SOFTWARE TOOL ARTICLE

Epiviz Web Components: reusable and extensible component library to visualize functional genomic datasets [version 1; peer review: 1 approved, 2 approved with reservations]

Jayaram Kancherla[1,2], Alexander Zhang[3], Brian Gottfried[3], Hector Corrada Bravo[1-3]

[1]Center for Bioinformatics and Computational Biology, University of Maryland, College Park, College Park, Maryland, 20742, USA
[2]University of Maryland Institute of Advanced Computer Studies, University of Maryland, College Park, College Park, Maryland, 20742, USA
[3]Department of Computer Science, University of Maryland, College Park, College Park, Maryland, 20742, USA

v1 **First published:** 17 Jul 2018, **7**:1096
https://doi.org/10.12688/f1000research.15433.1
Latest published: 17 Jul 2018, **7**:1096
https://doi.org/10.12688/f1000research.15433.1

Abstract

Interactive and integrative data visualization tools and libraries are integral to exploration and analysis of genomic data. Web based genome browsers allow integrative data exploration of a large number of data sets for a specific region in the genome. Currently available web-based genome browsers are developed for specific use cases and datasets, therefore integration and extensibility of the visualizations and the underlying libraries from these tools is a challenging task. Genomic data visualization and software libraries that enable bioinformatic researchers and developers to implement customized genomic data viewers and data analyses for their application are much needed.

Using recent advances in core web platform APIs and technologies including Web Components, we developed the Epiviz Component Library, a reusable and extensible data visualization library and application framework for genomic data. Epiviz Components can be integrated with most JavaScript libraries and frameworks designed for HTML. To demonstrate the ease of integration with other frameworks, we developed an R/Bioconductor *epivizrChart* package, that provides interactive, shareable and reproducible visualizations of genomic data objects in R, Shiny and also create standalone HTML documents. The component library is modular by design, reusable and natively extensible and therefore simplifies the process of managing and developing bioinformatic applications.

Keywords

genomics, visualization, epigenetics, bioinformatics, web components

Open Peer Review

Reviewer Status ? ? ✓

	Invited Reviewers		
	1	**2**	**3**
version 1 17 Jul 2018	? report	? report	✓ report

1. **Bianca H. Habermann**, Aix-Marseille Univ, Aix-Marseille Univ, CNRS, IBDM UMR 7288, Marseille, France
2. **Ian Holmes**, University of California, Berkeley, Berkeley, USA
3. **Zhaohui Steve Qin**, Emory University, Atlanta, USA

Any reports and responses or comments on the article can be found at the end of the article.

This article is included in the Bioconductor gateway.

Corresponding author: Hector Corrada Bravo (hcorrada@umiacs.umd.edu)

Author roles: Kancherla J: Software, Supervision, Writing – Original Draft Preparation; **Zhang A**: Software; **Gottfried B**: Software; **Bravo HC**: Conceptualization, Supervision, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: This work was supported by the grant funded by the National Institutes of Health [R01GM114267].
The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

How to cite this article: Kancherla J, Zhang A, Gottfried B and Bravo HC. **Epiviz Web Components: reusable and extensible component library to visualize functional genomic datasets [version 1; peer review: 1 approved, 2 approved with reservations]** F1000Research 2018, **7**:1096 https://doi.org/10.12688/f1000research.15433.1

First published: 17 Jul 2018, **7**:1096 https://doi.org/10.12688/f1000research.15433.1

Introduction

The complex and diverse genomic data sets require flexible software libraries and tools to perform integrative data exploration and analyses. Web-based genome browsers and genomic data visualization tools like the UCSC Genome Browser[1] and the Integrated Genomics Viewer[2] are developed for specific use cases i.e., integrative data exploration of a large number of datasets for a region in the genome. Genomic exploration of data on these platforms is usually track-based, where the data is aligned to a reference genome and visualized as a line track. Since these tools are developed for specific use cases, integration and extensibility of these visualizations and libraries is a challenging task.

The Web as a platform has been used to serve static HTML documents traditionally. The implementation of HTML5 and the newer APIs made the Web more of a platform that supports rich and dynamic web applications. But HTML is still restrictive and limited to the tags/elements defined as part of the markup language and is not extensible. Various existing frameworks like Vue.js and React have introduced modular components, but components built for one framework do not work with another framework. Newer web platform APIs and technologies like Web Components introduced a standards-based component model that allows developers to create custom HTML elements that are natively extensible and reusable. Custom components work across modern web browsers and can be used along with most JavaScript libraries or frameworks designed for HTML. Web Components provide the ability to natively extend, import and encapsulate HTML elements. This makes the process of creating and managing web applications easier and a much smoother process. These components are modular, making the code cleaner and less expensive to maintain compared to JavaScript libraries and frameworks like BioJs[3].

We present the Epiviz Component Library, an open source reusable and extensible data visualization library and application framework for functional genomic data. Building upon the Web Component framework, we developed various HTML elements/tags as part of our design as shown in Figure 1. The *visualization components* (***epiviz-charts***) are the core of the library and render extensible and interactive track and feature-based charts. In addition to the chart library, we developed components for creating interactive genomic applications for different use cases and datasets. These include *app components* (***epiviz-navigation*** and ***epiviz-environment***) to coordinate interactions (linking data across visualizations to implement brushing and events) and manage layouts, *datasource components* including ***epiviz-data-source***

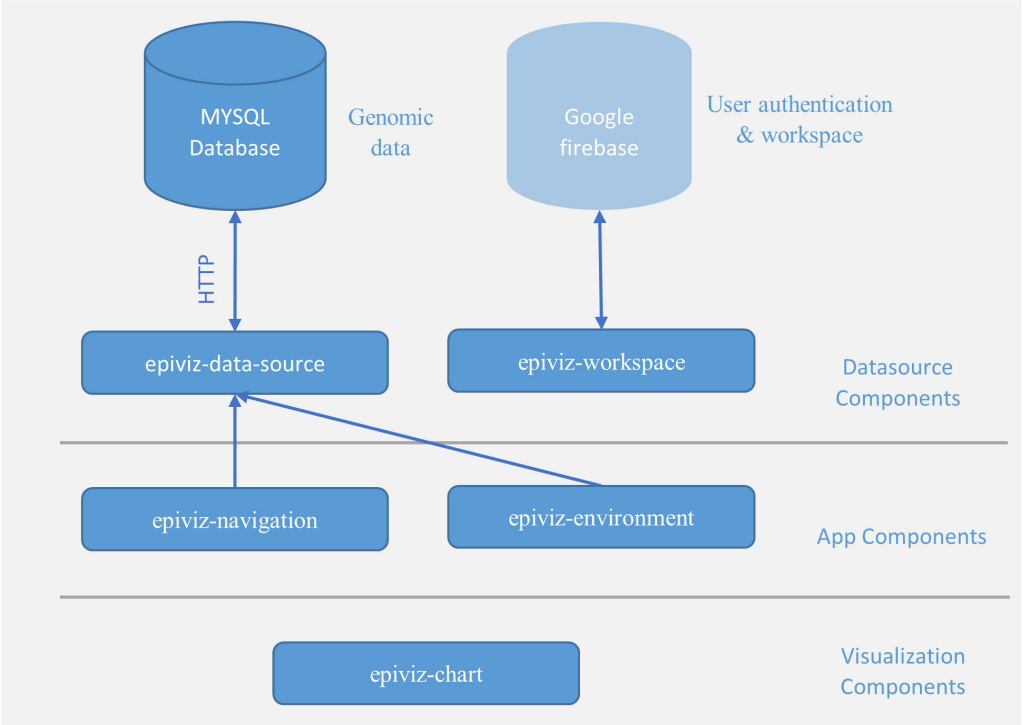

Figure 1. Overview of Epiviz web components architecture. The epiviz web components architecture is organized into three categories: 1) *visualization components* is a library of extensible and interactive D3JS based chart components specifically designed for genomic data; 2) *app components* are responsible for managing the layouts, events arising from genomic coordinate navigation, linking data across visualization components to implement brushing and coordinating data requests across multiple charts; 3) *datasource components* manage requests to web server or WebSocket connections using the *epiviz-data-source* component. *epiviz-workspace* handles user authentication and saves the state of the app to a google firebase instance allowing users to create shareable and reproducible visual analytics workspaces.

to manage data requests from a web server or WebSocket backend and ***epiviz-workspace*** for handling user authentication and to create shareable and reproducible visual analytical workspaces. The design of the component library is based on visualizations and features of the Epiviz[4] web application for visual exploration and analysis of functional genomic data.

Bioconductor[5] is an open source community that develops bioinformatics software tools and pipelines. Ease of developing integrative analyses and a framework for interactive visualizations is one of the core infrastructure needs of the Bioconductor community. Since the web components introduced in this paper can be easily embedded or integrated with any web-based application, the library reduces the effort to visualize and create applications for genomic datasets encapsulated in Bioconductor infrastructure data representations. We developed an R/Bioconductor package, ***epivizrChart***[6] to visualize genomic data objects within HTML documents created using RMarkdown. We also integrated our components with Shiny[7], a web application framework for R to interactively visualize functional genomic data.

Methods

Implementation

Visualization components. ***epiviz-chart*** components are a collection of reusable and extensible data visualizations specifically designed for genomic data. The library provides multiple data visualizations for both location (visualizing data along the genome genes tracks (*epiviz-genes-track*) or line tracks (*epiviz-line-track*)) and feature based data (visualizing quantitative measurements like gene expression with scatterplots (*epiviz-scatter-plot*) and heatmaps (*epiviz-heatmap-plot*)). We use D3.js[8] (version: 3.5.17) JavaScript library to render customizable and interactive charts.

An ***epiviz-chart*** component requires two attributes to render a visualization on the page 1) *data* attribute - a JSON (JavaScript Object Notation) representation of genomic data. 2) *dimensions* (or columns) from the *data* attribute to visualize. Figure 2 demonstrates the ease of embedding or adding an ***epiviz-chart*** to a HTML document or web application.

epiviz-chart components are reactive components that render the visualization only after the *json-data* attribute is initialized on the element. Any change to the *json-data* attribute triggers an event to revisualize the chart. Visualizations are extensible and easily customizable to define various settings and colors. To demonstrate the extensibility of the components, we created a component *epiviz-genes-table* extending *epiviz-genes-track* and displays a table of all the genes in the current genomic region (Supplementary File 1).

In addition to visualizing data, chart elements can also perform client-side operations on data sets/measurements. For example, if an *epiviz-line-track* is visualizing methylation data from multiple samples (tumor and normal), samples can be aggregated using a metric (mean, min, max, etc.) to visualize the difference in methylation between normal and tumor samples. Similarly, *epiviz-heatmap-plot* interactively and dynamically clusters data and renders the clustered dendrogram. Settings are available to change clustering type and the distance metrics.

Chart components provides performance optimizations for visualizing large amounts of data by precomputing and grouping overlapping data points to a single visual object on the chart. This minimizes the number of overlapping data points to visualize and reduce rendering time of charts.

Data model

The *json-data* attribute on an epiviz element is a JSON object that represents genomic data in a columnar format as shown in Figure 2. The required *keys* in the JSON are *chr, start*, *end* and *data* columns to visualize. Developers can also extend the *epiviz-data-behavior* element to implement custom data parsers and formats.

Linked data selection/brushing

Chart components implement a linked data selection/highlighting (brushing) feature, to provide a quick overview and visually link the highlighted genomic region across all visualizations and datasets. The linking happens on the client side by finding positions that overlap with the highlighted region. In feature-based visualizations, for example in scatter plots and heatmaps, the visual objects on the chart are aggregated and mapped to multiple data objects across genomic regions. This mapping allows for implementing brushing and propagating events to other charts when using plots. In track-based visualization, events for brushing and selection are propagated based on the region (*chr, start* and *end*) in the chart.

Another essential part of the epiviz design is that data and plots are separated. Users can visualize multiple charts from the same data object without having to replicate the data. This way data queries are made by the data object and not per chart, which leads to a more responsive design of the system.

epiviz-chart components are simple user interface (UI) elements. They cannot make data requests or can directly interact with other ***epiviz-**** elements on the page. Chart elements create hover events that propagate up the document object model (DOM) hierarchy. To build interactive web applications or to coordinate interactions by linking data across charts, implement brushing and manage data requests across chart elements, we encapsulate charts inside app components.

App components. ***epiviz-app*** components are abstract components that 1) Manage layouts of multiple visualizations, 2) Coordinate interactions across charts by genomic position to implement brushing, and 3) Manage data requests.

There are two different types of ***epiviz-app*** elements -

epiviz-environment elements are not linked to a specific genomic region. If a genomic region (*chr*, *start* & *end* attributes) is not initialized on the element, charts visualize the entire data set genome-wide. This helps identify patterns or interesting regions in the dataset and then investigate specific regions of interest.

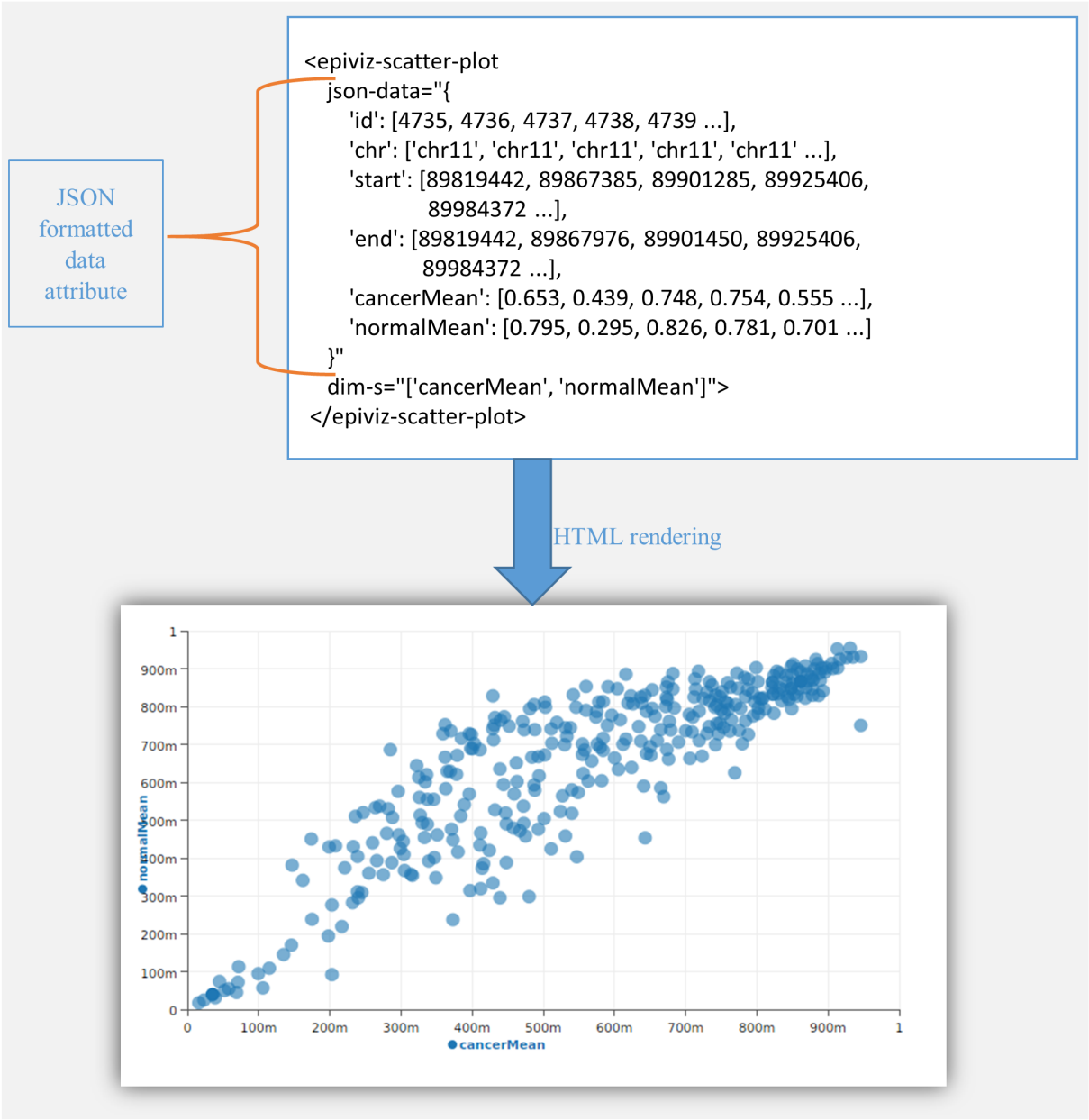

Figure 2. Example using components in HTML page. Epiviz components can be inserted in any HTML page using tags defined by the component library (e.g., *epiviz-json-scatter-plot* in this example). Data is supplied to the chart via the *json-data* attribute of the HTML tag. In this example, we show a sample JSON object representing genomic data. In this figure, we are only showing the first 5 data points although the plot renders more visual objects. When used in conjunction with *epiviz-data-source* components, data can be queried from a web server or via a WebSocket connection through a corresponding assignment of the *json-data* attribute. Adding the epiviz element to the HTML page renders the interactive scatter plot.

epiviz-navigation is a specific instance of ***epiviz-environment*** with genomic region linked to the element using the *chr*, *start* and *end* attributes. Navigation elements provide UI functionality to search for a gene/microarray probe (since we serve data from the Gene Expression Barcode project[9]) or update the location to a specific region of interest. Figure 3 (bottom) shows a navigation element with various charts when expanded. The top header bar contains functionality to navigate left/right and zoom in/out around the current genomic location. Navigation elements implement the usual genome browser interactions (pan, zoom, location input and gene name search). The chromosome location text box identifies the current location of the navigation element and can be updated to change the genomic region. Hovering over the chromosome location sends a brushing event to highlight this region across other charts encapsulated within the component. Navigation elements can be collapsed (as shown in the top panel)

Figure 3. Overview of the Epiviz2 web application for Epigenome Roadmap data. In this workspace, we explore data from the Epigenome Roadmap project in two genomic regions simultaneously (*Epiviz Navigation* components) along with a genome-wide scatterplot of gene expression (top left). The *environment* element is not constrained to a specific genomic region, and hence charts included within them visualize entire datasets. In this example, the scatter plot in the top left shows RNA-seq data for esophagus and colon tissues across the entire genome. *EpivizNavigation* components, on the other hand, are constrained to specific genomic regions. Given genomic regions or genes of interest in the dataset to further investigate, multiple navigation elements, each corresponding to distinct genomic regions can be added to the workspace. In this example, the navigation element at the bottom of the page visualizes (in order from top to bottom): 1) a genes track showing gene location span and strand, 2) a stacked-blocks track of ChIP-seq peaks in esophagus and colon across two different histone markers (H3K27me3 and H3k9me3), and 3) a line track that visualizes the fold change signal data for the same ChIP-seq data. The line track shows that the region around the gene "ATP6V1C1" shows a peak for H3K27me3 in ***Esophagus*** but not in ***Colon***. The stacked blocks track compares the peak regions with other histone markers (H3K9me3). We can also investigate this region further by exploring methylation and gene expression data from these tissues by adding a navigation element (top right). The component library provides and interactive and integrative environment for genomic data exploration. This example workspace can be accessed at http://epiviz2.cbcb.umd.edu/#/epiviz-C7O4UmIb.

to allow users to flexibly focus on specific genomic regions of interest while providing an overview of other regions of interest. When collapsed, navigation components show an ideogram of the corresponding chromosome with an indication of the specific genomic region encompassed within the components (yellow rectangle). No data requests are made from charts within collapsed navigation components.

App elements coordinate events across charts, i.e., when a chart element is highlighted, an event is propagated to all other charts in the workspace (including those visualizing genome-wide data). App elements also manage layouts for positioning and resizing chart elements. The default grid layout splits the available width into six equal columns. When charts are added to a workspace, track-based charts extend across all the six columns

but plot-based chart elements only span across two columns. App components have the functionality to navigate the genome and add new visualizations. Adding a new visualization opens a measurement browser, a UI interface that allows filtering and selection of measurements across different data sets.

App components can also detect if the application or page has an active web server or WebSocket connection initialized using the datasource components. If the page has no active datasource component, interactive features that generate data requests (for example – navigating to a new genomic region or adding new charts) are disabled.

Datasource components. ***epiviz-data-source*** component provides functionality for the epiviz app components to interact with an active web server or a WebSocket connection. Datasource components require the API endpoint (*provider-url*) attribute where the web server or WebSocket is located and the *provider-type* attribute that specifies if it's a web server or a WebSocket connection. When the user interacts with epiviz components, for example, adding a new visualization or navigating to a new genomic region, these interactions generate data requests that are eventually propagated and managed by the datasource elements.

WebServer data provider

We developed a Python Flask (version 0.12.4) based data provider that queries genomic data stored in MySQL database and responds to data requests. The data provider enables summarization where we bin small regions together and average the value for the measurements. We see a significant improvement in draw times of charts by summarizing data as discussed in the Benchmarks section of this paper. We also implemented data import functions for commonly used Bioconductor datatypes like *GenomicRanges*, *SummarizedExperiment,* etc., in our R/Bioconductor ***epivizrData*** package.

WebSocket data provider

The JavaScript data types that manage genomic data in epiviz components are designed similarly to Bioconductor data types. This enables easy integration and visualization of Bioconductor data objects using the visualization components. The R/Bioconductor ***epivizrChart*** package is an API to interactively visualize Bioconductor data objects. We discuss more about ***epivizrChart*** package in the Use case section of the paper.

Workspace component. ***epiviz-workspace*** component is built upon the Google Firebase infrastructure to manage user authentication, create shareable and reproducible visual analysis workspaces. Workspace components are easily reconfigurable and allow developers to customize this component to their firebase instance.

Operation

Web components are a set of standardized browser APIs still being implemented across various browsers. Web components implement the Shadow DOM feature, wherein the element defined by the component is rendered separately from the rest of the HTML document avoiding namespace collisions and is isolated to keep element styling and access private to the element. Web Components are natively supported in Chrome and Safari and are still in development in Mozilla Firefox and Microsoft Edge browser.

Epiviz Components are developed using the Google Polymer library. For browsers without native web component support, the Google Polymer library provides polyfill that helps developers use components seamlessly with little performance overhead. It uses a dynamic loader to lazy load polyfill libraries for missing implementations. Documentation on attributes and methods in epiviz components is available from GitHub. The component library can visualize data by adding the chart tag to a HTML page with the data attribute as shown in Figure 2.

The epivizrChart package requires R version 3.4.0 or higher and packages from Bioconductor version 3.6 or higher. The memory requirements for using the epivizrChart package depends on the size of the dataset. However, for most use cases, a standard laptop will handle most applications visualizing data using the component library and the epivizrChart package. To visualize a Bioconductor data object, supply the supported object to the *epivizChart()* function.

Use cases

Epiviz2 web application

Epiviz2 is an interactive and integrative genome browser that sends requests to a Python Flask data provider and a MySQL database. ***Epiviz2*** allows users to interactively explore and simultaneously visualize datasets across multiple genomic regions, a feature not available in most current genome browsers. The real advantage of the genome browser lies in the ability to visualize data from multiple regions of the genome or the entire dataset to identify genomic regions of interesting patterns or outliers. Users can then further explore and visualize annotations or measurements from other datasets in these regions to gain insights. Figure 3 illustrates this workflow of exploratory data analysis. The gene expression scatter plot is encapsulated inside the environment element and visualizes the entire dataset, whereas the navigation elements are linked to a specific genomic region. We also implemented a color by region for genome-wide scatterplots, where visual objects in the scatter plot will be colored with a different color specific to each of the genomic regions shown in navigation elements. Our instance of the ***Epiviz2*** application is hosted here.

The ***Epiviz2*** instance we host at the University of Maryland contains data from the NIH Roadmap Epigenomics[10] project. The NIH Roadmap Epigenomics Mapping Consortium leverages next-generation sequencing technologies to map DNA methylation, histone modifications, chromatin accessibility and small RNA transcripts in tissues selected to represent the normal counterparts of tissues and organ systems frequently involved in human disease. Our instance of the roadmap database contains DNA methylation, RNA seq, and histone modification (for markers: h3k9ac, h3k9me3, h3k27ac, h3k27me3) fold change and peak data for seven different tissue types – Breast Myoepithelial cells, Brain Hippocampus Middle, Lung, Liver, Sigmoid Colon, Pancreas and Esophagus. The corresponding data files are downloaded from Bioconductor's ***AnnotationHub*** repository and imported into the MySQL database using the functions available in the ***epivizrData*** package.

epivizrChart R/Bioconductor package

The Bioconductor open source software community creates bioinformatics workflows and pipelines to analyze and visualize genomic data sets. To support interactive visualization of Bioconductor data objects, we developed an R/Bioconductor package ***epivizrChart***, an API package to programmatically create and visualize genomic datasets using epiviz components without having to import data into a MySQL database. ***epivizrChart*** demonstrates the ease of integration with existing frameworks and can create interactive web pages or RMarkdown documents as shown in Figure 4. Integrating with a statistical and powerful state-of-the-art bioinformatics data analysis platform allows users to quickly explore, analyze and visualize genomic datasets with various packages available through Bioconductor.

online vs offline. Using the ***epivizrChart*** package in an *online* mode creates an active WebSocket server and allows interactions between the components and the R-session. In *online* mode, components make data requests using the WebSocket connection. In *offline* mode, data is attached to the components and a standalone HTML page is generated. This allows researchers to create interactive, shareable and reproducible visualization documents.

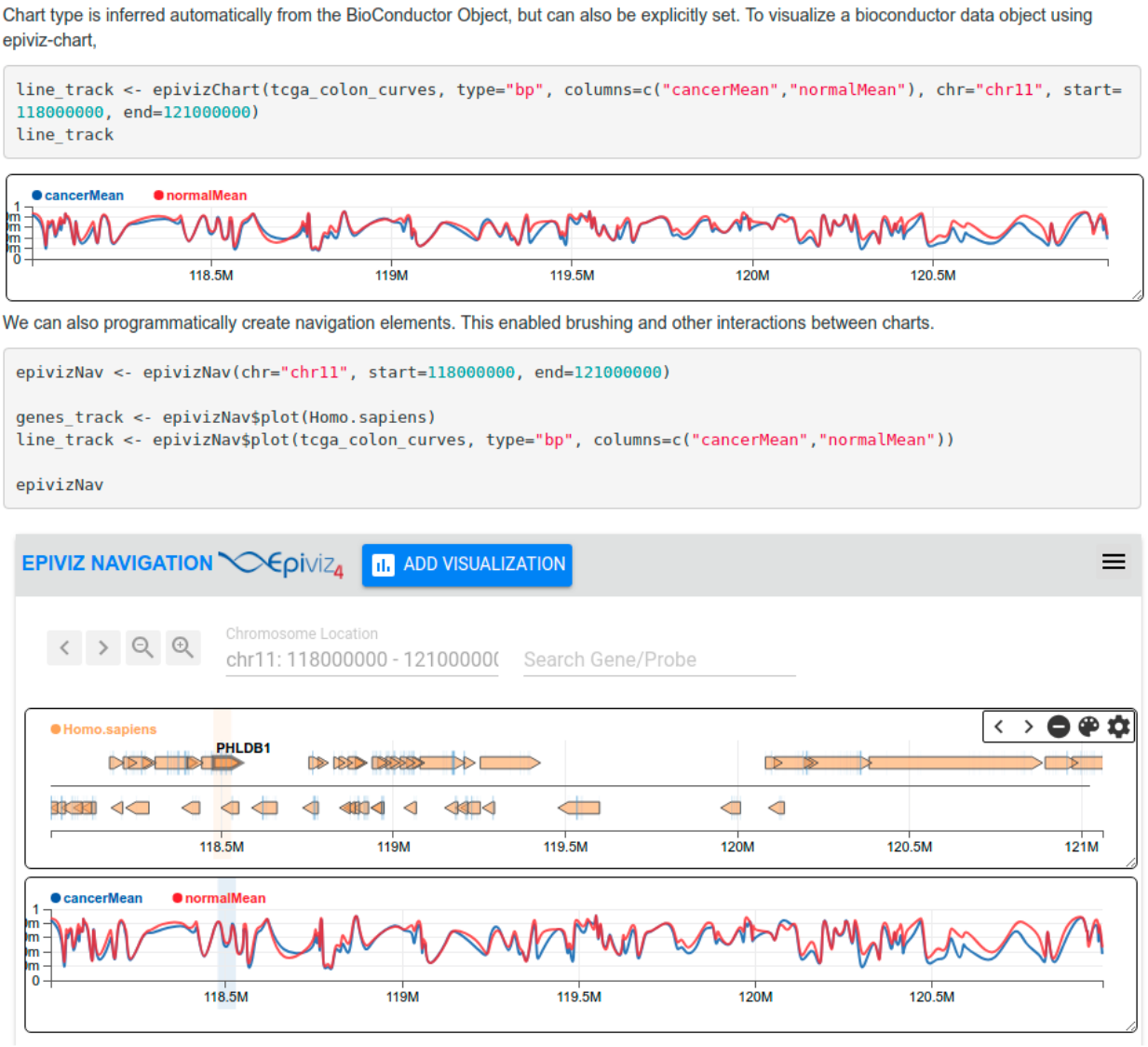

Figure 4. Interactive visualization of R/Bioconductor data objects using the *epivizrChart* package. This figure is part of an RMarkdown document and demonstrates the ease of integrating the visualizations from the Epiviz component library with existing frameworks. The *epivizChart* function infers the chart type based on the data object parameter. *"Homo.sapiens"* from the top panel is a UCSC Gene annotation object for human hg19 reference genome and is visualized by *epivizChart* as a genes track. t*cga_colon_curves* is a sample dataset from The Cancer Genome Atlas for colon tissue. This is a *GRanges* object and is visualized as a Line Track. The *epivizrChart* package can also programmatically create navigation elements. This enables interactions and brushing across the charts as shown in the bottom panel around the gene "PHLDB1". A vignette describing more examples and use cases is available in the package either on the GitHub repository or through Bioconductor (http://bioconductor.org/packages/release/bioc/html/epivizrChart.html).

Integration with Shiny. Shiny is a web application framework to create standalone web applications on a webpage or in an RMarkdown document. Since Shiny supports HTML, epiviz components can be embedded or integrated in Shiny applications or dashboards to interactively visualize genomic data. The vignette *IntegrationWithShiny.Rmd* in the *epivizrChart* package demonstrates 1) a simple application that integrates Shiny to visualize R/Bioconductor data objects using epiviz components 2) interactions with non-epiviz components in Shiny as shown in Figure 5.

Benchmarks

We use the google chrome headless *puppeteer* tool to measure request times and chart draw times to compare our web component implementation of the ***Epiviz2*** application (with Python-MySQL backend) to the current ***Epiviz***[4] application (with PHP-MySQL backend). We compare the times by varying the genomic region on the scatter plot component (***epiviz-scatter-plot***) across two different backend implementations: 1) Summarized responses (current implementation), where we bin the genomic region into 2000 intervals and average the data values for the measurement within each interval and, 2) Unsummarized responses (previous epiviz implementation), where the entire dataset for the region is sent back to the UI. When visualizing large genomic regions, data points tend to overlap on scatter plots and other visualizations because of pixel and chart size limitations on the page. Summarizing reduces the draw times in rendering charts because of fewer overlapping points as shown in Figure 6. However, the response times for data requests have not changed significantly because the computation time for summarization is usually similar to the time taken to transfer the entire dataset in the unscaled implementation. The scripts for the benchmarks are

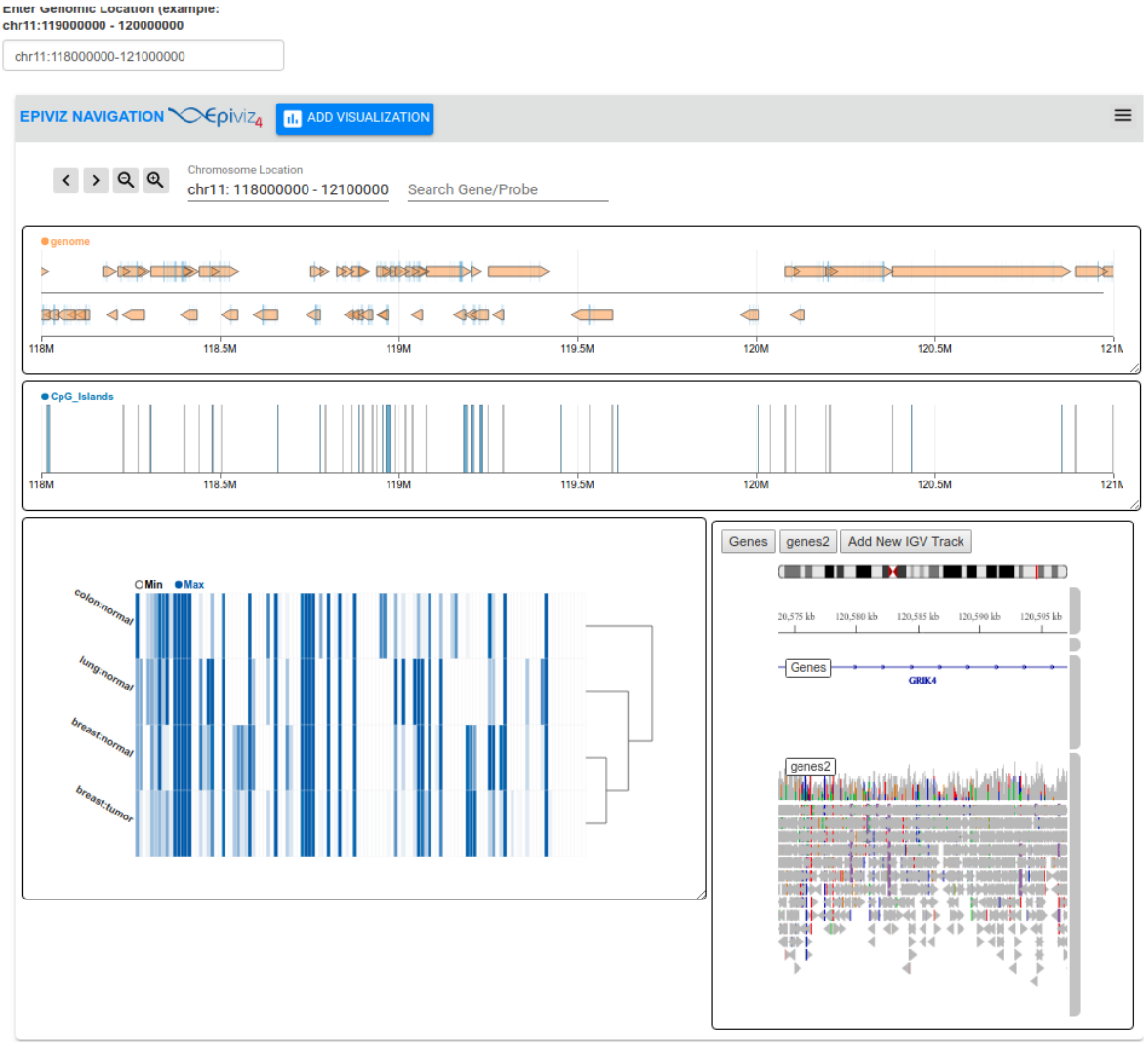

Figure 5. Interactive visualization of R/Bioconductor data objects in Shiny. In this Shiny application, we explore gene expression from the Gene Expression Barcode Project[11] for colon, lung and breast tissues for tumor and normal samples as a heatmap. We visualize annotation tracks for the genes and position of CpG islands in the current region. We also integrated IGV with epiviz components and the igv track (bottom right) displays the gene position and the aligned illumina reads for HG01879 sample from 1000 genomes[12] project. The IGV track queries the file directly to get data and visualize the reads. We also have a genomic location text box (top left) that is a non-epiviz component and can be used to interact with epiviz components within the Shiny application. Changing the location, updates the genomic region in the navigation element and all charts.

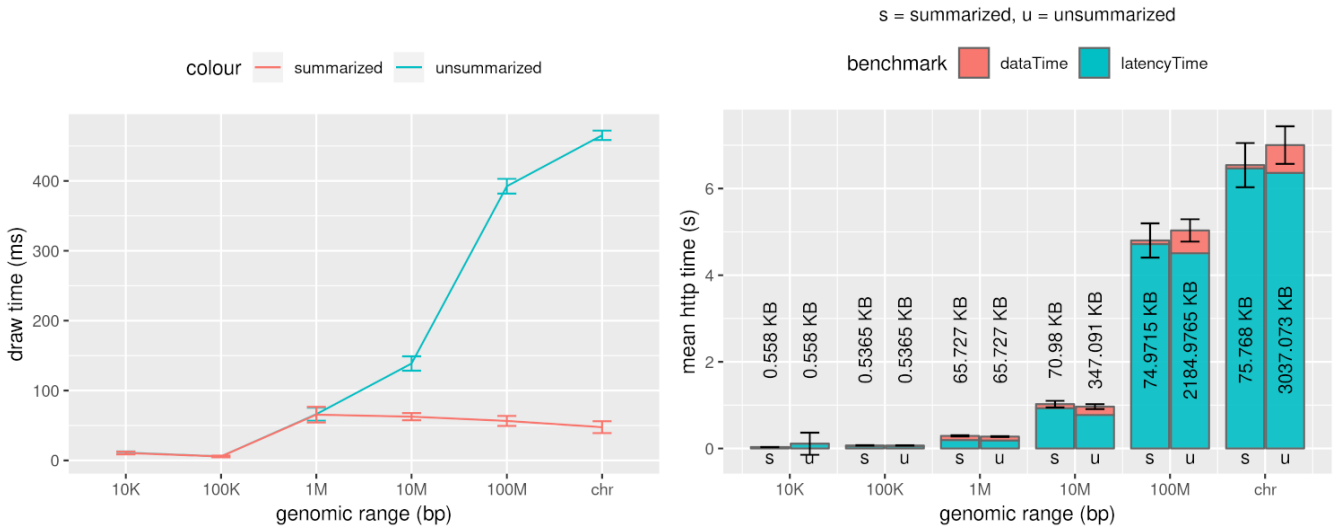

Figure 6. Effect of data summarization on the Epiviz Python data. Here we compare average data request and data rendering time for continuous data along the genome to study the effect of summarizing data on the data backend across 10 runs. Lines for 'unsummarized (u)' correspond to the previous Epiviz implementation where all data within a genomic region is returned by the php-backend to the web browser client. Lines for 'summarized (s)' correspond to our new implementation of the python-data backend, where data summarization within genomic regions is performed in the backend. The left panel shows the mean draw times between these scenarios where we see a significant improvement in the draw times when the data is summarized in the backend. The bar plot in the right panel shows the total http time and is separated to show mean latency times and data transfer times. The number of bytes transferred for summarized and unsummarized backends is also displayed. The error bars represent one standard deviation away from the mean draw time in the left panel and mean http time in the right panel. We observe that the total http request time (summarization plus data transfer) is comparable to transfer time for the larger unsummarized data scenarios.

available in the ***epiviz-chart*** GitHub repository. The benchmark scripts can also save a screenshot of the page rendered to make sure that the page is completely loaded and rendered.

Discussion

The component library is an extension to our ***Epiviz*** web application for visualizing functional genomic data sets. The component library is our solution to creating reusable and extensible visualization elements that work with any modern web browsers. The value of a data visualization library depends on its usability and easy integration with existing web frameworks. Epiviz components can be integrated with any framework that supports HTML.

The Web has now become the platform for application development and the demand for modular, extensible and reusable frameworks like web components is on the rise. Since epiviz components are modular, we believe it simplifies the process of developing and managing genomic web applications. We also welcome developers to contribute to and extend our component library.

Conclusion

To our knowledge, the Epiviz component library is the first genomic data visualization library based on web components. The library provides an easy and efficient way for bioinformatics developers to add interactive data visualization features to their web applications or datasets with minimal programming experience. It is cross-platform, modular and runs on any modern web browser. We introduced our ***Epiviz2*** web application to demonstrate the features and interactions that can be developed using the component library. We also showed the ease of integration with other frameworks by the R/Bioconductor ***epivizrChart*** package, that provides interactive, reproducible visualizations of data objects in R and also create interactive standalone HTML documents.

Future work

One of the advantages of web components is that HTML is now more readable. With a more declarative implementation, elements can be self-descriptive. We would like to implement a visualization grammar[13] similar to ggvis as attributes/properties on the epiviz elements. We plan to further develop the library to extend our current set of visualizations and support various genomic data types including those implemented in Metaviz[14] an interactive and statistical metagenomic data browser. We plan to implement canvas-based rendering of charts to scale and significantly reduce draw times especially when rendering large datasets.

Data availability

The Datasets used for the use case describing the Epiviz Application come from the NIH Roadmap Epigenomics Project. The data files are downloaded from Bioconductor's ***AnnotationHub*** repository and imported into the MySQL database using the functions available in the ***epivizrData*** package. For the

epivizrChart package, the datasets used are included as part of the package. The vignettes describing the use cases are also available on GitHub or through Bioconductor.

Software availability

Epiviz component library is open sourced and is available on GitHub. The collection of components discussed in this article are available at:

- epiviz charts - http://github.com/epiviz/epiviz-chart
- epiviz data - http://github.com/epiviz/epiviz-data-source
- epiviz workspace - http://github.com/epiviz/epiviz-workspace
- epiviz app - http://github.com/epiviz/epiviz-app

The scripts for benchmarks are available in the ***epiviz-charts*** repository. R/Bioconductor ***epivizrChart*** package is available either through Bioconductor (http://bioconductor.org/packages/release/bioc/html/epivizrChart.html) or GitHub (http://github.com/epiviz/epivizrChart). Both the respositories also contain the vignettes described in Figure 4 and Figure 5. The Python Flask API data provider is available at http://github.com/epiviz/epiviz-data-provider. Documentation is available at http://epiviz.github.io.

Archived source code at the time of publication – https://doi.org/10.5281/zenodo.1299990[15]

Software license: MIT License.

Competing interests

No competing interests were disclosed.

Grant information

This work was supported by the grant funded by the National Institutes of Health [R01GM114267].

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

Acknowledgements

We thank Lan Tran for his early work on the development of the component library.

Supplementary material

Supplementary File 1 – Document containing extending-genes-track contains code that extends an existing epiviz chart component (epiviz-genes-track) and displays a table of genes and their genomic positions.

Click here to access the data.

References

1. Kent WJ, Sugnet CW, Furey TS, *et al.*: **The Human Genome Browser at UCSC.** *Genome Res.* 2002; **12**(6): 996–1006.
PubMed Abstract | Publisher Full Text | Free Full Text
2. Robinson JT, Thorvaldsdóttir H, Winckler W, *et al.*: **Integrative genomics viewer.** *Nat Biotechnol.* 2011; **29**(1): 24–26.
PubMed Abstract | Publisher Full Text | Free Full Text
3. Gómez J, García LJ, Salazar GA, *et al.*: **BioJS: an open source JavaScript framework for biological data visualization.** *Bioinformatics.* 2013; **29**(8): 1103–1104.
PubMed Abstract | Publisher Full Text | Free Full Text
4. Chelaru F, Smith L, Goldstein N, *et al.*: **Epiviz: interactive visual analytics for functional genomics data.** *Nat Methods.* 2014; **11**(9): 938–40.
PubMed Abstract | Publisher Full Text | Free Full Text
5. Huber W, Carey VJ, Gentleman R, *et al.*: **Orchestrating high-throughput genomic analysis with Bioconductor.** *Nat Methods.* 2015; **12**(2): 115–121.
PubMed Abstract | Publisher Full Text | Free Full Text
6. Gottfried B, Kancherla J, Corrada Bravo H: **epivizrChart: R interface to epiviz web components.** R package version 1.2.0. 2018.
Publisher Full Text
7. Chang W, Cheng J, Allaire JJ, *et al.*: **shiny: Web Application Framework for R.** R package version 1.1.0. 2018.
Reference Source
8. 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
9. McCall MN, Jaffee HA, Zelisko SJ, *et al.*: **The Gene Expression Barcode 3.0: improved data processing and mining tools.** *Nucleic Acids Res.* 2014; **42**(Database issue): D938–D943.
PubMed Abstract | Publisher Full Text | Free Full Text
10. Roadmap Epigenomics Consortium, Kundaje A, Meuleman W, *et al.*: **Integrative analysis of 111 reference human epigenomes.** *Nature.* 2015; **518**(7539): 317–30.
PubMed Abstract | Publisher Full Text | Free Full Text
11. Timp W, Bravo HC, McDonald OG, *et al.*: **Large hypomethylated blocks as a universal defining epigenetic alteration in human solid tumors.** *Genome Med.* 2014; **6**(8): 61.
PubMed Abstract | Publisher Full Text | Free Full Text
12. Sudmant PH, Rausch T, Gardner EJ, *et al.*: **An integrated map of structural variation in 2,504 human genomes.** *Nature.* 2015; **526**(7571): 75–81.
PubMed Abstract | Publisher Full Text | Free Full Text
13. Wickham H: **ggplot2.** *Media.* **35**, Springer New York, 2009.
Reference Source
14. Wagner J, Chelaru F, Kancherla J, *et al.*: **Metaviz: interactive statistical and visual analysis of metagenomic data.** *Nucleic Acids Res.* 2018; **46**(6): 2777–2787.
PubMed Abstract | Publisher Full Text | Free Full Text
15. Kancherla J, Zhang AY, Bradford A, *et al.*: **epiviz/epiviz-chart: epiviz-chart.** 2018.
Data Source

Open Peer Review

Current Peer Review Status: ? ? ✓

Version 1

Reviewer Report 14 August 2018

https://doi.org/10.5256/f1000research.16818.r36145

✓ **Zhaohui Steve Qin**
Department of Biostatistics and Bioinformatics, Rollins School of Public Health, Emory University, Atlanta, GA, USA

In this paper, Kancherla and colleagues present the Epiviz Component Library, an open source reusable and extensible data visualization library and application framework for functional genomic data. According to the authors, the Epiviz component library is the first genomic data visualization library based on web components. The library provides an easy and efficient way for bioinformatics developers to add interactive data visualization features to their web applications or datasets with minimal programming experience.

Overall, I found the tools describe in the paper very useful and promising. It makes complex visualization of multiple types of omics data easy and convenient. The new infrastructure is built upon the established epiviz tool that the group has developed earlier. This group is highly experienced in genomics data visualization and are developing state-of-art infrastructure using the latest technologies. The utilities of the tools great outweigh their limitations and shortcomings. I only have a few minor points.

Figure 3. In this use case, I don't quite understand the connection between the gene expression scatter plot and the other panel. For me, the figure is more like the demonstration of all types of plots it can produce. It is not obvious what biological insight can be gleaned from the plots. It is unclear what kind of biological question the user is trying to ask when making these plots. To be fair, this is always difficult. In most cases, discovery is made unintentionally, out of luck. Hence just demonstrating all the plotting capabilities is probably okay. If so, may be a systematic catalog of all the plots that can be produced will be helpful.

Figure 6 is very informative and interesting. I noticed that the numbers of bytes transferred for summarized and unsummarized backends are exactly the same for 10K, 100K and 1M, but dramatically different for 10M, 100M and chr. Why is that?

It would be very helpful to the general audience if the authors can articulate in more details, and

intuitively, the benefits and advantages of web components-based visualization library for genomics data, compared to the current technologies.

Is the rationale for developing the new software tool clearly explained?
Yes

Is the description of the software tool technically sound?
Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Yes

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?
Yes

Competing Interests: No competing interests were disclosed.

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

Reviewer Report 08 August 2018

https://doi.org/10.5256/f1000research.16818.r36142

?

Ian Holmes
Department of Bioengineering , University of California, Berkeley, Berkeley, CA, USA

This paper describes a library for integrative genomic data exploration, using modern web technologies, particularly HTML Web Components, with solid R integration. The interactive figures in the paper are very interesting, and show how the library can be used to build basic dashboards or RMarkdown pages with interactive charts and genome views. The dashboards look a little spartan, and a bit cryptic at times, but are a great illustration of the potential for this kind of thing and could develop into something really powerful. Plus, with the Bioconductor integration, they offer a lot of ready-to-go useful data for use cases around human genomics. All round a good contribution, although I think marred (very slightly) by some opening text in the paper which

presents the software as being very general-purpose and detached from specific use cases - a style of presentation which I view as a mistake that risks detracting from a reader's clear understanding of what the software can actually do. Modularity is a virtue, but developers often overestimate readers' level of interest in it.

For example, according to the authors, previous genome browsers have all been based on "specific use cases", whereas Epiviz Web Components are more amenable to "integration and extensibility". On closer reading, it turns out that the primary use case (the NIH Roadmap Epigenomics project) is just a bit further down in the paper, as is the platform (R/Bioconductor) currently required to load any useful data or the documentation vignettes. So I think the generality of the tool is possibly a bit hyperbolically described, in a way that risks obscuring the actual current uses (and software dependencies) of the tool.

The authors claim that, in Epiviz Web Components, they have developed the first genome browser that uses Web Components, which form a collection of browser features and APIs that have emerged from JavaScript libraries like React. Due to the fast-changing nature of the web, technologies such as this emerge with rapidity and regularity, and I find it plausible that this is the first genome browser to use these tools. They should be useful and it's interesting to see them described in practice. The Epiviz Web Components tool is a useful addition to the bioinformatics visualization ecosystem, bridging R and the modern web.

With that said, I think perhaps that web bioinformatics software - and its associated publications - should be measured on several axes. Compliance with emerging web standards (such as Web Components) is certainly one such axis, but compliance with established bioinformatics standards would be another axis (how many formats/database schemas does the software support? how many other resources does it integrate with? what choices were made about which data sources to support, and which to omit? are those choices explained in the text? is there a guiding philosophy that can help inform readers who might be considering using this software?)

In terms of bioinformatics compatibility, bioconductor's AnnotationHub seems intended to be the primary way of loading data into the browser, though it isn't quite presented this way. Bioconductor is mentioned in the abstract as an example that was developed "to demonstrate the ease of integration with other frameworks"; I think it would be more accurate to say that Bioconductor is the only framework that this tool supports, and while the tool was written in such a way as to be hypothetically platform-independent, that hypothesis has not yet been seriously tested.

Epiviz Web Components does have its own JSON formats for data, which is promising in terms of backing up the claim that data import would be straightforward, but as far as I can tell there are no tools to import common file formats (FASTA, GFF, etc) and it's not clear whether there are any plans for developing such import tools, or if the Bioconductor dependency will in practice be permanent.

The paper's interactive figures (Figures 3 & 4) are intriguing. I unfortunately had some problems installing the R from source (several dependencies of epivizrChart failed to build on my Mac Pro running macOS Sierra 10.12.6, R version 3.3.3), so I have not tried them on my local machine, only the JavaScript running in the web browser client. I found the captions to the Figures hard to follow: for example, I found myself wondering if there was a way for the user in the web browser to figure

out how the components in Figure 3 (the genes track, stacked-blocks track, and line track) are linked, to use the same underlying data sets? In other words, does that linking only happen via configuration files and other back-end interfaces, or is it exposed to the end user? How were the markers H3K27me3 and H3k9me3 selected, why esophagus & colon, why drill into methylation and expression, what's the biomedical back-story here? And just in terms of how to use the app to explore these cases as described, there could be more exposition. There are quite a lot of interactive buttons and menus and nested windows, so I was looking around for explanations of all that, passing by the documentation at https://epiviz.github.io/epiviz-chart/ (which is apparently oriented toward developers, not end-users, and is pretty minimal) and ending up at the https://epiviz.github.io/ video tutorial ("Epiviz quick tour"). I couldn't find any thorough written description of the Epiviz web app, only that 3-minute YouTube video. I'm not very confident in my understanding of the totality of its capabilities after watching the video and looking at Figure 3. The potential to build pages out of RMarkdown, as shown in Figure 4, is pretty exciting.

In summary: I would suggest improving the article by adding more extensive descriptions of the biological use cases, and how they might be investigated using the tool. Make another video if you insist, but what I'd really be looking for is clearly-written tutorial text with figures. This will complement and enhance the current concise description of the software, and offer an alternative approach for a broader class of readers.

Is the rationale for developing the new software tool clearly explained?
Partly

Is the description of the software tool technically sound?
Partly

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Partly

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
No

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?
Yes

Competing Interests: I work on web-based genome browsers.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Reviewer Report 03 August 2018

https://doi.org/10.5256/f1000research.16818.r36143

?

Bianca H. Habermann
Aix-Marseille Univ, Aix-Marseille Univ, CNRS, IBDM UMR 7288, Marseille, France

Kancherla et al describe a set of re-usable web-components for the Epiviz tool, which are designed for visualizing genomic/epigenetic data in an integrative and interactive way. The components include the visualization components, app components and datasource components, are designed in a modular way and can be integrated with JavaScript libraries and other HTML frameworks. Moreover, there is an R-version available, which can be used with R / Shiny and to produce stand-alone HTML documents.

The general concept and design of the Epiviz web components is very good. With growing amounts of -omics data, we need easy-to-use, re-usable and extensible code for interactive data visualization, since visualizing large-scale data enhances our understanding of inherent trends and features of the datasets under analysis. We therefore have a growing need for easy-to-use and re-usable software for their visualization; re-useable and extensible tools like Epiviz are highly sought-after.

The manuscript is in general well written and the software has a nice look and feel. Though, there are a couple of points that should be addressed prior to indexing.

- While I find the description of the tools quite good and explicit, the description of the used biological examples, as well as the plots, is in my view too sparse. In fact, visualization only makes sense to the reader, if he/she understands, what is plotted and how to interpret the plots. With the manuscript at hand, as well as the user manual, this is not made easy. For instance, there is a general lack of axis labels on the scatter plots. E.g. Figure 2 and 3 show scatter plots, however from different data sources (peak positions/height or differential detection(?) on the genome, RNA-seq data (?)). What are the numbers on Figure 3 (ie the scatter plot of the RNA-seq data)? Are these rpkms? Log2 fold change? As the plots are quite useful and might be used directly for publication, this is a feature that should be implemented.
- Also, in the same Figure/example (methylation workspace following the given link of the legend to Figure 3), how does the RNA-seq data relate to the chip-seq/methylation data? Is there a possibility to see the identity of the genes, when hovering over the dots in the scatter plot? At least for the pre-selected example shown on the web-page, there is no brushing over from the zoomed-chromosome plot to the scatter plot and vice versa. However, this feature seems to work, when changing the chromosomal region. Is this due to the fact that the genes in the selected region are not present in the RNA-seq data? If so, the authors should think about changing the selected region for their demonstration.
- Also, what is the relationship of genes in the scatter plot that are highlighted together, when one is selected? Do they share the same enhancer/methylation peaks?
- It might be useful to provide information on the genes selected in the RNA-seq plot also via an info box or pop-up, which shows the name of the gene(s) and its(their) associated differential expression values.

- Is there the possibility to e.g. change the region of the chromosome-zoom/the chromosome view, when selecting dots in the scatter plot of the RNA-seq data? This seems currently not possible, however would be desirable.
- In the same example, the brushing of the chromosome view vs the zoomed-in chart does not work. When changing the genomic region, there is no highlighting any more in the chromosome view, so only the pre-selected region seems to work. This needs to be fixed.
- Finally, in the same plot, changing the chromosome for zoomed-in visualization has no effect on the whole-chromosome view, which I assume should also be updated. In my opinion, the whole-chromosome view is in its present form not really useful and could be omitted.
- In the same workspace (methylation workspace), there is a scatter plot at the bottom of the page; what do the numbers in the scatter plot refer to? See data labels problem already discussed above.
- On the binning of data values, the authors describe that the genomic region is binned into 2000 intervals. To which overall length does this refer to? Is always a fixed length chosen or can the user determine the length? Binning e.g. 100000 bp would then give quite different results than e.g. 1000000 bp.

There are also some errors in the manuscript, more specifically in the description of Figure 3. It currently states:
“In this example, the navigation element at the bottom of the page visualizes (in order from top to bottom): 1) a genes track showing … “
Instead, it should read, I assume, as the genes track is at the bottom of the figure:
“In this example, the navigation element at the bottom of the page visualizes (in order from **bottom to top**): 1) a genes track showing … “

Is the rationale for developing the new software tool clearly explained?
Yes

Is the description of the software tool technically sound?
Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Partly

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?
Partly

Competing Interests: No competing interests were disclosed.

I confirm that I have read this submission and believe that I have an appropriate level of

expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.