BIOINFORMATICS APPLICATIONS NOTE Vol. 29 no. 3 2013, pages 407–408
doi:10.1093/bioinformatics/bts707

Data and text mining

Advance Access publication December 14, 2012

minerva and minepy: a C engine for the MINE suite and its R, Python and MATLAB wrappers

Davide Albanese[1,†], Michele Filosi[1,2,3,†], Roberto Visintainer[1,4,†], Samantha Riccadonna[1], Giuseppe Jurman[1] and Cesare Furlanello[1,*]

[1]Fondazione Bruno Kessler, via Sommarive 18, I-38123 Povo (Trento), Italy, [2]CIBIO, University of Trento, via delle Regole 101, I-38123 Mattarello (Trento), Italy, [3]The Wistar Institute, 3601 Spruce Street, Philadelphia, PA 19104, USA and [4]DISI, University of Trento, Via Sommarive 5, I-38123 Povo (Trento), Italy

Associate Editor: Jonathan Wren

ABSTRACT

Summary: We introduce a novel implementation in ANSI C of the MINE family of algorithms for computing maximal information-based measures of dependence between two variables in large datasets, with the aim of a low memory footprint and ease of integration within bioinformatics pipelines. We provide the libraries minerva (with the R interface) and minepy for Python, MATLAB, Octave and C++. The C solution reduces the large memory requirement of the original Java implementation, has good upscaling properties and offers a native parallelization for the R interface. Low memory requirements are demonstrated on the MINE benchmarks as well as on large ($n = 1340$) microarray and Illumina GAII RNA-seq transcriptomics datasets.

Availability and implementation: Source code and binaries are freely available for download under GPL3 licence at http://minepy.sourceforge.net for minepy and through the CRAN repository http://cran.r-project.org for the R package minerva. All software is multiplatform (MS Windows, Linux and OSX).

Contact: furlan@fbk.eu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on August 29, 2012; revised on December 6, 2012; accepted on December 9, 2012

1 INTRODUCTION

The Maximal Information-based Non-parametric Exploration (MINE) family of statistics, including the Maximal Information Coefficient (MIC) measure, was recently introduced in (Reshef *et al*., 2011), aimed at fast exploration of two-variable relationships in many-dimensional datasets. MINE consists of the algorithms for computing four measures of dependence—MIC, Maximum Asymmetry Score (MAS), Maximum Edge Value (MEV), Minimum Cell Number (MCN)—between two variables, having the generality and equitability property. Generality is the ability of capturing variable relationships of different nature, while equitability is the property of penalizing similar levels of noise in the same way, regardless of the nature of the relation between the variables. The MINE suite received immediate appraisal as a real breakthrough in the data mining of complex biological data (Nat. Biotech., 2012; Speed, 2011) as well as criticisms (See comments and referenced experiments by Simon and Tibshirani and by Gorfin *et al*. at http://comments.sciencemag.org/content/10.1126/science.1205438.) Many groups worldwide have already proposed its use for explorative data analysis in computational biology, from networks dynamics to virus ranking (Anderson *et al*., 2012; Das *et al*., 2012; Faust and Raes, 2012; Karpinets *et al*., 2012; Weiss *et al*., 2012). Together with the algorithm description, the MINE authors provided a Java implementation (MINE.jar), two wrappers (R and Python) and four reference datasets (Reshef *et al*., 2011). However, applicability and scalability of MINE.jar on large datasets is currently limited owing to memory requirements and lack of programming interfaces. Further, a native parallelization, currently unavailable, would be of significant benefit. These issues are hurdles for a systematic application of MINE algorithms to high-throughput omics data—for example, as a substitute of Pearson correlation in network studies. Inspired by these considerations, we propose an ANSI C implementation of the MINE algorithms, and the interfaces for R (minerva), and for C++, Python and MATLAB/Octave (minepy).

2 THE MINE C ENGINE AND ITS WRAPPERS

The novel engine (libmine) is written in ANSI C as a clean-room implementation of the algorithms originally described in (Reshef *et al*., 2011), as the Java source code is not distributed. Libmine provides three structures describing the data, the parameter configuration and the maximum normalized mutual information scores. The core function `mine_compute_score()` takes a dataset structure and a configuration one as input, returning a score structure as output, from which four functions compute the MINE statistics. The minepy Python module works with Python ≥ 2.6, with NumPy $\geq 1.3.0$ as the sole requirement: the interface consists of the `class minepy.MINE` whose methods match the C functions. The R package minerva is built as an R wrapper (R ≥ 2.14) to the C engine: the main function mine takes the dataset and the parameter configuration as inputs and returns the four MINE statistics. Minerva allows native parallelization: based on the R package `parallel`, the number of cores can be passed as parameter to mine, whenever multi-core hardware is available. The curated version of the CDC15 Spellman yeast dataset

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first three authors should be regarded as joint First Authors.

(Spellman *et al*., 1998) used in (Reshef *et al*., 2011) is included as example. Documentation (online and PDF) for minepy is available at the minepy website, also as online help in R for minerva.

2.1 Performance comparison

The suite was tested for consistency with MINE.jar v1.0.1 on the Spellman and microbiome datasets from http://www.exploredata.net.

For the Spellman dataset (4381 transcripts and 23 timepoints), MIC values were computed for all features pairs with MINE.jar and minepy (both with $\alpha = 0.67$). Identical results were found up to five significance digits: see Supplementary Figure S3 and details in Supplementary Material. For the microbiome data, for the 77 top ranked association pairs listed in Supplementary Table S13 of (Reshef *et al*., 2011), we obtained 44 identical results and a difference less than 0.01 for other 29 values (details in Supplementary Material).

To compare performance for RAM and CPU usage between MINE.jar and the suite, MINE statistics were computed on all features pairs of the Spellman dataset, for increasing feature set sizes (details in Supplementary Material, Sec 2.2, 2 A). Minerva and minepy completed all tasks with limited RAM requirements: about 19 MB were needed for all 4382 variables (600 kB dataset size) by minepy, and 2 MB by the C++ interface. We were unable to run MINE.jar with >2000 variables, for which Java used 7.5 GB and minepy 16 MB, respectively (Fig. 1 and Supplementary Table S2). Minepy computing times are about twice those of the Java solution (Supplementary Table S3), but the speedup is close to 70 for minerva on 100 cores via MPI on a Linux cluster, with the default $\alpha = 0.6$ (Supplementary Fig. S6). Finer grids ($\alpha = 0.7$) require much higher computing time as sample size increases (Supplementary Fig. S7).

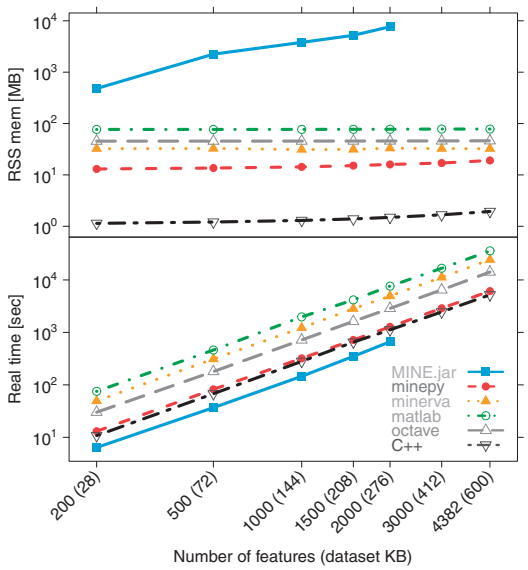

Fig. 1. Comparison of MINE.jar v1.0.1 and the novel interfaces (minerva, minepy, Matlab/Octave and C++) computing the four MINE statistics, the Pearson correlation coefficient and the non-linearity index for all pairs of features of the Spellman dataset. For increasing number of features: (top) Resident set size (RSS), i.e. the non-swapped physical memory (in megabytes); (bottom) elapsed real (wall clock) time used by the process, in seconds. MINE.jar can complete the task only up to 2000 features, even having reserved 8GB RAM to Java. In parentheses, the dataset ASCII file size in kilobytes (KB)

Table 1. Performance of minerva and minepy (one versus all) on microarray and RNA-seq datasets listed by GEO accession number

GEO Acc. no.			CPU		RAM	
	n	*p*	R	P	R	P
GSE25219[a]	1340	17 565	41 880	34 359	533 008	1 509 692
GSE34914[b]	16	20 422	42	3	35 716	31 956

n: number of samples. *p*: number of features. CPU: elapsed time used by the process (in seconds). RAM: resident set size (in kilobytes), for minerva (R) and minepy (P).
[a]Kang *et al*. (2011).
[b]Kalari *et al*. (2012).

We additionally tested the suite on two recent high-throughput transcriptomics datasets, of Affymetrix HumanExon 1.0ST human brain tissues and Illumina Genome Analyzer II-sequenced human non–small cell lung cancer (Table 1). Details on datasets and experiments are reported in Supplementary Material.

Funding: EU FP7 Project HiPerDART, The Autonomous Province of Trento Major Project ENVIROCHANGE, PHCT.

Conflict of Interest: none declared.

REFERENCES

Anderson,T. *et al*. (2012) Ranking viruses: measures of positional importance within networks define core viruses for rational polyvalent vaccine development. *Bioinformatics*, **28**, 1624–1632.

Das,J. *et al*. (2012) Genome-scale analysis of interaction dynamics reveals organization of biological networks. *Bioinformatics*, **28**, 1873–1878.

Faust,K. and Raes,J. (2012) Microbial interactions: from networks to models. *Nature Rev. Microbiol*., **10**, 538–550.

Kalari,K. *et al*. (2012) Deep sequence analysis of non-small cell lung cancer: integrated analysis of gene expression, alternative splicing, and single nucleotide variations in lung adenocarcinomas with and without oncogenic KRAS mutations. *Front. Oncol*., **2**, 12.

Kang,H. *et al*. (2011) Spatio-temporal transcriptome of the human brain. *Nature*, **478**, 483–489.

Karpinets,T. *et al*. (2012) Analyzing large biological datasets with association networks. *Nucleic Acids Res*., **40**, e131.

Nat. Biotech. (2012) Finding correlations in big data. *Nat. Biotech*., **30**, 334–335.

Reshef,D. *et al*. (2011) Detecting novel associations in large datasets. *Science*, **6062**, 1518–1524.

Speed,T. (2011) A correlation for the 21st century. *Science*, **6062**, 1502–1503.

Spellman,P. *et al*. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, **9**, 3273–3297.

Weiss,J. *et al*. (2012) "Good enough solutions" and the genetics of complex diseases. *Circ. Res*., **111**, 493–504.