Fluid: towards transparent, self-explanatory research outputs

Data, curated by journalists and scientists into charts and other visual summaries, is how we understand and make decisions about our rapidly changing world. But even when these visual summaries are accompanied by the relevant data sets and source code, *understanding* how a potentially huge data source and complex visual output are related — what the visualisation actually means — is a challenge, requiring expertise and most likely significant time spent with the code. Even knowledgeable readers get it wrong. Important features of the data can be obscured or exaggerated by the choice of plot [11] or visual parameter [4]; innocent (but devastating) mistakes such as transposing two columns of data have gone unnoticed for several years in widely cited papers [5]. Politicians routinely misrepresent data to suit their purposes [6]. Making sense of (and trusting) a visualisation, even for an expert, means understanding what its visual attributes *represent*, which in turn means understanding the mapping between data source and visual elements, how the data was filtered, transformed and aggregated, and how different views of the same data might be related.

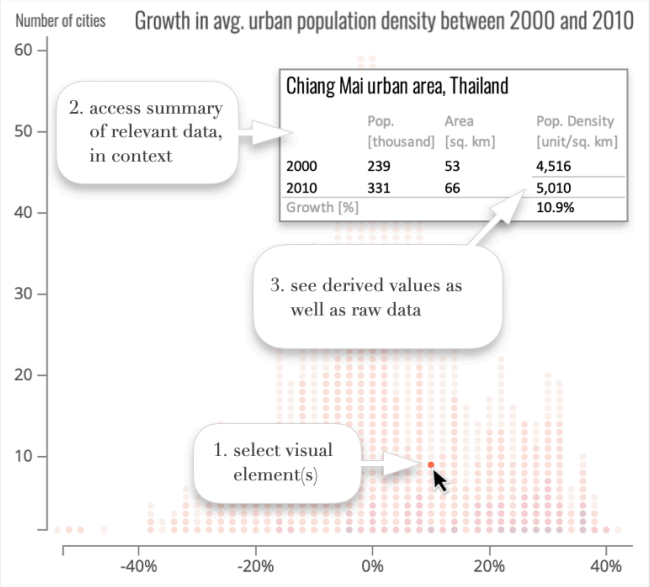

Fig. 1: Understanding how charts relate to data

If visualisations and other computed artefacts were able to reveal these sorts of relationships themselves, readers would be much more easily able to verify assumptions or fact-check claims *in situ*. Nadieh Bremer [3] showed the value of this in her award-winning visualisation of population density growth (Fig. 1). Selecting the circle representing Chiang Mai (call out 1) brings up a view of the data (call outs 2 and 3) used to calculate that the population density of the city grew by 10.9% between 2000 and 2010, the value which is encoded by the circle's position on the x axis. Interactions like these transform our ability to understand a visualisation, but implementing them by hand, as Bremer did, requires a huge investment of time. And of course the manual process of enriching the visualisation with explicit metadata about the visual analysis introduces a significant maintenance obligation plus an additional source of potential error.

1 Fluid: a programming language with integrated data provenance

This talk will make the case that new PL infrastructure, based on techniques from data provenance, can be used to equip outputs with the kind of transparency features showcased in Bremer's work, but in an automatic or semi-automatic way. We introduce a programming language called Fluid which incorporates a bidirectional dynamic dependency analysis into its runtime; Fluid tracks dependencies as outputs (such as charts) are computed from data, and automatically enriches the rendered output with interactions allowing a reader to make sense of a chart interactively by exploring its relationship to data. Fluid builds on program slicing techniques based on *Galois connections* [7, 8, 9], which establish that the two directions of the dependency analysis are adjoint: in particular, the subset of the data identified by the backwards analysis for a given output selection contains enough information, if fed into the forward analysis, to reproduce the selected output. Using our current implementation, we will show how charts created in Fluid are able to reveal the data related to specific parts of the chart, and will argue that this kind of feature has the potential to make data-driven claims easier to critique and understand for experts and non-experts alike.

We will also show that, because every data selection has a complement, our dependency analysis has a De Morgan dual which inverts the direction of the Galois connection by transposing the roles of the two adjoints. We use this to implement a popular a visual comprehension aid called *brushing and linking* [2], where selections in one chart automatically select corresponding elements in another. While geospatial applications like GeoDa [1] and libraries like Plotly provide partial support for this already, our implementation is able to do so in a fully automatic and mathematically robust way. We close the first part of the talk by discussing some of the limitations of the present system.

2 Towards self-explanatory data visualisation

The second part of the talk will sketch how we plan to move this research forward to support data visualisations which are not only transparently linked to data but can reveal the computational nature of those relationships — the "how" as well as the "what". We will motivate this in the context of the brushing and linking feature introduced earlier. In Bremer's visualisation (Fig. 2), selecting Thailand on the left (call out 1) automatically selects all cities in Thailand on the right (call out 2). Existing implementations of brushing and linking, including ours, are *opaque* in that they say nothing about why or how selections are related. (Here, even an informed reader might mistakenly equate Average Urban Population Density of a country with the average of the individual population densities of its constituent cities.) Call out 3 shows how one might extend Fluid to provide a more transparent, robust and pervasive [10] form of brushing and linking, where linked visual elements are able to provide a concise view of the data and computations that explain why they are linked. While there are many challenges involved in turning this into a reality, we conclude that new PL infrastructure offer the best prospects for doing so in a robust, largely automatic way.

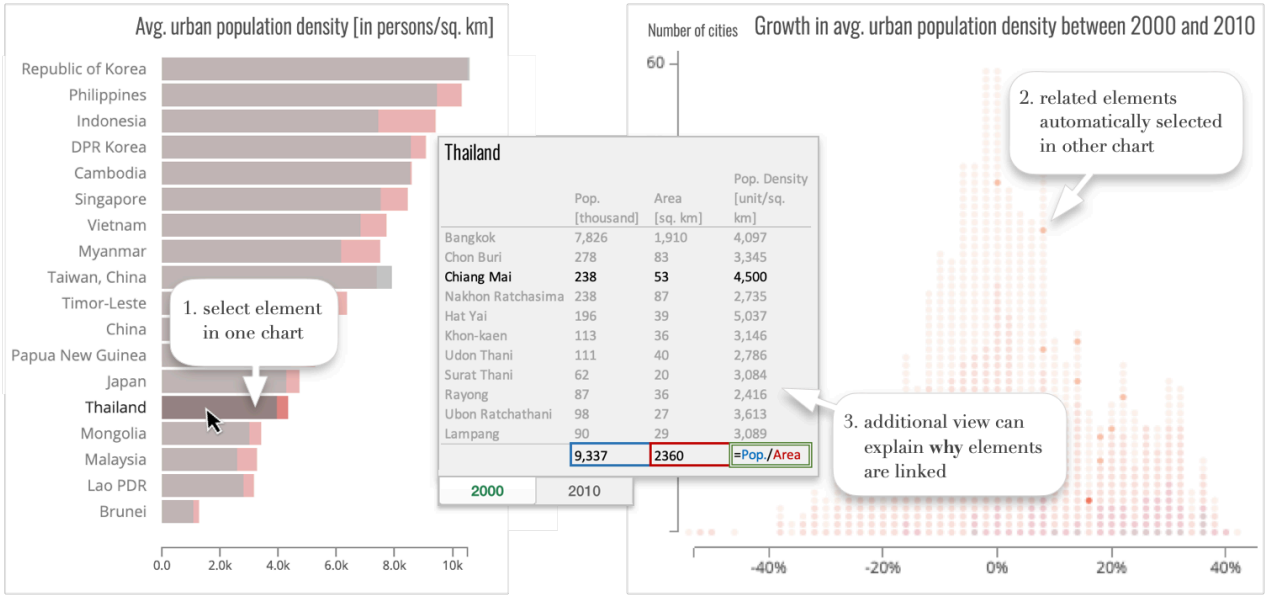

Fig. 2: Understanding how two visualisations are related through transparent brushing and linking

References

[1] L. Anselin, I. Syabri, Y. Kho. GeoDa: An Introduction to Spatial Data Analysis. *Geographical Analysis*, 38(1):5–22, 2006. DOI: 10.1111/j.0016-7363.2005.00671.x.

[2] R. A. Becker, W. S. Cleveland. Brushing Scatterplots. *Technometrics*, 29(2):127–142, May 1987. DOI: 10.1080/00401706.1987.10488204.

[3] N. Bremer, M. Ranzijn. Urbanization in East Asia between 2000 and 2010. `http://nbremer.github.io/urbanization/`, 2015.

[4] M. Correll, M. Li, G. Kindlmann, C. Scheidegger. Looks Good To Me: Visualizations As Sanity Checks. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):830–839, January 2019. DOI: 10.1109/TVCG.2018.2864907.

[5] G. Miller. A Scientist's Nightmare: Software Problem Leads to Five Retractions. *Science*, 314(5807):1856–1857, 2006. ISSN 0036-8075. DOI: 10.1126/science.314.5807.1856.

[6] A. Panjwani. You Need to Read the Small Print on this Lib Dem Voting Intention Graph. `https://fullfact.org/news/lib-dems-somerset-poll/`, 2019.

[7] R. Perera, U. A. Acar, J. Cheney, P. B. Levy. Functional Programs That Explain Their Work. *Proceedings of the 17th ACM SIGPLAN International Conference on Functional Programming*, ICFP '12, pp. 365–376, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1054-3. DOI: 10.1145/2364527.2364579.

[8] R. Perera, D. Garg, J. Cheney. Causally Consistent Dynamic Slicing. J. Desharnais, R. Jagadeesan, editors, *Concurrency Theory, 27th International Conference, CONCUR '16*, Leibniz International Proceedings in Informatics (LIPIcs), Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. DOI: 10.4230/LIPIcs.CONCUR.2016.18.

[9] W. Ricciotti, J. Stolarek, R. Perera, J. Cheney. Imperative Functional Programs That Explain Their Work. *Proceedings of the ACM on Programming Languages*, 1(ICFP):14:1–14:28, 2017. ISSN 2475-1421. DOI: 10.1145/3110258.

[10] J. C. Roberts, M. A. E. Wright. Towards Ubiquitous Brushing for Information Visualization. *Tenth International Conference on Information Visualisation (IV'06)*, pp. 151–156, July 2006. DOI: 10.1109/IV.2006.113.

[11] T. L. Weissgerber, N. M. Milic, S. J. Winham, V. D. Garovic. Beyond Bar and Line Graphs: Time for a New Data Presentation Paradigm. *PLOS Biology*, 2015. DOI: 10.1371/journal.pbio.1002128.