# Ten simple rules for developing visualization tools in genomics

E. Durant, M. Rouard, E.W. Ganko, C. Muller, A.M. Cleary, A.D. Farmer, M. Conte, François Sabot

PLOS COMPUTATIONAL BIOLOGY

EDUCATION

# Ten simple rules for developing visualization tools in genomics

**Eloi Durant[1,2,3,4]*, Mathieu Rouard[3,4], Eric W. Ganko[5], Cedric Muller[2], Alan M. Cleary[6], Andrew D. Farmer[6], Matthieu Conte[2], Francois Sabot[1,4]***

**1** DIADE, University of Montpellier, CIRAD, IRD, Montpellier, France, **2** Syngenta Seeds SAS, Saint-Sauveur, France, **3** Bioversity International, Parc Scientifique Agropolis II, Montpellier, France, **4** French Institute of Bioinformatics (IFB)—South Green Bioinformatics Platform, Bioversity, CIRAD, INRAE, IRD, Montpellier, France, **5** Seeds Research, Syngenta Crop Protection, LLC, Research Triangle Park, Durham, North Carolina, United States of America, **6** National Center for Genome Resources, Santa Fe, New Mexico, United States of America

* eloi.durant@ird.fr (ED); francois.sabot@ird.fr (FS)





**Citation:** Durant E, Rouard M, Ganko EW, Muller C, Cleary AM, Farmer AD, et al. (2022) Ten simple rules for developing visualization tools in genomics. PLoS Comput Biol 18(11): e1010622. https://doi.org/10.1371/journal.pcbi.1010622

**Editor:** Scott Markel, Dassault Systemes BIOVIA, UNITED STATES

**Published:** November 10, 2022



**Funding:** This work was supported by a CIFRE doctoral grant (2018/1475 to ED). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

**Competing interests:** The authors have declared that no competing interests exist.

## Introduction

Visualization is key for expanding and communicating knowledge to both specialized and broad audiences—after all, "*a picture is worth a thousand words*," right? That much has become clear during the Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) outbreak when "*Flatten the curve*!" [1] turned into a catchier watchword than the usual "*Wash your hands*!," by referring to the related graphics rather than the health discourses themselves. Coronavirus Disease 2019 (COVID-19) visualizations, good and bad [2], became omnipresent in the public debates, with interactive and dynamic visualization platforms like Nextstrain [3] gaining a lot of popularity.

We are now used to seeing graphs and charts in our everyday lives and creating them for scientific papers as part of research. Unfortunately, most of the time, classic visual representations are insufficient to effectively communicate data complexity, and as O'Donoghue puts it, "*often dedicated communication approaches need to be developed to address specific data challenges*, *especially when conveying complex or unfamiliar ideas*" [4]. Given the gap between creating static figures and building a visualization tool from the ground up, it can be easy to get lost in this nontrivial journey.

Our following 10 simple rules are dedicated to biologists and bioinformaticians who, while already being at the crossroads of many fields, want to venture further into the land of Data Visualization ("datavis" or "dataviz" for short). They combine tips and advice that we would have wanted when we first started our own journeys, gathered from our experiences in building genomic and/or datavis tools, and the time spent with related communities. Additionally, they address current challenges in computational biology and the needs of the community.

We aim these rules at bioinfo-to-datavis novices looking for guidelines, particularly regarding **genomics** visualization tools, but experienced practitioners may find it useful to see them gathered in one place too.

For better reading comfort, we organized the rules chronologically depending on when they would matter the most during a visualization tool's life cycle, starting with the design (Rules 1 to 5) then development (Rules 5 to 8-ish) phases until it is shared with the rest of the world. Please note, however, that they are still relevant during the whole gestation and that creating a visualization tool is rarely a fully linear process—it does not even end after the initial release, as continued development and support is a cyclic process to be sustained over the years.

## Rule 1: Articulate the need for new visualization tools

For the past few decades, most genomics data have been visualized through genome browsers (JBrowse [5], Ensembl [6], IGV [7], etc.). With the advent of Next Generation Sequencing technologies, the number of draft or high-quality assembled reference genomes exploded, and the need to federate (gen)omic data across assemblies within and between species and to visualize them has become a major challenge.

All these data types and tools lead to critical questions, such as "***What am I trying to visualize?***" and "***Why?***".

More specifically: What data do I have? Am I interested in structural variations? In epigenetic changes? At genome scale or at a defined locus? Are there gene annotations available? Can I get sequence alignments? Is there already a tool fulfilling all my expectations?. . .

Such questions must drive your quest for the appropriate datavis tool. A good first step would be to explore the related bibliography and tool reviews. Some reviews focus on the use cases [8] or layouts [9] of genomic visualization tools in general, while other reviews are specific to certain subjects, such as structural variations [10] or Hi-C data [11]. Your data and visualization needs may already be compatible with an existing tool or additional development of an existing tool could bring the feature you need. Getting a feature added to a tool can be done, for example, with GitHub by making a feature request or even adding the feature yourself with a pull request—consulting the dev team beforehand is usually a good idea, to be sure that what you propose is in line with their vision and community. Alternatively, some visualization tools allows the development of extensions or plugins, as exemplified by MetaPGN [12] which is built on top of a Cytoscape [13] plugin or the JBrowse plugin store (https://jbrowse.org/jb2/plugin_store).

If after all these steps nothing matches your needs, it then appears necessary to develop a new visualization tool. If so, your big challenge will be to **clearly identify the scope of your visualization tool** and to **keep your end goal in mind**.

## Rule 2: Involve others early on

There are many "others" that you could and often should work with when creating a datavis tool. Sedlmair and colleagues [14] identified 6 non-mutually exclusive collaborator roles of interest for design studies of such tools, with one of the main roles being the **front-line analysts**.

**Front-line analysts** are your end users, people who will use your tool and need it to complete certain tasks, and who should therefore be handled with particular care. It is crucial to engage them in discussion as early as possible since they are the ones who will (hopefully) adopt your tool. **Front-line analysts** can help you to properly define the tool tasks (i.e., both high- and low-level tasks that your tool should achieve) and provide test data as well as valuable feedback during both the design and development phases. There are many good ways to engage with your end users, including face-to-face interviews, surveys, design sprints, and even hands-on sessions that can often reveal edge cases and sometimes bugs that would have been hard to find on your own—beta-testing at its finest.

Moreover, there are fields dedicated to datavis: understanding how they are perceived by the human brain (visual perception and cognitive vision science), improving how they can be used with machines (Human Computer Interaction), and growing communities of **datavis designers**, full of people who could help build a visualization tool. They can be rather generalist, like the Data Visualization Society (https://www.datavisualizationsociety.org), or more dedicated to the visualization of life sciences data, such as the BioVis community (http://biovis.net), which regularly organizes conferences on related topics. Look around, there might be someone there willing to give you a hand!

## Rule 3: Think about visual scalability and resolution

If the history of genomics teaches us one thing, it is that what was once a huge dataset can be considered a toy set 5 years later [15]. The exponential growth of genomic datasets leaves us no choice but to consider the scalability of our tools' design and the efficiency of their visual encodings (i.e., how well the chosen visual representations reflect the underlying data). Some designs might be good visual representations for small datasets, but scale poorly for larger datasets.

For example, Venn diagrams are fine for up to 3 sets, but with more sets they become messy at best and a nightmare in worst case scenario. UpSet bypasses this issue by focusing on the set intersections and presenting them in an ordered "table," but it still suffers a loss of readability with increasing sets (there is simply too much to see) and is not designed for visualizing hundreds of sets [16]. Similarly, networks and graphs do not adapt well to big datasets with many nodes and edges, resulting in the infamous "hairball effect" [17]. Your chosen visual encoding's clarity can depend on your dataset, with different behaviors between simulated or real data of limited scope and larger, more complex data (Fig 1). **Your data will come with edge cases that you need to anticipate if you want your design to work properly at scale** [18]. Try varied configurations (good ol' pen and paper are still useful to quickly draft multiple layouts), get familiar with your design's limitations, and offer alternatives if you have the resources for it.

Genomic data has specificities and layers that you need to consider at different resolutions. Your organism(s) may have multiple chromosomes (linear or circular) to consider, heterozygosity or even polyploidy, and may present macro structural rearrangements at a chromosome level that will not interest you at the nucleotide sequence level. Along with the genomic sequences, you may also have access to additional data types, such as Hi-C, epigenomic signatures, or detections of transcription factor binding sites, all of which can blur the respective message in all-in-one visualizations. Instead, consider different visual representations for each data type and how they can complement each other through comparisons and interactions.

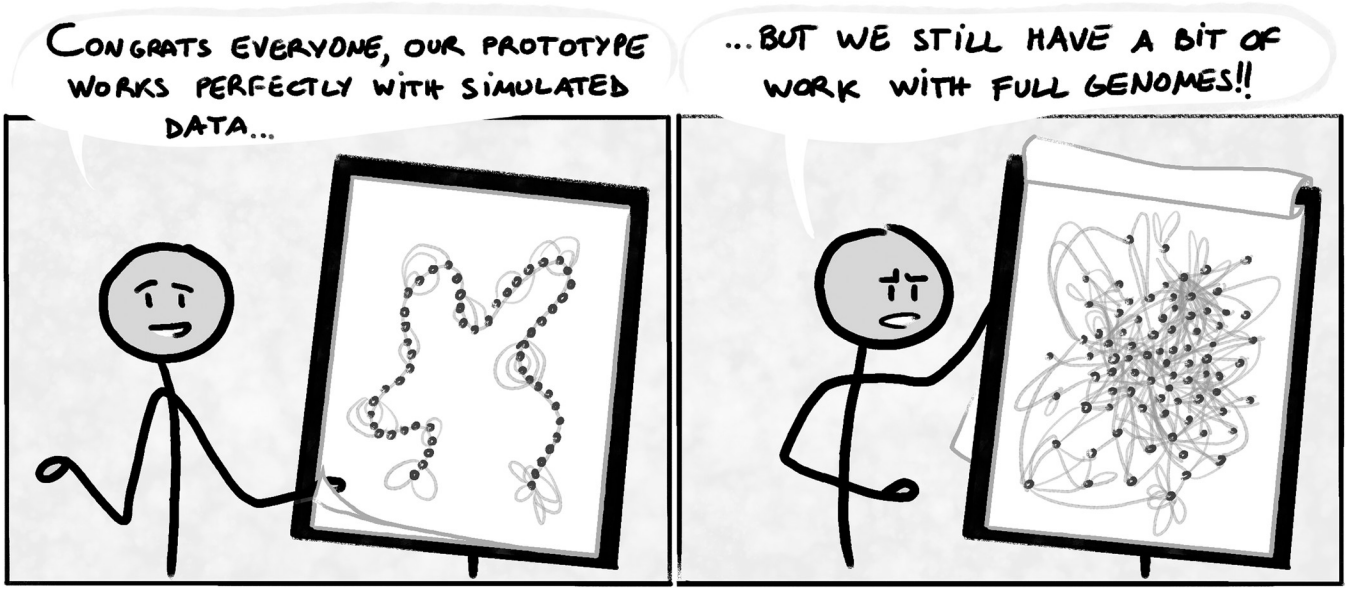


**Fig 1. The readability of a visual encoding depends on the data provided.**

https://doi.org/10.1371/journal.pcbi.1010622.g001

## Rule 4: Be creative, be bold

To develop a successful visualization application, you will have to find the right mix of technology, usability, and aesthetics.

Regarding technology, you can envision that scientists will soon have access to equipment similar to what Tom Cruise used in *Minority Report*, which will interact with virtual screens to perform multidimensional genomic data analysis. Virtual reality headsets and augmented reality devices offer a first, more affordable step in this unfamiliar 3D environment for datavis—some have already been put to use for GWAS [19] and visualization of 3D structure of chromatin [20] or genome graphs [21]. However, while technically feasible, such technologies are not popularized yet and that choice would likely reduce the number of end users. Some tools compromise by projecting a 3D space onto a 2D computer screen, like Graphia [22] that uses perspective views and shading to simulate depth perception. There are plenty of technology options available and you will need to keep up to date, but make sure that your audience can effectively and correctly use them!

As for aesthetic and artistic ways to present data, it is partly subjective. Something nice and eye-pleasing to one person may not be appreciated and understood the same way by another person. However, there is a good amount of literature providing advice for efficient datavis design [23–25]. One example is to pay special attention to how you use colors [26]. Make your tool more accessible by accommodating visually impaired people [27], who make up more than 3% of the global population [28], and consider providing a way to let users customize their visualizations with their own color patterns.

Overall, **be creative, do not be afraid to create designs that will not make it to the final cut**, and test different graphical strategies following user experience (UX) design processes [29]. Finally, even though too much novelty could be a barrier to adoption, a smart and beautiful design will encourage your users to engage more by offering them visual clarity of their data. If art is really your thing, though, you could totally find a place among the "*Xenographics*" (https://xeno.graphics) or create your own science-inspired pieces in your free time (check out Martin Krzywinski's π-art gallery: http://mkweb.bcgsc.ca/pi/art)!

## Rule 5: Make data complexity intelligible

A familiar way of making complex data accessible is to compute derived measures and statistics or to apply dimension reduction techniques. Still, visualization is recommended to detect patterns that would not be found with these means alone, as illustrated in Anscombe's quartet [30], and the more recent Datasaurus Dozen [31]—based on Alberto Cairo's eponymous dataset—which inspired the Fig 2. Unfortunately, the human mind cannot make sense of everything that it perceives at once, and our attention is instead focused on fragments of what we see.

In practice, our understanding of graphical representations comes in part from the immediate identification of salient elements that visually "pop out" (i.e., *preattentive processing* [32]), but it mainly comes from active exploration [33]. Therefore, a key component of datavis is to make exploration easier and to reduce the workload for a user's brain starting with careful design as to relieve their working memory [24].

A universally praised rule of thumb to this end is Shneiderman's mantra: "***Overview first, zoom and filter, then details-on-demand***" [34]. This approach encourages visualization developers to make clever use of the main advantage of visualization tools over visual representations: interactivity. With such a divide and conquer approach it becomes easier to make sense of the displayed visual information as a user. For example, adding details with dynamic tooltips

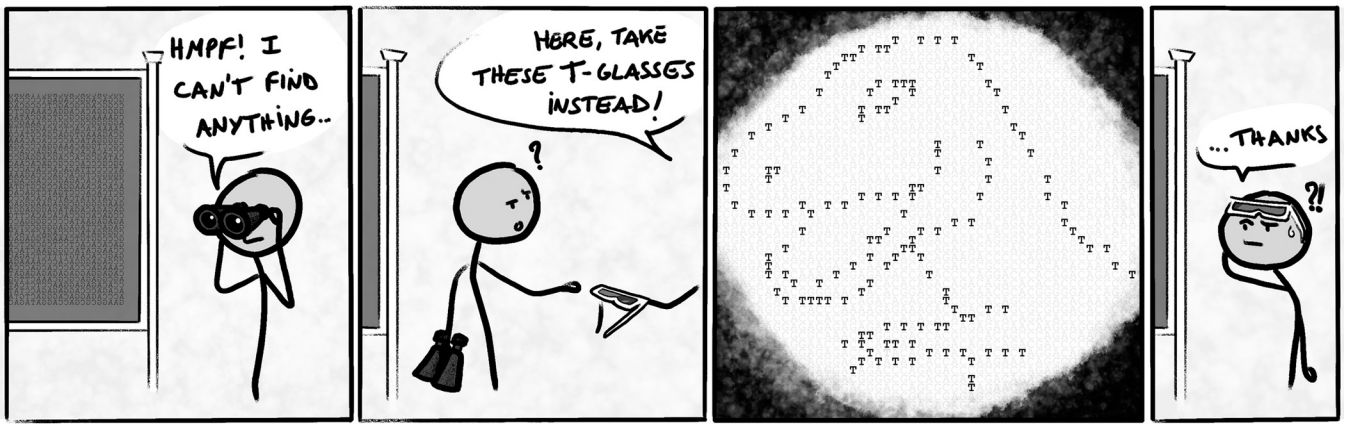


**Fig 2. A good datavis makes pattern identification easier (feat. the Datasaurus).**

https://doi.org/10.1371/journal.pcbi.1010622.g002

or including dedicated linked views enables users to build a deep understanding of the data provided without hiding the overall patterns.

## Rule 6: Let your inner nerd shine if needed

Technology matters, but it is not an end in itself. If you can design and produce an efficient tool using reliable methods instead of the latest and trendiest ones, there is no reason not to. Using the latest framework or published method can be interesting when they add significant advantages, but they may come with stability issues and will not always be the safest option for a robust and long-term tool (as an example, we can think of the infatuation with Objective-C in the mid-2000s leading to the creation of the BioCocoa framework (http://bioinformatics.org/biococoa), without any evolution since 2011). Moreover, accumulating niche technologies and methods would make it difficult to find someone with the exact skillset needed to maintain or help develop your tool; consider the consequences of that choice carefully.

On the other hand, it is also important to recognize that software technologies evolve quickly, and once-common approaches to user interface development may lose favor or become inviable (consider applets and other rich-client alternatives to browser-based development or the genomic visualization tool Gobe [35] that unfortunately came out after the first signs of Adobe Flash's downfall). Sometimes a well-conceived and popular tool may need to be completely overhauled in order to remain relevant and usable. This is what happened with genome browsers whose implementations transitioned from HTML image-maps generated server-side to client-side JavaScript frameworks.

You will need to use relevant technologies for your tool: using LaTeX to produce pangenomics visualizations is fun but needlessly complicated [36] and there are better, dedicated technologies out there. Common tools in datavis differ from common tools in bioinformatics in general, and you will have to try and get accustomed to unfamiliar technologies and concepts. For web-based visualization, you should first familiarize yourself with the differences between Scalable Vector Graphics (**SVG**, geometrically defined images) versus **HTML5 Canvas** (dynamic pixel-defined images) elements. A must-know library for working with visualization is **D3.js**, which enables the manipulation of (graphic) elements on a webpage. Finally, if your goal is to display numerous elements at a time (say thousands or millions), you should consider working with **WebGL** (an API using the graphic card for faster display)—check out the Three.js library or the GenomeSpy [37] or Gosling [38] visualization grammars if interested.

## Rule 7: Benchmark with diverse datasets

While it is important to have a well-thought-out design, you cannot ignore performance, as a smooth user experience is key for your tool's adoption [39]. Benchmarking with different datasets is critical for your visualization tool, and each type has its own use.

- **Small simulated** datasets can and should be used during development, as they are better suited for fast iteration while debugging. Make sure that what you see is faithful to the data you have. Handcrafted files can be useful to make extra sure that the resulting visuals match your expectations.
- **Big simulated** datasets are meant to be used for stress tests, to assess scalability and performance: how much data can you feed to your tool until it breaks? Until the tool starts slowing down too noticeably? Performance is an important aspect of your tool that should be determined, at least to let users know what to expect if they want to use their data with your tool.
- **Real** data, finally, to make sure that your tool is working properly with real-life data. Look out for unexpected behaviors and edge cases related to your design or data format loopholes that may hinder the user experience.

With the decrease of sequencing costs, it is quite frequent to have huge projects with hundreds if not thousands of samples: humans, *Arabidopsis*, mosquitoes, rice, bacteria, viruses, and so on. A modern tool would be expected to manage hundreds of datasets, with as few lags as possible and no memory crashes. Keep in mind that "hundreds of genomes" may not correspond to the same size depending on the studied organisms: 1,000 HIV genomes represent approximately 9.2 Mb, which is less than 0.14% the size of a single human genome. You should therefore consider performance through 2 axes: count capacity (number of genomes or measures that can be added) and size capacity (efficient and maximum file sizes that can be used). You may also need to consider characteristics such as the number of chromosomes (or scaffolds) within a genome as well as the distribution of sizes among these elements (e.g., an abnormally large chromosome in an otherwise "normally sized" genome could break the assumptions of your data structures).

If performance becomes an issue, compiled code and optimized graphics libraries could be a good alternative to scripting languages (e.g., JavaScript, python) to generate the display of large amounts of data.

## Rule 8: Be aware of the genomic tool ecosystem and promote interoperability

Your application will likely need to work among an existing ecosystem of databases and tools that vary by community, even when developed as a stand-alone instance. To lower your tool's barriers to adoption, **pay special attention to implementation, distribution, documentation, and deployment**.

Regarding implementation, your tool should be as light as possible, OS agnostic [40], and if possible, run client-side. Implementing a standard Genomics API [41] to consume input datasets will also enable seamless integration and better interoperability with other databases and applications. Moreover, some users may want to include your visualizations in automated workflows, so the ability to get output via a command-line and companion scripts can be a nice additional feature to provide.

For distribution, consider multiple options—a Docker or Singularity containers [42,43], or the Conda [44]/Bioconda [45] environment management systems can make complex setups much easier to install, and providing users with the details to install with dependencies on

other systems is also valuable. Web applications (e.g., those built with JavaScript) are great for avoiding multiple setup issues as they can work with any web browser.

When time and resources allow, making a popular tool available on several platforms and programming languages is a good option for sustainability. JBrowse is a good illustration since it is available in both web and desktop versions [5], can be embedded in large genome portals as a Drupal module [46], has an R markdown and R Shiny compliant version (JBrowseR [47]), and exists as a Jupyter package [48].

Hosting an example of your software on a website can be useful for trial purposes but can be difficult to maintain over time. Always seek to make it easy for potential users to try your tool on their systems—include clear instructions and a straightforward way to download and run the tool. Good trial data helps users understand the tool and is also a good test to make sure the tool is running properly following installation. Moreover, data formats in bioinformatics can be tricky [49] so clearly document the formats your tool needs and provide examples.

## Rule 9: Keep up to date with related work

Genomics has been quickly evolving in the past years and shows no sign of slowing down in the future. Envisioned future developments and challenges are linked to more diverse and applied omics approaches (especially in health), data ethics and security, reference-free studies and others, all with ever more data and a growing need for integrated solutions [50].

With fast evolving needs and many people working on these subjects, it can be easy to be the needle in a haystack and be forgotten in favor of another tool: As of May 30, 2022, there are 434 visualization tools listed in the *awesome-genome-visualization* list [51] and 151 research articles (already 13 out in 2022 and counting) found via Europe PMC whose listed keywords matched "*visualization*" and "*genomics*." Make sure you keep up to date with data and technologies in genomics so that your ideas and tools stay fresh and relevant. Do not let your focus wander off too much either: A polished and thoroughly executed idea is better than a hybrid monstrosity of hastily (re)defined goals.

Moreover, you should try to **ride the wave rather than fight against it**: Make your work known and alive (in conferences, articles, Twitter. . .) and people will start noticing and using it. After all, you could very well make the new state-of-the-art tool!

## Rule 10: Grow and support your user community

Gaining users is the final key to a great visualization tool. To do that, you need to communicate your tool to prospective users as well as those who are actively working with it. **Communication can take time, but it is rewarding**—not only to ensure that people are using your tool, but also to gather ideas for improvements.

User documentation is a simple starting point—make sure you have an up-to-date README or manual for your software, including multiple examples on how to actually use your software. GitHub is an easy way to centralize this information and engage with those that have downloaded your project and are attempting to use it, especially through the "Discussions" and "Issues" features. Projects with no activity or updates are a warning sign to potential users, whereas those with activity are more enticing. Plus, you can centralize answers to frequent questions and, if you have an active user base, GitHub Discussions now supports polls, making it a convenient way to get input on new ideas. Furthermore, as reproducible research and FAIR-sharing principles are applicable to software products [52,53], it may encourage users to cite a released version of your tool by using tools like Zenodo to issue persistent Digital Object Identifiers (DOI) against your releases.

When presenting at a conference, try to end with 1 to 2 questions about new feature requests or biggest issues, along with contact details. You could create dedicated groups on online platforms like meet-up or organize training sessions towards federating a user community. Also consider setting up a google search notifications around the name of your tool—you might find mentions on Biostars, publications, and other sites you were not expecting.

Finally, do not forget to use social media to cast a wider net by advertising releases of the tool as well as occasional examples of its use [54].

## Conclusions

Our 10 simple rules for developing genomic visualization tools will not replace the experience gained by trial-and-error but will hopefully make the development process less painful for future bio-datavis practitioners and other datavis enthusiasts. Moreover, these rules merely cover the tip of the iceberg, and we strongly encourage our readers to take a look at the references included in this paper to further improve their understanding. Other interesting matters that did not make it through the introductory cut include visual design validation [55,56], inclusion and accessibility matters [57,58], and visual representation of uncertainty [59,60], which are subfields on their own.

Most of the elements presented here would also be applicable to non-genomic datavis tools; what may be more specific to genomics here is that the datasets are increasing at fast pace [15], with a target audience of scientists that have specialized questions covering a wide range of data types and scales. Depending on your goal (see Rule 1), you may have to create a multi usage tool that could work for pre-established analysis workflows and exploratory usages as well as for capturing printable versions of your visual representations at a given state.

These static snapshots could be used for publication, as you should also considering publishing papers about your tool and the design choices behind it (datavis-oriented columns are gaining in popularity, for example, with the datavis section in *Frontiers in Bioinformatics*).

To wrap up these 10 rules, here is a summary of what we deem most important, our take-home message for those who did not have time to read the article in detail:

- Going blindly into building a visualization tool is a bad idea; take time to learn about datavis principles first (we highly recommend Nature Methods' "Points of View" column [61]) and to carefully consider your tool's visual representations and interface designs.
- Genomics comes with multiple challenges, due in part to its diversity and scale of data. Make sure you have correctly identified the tasks your tool should complete and consider implementing interactions between different visual representations.
- Interact with your community at all times: during the design phase to correctly assess your end goal; during development to use real data and make sure you are going in the right direction; and on an ongoing basis once your tool is available for all to enjoy, to prevent it from ending in the "*oubliettes de l'histoire*."

## Supporting information

**S1 File. Extraction processes and data used in Rule 9.** Text file including the extraction method and request string used to count tools from the awesome genome visualization list and publications related to genomic visualization from PMC.
(TXT)

## References


1. Brian É. "Flatten the Curve!" But Which Curve? Hist Mes. 2020; XXXV:233–246. https://doi.org/10.4000/histoiremesure.13544
2. Krzywinski M. A pandemic of bad charts. 2022. Available from: https://www.youtube.com/watch?v=_YGmfsKL8N8.
3. Hadfield J, Megill C, Bell SM, Huddleston J, Potter B, Callender C, et al. Nextstrain: real-time tracking of pathogen evolution. Bioinformatics. 2018; 34:4121–4123. https://doi.org/10.1093/bioinformatics/bty407 PMID: 29790939
4. O'Donoghue SI. Grand Challenges in Bioinformatics Data Visualization. Front Bioinform. 2021; 1. Available from: https://www.frontiersin.org/article/10.3389/fbinf.2021.669186.
5. Buels R, Yao E, Diesh CM, Hayes RD, Munoz-Torres M, Helt G, et al. JBrowse: a dynamic web platform for genome visualization and analysis. Genome Biol. 2016; 17:66. https://doi.org/10.1186/s13059-016-0924-1 PMID: 27072794
6. Cunningham F, Allen JE, Allen J, Alvarez-Jarreta J, Amode MR, Armean IM, et al. Ensembl 2022. Nucleic Acids Res. 2022; 50:D988–D995. https://doi.org/10.1093/nar/gkab1049 PMID: 34791404
7. Thorvaldsdóttir H, Robinson JT, Mesirov JP. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. Brief Bioinform. 2013; 14:178–192. https://doi.org/10.1093/bib/bbs017 PMID: 22517427
8. Pavlopoulos GA, Malliarakis D, Papanikolaou N, Theodosiou T, Enright AJ, Iliopoulos I. Visualizing genome and systems biology: technologies, tools, implementation techniques and trends, past, present and future. GigaScience. 2015; 4:38. https://doi.org/10.1186/s13742-015-0077-2 PMID: 26309733
9. Nusrat S, Harbig T, Gehlenborg N. Tasks, Techniques, and Tools for Genomic Data Visualization. Computer Graphics Forum. 2019; 38:781–805. https://doi.org/10.1111/cgf.13727 PMID: 31768085
10. Yokoyama TT, Kasahara M. Visualization tools for human structural variations identified by whole-genome sequencing. J Hum Genet. 2020; 65:49–60. https://doi.org/10.1038/s10038-019-0687-0 PMID: 31666648
11. Yardımcı GG, Noble WS. Software tools for visualizing Hi-C data. Genome Biol. 2017; 18:26. https://doi.org/10.1186/s13059-017-1161-y PMID: 28159004
12. Peng Y, Tang S, Wang D, Zhong H, Jia H, Cai X, et al. MetaPGN: a pipeline for construction and graphical visualization of annotated pangenome networks. GigaScience. 2018; 7:giy121. https://doi.org/10.1093/gigascience/giy121 PMID: 30277499
13. Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, et al. Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. Genome Res. 2003; 13:2498–2504. https://doi.org/10.1101/gr.1239303 PMID: 14597658
14. Sedlmair M, Meyer M, Munzner T. Design Study Methodology: Reflections from the Trenches and the Stacks. IEEE Trans Vis Comput Graph. 2012; 18:2431–2440. https://doi.org/10.1109/TVCG.2012.213 PMID: 26357151
15. Stephens ZD, Lee SY, Faghri F, Campbell RH, Zhai C, Efron MJ, et al. Big Data: Astronomical or Genomical? PLoS Biol. 2015; 13:e1002195. https://doi.org/10.1371/journal.pbio.1002195 PMID: 26151137
16. Lex A, Gehlenborg N, Strobelt H, Vuillemot R, Pfister H. UpSet: Visualization of Intersecting Sets. IEEE Trans Vis Comput Graph. 2014; 20:1983–1992. https://doi.org/10.1109/TVCG.2014.2346248 PMID: 26356912
17. Kosara R. Graphs Beyond the Hairball. In: eagereyes [Internet]. 2012 [cited 2022 May 24]. Available from: https://eagereyes.org/techniques/graphs-hairball.
18. Walny J, Frisson C, West M, Kosminsky D, Knudsen S, Carpendale S, et al. Data Changes Everything: Challenges and Opportunities in Data Visualization Design Handoff. IEEE Trans Vis Comput Graph. 2020; 26:12–22. https://doi.org/10.1109/TVCG.2019.2934538 PMID: 31478857
19. Westreich ST, Nattestad M, Meyer C. BigTop: a three-dimensional virtual reality tool for GWAS visualization. BMC Bioinformatics. 2020; 21:39. https://doi.org/10.1186/s12859-020-3373-5 PMID: 32005132
20. Tang B, Li X, Li G, Tian D, Li F, Zhang Z. Delta.AR: An augmented reality-based visualization platform for 3D genome. Innovation (N Y). 2021; 2:100149. https://doi.org/10.1016/j.xinn.2021.100149 PMID: 34557786
21. Kuznetsov M, Elor A, Kurniawan S, Bosworth C, Rosen Y, Heyer N, et al. The Immersive Graph Genome Explorer: Navigating Genomics in Immersive Virtual Reality. 2021 IEEE 9th International Conference on Serious Games and Applications for Health (SeGAH). 2021. p. 1–8. https://doi.org/10.1109/SEGAH52098.2021.9551857

22. Freeman TC, Horsewell S, Patir A, Harling-Lee J, Regan T, Shih BB, et al. Graphia: A platform for the graph-based visualisation and analysis of high dimensional data. PLoS Comput Biol. 2022; 18: e1010310. https://doi.org/10.1371/journal.pcbi.1010310 PMID: 35877685

23. Munzner T. Visualization analysis and design. CRC Press; 2014.

24. Franconeri SL, Padilla LM, Shah P, Zacks JM, Hullman J. The Science of Visual Data Communication: What Works. Psychol Sci Public Interest. 2021; 22:110–161. https://doi.org/10.1177/15291006211051956 PMID: 34907835

25. Wilke CO. Fundamentals of Data Visualization. O'Reilly Media; 2019. Available from: https://clauswilke.com/dataviz/index.html.

26. Hattab G, Rhyne T-M, Heider D. Ten simple rules to colorize biological data visualization. PLoS Comput Biol. 2020; 16:e1008259. https://doi.org/10.1371/journal.pcbi.1008259 PMID: 33057327

27. Kim NW, Joyner SC, Riegelhuth A, Kim Y. Accessible Visualization: Design Space, Opportunities, and Challenges. Comput Graph Forum. 2021; 40:173–188. https://doi.org/10.1111/cgf.14298

28. Ackland P, Resnikoff S, Bourne R. World blindness and visual impairment: despite many successes, the problem is growing. Community Eye Health. 2017; 30:71–73. PMID: 29483748

29. Babich N. The 15 Rules Every UX Designer Should Know | Adobe XD Ideas. In: Ideas [Internet]. 21 Feb 2020 [cited 2022 May 29]. Available from: https://xd.adobe.com/ideas/career-tips/15-rules-every-ux-designer-know/.

30. Anscombe FJ. Graphs in statistical analysis. Am Stat. 1973; 27:17–21.

31. Matejka J, Fitzmaurice G. Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing. Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. New York, NY, USA: Association for Computing Machinery; 2017. p. 1290–1294. https://doi.org/10.1145/3025453.3025912

32. Treisman A. Preattentive processing in vision. Comput Vis Graph Image Process. 1985; 31:156–177. https://doi.org/10.1016/S0734-189X(85)80004-9

33. Boger T, Most S, Franconeri S. Jurassic Mark: Inattentional Blindness for a Datasaurus Reveals that Visualizations are Explored, not Seen. 2021 IEEE Visualization Conference (VIS). 2021. https://doi.org/10.1109/VIS49827.2021.9623273

34. Shneiderman B. The eyes have it: a task by data type taxonomy for information visualizations. Proceedings 1996 IEEE Symposium on Visual Languages. 1996. p. 336–343. https://doi.org/10.1109/VL.1996.545307

35. Pedersen BS, Tang H, Freeling M. Gobe: an interactive, web-based tool for comparative genomic visualization. Bioinformatics. 2011; 27:1015–1016. https://doi.org/10.1093/bioinformatics/btr056 PMID: 21296748

36. Yuvaraj I, Sridhar J, Michael D. Sekar K. PanGeT: Pan-genomics tool. Gene. 2017; 600:77–84. https://doi.org/10.1016/j.gene.2016.11.025 PMID: 27851981

37. Lavikka K, Oikkonen J, Lehtonen R, Hynninen J, Hietanen S, Hautaniemi S. GenomeSpy: grammar-based interactive genome visualization. F1000Res. 2020: 9. https://doi.org/10.7490/f1000research.1118237.1

38. LYi S, Wang Q, Lekschas F, Gehlenborg N. Gosling: A Grammar-based Toolkit for Scalable and Interactive Genomics Data Visualization. IEEE Trans Vis Comput Graph. 2022; 28:140–150. https://doi.org/10.1109/TVCG.2021.3114876 PMID: 34596551

39. Galletta DF, Henry R, McCoy S, Polak P. Web Site Delays: How Tolerant are Users? J Assoc Inf Syst. 2004: 5. https://doi.org/10.17705/1jais.00044

40. Mangul S, Mosqueiro T, Abdill RJ, Duong D, Mitchell K, Sarwal V, et al. Challenges and recommendations to improve the installability and archival stability of omics computational tools. PLoS Biol. 2019; 17:e3000333. https://doi.org/10.1371/journal.pbio.3000333 PMID: 31220077

41. Swaminathan R, Huang Y, Moosavinasab S, Buckley R, Bartlett CW, Lin SM. A Review on Genomics APIs. Comput Struct Biotechnol J. 2016; 14:8–15. https://doi.org/10.1016/j.csbj.2015.10.004 PMID: 26702340

42. Boettiger C. An introduction to Docker for reproducible research. SIGOPS Oper Syst Rev. 2015; 49:71–79. https://doi.org/10.1145/2723872.2723882

43. Kurtzer GM, Sochat V, Bauer MW. Singularity: Scientific containers for mobility of compute. PLoS ONE. 2017; 12:e0177459. https://doi.org/10.1371/journal.pone.0177459 PMID: 28494014

44. Anaconda Software Distribution. Anaconda Documentation. Anaconda; 2020. Available from: https://docs.anaconda.com/.

45. Grüning B, Dale R, Sjödin A, Chapman BA, Rowe J, Tomkins-Tinch CH, et al. Bioconda: sustainable and comprehensive software distribution for the life sciences. Nat Methods. 2018; 15:475–476. https://doi.org/10.1038/s41592-018-0046-7 PMID: 29967506

46. Staton M, Cannon E, Sanderson L-A, Wegrzyn J, Anderson T, Buehler S, et al. Tripal, a community update after 10 years of supporting open source, standards-based genetic, genomic and breeding databases. Brief Bioinformatics. 2021 [cited 2021 Jul 13]. https://doi.org/10.1093/bib/bbab238 PMID: 34251419

47. Hershberg EA, Stevens G, Diesh C, Xie P, De Jesus MT, Buels R, et al. JBrowseR: an R interface to the JBrowse 2 genome browser. Bioinformatics. 2021; 37:3914–3915. https://doi.org/10.1093/bioinformatics/btab459 PMID: 34196689

48. Martinez TDJ, Hershberg EA, Guo E, Stevens GJ, Diesh C, Xie P, et al. JBrowse Jupyter: A Python interface to JBrowse 2. bioRxiv. 2022. p. 2022.05.11.491552. https://doi.org/10.1101/2022.05.11.491552

49. Niu YN, Roberts EG, Denisko D, Hoffman MM. Assessing and assuring interoperability of a genomics file format. Bioinformatics. 2022:btac327. https://doi.org/10.1093/bioinformatics/btac327 PMID: 35575355

50. Cheifet B. Where is genomics going next? Genome Biol. 2019; 20:17. https://doi.org/10.1186/s13059-019-1626-2 PMID: 30670080

51. Diesh C. awesome-genome-visualization. [cited 2022 May 6]. Available from: https://cmdcolin.github.io/awesome-genome-visualization.

52. Lamprecht A-L, Garcia L, Kuzak M, Martinez C, Arcila R, Martin Del Pico E, et al. Towards FAIR principles for research software. Data Sci. 2020; 3:37–59. https://doi.org/10.3233/DS-190026

53. Katz DS, Gruenpeter M, Honeyman T. Taking a fresh look at FAIR for research software. Patterns. 2021; 2:100222. https://doi.org/10.1016/j.patter.2021.100222 PMID: 33748799

54. Social media for scientists. Nat Cell Biol. 2018; 20:1329–1329. https://doi.org/10.1038/s41556-018-0253-6 PMID: 30482942

55. Carpendale S. Evaluating Information Visualizations. In: Kerren A, Stasko JT, Fekete J-D, North C, editors. Information Visualization: Human-Centered Issues and Perspectives. Berlin, Heidelberg: Springer; 2008. p. 19–45. https://doi.org/10.1007/978-3-540-70956-5_2

56. Meyer M, Sedlmair M, Munzner T. The four-level nested model revisited: blocks and guidelines. Proceedings of the 2012 BELIV Workshop: Beyond Time and Errors—Novel Evaluation Methods for Visualization. New York, NY, USA: Association for Computing Machinery; 2012. p. 1–6. https://doi.org/10.1145/2442576.2442587

57. Zong J, Lee C, Lundgard A, Jang J, Hajas D, Satyanarayan A. Rich Screen Reader Experiences for Accessible Data Visualization. 2022 [cited 2022 May 29]. Available from: http://vis.csail.mit.edu/pubs/rich-screen-reader-vis-experiences/.

58. Elavsky F, Bennett C, Moritz D. How accessible is my visualization? Evaluating visualization accessibility with Chartability. Chartability. Rome, Italy: John Wiley & Sons; 2022. Available from: https://www.frank.computer/chartability/.

59. Kale A, Kay M, Hullman J. Visual Reasoning Strategies for Effect Size Judgments and Decisions. IEEE Trans Vis Comput Graph. 2021; 27:272–282. https://doi.org/10.1109/TVCG.2020.3030335 PMID: 33048681

60. Weiskopf D. Uncertainty Visualization: Concepts, Methods, and Applications in Biological Data Visualization. Front Bioinform. 2022; 2. Available from: https://www.frontiersin.org/article/10.3389/fbinf.2022.793819.

61. Evanko D. Data visualization: A view of every Points of View column. Protoc Methods Community. 2013 [cited 2022 May 29]. Available from: https://protocolsmethods.springernature.com/posts/43650-data-visualization-a-view-of-every-points-of-view-column.