Sequence analysis

**MSAViewer: interactive JavaScript visualization of multiple sequence alignments**

Guy Yachdav,1,2 Sebastian Wilzbach,1 Benedikt Rauscher,1 Robert Sheridan,3 Ian Sillitoe,4 James Procter,5 Suzanna E. Lewis,6 Burkhard Rost1,2 and Tatyana Goldberg1,*

1Bioinformatik - I12, TUM, Garching, 85748, Germany, 2Biosof LLC, New York, NY 10001, USA, 3Department of Systems Biology, Harvard Medical School, Boston, MA 02115, USA, 4Institute of Structure and Molecular Biology, University College London, London, UK, 5Biological Chemistry and Drug Discovery, University of Dundee, Dundee, UK and 6Lawrence Berkeley National Laboratory, Berkeley, USA

*To whom correspondence should be addressed.

Associate Editor: John Hancock

Received on April 16, 2016; revised on June 3, 2016; accepted on June 29, 2016

**Abstract**

**Summary:** The MSAViewer is a quick and easy visualization and analysis JavaScript component for Multiple Sequence Alignment data of any size. Core features include interactive navigation through the alignment, application of popular color schemes, sorting, selecting and filtering. The MSAViewer is ‘web ready’: written entirely in JavaScript, compatible with modern web browsers and does not require any specialized software. The MSAViewer is part of the BioJS collection of components.

**Availability and Implementation:** The MSAViewer is released as open source software under the Boost Software License 1.0. Documentation, source code and the viewer are available at http://msa.biojs.net/.

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

**Contact:** msa@bio.sh

**1 Introduction**

Multiple Sequence Alignment (MSA) is a fundamental procedure to capture similarities between sequences of nucleotides (DNA/RNA) or of amino acids (protein). Biologically meaningful MSAs highlight and capture sites with significant evolutionary conservation. MSAs are essential to predict aspects of protein structure (e.g. secondary structure (Rost, 2001) or protein disorder (Schlessinger et al., 2011)) and function (e.g. binding sites (Ofran et al., 2007) or localization (Goldberg et al., 2014)). MSAs can be used to understand genomic rearrangements (Darling et al., 2004), to derive sequence homology (Altschul et al., 1990) and to identify evolutionary rates (Pupko et al., 2002).

MSAs are widely used to display complex annotations relating to structure and function, and to transfer those annotations to sequences that lack annotations (Waterhouse et al., 2009). Many tools are available to view and analyze MSAs, including standalone applications (Larsson, 2014; Waterhouse et al., 2009) and web applets (Waterhouse et al., 2009). With the recent widespread adoption of JavaScript as the leading programming language for interactive web applications, new MSA viewing tools compatible with modern web browsers have been developed and made available (Martin, 2014). BioJS is one particular collection of JavaScript components with growing applications in biology (Corpas et al., 2014); it is interoperable with many other data visualization tools.

Here, we describe the BioJS MSAViewer. It is readily loaded into web pages to visualize and analyze MSA datasets of arbitrary sizes. MSAViewer implements most features commonly available in other popular MSA viewing software, including scrolling, selecting, highlighting, cross-referencing with protein feature annotations and phylogenetic trees (a detailed comparison of features is in Supplementary Table S1).
2 Visualization

The MSAViewer loads MSA data in FASTA (Pearson, 2000) or ClUSTAL (Larkin et al., 2007) formats from a user’s local computer or a web server. It then draws two main Canvas panels—the main panel and the overview MSA panel (Fig. 1). The choice to use Canvas over other rendering technologies is discussed in Supplementary Material Section S1.

Navigation through the alignment is enabled through various controls. First, users can scroll or use the ‘jump to a column’ menu item to navigate to a certain column number. Second, users can pan within the main panel to scroll through the alignment; this has proven to be a useful feature for large alignments. Finally, a second panel—the overview panel (drawn under the main panel)—provides a ‘bird’s eye view’ perspective over the entire alignment and can also be used for navigation.

The alignment is sortable by unique identifiers, sequence labels and sequences. The percentage of gaps and the sequence identity to the consensus sequence, which is calculated from most frequent bases at each position in the alignment, can also be used for sorting. Users can select sequences, columns, or arbitrary regions for analysis, and hide them from the main panel based on their selection, conservation to the consensus sequence or the percentage of gaps. An alignment can also be searched for motifs using a regular expression (e.g. K(K|R)RK for a nuclear localization signal), which are highlighted with a red frame if matched. Sequence position annotations, such as binding sites, are provided by a user and displayed as filled rectangles below the corresponding sequence in the alignment. Users can switch between 15 predefined color schemes. The MSAViewer exports alignments and annotations as ASCII files, and the visual representation as a publication-quality figure.

3 Summary

MSAViewer is a lightweight viewer for MSAs of arbitrary size. Being JavaScript-based, it can be used on any modern web browser without installing any specialized software or add-ons. As part of BioJS, the MSAViewer can interoperate with a growing set of biological data viewers. The phylogenetic tree and the sequence logo viewers are already integrated in the current release. Integration with new MSA visualization techniques, e.g. sequence bundles (Kulys et al., 2014) is planned. The MSAViewer has already been found useful and became part of Galaxy (Giardine et al., 2005) https://cpt.tamu.edu/clustalw-msa-and-visualisations and JalView (Waterhouse et al., 2009) http://www.jalview.org/help/html/features/biojsmsa.html.

Funding

This work was supported by a grant from the German Federal Ministry for Education and Research (BMBF), Ernst Ludwig Ehrlich Studienwerk and the Google Summer of Code program sponsored by Google Inc.

Conflict of Interest: none declared.

References
