Klinke, Sigbert; Ziegenhagen, Uwe; Guri, Yuval

Working Paper

Yxilon: a modular open-source statistical programming language

SFB 649 Discussion Paper, No. 2005,018

Provided in Cooperation with:
Collaborative Research Center 649: Economic Risk, Humboldt University Berlin

Suggested Citation: Klinke, Sigbert; Ziegenhagen, Uwe; Guri, Yuval (2005) : Yxilon: a modular open-source statistical programming language, SFB 649 Discussion Paper, No. 2005,018, Humboldt University of Berlin, Collaborative Research Center 649 - Economic Risk, Berlin

This Version is available at:
https://hdl.handle.net/10419/25037

SFB 649 Discussion Paper 2005-018

Yxilon – a Modular Open-Source Statistical Programming Language

Sigbert Klinke*
Uwe Ziegenhagen*
Yuval Guri*

* CASE - Center for Applied Statistics and Economics,
Humboldt-Universität zu Berlin, Germany

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".

http://sfb649.wiwi.hu-berlin.de
ISSN 1860-5664

SFB 649, Humboldt-Universität zu Berlin
Spandauer Straße 1, D-10178 Berlin

Yxilon - a modular open-source statistical programming language

Sigbert Klinke, Uwe Ziegenhagen and Yuval Guri
Humboldt-Universität zu Berlin, School of Business and Economics, Institute for Statistics and Econometrics, Spandauer Strasse 1, D-10178 Berlin, Germany
sigbert@wiwi.hu-berlin.de, ziegenhagen@wiwi.hu-berlin.de

Statistical research has always been at the edge of available computing power. Huge datasets, e.g in Data Mining or Quantitative Finance, and computationally intensive techniques, e.g. bootstrap methods, always require a little bit more computing power than is currently available. But the most popular statistical programming language `R`, as well as statistical programming languages like `S` or XploRe, are interpreted which makes them slow in computing intensive areas. The common solution is to implement these routines in low-level programming languages like C/C++ or Fortran and subsequently integrate them as dynamic linked libraries (DLL) or shared object libraries (SO) in the statistical programming language.

XploRe vs. Yxilon

Nearly 15 years ago the first versions of XploRe were implemented. In opposite to more mouse-oriented packages as SPSS and Minitab, XploRe was always targeted on a language based approach. While this approach inheres a usually flatter learning curve in comparison with mouse-based interaction we have learned from lectures and user feedback that the users receive a deeper understanding of the underlying theories and are more actively involved in the process of analysis.

With Yxilon we start an open source project to reimplement the XploRe language. Our aim is to compile directly to a low-level language as C/C++ or Java. The generated code can then be compiled either to DLL's/SO's or stand-alone programs. Further low-level languages, e.g. C#, are possible.

To download the current version visit http://www.quantlet.org.

Yxilon

For two reasons we have choosen XploRe rather than `R` as the basis of this project: XploRe is a non object-oriented programming language, for which a parser can be easily written and some of us have been involved for more than a decade in the development of the language. To produce a reasonable system we currently develop four components:

Yxilon GUI: a user interface for writing, interpreting, compiling and executing Yxilon code

Yxilon parser: a parser which compiles Yxilon programs to C/C++, Java or parse trees

Yxilon-J/Yxilon-C: libraries for matrix/array computations in Java and C/C++

Yxilon-RT: a run-time environment for the interpretation of parse trees

Other components, like the object database with web components, the middleware (see Figure 1) will be developed at a later stage of the project. The figure also depicts our intention of having "strong modularity" between the components which should simplify the development process. Changes targeted at improving modularisation and simplification will also have some impact on the XploRe language itself, see Härdle, Klinke and Ziegenhagen (2004).

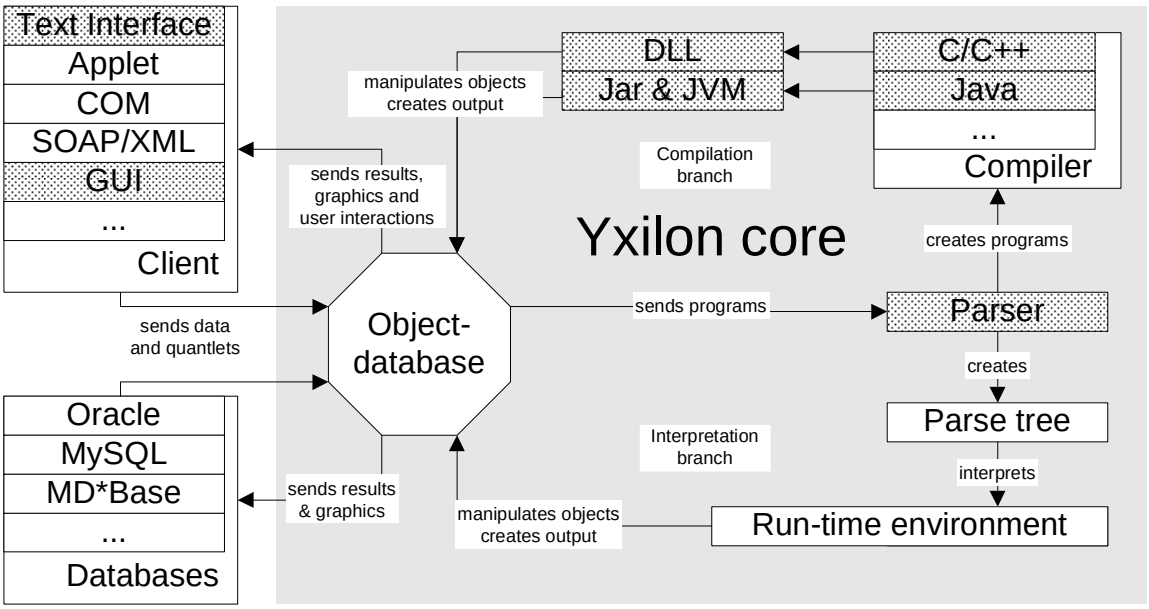

Figure 1: The Yxilon components (dotted components are work in progress or finished).

Following Chambers and Lang (1999) we provide a list of requirements for statistical applications, among them: usability from multiple front-ends, internet abilities to read and write data to networks and database support to enable large-scale analysis. We added: modularity and extensibility, support for multiple languages and valuable, integrated user support (tutorial, help system).

Yxilon GUI

The aim of the current graphical user interface is to satisfy two major needs:

- Integrating the parsing and interpretation modules under one roof
- Providing one experimental platform to analyze the different needs of different types of users
- Implementing and testing efficient ways of interaction and communication

In the analysis of user behavior we focus mainly on the central items of *usability* from Nielsen (1993): Learnability, Efficiency, Memorability, Errors and Satisfaction. Shneiderman (1998) furthermore defines 'Golden Rules of Interface Design', including questions concerning common rules (consistency, error handling and feedback) in the communication between user and software as well as psychological aspects (loss of control in interaction and limited human short term memory capacity).

Working especially with language based statistical software requires a significant commitment from the user, especially in the initial learning phase. With `log4j` from the Apache Foundation (`http://www.apache.org/log4j`) we have a powerful tool to log each interaction between user and software, together with questionnaires and interviews we hope to analyze and improve the user/software interface.

The layout and design of the Yxilon GUI is a mixture of the above design principles and earlier XploRe GUIs. All window and menu captions are provided via an initialization file. Java

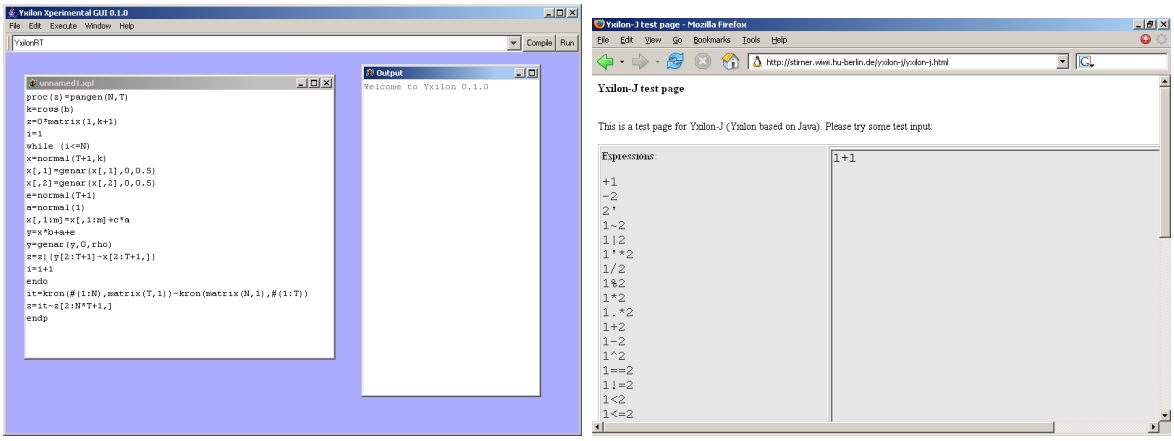

Figure 2: Left: The Yxilon GUI with editor and output window. Right: The Yxilon-J WWW test page with the current language implementation, see `http://stirner.wiwi.hu-berlin.de/yxilon-j/yxilon-j.html`.

offers native Unicode support, so the adaptation to different languages is easy. Furthermore the initialization file contains the relevant settings for the Java and C++ compiler. Our aim is to hide the technical details from the user as much as possible, thus sparing the implementation details of C++ and Java. This approach inheres possible error sources, if the Yxilon code contains either semantic or logical errors the generated code will not work either. It is therefore a main task in the development of the parser and the user interface to provide the user with valuable error messages.

Yxilon parser and interpreter

The Yxilon parser is a C/C++ program based on tools, compatible to lex and yacc, and generates different outputs: C/C++, Java and parse trees. Since we want to avoid the development of separate parsers for different target languages, which surely would lead to incompatible Yxilon language branches in the future, we decided to create plug-ins for the parser which generate the appropriate output. Other plug-ins than C/C++ or Java are possible.

The C/C++ code can be used to generate dynamic link libraries (Windows) or shared object libraries (Unix/Linux) for standalone programs. Slight modifications in the C/C++ code will allow to create COM code which can be embedded e.g. in Microsoft Office products (Excel, Word).

The third output method is the generation of a binary parse tree that contains Yxilon code in form of an execution tree. The binary data format is platform-independent. Similar to the Java programming system, a run-time environment (RTE) is able to interprete and execute these trees. The parser and the RTE together form an interpreter.

Yxilon-J: Yxilon compilation with Java

The Yxilon-J component which compiles Yxilon code to Java consists of four components:

1. the JavaWalker: a C/C++ plug-in for the parser which writes Java code. The parser builds up an expression tree which contains the code in a hierarchical structure; the Walker program analyzes the expression tree top-down and generates the appropriate Java code. Each Yxilon subprogram, called "quantlet" in XploRe, will translate to its own Java class.
2. Skeleton files: they define how the class header and footers for the main Yxilon program and

subprograms look like. Currently there are `@main.java`, containing a `main` method to run the program, and `@quantlet.java`, which can be called by either quantlets or the `main` method.

3. Colt 1.0.3: an efficient implementation of matrix/array classes in Java by Hoschek (2002).

4. `yxilon.jar`: a set of classes implementing the basic Yxilon functionalities (operators and selected commands).

Due to restrictions in the Java language, each Yxilon program will be stored in a separate `class` file, such that it can be reused without recompilation. These files (java source and bytecode) will be part of the Yxilon-J distribution.

Figure 2 (right) shows the Yxilon-J WWW test page which is similar to the CGI Interface of RWeb-Server page (see Banfield 1998). Left and below the form are same (non-)runnable code pieces. The form takes the Yxilon code, generates a Java file, compiles it and calls the Java Virtual Machine to execute it. Error messages and/or output is catched and displayed on a resulting web page.

Summary

Yxilon is an open source project for developing a new statistical programming language. It does as well generate C/C++ and Java code for compilation into dynamic link libraries or shared object libraries as it interpretes its own code by means of parse trees. Interested people are invited to join the project as developers or testers (http://www.quantlet.org).

REFERENCES

Banfield J. (1998). RWeb, http://www.math.montana.edu/Rweb/

Chambers, J., Lang, D. (1999). Omegahat – a component-based statistical computing environment, 'Proceedings of the 52nd ISI Session'

Härdle W., Klinke S., Ziegenhagen U. (2004). Yxilon - Designing the next generation, vertically integrable statistical software environment. 'Proceedings of the 36th Symposium on the Interface'

Hoscheck W. (2002). The Colt distribution, Version 1.0.3,
http://hoschek.home.cern.ch/hoschek/colt/

Nielsen J. (1993). Usability Engineering, AP Professional

Shneiderman, B. (1997). Designing the User Interface, 3. edn, Addison-Wesley Longman

ACKNOWLEDGEMENT

The financial support from the Deutsche Forschungsgemeinschaft, Sonderforschungsbereich 649 Economic Risk at Humboldt-Universitat zu Berlin, is gratefully acknowledged.

SFB 649 Discussion Paper Series

For a complete list of Discussion Papers published by the SFB 649, please visit http://sfb649.wiwi.hu-berlin.de.

001 "Nonparametric Risk Management with Generalized Hyperbolic Distributions" by Ying Chen, Wolfgang Härdle and Seok-Oh Jeong, January 2005.
002 "Selecting Comparables for the Valuation of the European Firms" by Ingolf Dittmann and Christian Weiner, February 2005.
003 "Competitive Risk Sharing Contracts with One-sided Commitment" by Dirk Krueger and Harald Uhlig, February 2005.
004 "Value-at-Risk Calculations with Time Varying Copulae" by Enzo Giacomini and Wolfgang Härdle, February 2005.
005 "An Optimal Stopping Problem in a Diffusion-type Model with Delay" by Pavel V. Gapeev and Markus Reiß, February 2005.
006 "Conditional and Dynamic Convex Risk Measures" by Kai Detlefsen and Giacomo Scandolo, February 2005.
007 "Implied Trinomial Trees" by Pavel Čížek and Karel Komorád, February 2005.
008 "Stable Distributions" by Szymon Borak, Wolfgang Härdle and Rafal Weron, February 2005.
009 "Predicting Bankruptcy with Support Vector Machines" by Wolfgang Härdle, Rouslan A. Moro and Dorothea Schäfer, February 2005.
010 "Working with the XQC" by Wolfgang Härdle and Heiko Lehmann, February 2005.
011 "FFT Based Option Pricing" by Szymon Borak, Kai Detlefsen and Wolfgang Härdle, February 2005.
012 "Common Functional Implied Volatility Analysis" by Michal Benko and Wolfgang Härdle, February 2005.
013 "Nonparametric Productivity Analysis" by Wolfgang Härdle and Seok-Oh Jeong, March 2005.
014 "Are Eastern European Countries Catching Up? Time Series Evidence for Czech Republic, Hungary, and Poland" by Ralf Brüggemann and Carsten Trenkler, March 2005.
015 "Robust Estimation of Dimension Reduction Space" by Pavel Čížek and Wolfgang Härdle, March 2005.
016 "Common Functional Component Modelling" by Alois Kneip and Michal Benko, March 2005.
017 "A Two State Model for Noise-induced Resonance in Bistable Systems with Delay" by Markus Fischer and Peter Imkeller, March 2005.

SFB 649, Spandauer Straße 1, D-10178 Berlin
http://sfb649.wiwi.hu-berlin.de

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".

018 "Yxilon – a Modular Open-source Statistical Programming Language" by Sigbert Klinke, Uwe Ziegenhagen and Yuval Guri, March 2005.

SFB 649, Spandauer Straße 1, D-10178 Berlin
http://sfb649.wiwi.hu-berlin.de

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".