Simple programming tools for data exploration

Tomas Petricek

January 24, 2024

Preface

yo

Contents

Part I

Commentary

Chapter 1

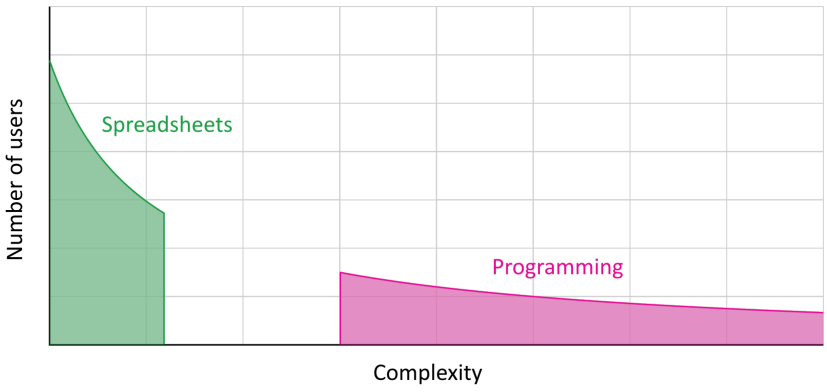
Introduction

The rise of big data, open government data initiatives (Attard et al., 2015),[1] and civic data initiatives means that there is an increasing amount of raw data available that can be used to understand the world we live in, while increasingly powerful machine learning algorithms give us a way to gain insights from such data. At the same time, the general public increasingly distrusts statistics (Davies, 2017) and the belief that we live in a post-truth era has become widely accepted over the last decade.

While there are complex socio-political reasons for this paradox, from a merely technical perspective, the general public's lack of engagement with data-driven insights should perhaps not be a surprise. We lack an accessible data exploration technologies that would allow non-programmers such as data journalists, educators and data-literate citizens to explore complex data and produce transparent data visualizations that can be understood, corrected and adapted by a broad range of end-users.

The technology gap is illustrated in Figure 1.1. On the one hand, graphical tools such as spreadsheets are easy to use, but they are limited to small tabular data sets and are error-prone (Panko, 2015) and do not aid transparency. On the other hand, programmatic tools for data exploration such as Python and R can tackle complex problems, but require expert programming skills for completing even the simplest tasks.

Figure 1.1: Illustration showing a gap between easy to use but limited spreadsheets and unlimited but hard to use programming tools. Adapted from Edwards (2015).



[1] See https://data.gov and https://data.gov.uk, but also https://opendata.gov.cz as examples.

The above illustration should not be taken at face value. Although there is no single accepted solution, there are multiple projects that exist in the gap between spreadsheets and programming tools. However, the gap provides a useful perspective for positioning the contributions presented in this thesis. They all attempt to fill or narrow the gap in some way. In particular, some of the work develops novel tools that aim to combine the simplicity of spreadsheets with the power of programming for the specific domain of data exploration. Some of the work focuses on making regular programming with data easier, or making simple programming with data accessible to a greater number of users.

1.1 Requirements of simple tools for data exploration

Although the tools and techniques presented in this thesis are more broadly applicable, the focus of this thesis is on a particular kind of data exploration tools. I focus on tools for data analysts who are technically skilled and information-literate, but are not programmers. This includes data journalists, public servants and analysts at non-government organizations or in the commercial sector. My work also focuses on the production of data analyses that are of interest to a broader readership. The produced reports should be accessible not just to data analysts, but also to a broader non-technical public. In the subsequent discussion, I distinguish between a *data analyst*, who is a technically skilled non-programmer and a *reader*, who is interested in the results, but not necessarily technical. This context leads to a number of requirements for the envisioned data exploration tools:

- *Gradual progression from simple to complex*. The system must allow non-experts with limited resources to easily complete simple tasks in an interface that later allows them to learn more and tackle more complex problems. In the technical dimensions of programming systems framework (Jakubovic et al., 2023), this is known as the staged levels of complexity approach to the learnability dimension.
- *Support transparency and openness*. The readers of the resulting data analyses must be able to understand how the analysis was done and question what processing steps and parameters have been used in order to critically engage with the problem.
- *Enable reproduction and learning by percolation*. A reader should be able to redo steps through which a data exploration was conducted in order to reproduce the results, but also to learn how to use the system. This is possible in the spreadsheet formula language (Sarkar and Gordon, 2018), but not in the graphical interface.
- *Encourage meaningful reader interaction*. The reader should not be just a passive consumer of the data analyses. They should be able to review and understand the analysis, but also make simple modifications such as changing analysis or visualization parameters, as is often done in interactive visualizations produced by journalists (Kennedy et al., 2021).

The above criteria point at the gap between spreadsheets and regular programming systems illustrated by Figure 1.1 and there are multiple possible solutions and approaches to satisfy the above criteria. This thesis explores one particular point in the design space, which is to treat data exploration as a programming problem. I aim to show that a simple programming tools for data exploration can satisfy the above criteria and make the production of open and transparent data analyses accessible even to non-programmers.

1.2 Data exploration as a programming problem

Data exploration is typically done using a combination of tools including spreadsheets, programming tools, online systems and ad-hoc utilities. Spreadsheets like Excel and business intelligence tools like Tableau (Wesley et al., 2011) are often used for manual data editing, reshaping and visualization. More complex and automated data analyses are done in programming languages like R and Python using a range of data processing libraries such as pandas and Tidyverse (Wickham et al., 2019). They are typically done in a computational notebook environments such as RStudio or Jupyter (Kluyver et al., 2016), which make it possible to interleave documentation, mathematical formulas and code with outputs such as visualizations. Online data processing environments like Trifacta provide myriads of tools for importing and transforming data, which are accessible through different user interfaces or scriptable programmatic interfaces, but even those have to be complemented with ad-hoc single-purpose tools, often invoked through a command line interface.

Finding a unified perspective for thinking about such hotchpotch of systems and tools is a challenge. The view taken in this thesis is that, most of the systems and tools can be considered as programming systems. This view makes it possible to apply the powerful methodology of programming languages research to the problem of data exploration. However, the programs that are constructed during data exploration have a number of specific characteristics that distinguishes them from programs typically considered in programming langauge research:

- *Data often exists alongside code*. Systems such as spreadsheets often mix data and code in a single environment. In conventional programming, this is done in image-based systems like Smalltalk, but not in the context of programming languages.
- *Concrete inputs are often known*. Moreover, data exploration is often done on a known collection of concrete input datasets. This means that program analysis can take this data into account rather than assuming arbitrary unknown inputs.
- *Programmers introduce fewer abstractions*. Even in programmatic data exploration using, for example, Python in a Jupyter notebook, data analysts often write code as a sequence of direct operations, rather than introducing abstractions such as reusable generic functions.
- *Most libraries are externally defined*. Finally, data exploration is often done using libraries and tools that are implemented outside of the tool that the analysts use. For example, spreadsheet formulas use mostly built-in functions, while data analyses in Python often use libraries implemented in C.

The above generally hold for simple data explorations that are done by data journalists, public servants, analysts and data-literate citizens that this thesis is primarily concerned with. The characteristics do not apply to all programs that work with data, which may include complex reusable and parameterized models, general-purpose algorithms and rich data processing pipelines. However, focusing on simple data exploration problems for which the above criteria are true allows us to narrow the design space and study a range of interesting problems that also make us rethink a number of accepted assumptions in programming language research.

1.3 Methodologies

HCI + PL + SYSTEMS

1.4 What is simple

rhetorical framework

1.5 Structure of the problem / thesis

DIAGRAM like https://public.dhe.ibm.com/software/data/sw-library/analytics/data-science-lifecycle/ but with acquisition, exploration, cleaning, visualization on the left (and model stuff on the right)

1.6 Contributions

Type providers F# data type providers + the gamma ecoop

Data exploration tools * wrattler + the gamma programming

Iterative prompting * the gamma vlhcc + AI assistants

Data visualizatin libraries compost + fluid

1.7 Projects

* F# Data * The Gamma * Wrattler * Compost and fluid

Chapter 2

Type providers

Chapter 3

Data wrangling

Chapter 4

Iterative prompting

Chapter 5

Data visualization

what? what?

Part II

Publications: Type providers

Chapter 6

Types from data: Making structured data first-class citizens in F#

Chapter 7

Data exploration through dot-driven development

Part III

Publications: Data wrangling

Chapter 8

Foundations of a live data exploration environment

Chapter 9

Wrattler: Reproducible, live and polyglot notebooks

Part IV

Publications: Iterative prompting

Chapter 10

The Gamma: Programmatic data exploration for non-programmers

Chapter 11

AI Assistants: A framework for semi-automated data wrangling

Part V

Publications: Data visualization

Chapter 12

Composable data visualizations

Chapter 13

Linked visualisations via Galois dependencies

Part VI

Conclusion

Chapter 14

Conclusion

if the society is to benefit...

AI and LLMs

in other words, technology has democratized opinions, but can it also democratize facts

Bibliography

Judie Attard, Fabrizio Orlandi, Simon Scerri, and Sören Auer. 2015. A systematic review of open government data initiatives. *Government Information Quarterly* 32, 4 (2015), 399–418. https://doi.org/10.1016/j.giq.2015.07.006

William Davies. 2017. How statistics lost their power–and why we should fear what comes next. *The Guardian* (19 January 2017). https://www.theguardian.com/politics/2017/jan/19/crisis-of-statistics-big-data-democracy

Jonathan Edwards. 2015. *Transcript: End-User Programming Of Social Apps*. https://www.youtube.com/watch?v=XBpwysZtkkQ YOW! 2015.

Joel Jakubovic, Jonathan Edwards, and Tomas Petricek. 2023. Technical Dimensions of Programming Systems. *The Art, Science, and Engineering of Programming* 7, 3 (2023), 1–13.

Helen Kennedy, Martin Engebretsen, Rosemary L Hill, Andy Kirk, and Wibke Weber. 2021. Data visualisations: Newsroom trends and everyday engagements. *The Data Journalism Handbook: Towards a Critical Data Practice* (2021), 162–173.

Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian E Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica B Hamrick, Jason Grout, Sylvain Corlay, et al. 2016. Jupyter Notebooks-a publishing format for reproducible computational workflows. In *20th International Conference on Electronic Publishing*, Fernando Loizides and Birgit Schmidt (Eds.). 87–90. https://doi.org/10.3233/978-1-61499-649-1-87

Raymond R. Panko. 2015. What We Don't Know About Spreadsheet Errors Today. In *Proceedings of the EuSpRIG 2015 Conference "Spreadsheet Risk Management"*. European Spreadsheet Risks Interest Group, 1–15.

Advait Sarkar and Andrew D Gordon. 2018. How do people learn to use spreadsheets?. In *Proceedings of the Psychology of Programming Interest Group (PPIG)*, Mariana Marasoiu Emma S oderberg, Luke Church (Ed.).

Richard Wesley, Matthew Eldridge, and Pawel T. Terlecki. 2011. An analytic data engine for visualization in tableau. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data* (Athens, Greece) *(SIGMOD '11)*. Association for Computing Machinery, New York, NY, USA, 1185–1194. https://doi.org/10.1145/1989323.1989449

Hadley Wickham, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, Alex Hayes, Lionel Henry, Jim Hester, et al. 2019. Welcome to the Tidyverse. *Journal of open source software* 4, 43 (2019), 1686.