BMC Bioinformatics

SOFTWARE Open Access

svist4get: a simple visualization tool for genomic tracks from sequencing experiments

Artyom A. Egorov[1,2*], Ekaterina A. Sakharova[3], Aleksandra S. Anisimova[1,4], Sergey E. Dmitriev[1,4,5], Vadim N. Gladyshev[1,6] and Ivan V. Kulakovskiy[1,3,5,7*]

Abstract

Background: High-throughput sequencing often provides a foundation for experimental analyses in the life sciences. For many such methods, an intermediate layer of bioinformatics data analysis is the genomic signal track constructed by short read mapping to a particular genome assembly. There are many software tools to visualize genomic tracks in a web browser or with a stand-alone graphical user interface. However, there are only few command-line applications suitable for automated usage or production of publication-ready visualizations.

Results: Here we present svist4get, a command-line tool for customizable generation of publication-quality figures based on data from genomic signal tracks. Similarly to generic genome browser software, svist4get visualizes signal tracks at a given genomic location and is able to aggregate data from several tracks on a single plot along with the transcriptome annotation. The resulting plots can be saved as the vector or high-resolution bitmap images. We demonstrate practical use cases of svist4get for Ribo-Seq and RNA-Seq data.

Conclusions: svist4get is implemented in Python 3 and runs on Linux. The command-line interface of svist4get allows for easy integration into bioinformatics pipelines in a console environment. Extra customization is possible through configuration files and Python API. For convenience, svist4get is provided as pypi package. The source code is available at https://bitbucket.org/artegorov/svist4get/

Keywords: Genomic tracks, Visualization, Next-generation sequencing, High-throughput sequencing, Ribo-Seq, RNA-Seq, Python, Genome browser

Background

Next-generation sequencing gave birth to multiple high-throughput methods of the life sciences, many of which are based on mapping short sequence reads to an existing genome assembly. Visualization of mapped read densities and computationally derived genome signal tracks is one of the most routine tasks in bioinformatics sequencing data analysis. One approach is the usage of dedicated genome browsers. The most popular universal tools such as UCSC Genome Browser [1] or Zenbu [2] are web-based and allow interactive exploration of existing genome annotation along with uploaded user data. However, in some cases, it is not convenient to upload the user data to a remote server, and the data can be visualized and explored with the help of stand-alone applications with graphical user interface such as Integrated Genome Viewer [3] and Integrated Genome Browser [4], or even directly in the console environment [5]. Finally, there are hybrid approaches, for example, BioUML bioinformatics platform [6] provides genome browsing functionality in both web-based and stand-alone versions.

Web-based genome browsers are great for exploratory data analysis of processed public data, but stand-alone tools are better suited for generation of custom paper-quality graphics, and for exploring data from ongoing or private experiments. In addition, it is often convenient to have a programmatic way to generate multiple images for several genomic loci. There are several existing tools aimed

* Correspondence: artyom.egorov@hotmail.com; ivan.kulakovskiy@gmail.com
[1]Belozersky Institute of Physico-Chemical Biology, Lomonosov Moscow State University, Leninskiye gory 1, Moscow 119234, Russia
Full list of author information is available at the end of the article

© The Author(s). 2019 **Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated.

specifically to solve this task via scripting or command-line interface (Table 1). Among the most advanced tools, there are Gviz [7] and ggbio [8], the Bioconductor R packages dedicated to the production of paper-quality figures of genomic tracks and annotations. Users preferring command-line utilities can use fluff [9] and ngs.plot [10]. These tools provide some advanced functions for data analysis but allow only a minimalistic approach to the visualization of genomic track segments in particular genomic windows. Here we present svist4get, the software tool allowing detailed paper-quality visualization of signal tracks along transcriptome annotation at a particular genomic location.

All the listed tools can function in Linux environment and support bed or bedGraph format for genomic signal tracks and gtf or gff for genomic annotation. Most of the tools are not focused on visualization of genomic windows and include advanced functions for data analysis or exploration.

Implementation

Svist4get is implemented in Python 3 and uses multiple pypi packages (argparse, biopython [11], configs, reportlab, pybedtools [12], wand, wheel). Pypi 'wand' package and ImageMagick are utilized for pdf-to-png conversion. Svist4get was developed and tested in Linux environment. The python svist4get package is available in pypi (`python3 -m pip install svist4get`), the source code and example data are provided in Additional file 1. Details of svist4get installation are given in Additional file 2.

As input data, svist4get supports bedGraph format for genomic signals and gtf format of the genome annotation. As the output data, svist4get can generate vector graphics in pdf and export raster graphics in png. ImageMagick is used to provide raster (png) output.

Given a particular genomic window and a set of genomic signal tracks, svist4get automatically performs moving-average smoothing of the signal tracks, if necessary, taking into account the image width and the visible length of the genomic window. However, svist4get is a pure visualization tool, thus the technical data conversion and pre-processing, such as read depth normalization, should be performed with external tools, such as deeptools [13], bedtools [14], or UCSC utilities [15].

To facilitate application of svist4get in standard scenarios and data exploration, the command line interface covers several practical use cases that arise in transcriptomic studies, without additional effort for user-side scripting. Furthermore, svist4get provides a Python API allowing additional customization and programmatic usage from within a Python program. The use cases and examples of svist4get results are described in the next section.

Results and discussion

Svist4get capabilities are demonstrated in [16], where figures were produced with svist4get Python API. Here we show several practical use cases of the command-line interface by visualizing particular genomic windows related to genes and transcripts using existing genome annotation. The command line parameters to reproduce the presented images are provided in Additional file 2.

The basic cases (Figs. 1 and 2) are illustrated by using yeast ribosome profiling (Ribo-Seq) and RNA-Seq data from [17] downloaded from GWIPs-viz [18], and yeast genome reference annotation from Ensembl [19]. For convenience, in the svist4get package, we include truncated sample data that is used for demonstration purposes. Visualization of tissue-specific expression of different transcript isoforms (Fig. 3) uses mouse Ribo-Seq data [20, 21] that was downloaded from GWIPs-viz [18, 22] and GENCODE M13 [23] mouse genome annotation.

Basic visualization of genomic windows

We employed svist4get to generate a visualization of the genomic window containing the YFL031W transcript of HAC1 gene (Fig. 1). Based on genome annotation and a transcript identifier, svist4get selects a genomic window that includes a particular transcript. Alternative scenarios include the selection of a genomic window based on gene identifier and visualization of all transcripts in a given window (Additional file 2). Svist4get renders the transcript structure (based on genome annotation) as the top track, below it places the signal tracks (based on data in bedGraph format), and the structure of open reading frames (0, + 1, + 2, based on the nucleotide sequence of the displayed window) is shown at the bottom.

Table 1 An overview of existing programmatic visualization tools for genomic signal tracks

	svist4get	fluff	ngs.plot	ggbio	Gviz	ASCIIGenome
Command-line interface	yes	yes	yes	no	no	yes
Programming language	Python	Python	R, Python	R	R	Java
API	yes	no	no	yes	yes	no
Output	pdf, png	png	png, tiff	pdf, png	pdf, png	console (text)
Vector graphics output	yes	no	no	yes	yes	no
Reference		[9]	[10]	[8]	[7]	[5]

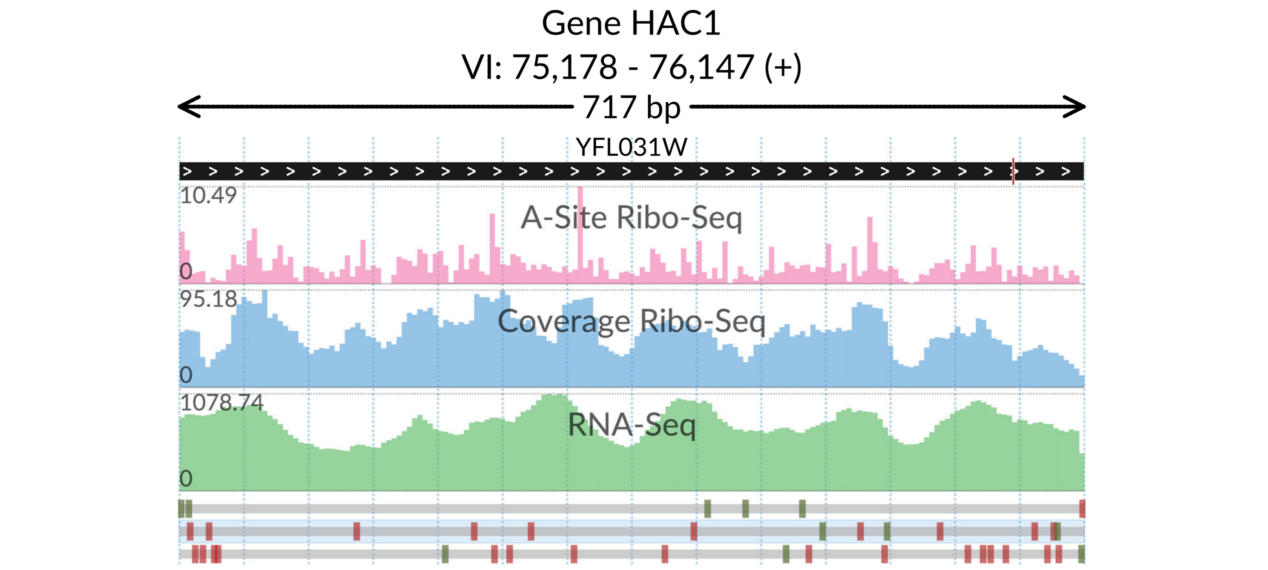

Fig. 1 Transcript-centric selection and visualization of a genomic window. The top track shows the YFL031W transcript structure with the collapsed intronic region (short red bar on the right). The tracks in the middle show Ribo-Seq (ribosome A-sites and aggregated read density) and RNA-Seq (aggregated read density) signals. The bottom track shows the 0, + 1, and + 2 reading frames with the start and stop codons marked by green and red bars, respectively. The transcript open reading frame is highlighted. The data is taken from [17]

Visualizing a genomic window at the single-nucleotide resolution

We also used svist4get to show a surrounding region of a translation initiation site of DFG16 yeast gene (Fig. 2), including an upstream open reading frame (ORF). The general layout of tracks in Fig. 2 is similar to that of Fig. 1. An additional track is used to show arbitrary genomic segments with user-defined labels (upstream ORF and CDS). A smaller genomic region surrounding DFG16 translation initiation site was selected based on transcript ID. A wider template (the predefined configuration file) allowed single-nucleotide resolution.

Visualizing ribosome occ2upancy in overlapping transcripts

We also show a multi-track visualization illustrating differential ribosome occupancy in mouse kidney and liver Ribo-Seq data (Fig. 3). Reconcilable parts of introns of two annotated transcripts are collapsed (red vertical marks on the transcript structure tracks) to facilitate a

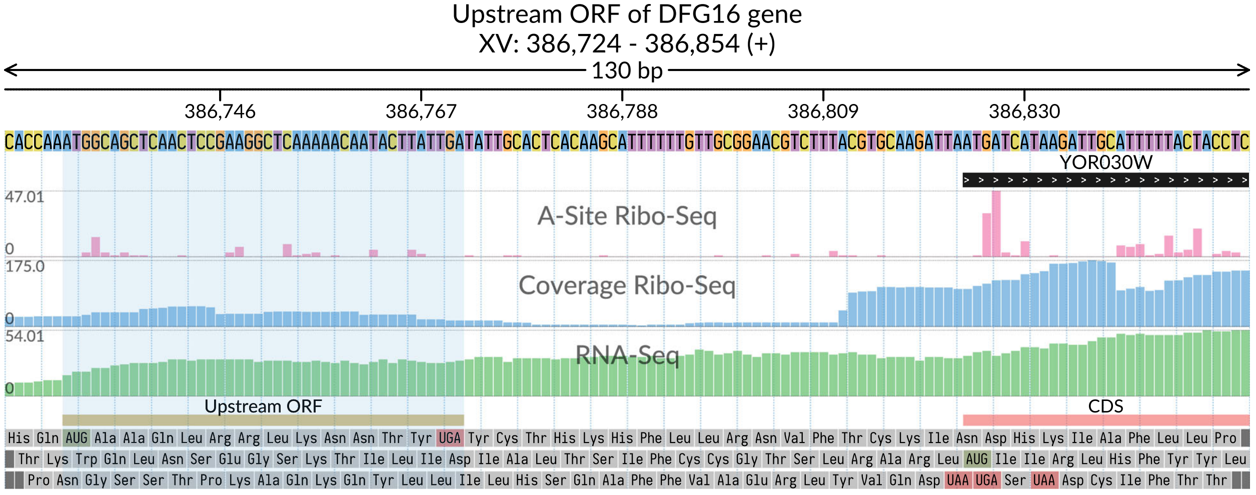

Fig. 2 Ribo-Seq (ribosome A-sites and aggregated coverage) and RNA-Seq (aggregated coverage) signals in the vicinity of the translation initiation site of DFG16 gene. Upstream ORF in 5′ region is highlighted. In comparison to Fig. 1, the genomic window has lower length and the image uses a wider template, allowing single-nucleotide resolution. The data is taken from [17]

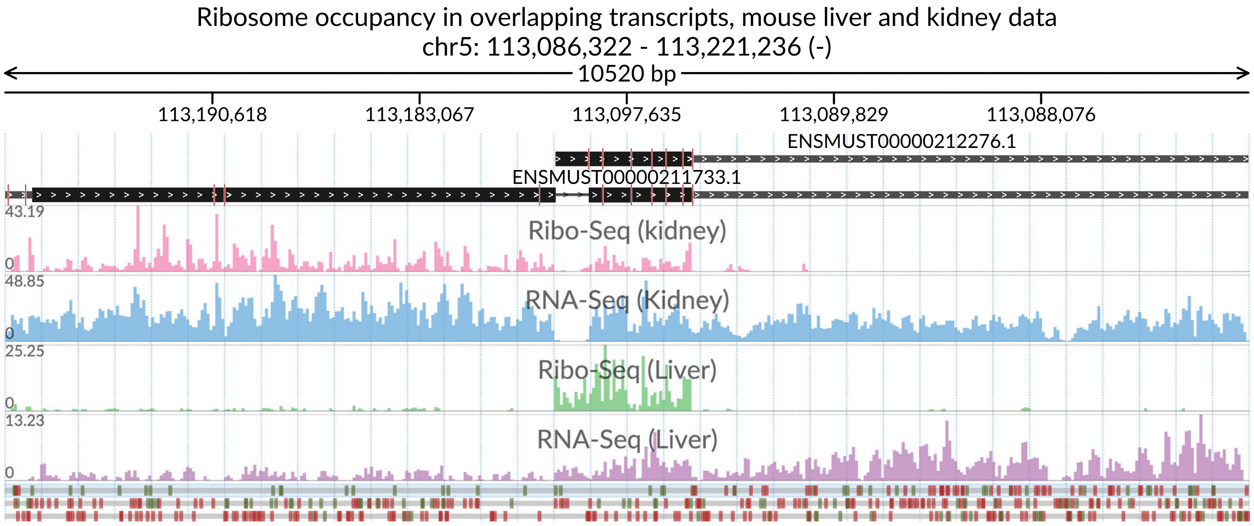

Fig. 3 Ribo-Seq and RNA-Seq aggregated coverage signals in mouse kidney and liver data. The genomic window is centered on overlapping annotated transcripts displaying tissue-specific ribosome occupancy (Ribo-Seq tracks) and transcript abundance (RNA-Seq tracks). The red marks on the transcript structure track (on top) correspond to the collapsed intronic regions which are reconcilable for both shown transcripts. The data is taken from [20, 21]

non-interrupted view of the translated shortened open reading frame that is specific to the liver.

Advanced features and customization

A basic bedGraph track is potentially useful to display various transcriptomic and genomic signals, such as DNase-Seq or ChIP-Seq. However, it is often necessary to visually separate signals on the primary and the reverse complementary DNA strands. To this end, svist4get provides paired bedGraph tracks, which use a single Y-axis to plot signals from a given pair of bedGraph files in the positive and negative value ranges (Fig. 4). Figure 4 also demonstrates multiple highlighting by showcasing translated segments of the MATa locus transcripts.

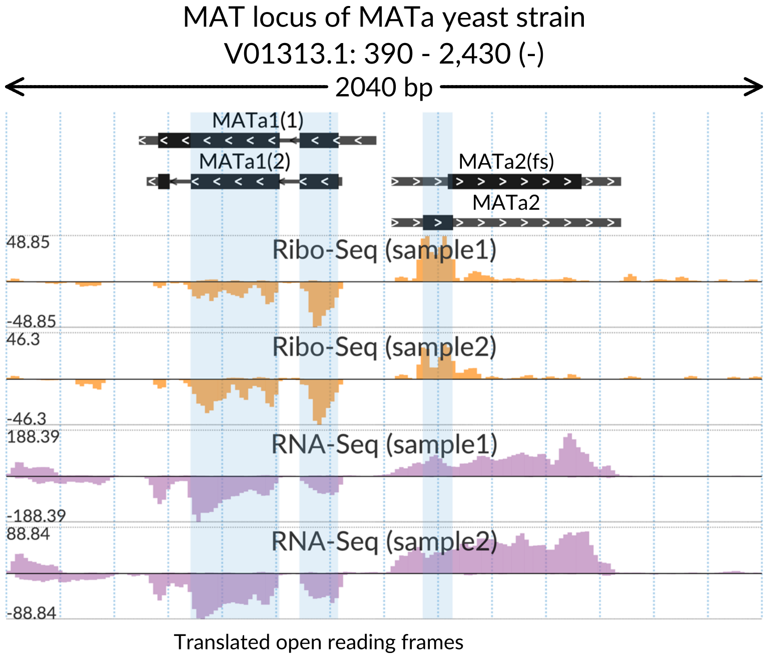

Fig. 4 Ribo-Seq and RNA-Seq aggregated coverage of MAT locus in MATa yeast strain. Translated segments of transcripts are highlighted. Paired bedGraph tracks with custom colors are used to show coverage of two DNA strands separately. The data is taken from [16]

The visualization of svist4get is highly customizable. Some essential options, such as custom track coloring, are available directly through the command-line interface. Other parameters, such as color palette, bitmap DPI setting, font typeface, and page size are defined in configuration files (see Additional file 2 for details). The package includes default color palette and editable configuration files for generating figures to fit one- and two-column layout of an A4 page.

Conclusions

Data from high-throughput sequencing requires specialized visualization tools. Here, we present svist4get, which produces publication-quality images of signal tracks along transcript structure in arbitrary genomic windows. We believe svist4get provides a reasonable compromise between tools with advanced R APIs and user-friendly graphical interfaces and can be useful as a component of bioinformatics pipelines as well as a stand-alone tool for data exploration.

Availability and requirements

Project name: svist4get.

Project home page: https://bitbucket.org/artegorov/svist4get

Operating system(s): Linux.

Programming language: Python 3.

Other requirements: pypi packages (argparse, biopython, configs, reportlab, pybedtools, wand, wheel), ImageMagick (OS-level requirement for wand).

License: WTFPL http://www.wtfpl.net

Additional files

Additional file 1: Svist4get source code and sample data. Python code and samples of yeast data used for generation of figures. (TGZ 5787 kb)

Additional file 2: Svist4get installation instructions and command-line examples. Installation instructions and console commands to reproduce the figures. (PDF 393 kb)

Acknowledgments

We thank Ilya Vorontsov and Andrey Buyan for valuable feedback and testing the software.

Funding

The work was primarily supported by a Russian Federation grant (14.W03.31.0012). NGS data analysis pipeline was supported by Russian Foundation for Basic Research (grant 18–34-20024 to IVK).

Availability of data and materials

No datasets were generated during the current study. The samples of yeast data of [17] analyzed during this study are included in this published article and its supplementary information files. The mouse data of [20, 21] are available at GWIPs-viz [18].

Authors' contributions

AAE implemented the program and prepared the figures; EAS performed code review, refactoring and testing; ASA, SED, and VNG designed the use cases and suggested key features; IVK supervised the project and drafted the manuscript together with AAE. All the authors have participated in the manuscript preparation.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details

[1]Belozersky Institute of Physico-Chemical Biology, Lomonosov Moscow State University, Leninskiye gory 1, Moscow 119234, Russia. [2]Department of Medical Physics, Faculty of Physics, Lomonosov Moscow State University, Leninskiye gory 1-2, Moscow 119991, Russia. [3]Vavilov Institute of General Genetics, Russian Academy of Sciences, Gubkina 3, Moscow 119991, Russia. [4]Faculty of Bioengineering and Bioinformatics, Lomonosov Moscow State University, Leninskiye gory 1-73, Moscow 119234, Russia. [5]Engelhardt Institute of Molecular Biology, Russian Academy of Sciences, Vavilova 32, Moscow 119991, Russia. [6]Division of Genetics, Department of Medicine, Brigham and Women's Hospital and Harvard Medical School, Boston, MA 02115, USA. [7]Institute of Mathematical Problems of Biology RAS - the Branch of Keldysh Institute of Applied Mathematics of Russian Academy of Sciences, Vitkevicha 1, Pushchino 142290, Moscow Region, Russia.

Received: 24 October 2018 Accepted: 27 February 2019
Published online: 06 March 2019

References

1. Fujita PA, Rhead B, Zweig AS, Hinrichs AS, Karolchik D, Cline MS, et al. The UCSC genome browser database: update 2011. Nucleic Acids Res. 2011;39 Database issue:D876–82. https://doi.org/10.1093/nar/gkq963.
2. Severin J, Lizio M, Harshbarger J, Kawaji H, Daub CO, Hayashizaki Y, et al. Interactive visualization and analysis of large-scale sequencing datasets using ZENBU. Nat Biotechnol. 2014;32:217–9.
3. Thorvaldsdóttir H, Robinson JT, Mesirov JP. Integrative genomics viewer (IGV): high-performance genomics data visualization and exploration. Brief Bioinform. 2013;14:178–92.
4. Freese NH, Norris DC, Loraine AE. Integrated genome browser: visual analytics platform for genomics. Bioinformatics. 2016;32:2089–95. https://doi.org/10.1093/bioinformatics/btw069.
5. Beraldi D. ASCIIGenome: a command line genome browser for console terminals. Bioinformatics. 2017;33:1568–9.
6. Valeev T, Yevshin I, Kolpakov F. BioUML Genome Browser. Virtual Biol. 2013;1:e8.
7. Hahne F, Ivanek R. Visualizing genomic data using Gviz and bioconductor. In: Methods in molecular biology; 2016. p. 335–51.
8. Yin T, Cook D, Lawrence M. Ggbio: an R package for extending the grammar of graphics for genomic data. Genome Biol. 2012;13:R77.
9. Georgiou G, van Heeringen SJ. Fluff: exploratory analysis and visualization of high-throughput sequencing data. PeerJ. 2016;4:e2209.
10. Shen L, Shao N, Liu X, Nestler E. Ngs.Plot: quick mining and visualization of next-generation sequencing data by integrating genomic databases. BMC Genomics. 2014;15:284.
11. Cock PJA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. Bioinformatics. 2009;25:1422–3.
12. Dale RK, Pedersen BS, Quinlan AR. Pybedtools: a flexible Python library for manipulating genomic datasets and annotations. Bioinformatics. 2011;27:3423–4.
13. Ramírez F, Dündar F, Diehl S, Grüning BA, Manke T. DeepTools: a flexible platform for exploring deep-sequencing data. Nucleic Acids Res. 2014:W187–91.
14. Quinlan AR. BEDTools: the Swiss-Army tool for genome feature analysis. Curr Protoc Bioinforma. 2014;8:11.12.1–34.
15. Meyer LR, Zweig AS, Hinrichs AS, Karolchik D, Kuhn RM, Wong M, et al. The UCSC genome browser database: extensions and updates 2013. Nucleic Acids Res. 2013;41(Database issue):D64–9.

16. Makeeva DS, Lando AS, Anisimova AS, Egorov AA, Logacheva MD, Penin AA, et al. Translatome and transcriptome analysis of TMA20 (MCT-1) and TMA64 (eIF2D) knockout yeast strains: Data Br. 2019;23:103701. https://doi.org/10.1016/j.dib.2019.103701.
17. Albert FW, Muzzey D, Weissman JS, Kruglyak L. Genetic influences on translation in yeast. PLoS Genet. 2014;10:e1004692.
18. Michel AM, Kiniry SJ, O'Connor PBF, Mullan JP, Baranov PV. GWIPS-viz: 2018 update. Nucleic Acids Res. 2018;46:D823–30.
19. Zerbino DR, Achuthan P, Akanni W, Amode MR, Barrell D, Bhai J, et al. Ensembl 2018. Nucleic Acids Res. 2018;46:D754–61.
20. Castelo-Szekely V, Arpat AB, Janich P, Gatfield D. Translational contributions to tissue specificity in rhythmic and constitutive gene expression. Genome Biol. 2017;18:116.
21. Janich P, Arpat AB, Castelo-Szekely V, Lopes M, Gatfield D. Ribosome profiling reveals the rhythmic liver translatome and circadian clock regulation by upstream open reading frames. Genome Res. 2015;25:1848–59.
22. Michel AM, Fox G, M. Kiran A, De Bo C, O'Connor PBF, Heaphy SM, et al. GWIPS-viz: development of a ribo-seq genome browser. Nucleic Acids Res 2014;42:D859-D864.
23. Harrow J, Frankish A, Gonzalez JM, Tapanari E, Diekhans M, Kokocinski F, et al. GENCODE: the reference human genome annotation for the ENCODE project. Genome Res. 2012;22:1760–74.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

BMC