Statistical Graphics Support for Vega-Lite

Ayush Saraf
University of Washington
Computer Science & Engineering
ayush29f@cs.washington.edu
+1 (206) 393-2864

ABSTRACT

In this paper I present an extended grammar for primitive mark types like rule, bar and area to support ranged marks in vega-lite [1], a high-level grammar that enables rapid specification of interactive data visualizations. This extended grammar for primitive marks along with layering [1] in vega-lite then allows for representation of various statistical graphics like error bars, box and whisker plots, confidence intervals, difference charts, bullet graphs, candlestick plots and various other frequently used statistical graphics that are used in a various fields of study like computer science, statistics, financial risk analysis, perceptual psychology, semiotics and many others. I also present a new concept of composite marks in vega-lite to reduce redundancy of the specification grammar for most commonly used statistical graphics like error bars or traditional box and whisker plots. This helps in keeping the vega-lite specification maintain its concise nature.

Author Keywords

…

ACM Classification Keywords

Information visualization, statistical graphics, systems, toolkits, declarative specification.

INTRODUCTION

Declarative languages to represent graphics have recently been widely accepted in the community, however they widely range in expressiveness of the graphics. On one side, protoviz [2], d3.js [3] and vega [4] often referred as visualization grammars (higher level grammars) tend to focus more on the expressivity of the graphic rather ease of use of conciseness of the grammar. On the other, we have ggplot2 [5] and vega-lite [1] often referred as visual analysis grammars (lower lever grammars) which focus more on the ease of use and conciseness of the grammar.

Although, we aim to keep the concise and easy-to-use nature of vega-lite it is worth noting that vega-lite currently does not allow enough expressivity to express most statistical graphics like error bars, box and whisker plots, confidence intervals, difference plots, bullet graphs, candlestick plots and various other frequently. Most of these statistical graphics are widely used in a various fields of study like computer science, statistics, financial risk analysis, perceptual psychology, semiotics and many others. Therefore, we decided we want to extend vega-lite's grammar to be able to support various statistical graphics.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CHI 2009, April 4–9, 2009, Boston, Massachusetts, USA.
Copyright 2009 ACM 978-1-60558-246-7/09/04...$5.00.

RELATED WORK

This project extends the currently existing visual analysis grammar vega-lite [1] which compiles down to a higher level visualization grammar called vega [4]. I also draw some inspirations from visualization grammars like protoviz [2], d3.js [3] ggplot2 [5] and visualization tools like Tableau [6] in the design process.

METHODOLOGY

While extending the current version of vega-lite I make sure to continue to follow the key ideas and principles suggested by Wilkinson's "The Grammar of Graphics" [7].

Ranged Marks

The biggest limitation of vega-lite is the absence of ranged marks, which means it is not possible to create marks like rule, bar, or area to encode both a start and end position on either *x-axis* or *y-axis* (see Figure 2). Therefore, this makes it impossible to make statistical graphics which usually encodes summarized data that requires both a start and end position. Therefore, in the next few sub-sections I will present various specification syntax designs that were considered in the design process.

Encoding Channels

The core idea behind the ranged marks is to have a notion of a start and end of a certain mark (rule, bar, and area mark types). Vega being a higher level grammar does support ranged marks and have encoding channels *x2, y2* along with *x, y* that vega-lite does not. Therefore, the first step was to introduce new encoding channels like *x2, y2* in vega-lite for mark types: *rule, bar, and area.* Since, vega-lite is essentially a compiler that compiles a concise vega-lite specification into a full blown vega it was important to keep the naming conventions consistent with vega. For instance, if someone who used vega before migrating to vega-lite should be able to expect similar behavior. Therefore, we decided not to use something like *yStart, yEnd, xStart, xEnd* that ggplot2 [5] uses to represent error bars. Due to the reasons mentioned above we finally decided to go with *x2, y2* as our names for our new encoding channels.

Also, in cases where the user passes an invalid specification where he uses all four encoding channels *x, y, x2, y2* we decided to disregard *x2* because a by default vega-lite draws a vertical chart therefore *x2.* Further, if the user specifies *y2* without *y* or *x2* without *x* we normalize the spec before passing it to the compiler to implicitly convert *x2* ⇒ *x* and *y2* ⇒ *y.*

Scale

Due to the introduction of the new encoding channels *x2, y2* the Encoding ⇒ Scale in no longer a one-to-one mapping but instead is a many-to-one mapping. Before the introduction of *x2, y2* there were no two encoding channels (*x, y, color, size, opacity, shape, detail, text, row, column*) that mapped to the same scale. However, now (*x, x2*) maps to scale *x* and (*y, y2*) map to scale y. Therefore, when we generating scales in vega, we make sure to collapse not to create scales for *x2 and y2* encoding channels.

Further, since we (*x, x2*) or (*y, y2)* maps to scales *x* and *y* when calculating the domain for the either of those scales we union the individual domains for the two encodings into one in the vega specification.

Finally, it disregards scale properties passed in *x2* or *y2* because encoding channels *x* and *y* also refer to the same scales.

Axis

Similar to scales, due to the introduction of the new encoding channels *x2*, *y2* now there exists two encoding channel corresponding to both *x-axis* and *y-axis*. Therefore, when creating axis for the vega specification, it only considers *x* and *y* encoding channels. Finally, when *x2* and *y2* are asked for their corresponding axis's it returns the *x-axis* and *y-axis* respectively.

Rule Mark

Now that we have introduced the new encoding channels *x2, y2* and modified scales & axis to act accordingly we can finally draw ranged marks. The final specification to make create a ruled mark in vega-lite is as as follows.

```
{
 "data": {
   "values": [
     {"x": "A","y": 28, "y2": 60},
     {"x": "B","y": 55, "y2": 80},
     ...
   ]
 },
 "mark": "rule",
 "encoding": {
   "x": {"field": "x","type": "ordinal"},
   "y": {"field": "y”,"type": "quantitative"},
   "y2": {"field": "y2", "type": "quantitative"}
 }
}
```

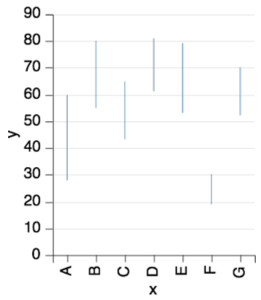

Figure 2. (a) Ranged Rule Mark

Bar Mark

Similar to Rule mark we can create ranged bar marks. Note that we can use aggregate functions like min/max before to make summarized ranged marks.

```
{
 "data": {"url": "data/cars.json"},
 "mark": "bar",
 "encoding": {
   "x": {"field": "Year","type": "ordinal"},
   "y": {
     "aggregate": "min",
     "field": "Miles_per_Gallon",
     "title": "Miles/Gallon ",
     "type": "quantitative"
   },
   "y2": {
     "aggregate": "max",
     "field": "Miles_per_Gallon",
     "type": "quantitative"
   }
 }
}
```

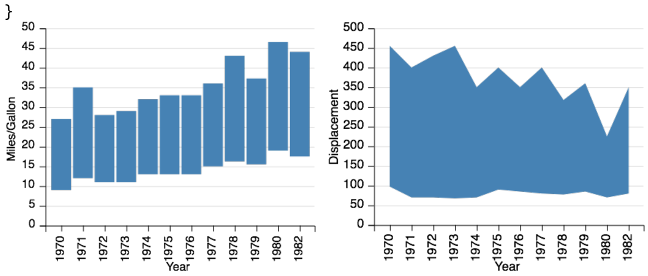

Figure 2. (b) Ranged Bar Mark (c) Area Ranged Mark

Area Mark

Similar to Rule or Bar mark we can create ranged area marks.

```
{
 "data": {"url": "data/cars.json"},
 "mark": "area",
 "encoding": {
   "x": {
     "field": "Year",
     "type": "temporal",
     "timeUnit": "year"
   },
   "y": {
     "aggregate": "min",
     "field": "Displacement",
     "type": "quantitative"
   },
   "y2": {
     "aggregate": "max",
     "field": " Displacement",
     "type": "quantitative"
   }
 }
}
```

Statistical Graphics Using Ranged Marks & Layering

Now that we have added the support for ranged marks (*bar, rule, area*) we have the capability to creating a great variety of statistical graphics using layering vega-lite [1]. Any statistical graphic can be broken into multiple primitive marks like *rule, ticks, bar, area etc.* To demonstrate, here is a simple specification for a confidence interval showing the mean using a *line* mark along maximum and minimum values using a *ranged area* mark.

```
{
 "data": {"url": "data/cars.json"},
 "layers": [
   {
     "mark": "area",
     "encoding": {
       "x": {
         "field": "Year",
         "type": "temporal",
         "timeUnit": "year"
       },
       "y": {
         "aggregate": "min",
         "field": "Miles_per_Gallon",
         "type": "quantitative",
         "axis": {"title": "Miles/Gallon"}
       },
       "y2": {
         "aggregate": "max",
         "field": "Miles_per_Gallon",
         "type": "quantitative"
       },
       "opacity": {"value": 0.3}
     }
   },{
     "mark": "line",
     "encoding": {
       "x": {
         "field": "Year",
         "type": "temporal",
         "timeUnit": "year"
       },
       "y": {
         "aggregate": "mean",
         "field": "Miles_per_Gallon",
         "type": "quantitative"
       }
     }
   }
 ]
}
```

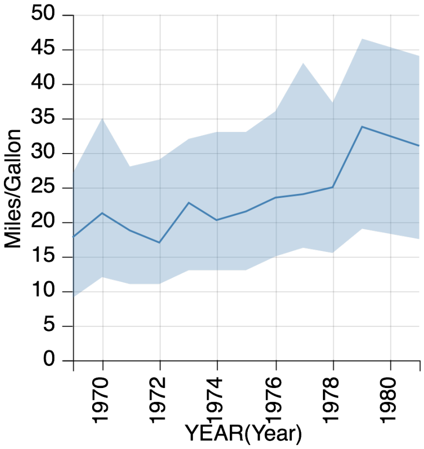

Figure 3. Confidence Interval of maximum and minimum with mean

Composite Marks

As demonstrated in previous section, although we can now create most of the statistical graphics we not only are writing a very redundant specification grammar for a simple confidence interval we also loose the concise nature of vega-lite [1], which is really the most important distinction from its parent grammar vega [4]. For instance, to create a simple box and whisker plot using ranged marks and layering would require 5 layers: 2 for ticks representing the maximum and minimum, 2 for bars representing the quartiles and median, and finally 1 rule for whisker line.

Since we want to keep the concise nature of vega-lite while supporting most commonly used statistical graphics, we propose a new concept of **composite marks** in vega-lite. The concept is really simple; we introduce a few commonly used statistical graphics as **composite marks** which allow our grammar to represent statistical graphics in a concise and less redundant specification. A similar idea used in ggplot2 [5] where they have various statistics marks prefixed with *stat_*.

The way we implement these composite mark is by simply normalizing any composite mark unit specs into its corresponding layers in the normalization step before it even reaches the compilation process. This allows us to easily take advantage to layering and the current design of vega-lite.

Error Bars

Error bars are often used along with other primitive mark types like *point, line, or bar* in order to show uncertainty [8] in data distributions. Therefore, it is useful to be able to quickly add a layer of error bars on top of these primitive marks. An error bar is represented by 3 layers: 2 tick layers for the ends, and 1 layer for the rule. We call the mark type ***errorBar*** and following is a sample specification () for error bars.

```
{
 "data": {"url": "data/gapminder.json"},
 "transform": {"filter": "datum.cluster == 4"},
 "mark": "errorBar",
 "encoding": {
   "x": {"field": "country","type": "nominal"},
   "y": {
     "field": "life_expect",
     "type": "quantitative",
     "aggregate": "min",
     "title": "Life Expectancy"
   },
   "y2": {
     "field": "life_expect",
     "type": "quantitative",
     "aggregate": "max"
   }
 }
}
```

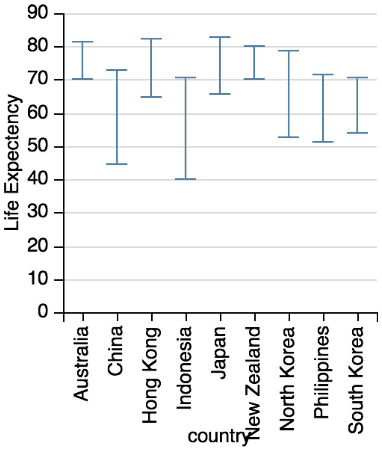

Figure 4. Error Bars showing maximum and minimum value

Box and Whisker Plots

Box and whisker plot is one of the most popular choice for data summarization among the statisticians. Creating a composite mark for a box and whisker plots is very essential because not only it is widely used but also it is made up of 5 layers as mentioned earlier. Another we decided to make a composite mark for box and whisker plot is that it essentially has 2 critical encoding channels *x* and *y*, and this makes the specification to be very concise. Here is a specification we considered for box and whisker plots.

```
{
 "data": {"url": "data/gapminder.json"},
 "transform": {"filter": "datum.cluster == 4"},
 "mark": "errorBar",
 "encoding": {
   "x": {"field": "country","type": "nominal"},
   "y": {
       "aggregate": "boxMinMax",
       "field": "life_expect",
       "type": "quantitative"
   }
 }
}
```

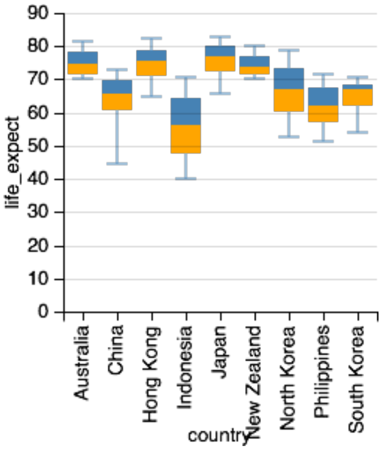

Figure 5. Box Plot with showing minimum and maximum, median and the quartiles

Since there are various varieties of box and whisker plots we decided to support a small subset of those (currently only the minimum and maximum). The aggregate field specifies the type of aggregation needs to applied to the box plots. For example, ***boxMinMax*** is used to create the simplest box and whisker plot show in Figure 5.

RESULTS

As a result of this project, we have now added

DISCUSSION

FURURE WORK

CONCLUSION

It is important that you write for the SIGCHI audience. Please read previous years' *Proceedings* to understand the writing style and conventions that successful authors have used. It is particularly important that you state clearly what you have done, not merely what you plan to do, and explain how your work is different from previously published work, i.e., what is the unique contribution that your work makes to the field? Please consider what the reader will learn from your submission, and how they will find your work useful. If you write with these questions in mind, your work is more likely to be successful, both in being accepted into the Conference, and in influencing the work of our field.

ACKNOWLEDGMENTS

I would like to thank my instructor and TAs for Jeef Heer, Michael Connel.

REFERENCES

1. Adobe Acrobat Reader 7. http://www.adobe.com/products/acrobat/.
2. Anderson, R.E. Social impacts of computing: Codes of professional ethics. *Social Science Computing Review 10*, 2 (1992), 453-469.
3. How to Classify Works Using ACM's Computing Classification System. http://www.acm.org/class/how_to_use.html.
4. Klemmer, R.S., Thomsen, M., Phelps-Goodman, E., Lee, R. and Landay, J.A. Where do web sites come from? Capturing and interacting with design history. In *Proc. CHI 2002*, ACM Press (2002), 1-8.
5. Mather, B.D. Making up titles for conference papers. *Ext. Abstracts CHI 2000*, ACM Press (2000), 1-2.
6. Schwartz, M. *Guidelines for Bias-Free Writing*. Indiana University Press, Bloomington, IN, USA, 1995.
7. Zellweger, P.T., Bouvin, N.O., Jehøj, H., and Mackinlay, J.D. Fluid Annotations in an Open World. *Proc. Hypertext 2001*, ACM Press (2001), 9-18.