BMC Bioinformatics

SOFTWARE Open Access

BPG: Seamless, automated and interactive visualization of scientific data

Christine P'ng[1†], Jeffrey Green[1†], Lauren C. Chong[1], Daryl Waggott[1], Stephenie D. Prokopec[1], Mehrdad Shamsi[1], Francis Nguyen[1], Denise Y. F. Mak[1], Felix Lam[1], Marco A. Albuquerque[1], Ying Wu[1], Esther H. Jung[1], Maud H. W. Starmans[1], Michelle A. Chan-Seng-Yue[1], Cindy Q. Yao[1,2], Bianca Liang[1], Emilie Lalonde[1,2], Syed Haider[1], Nicole A. Simone[1], Dorota Sendorek[1], Kenneth C. Chu[1], Nathalie C. Moon[1], Natalie S. Fox[1,2], Michal R. Grzadkowski[1], Nicholas J. Harding[1], Clement Fung[1], Amanda R. Murdoch[1], Kathleen E. Houlahan[1,2], Jianxin Wang[1,4], David R. Garcia[1], Richard de Borja[1], Ren X. Sun[1,3], Xihui Lin[1], Gregory M. Chen[1], Aileen Lu[1,3], Yu-Jia Shiah[1,2], Amin Zia[1], Ryan Kearns[1] and Paul C. Boutros[1,2,3,5,6,7,8*]

Abstract

Background: We introduce BPG, a framework for generating publication-quality, highly-customizable plots in the R statistical environment.

Results: This open-source package includes multiple methods of displaying high-dimensional datasets and facilitates generation of complex multi-panel figures, making it suitable for complex datasets. A web-based interactive tool allows online figure customization, from which R code can be downloaded for integration with computational pipelines.

Conclusion: BPG provides a new approach for linking interactive and scripted data visualization and is available at http://labs.oicr.on.ca/boutros-lab/software/bpg or via CRAN at https://cran.r-project.org/web/packages/BoutrosLab.plotting.general

Keywords: Data-visualization, Interactive plotting, Software, Web-resources

Background

Biological experiments are increasingly generating large, multifaceted datasets. Exploring such data and communicating observations is, in turn, growing more difficult and the need for robust scientific data-visualization is accelerating [1–4]. Myriad data visualization tools exist, particularly as web-based interfaces and local software packages. Unfortunately these often do not integrate easily into R-based statistical pipelines, such as the widely used Bioconductor [5]. Within R, many visualization packages exist, including base graphics [6], ggplot2 [7], lattice [8], Sushi [9], circlize [10], multiDimBio [11], NetBioV [12], GenomeGraphs [13] and ggbio [14]. There is also a broad range of activity-specific visualization packages focused on specific tasks or analysis-types [15–24]. Some of these lack publication-quality defaults such as high-resolution, appropriate label-sizing and default colour palettes appropriate for gray-scale use and visible for those with red-green colour-blindness. Many can require significant parameterization. Others contain limited plot types, provide limited scope for automatic generation of multi-panel figures or are constrained to specific data-types. Few allow interactive visualization, where specific plot elements can be highlighted and the set of parameters available to customize them automatically identified and allowing interactive generation of R code through a GUI interface that visualizes plot changes in real-time. Thus while each of these visualization packages has significant value and user-bases, each lacks some features beneficial for computational biologists and data scientists.

Good visualization software must create a wide variety of chart-types in order to match the diversity of data-types available. It should provide flexible parametrization for

* Correspondence: pboutros@mednet.ucla.edu
†Christine P'ng and Jeffrey Green contributed equally to this work.
[1]Ontario Institute for Cancer Research, Toronto, Canada
[2]Department of Medical Biophysics, University of Toronto, Toronto, Canada
Full list of author information is available at the end of the article

highly customized figures and allow for multiple output formats while employing reasonable, publication-appropriate default settings, such as producing high resolution output. In addition, it should integrate seamlessly with existing computational pipelines while also providing an easily intuitive, interactive mode. There should be an ability to transition between pipeline and interactive mode, allowing cyclical development. Finally, good design principles should be encouraged, such as suggesting appropriate color choices and layouts for specific use-cases. To help users quickly gain proficiency, detailed examples, tutorials, an ability for real-time interactive plot-tuning and an application programming interface (API) are required. To date, no existing visualization suite fully fills these needs.

Implementation

To address this gap, we have created the BPG (BoutrosLab.plotting.general) library, which is implemented in R using the grid graphics system and lattice framework. It generates a broad suite of chart-types, ranging from common plots such as bar charts and box plots to more specialized plots, such as Manhattan plots (Fig. 1; code is in Additional file 1). These include some novel plot-types, including the dotmap: a grid of circles inset inside a matrix, allowing representation of four-dimensional data (Fig. 1n). Each plotting function is highly parameterized, allowing precise control over plot aesthetics. The default parameters for BPG produce high resolution (1600 dpi) TIFF files, appropriate for publication. The file type is specified simply by specifying a file extension. Other default values contribute to graphical consistency including: the inclusion of tick marks, selection of fonts and default colors that work together to create a consistent plotting style across a project. Default values have been optimized to generate high-quality figures, reducing the need for manual tuning. However, even good defaults will not be appropriate for every use-case [15]. Additional file 2: Figure S1 demonstrates a single scatter plot created using four separate graphics frameworks with either default or optimized settings: BPG, base R graphics, ggplot2, and lattice. BPG required half as much code as the other frameworks for both default and optimized plots, while producing plots with at least similar quality (Additional file 3).

To facilitate rapid graphical prototyping, an online interactive plotting interface was created (http://bpg.oicr.on.ca). This interface allows users to easily and rapidly see the results of adjusting parameter values, thereby encouraging precise improvement of plot aesthetics. The R code generated by this interface is also made available for download, as is a methods paragraph allowing careful

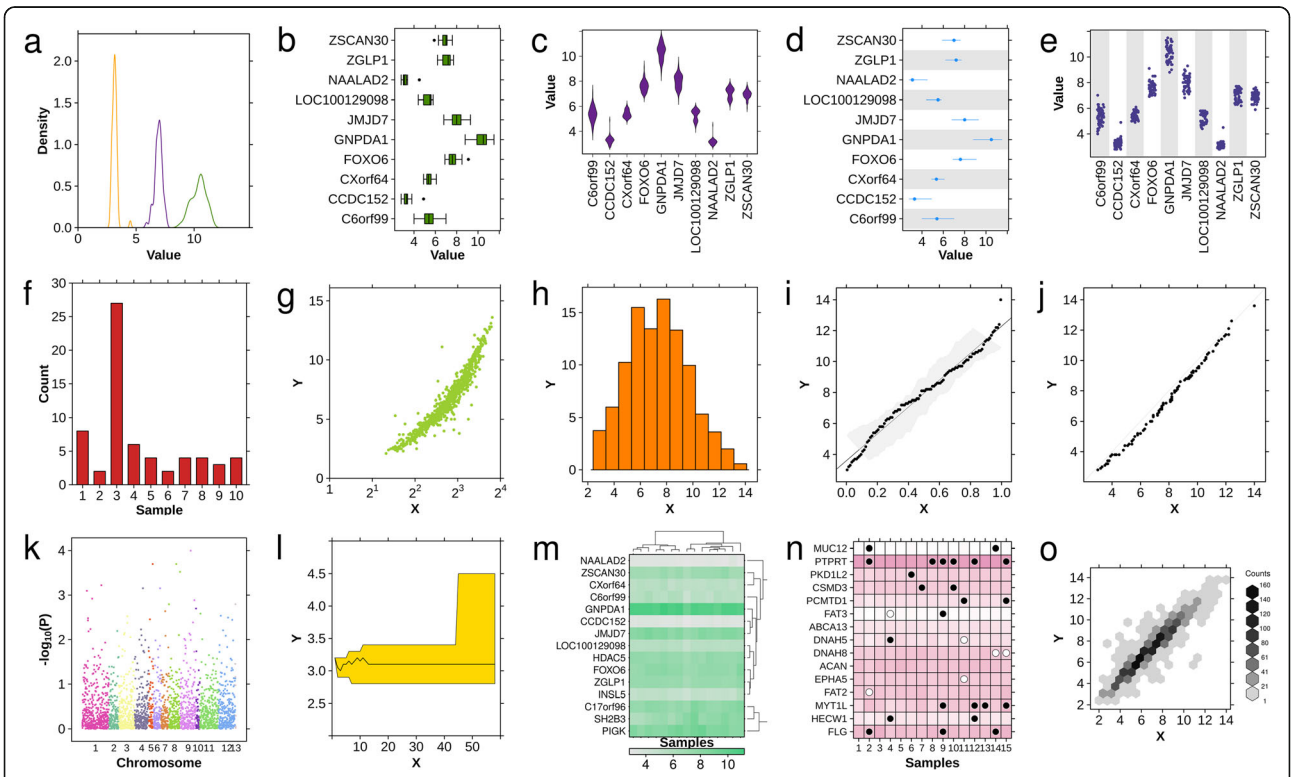

Fig. 1 Available chart-types. The basic chart-types available in BPG: **a** density plot, **b** boxplot, **c** violin plot, **d** segplot, **e** strip plot, **f** barplot, **g** scatterplot, **h** histogram, **i** qqplot fit, **j** qqplot comparison, **k** Manhattan plot, **l** polygon plot, **m** heatmap, **n** dotmap and **o** hexbinplot. All plots are based upon the datasets included in the BPG package and code is given in Additional file 1

reporting of plotting options. A public web-interface is available, and local interfaces can be easily created.

One critical feature of BPG is its ability to combine multiple plots into a single figure: a technique used widely in publications. This is accomplished by the create.multiplot function, which automatically aligns plots and standardizes parameters such as line widths and font sizes across all plot elements within the final figure. This replaces the slow and error-prone manual combination of figures using PowerPoint, LaTeX or other similar software, or the time-consuming parameterization of manually align plot locations directly in R with functions like layout(). The necessity of combining multiple plots arises from the complexity of datasets – with high dimensional data, it is often difficult to convey all relevant information within a single chart-type. Combining multiple chart-types allows more in depth visualization of the data. For example, one plot might convey the number of mutations present in different samples; a second plot could add the proportion of different mutation types, while a third could give sample-level information (Fig. 2). We have included a series of example datasets directly in BPG, including the one used to create the visualization in Fig. 2, and the source-code for creating this plot from these datasets is given in Additional file 4.

A number of utility functions in BPG assist in plot optimization, such as producing legends and covariate bars, or formatting text with scientific notation for *p*-values. One difficult step in creating figures is the

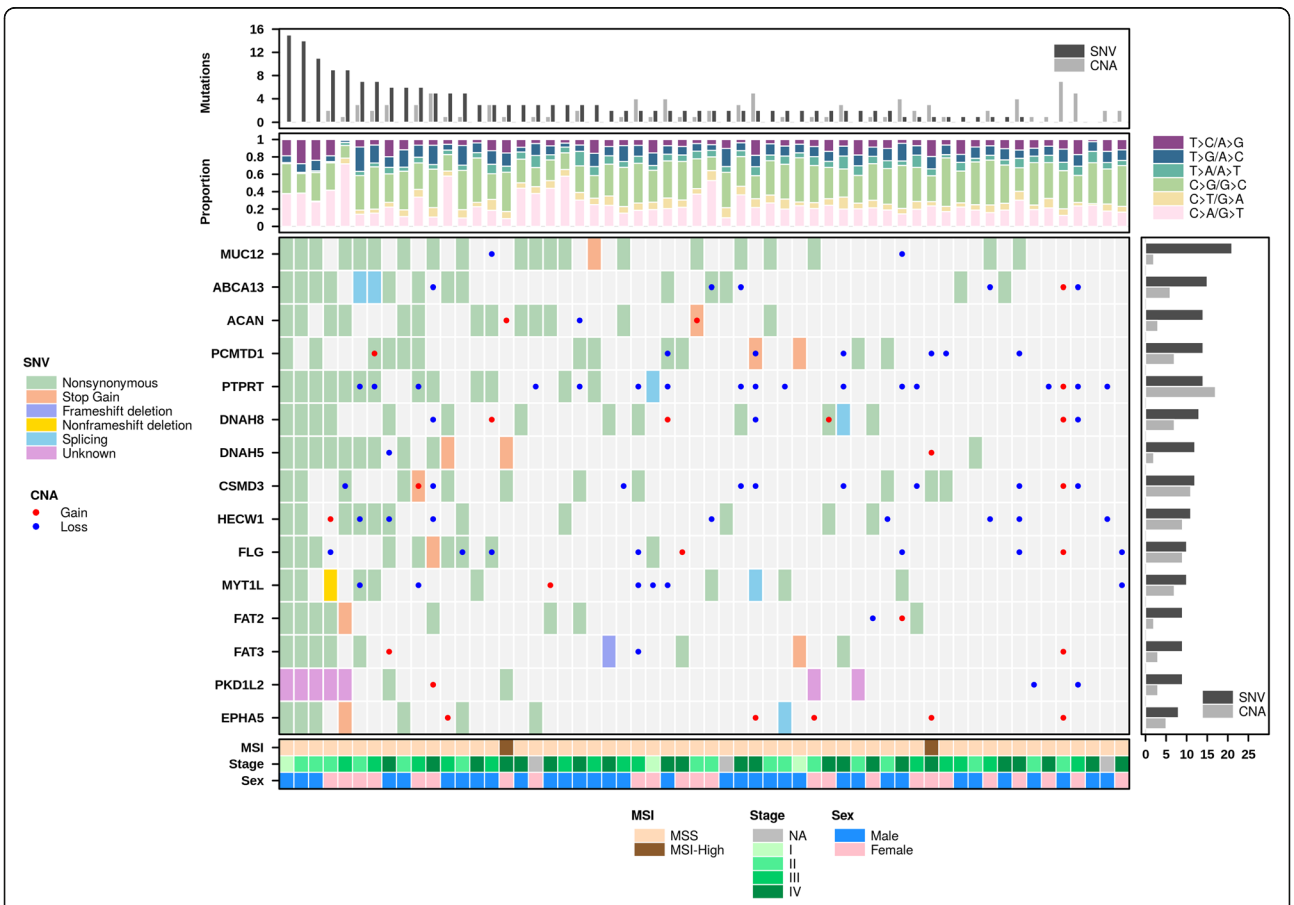

Fig. 2 Multiplot example. The create.multiplot function is able to join multiple chart-types together into a single figure. In this example, a central dotmap conveys the somatic mutations present in a selection of genes (y-axis) for a number of colorectal tumours (x-axis), while adjacent barplots and heatmaps provide additional information. Within this central dotmap, shaded cells reflect single nucleotide variants (SNVs), while dots in cells reflect copy number aberration (CNAs), which some patients have both types of aberration in a single gene (shaded cells harbouring a dot). The colour of the cell or dot indicates the specific type of mutation, using the legend on the left. The bottom heatmap shows key clinical information about each patient, including their Sex, the Stage of their disease and their microsatellite status (instable, MSI; stable MSS). The barplot to the right shows the percentage of patients with a SNV or a CNA in that gene. The barplot at the top, equivalently, shows the number of SNVs and CNAs for each patient. Finally, the second barplot from the top categorizes all SNVs based on the type of base-change that mutation reflects, showing their proportion as a fraction of the total mutation number. Code used to generate this figure is available in Additional file 4

selection of color schemes that are both pleasing and interpretable [25, 26]. BPG provides a suite of 45 color palettes including qualitative, sequential, and diverging color schemes [27], shown in Additional file 5: Figure S2. Many optimized color schemes exist for numerous use cases including tissue types, chromosomes and mutation types. The default.colors function produces a warning when a requested color scheme is not grey-scale compatible, a common concern for figures reproduced in black and white. This is determined by converting each color to a grey value between 1 and 100, and indicating differences of < 10 as not grey-scale compatible to approximate a color scheme's visibility when printed in grey-scale. To facilitate reproducibility, image metadata is automatically generated for all plots, creating descriptors such as software and operating system versions.

Results

Extensive documentation is provided to help new users learn how to use BPG. To assist researchers in determining which chart-type is appropriate for their dataset, we provide plotting examples in the documentation which are derived from a real dataset and a plotting guide is included to explain the intended use-case of each function. This guide also contains explanations of typography, basic color theory and layout design which help to improve the design of figures [28, 29]. In addition, an online API is available with both simple and complex use-case examples for each plot-type to help users quickly learn the range of functionality available.

Conclusions

BPG has been used in over 60 publications to date (Additional file 6: Table S1) [30–35]. These plotting functions have been integrated into numerous R analysis pipelines for automated figure generation as part of the analysis of large –omic data. The plots created by this package are reproducible and maintain a consistent aesthetic. We believe that BPG will facilitate improved visualization and communication of complex datasets.

Additional files

Additional file 1: Code to generate Fig. 1. (TXT 14 kb)

Additional file 2: Figure S1. Comparison of graphical software options in R. (a-b) are created with base R graphics, (c-d) are created using ggplot2, (e-f) are made in lattice and (g-h) use BPG. The first plot in each pair uses default settings, while the second plot has been adjusted for font sizes, axes ranges, tick mark locations, grid lines, diagonal lines, background shading and highlighted datapoints. The number of lines of code used to create default plots are: 10 for base R, 10 for ggplot2, 14 for lattice, and 5 for BPG. The customized plots use 73 lines for base R, 83 for ggplot2, 86 for lattice, and 42 for BPG. Code for generating this figure is provided in Additional file 3. (TIFF 1590 kb)

Additional file 3: Code to generate Additional file 2: Figure S1. (TXT 6 kb)

Additional file 4: Code to generate Fig. 2. (TXT 11 kb)

Additional file 5: Figure S2. Color palettes. Color palettes are provided using the default.colors function for (a) generic use-cases and force.color.scheme for (b) specific use-cases. This display is generated using the show.available.palettes function. Interactive display of colors is also available using the display.colors function. (TIF 1036 kb)

Additional file 6: Table S1. Publications using BPG. (DOC 67 kb)

Abbreviations

API: Application Programming Interface; BPG: BoutrosLab.Plotting.General; CNA: Copy Number Aberration; CRAN: Comprehensive R Archive Network; DPI: Dots per Inch; GUI: Graphical User Interface; MSI: Microsatellite Instable; MSS: Microsatellite Stable; SNV: Single Nucleotide Variant

Acknowledgements

The authors thank all members of the Boutros lab for their assistance in testing and improving the software and our many collaborators for their suggestions and support, particular Drs. Robert Bristow, Michael Fraser, Raimo Pohjanvirta, Allan Okey and Linda Penn. We thank the OICR webdev and systems teams for support, particularly Joseph Yamada and Rob Naccarato.

Funding

This study was conducted with the support of the Ontario Institute for Cancer Research to PCB through funding provided by the Government of Ontario. This work was supported by Prostate Cancer Canada and is proudly funded by the Movember Foundation - Grant number RS2014–01. PCB was supported by a Terry Fox Research Institute New Investigator Award and a Canadian Institutes of Health Research (CIHR) New Investigator Award. This research is funded by the Canadian Cancer Society (grant number 702528). This work was supported by the Discovery Frontiers: Advancing Big Data Science in Genomics Research program, which is jointly funded by the Natural Sciences and Engineering Research Council (NSERC) of Canada, the Canadian Institutes of Health Research (CIHR), Genome Canada, and the Canada Foundation for Innovation (CFI). This project was supported by Genome Canada through a Large-Scale Applied Project contract to PCB and Drs. Sohrab Shah and Ryan Morin. This study was conducted with the support of the Ontario Genomics Institute (to CP), the Canadian Breast Cancer Foundation (to CQY), the Ontario Graduate Scholarship (to EL), the Oncology Research and Methods Training Program (to YW, NCM and XL), the Canadian Institutes of Health Research (CIHR) (to EL, KEH, NSF and GMC), the Medical Biophysics Excellence University of Toronto Fund Scholarship (to NSF) and the CTMM framework (AIRFORCE project) and EU 7th framework program (ARTFORCE) to MHWS. This work was funded by the Government of Canada through Genome Canada and the Ontario Genomics Institute (OGI-125). The funders played no role in the design of the study, nor in the writing of the manuscript.

Availability of data and materials

Not applicable.

Authors' contributions

PCB conceived of the project. All authors wrote software, documentation and debugged. FL and NAS developed the interactive plotting method, which JG significantly extended. CP wrote the manuscript, which all authors edited and approved.

Ethics approval and consent to participate

Not Applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Author details
[1]Ontario Institute for Cancer Research, Toronto, Canada. [2]Department of Medical Biophysics, University of Toronto, Toronto, Canada. [3]Department of Pharmacology and Toxicology, University of Toronto, Toronto, Canada. [4]Present address: Center for Computational Research, Buffalo Institute for Genomics and Data Analytics, NYS Center for Excellence in Bioinformatics & Life Science, University at Buffalo, Buffalo, USA. [5]Department of Human Genetics, University of California, Los Angeles, USA. [6]Department of Urology, University of California, Los Angeles, USA. [7]Institute for Precision Health, University of California, Los Angeles, USA. [8]Jonsson Comprehensive Cancer Center, University of California, Los Angeles, USA.

Received: 11 May 2018 Accepted: 4 January 2019
Published online: 21 January 2019

References

1. Grinstein G, Trutschl M, Cvek U. Proceedings of the visual data mining workshop. KDD. 2001:7–19.
2. Anscombe FJ. Graphs in Statistical Analysis. Am Stat. 1973;27:17–21.
3. Shoresh N, Wong B. Data exploration. Nat Methods. 2012;9:5.
4. O'Donoghue SI, et al. Visualizing biological data-now and in the future. Nat Methods. 2010;7:S2–4.
5. Gentleman RC, et al. Bioconductor: open software development for computational biology and bioinformatics. Genome Biol. 2004;5:R80.
6. R Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. 2014. http://www.R-project.org/. Accessed 10 Jan 2019.
7. Wickham, H. ggplot2: elegant graphics for data analysis. Springer, New York, USA, 2009.
8. Sarkar, D. Lattice: multivariate data visualization with R. Springer, New York, USA, 2008.
9. Phanstiel DH, Boyle AP, Araya CL, Snyder MP. Sushi R: flexible, quantitative and integrative genomic visualizations for publication-quality multi-panel figures. Bioinformatics. 2014;30:2808–10.
10. Gu Z, Gu L, Eils R, Schlesner M, Brors B. circlize Implements and enhances circular visualization in R. Bioinformatics. 2014;30:2811–2.
11. Scarpino SV, Gillette R, Crews D. (R package). 2013. http://cran.r-project.org/web/packages/multiDimBio/index.html. Accessed 10 Jan 2019.
12. Tripathi S, Dehmer M, Emmert-Streib F. NetBioV: an R package for visualizing large network data in biology and medicine. Bioinformatics. 2014;30:2834–6.
13. Durinck S, Bullard J, Spellman PT, Dudoi S. GenomeGraphs: integrated genomic data visualization with R. BMC Bioinformatics. 2009;10:2.
14. Yin T, Cook D, Lawrence M. ggbio: an R package for extending the grammar of graphics for genomic data. Genome Biol. 2012;13:R77.
15. He W, Zhao S, Zhang C, Vincent MS, Zhang B. QuickRNASeq: Guide for Pipeline Implementation and for Interactive Results Visualization. Methods Mol Biol. 2018;1751:57–70.
16. Waggott D, Chu K, Yin S, Wouters BG, Liu FF, Boutros PC. NanoStringNorm: an extensible R package for the pre-processing of NanoString mRNA and miRNA data. Bioinformatics. 2012;28(11):1546–8.
17. Sendorek DH, Lalonde E, Yao CQ, Sabelnykova VY, Bristow RG, Boutros PC. NanoStringNormCNV: pre-processing of NanoString CNV data. Bioinformatics. 2018;34(6):1034–6.
18. Ranjitha Dhanasekaran A, Gardiner KJ. RPPAware: A software suite to preprocess, analyze and visualize reverse phase protein array data. J Bioinform Comput Biol. 1850001 (2018).
19. Lee TR, Ahn JM, Kim G, Kim S. IVAG: An Integrative Visualization Application for Various Types of Genomic Data Based on R-Shiny and the Docker Platform. Genomics Inform. 2017;15(4):178–82.
20. Renault V, Tost J, Pichon F, Wang-Renault SF, Letouzé E, Imbeaud S, Zucman-Rossi J, Deleuze JF, How-Kit A. aCNViewer: Comprehensive genome-wide visualization of absolute copy number and copy neutral variations. PLoS One. 2017;12(12):e0189334.
21. Zhu X, Wolfgruber TK, Tasato A, Arisdakessian C, Garmire DG, Garmire LX. Granatum: a graphical single-cell RNA-Seq analysis pipeline for genomics scientists. Genome Med. 2017;9(1):108.
22. Jalili V, Matteucci M, Masseroli M. Ceri S. Explorative visual analytics on interval-based genomic data and their metadata. BMC Bioinformatics. 2017;18(1):536.
23. Turner D, Sutton JM, Reynolds DM, Sim EM, Petty NK. Visualization of Phage Genomic Data: Comparative Genomics and Publication-Quality Diagrams. Methods Mol Biol. 2018;1681:239–60.
24. Li J, Akbani R, Zhao W, Lu Y, Weinstein JN, Mills GB, Liang H. Explore, Visualize, and Analyze Functional Cancer Proteomic Data Using the Cancer Proteome Atlas. Cancer Res. 2017;77(21):e51–4.
25. Rougier NP, Droettboom M, Bourne PE. Ten Simple Rules for Better Figures. PLoS Comput Biol. 2014;10:e1003833.
26. Wong B. Color coding. Nat Methods. 2010;7:573.
27. Wong B. Color blindness. Nat Methods. 2011;8:441.
28. Harrower H, Brewer CA. ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps. Cartogr J. 2003;40:27–37.
29. Wong B. Points of review (part 1). Nat Methods. 2011;8:101.
30. Haider S, et al. Pathway-based subnetworks enable cross-disease biomarker discovery. Nat Commun. 2018;9(1):4746.
31. Lee AY, et al. Combining accurate tumor genome simulation with crowdsourcing to benchmark somatic structural variant detection. Genome Biol. 2018;19(1):188.
32. Espiritu SMG, et al. The Evolutionary Landscape of Localized Prostate Cancers Drives Clinical Aggression. Cell. 2018;173(4):1003–13.
33. Fraser M, et al. Genomic hallmarks of localized, non-indolent prostate cancer. Nature. 2017;541(7637):359–64.
34. Boutros PC, et al. Spatial genomic heterogeneity within localized, multifocal prostate cancer. Nat Genet. 2015;47(7):736–45.
35. Ewing AD, et al. Combining tumor genome simulation with crowdsourcing to benchmark somatic single-nucleotide-variant detection. Nat Meth. 2015;12:623–30.