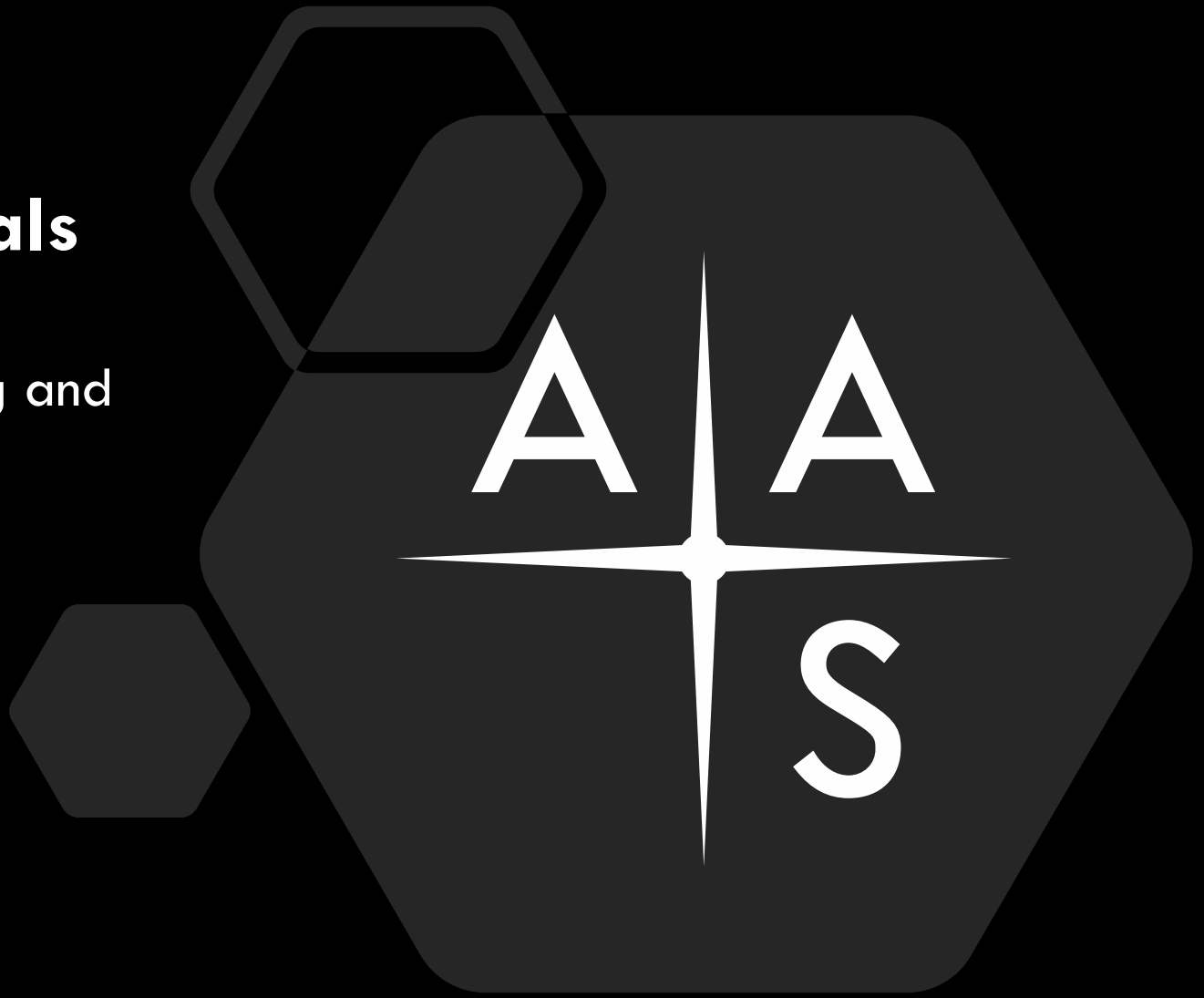
How to Write a Software Paper for the AAS Journals

(or “How I learned to stop worrying and publish Software Papers”)

Mubdi Rahman
Scientific Editor, AAS Journals
Sidrat Research



AMERICAN ASTRONOMICAL SOCIETY

What is a Software Paper

A **software paper** is a peer-reviewed paper appearing in one of the AAS Journals (ApJ, AJ, ApJL, ApJSS, or PSJ) that describes the code/software that has been developed in aid of scientific research.

AAS's Software Policy

Policy Statement on Software - AAS Journals

Policy Statement on Software

AAS Journals have adopted a policy that reflects the importance of software to the astronomical community, and the need for clear communication about such software which ensures that credit is appropriately given to its authors. The policy provides clear guidelines for citing software in all manuscripts, and supports the publication of descriptive articles about software relevant to research in astronomy and astrophysics. (Update February 2024: See also our policy on computational notebooks in the AAS Journals.)

Guidelines for software articles

AAS Journals welcome articles which describe the design and function of software of relevance to research in astronomy and astrophysics. Such articles should contain a description of the software, its novel features and its intended use. Such articles need not include research results produced using the software, although including examples of applications can be helpful. There is no minimum length requirement for software articles.

If a piece of novel software is important to published research then it is likely appropriate to describe it in such an article.

We highly recommend that authors release code described in an article under an appropriate open source license (see http://opensource.org/faq#osd or http://choosealicense.com/) and archive the published version of their code using a service such as Zenodo (https://zenodo.org/) or FigShare (http://figshare.com/) which will provide a unique digital object identifier (DOI) and ensure that the code is accessible in the long term. However, any articles which provide a clear statement on how to access the code – for example, by contacting the author – are acceptable.

Workflows for publishing code with a DOI include *Making your Code Citable* from GitHub & Zenodo.

What is a Software Paper **not**

- It is not the documentation of your software
- It is not a requirement for you to forever maintain or support the code/software you develop
- It is not even a requirement for you to publish your code (though we do think that's a good idea nonetheless)
- It is not a statement of how important your code is.
- It does not mean your code is perfect.

Why should you publish a software paper?

- You want to make sure everyone who was involved in development gets the appropriate credit (citations, papers, oh my!)
- You want to validate the science in your code
- You want to show off the cool software that you've developed
- You want other members of the community to find and use code you've built
- You want to inspire other researchers with the scientific and software methods that you are developing

What does a software paper look like?

There's no one-size-fits-all approach, but generally speaking, a software paper answers the following questions:

- What is the challenge that the software seeks to address?
- How is the software technically implemented?
- What scientific choices have been made in the software?
- How does one use the software/code?
- What's an example use case of the software/code?

The audience is other astronomers (not computer scientists)

When should I publish a software paper?

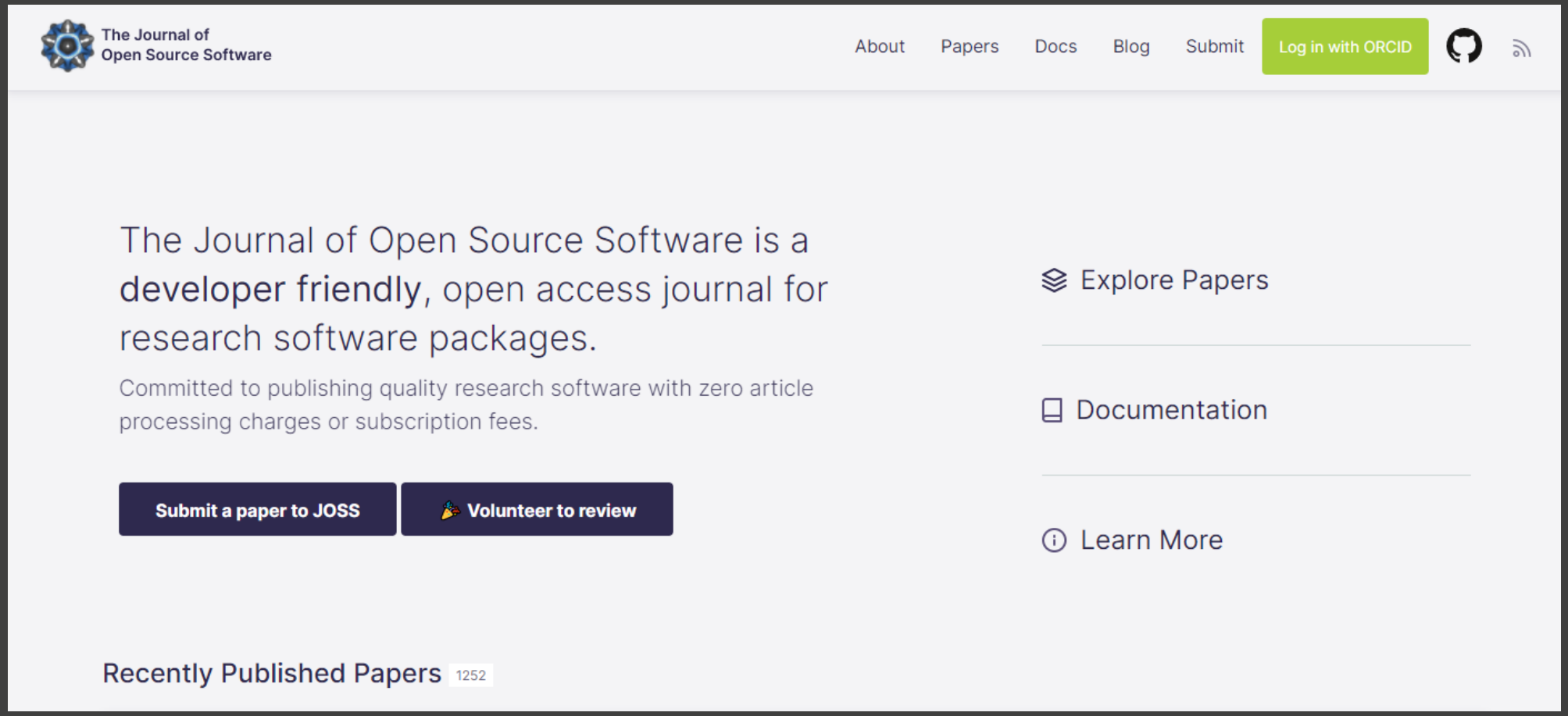
If your code/software is important to publishable research, it is ready for a software paper.

Your code does not need to be *final* or *perfect* to merit a software paper

But Mubdi, my code is evolving?

This is why we support **“Living Papers”**:

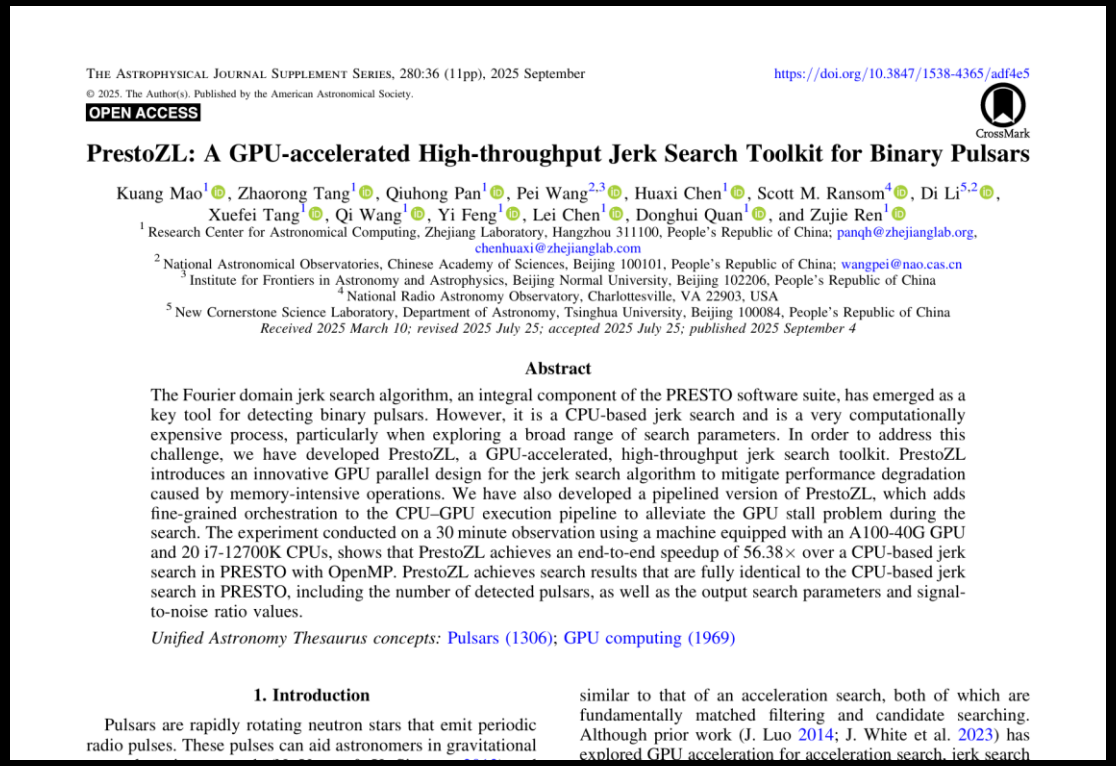
- A living paper can be updated and added on to
- You can add new co-authors to the paper
- Each additional version can be specifically cited
- Each additional version goes through the referee process



Value-Added: Journal of Open Source Software

If you're writing a software paper based on Open Source software, we'll help you coordinate with the **Journal of Open Source Software**: will validate, review, and publish your code

Some recent hits



The Astrophysical Journal Supplement Series, 280:36 (11pp), 2025 September https://doi.org/10.3847/1538-4365/adf4e5

PrestoZL: A GPU-accelerated High-throughput Jerk Search Toolkit for Binary Pulsars

Kuang Mao[1], Zhaorong Tang[1], Qiuhong Pan[1], Pei Wang[2,3], Huaxi Chen[1], Scott M. Ransom[4], Di Li[5,2], Xuefei Tang[1], Qi Wang[1], Yi Feng[1], Lei Chen[1], Donghui Quan[1], and Zujie Ren[1]
[1] Research Center for Astronomical Computing, Zhejiang Laboratory, Hangzhou 311100, People's Republic of China; panqh@zhejianglab.org, chenhuaxi@zhejianglab.com
[2] National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100101, People's Republic of China; wangpei@nao.cas.cn
[3] Institute for Frontiers in Astronomy and Astrophysics, Beijing Normal University, Beijing 102206, People's Republic of China
[4] National Radio Astronomy Observatory, Charlottesville, VA 22903, USA
[5] New Cornerstone Science Laboratory, Department of Astronomy, Tsinghua University, Beijing 100084, People's Republic of China
Received 2025 March 10; revised 2025 July 25; accepted 2025 July 25; published 2025 September 4

Abstract

The Fourier domain jerk search algorithm, an integral component of the PRESTO software suite, has emerged as a key tool for detecting binary pulsars. However, it is a CPU-based jerk search and is a very computationally expensive process, particularly when exploring a broad range of search parameters. In order to address this challenge, we have developed PrestoZL, a GPU-accelerated, high-throughput jerk search toolkit. PrestoZL introduces an innovative GPU parallel design for the jerk search algorithm to mitigate performance degradation caused by memory-intensive operations. We have also developed a pipelined version of PrestoZL, which adds fine-grained orchestration to the CPU–GPU execution pipeline to alleviate the GPU stall problem during the search. The experiment conducted on a 30 minute observation using a machine equipped with an A100-40G GPU and 20 i7-12700K CPUs, shows that PrestoZL achieves an end-to-end speedup of 56.38× over a CPU-based jerk search in PRESTO with OpenMP. PrestoZL achieves search results that are fully identical to the CPU-based jerk search in PRESTO, including the number of detected pulsars, as well as the output search parameters and signal-to-noise ratio values.

Unified Astronomy Thesaurus concepts: Pulsars (1306); GPU computing (1969)

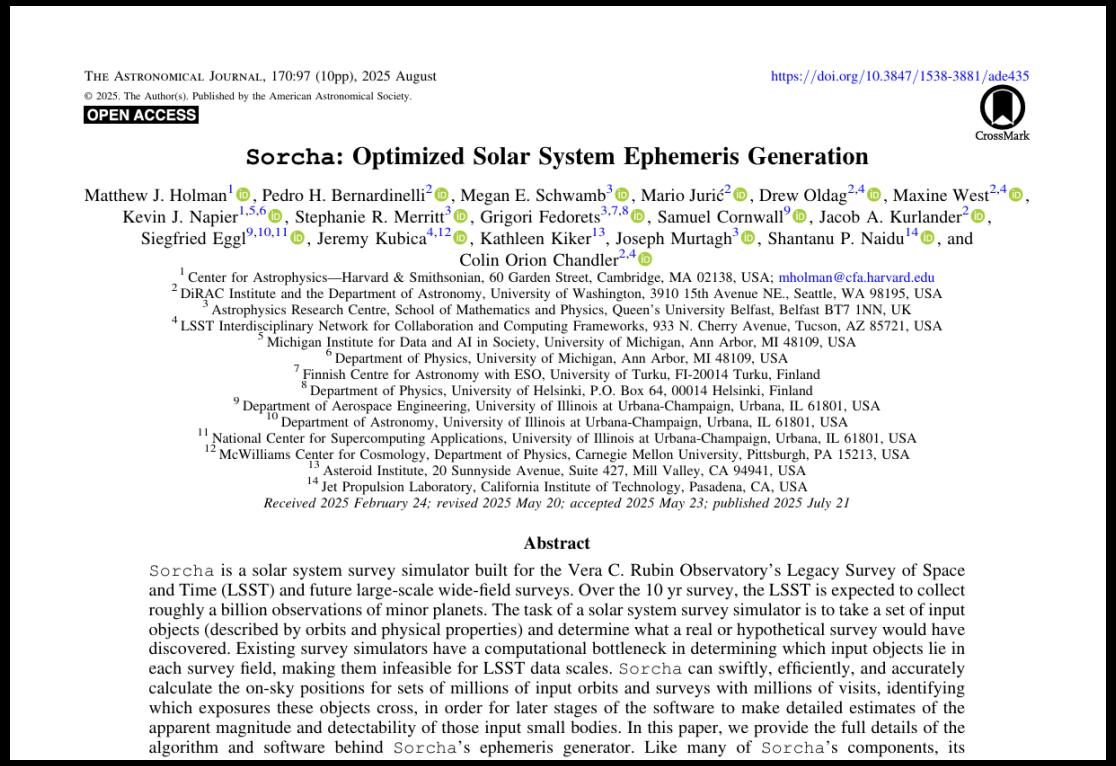
1. Introduction

Pulsars are rapidly rotating neutron stars that emit periodic radio pulses. These pulses can aid astronomers in gravitational

similar to that of an acceleration search, both of which are fundamentally matched filtering and candidate searching. Although prior work (J. Luo 2014; J. White et al. 2023) has explored GPU acceleration for acceleration search, jerk search

[PrestoZL: A GPU-accelerated High-throughput Jerk Search Toolkit for Binary Pulsars - Astrophysics Data System]

Some recent hits



The Astronomical Journal, 170:97 (10pp), 2025 August https://doi.org/10.3847/1538-3881/ade435

Sorcha: Optimized Solar System Ephemeris Generation

Matthew J. Holman[1], Pedro H. Bernardinelli[2], Megan E. Schwamb[3], Mario Jurić[2], Drew Oldag[2,4], Maxine West[2,4], Kevin J. Napier[1,5,6], Stephanie R. Merritt[3], Grigori Fedorets[3,7,8], Samuel Cornwall[9], Jacob A. Kurlander[2], Siegfried Eggl[9,10,11], Jeremy Kubica[4,12], Kathleen Kiker[13], Joseph Murtagh[3], Shantanu P. Naidu[14], and Colin Orion Chandler[2,4]

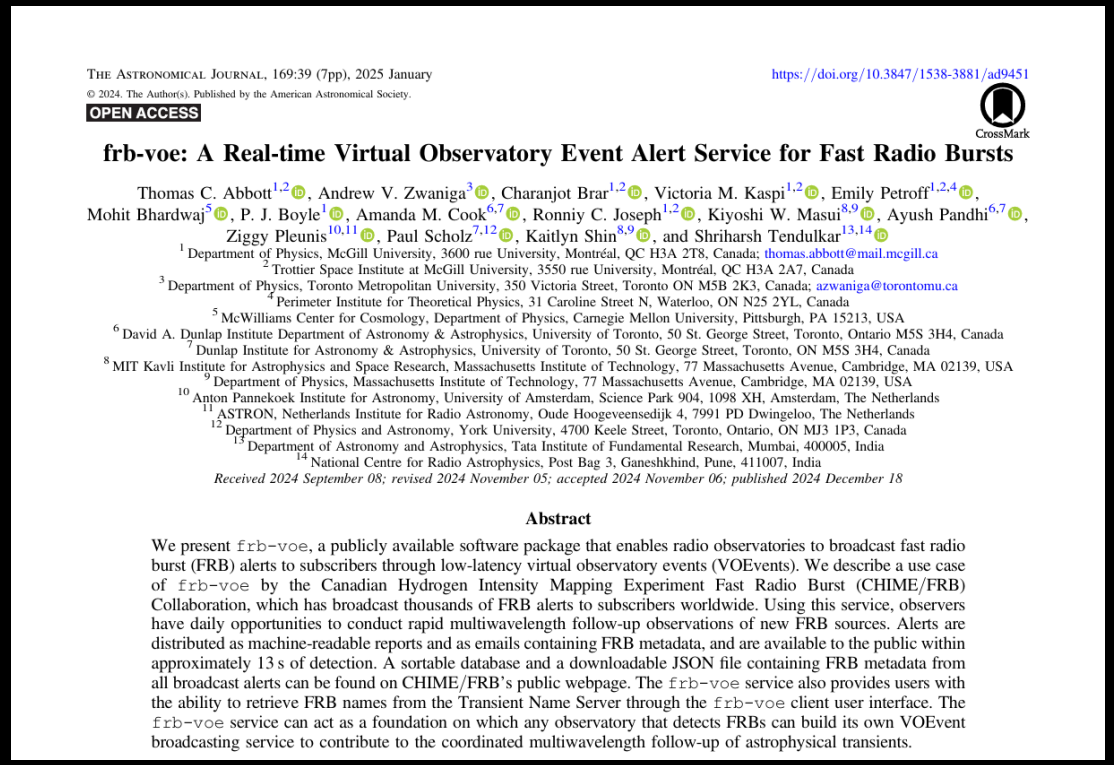
[1] Center for Astrophysics—Harvard & Smithsonian, 60 Garden Street, Cambridge, MA 02138, USA; mholman@cfa.harvard.edu
[2] DiRAC Institute and the Department of Astronomy, University of Washington, 3910 15th Avenue NE., Seattle, WA 98195, USA
[3] Astrophysics Research Centre, School of Mathematics and Physics, Queen's University Belfast, Belfast BT7 1NN, UK
[4] LSST Interdisciplinary Network for Collaboration and Computing Frameworks, 933 N. Cherry Avenue, Tucson, AZ 85721, USA
[5] Michigan Institute for Data and AI in Society, University of Michigan, Ann Arbor, MI 48109, USA
[6] Department of Physics, University of Michigan, Ann Arbor, MI 48109, USA
[7] Finnish Centre for Astronomy with ESO, University of Turku, FI-20014 Turku, Finland
[8] Department of Physics, University of Helsinki, P.O. Box 64, 00014 Helsinki, Finland
[9] Department of Aerospace Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
[10] Department of Astronomy, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
[11] National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
[12] McWilliams Center for Cosmology, Department of Physics, Carnegie Mellon University, Pittsburgh, PA 15213, USA
[13] Asteroid Institute, 20 Sunnyside Avenue, Suite 427, Mill Valley, CA 94941, USA
[14] Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA
Received 2025 February 24; revised 2025 May 20; accepted 2025 May 23; published 2025 July 21

Abstract

Sorcha is a solar system survey simulator built for the Vera C. Rubin Observatory's Legacy Survey of Space and Time (LSST) and future large-scale wide-field surveys. Over the 10 yr survey, the LSST is expected to collect roughly a billion observations of minor planets. The task of a solar system survey simulator is to take a set of input objects (described by orbits and physical properties) and determine what a real or hypothetical survey would have discovered. Existing survey simulators have a computational bottleneck in determining which input objects lie in each survey field, making them infeasible for LSST data scales. Sorcha can swiftly, efficiently, and accurately calculate the on-sky positions for sets of millions of input orbits and surveys with millions of visits, identifying which exposures these objects cross, in order for later stages of the software to make detailed estimates of the apparent magnitude and detectability of those input small bodies. In this paper, we provide the full details of the algorithm and software behind Sorcha's ephemeris generator. Like many of Sorcha's components, its

[Sorcha: Optimized Solar System Ephemeris Generation - Astrophysics Data System]

Some recent hits



The Astronomical Journal, 169:39 (7pp), 2025 January https://doi.org/10.3847/1538-3881/ad9451

frb-voe: A Real-time Virtual Observatory Event Alert Service for Fast Radio Bursts

Thomas C. Abbott[1,2], Andrew V. Zwaniga[3], Charanjot Brar[1,2], Victoria M. Kaspi[1,2], Emily Petroff[1,2,4], Mohit Bhardwaj[5], P. J. Boyle[1], Amanda M. Cook[6,7], Ronniy C. Joseph[1,2], Kiyoshi W. Masui[8,9], Ayush Pandhi[6,7], Ziggy Pleunis[10,11], Paul Scholz[7,12], Kaitlyn Shin[8,9], and Shriharsh Tendulkar[13,14]
[1] Department of Physics, McGill University, 3600 rue University, Montréal, QC H3A 2T8, Canada; thomas.abbott@mail.mcgill.ca
[2] Trottier Space Institute at McGill University, 3550 rue University, Montréal, QC H3A 2A7, Canada
[3] Department of Physics, Toronto Metropolitan University, 350 Victoria Street, Toronto ON M5B 2K3, Canada; azwaniga@torontomu.ca
[4] Perimeter Institute for Theoretical Physics, 31 Caroline Street N, Waterloo, ON N25 2YL, Canada
[5] McWilliams Center for Cosmology, Department of Physics, Carnegie Mellon University, Pittsburgh, PA 15213, USA
[6] David A. Dunlap Institute Department of Astronomy & Astrophysics, University of Toronto, 50 St. George Street, Toronto, Ontario M5S 3H4, Canada
[7] Dunlap Institute for Astronomy & Astrophysics, University of Toronto, 50 St. George Street, Toronto, ON M5S 3H4, Canada
[8] MIT Kavli Institute for Astrophysics and Space Research, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA
[9] Department of Physics, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA
[10] Anton Pannekoek Institute for Astronomy, University of Amsterdam, Science Park 904, 1098 XH, Amsterdam, The Netherlands
[11] ASTRON, Netherlands Institute for Radio Astronomy, Oude Hoogeveensedijk 4, 7991 PD Dwingeloo, The Netherlands
[12] Department of Physics and Astronomy, York University, 4700 Keele Street, Toronto, Ontario, ON MJ3 1P3, Canada
[13] Department of Astronomy and Astrophysics, Tata Institute of Fundamental Research, Mumbai, 400005, India
[14] National Centre for Radio Astrophysics, Post Bag 3, Ganeshkhind, Pune, 411007, India
Received 2024 September 08; revised 2024 November 05; accepted 2024 November 06; published 2024 December 18

Abstract

We present `frb-voe`, a publicly available software package that enables radio observatories to broadcast fast radio burst (FRB) alerts to subscribers through low-latency virtual observatory events (VOEvents). We describe a use case of `frb-voe` by the Canadian Hydrogen Intensity Mapping Experiment Fast Radio Burst (CHIME/FRB) Collaboration, which has broadcast thousands of FRB alerts to subscribers worldwide. Using this service, observers have daily opportunities to conduct rapid multiwavelength follow-up observations of new FRB sources. Alerts are distributed as machine-readable reports and as emails containing FRB metadata, and are available to the public within approximately 13 s of detection. A sortable database and a downloadable JSON file containing FRB metadata from all broadcast alerts can be found on CHIME/FRB's public webpage. The `frb-voe` service also provides users with the ability to retrieve FRB names from the Transient Name Server through the `frb-voe` client user interface. The `frb-voe` service can act as a foundation on which any observatory that detects FRBs can build its own VOEvent broadcasting service to contribute to the coordinated multiwavelength follow-up of astrophysical transients.

frb-voe: A Real-time Virtual Observatory Event Alert Service for Fast Radio Bursts - Astrophysics Data System

We're here to help!

Want to know if you should publish a software paper? Have questions about how to go about it or structure your paper?

We're happy to answer questions or set up a consultation meeting!

E-mail: mubdi.rahman@aasjournals.org

mubdi@sidratresearch.com

Bluesky: @mubdi.bsky.social