Harvard Data Science Review • Issue 3.2, Spring 2021

On Democratizing Data Science: Some iNZights Into Empowering the Many

Chris J. Wild¹ Tom Elliott¹ Andrew Sporle^{1,2}

¹Department of Statistics, University of Auckland, Auckland, New Zealand,

²Research Data Alliance, European Commission and American National Science Foundation and National Institute of Standards and Technology and the Australian Department of Innovation

Published on: Jun 07, 2021

DOI: <https://doi.org/10.1162/99608f92.85206ff9>

License: [Creative Commons Attribution 4.0 International License \(CC-BY 4.0\)](https://creativecommons.org/licenses/by/4.0/)

Introduction

We have been interested in Berkeley's efforts in advancing undergraduate data science for several years because they are taking a fully integrated approach rather than largely just assembling existing building blocks from statistics and computer science. It is great to see, in Adhikari et al.'s article "[Interleaving Computational and Inferential Thinking in an Undergraduate Data Science Curriculum](#)" (this issue), such a nice whole-program overview together with many of its very compelling philosophical underpinnings. The program is driven by real-world investigations, attends to integrating statistical and computational thinking, is rich, deep and varied, and steeped in application areas through connector courses. We share a belief in simulation-based inference as the most accessible way of introducing beginners to inferential ideas. Our focus has been on conceptual development via unifying dynamic visualizations driven by a graphical user interface (GUI) (see Wild et al., 2017) rather than on randomization as a computational device. Interesting qualitative research is reported by Fergusson and Pfannkuch (in press) on teachers participating in activities involving both tool-driven visualization and computational approaches. They uncovered holes in teacher understanding with both approaches, something that tended to be rectified when participants made mappings between what was going on in each approach at each step of the process.

The Berkeley program, starting with Data 8, blends a computer science ethos of developing tool builders with a statistical ethos of developing inferential thinkers, a very appropriate goal for data science education. But there is another relevant contrast: that between tool builders and tool users. The essence of a number of definitions of data science is 'the science of learning from data.' For this, skills in tool building and skills in tool use are both simply enablers—a means toward the practical ends of 'learning from data.'

The majority of applied statisticians spend much more of their time as tool users than as tool builders (with notable exceptions of course). Moreover, the programming they do makes heavy use of calls to high-level functions (themselves tools), written by others, that automate a lot of sophisticated processing. There are good reasons for this apart from personal inclinations. 'Do it yourself' is very bad practice when well-tested software exists. It is not just an inefficient use of time, it also leads to large numbers of unnecessary errors. (Additionally, large numbers of learning-from-data problems can be solved using existing tools.) Students in Data 8, and students almost universally, tend to learn to program on problems where good solutions already exist. In so doing, they learn skills whose value is not in being able to solve the specific problems they are trained on, but in fostering the underlying thinking modes, capabilities, and confidence to tackle unforeseen problems—ones that lack good software—later in life. We wonder how the Berkeley program addresses this dilemma and habituates its lessons.

Our Context

Third author Andrew Sporle is involved with statistical and health agencies in several small nations in the Pacific. Overwhelmed by the information needs of their own societies and by the reporting demands of key international agencies such as the United Nations, they face the triple burden of distance, small workforces, and insufficient funding. As a result, these countries find it difficult to recruit and retain people with good data science skills, or afford to contract outside expertise. There are increasing numbers of community groups; for example, Andrew is involved with national movements linked by the [Global Indigenous Data Alliance](#), who want to make data resources to contribute to addressing their administrative, planning, and lobbying needs, but don't have the skills or ability to pay for them. The situation is not substantially different for subject matter researchers from many areas who have projects with potential societal value, but who lack funds or access to technical skills.

When it comes to data science needs, Greta Thunberg's global-warming image is most apposite, "While we may all be in the same storm, we are not all in the same boat." The unmet data-science needs of those lacking in money and data education are every bit as real and important as those who have more. While we admire the altruism of volunteer groups like [Statistics Without Borders](#), who provide pro bono services in statistics and data science, there will never be enough altruistic volunteers to fill the gaps. It is important, therefore, to empower more people to do more for themselves. In particular, government agencies in the Pacific and other groups need to be empowered so their already-busy generalists can do things that currently only specialists can do.

Generalists and part-time statisticians are time-poor and have very little data-related education. Addressing the lack-of-knowledge problem runs headlong into the lack-of-time problem. Generalists do many different things and tend to work in specific processes infrequently. So, working around the rapid fading of memories of how to do things and what they mean is also a major factor. This collides with 'the problem of names.' When relying on programming, and also on almost all GUI systems, you basically cannot do anything until you know what you want to do and know and remember its name. This is a significant barrier to getting started and also results in significant time losses getting back up to speed after a period of inactivity. These considerations disqualify writing code as a realistic way forward for most members of our audiences. To empower them we need tools that reduce the demands for up-front knowledge and place much less reliance on leaky memories. But a minority will face tasks that require automating sequences of operations, an enterprise that does cry out for code.

We will next discuss some strategies we are using in our open source iNZight project (<https://inzight.nz/>, <https://github.com/iNZightVIT>; Elliott et al., 2021b; Wild & Ridgway, in press) to address these problems before returning to pedagogical issues and connections with the Berkeley program in the closing section of our discussion.

Introducing iNZight

The name iNZight is a play on words blending ‘insight’ with the initials of our home country of New Zealand. iNZight is both a vehicle for exploring and prototyping ideas about increasing accessibility to data capabilities (the best way to think hard about something is to build it) and a very useful working system. It is a GUI system written in R and comes in two versions, a desktop version that requires installation and an online version (iNZight Lite) that just needs a browser. They share the same R backend but have different interfaces, the desktop interface is based on RGtk2 while the online version is a large R Shiny app.

Our overarching goal is to facilitate serious capabilities for obtaining human insights from data for a broad spectrum of people, by reducing *prior*-learning requirements to a minimum, and eliminating almost all requirements to know and remember the names of things. Our approach is illustrated in the description that follows of how iNZight works in its basic mode.

Basic Mode

The basic mode of iNZight provides visualizations and analysis for data in standard rectangular units or cases-by-variables format. In basic mode, the program is driven by assigning roles to variables with immediate responses determined by variable type and defaults (variable types currently recognized are: categorical, numeric, and date-time). The underlying metaphor is, ‘**Tell me about ...**,’ so tell me about a variable, or a relationship between two variables—either alone or subsetted or faceted by other variables. Here, ‘Tell me about’ is really shorthand for ‘*Show and Tell*’ because what is delivered instantly is graphics. We believe graphics are the most accessible artifacts for broad audiences, and also that people are less likely to do silly things when they look at their data first.

To obtain numeric information, you explicitly ask for it by clicking *Get Summary* or *Get Inference*. The underlying metaphor for both is, ‘*give me the types of information analysts generally want to see in a situation like this.*’ So, *Get Inference* gives users their analyses of variance, chi-squared tests, regression panels, and so on, in situations where they are appropriate, accompanied by sets of relevant confidence intervals. But users don’t have to know or remember what to ask for and how to ask for it, thus reducing the barriers to entry and to reentry after a time gap. After-the-fact information concerning ‘how can I read this and what does it mean?’ has compelling relevance when you have output in front of you. The up-front knowledge required by iNZight in its basic mode is simply some high-level familiarity with rectangular data, variables, and the ability to identify situations where they might want to override a variable-type default (e.g., numeric codes used as group labels).

Here we have automated everything using defaults and delivered immediate results. This, however, begs the question, ‘**How else can I look at this?**’ A plot-type selection box allows scrolling (with a mouse wheel or arrow key) through plots from all the applicable graph-types in the [Financial Times Visual Vocabulary](#) with some additions. Use of defaults also begs, ‘**How else can I do this?**’ that is, making available alternatives to default methods. Inferences, for example, may be based on normal theory or bootstrapping, and a switch can

turn on epidemiological versions of outputs (odds-ratios, risk-ratios, etc.). Options for plot enhancements are extensive, including: overriding small-data/large-data defaults for graphics; information-adding mechanisms like coloring and sizing by further variables (also plotting symbol by variable); adding trends and other inferential markup; identification and labeling of points; motion (playing through a set of faceted graphs); interactivity; and many additional modifications that might be desired for aesthetic reasons.

For the look and feel of iNZight, including screen shots, and so forth, and technical design and implementation information, see Elliott et al. (2021b).

R Code History

Although iNZight is primarily a GUI system, we also want to provide facilities to ease interested users into code. iNZight's main modules construct and execute R code. The code for anything that changes the data is automatically stored and made available to the user via 'R code history.' The code for other actions such as graphics is also stored if the user so requests. It is not a facility provided for beginners—it is there for more advanced users: to provide audit trails, as an aid to learning R and how to do things in R, and as a way of generating code elements for use in R programs, for example, one that automates a sequence of steps.

Data Input and Data Wrangling

The aim here is to make everything painless for almost all users almost all of the time by automating everything we can. On import, a data set is assumed to have a rectangular case-by-variables format, but wrangling facilities allow many types of transformation of the input data set. The interface calls a smart read–function that, by default, uses filename extensions to decide what R import function to use. The chosen call is constructed and also stored for the R code history. Currently, iNZight supports files in CSV, tab-delimited text, Excel, SAS, Stata, SPSS, R-data, and JSON formats.

iNZight has an extensive repertoire of data-manipulation methods, including those for filtering, aggregation, reshaping, and joins, and also extensive variable-transformation and construction capabilities, including most of those operations described in Wickham and Grolemund (2017). Users are led through dialogs (linked to help files) to specify actions. Also included are quick data set reports and data-validation facilities. For more complex wrangling operations, users are shown pre- and post-operation views of the data before committing to an operation.

The main methods are based on tidyverse R code (Wickham et al., 2019) and, behind the scenes, the constructed tidyverse code for the operation is written to the code history—thus providing a means of generating R code for any of these operations for subsequent use in an R program.

Facilities for Complex Survey Data

Agencies, social researchers, and population-health researchers are often working with data obtained from complex survey designs. Ignoring the design structure in analysis generally leads to misleading results. iNZight handles survey designs behind the scenes, requiring the user to specify the design structure either manually via dialog panels or by importing a special survey design file. Once specified, the user can forget about the survey design and use iNZight as normal for basic graphics and statistics, generalized linear modeling, and even survival analysis. Code in the R history contains the survey-customized R code.

The newly introduced survey design files will permit data publishers to distribute data in a way that enables ‘data analysis as usual’ without users having to confront design issues and their implications for analysis.

‘Advanced’ Modules

We have added capability for many more specialist data types and analyses that fall outside the basic mode accommodated by including additional modules. All endeavor to keep *prior*-learning requirements and also any need to know/remember names to an absolute minimum, and to provide immediate output by using defaults to automate everything we can. Additionally, they all invoke the metaphors described, ‘Tell me about ...,’ ‘How else can I look at that?’ ‘How else can I do that?’ and adding, ‘What more can I do next?’

For time-poor people, just using different tools for different jobs is suboptimal. Why? Every time someone reaches for a new system, there is substantial downtime in figuring out/remembering how to get your data in and in the right format, and how the new system ‘thinks.’ So there are understandability and time-efficiency benefits in having a tool set that employs metaphors as consistently as possible across modules and that shares import/export/output and data-wrangling facilities.

Currently included in iNZight are modules for generalized linear modeling (for ‘regular’ and complex survey data), seasonal time series, multiple-response data, data on maps, a multivariate graphics module, and an experimental-design module (online version only). Additionally, we have student-built prototypes for text harvesting and text analytics (a highly featured module ready for exposure), network data, longitudinal data, novel graphics for hierarchical data, supervised learning (predictive modeling), and unsupervised learning (primarily cluster analysis).

For almost all of its life, the iNZight project has been an educationally focused project driven by first author Wild’s desire to find ways for students, early in their education, to experience much more of what is possible in the data world more quickly and with software-enabled capabilities for doing a lot of it(cf. Wild, 2015). Third author Sporle spotted synergies between this and his own efforts to help people in the Pacific and community groups address data needs, and also began bringing in funding that enabled expanding our vision. Most of the work on these modules arose from the original vision, including Wild’s involvement in the [International Data Science in Schools Project \(IDSSP\)](#) when he was trying to get to grips with what could be made accessible to a

broad audience at senior high-school levels using short software-assisted teaching modules (the answer, suggested by these prototypes, is ‘a lot!’).

Our goal is to enable serious practical capabilities across a broad spectrum of areas extremely quickly and with minimal learning curves. In each case, we try to identify what is special about a type of data, the biggest high-level ideas that define it, and what you might want to do with it. Our example that gets closest to our desired ‘level’ is the discussion of maps in Wild (2020). We prioritize areas and capabilities that have reasonably understandable outputs, so much of the emphasis is on graphics. Then we try to think through which of these capabilities we can make accessible via a structured interface that provides a top-down way of thinking about and working with the data type. Context-aware choice-sets provide the options that are relevant to where in the process the user currently is, thus invoking sequential revelation of, ‘What can I do with that?’ and ‘What can/should I do next?’ All of this is helped by the fact that, for many people today, buttons and controls prompt, ‘I wonder what that does?’ and a desire to press them to find out.

Having a design with many largely self-contained modules also makes adding features to provide for particular needs of a particular audience more achievable for a very small, mostly part-time, team with spotty funding. For example, work is currently underway on a Bayesian small-area estimation module primarily addressed at the small Pacific nations. Then we will extend our seasonal time series module to more comprehensively cater for their time-series needs. For more about the software, work in progress, and near-term ambitions, see Elliott et al. (2021b).

Usage

iNZight’s user base to date is very diverse despite it not having yet been heavily promoted. It is widely used in New Zealand high schools and, at university level, it is in use even at the final year of undergraduate studies. Beyond this, the only way we know about users is when they ask for help. They include researchers across the world from many fields, people working in agencies, local government, companies and nonprofits, administrators, and also data journalists, courtesy of Alberto Cairo’s inclusion of iNZight among his recommended tools (cf. Philp, 2020). Recent help requests came from an Olympic sailor and a writer of children’s books!

More About R Code

In addition to the R code history function, if activated in the Preferences panel, code windows appear for plots and in text-output windows. These display the function call that generated the current output and allow it to be stored, or modified and rerun. Settings in the GUI determine the function call and the output. Changing and rerunning a valid function call changes the settings in the GUI to match the user’s code. The reasons for keeping things this simple are educational with coding-beginners in mind: ‘the code that does it’ is always in view to foster learning by osmosis; the mappings between GUI settings, argument values of the function call, and output are direct, to foster seeing the relationships between them; because the system is responsive, each

request in the GUI for a minor change or customization triggers an instant change both in the output and in the code, also helping highlight what particular code elements do; and opportunity is provided to experiment with the code within a familiar environment with expected behaviors and expected outputs. Restricting what is always being shown to just the current function call keeps things simple and makes the mappings between code and GUI settings more obvious. (Other strategies are required for learning to ‘string together’ code.) Similar functionality is being introduced for data-wrangling operations. A more extensive role for code interactions with the system is under consideration but we are at present still somewhat ambivalent about this.

A side effect of our GUI philosophies is new extremely high-level R functions like `inzplot()`, `inzsummary()`, and `inzinference()` in the `iNZightPlots` package (Elliott et al., 2021a) that implement the behavior of iNZight’s basic mode. This provides options in teaching data science for achieving in a programming environment many of the prior-knowledge and memory-load efficiencies of the GUI environment. In a programming environment, the interface’s what-can-I-do-to-change-this prompts can be replaced by learning to use help files. For further discussion of these and related issues, see Burr et al. (in press, section 3).

Some Pedagogical Ramifications

We live in a world where software producers are continually making more things easier for more people. The whole R-package culture is about this, with parallels in Python. We live in a world where anything generically useful that can be automated will be automated, where today’s complex programming job is tomorrow’s mouse click (thus death-dating overly specific skills), and where ‘good-enough, fast’ very often trumps ‘better but slower.’ The biggest chances of standing the test-of-time lie with big-picture conceptions, peripheral awareness (what is out there), and skills that facilitate doing new things. This last is why Berkeley is heavily emphasizing programming and why it is a vital part of educating professionals. But for our audiences above, not so much.

Most of our discussion has been motivated by a desire to empower people who need data science assistance to help solve important problems, but who do not have the resources to pay for it. The needs of many service course students might be expected to be somewhat similar. Even data science specialists could benefit from wide peripheral awareness, and broad practical capabilities very efficiently gained, in areas outside their specialties. We think it is an oxymoron that the world needs many more people with the skills to build wonderful new computational tools wherever they are required. Berkeley and data science programs around the world are building student flows to increase that supply. Personally, we have the “‘builder’ spirit” the Berkeley authors speak of. We continually feel the pleasures of ‘creating working artifacts’ empowered by writing computer programs. But given the power of the data tools already available, with ever more powerful tools always coming on stream, and given the lower barriers tool-use provides to entry, we believe the data world needs many more skilled tool users than it does tool builders.

In the ideal to which we have been aspiring for general empowerment tools, everything that can be automated is automated, prior-learning requirements are reduced to the barest minimum needed for getting started, and

intelligent GUIs guide investigators through thinking processes, supplemented by just-in-time information. As much cognitive and human memory-load as possible is eliminated from the process of getting information out of software so that it can be focused where it is most emphatically needed, in trying to understand what data is saying about a real-world context and the implications of that for action, and for critique.

Removing needs for up-front knowledge does not, of course, remove the need for knowledge. The knowledge crucial to interpretation and critique, for example, is still vital. We are shifting when knowledge is required, and thus its ‘learning deadline.’ We are also restricting what has to be learned to just what is needed for the current context or decision. Using our work on iNZight as an example, while iNZight was just an educational project, the assumption could always be made that there could be skilled teaching to back it up. The process of transforming iNZight into a serious empowerment tool has highlighted the need for more substantial documentation, and even courseware, with appropriate pedagogy.

STEM education typically uses bottom-up learning models whereby students start with low-level skills and conceptions. Later threads involving them are drawn together to arrive at higher level conceptions. In contrast, pedagogy synergistic with our vision of empowerment tools is necessarily top-down—working from big-picture understandings and appreciations of the relationships between the largest moving parts—leaving many details largely to software with heavy use of default-enabled automation, but with provision for drilling down into them when required or desired. Our efforts to date have convinced us that an ability to focus on the important features of data-science forests, unobscured by the overwhelming detail of the profusion of individual trees, is something that can often make an impossible task possible.

There have been some efforts in this area that are more ambitious than ours in terms of automation, if less ambitious in terms of breadth of coverage. Circa 2010, pioneering data scientist Leland Wilkinson devised a system called [AdviseStat](#) (2012) in which “users direct the program with plain verbs like ‘predict,’ ‘compare,’ or ‘cluster’ ... The program automatically transforms the data and addresses subtle diagnostic issues, before producing a whitepaper result with a customized explanation of the chosen methodology and significant findings within the data, complete with interactive graphs and full bibliography.” This essentially describes an expert system for learning from data. AdviseStat was sold to Skytree in 2012, but tragically, the project was shut down not long after. We never got to see more than a brief Joint Statistical Meetings talk. But in the last few days we have found another project with similar aims, “The Automatic Statistician” (Steinruecken et al., 2019), that is ongoing. We will watch their progress with interest, but in the meantime believe that we can deliver more assistance to the frontline faster using less automated, more highly modular and nimble approaches.

We are a very small team trying to help democratize data science and we would love more collaborators. We would also like to encourage more people in data science who are smarter, more knowledgeable, better connected and better resourced than we are to also work on general empowerment strategies and tools.

Disclosure Statement

Chris J. Wild, Tom Elliott, and Andrew Sporle have no financial or non-financial disclosures to share for this article.

References

- Advise Analytics (2012). *AdviseStat*. Scientific Computing World. <https://www.scientific-computing.com/press-releases/advise-stat>
- Burr, W., Fanny Chevalier, F., Collins, C., Gibbs, A. L., Ng, R., & Wild, C. J. (in press). Computational skills by stealth in introductory data science teaching. *Teaching Statistics*, 43(S11).
- Elliott, T., Soh, Y. H., & Barnett, D. (2021a). iNZightPlots: Graphical tools for exploring data with iNZight. (R package version 2.13.) <https://CRAN.R-project.org/package=iNZightPlots>
- Elliott, T., Wild, C., Barnett, D., & Sporle, A. (2021b). iNZight: A graphical user interface for data visualisation and analysis through R [Manuscript in preparation]. https://inzight.nz/papers/2021_jss.pdf
- Fergusson, A., & Pfannkuch, M. (in press). Introducing teachers who use GUI-driven tools for the randomization test to code-driven tools. *Mathematical Thinking and Learning*.
- Philp, R. (2020, November 24). *My favorite tools: Alberto Cairo on data visualization*. Global investigative Journalism Network. <https://gijn.org/2020/11/24/my-favorite-tools-alberto-cairo-on-data-visualization/>
- Steinruecken, C., Smith, E., Janz, D., Lloyd, J., & Ghahramani, Z. (2019). The automatic statistician. In F. Hutter, L. Kotthoff, & J. Vanschoren (Eds.), *Automated machine learning: Methods, systems, challenges* (pp. 161–173). Springer. https://doi.org/10.1007/978-3-030-05318-5_9
- Wickham H., & Grommund, G. (2017). *R for data science*. O'Reilly Media. <https://r4ds.had.co.nz/>
- Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grommund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., & Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43), Article 1686. <https://doi.org/10.21105/joss.01686>
- Wild, C. J. (2015). Further, faster, wider. Online discussion of Cobb, G. W. (2015), “Mere renovation is too little too late: We need to rethink our undergraduate curriculum from the ground up.” *The American Statistician*, 69, 266–282. https://s3-eu-west-1.amazonaws.com/pstorage-tf-iopjsd8797887/2615430/utas_a_1093029_sm0202.pdf

Wild, C. J. (2020). *About presenting data on maps*. iNZight. https://inzight.nz/user_guides/add_ons/?topic=aboutmaps

Wild, C. J., & Ridgway, J. (in press). Civic statistics and iNZight; illustrations of some design principles for educational software. In J. Ridgway (Ed.), *Statistics for empowerment and social engagement—Teaching civic statistics to develop informed citizens*. Springer.

Wild, C. J., Pfannkuch, M., Regan, M., & Parsonage, R. (2017). Accessible conceptions of statistical inference: Pulling ourselves up by the bootstraps. *International Statistical Review*, 85(1), 84–107.
<https://doi.org/10.1111/insr.12117>

©2021 Chris J. Wild, Tom Elliott, and Andrew Sporle. This article is licensed under a Creative Commons Attribution (CC BY 4.0) [International license](#), except where otherwise indicated with respect to particular material included in the article.