WEB TOOL

PPI layouts: BioJS components for the display of Protein-Protein Interactions [version 1; referees: 2 approved]

Gustavo A. Salazar, Ayton Meintjes, Nicola Mulder

Computational Biology Group, University of Cape Town, Cape Town, South Africa

v1 **First published:** 13 Feb 2014, **3**:50 (doi: 10.12688/f1000research.3-50.v1)
Latest published: 13 Feb 2014, **3**:50 (doi: 10.12688/f1000research.3-50.v1)

Abstract

Summary: We present two web-based components for the display of Protein-Protein Interaction networks using different self-organizing layout methods: force-directed and circular. These components conform to the BioJS standard and can be rendered in an HTML5-compliant browser without the need for third-party plugins. We provide examples of interaction networks and how the components can be used to visualize them, and refer to a more complex tool that uses these components.
Availability: http://github.com/biojs/biojs;
http://dx.doi.org/10.5281/zenodo.7753

BIOJS This article is included in the BioJS channel.

Open Peer Review

Referee Status: ✓ ✓

	Invited Referees 1	2
version 1 published 13 Feb 2014	✓ report	✓ report

1 **Laurent Gatto**, University of Cambridge UK

2 **Hagen Blankenburg**, European Academy Bozen/Bolzano (EURAC) Italy

Discuss this article

Comments (0)

Corresponding author: Gustavo A. Salazar (gsalazar@cs.uct.ac.za)

How to cite this article: Salazar GA, Meintjes A and Mulder N. ***PPI layouts*: BioJS components for the display of Protein-Protein Interactions [version 1; referees: 2 approved]** *F1000Research* 2014, **3**:50 (doi: 10.12688/f1000research.3-50.v1)

Grant information: This project was funded by the Computational Biology Group at the University of Cape Town.
The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: No competing interests were disclosed.

First published: 13 Feb 2014, **3**:50 (doi: 10.12688/f1000research.3-50.v1)

Introduction

"Proteins are the building blocks of life". This is probably the most commonly used analogy in the molecular biology world. Despite the danger of falling into a *cliché*, we will elaborate on this. A brick is not a building, not a room, not even a wall; in the same way, proteins rarely perform functions in isolation, it is their interactions which form the complex networks that are responsible for almost all cellular processes.

The number of reported Protein-Protein Interaction (PPI) networks has grown considerably, partly due to advances in high-throughput experimentation and partly due to the new predictions that result from these empirical data.

Various strategies have been used to store this data for ease of retrieval and analysis of PPI networks[1]. For instance, IntAct[2] is an open source repository for molecular interactions reported in the literature or directly deposited by the user; the STRING database[3] includes physical and functional associations derived from genomic context, high-throughput experiments, coexpression analysis or the literature. Other PPI sources are described in:[4–9].

The volume of data now contained in PPI repositories has made the analysis and understanding thereof a challenge that can be enriched by the use of visualization techniques. PPI networks have been found to follow a behaviour that in graph theory is referred to as *scale-free*, which uses a few highly connected nodes and many nodes with few interactors. This feature allows a user to predefine visualization strategies to highlight certain characteristics. This approach has been used for multiple tools, some are stand-alone applications (e.g. Cytoscape[10]), Java applications that are web available via Applets (e.g.VisANT[11]), or those developed on Flash (e.g. STITCH[4]). Details about other tools, comparisons between them, highlighted features and supported layouts can be found in any of the following reviews:[12–14].

The web technologies covered under the umbrella term of HTML5 allow the display of some of those layouts natively on the web in a fully interactive way. We have used these technologies to implement two of the most popular layouts for PPI network visualization: Force-directed and Circle.

The components were developed following the BioJS[15] standard and are freely available at its registry: http://goo.gl/064ChR, http://goo.gl/RGlRLB.

BioJS components

We have implemented the standards defined by BioJS in the development of two components for the visualization of PPI networks. The use of BioJS gives visibility to the components so that they can be discovered by third parties interested in the visualization of protein networks on the web.

We consider the publication via the BioJS registry beneficial for both the potential user and the developer of the component. The user of the component will save time by adopting a component instead of developing it from scratch. The particular needs of the user may require the implementation of specifics which could contribute to the further development of the component in a communal effort. All this is in the nature of open source projects, but without visibility it would hardly ever happen.

Both PPI network visualizer components are implemented as wrappers of layouts included in the Data Driven Document (D3) library[16], a popular JavaScript library for the processing and visualization of data. Besides making these layouts follow the BioJS structure, most of the effort has been directed at providing methods that are closer to the PPI vocabulary (e.g. *addProtein()*,*addInteraction()*). Both components follow the same API (Application Program Interface) and therefore any script developed to display on one layout can be used on the other.

Force-directed layout

The D3 library provides an implementation of the Barnes-Hut simulation in order to efficiently create a self-organized network. This algorithm performs better than others because it aggregates the forces of close nodes, to avoid individual calculation on the forces whenever possible. Its performance facilitates a smoother transition between the execution steps of the algorithm, creating an appealing visual effect of live self organization. We have used this generic network layout available in D3 to provide a set of methods and events to represent PPI networks.

Figure 1 displays a subset of proteins from the organism *Mycobacterium tuberculosis*, that interact with furB. It includes all the interactions between these proteins with each protein color coded by reported functional class. FurB is a zinc uptake regulation protein involved in repression of many genes involved in zinc homeostasis[17].

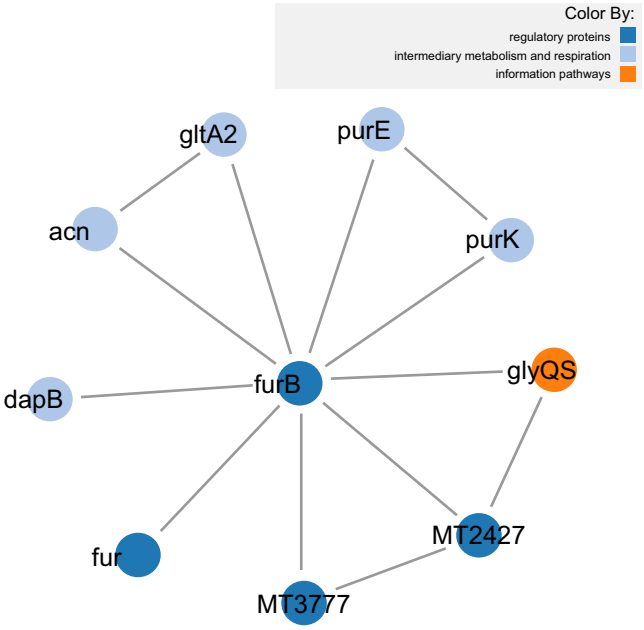

Figure 1. *Mycobacterium tuberculosis* proteins that interact with *furB* and the interactions between them (using force-directed layout).

FurB interacts with a number of different genes, with related functions, including the transcriptional regulator FurA, which represses transcription of katG, the catalase-peroxidase gene as well as itself, and the HTH-type transcriptional repressor SmtB, also involved in zinc homeostasis. FurB also interacts with the cAMP receptor protein, a global transcriptional regulator, although no link to metal ions has been annotated for this protein, a disruption in the protein results in slow growth. Additional interactions for FurB are with enzymes involved in amino acid biosynthesis (dapB, acn) or purine metabolism (purE, purK). One thing in common between these proteins appears to be their relation to growth.

BioJS provides instructions regarding the installation of each component including their required dependencies, style files and snippets of JavaScript code to be inserted into the web page where the component will be embedded. For example, installation instructions for the force-directed layout can be found on the *Installation* tab of the component page in the BioJS Registry (http://goo.gl/064ChR).

The full script used to generate Figure 1 is available as a jsFiddle (a web resource for the online edition and display of snippets of JavaScript code) at this URL: http://jsfiddle.net/Bvh6k/1/. Below is a description of the main components of that code:

A developer should start by creating an instance of the component:

```
var instance = new Biojs.InteractionsD3({
    target: "example"
});
```

All the proteins in the graphic should then be added. The following example shows how to add the protein with UniProt id O05839 (http://www.uniprot.org/uniprot/O05839). Note how the organism to which it belongs is included, along with which feature should be used for the label:

```
instance.addProtein({
       id: 'O05839', organism: "MTB",
       gene_name: "furB",
       typeLegend: "gene_name" });
```

In the same way, the interactions should be added to the graphic, ensuring that the interactions are between proteins that have already been added. For instance:

```
instance.addInteraction(
        "O05839", "P0A582",
        {id: "P0A582_P0A582"});
```

Once all the proteins and interactions have been declared, the graphic can be restarted so it reflects the additions:

```
instance.restart ();
```

The example also includes some instructions for coloring and format. For more details on those methods see the component on the BioJS registry: http://goo.gl/064ChR.

Circle layout

As mentioned previously, both components have the same API with the exception of methods that help to control the force layout of the first component which do not apply to the Circle layout. Thanks to this, the script to generate a PPI visualization of the same network is very similar in both. In order to demonstrate this we have created a second script that visualizes the same network as in Figure 1 but now on a Circle layout (Figure 2). The only difference between both scripts is the declaration of the object. Instead of using the object *Biojs.InteractionsD3* it uses *Biojs.InteractionsBundleD3* and rest of the script is exactly the same. This can be seen in the fiddle: http://jsfiddle.net/J4CE7/1/.

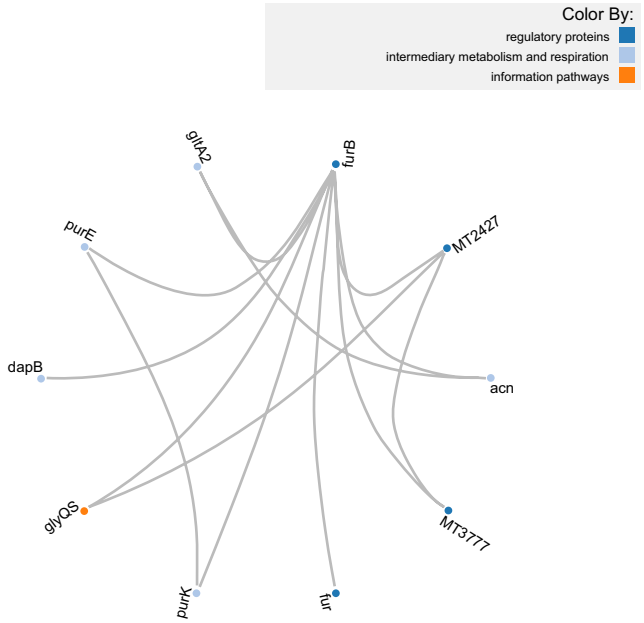

Figure 2. *Mycobacterium tuberculosis* proteins that interact with *furB* and the interactions between them (using Circle layout).

Use Case: PINV

We have used these two layouts and a few other BioJS components to create a web native application that allows the querying and visual exploration of PPI networks. It is called PINV: Protein Interactions Network Visualizer, and is freely available at http://biosual.cbio.uct.ac.za/pinv.html.

With PINV, we are looking to offer an alternative for the visualization of PPI data that takes advantage of the web to provide collaborative tools.

Currently PINV can display the PPI networks in three ways: the two layouts discussed above and through a table of the raw data. The

addition of other layouts in the future is not complicated, thanks to the standards defined by BioJS and the adoption of the same API.

The same example used for Figure 1 and Figure 2 is available on PINV under this link: http://goo.gl/r4XpOS.

Conclusions

One of the limitations of current web-based methods for PPI visualization is that they require third-party browser plugins. We demonstrate that the HTML5 standard provides enough functionality to render these networks in compliant browsers, without the need to install additional browser components. Adoption of the BioJS standard has the advantage of greater exposure to potential users, and an established set of features such as testing and documentation. Finally, the components abstract much of the details of rendering complex scalable vector graphics (SVG) behind an intuitive API, allowing users to focus on building richer applications.

Software availability

Zenodo: BioJS components for the display of Protein-Protein Interactions, doi: 10.5281/zenodo.7753[18].

GitHub: BioJs, http://github.com/biojs/biojs.

Author contributions

Critical revision of the manuscript for important intellectual input: GS, AM and NM. Supervision: NM. Study concept: GS, AM and NM. Software development: GS. Drafting of the manuscript: GS and AM. All authors have read and approved the final manuscript.

Competing interests

No competing interests were disclosed.

Grant information

This project was funded by the Computational Biology Group at the University of Cape Town.

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Acknowledgements

We want to thank J. Holtzhausen for her contribution to the editorial matters of this article.

References

1. Mathivanan S, Periaswamy B, Gandhi TK, *et al.*: **An evaluation of human protein-protein interaction data in the public domain.** *BMC Bioinformatics.* 2006; **7**(Suppl 5): S19.
PubMed Abstract | Publisher Full Text | Free Full Text
2. Kerrien S, Aranda B, Breuza L, *et al.*: **The IntAct molecular interaction database in 2012.** *Nucleic Acids Res.* 2012; **40**(Database issue): D841–D846.
PubMed Abstract | Publisher Full Text | Free Full Text
3. Franceschini A, Szklarczyk D, Frankild S, *et al.*: **String v9.1: protein-protein interaction networks, with increased coverage and integration.** *Nucleic Acids Res.* 2013; **41**(Database issue): D808–D815.
PubMed Abstract | Publisher Full Text | Free Full Text
4. Kuhn M, von Mering C, Campillos M, *et al.*: **STITCH: interaction networks of chemicals and proteins.** *Nucleic Acids Res.* 2008; **36**(Database issue): D684–D688.
PubMed Abstract | Publisher Full Text | Free Full Text
5. Brown KR, Jurisica I: **Unequal evolutionary conservation of human protein interactions in interologous networks.** *Genome Biol.* 2007; **8**(5): R95.
PubMed Abstract | Publisher Full Text | Free Full Text
6. Keshava Prasad TS, Goel R, Kandasamy K, *et al.*: **Human protein reference database—2009 update.** *Nucleic Acids Res.* 2009; **37**(Database issue): D767–D772.
PubMed Abstract | Publisher Full Text | Free Full Text
7. Mathivanan S, Ahmed M, Ahn NG, *et al.*: **Human proteinpedia enables sharing of human protein data.** *Nat Biotechnol.* 2008; **26**(2): 164–167.
PubMed Abstract | Publisher Full Text
8. Licata L, Briganti L, Peluso D, *et al.*: **Mint, the molecular interaction database: 2012 update.** *Nucleic Acids Res.* 2012; **40**(Database issue): D857–D861.
PubMed Abstract | Publisher Full Text | Free Full Text
9. Chen JY, Mamidipalli S, Huan T: **HAPPI: an online database of comprehensive human annotated and predicted protein interactions.** *BMC Genomics.* 2009; **10**(Suppl 1): S16.
PubMed Abstract | Publisher Full Text | Free Full Text
10. Shannon P, Markiel A, Ozier O *et al.*: **Cytoscape: A software environment for integrated models of biomolecular interaction networks.** *Genome Res.* 2003; **13**(11): 2498–2504.
PubMed Abstract | Publisher Full Text | Free Full Text
11. Hu Z, Chang YC, Wang Y, *et al.*: **VISANT 4.0: Integrative network platform to connect genes, drugs, diseases and therapies.** *Nucleic Acids Res.* 2013; **41**(Web Server issue): W225–W231.
PubMed Abstract | Publisher Full Text | Free Full Text
12. Suderman M, Hallett M: **Tools for visually exploring biological networks.** *Bioinformatics.* 2007; **23**(20): 2651–2659.
PubMed Abstract | Publisher Full Text
13. Gehlenborg N, O'Donoghue SI, Baliga NS, *et al.*: **Visualization of omics data for systems biology.** *Nat Methods.* 2010; **7**(3 Suppl): S56–68.
PubMed Abstract | Publisher Full Text
14. Agapito G, Guzzi PH, Cannataro M: **Visualization of protein interaction networks: problems and solutions.** *BMC Bioinformatics.* 2013; **14**(Suppl 1): S1.
PubMed Abstract | Publisher Full Text | Free Full Text
15. Gómez J, García LJ, Salazar GA, *et al.*: **BioJS: An open source javascript framework for biological data visualization.** *Bioinformatics.* 2013; **29**(8): 1103–4.
PubMed Abstract | Publisher Full Text | Free Full Text
16. Bostock M, Ogievetsky V, Heer J: **D3; data-driven documents.** *IEEE Trans Vis Comput Graph.* 2011; **17**(12): 2301–2309.
PubMed Abstract | Publisher Full Text
17. Milano A, Branzoni M, Canneva F, *et al.*: **The mycobacterium tuberculosis rv2358–furb operon is induced by zinc.** *Res Microbiol.* 2004; **155**(3): 192–200.
PubMed Abstract | Publisher Full Text
18. Salazar GA, Meintjes A, Mulder N: **Biojs components for the display of protein-protein interactions.** *Zenodo.* 2014.
Data Source

Open Peer Review

Current Referee Status: ✔ ✔

Version 1

Referee Report 28 March 2014

doi:10.5256/f1000research.3677.r4104

✔ **Hagen Blankenburg**
Center for Biomedicine, European Academy Bozen/Bolzano (EURAC), Bolzano, Italy

The two JavaScript components presented in this article facilitate the visualization of interaction networks in HTML5-enabled web browsers without requiring any third-party plugins like Flash. As they follow the BioJS standard, they are listed in the BioJS registry and should be interoperable with other components.

The manuscript is comprehensibly written and, despite the length limitations, successfully introduces both tools and their basic usage. The availability of the source code and particularly the jsFiddle live demo setup enable the interested reader to explore the software without complicated installation procedures.

In the following, I will list a few comments that might help to assure that the tools unlock their full potential. Note that while there is an overlap to the review by Laurent Gatto, I will repeat certain aspects for the sake of completeness.

Interoperability with other BioJS components / PINV:
The ability to show additional information about protein interactors (e.g., database identifiers, cross-references) and their interactions (e.g. links to source databases and publications, confidence scores) is what distinguishes a truly interactive network display from a nice visualization. In contrast to the two basic examples currently provided, PINV nicely demonstrates that this can be accomplished. Given that BioJS is essentially about modularity and re-usability of code, it would be great if the authors could provide more detailed examples that illustrate how their layout components can be combined with other BioJS components, e.g., to visualize additional information in popups. If manuscript space permits, additional commented source code files or a jsFiddle setup would be great and would further ease the application of the tools.

***M. tuberculosis* example:**
As part of the BioJS collection, the manuscript generally seems to be written for computer scientists / developers rather than biologists. As such, the paragraph describing the *M. tuberculosis* interaction network example is too extensive and does not fit the focus of the remaining article. This example could be shortened and moved into the corresponding figure legends, making room for the aforementioned, more thorough description of other parts.

Cytoscape.js:
While the authors mention similar software like Cytoscape and VisANT, the HTML5 library Cytoscape.js (http://cytoscape.github.io/cytoscape.js/) and its predecessor Cytoscape Web (http://cytoscapeweb.cytoscape.org/) are omitted, although they are likely the closest competitors.

Source code examples:
From the manuscript or the source code examples it is not obvious why the UniProtKB accession numbers and the organism information is provided, as neither is used or shown in the examples. It should be clarified, if tools require this information or if this might be used for connecting with other BioJS components.

Title:
Looking only at the title and the abstract, it appears that “PPI layouts” is the name of the BioJS component that is described in the article. As this is not the case and the name “PPI layouts” is never used again, it could be removed from the title to prevent confusion.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Competing Interests: No competing interests were disclosed.

Referee Report 10 March 2014

doi:10.5256/f1000research.3677.r3688

Laurent Gatto
Cambridge Centre for Proteomics, University of Cambridge, Cambridge, UK

The manuscript *PPI layouts:BioJS components for the display of Protein-Protein Interactions* by Salazar and colleagues briefly illustrate the BioJS component for the interactive html5-based visualisation of PPI.

The manuscript is easy to read and provides a quick, yet useful introduction to the BioJS PPI visualisation tools. The component described in the manuscript and the BioJS project in general are a welcome initiative. The PPI interaction tools was straightforward to assess thanks to support for the online edition, simple implementation, and its distribution to the bioinformatics community under an open access license.

Below, I have a few suggestions:

BioJS component section
In the first paragraph the sentence "The use of BioJS gives visibility to the components so that they can be [discovered] by third parties...". Do the authors mean that the individual component will benefit from the BioJS visibility, or is there a specific discovery mechanism?

***M. tuberculosis* example network**
While I appreciate that it is not easy to find the right balance between a simple yet biologically relevant illustration of the tool, I believe that the paragraph that describes the interaction network could be improved. For example, some proteins are named and described while others are not.

While reading the description of the code generating the network in Figure 1 and 2 and using the jsFiddle instance, several questions arose that could be addressed in the manuscript.

- the authors provide a UniProt identifier. Could the nodes be links to the relevant protein UniProt pages? Could other identifiers be used and how would the links to other arbitrary pages be defined?
- What is the use of the species name in the protein instance definition? Is it mandatory? Are there examples of PPIs with multiple species?
- Graph annotation. Is it possible to use different symbols for the nodes? How to annotate the edges (through colours and/or labels)? Any comments on the use of node colours as a means to illustrate additional meta-data: are specific colour palettes (for continuous data for example) or transparency available?

PINV
The PINV application is a step further than the examples described in the manuscript, that possibly deserves a bit more description in the text and could guide the reader on how to build up from simple cases to more complex applications.

Before Figure 2
Instead of using the object Biojs.InterationsD3 it uses Biojs.InteractionsBundleD3 and [the] rest of the script is exactly the same.

Compliant browsers
The authors repeat the requirement for compliant browsers. I was wondering if examples of compliant browsers could be provided to guide users new to html5 and issues related to browser compatibility.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Competing Interests: No competing interests were disclosed.