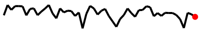
Sparqlines: SPARQL to Sparkline

Sarven Capadisli[1,✉]

[1]Enterprise Information Systems Department, University of Bonn, Bonn, Germany
✉`info@csarven.ca`

Document ID: http://csarven.ca/sparqlines-sparql-to-sparkline

Abstract. This article presents *sparqlines*: statistical observations fetched from SPARQL endpoints and displayed as inline-charts. An inline-chart, also known as a *sparkline*, is concise, and located where it is discussed in the text, complementing the supporting text without breaking the reader's flow. For example, the GDP per capita growth (annual %) [Canada]  claimed by the World Bank Linked Dataspace. We demonstrate an implementation which allows scientists or authors to easily enhance their work with sparklines generated from their own or public statistical linked datasets. This article includes an active demonstration accessible at http://csarven.ca/sparqlines-sparql-to-sparkline.

Keywords: Linked Data • Semantic publishing • Sparkline • SPARQL • Statistics • User interface

1 Introduction

In this article we introduce *sparqlines*, an integration of statistical data retrieval using SPARQL with displaying observations in the form of word-size graphics: sparklines. We describe an implementation which is part of a Web based authoring tool (dokieli). We cover how the data is modelled and exposed in order to be suitable for embedding; demonstrate how to embed data as both static and dynamic sparklines and discuss the technical requirements of each; and walk through the user interactions to do so.

Our contribution is the generation of a well-established visual aid to reading statistical data (the sparkline) directly from the dataset itself, at the time of authoring the supporting text as part of the writing workflow. This enables authors who are already publishing data to use it directly, as well as encouraging them to make their data available for others to use, and offers an easy way to present the reader with a way to better understand the information.

We conclude with a discussion, including design considerations. The code of our implementation is open source, and we invite you to try it out and make requests for more advanced features: https://github.com/linkeddata/dokieli.

2 Related Work

2.1 Sparklines

The earliest known implementation of an inline-chart was designed and programmed by Peter Zelchenko and Mike Medved to represent historical charts efficiently in the QuoteTracker software in early 1999 [1]. They are "datawords", carrying dense information with the resolution of typography, particularly useful in places where the available screen real estate is minimal. Edward Tufte describes sparkline as "small intense, simple, word-sized graphic with typographic resolution" [2]. They are designed to be included anywhere, for example embedded in a sentence, table or even a map, within the relevant context. When embedded in a sentence, they support the text, allowing continuous reading without the need to refer to a figure disjoint from the original context, whilst still providing an opportunity for the reader to investigate further by clicking on the data line to access each point of source, per Figure 1.

Sparkline graphics typically have a variable long dimension and a constrained short dimension. In the case of a typographic line, the constraint can be fixed to the height of the font-size of the encapsulating component. For example, the computed CSS `height` value of the `embed` HTML element that contains the sparkline on the current viewing device is `20px`, and so the embedded sparkline in this paragraph will have a vertical aspect ratio as such.

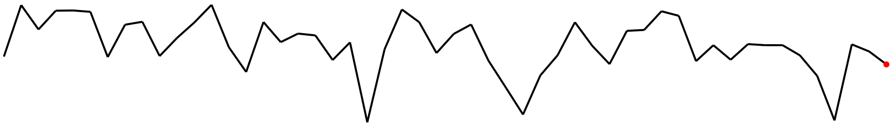
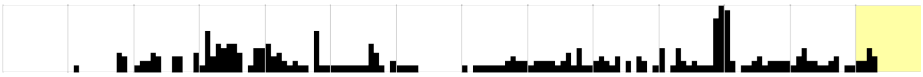


Fig. 1. A typical static figure in an article disjoint from the original context

Sparklines appear in many places where small datafeeds are useful; programmatical insertion in text-editors and spreadsheets, fitness feeds from wearable watches, social media analytics, streaming real-time quotes, electroencephalograms, system dashboards and trays, temperature and stock activity, to name a few. Studies show that novice and experienced investors using stock reports with Sparklines will experience reduced cognitive load [3].

Sparklines in line, bar, column or win/loss graphs can be programatically included in *Google Drive* documents by including data from an embedded table or sequence of numbers via the Google Charts API [4].

The *Wayback Machine* uses sparklines to show an application of the snapshots of a URL through time:



There are sparkline implementations in JavaScript libraries like d3.js and jQuery. Sparkline implementations also exist for command-line interfaces. These tools tend to take input data in tabular form (CSV). Sparklines can also be created by simple use of Unicode characters: ▂▁▄▃▆▅█▇.

2.2 RDF Data Cube and SPARQL

Statistical data that is modelled with the RDF Data Cube vocabulary [5] makes it possible to discover and identify artefacts in a uniform way. This is in contrast to writing applications to consume data from endpoints with heterogeneous data models. For front-end Web applications, data can be fetched, explored, and filtered from statistical linked dataspaces with SPARQL endpoints, e.g., http://270a.info/. Utilising this method of access from within various types of articles on the Web, makes it possible to build applications which put more focus on user-interfaces rather than handling different data models case by case, or burdensome data integration tasks. Furthermore, having easy access to highly structured multidimensional data - essentially through an `HTTP GET` request - makes it desirable to create static and real-time visualisations.

Sgvizler is a SPARQL result set visualisation JavaScript library that uses Google Charts API to create sparkline images [6]. These are block-level raster images.

Investigation of analysis and visualisation of piracy reports have been conducted through endpoint querying with a SPARQL client for R [7].

CubeViz was developed to visualise multidimensional statistical data. It is a faceted browser, which utilizes the RDF Data Cube vocabulary, with a chart visualisation component [8].

Linked Statistical Data Analysis [9], presents a way to reuse data through federated SPARQL queries, and generation of statistical analyses and scatter plots. The stats.270a.info service stores computed analysis, and makes it possible for future discovery.

3 Data Provision

In order to use sparqlines, data has to be both well-formed and available over a SPARQL endpoint. Here we briefly discuss both of these requirements.

The RDF Data Cube vocabulary is used to describe multidimensional statistical data. It makes it possible to represent significant amounts of heterogeneous statistical data as Linked Data which can be discovered and identified in a uniform way. To qualify for consumption as a sparqline, the data must conform with some of the integrity constraints of the RDF Data Cube model, e.g., IC-1 (Unique DataSet), IC-11 (All dimensions required), IC-12 (No duplicate observations), IC-14 (All measures present).

Additional enrichments on the data cubes can improve their discovery and reuse. Examples include but not limited to; providing human-readable *labels* for the datasets (with language tags), *classifications*, and *data structure definition*, as well as *provenance* level data like license, last updated.

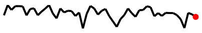
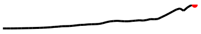
In order to allow user interfaces which can utilise a group of observations in a dataset, *slices* should be made available in the data. This enables consuming applications to dissect datasets (through SPARQL queries) for arbitrary subsets of observations. For example, while it is possible to construct a general query to get all of the observations in a dataset which have a particular dimension, it may be preferable to only query for such subsets provided that their structures can be identified and externally referenced. In the case of sparklines, one common use case for slices is to present data in time-series.

SPARQL queries are used to filter for graph patterns in the RDF Data Cube datasets. Depending on the user interface application, there may be multiple queries made to the SPARQL endpoints in order to filter the data based on user input. For example, an initial query may be a cursory inspection to discover suitable datasets with given parameters, e.g., what the dataset is about, the type of dimensions and their values, and subsequent queries may be to retrieve the matching datasets or slices with observations and their measure values.

4 Static and Dynamic Sparqlines

The data behind a sparqline can be static: a fixed historical set to which no new points are added; or dynamic: subject to change as new data is gathered. Both of these cases are accommodated by our implementation.

Table 1. Static and Dynamic Sparklines

	Use	Methods	Example
Static	Historical data or a fixed snapshot	• Pre-rendered SVG • Embedded directly from datastore	
Dynamic	Data which may be subject to updates	• Re-fetches data on page load or polls in real-time • Embed source as API endpoint which returns the sparkline	 (reload article in browser)

5 Embedding Sparqlines

Our implementation allows authors to select text they have written which describes the data they want to visualise; it searches available datasets for those relevant to the text, and lets the user choose the most appropriate if there's more than one. The sparqline is inserted along with a reference to the source.

A specific example workflow is demonstrated when this article is viewed in a Web browser (at its canonical URL: http://csarven.ca/sparqlines-sparql-to-sparkline). Enable the Edit mode from the ≡ menu and highlight the text `GDP of Canada`. What occurs is as follows:

1. User enters text in a sentence e.g., `GDP of Canada`.
2. User selects text `GDP of Canada` with their mouse or keyboard.
3. The user select the "sparkline" option from presented authoring toolbar.
4. The input text is split into two: 1) `GDP` and 2) `Canada` segments, whereby the first term is the concept, and the second is a reference area. Reference areas are disambiguated against an internal dictionary.
5. System constructs a SPARQL query URL and sends it to the World Bank Linked Dataspace endpoint, looking for a graph pattern where the datasets of labels have "GDP" in them in which there is at least one observation for the reference area "Canada".
6. User is given a list of datasets to select from which match the above criteria, and the user selects desired dataset.
7. System sends a SPARQL query to get the observations of the selected dataset for Canada.
8. A sparkline is created and displayed for the user, also indicating the number of observations it has.
9. If the user is happy with this visualisation they include it in the text. A hyperlink to the dataset, and a sparkline SVG is inserted back into the sentence replacing `GDP of Canada` with `GDP per capita growth (annual %)`.

6 Semantic Publishing

Our implementation in dokieli automatically includes semantic annotations within the embedded sparqlines. The sparqline resource has its own URI that can be used for global referencing. The RDF statements are represented using the HTML+RDFa syntax, and they preserve the following information:

- The part of the document to which the sparqline belongs (`rel="schema:hasPart"`).
- The human-readable name for the figure (based on the dataset used), where it was derived from (the `qb:DataSet` instance), and the generated SVG.
- The SVG resource has statements to indicate:
  - linked statistical dataset which was used (`rel="prov:wasDerivedFrom"`).
  - human-readable name of the dataset (`property="schema:name"`).
  - license for the generated SVG (`rel="schema:license"`).
  - further information for each `qb:Observation` (`rel="rdfs:seeAlso"`).

This information can be discovered and parsed as RDF, thus making easy to access and reuse by third-party applications. For example, another author can cite or include these sparqlines in their work.

7 Discussion and Conclusions

We have presented a preliminary implementation of sparklines generated from SPARQL endpoints and embedded directly through authoring tool. This allows authors to visualise their data in an optimal way without breaking their workflow. However, there is a lot of scope for future work in this area. We now discuss some areas for further development.

Design principles: Tufte makes recommendations on readability, as well as applying Cleveland's analytical method of choosing aspect ratios *banking to 45°* [2, 10, 11]. Cleveland's method has been extended to generate banked sparklines by providing the vertical dimension to fit a typographical line. These approaches help maximize the clarity of the line segments [12]. Applying these methods is a future implementation in dokieli (issue 159).

Dataset interaction: Building on existing work in faceted searching and browsing of RDF data, authors can explore suitable datasets with a combination of searching using natural-language and filtering through available datasets and dimensions of interest. This approach is convenient for datasets in RDF Data Cubes since they are highly structured and classified. Further work is needed to improve the process for disambiguation of the author's input in natural language in order to discover appropriate URIs in the dataset.

Privacy considerations: Many researchers collect experimental data which has sensitive or identifiable information. This information should not be exposed through public SPARQL endpoints. Measures such as access control lists can allow researchers to generate sparqlines over sensitive data.

Data availability: SPARQL endpoints are notoriously unreliable and they may have high setup costs for new datasets. Applications which rely on endpoints to generate sparqlines with dynamic data, may want to initially include a local cached copy from the last access point in the article. The application can then asynchronously fetch or subscribe for new updates.

Acknowledgements

The motivation and work on sparqlines was inspired by Edward Tufte's education and popularisation of sparklines. Special thanks to Amy Guy and Ilaria Liccardi for their great support and tireless nagging to get this written up, as well as Jindřich Mynarz for help with SPARQL query optimisations.

References

1. Zelchenko, P., Medved, M.: QuoteTracker, http://pete.zelchenko.com/portfolio /screen/2gk.htm
2. Tufte, E.: Beautiful Evidence, Graphics Press, 2006, ISBN 9781930824164, http://www.worldcat.org/title/beautiful-evidence /oclc/70203994&referer=brief_results
3. P. Meharia: Use of Visualization in Digital Financial Reporting: The effect of Sparkline (2012). Theses and Dissertations--Business Administration. Paper 1, http://uknowledge.uky.edu/cgi/viewcontent.cgi?article=1000& context=busadmin_etds
4. Google Docs Sparklines, https://support.google.com/docs/answer/3093289?hl=en
5. Cyganiak, R., Reynolds, D.: The RDF Data Cube vocabulary, W3C Recommendation, 2014, https://www.w3.org/TR/vocab-data-cube/
6. Skjæveland, M. G.: Sgvizler: A JavaScript Wrapper for Easy Visualization of SPARQL Result Sets, 2012, http://2012.eswc-conferences.org/sites/default/files /eswc2012_submission_303.pdf
7. Hage, W. R. v., Marieke v., Malaisé., V.: Linked Open Piracy: A story about e-Science, Linked Data, and statistics (2012), http://www.few.vu.nl/~wrvhage /papers/LOP_JoDS_2012.pdf
8. Percy E. Rivera Salas, P. E. R., Mota, F. M. D., Martin, M., Auer, S., Breitman, K., Casanova, M. A.: Publishing Statistical Data on the Web, ISWC (2012), http://svn.aksw.org/papers/2012/ESWC_PublishingStatisticData/public.pdf
9. Capadisli, S., Auer, S. Riedl, R.: Linked Statistical Data Analysis, ISWC SemStats (2013), http://csarven.ca/linked-statistical-data-analysis
10. Edward Tufte forum: Sparkline theory and practice Edward Tufte, http://www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=0001OR& topic_id=1
11. Cleveland, W.: Visualizing Data, Hobart Press, 1993, ISBN 9780963488404, http://dl.acm.org/citation.cfm?id=529269
12. Heer, J., Maneesh, A.: Multi-Scale Banking to 45°, IEEE Transactions on Visualization and Computer Graphics, Vol. 12, No. 5, 2006, http://vis.berkeley.edu /papers/banking/2006-Banking-InfoVis.pdf