

Report

GP Team 2015

February 4, 2016

Contents

1	Introduction	3
2	Literature review	4
2.1	Data Visualization	4
2.1.1	Definition	4
2.1.2	Data visualization importance	4
3	Grammar of graphics	5
3.1	What is the meaning of Grammar?	5
3.2	What is the Grammar of Graphics?	5
3.3	Grammar of graphics features	5
4	Comparative study among GoG libraries	6
5	Our Approach	6
6	Team members playrolls	6
6.1	Raafat Sobhy	6
6.2	Sherif Embarak	7
6.3	Ahmed Fouad	9
6.4	Yusuf Mohamed	9
7	Gantt Chart	10
8	Deliverable	10

1 Introduction

What is a graphic? How can we succinctly describe a graphic? And how can we create the graphic that we have described?

These are important questions for the field of statistical graphics. One way to answer these questions is to develop a grammar . A good grammar will allow us to gain insight into the composition of complicated graphics, and reveal unexpected connections between seemingly different graphics.

A grammar provides a strong foundation for understanding a diverse range of graphics. A grammar may also help guide us on what a wellformed or correct graphic looks like, but there will still be many grammatically correct but nonsensical graphics.

This is easy to see by analogy to the English language: good grammar is just the first step in creating a good sentence. Grammar makes language expressive. A language consisting of words and no grammar (statement = word) expresses only as many ideas as there are words. By specifying how words are combined in statements, a grammar expands a language's scope.

In other hand grammar of graphics is a tool that enable us to concisely describe the components of graphic. Such a grammar allow us to move beyond named graphics scatterplot and gain insight into the deep structure that underlies statistical graphics .

The power of the grammar is illustrated with a selection of examples that explore different components and their interactions .

2 Literature review

2.1 Data Visualization

2.1.1 Definition

Data visualization is the presentation of data in a pictorial or graphical format. For centuries, people have depended on visual representations such as charts and maps to understand information more easily and quickly.

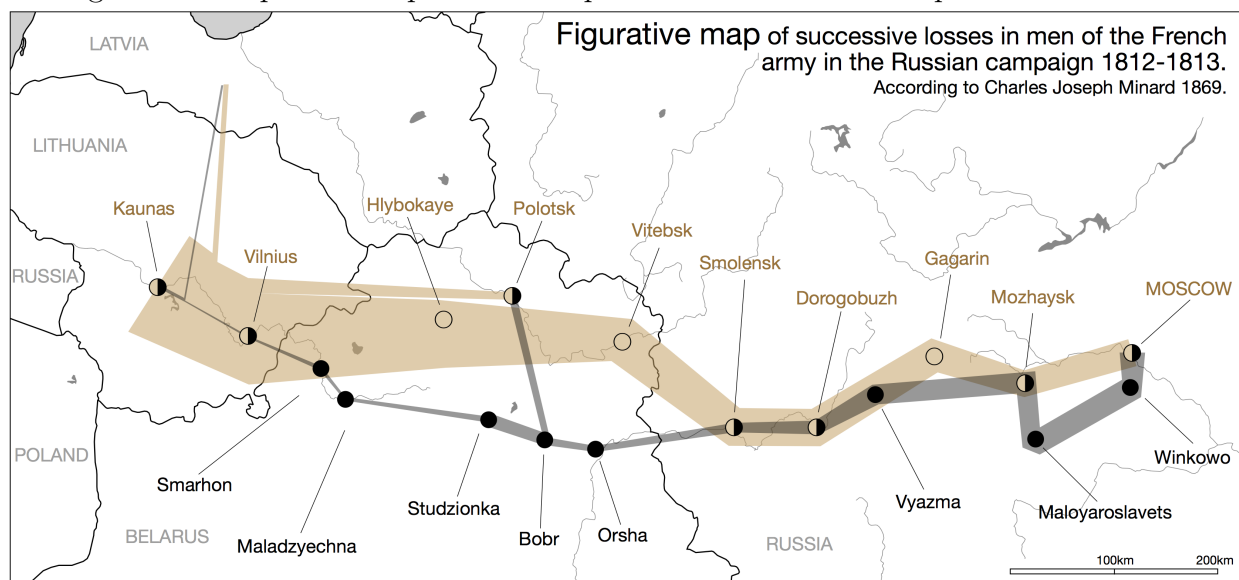
Because of the way the human brain processes information, it is faster for people to grasp the meaning of many data points when they are displayed in charts and graphs rather than poring over piles of spreadsheets or reading pages and pages of reports.

2.1.2 Data visualization importance

Visualizations help people see things that were not obvious to them before.

A spreadsheet cannot visually represent the information due to data presentation limitations, would spend hours searching among thousands of rows and columns of data with still no concrete answer about the relationship between two factors.

Figure 1: The path of Napoleon's troops across the Russian Empire of Alexander I



3 Grammar of graphics

3.1 What is the meaning of Grammar?

the whole system and structure of a language or of languages in general, usually taken as consisting of syntax and morphology (including inflections) and sometimes also phonology and semantics.

So if we think about the graphics as a language from the perspective of the grammar we should put in our minds those rules (grammar) that manage that manage these sentences or the components of the language (charts).

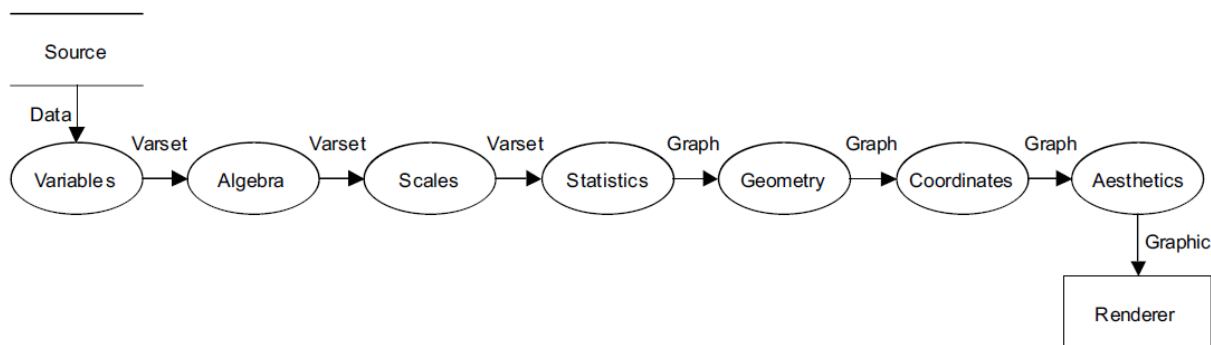
3.2 What is the Grammar of Graphics?

The regime of constructing the graphs depending on predefined rules, Hence The grammar of graphics takes us beyond a limited set of charts (words) to an almost unlimited world of graphical forms (statements).

3.3 Grammar of graphics features

- Orthogonal set of features describes all common charts, Virtually all uncommon charts.
- Language is flexible enough to
 - describe our known chart types
 - describe unknown chart types

Figure 2: Grammar of graphics layers



4 Comparative study among GoG libraries

Table 1: Comparative Study

Library	ggplot2	ggvis	ggd3	vega
Implementation	Fully Implemented	Fully Implemented	Partially Implemented	Partially Implemented GOG
Language	R	R	JavaScript	JavaScript
Issue	No interactivity.	Limited Interactivity	-Have fixed set of geoms. -No interactivity.	Set of static geometries.
Description	It takes care of many of the fiddly details that make plotting a hassle (like drawing legends) as well as providing a powerful model of graphics that makes it easy to produce complex multi-layered graphics.	-Declaratively describe data graphics with a syntax similar in spirit to ggplot2. -Create rich interactive graphics that you can play with locally in Rstudio or in your browser.	Stop development	Vega is a declarative format for creating, saving, and sharing visualization designs.

5 Our Approach

Design & Implement a library that is based on Grammar of graphics to be the baseline of continuous contribution in the field of data visualization.

1. GoG library.
 - built with javascript runs on Node.js environment
2. Application Layer.
 - built on top of Electron.
3. Package Manager.
 - which will manage application extensions and plugins.

6 Team members playrolls

6.1 Raafat Sobhy

- List & implement Coordinate systems
 - Number line
 - Cartesian coordinate system

- Polar coordinate system
- Cylindrical and spherical coordinate systems
- Homogeneous coordinate system

6.2 Sherif Embarak

- adding full documentation to all function

- Example :

```
# Project ||GoGLib
#Module ||Stat Methods
#Author ||Sherif Emabrak
#Description ||Given a list of  $n$  nodes ( $n$  assumed even) represented by the
#               indices  $(1, 2, 3, \dots)$ , the join method returns a list of edges
#               represented by the indices  $(1, 1 + n/2), (2, 2 + n/2), \dots$ . This method
#               is useful for blending two varsets of nodes because it pairs the
#               first node in the first column with the first node
#               in the second, and so on.
# -----
```

- partial implementation for link, bin, region and summary functions

- link functions :

- * join.

```
join = (input_array ...) ->
  n = input_array.length
  result = []
  for key, value of input_array
    result.push([value, value+n/2])
  result
```

- * sequence.

```
sequence = (input_array ...) ->
  result = []
  for key, value of input_array
    temp=key
    key=value
    value=input_array[parseInt(temp)+1]
    result.push([key, value])
  result
```

- bin functions :
- summary functions :

* sum.

```
sum = (input_array...) ->
    total=0;
    for count in input_array
        total+=count
    total
```

* mean.

```
sum = (input_array...) ->
    total=0;
    for count in input_array
        total+=count
    total

mean = (input_array...) ->
    (sum input_array...)/input_array.length
```

* mode.

```
mode = (input_array...) ->
    counter = 1
    max = 0;
    result = input_array[0]

    input_array.sort()

    for i in [1...input_array.length-1]
        if (input_array[i]==input_array[i+1])
            counter++
            if (counter > max)
                max=counter
                result = input_array[i]
        else
            counter = 1

    result
```

* median.


```

median = (input_array...) ->
    size = input_array.length
    posetion = 0
    result = 0
    if size%2 isnt 0
        posetion = (size-1)/2
        result = input_array[posetion]
    else
        posetion = size / 2
        result = (input_array[posetion]+
            input_array[posetion-1])/2
    result

```

* log.

```

log = (input) ->
    Math.log(input)

```

- region functions :
- smooth functions :
 - * sum.
 - * mean.
 - * mode.
 - * median.
 - * log.
- importing csv files to js objects
 - * Import CSV files and treat each row as JSON object
 - * Show data in the browser as table
 - * Add Type, Tag, or etc for table or column.

6.3 Ahmed Fouad

- List predefined figures in GoG book
- implementing geometric components (point, line, circle)

6.4 Yusuf Mohamed

- Package Manager for application layer.

- * A cli utility tool which fetches & removes plugins (extensible piece of code) from npm and places it in specific folder. once the application bootstraps it will check all the existing plugins in the folder to add them which will provide feature rich application.
- * why use a new directory for plugins?
 - The idea behind using a new directory other than `node_modules` is that the application will lookup plugins in it, and application shouldn't lookup plugins in the `node_modules` folders, as it will be another problem to differentiate between plugins and other node modules. So, the separation is needed to help the visualization application lookup plugins in a folders containing plugins only.
- * Works on all major platforms like Windows, Linux and Mac.
- * Took in consideration various plugin manager design like apm (`atom package manager`) and Brackets plugins.
- * why didn't we use apm or brackets plugin manager?
 - APM : no documentation or how-to, limits the plugins to be hosted on github only.
 - Brackets : limits the developer with a specific structure and specific keywords to make his plugin up and running.
- * Usage
 - Add Or Update a plugin.

vispack install plugin_name

- Remove a plugin

vispack remove plugin_name

– Application Layer

- * Made with electron formerly known as NW.js which is a cross platform application to distribute the application on all major platforms and photon kit for styling.
- * Interprocess communication `ipc` to coordinate activities among different program process (main process and rendering process) via sending and receiving messages between the mentioned processes.

7 Gantt Chart

8 Deliverable

Partial implementation of GoG library which will make us able to generate the following charts

Figure 3: Prototype Design

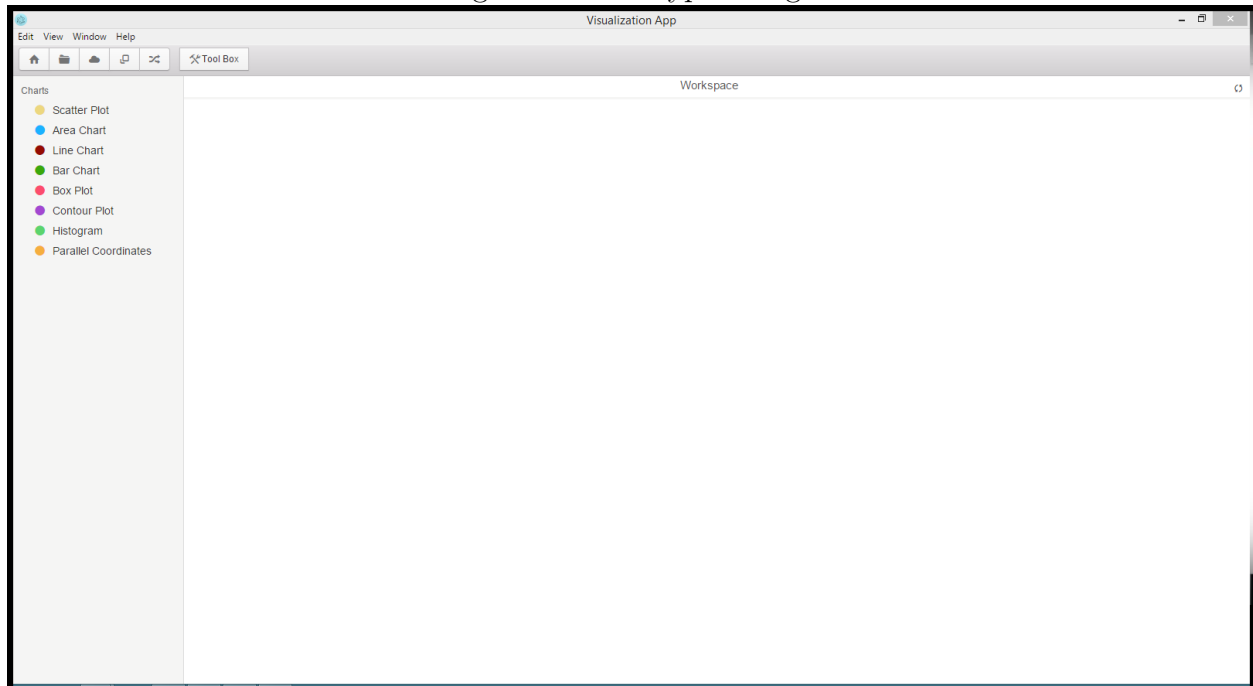
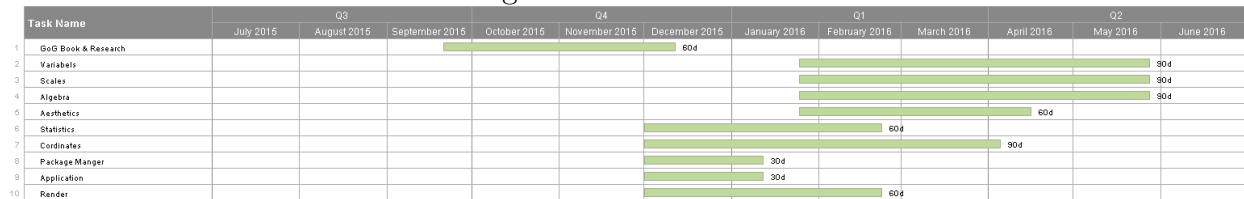


Figure 4: Gantt chart



- Scatter Plot
- Area Chart
- Line Chart
- Bar Chart
- Histogram
- Parallel Coordinates
- Scatter Plot Matrix

References

- [1] Vega <http://vega.github.io/>
- [2] ggplot2 <http://ggplot2.org/>

- [3] ggvis <http://ggvis.rstudio.com/>
- [4] ggd3 <http://benjh33.github.io/ggd3/>