BIOINFORMATICS APPLICATIONS NOTE Vol. 22 no. 20 2006, pages 2565–2566
doi:10.1093/bioinformatics/btl416

Data and text mining

PYCHEM: a multivariate analysis package for python

Roger M. Jarvis[1,4,*], David Broadhurst[1,4], Helen Johnson[2], Noel M. O'Boyle[3] and Royston Goodacre[1,4]

[1]School of Chemistry, The University of Manchester, PO Box 88, Sackville Street, Manchester M60 1QD, UK, [2]Faculty of Life Sciences, University of Manchester, Stopford Building, Oxford Road, Manchester M13 9PT, UK, [3]Unilever Centre for Molecular Science Informatics, Department of Chemistry, University of Cambridge, Lensfield Road, CB2 1EW, UK and [4]Manchester Interdisciplinary Biocentre, 131 Princess Street, Manchester M1 7DN, UK

Received on April 4, 2006; revised on July 5, 2006; accepted on July 26, 2006
Advance Access publication July 31, 2006
Associate Editor: Martin Bishop

ABSTRACT

Summary: We have implemented a multivariate statistical analysis toolbox, with an optional standalone graphical user interface (GUI), using the Python scripting language. This is a free and open source project that addresses the need for a multivariate analysis toolbox in Python. Although the functionality provided does not cover the full range of multivariate tools that are available, it has a broad complement of methods that are widely used in the biological sciences. In contrast to tools like MATLAB, PyChem 2.0.0 is easily accessible and free, allows for rapid extension using a range of Python modules and is part of the growing amount of complementary and interoperable scientific software in Python based upon SciPy. One of the attractions of PyChem is that it is an open source project and so there is an opportunity, through collaboration, to increase the scope of the software and to continually evolve a user-friendly platform that has applicability across a wide range of analytical and post-genomic disciplines.

Availability: http://sourceforge.net/projects/pychem

Contact: Roger.Jarvis@manchester.ac.uk or admin@pychem.org.uk

Supplementary information: Further information is available from the project home page at http://pychem.sf.net/ whilst details of data generation are available at http://biospec.net/

1 INTRODUCTION

Increasingly in the life sciences many experiments generate data which are of a multivariate nature, where many observations are recorded for each sample under analysis. Interpretation of such complex data cannot generally be performed by taking a univariate approach, since no single measurement is necessarily adequate enough to describe the problem being addressed. In fact, the application of univariate methodology is in many cases totally inappropriate as the complexity of information contained within large biological datasets reflects the complexity of the system(s) being studied. Typical multivariate analysis problems involve unsupervised learning such as factor analysis, for reducing the dimensionality of data and modeling of variance; linear regression, for formulating input to output transformation models based on supervised learning which are predictive generally for quantitative trait(s); and discriminant analysis, for distinguishing between different sample groups and for subsequent predictions on new samples. In fact, multivariate analysis encompasses many more methods than these examples of linear modeling imply (Brereton, 2003); but these tools are perhaps those most commonly used for the modeling of biological data.

Many programs currently exist for multivariate analysis. Flexible environments for mathematical computing are available in the form of MATLAB (The Mathworks, Natick, MA, USA), GNU Octave (http://www.octave.org/) which aims to be a free equivalent of MATLAB, and R (http://www.r-project.org/); which has many other bio-analysis modules, such as Vegan (for environmetrics) and Bioconductor (for genomic analysis). These products provide powerful tools for multivariate analysis through command line interpreters, which allow the user to perform their analysis with a great degree of flexibility. However, they require some investment in time to become familiar with the interpreters syntax, and are not necessarily straightforward for people with little computational experience. In addition, a number of graphical multivariate software tools are also available; Evince (UmBio, Umeå, Sweden), The Unscrambler (CAMO, Woodbridge, NJ, USA), Pirouette (Infometrix, Bothell, WA, USA), S-Plus (Insightful, Seattle, WA. USA) and SIMCA (Umetrics, Umeå, Sweden) are all good tools for basic multivariate analysis although, with the exception of S-Plus, they lack the flexibility of the interpreter style interfaces.

Thus there is currently a requirement for a flexible, extensible, free and open source graphical environment for performing multivariate analysis, which can be used by both experts and casual users. The increasing popularity of scripting languages such as Python (http://www.python.org/) within the life sciences community offers the technology and critical mass for such a project. A platform of this type addresses the requirements outlined above, with the additional benefit that it allows for the rapid development of new cross-platform software approaches, and the integration of currently available software libraries through application programming interfaces (APIs).

2 THE MULTIVARIATE ANALYSIS TOOLBOX FOR PYTHON

The PyChem project aims to provide a simple multivariate analysis toolbox with a powerful and intuitive GUI front-end.

*To whom correspondence should be addressed.

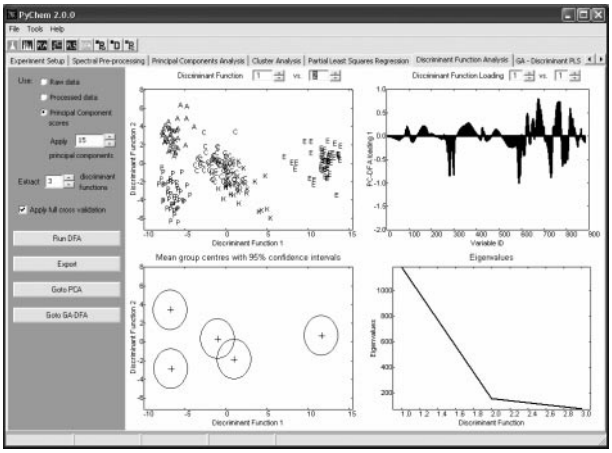

Fig. 1. A screenshot demonstrating the feature selection functionality available in PyChem, in this example microarray data (Golub *et al*., 1999) have been analysed consisting of 72 samples represented by 7070 genes. The GA directed search can be used to highlight genes that are particularly important for discrimination.

The project is implemented in Python and utilizes the wxPython (http://www.wxpython.org/), Boa Constructor (http://boa-constructor.sourceforge.net/) and SciPy (http://scipy.org/) packages (see Fig. 1 for an example screenshot) amongst others. The software was designed to provide a range of algorithms that address three fundamental questions commonly asked by the researcher.

(1) What is the shape of the data—including sources of variance and outlier identification?

(2) How similar are different samples?

(3) Which measurements from the original data can be attributed to observed differences and/or similarities?

To help answer these questions, the initial release includes algorithms for the pre-processing of multivariate data (such as scaling, baseline correction, filtering and derivatization), principal components analysis (PCA) (Jolliffe, 1986), partial least squares regression (PLS1) (Martens and Naes, 1989), discriminant function analysis (DFA) (Manly, 1994), cluster analysis [using the C clustering library for Python (http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/cluster/) (Eisen *et al*., 1998; de Hoon *et al*., 2004)], and a number of genetic algorithm (GA) based tools for performing feature selection (Jarvis and Goodacre, 2005), see Fig. 1.

The software is able to handle any 2D dataset where each sample is defined by a series of discrete or continuous measurements. Data can be imported from flat ASCII files that use the standard delimiters. Typical data of this type include those generated from microarrays, proteomics, spectroscopic methods (UV-Vis, infrared and Raman), mass spectrometry, NMR, or indeed any data arrays representing samples for which multiple discrete measurements have been acquired. Once data have been imported into PyChem they can be saved in an XML format [implemented using cElementTree (http://effbot.org/)] as a PyChem experiment, which allows for the subsequent storage of multiple experimental results within a single file. This allows for the capture of the state of the system at a point in time, so that results of multivariate analyses can be stored and the progression of the analysis recorded, which is particularly useful for tracking data analyses as part of GLP. The additional benefit of using the XML data structure for storage is that it introduces the potential for engineering simple bespoke interfaces to database storage systems.

PyChem provides simple grid-style user interfaces for the input of experimental and sample metadata, so producing a series of vectors describing the origin and identity of each sample and measured variable. For unsupervised analyses, such as PCA, the software simply requires a vector, or multiple vectors of sample labels for plotting; in addition for supervised analyses, vectors are required to (1) represent putative class structures or some quantitative trait (e.g., level of abiotic or biotic interference) and (2) identify groups in to which the data should be split for the purpose of cross-validation. In supervised analyses the issue of model validation is crucial; when a model is formulated there is a possibility that it will overfit the data and find a relationship between the data and the target class structure or dependent variables, which does not hold for subsequent predictions; i.e. the model has learnt the training data perfectly and is not able to generalize. This situation can be avoided by performing some form of model validation. In the current version of PyChem (2.0.0) we use the preferred approach of data splitting (Brereton, 2003), which works by dividing the measured X-variables in to three groups; a model training set, model cross-validation data and finally an independent test set. The model is trained on the first set, optimized on the second set and then tested for accuracy on the third set of 'hold-out' data.

A major emphasis of this work has been in providing clear and useful graphical reports for the interpretation of results. The GUI uses wxPyPlot (http://www.cyberus.ca/~g_will/wxPython/wxpyplot.html), with a small modification to include text plotting. In the future even more focus will be given to the structure of graphical reporting in PyChem, as well as the functionality associated with the plotting canvases. Finally, all results, both graphical and numerical, can easily be exported from PyChem, with numerical results in ASCII file format to allow for use in other software applications.

ACKNOWLEDGEMENTS

R.M.J., D.B., H.J., N.M.O.B. and R.G. would like to thank the BBSRC for funding (NMOB; grant BB/C51320X/1). Funding to pay the Open Access publication charges for this article was provided by the BBSRC.

Conflict of Interest: none declared.

REFERENCES

Brereton,R. (2003) Chemometrics: data analysis for the laboratory and chemical plant, 1st edn. Chichester: John Wiley & Sons Ltd.

Eisen,M. *et al*. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl Acad. Sci.USA*, **95**, 14863–14868.

de Hoon,M. *et al*. (2004) Open source clustering software. *Bioinformatics*, **20**, 1453–1454.

Golub,T. *et al*. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, **286**, 531–537.

Jarvis,R. and Goodacre,R. (2005) Genetic algorithm optimization for pre-processing and variable selection of spectroscopic data. *Bioinformatics*, **21**, 860–868.

Jolliffe,I.T. (1986) *Principal Component Analysis*. Springer-Verlag, New York.

Manly,B.F.J. (1994) *Multivariate Statistical Methods: A Primer*. Chapman & Hall/CRC, New York.

Martens,H. and Naes,T. (1989) *Multivariate Calibration*. John Wiley & Sons, Chichester.