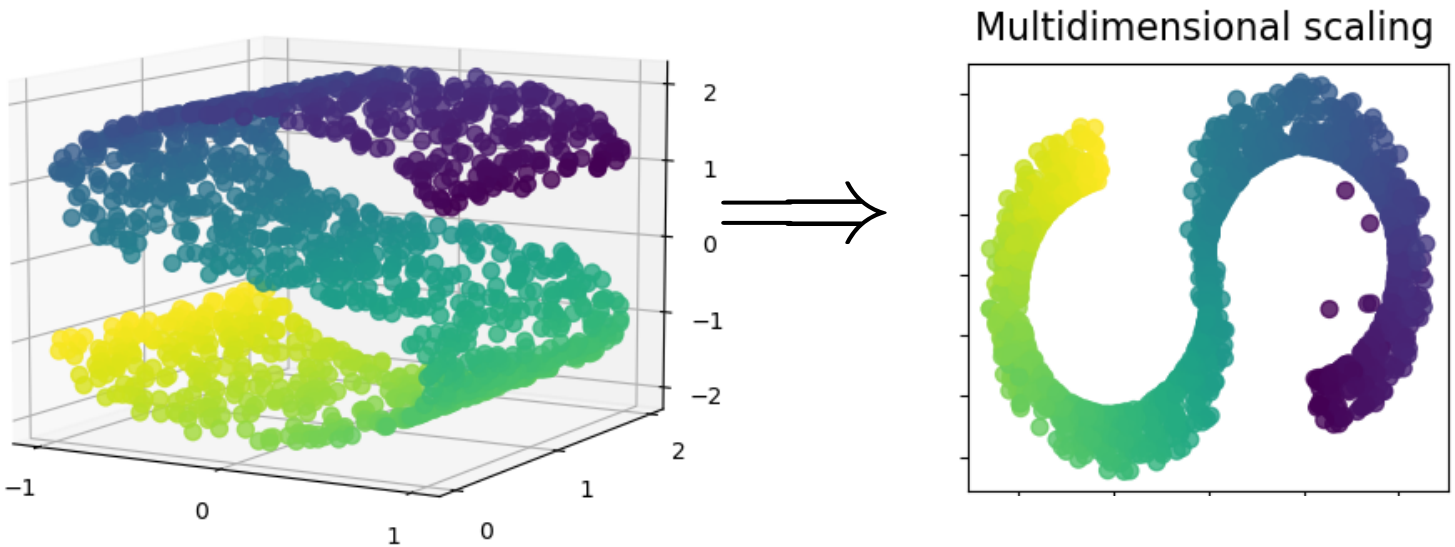
Projecting to low dimensions while preserving distances

Let D be the matrix of measured pairwise distances.
Multidimensional scaling (MDS) finds $x_i \in \mathbb{R}^k$ to minimize

$$\text{STRESS}(x) = \sum_{i>j} (D_{ij} - \|x_i - x_j\|)^2$$

where k is small.



[Pedregosa et al., 2011]

Bayesian MDS

- Want to specify priors for x_i
- First published by Oh and Raftery [Oh and Raftery, 2001]
- Let D be the matrix of measured distances between points
- Assumes

$$D_{ij} \sim N(\|x_i - x_j\|, \sigma^2) I(D_{ij} > 0)$$

- where $x_i \in \mathbb{R}^k$
- Estimate parameters with Markov Chain Monte Carlo (MCMC) methods
- Extremely computationally expensive!

MassiveMDS: an OpenCL implementation of likelihood and gradient functions for MDS

JOURNAL OF COMPUTATIONAL AND GRAPHICAL STATISTICS
2021, VOL. 30, NO. 1, 11–24
https://doi.org/10.1080/10618600.2020.1754226

Massive Parallelization Boosts Big Bayesian Multidimensional Scaling

Andrew J. Holbrook[a], Philippe Lemey[b], Guy Baele[b], Simon Dellicour[b], Dirk Brockmann[c], Andrew Rambaut[d,e], and Marc A. Suchard[a,f,g]

[a]Department of Biostatistics, University of California, Los Angeles, Los Angeles, CA; [b]Department of Microbiology, Immunology and Transplantation, Rega Institute, KU Leuven, Leuven, Belgium; [c]Institute for Theoretical Biology, Humboldt University Berlin, Berlin, Germany; [d]Institute of Evolutionary Biology, University of Edinburgh, Edinburgh, UK; [e]Fogarty International Center, National Institutes of Health, Bethesda, MD; [f]Department of Human Genetics, University of California, Los Angeles, Los Angeles, CA; [g]Department of Biomathematics, University of California, Los Angeles, Los Angeles, CA

ABSTRACT

Big Bayes is the computationally intensive co-application of big data and large, expressive Bayesian models for the analysis of complex phenomena in scientific inference and statistical learning. Standing as an example, Bayesian multidimensional scaling (MDS) can help scientists learn viral trajectories through space-time, but its computational burden prevents its wider use. Crucial MDS model calculations scale quadratically in the number of observations. We partially mitigate this limitation through massive parallelization using multi-core central processing units, instruction-level vectorization and graphics processing units (GPUs). Fitting the MDS model using Hamiltonian Monte Carlo, GPUs can deliver more than 100-fold speedups over serial calculations and thus extend Bayesian MDS to a big data setting. To illustrate, we employ Bayesian MDS to infer the rate at which different seasonal influenza virus subtypes use worldwide air traffic to spread around the globe. We examine 5392 viral sequences and their associated 14 million pairwise distances arising from the number of commercial airline seats per year between viral sampling locations. To adjust for shared evolutionary history of the viruses, we implement a phylogenetic extension to the MDS model and learn that subtype H3N2 spreads most effectively, consistent with its epidemic success relative to other seasonal influenza subtypes. Finally, we provide MassiveMDS, an open-source, stand-alone C++ library and rudimentary R package, and discuss program design and high-level implementation with an emphasis on important aspects of computing architecture that become relevant at scale.

ARTICLE HISTORY
Received May 2019
Revised December 2019

KEYWORDS
Bayesian phylogeography; Graphics processing unit; Hamiltonian Monte Carlo; Massive parallelization; Single-instruction, multiple-data

Some questions I still have

- The code contains a lot of fitting "precision" of the MDS likelihood, but nothing in the paper about this. Is this referring to the float precision used to compute the likelihood?
- There are also mentions of learning traits in the code but not the paper, which I think has to do with the integration of this package to BEAST


Oh, M.-S. and Raftery, A. E. (2001).
Bayesian multidimensional scaling and choice of dimension.
Journal of the American Statistical Association,
96(455):1031–1044.


Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011).
Scikit-learn: Machine learning in Python.
Journal of Machine Learning Research, 12:2825–2830.