SeisIO: a fast, efficient geophysical data architecture for the Julia language

Joshua Jones[1*], Tim Clements[2], Kurama Okubo[2], and Marine A. Denolle[2]

[1]I have no idea what to list here.

[2]Department of Earth and Planetary Sciences, Harvard University, MA, USA

[*]Corresponding author: Joshua Jones (jpjones.gphys@gmail.com)

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1 Introduction

The Julia language, developed for fast, efficient numerical computing, was released August 2018 and has been in development since 2009 (Bezanson et al., 2017, 2018). It combines the ease of high-level languages like MATLAB and Python with excellent performance: while still in beta testing, it became the fourth programming language to achieve a petaflop, after Fortran, C, and C++ (Reiger et al., 2018).

2 SeisIO

The SeisIO package was created in 2016 with the intention of facilitating rapid, efficient analysis of univariate geophysical data in the Julia language. The design allows the user to read disparate types of univariate geophysical data into a single structure, including gapped and irregularly-sampled data.

SeisIO includes well-tested read support for a number of geophysical time-series formats, including Ad-Hoc (AH), Adaptable Seismic Data Format (ASDF; Krischer et al., 2016), GeoCSV time-sample pair (tspair) and sample list (slist), mini-SEED, SEG Y and the PASSCAL variant, SAC, SUDS (Ward, 1989), UNAVCO Bottles, University of Washington (UW), and WIN (NIED, 2019). Metadata support includes readers for station and event XML (e.g., Schorlemmer et al., 2012), SAC pole-zero files, and dataless SEED response files. Standard data processing operations supported by SeisIO include removal of the mean and linear trend, bandpass filtering, instrument response translation and removal (i.e., flattening to DC), resampling, cosine tapering, merging, seismogram differentiation/integration, and time synchronization. Online tools for data acquisition support the three standard FDSN protocols, IRIS timeseries requests, the IRIS TauP interface (Crofwell et al., 1999), and SeedLink.

SeisIO has been part of the Julia package ecosystem since early 2019. Automated testing in travis-ci and appveyor supports installation in Linux, Mac OS, and Windows. Code coverage estimates of 98% on CodeCov.io and Coveralls exceed the standards of enterprise-level software releases.

3 Benchmarking

To investigate computational speed and efficiency, we conduct benchmark tests using a 64-bit desktop computer equipped with an Intel DH67CL motherboard, i7-2600 (3.4 GHz) CPU, and 16 GB Kingston DDR3 RAM, running Julia v1.1.0 on 64-bit Ubuntu Linux 18.04.3 (kernel 5.0.0-29).

File read tests are described in Table 1. Tests use SeisIO v0.4.0 and BenchmarkTools.jl, with 100 samples per benchmark and one evaluation per sample. Because the Julia language uses a JIT compiler, an initial ”precompile” run precedes each test. Results appear in Table 2 and Fig. 1.

We define the memory overhead of each benchmark as obj_sz/file_sz $- 1.0$, where obj_sz is object size in memory (headers plus data) and file_sz is size on disk. In most cases overhead is negligible. File read times of GeoCSV are large because the format stores all numbers as ASCII strings.

3.1 Comparative Performance

We now compare the performance of SeisIO to two popular, well-established data architectures: ObsPy (Beyreuther et al, 2010; Megies et al., 2011) and SAC (Goldstein et al., 2003; Goldstein and Snoke, 2005). In both cases, we perform all tests in Table 1 for which file readers exist. Comparative memory use is shown in Fig. 2 and read times are shown in Fig. 3. ObsPy v1.1.1 is tested in a dedicated Python 3.7.3 environment created with conda 4.7.12, using memory-profiler 0.55.0 and timeit.py; comparisons are limited because ObsPy has no data reader for SUDS, UW, or PASSCAL SEG Y, and neither GeoCSV nor Win32 v1 parses correctly. SAC v106.a is tested with perf v5.0.21 for task-clock monitoring and time -v for memory. The time and memory associated with starting and exiting SAC (without executing any commands) was estimated with perf and removed. In all test cases, SeisIO uses less memory and reads files more quickly.

4 Tutorial

To add: info. on Tim's tutorial? Can that be revised? Should Josh add that to GitHub? It was excellent.

5 Conclusions

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Acknowledgements

The authors thank Andy Nowacki (University of Leeds, UK) for discussions on the Julia language; Douglas Neuhauser (University of California Seismological Laboratory, USA) for discussions of the SAC data format; and Chad Trabant (Incorporated Research Institutions for Seismology, USA), Robert Casey (Incorporated Research Institutions for Seismology, USA), Ellen Yu (California Institute of Technology, USA), and Aparna Bhaskaran (California Institute of Technology, USA) for assistance with FDSN and correspondence. J. Jones extends his thanks to Wendy McCausland (USGS-VDAP, USA) and Ken Creager (University of Washington, USA) for contributing test data, and R. Carniel (Universita di Udine, Italy) for early testing. mini-SEED handling was originally based on rdmseed.m for MATLAB by Francois Beauducel (Institut de Physique du Globe de Paris, France); SAC routines were originally based on SacIO.jl by Ben Postlethwaite. This research was supported by a grant from the Packard Foundation.

Data Availability

Data used in benchmark tests can be found in the SeisIO GitHub repository with exceptions given in Table 1. Benchmarking scripts are available on the SeisIO GitHub page.

References

M. Beyreuther, R. Barsch, L. Krischer, T. Megies, Y. Behr and J. Wassermann (2010), ObsPy: A Python Toolbox for Seismology, SRL, 81(3), 530-533. DOI: 10.1785/gssrl.81.3.530

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. SIAM review, 59(1), 65-98.

Bezanson, J., Chen, J., Chung, B., Karpinski, S., Shah, V. B., Vitek, J., & Zoubritzky, L. (2018). Julia: dynamism and performance reconciled by design. Proceedings of the ACM on Programming Languages, 2(OOPSLA), 120.

Crotwell, H. P., T. J. Owens, and J. Ritsema (1999). The TauP Toolkit: Flexible seismic travel-time and ray-path utilities, Seismological Research Letters 70, 154?160.

Goldstein, P., A. Snoke, (2005), ?SAC Availability for the IRIS Community?, Incorporated Institutions for Seismology Data Management Center Electronic Newsletter.

Goldstein, P., D. Dodge, M. Firpo, Lee Minner (2003) ?SAC2000: Signal processing and analysis tools for seismologists and engineers, Invited contribution to ?The IASPEI International Handbook of Earthquake and Engineering Seismology?, Edited by WHK Lee, H. Kanamori, P.C. Jennings, and C. Kisslinger, Academic Press, London.

Jones, J.P., & Malone, S. D. (2005). Mount Hood earthquake activity: Volcanic or tectonic origins?. Bulletin of the Seismological Society of America, 95(3), 818-832.

Lion Krischer, James Smith, Wenjie Lei, Matthieu Lefebvre, Youyi Ruan, Elliott Sales de Andrade, Norbert Podhorszki, Ebru Bozdağ, Jeroen Tromp, An Adaptable Seismic Data Format, Geophysical Journal International, Volume 207, Issue 2, November, 2016, Pages 1003?1011, https://doi.org/10.1093/gji/ggw319.

T. Megies, M. Beyreuther, R. Barsch, L. Krischer, J. Wassermann (2011), ObsPy ? What can it do for data centers and observatories?, Annals Of Geophysics, 54(1), 47-58, DOI: 10.4401/ag-4838.

National Research Institute for Earth Science and Disaster Resilience (2019), NIED Hi-net, National Research Institute for Earth Science and Disaster Resilience, doi:10.17598/NIED.0003.

Regier, J., Fischer, K., Pamnany, K., Noack, A., Revels, J., Lam, M., Howard, S., Giordano, R., Schlegel, D., McAuliffe, J. and Thomas, R., 2019. Cataloging the visible universe through Bayesian inference in Julia at petascale. Journal of Parallel and Distributed Computing, 127, pp.89-104.

Schorlemmer, D., Euchner, F., Kstli, P., & Saul, J. (2011). QuakeML: status of the XML-based seismological data exchange format. Annals of Geophysics, 54(1), 59-65.

Ward, Peter L. (1989). SUDS; seismic unified data system, USGS Open-File Report 89-188, doi:10.3133/ofr89188.

Table 1: Benchmark tests. Columns: Test Name is how the test is referenced in this manuscript; Filename is the name or search pattern in SeisIO/test/SampleFiles/; Sz is file size in MB; Src is data source, given below (for request parameters, please contact the corresponding author).

1. contributed by Prof. K. Creager, University of Washington, USA.
2. contributed by M. Denolle.
3. retrieved with IRIS FDSN dataselect.
4. from IRIS _STHELENS-1980 virtual network and data set; available from IRIS, USA.
5. redistribution restricted; to request this file please contact Dr. W. McCausland, USGS-VDAP.
6. extracted from Pacific Northwest Seismic Network archives; data from Jones and Malone (2005).
7. data from HiNet (NIED, 2019); redistribution prohibited.

Test Name	**Filename**	**Format**	**Sz [MB]**	**Src**
AH	20050904.PA01.E.sac.ah	AH	0.14	1
ASDF	2019_07_07_00_00_00.h5	ASDF	21.96	2
GeoCSV-tspair	FDSNWS.IRIS.geocsv	GeoCSV tspair	3.31	3
GeoCSV-slist	geocsv_slist.csv	GeoCSV slist	8.25	3
MSEED-1	one_day.mseed	mini-SEED	19.09	2
MSEED-2	SHW.UW.mseed	mini-SEED	1.79	4
PASSCAL	test_PASSCAL.segy	PASSCAL SEG Y	32.96	3
SAC	one_day.sac	SAC	32.96	2
SUDS	10081701.WVP	SUDS	1.26	5
UW	99011116541W	UW	23.15	6
WIN	2014092709*.cnt	WIN	4.49	7

Table 2: SeisIO v0.4.0 benchmarks. Columns: Sz is size of object in memory; Mem is total memory use; Ovh is memory overhead (defined in text); T is read time in ms; for Notes, 1 = test uses SeisIO read_hdf5; 2 = test uses SeisIO read_data with KW full=true; 3 = test uses SeisIO read_data with KW nx_new=36000, nx_add=1400000. Other tests use SeisIO read_data with default parameters.

Test Name	**Sz [MB]**	**Mem [MB]**	**Ovh [%]**	**T [ms]**	**Notes**
AH	0.33	0.33	1.16	0.63	
ASDF	26.37	26.49	0.45	93.92	1
GeoCSV-tspair	0.39	0.44	12.30	209.84	
GeoCSV-slist	6.80	6.87	1.07	123.08	
MSEED-1	32.96	32.96	0.01	71.06	
MSEED-2	5.35	6.19	15.75	9.53	3
PASSCAL	32.96	32.99	0.08	22.42	2
SAC	32.96	32.97	0.04	12.88	2
SUDS	2.53	2.59	2.43	1.36	
UW	37.66	40.29	6.98	26.9	
WIN	10.99	11.25	2.33	24.67	

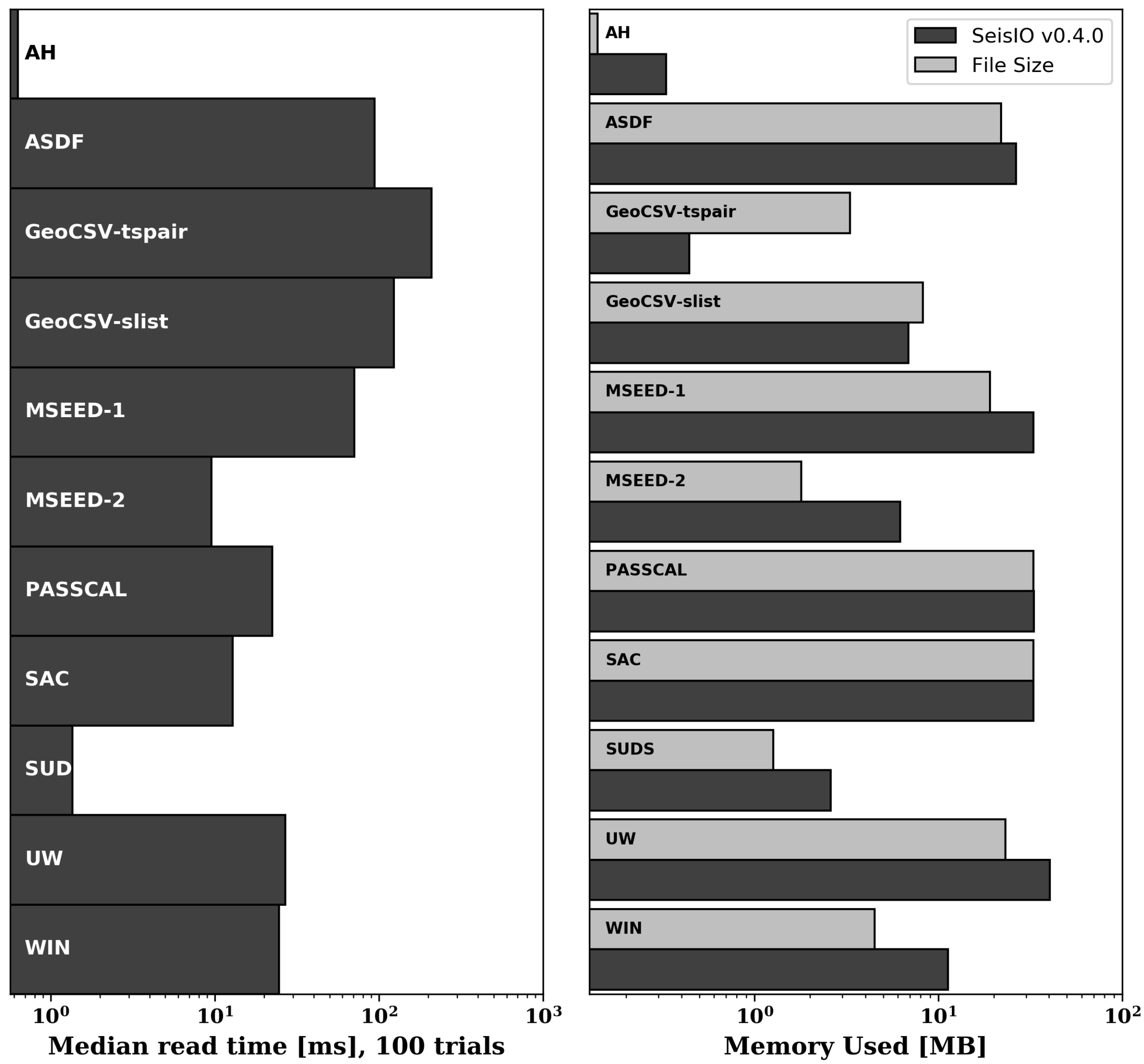

Figure 1: Benchmarks tests (Table 1) in Julia v1.1.0 with SeisIO v0.4.0. Left: file read times. Right: peak memory use in SeisIO and file size on disk.

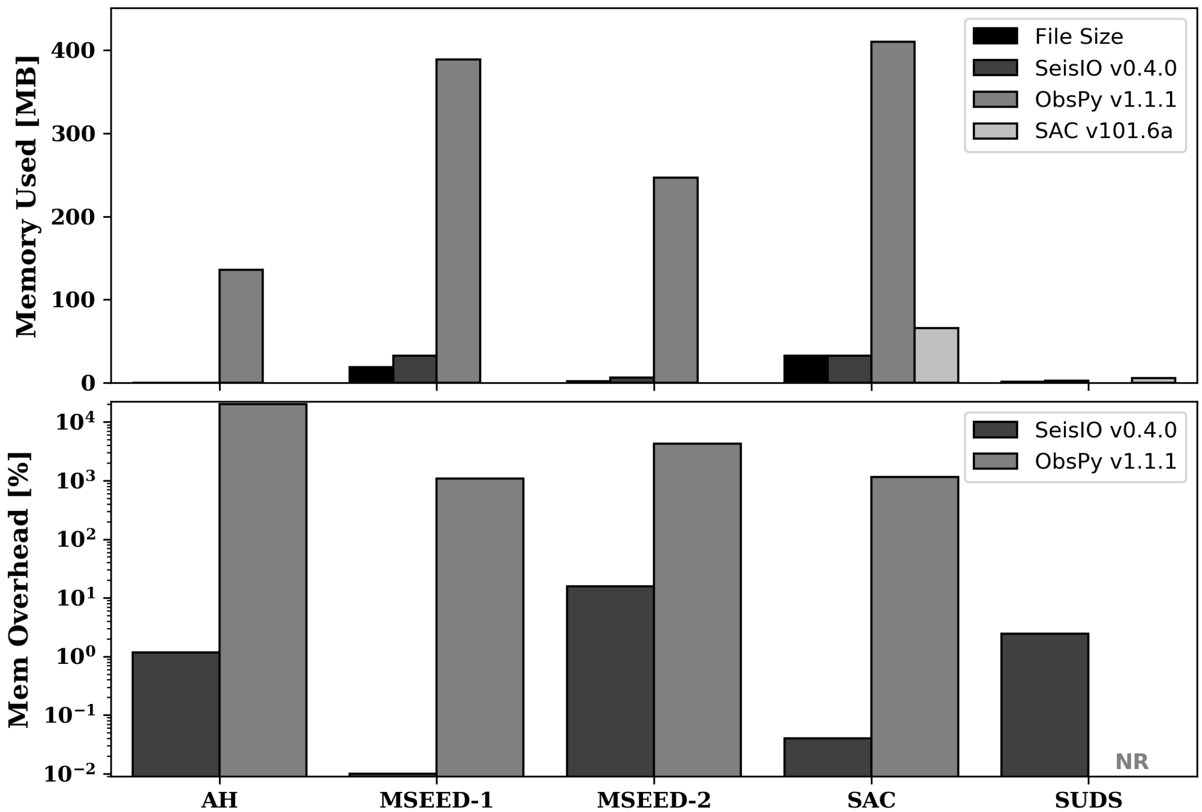

Figure 2: Memory use and overhead for all benchmarks in Table 1 that were testable in at least two of ObsPy, SAC, and SeisIO. (top) Memory usage and file sizes on disk. (bottom) Memory overhead. The *y*-axis is logarithmic. A missing bar with text label NR" indicates no reader.

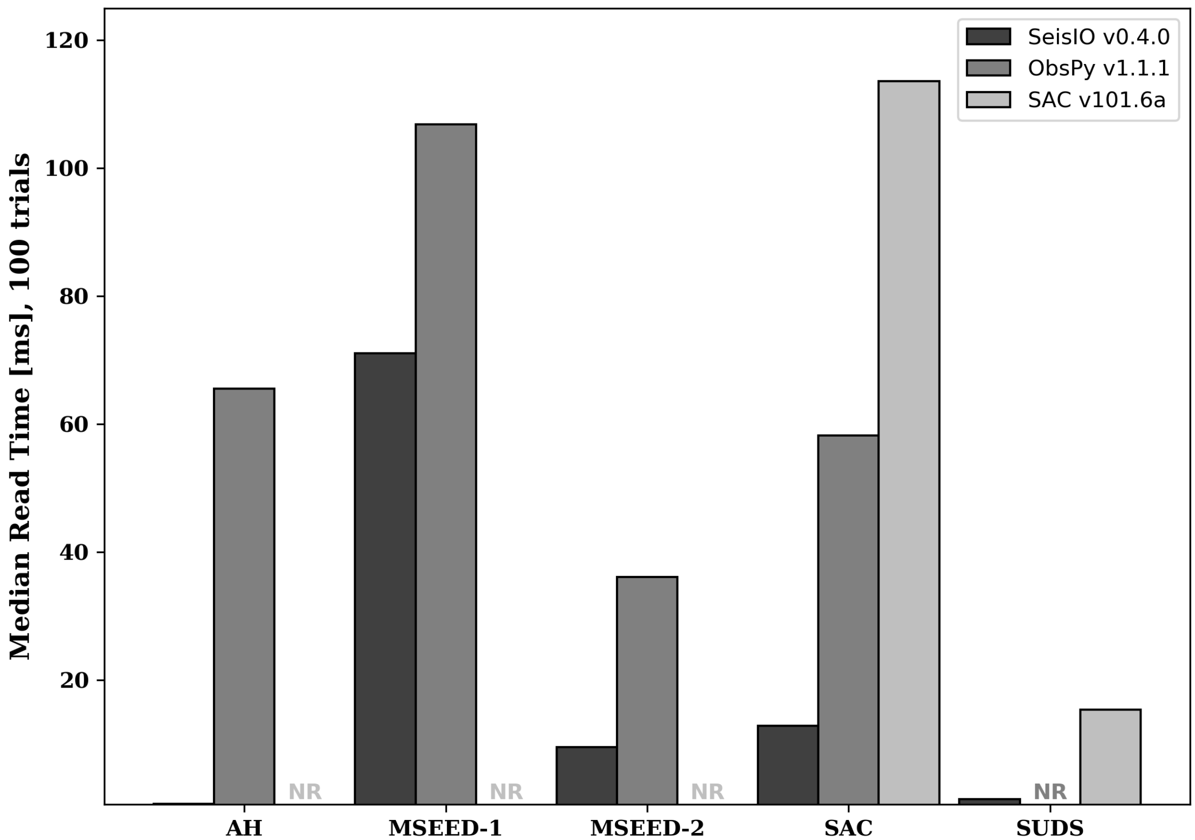

Figure 3: Read times in milliseconds for all benchmarks in Table 1 that were testable in at least two of ObsPy, SAC, and SeisIO. A missing bar with text label NR" indicates "no reader".