The Gamma: Data Exploration through Iterative Prompting

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

ABSTRACT

Governments, non-profit organizations and citizen initiatives publish increasing amounts of data, but extracting insights from such data and presenting them to the public is hard. First, data comes in a variety of formats that each requires a different tool. Second, many data exploration tools do not reveal how a result was obtained, making it difficult to reproduce the results and check how they were obtained. We contribute The Gamma, a novel data exploration environment for non-experts. The Gamma is based on a single interaction principle and using it results in transparent and reproducible scripts. This allows transfer of knowledge from one data source to another and learning from previously created data analyses. We evaluate the usability and learnability of The Gamma through a user study on non-technical employees of a research institute. We argue that the our approach allows journalists and the public to benefit from the rise of open data, by making data exploration easier, more transparent and more reproducible.

Author Keywords

Data exploration; End-user programming; Data journalism; Programming languages; Type providers

INTRODUCTION

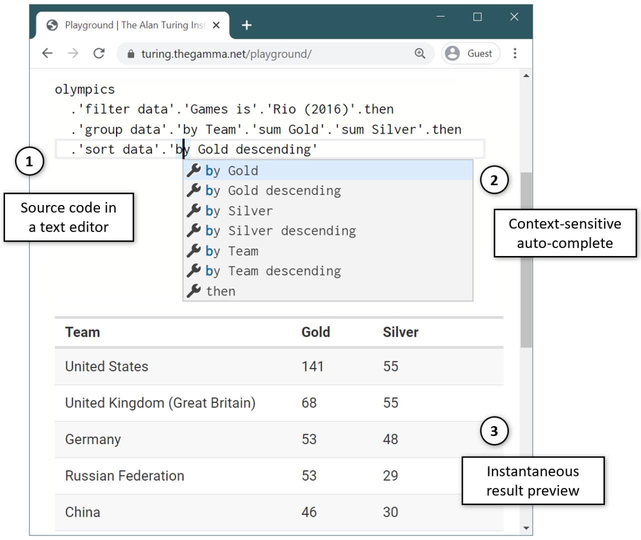
Data science has more capabilities to help us understand the world than ever before, yet at the same time post-truth politics and increasing public distrust in statistics makes data-driven insights increasingly less relevant in public discourse [5]. This should perhaps not be a surprise. Journalists can access increasing amounts of data, but producing engaging and transparent data-driven reports that are easy to interpret is expensive and requires expert programming skills [7].

The design of a data exploration tool for journalists poses a unique mix of challenges. First, the tool needs to be easy to learn for end-users working under tight deadlines. Second, it needs to support a wide range of data sources in a way where the expertise gained when working with one data source is relevant for other data sources. Third, the resulting data-driven insights need to be transparent, allowing the readers to verify the claims and learn how to reproduce the work.

UIST'20, October 20–23, 2020, Minneapolis, MN, USA

ISBN 978-1-4503-6708-0/20/04…$15.00

DOI: https://doi.org/10.1145/3313831.XXXXXXX


Playground | The Alan Turing Inst
turing.thegamma.net/playground/
Guest
1
Source code in a text editor
2
Context-sensitive auto-complete
3
Instantaneous result preview

```
olympics
  .'filter data'.'Games is'.'Rio (2016)'.then
  .'group data'.'by Team'.'sum Gold'.'sum Silver'.then
  .'sort data'.'by Gold descending'
```

```
by Gold
by Gold descending
by Silver
by Silver descending
by Team
by Team descending
then
```

Team	Gold	Silver
United States	141	55
United Kingdom (Great Britain)	68	55
Germany	53	48
Russian Federation	53	29
China	46	30

Figure 1. Teams with the greatest number of gold medals from Rio 2016 Olympics with a reproducible The Gamma script (1), an auto-complete prompt offering ways of sorting the data (2) and instant preview (3).

We present The Gamma, a text-based data exploration environment for non-experts. The Gamma is based on a single interaction principle, which provides uniform access to a range of data sources including data tables, graph databases and data cubes. An anlysis created in The Gamma is a transparent script that can be followed to reproduce the result from scratch. This allows learning from existing analyses and encourages readers to engage with the results. Our key contributions are:

- We identify the design requirements for a data exploration tool for journalists (Section 3) and follow those to build a novel programming environment The Gamma (Section 4).
- We introduce *iterative prompting* (Section 5), an interaction design principle that can be used to complete a variety of programming tasks in a uniform way that allows transfer of knowledge between different tasks.
- We show how to use the iterative prompting principle for querying of distinct data sources including data tables, graph databases and data cubes (Section 6).
- We discuss a number of case studies (Section 7) and conduct a user study to evaluate the usability of The Gamma and the extent to which users can, (i) learn from examples and (ii) transfer knowledge between tasks (Section 8).

The Gamma is available as open-source at `thegamma.net`.

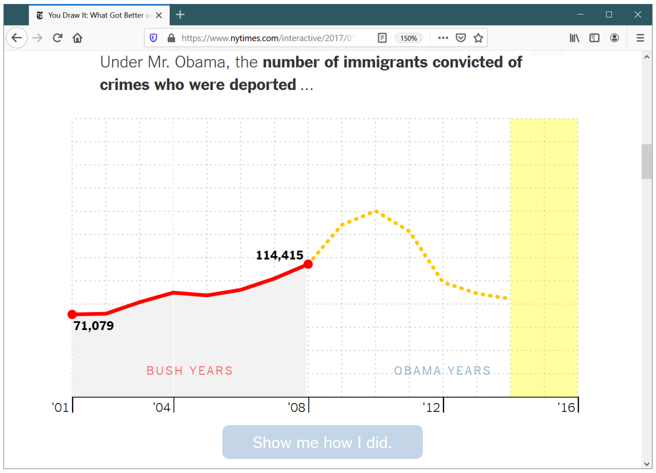
RELATED WORK

Visual tools.

Programming tools. Notebooks

Journalism. Idyll [4]

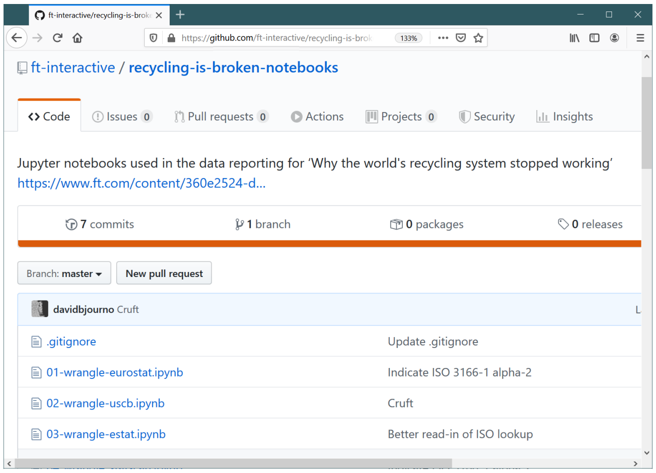
Type providers. PL work



Under Mr. Obama, the **number of immigrants convicted of crimes who were deported** ...

You Draw It: What Got Better o
https://www.nytimes.com/interactive/2017/01
150%
114,415
71,079
BUSH YEARS
OBAMA YEARS
'01
'04
'08
'12
'16
Show me how I did.

Figure 2. New York Times article on Obama's legacy [8]. The article asks the reader to make a guess (engagement), but only lists "Immigration and Customs Enforcement" as a source of data.



ft-interactive / **recycling-is-broken-notebooks**

Jupyter notebooks used in the data reporting for 'Why the world's recycling system stopped working'
https://www.ft.com/content/360e2524-d...

7 commits · **1** branch · **0** packages · **0** releases

Branch: **master** · **New pull request**

davidbjourno Cruft

.gitignore	Update .gitignore
01-wrangle-eurostat.ipynb	Indicate ISO 3166-1 alpha-2
02-wrangle-uscb.ipynb	Cruft
03-wrangle-estat.ipynb	Better read-in of ISO lookup

ft-interactive/recycling-is-broke
https://github.com/ft-interactive/recycling-is-brok
133%
Code
Issues 0
Pull requests 0
Actions
Projects 0
Security
Insights

Figure 3. Financial Times analysis of recyclable waste. Full source is provided as Jupyter Notebooks on GitHub [2], but re-running the analysis is difficult, even for an expert.

BACKGROUND AND DESIGN GOALS

The Gamma aims to adapt the recent innovations in programming language research, especially the work on type providers, into a form where it could be used in practice by journalists and other non-expert interested in data exploration. We start with a careful consideration of our target application domain, i.e. data analyses produced by journalists and citizen data scientists that are published online. We look at both practical requirements for such programming environment and requirements arising from our focus on journalism. This analysis is based on the author's experience of collaborating with journalists[1], review of literature on data journalism, e.g. [7, 11, 1] and more general trends in journalism.

Open Journalism

Journalism continually develops and responds to the many challenges it faces [6]. Two recent challenges are relevant to our work. The first is building trust in media. One way of establishing trust in the age of fake news is to be more transparent about editorial decisions, process and original sources. Many journalists believe that opening up the process shows the quality and trustworthiness of their work [10]. The second challenge is reader engagement. To develop a relationship with readers, journalists are increasingly looking for meaningful ways of engagement. This includes reader comments, involvement of citizen journalists [9, 3] and the development of new interactive formats [8]. To address the above challenges, a tool for data exploration should satisfy the following three requirements.

Trust Through Transparency

To support trustworthiness, data analyses should be transparent. The reader should be able to determine what is the source of analysed data and how has the data been transformed. As much as possible, these capabilities should also be accessible to non-expert readers.

Reproducibility for Fact Checking

It should be possible to re-run the analysis to verify that it produces the presented results. However, running an opaque script is not enough. A reader should be able to recreate the analysis by following the necessary steps from the original data source to the end result.

Encouraging Meaningful Engagement

The tool should support a mechanism through which readers can engage in a meaningful discussion. For example, it should allow modifying of parameters of a data visualization in order to show how different choices affect the final result.

End-user Data Exploration

Our aim is to make programmatic data exploration accessible to journalists, but we want to keep the desirable properties of text-based programming. In particular, source code of a data exploration should provide a full reproducible record of how the data analysis has been done. As end-users, journalists have a number of interesting characteristics. They work under tight deadline and data exploration is only a complementary skill. They also need to work with a wide range of data sources, including big data tables (e.g. Iraq War documents leak) or graph databases (e.g. Panama Papers). This leads to a number of practical requirements on the programming environment.

Conceptual Simplicity

We target end-users who cannot dedicate much time to learning about a tool prior to using it. Consequently, using the tool should require understanding of only a small number of concepts. Once the user understand a small number of concepts, they should be able to complete basic data exploration tasks.

Uniformity across Data Sources

The users should be able to navigate through large databases, query relational databases and query graph databases through the same mechanism. Ideally, expertise gained with one data source should also be transferable to working with another data source.

[1] Citations removed to preserve anonymity.

Learning without Experts

Sarkar [12] reports that users learn how to use Excel either by talking to experts, or by seeing a feature in a spreadsheet received from a colleague. In our circumstances, experts are unlikely to be available, so the tool should support learning from examples. When looking at a work done and published by another person, the user needs to see (and be able to understand) how a task was completed.

OVERVIEW

The Gamma is a text-based programming environment that allows non-experts create simple data exploration scripts using a single interaction principle – choosing an item from an auto-complete list. It supports a range of data sources including tabular data, graph data and data cubes.

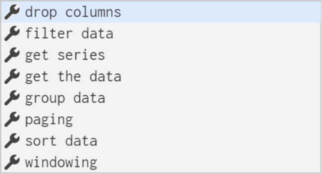
Querying Travel Expenses

To introduce The Gamma, we walk through a simple problem that a local journalist might want to solve. The UK government publishes travel expense claims by members of the House of Lords. A journalist wants to find out which of the members representing the Kent county spend the most on travel. The following shows a subset of the data:[2]

```
Name, County, Days Attended, Days Away, Travel Costs
Lord Adonis, London, 8, 0, 504
Baroness Afshar, Yorkshire, 2, 0, 0
Lord Alderdice, Oxfordshire, 3, 0, 114
Lord Alli, London, 5, 0, 0
Baroness Amos, London, 3, 0, 0
```

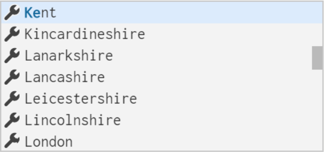
After the analyst imports the CSV file (through a web interface), the environment is initialized with code that refers to the imported variable expenses. The analyst then types '.' (dot):

```
expenses.
```



The type provider for tabular data allows analysts to construct simple queries. It first offers a list of operations that the analyst might want to perform such as grouping, filtering and sorting. To find members of the House of Lords from Kent, the analyst chooses filter data, types '.' and then chooses County is from the offered list, types '.' and starts typing Kent:

```
expenses
  .'filter data'.'County is'.Ke
```

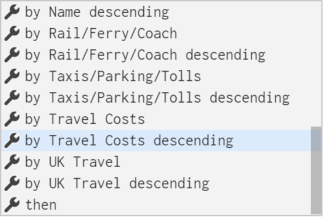


The completion list is generated from the values in the County column of the dataset. After selecting Kent, the live preview is updated to only show records according to the specified filter:

Name	County	Days Attended	Days Away	Travel Costs
Lord Astor of Hever	Kent	7	0	85
Lord Freud	Kent	1	0	0
Lord Harris of Peckham	Kent	3	0	0
Baroness Noakes	Kent	6	0	135

To finish specifying filtering conditions, the analyst chooses then and is offered the same list of querying operations as in the first step. To sort House of Lords members by their travel costs, she now chooses sort data and types '.' (dot):

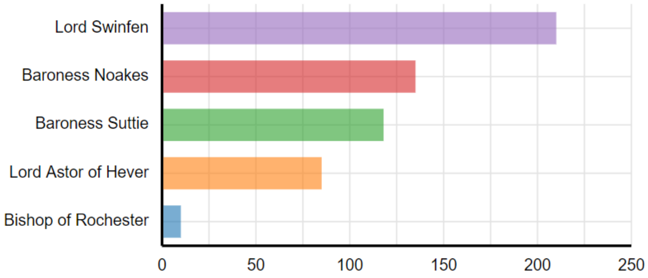
```
expenses
  .'filter data'.'County is'.Kent.then
  .'sort data'.
```



The auto-complete offers a list of columns that can be used for sorting, each with ascending (default) and descending order option. After choosing one or more sort keys, the analyst selects the then member and is, again, offered the list of querying operations. They use paging to get top 5 records and get series to obtain a data series with just the House of Lords member name and their travel expenses.

```
expenses
  .'filter data'.'County is'.Kent.then
  .'sort data'.'by Travel Costs descending'.then
  .paging.take(5)
  .'get series'.'with key Name'.'and value Travel Costs'
```

When the code evaluates to a data series with a categorical (textual) key and a numerical value, The Gamma switches from displaying the result as a table to a column chart:



Querying via Iterative Prompting

The

[2] https://www.parliament.uk/mps-lords-and-offices/members-allowances/house-of-lords/holallowances/

DESIGN

interaction = iterative prompting how is this different from just auto-complete?

trace = for transparency and learning

One important observation from the above list is that our tool should be accessible to non-expert users such as readers and non-technical journalists, while providing extra capabilities for more technical users. It should be easy to use if one just wants to modify existing code and should encourage experimentation. Considering these challenges, we identify the following technical design principles. todoThere must be papers on learning programming that can be referenced here.

B1. Learning from examples and by experimentation.
We should support two ways of learning. Users of tools such as spreadsheets often learn by looking at existing problem solutions todoAdvait's PPIG. Our design should allow this by making it possible to inspect and retrace steps used while solving a problem in an existing application. Another principle of spreadsheets that we want to keep is the ability to experiment and see results immediately. Our design should allow users to try invoking an operation or modifying a parameter and quickly see if this leads to the desired results.

B2. Choice over construction.
To minimize the amount of information that users have to learn and remember, our system should work in a way that allows constructing programs by choosing from options that can reasonably appear in a current context, rather than requiring users to recall particular syntax or exact identifier name. todorecognition over recall?

B3. Make simple things easy and complex things possible.
Some users of the system may, over time, become advanced users and the system should support those. In other words, the upper bound on what can be achieved should be well above the most common use cases. At the same time, the complex features that power users might need should not affect the most elementary uses of the system and should remain completely hidden until needed. In other words, the lower bound on what one needs to know to use the system for basic tasks should be as low as possible. todoI think I got this idea of "boundaries" on what is possible from some paper, but cannot recall which...

B4. Visibility of state.
To support transparency, the system should make its entire state transparent – when reviewing a data analysis, all parameters should be immediately visible and the user should not need to, e.g. navigate through complex user interface to find them.

IMPLEMENTATION

Language

Type providers

USE CASES

USER STUDY

this is between-user user study

FIRST: How one area transfers to another?

DEMO: WorldBank explain everything

DISPLAY: Public order and safety, Defence

```
expenditure.byService.'Public order and safety'.inTermsO
expenditure.byService.Defence.inTermsOf.GDP
```

DEMO: Graph DB explain everything

```
drWho.Character.Doctor.'ENEMY_OF'.'[any]'
  .'APPEARED_IN'.'[any]'.'explore_properties'.explore
  .'group data'.'by 1-name'.'count distinct 2-title'
```

DISPLAY: Who has larges travel expenses? (and in London only?)

```
lords.'sort data'.'by Travel Costs descending'

lords
  .'filter data'.'County is'.London.then
  .'sort data'.'by Travel Costs descending'
```

SECOND: No experts are needed

DEMO: explain how live preview works, explain how '.' works, explain how newlines and indentation work

DISPLAY 'CO2 emissions (metric tons per capita)'

```
worldbank.byCountry.'United Kingdom'
  .'Economy & Growth'.'GDP per capita (current US$)'

worldbank.byCountry.Germany
  .'Economy & Growth'.'GDP per capita (current US$)'

worldbank.byCountry.'Czech Republic'
  .'Economy & Growth'.'GDP per capita (current US$)'
```

DISPLAY: Top athletes from London

```
olympics
  .'group data'.'by Team'.'sum Gold'.then
  .'sort data'.'by Gold descending'.then
  .paging.take(5)
  .'get series'.'with key Team'.'and value Gold'
```

THIRD: 'then'

DEMO: Show worldbank GIVE: Commented source code using 'olympics' DISPLAY: Top athletes from London (think-aloud)

QUESTIONS

1) Did I tell you enough in the introduction to get started?

Say we want to provide educational materials for journalists (with limited budgets), what would be the most important?

2) Video or just code samples?

We'll have more data sources than we can write tutorials for

3) Do we just teach them how the environment works?

4) What do we need to teach about a data source?

DISCUSSION

Study limitations

exploratory in nature so we do not make any quantitative claims about effects

not comparing against other systems

Design principles

How well did we do wrt design principles?

Design issues

future challenges and limitations of the model - such as issues when modifying code in the middle of the call chain

CONCLUSIONS

ACKNOWLEDGMENTS

Yo

REFERENCES

[1] 2018. *Proceedings of the 2nd European Data and Computational Journalism Conference*. University College Dublin.

[2] David Blood. 2018. Recycling is broken – notebooks. (11 2018). https://github.com/ft-interactive/recycling-is-broken-notebooks

[3] Axel Bruns, Tim Highfield, and Rebecca Ann Lind. 2012. Blogs, Twitter, and breaking news: The produsage of citizen journalism. *Produsing theory in a digital world: The intersection of audiences and production in contemporary theory* 80, 2012 (2012), 15–32.

[4] Matthew Conlen and Jeffrey Heer. 2018. Idyll: A Markup Language for Authoring and Publishing Interactive Articles on the Web. In *The 31st Annual ACM Symposium on User Interface Software and Technology, UIST 2018, Berlin, Germany, October 14-17, 2018*, Patrick Baudisch, Albrecht Schmidt, and Andy Wilson (Eds.). ACM, 977–989. DOI: http://dx.doi.org/10.1145/3242587.3242600

[5] William Davies. 2017. How statistics lost their power - and why we should fear what comes next. The Guardian. (19 Jan 2017). Retrieved March 6, 2020 from https://www.theguardian.com/politics/2017/jan/19/crisis-of-statistics-big-data-democracy.

[6] Bob Franklin. 2012. THE FUTURE OF JOURNALISM. *Journalism Studies* 13, 5-6 (2012), 663–681. DOI: http://dx.doi.org/10.1080/1461670X.2012.712301

[7] Jonathan Gray, Lucy Chambers, and Liliana Bounegru. 2012. *The data journalism handbook: how journalists can use data to improve the news*. O'Reilly Media, Inc.

[8] Haeyoun Park Larry Buchanan and Adam Pearce. 2017. You Draw It: What Got Better or Worse During Obama's Presidency. New York Times. (15 Jan 2017). Retrieved March 3, 2020 from https://www.nytimes.com/interactive/2017/01/15/us/politics/you-draw-obama-legacy.html.

[9] Edith Manosevitch and Dana Walker. 2009. Reader comments to online opinion journalism: A space of public deliberation. In *International Symposium on Online Journalism*, Vol. 10. 1–30.

[10] Klaus Meier. 2009. Transparency in Journalism. Credibility and trustworthiness in the digital future. In *Actas II Congreso The Future of Journalism*.

[11] Tomas Petricek, Bahareh R. Heravi, Jennifer A. Stark, and et al. 2017. *Proceedings of the European Data and Computational Journalism Conference*. University College Dublin.

[12] Advait Sarkar and Andrew Donald Gordon. 2018. How do people learn to use spreadsheets? (Work in progress). In *Proceedings of the 29th Annual Conference of the Psychology of Programming Interest Group (PPIG 2018)*. 28–35.