CSB: a Python framework for structural bioinformatics

Ivan Kalev1,*, Martin Mechelke1, Klaus O. Kopec1, Thomas Holder1, Simeon Carstens1 and Michael Habeck1,2,*

1Department of Protein Evolution, Max Planck Institute for Developmental Biology, Spemannstrasse 35 and 2Department of Empirical Inference, Max Planck Institute for Intelligent Systems, Spemannstrasse 38, 72076 Tübingen, Germany

ABSTRACT

Summary: Computational Structural Biology Toolbox (CSB) is a cross-platform Python class library for reading, storing and analyzing biomolecular structures with rich support for statistical analyses. CSB is designed for reusability and extensibility and comes with a clean, well-documented API following good object-oriented engineering practice.

Availability: Stable release packages are available for download from the Python Package Index (PyPI) as well as from the project’s website http://csb.codeplex.com.

Contacts: ivan.kalev@gmail.com or michael.habeck@tuebingen.mpg.de

Received on July 10, 2012; revised on August 6, 2012; accepted on August 24, 2012

1 INTRODUCTION

The Python programming language is becoming an increasingly popular choice in research. With its comprehensive numerical libraries and dynamic type system, Python facilitates rapid application development. But although rapid prototyping is very practical for experimenting with new techniques or features, systematic use of ad hoc scripting often turns into a burden preventing efficient code reuse. This problem is solved with the use of continuously developed, well-abstracted and tested software libraries. Productivity in building solid, reliable and extensible bioinformatics applications could therefore significantly benefit from the practice of using library code.

Here, we introduce the public release of CSB, a Python library designed for solving problems in the field of computational structural biology. CSB improves over existing libraries such as Biopython (Cock et al., 2009) with its granular, consistent and extensible object model and also provides new features like a comprehensive statistical API and support for new abstractions and file formats. This project is a quickly growing class library for structural bioinformatics, providing clean object-oriented APIs for working with biological macromolecular structures, sequences, sequence profiles and fragment libraries, and also a significant amount of statistical modules, including many probability distributions and samplers. We put a strong emphasis on quality and reliability achieved through continuous attention to good software design and best practices in test engineering.

*To whom correspondence should be addressed.
We compared the performance of \textit{RegularStructureParser} with PDB I/O modules from alternative libraries: Biopython, PyCogent (Cielik \textit{et al.}, 2011) and the C++ based Open Structure (Biasini \textit{et al.}, 2010). As expected, OpenStructure was the fastest and parsed 4000 PDB entries with 0.09 s per structure. CSB is positioned between Biopython (0.19 s) and PyCogent (0.43 s) with 0.32 s per structure, which suggests that the SEQRES mapping feature comes with an acceptable performance overhead.

Our library also hosts a collection of statistical models in the \texttt{csb.statistics} namespace. Among these models are standard univariate and multivariate probability distributions such as the Normal and the Gamma distribution and also more exotic distributions such as the multivariate normal inverse Gaussian distribution used to model multivariate heavy-tailed data. Several estimators based on maximum likelihood and Gibbs sampling are implemented. Moreover, we provide a general framework for Markov chain Monte Carlo simulation and implementation of standard schemes such as random walk Metropolis Hastings, Hamiltonian Monte Carlo (Duane \textit{et al.}, 1987) and replica-exchange Monte Carlo (Swendsen and Wang, 1986). Methods to analyze Monte Carlo output are also provided such as, for example, a non-parametric histogram reweighting scheme for the estimation of free energy differences (Habeck, 2012).

4 DEVELOPMENT

One of the key design goals of CSB is providing clean, extensible, object-oriented APIs with accompanying API documentation. This project puts a strong emphasis on quality, achieved through systematic use of abstraction, strong encapsulation, separation of responsibilities and refactoring with classic design patterns.

Our development team has adopted a continuous integration model. The reliability of the production code is controlled by CSB’s built-in high-coverage unit test framework. Stable builds will be gradually released to the public domain, and nightly builds can be obtained upon request. Portability is also a design goal, so CSB works without modification on every major platform (Windows, Linux and Mac) and any modern Python interpreter (version 2.6 or higher, including Python 3).

\textbf{Funding:} Contract research ‘Methoden in den Lebenswissenschaften’ of the Baden-Württemberg Stiftung, by Deutsche Forschungsgemeinschaft (DFG) grant HA 5918/1-1 and by the Max Planck Society.

\textbf{Conflict of Interest:} none declared.

\section{CSB APPLICATIONS}

CSB comes with a simple framework for writing console applications (\texttt{csb.apps}). These applications could be seen as short protocols built on top of the core library and consuming its APIs. Each release is bundled with a number of pre-installed, open-source applications. For example, \texttt{csb.apps.hhfrag} provides HHfrag, a CSB application for building dynamic fragment libraries (Kalev and Habeck, 2011), BFit is another app, which performs robust superposition of protein structures (Mechelke and Habeck, 2010). Every release package also contains EMBD, an application for sharpening of cryo-electron microscopy maps (Hirsch \textit{et al.}, 2011) using non-negative deconvolution and Promix, an application implementing Gaussian mixture models for identifying rigid domains in structure ensembles (Hirsch and Habeck, 2008).

\section{REFERENCES}


