaluable resource for the biomedical community. Although PubMed is freely available, the existing API is not designed for large-scale analyses and the XML structure of the underlying data is inconvenient for complex queries. We developed an R package called pmparser to convert the data in PubMed to a relational database. Our implementation of the database, called PMDB, currently contains data on over 31 million PubMed Identifiers (PMIDs) and is updated regularly. Together, pmparser and PMDB can enable large-scale, reproducible, and transparent analyses of the biomedical literature. pmparser is licensed under GPL-2 and available at https://pmparser.hugheylab.org. PMDB is available in both PostgreSQL (https://doi.org/10.5281/zenodo.4008109) and Google BigQuery (https://console.cloud.google.com/bigquery?project=pmdb-bq&d=pmdb).

1 pmparser and PMDB: resources for large-scale, open studies of
2 the biomedical literature

3 Joshua L. Schoenbachler[1] and Jacob J. Hughey[1,2]*

5 [1]Department of Biomedical Informatics, Vanderbilt University Medical Center, Nashville, TN,
6 USA
7 [2]Department of Biological Sciences, Vanderbilt University, Nashville, TN, USA

9 Corresponding author:
10 Jacob J. Hughey, jakejhughey@gmail.com

11 Abstract

12 PubMed is an invaluable resource for the biomedical community. Although PubMed is freely
13 available, the existing API is not designed for large-scale analyses and the XML structure of the
14 underlying data is inconvenient for complex queries. We developed an R package called
15 pmparser to convert the data in PubMed to a relational database. Our implementation of the
16 database, called PMDB, currently contains data on over 31 million PubMed Identifiers (PMIDs)
17 and is updated regularly. Together, pmparser and PMDB can enable large-scale, reproducible,
18 and transparent analyses of the biomedical literature. pmparser is licensed under GPL-2 and
19 available at https://pmparser.hugheylab.org. PMDB is available in both PostgreSQL
20 (https://doi.org/10.5281/zenodo.4008109) and Google BigQuery
21 (https://console.cloud.google.com/bigquery?project=pmdb-bq&d=pmdb).

22 Introduction

23 As biomedical researchers continue to advance knowledge, the literature continues to grow.
24 This growth, along with advances in technology, has created exciting opportunities at two levels.
25 First, it enables biomedical discovery using techniques such as natural language processing
26 (Kveler et al., 2018). Second, it enables meta-research into how research is organized,
27 performed, disseminated, and ultimately used (Boyack et al., 2011; Piwowar et al., 2018; Wu,
28 Wang & Evans, 2019; Abdill & Blekhman, 2019; Hutchins et al., 2019b; Fu & Hughey, 2019).
29
30 An invaluable resource for the biomedical literature is PubMed/MEDLINE, maintained by the
31 National Library of Medicine (NLM) of the U.S. National Institutes of Health. In addition to the
32 PubMed website, all data from PubMed are freely available through the E-utilities API and to
33 download in bulk. However, the API is designed for specific queries or small- to moderate-scale
34 studies, and the downloadable files store data in deeply nested XML that must first be parsed.
35 These limitations hinder large-scale analyses of PubMed data. Other large databases such as
36 Scopus and Web of Science are not freely available, which limits access to researchers at
37 particular institutions and discourages reproducibility and transparency.

38

39   To address these issues, we developed two companion resources: pmparser and PMDB.
40   pmparser is an R package allowing one to easily create and update a relational database of the
41   data in PubMed. PMDB is our implementation of the database, which is publicly available and
42   updated regularly.

Materials and Methods

44   The pmparser R package relies on the xml2, data.table, and DBI packages, which provide
45   efficient parsing of XML documents into tables, manipulating the tables, sending them to a
46   database, and manipulating the database from R. pmparser supports four database
47   management systems: PostgreSQL, MariaDB, MySQL, and SQLite. The last is recommended
48   only for small-scale testing.

49

50   NLM releases PubMed/MEDLINE data as a set of baseline XML files each December. Updates
51   to the baseline, also XML files, are released daily (called post-baseline files below). Each file
52   typically contains data on tens of thousands of PMIDs. pmparser parses the data into a set of
53   tables organized by data type and linked by PMID (Table S1).

Creating the database

55   To create the database, pmparser does the following:

56

57   1. Download the baseline XML files.
58   2. Initialize the tables in the database.
59   3. For each baseline file (in parallel) and for each data type:
60   a. Parse the XML into R data.table(s).
61   b. Append the data.table(s) to the corresponding table(s) in the database.
62   4. Add rows to the table containing the date and time at which each file was processed.
63   5. For each table in the database, keep only rows corresponding to the latest version of each
64   PMID. The vast majority of PMIDs have only one version, even when their data is
65   subsequently updated. Exceptions are articles from journals such as F1000Research.
66   6. Download the latest version of the NIH Open Citation Collection (Hutchins et al., 2019a) and
67   add it as a table in the database.

Updating the database

69   The post-baseline XML files contain data on new, updated, and deleted PMIDs. The procedure
70   here is similar to above, except the data are first parsed into a set of temporary tables before
71   being appended to the main tables.

72

73   1. Download the post-baseline XML files that have not yet been processed.
74   2. Initialize the temporary tables in the database.
75   3. For each post-baseline file (in parallel) and for each data type:
76   a. Parse the XML into R data.table(s).

77   b. Append the data.table(s) to the corresponding temporary table(s) in the database.
78 4. Add rows to the table containing the date and time at which each file was processed.
79 5. Determine the set of PMIDs whose data have been updated, along with each PMID's most
80   recent version and the most recent XML file in which it was updated.
81 6. For each main-temporary table pair:
82   a. Delete rows in the main table for PMIDs whose data have been updated.
83   b. Append rows in the temporary table that contain each PMID's most recent data (based on
84   version number and XML file). For instance, if a PMID is updated once in one XML file and
85   again in a subsequent XML file, only the rows from the second update are appended. If a
86   PMID is marked as deleted in an XML file, then that PMID's rows are deleted from the
87   main table, but no rows are appended.
88 7. Delete the temporary tables.
89 8. If a newer version of the NIH Open Citation Collection exists, download it and overwrite the
90   corresponding table in the database.

91 # Results

92 Using pmparser, we created our implementation of the database, PMDB, from the baseline XML
93 files and the latest version of the NIH Open Citation Collection. Creating the database in
94 PostgreSQL using 24 cores on a computer with 128 GiB of memory took 5.6 hours. Updating
95 the database from the 7 post-baseline XML files released between December 14, 2020 and
96 December 17, 2020 took another 20.8 minutes. Step-by-step instructions for using the
97 PostgreSQL version of PMDB are available on Zenodo
98 (https://doi.org/10.5281/zenodo.4008109). The most recent compressed dump is 20.8 GiB and
99 includes data for 31,872,359 PMIDs (Table S2).
100
101 To make PMDB more widely accessible, we also made it publicly available on Google BigQuery
102 (https://console.cloud.google.com/bigquery?project=pmdb-bq&d=pmdb), which is designed for
103 large-scale and high-speed analytics. With BigQuery, users can access PMDB without having to
104 set up their own database server, and are responsible only for the costs of their queries. We
105 plan to update both versions of PMDB monthly.
106
107 To demonstrate PMDB's utility and scale, we performed two analyses as examples of possible
108 use cases. First, we used the pub_history and author tables to quantify the increase in the
109 number of authors per published article over time (Fig. 1). Second, we used the pub_history,
110 author, and author_identifier tables to quantify the increase in the fraction of author names with
111 an ORCID identifier over time (Fig. 2). Because these analyses involve multiple data types and
112 millions of PMIDs, performing them using the E-utilities API would be difficult and time-
113 consuming. In contrast, using PMDB makes them straightforward (~10 lines of SQL) and fast
114 (~1 minute). Code to reproduce our results using the Postgres and BigQuery versions of PMDB
115 is available on Zenodo (https://doi.org/10.5281/zenodo.4004909).

Discussion

117 pmparser offers similar functionality to a Python package called medic
118 (https://github.com/fnl/medic), which is no longer maintained. The main differences between
119 medic and pmparser are that the latter parses considerably more of the MEDLINE XML (e.g.,
120 author affiliations and identifiers and everything related to investigators), smoothly handles
121 PMID versions, and makes it easier to update the database.
122
123 pmparser also offers similar functionality to a recently developed Python package called
124 pubmed_parser (Achakulvisut, Acuna & Kording, 2020), with several differences. First, the
125 packages differ somewhat in which elements of the XML they parse. For example, pmparser
126 parses the History section, which contains data on when an article was submitted, accepted,
127 etc., but pubmed_parser does not. Second, although both packages parse the abstracts (a
128 common data source for biomedical NLP), pubmed_parser can also parse the full text of articles
129 in the PubMed Central Open Access Subset. Third, pmparser adds each XML file's data to
130 tables in a relational database, whereas pubmed_parser outputs the data as a list of Python
131 dictionaries. Finally, pmparser can update an existing database, whereas pubmed_parser is
132 designed to process an entire corpus at once. Overall, which package is more suitable will
133 depend on the use case. We anticipate, however, that most researchers will prefer to bypass
134 these packages and simply use PMDB in Postgres or BigQuery.
135
136 As our examples highlight, PMDB's structure offers great flexibility to query and join multiple
137 data types. What are the most common Medical Subject Heading (MeSH) terms associated with
138 articles published by a given author? What are the datasets linked to articles published by
139 authors with a given institutional affiliation? What are the articles published in the last five years
140 in a given journal that have cited a given article? Questions such as these are now easily
141 answerable using PMDB. Furthermore, in contrast to analyses based on subscription
142 databases, analyses based on PMDB can be released without restriction for the entire
143 community to verify and build on.
144
145 With pmparser and PMDB, researchers seeking to use the wealth of data in PubMed have two
146 new options: create their own implementation of the database or use ours. Either way, once the
147 database is ready, users can query it directly or pull the data into their tool of choice for
148 statistical analysis, machine learning, etc. Together, pmparser and PMDB can enable large-
149 scale, reproducible, and transparent analyses of the biomedical literature.

Acknowledgments

151 We thank Elliot Outland for contributing to pmparser. We thank Darwin Fu for helpful comments
152 on the manuscript.

Additional Information and Declarations

Funding

This work was supported by the National Institutes of Health (R35GM124685).

Competing Interests

The authors declare that they have no competing interests.

References

Abdill RJ, Blekhman R. 2019. Meta-Research: Tracking the popularity and outcomes of all bioRxiv preprints. *eLife* 8. DOI: 10.7554/eLife.45133.

Achakulvisut T, Acuna D, Kording K. 2020. Pubmed Parser: A Python Parser for PubMed Open-Access XML Subset and MEDLINE XML Dataset XML Dataset. *Journal of Open Source Software* 5:1979.

Boyack KW, Newman D, Duhon RJ, Klavans R, Patek M, Biberstine JR, Schijvenaars B, Skupin A, Ma N, Börner K. 2011. Clustering more than two million biomedical publications: comparing the accuracies of nine text-based similarity approaches. *PloS one* 6:e18029.

Fu DY, Hughey JJ. 2019. Releasing a preprint is associated with more attention and citations for the peer-reviewed article. *eLife* 8. DOI: 10.7554/eLife.52646.

Hutchins BI, Baker KL, Davis MT, Diwersy MA, Haque E, Harriman RM, Hoppe TA, Leicht SA, Meyer P, Santangelo GM. 2019a. The NIH Open Citation Collection: A public access, broad coverage resource. *PLoS biology* 17:e3000385.

Hutchins BI, Davis MT, Meseroll RA, Santangelo GM. 2019b. Predicting translational progress in biomedical research. *PLoS biology* 17:e3000416.

Kveler K, Starosvetsky E, Ziv-Kenet A, Kalugny Y, Gorelik Y, Shalev-Malul G, Aizenbud-Reshef N, Dubovik T, Briller M, Campbell J, Rieckmann JC, Asbeh N, Rimar D, Meissner F, Wiser J, Shen-Orr SS. 2018. Immune-centric network of cytokines and cells in disease context identified by computational mining of PubMed. *Nature biotechnology* 36:651–659.

178   Piwowar H, Priem J, Larivière V, Alperin JP, Matthias L, Norlander B, Farley A, West J,

179   Haustein S. 2018. The state of OA: a large-scale analysis of the prevalence and impact of

180   Open Access articles. *PeerJ* 6:e4375.

181   Wu L, Wang D, Evans JA. 2019. Large teams develop and small teams disrupt science and

182   technology. *Nature* 566:378–382.

183

Figure 1

Using PMDB to quantify the number of authors per publication between 1920 and 2020, grouped by year.

The database query involved joining the pub_history and author tables on the pmid field. The MEDLINE XML documentation states that for PMIDs created between 1984 and 1995, at most 10 authors were entered, and for PMIDs between 1996 and 1999, at most 25 authors were entered. These limits do not affect the median or interquartile range, but may introduce inaccuracies to the mean.
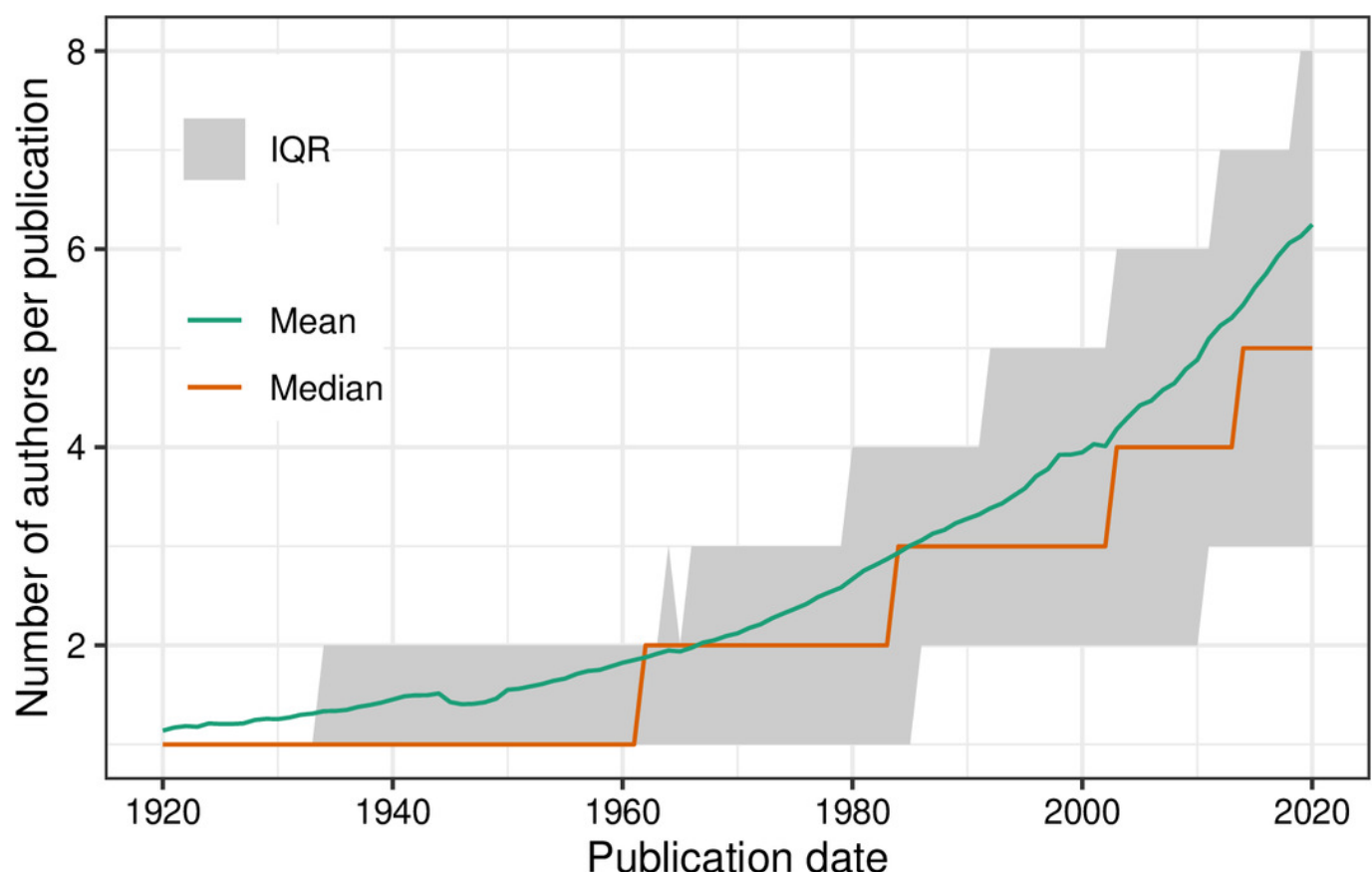
Figure 2

Using PMDB to quantify the percentage of author names with an ORCID identifier from January 2013 through August 2020, grouped by month.

The database query involved joining the pub_history, author, and author_identifier tables on the pmid and author_pos fields. ORCID identifiers became available in October 2012.