Using the Google Visualisation API with R: googleVis-0.2.1 Package Vignette

Markus Gesmann[*], Diego de Castillo[†]
Contact: rvisualisation@gmail.com

November 29, 2010

Abstract

The `googleVis` package provides an interface between R and the Google Visualisation API. The Google Visualisation API offers interactive charts which can be embedded into web pages. The most well known of those charts is probably the Motion Chart, popularised by Hans Rosling in his TED talks. With the `googleVis` package users can create web pages with interactive charts based on R data frames easily and display them either via the `R.rsp` package or within their own sites. Currently the package provides interfaces to Motion Charts, Annotated Time Lines, Maps, Geo Maps, Tables and Tree Maps.

[*]markus.gesmann@gmail.com
[†]decastillo@gmail.com

Contents

1 Introduction **3**

1.1 Motivation 3

1.2 Google Visualisation API 5

2 The `googleVis` package **6**

2.1 Installation 6

2.2 Using the `googleVis` package 8

2.3 Motion Chart Example 9

2.4 Displaying `gvis` objects 13

2.5 Using `googlVis` with `rsp` 14

3 Contact **14**

3.1 Collaboration 14

3.2 Citation 15

3.3 Training and consultancy 15

References **17**

1 Introduction

1.1 Motivation

More and more data is becoming available and accesible, and yet stories and insights are still often missed: we are lost in the data jungle and struggle to see the wood for the trees.

Hence new tools are required to bring data to life, to engage with users, to enable them to slice and dice the data, to view it from various angles and to find stories worth telling: outliers, trends or even the bleeding obvious.

In 2006 Hans Rosling gave an inspiring talk at TED [Ros06] about social and economic developments in the world over the last 50 years, which challenged the views and perceptions of many listeners. Rosling had used extensive data analysis to come to his conclusions. To visualise his talk, he and his team at Gapminder [Fou10a] had developed animated bubble charts, aka motion charts, see Figure 1.

Rosling's presentation popularised the idea and use of interactive charts, and as one result the software behind Gapminder was bought by Google and integrated as motion charts into their Visualisation API [Inc10i] a few years later.

We also noticed that data journalism has grown over the recent years. The data blogs of the Guardian (UK), and taz.de (Die Tageszeitung, Germany) have brought data analysis and data visualisation to a wider audience.

In 2010 Sebastián Pérez Saaibi [Saa10] presented at the R/Rmetrics Workshop on Computational Finance and Financial Engineering the idea to link Google motion charts with R using the `R.rsp` package [Ben09].

Inspired by those talks and the desire to use interactive data visualisation tools to foster the dialogue between data analysts and others the authors of this vignette started the development of the `googleVis` package [GdC10].

Of course there are many other alternative visualisation toolkits out there, e.g. Many Eyes [RtICsg10], Open Flash Chart (Flash) [JG10], OpenLayers (JavaScript) [Fou10b], Processing (Java) [FR10], simile (AJAX) [DKM10] and FLARE (ActionScript) [Lab10] to name but a few.

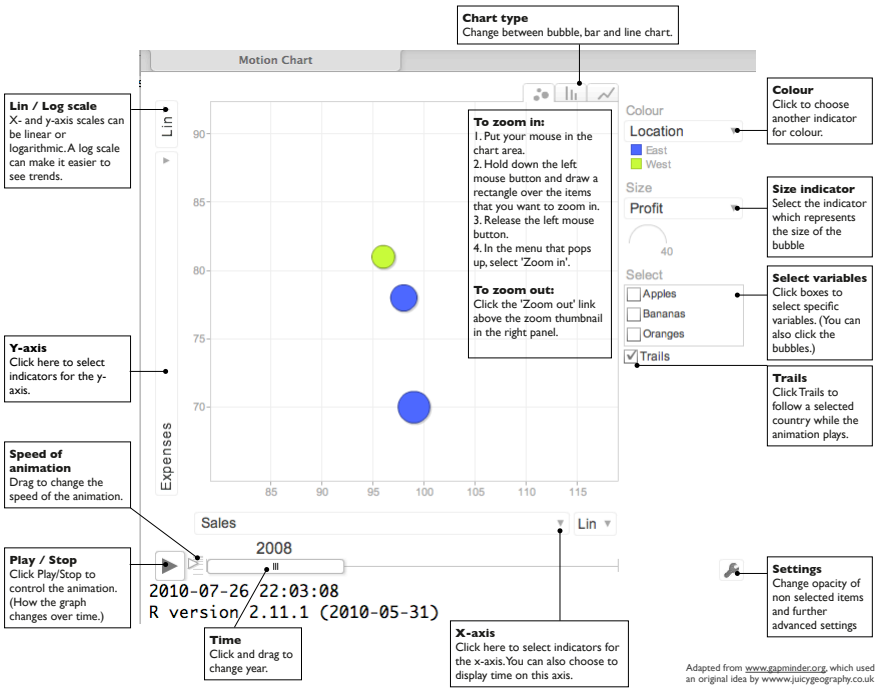


Figure 1: Overview of a Google Motion Chart. Screenshot of the output of `plot(gvisMotionChart(Fruits, idvar='Fruit', timevar='Year'))`

1.2 Google Visualisation API

The Google Visualisation API [Inc10i], [Inca] allows users to create interactive charts as part of Google documents, spreadsheets and web pages. In this text we will focus on the usage of the API as part of web sites.

The Google Public Data Explorer [Inc10e] provides a good example, demonstrating the use of motion charts and how they can help to analyse data. Please note, that all those charts are rendered within a browser using Adobe Flash [Incb].

The charting data can either be embedded into the html file or read dynamically. Key to the Google Visualisation API is that the data is structured in a DataTable [Inc10h], and this is where the `googleVis` package helps, as it uses the functionality of the `RJSONIO` package [Lan10] to transform R data frames into JSON [JSO06] objects as the basis for a DataTable.

As an example we shall look at the html-code of a motion chart from Google's visualisation gallery [Inc10d], which generates output similar to Figure 1:

```
<html>
  <head>
    <script type="text/javascript" src="http://www.google.com/jsapi">
    </script>
    <script type="text/javascript">
      google.load('visualization', '1', {'packages':['motionchart']});
      google.setOnLoadCallback(drawChart);
      function drawChart() {
        var data = new google.visualization.DataTable();
        data.addColumn('string', 'Fruit');
        data.addColumn('date', 'Date');
        data.addColumn('number', 'Sales');
        data.addColumn('number', 'Expenses');
        data.addColumn('string', 'Location');
        data.addRows([
          ['Apples',new Date (1988,0,1),1000,300,'East'],
          ['Oranges',new Date (1988,0,1),1150,200,'West'],
          ['Bananas',new Date (1988,0,1),300,250,'West'],
          ['Apples',new Date (1989,6,1),1200,400,'East'],
          ['Oranges',new Date (1989,6,1),750,150,'West'],
          ['Bananas',new Date (1989,6,1),788,617,'West']
          ]);
        var chart = new google.visualization.MotionChart(
                    document.getElementById('chart_div'));
        chart.draw(data, {width: 600, height:300});
      }
    </script>
  </head>
```

```
  <body>
    <div id="chart_div" style="width: 600px; height: 300px;"></div>
  </body>
</html>
```

You will notice that the above html-code has three generic parts:

- reference to a javascript function provided by Google, here `'motionchart'`,
- data to visualise as a `DataTable`,
- chart with chart id (`'chart_div'`) and options, here width and height.

Those principles hold true for most of the interactive charts of the Google Visualisation API, see the examples in Figure 2.

To display the visualisation, the html-page must be loaded from a web server in a browser with internet connection and Flash; it will not work when loaded as a local file. For more details see the Google Visualisation API documentation [Inc10d].

Fortunately, the package `R.rsp` [Ben09] provides an internal cross-platform web server which can be started from the R console; it serves the `googleVis` package as a tool to display web pages. Additionally the `R.rsp` web server has the capability to extract and execute R code from html code, similar to the approach taken by Sweave [Lei02] for LaTeX. For more details see the `R.rsp` documentation.

2 The `googleVis` package

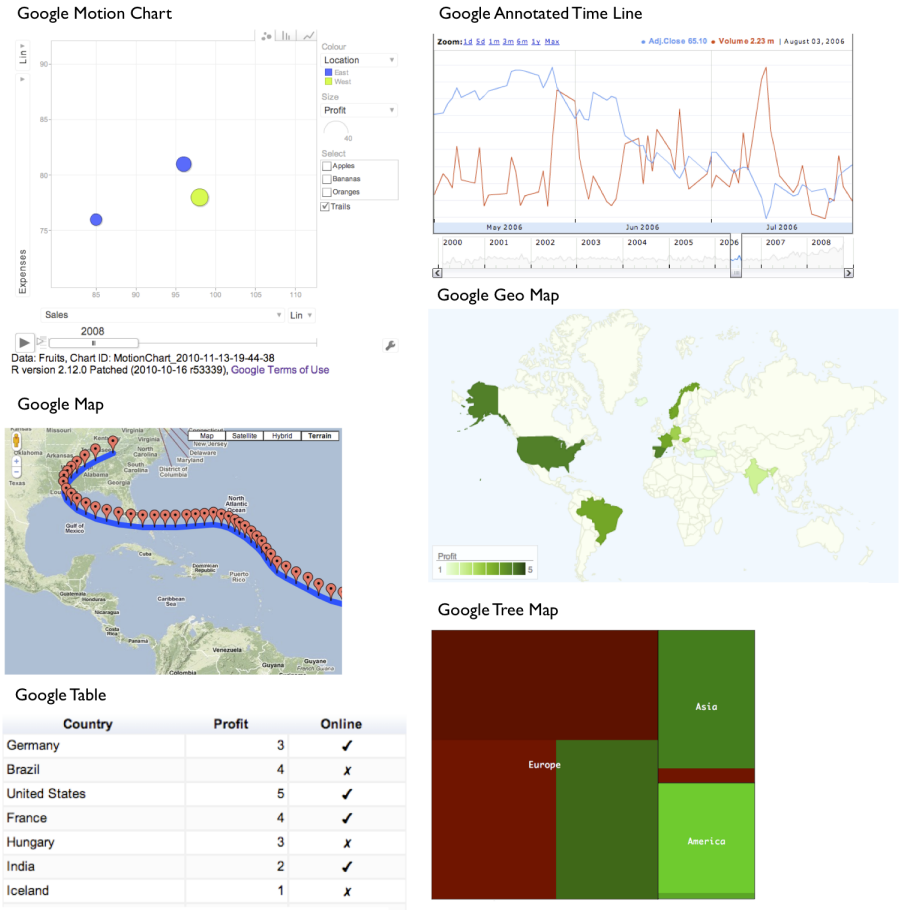
The `googleVis` package provides an interface between R and the Google Visualisation API. The functions of the package allow the user to visualise data stored in R data frames with the Google Visualisation API. To view the output a browser with Flash and internet connection is required.

Currently the package provides interfaces to Motion Chart [Inc10d], Annotated Time Line [Inc10a], Geo Map [Inc10b], Map [Inc10c], Table [Inc10f] and Tree Map [Inc10g], see Figure 2 for examples.

2.1 Installation

The `googleVis` package depends on the `RJSONIO` and `R.rsp` packages, so we need to install those as well. We can install `R.rsp`, `RJSONIO` and `googleVis` in the usual way from CRAN, e.g.:

```
R> install.packages(c('R.rsp', 'RJSONIO', 'googleVis'),
+                   repos='http://cran.r-project.org')
```



Country	Profit	Online
Germany	3	✓
Brazil	4	✗
United States	5	✓
France	4	✓
Hungary	3	✗
India	2	✓
Iceland	1	✗

Figure 2: Screenshot of some of the outputs of `demo(googleVis)` with `gvisMotionChart`, `gvisAnnotatedTimeLine`, `gvisMap`, `gvisGeoMap`, `gvisTable` and `gvisTreeMap` from top left to bottom right.

The installation was successful if the command `library(googleVis)` gives you the following message:

```
R> library(googleVis)
```

Please note that by default the visualisation output files will be written into the `rsp/myAnlysis` folder of the `googleVis` library. The following R-command will show the path to this folder:

```
R> system.file(file.path("rsp", "myAnalysis"), package="googleVis")
```

It is preferable that you have write-access to this directory. So you may want to install the package locally (e.g. in your home folder). Please contact your system administrator first if you need help.

2.2 Using the `googleVis` package

The individual functions of the `googleVis` package are documented in detail in the help pages. Here we will cover only the principles of the package.

As an example we will show how to generate a motion chart as displayed in Figure 1. It works analogue for the other APIs. Further examples are covered in the demos of the `googleVis` package, see also Figure 2.

The design of the visualisation functions are fairly generic. The name of the visualisation function is `'gvis' + ChartType`. So for the Motion Chart we have:

```
gvisMotionChart(data, idvar='id', timevar='date', options=list())
```

Here `data` is the input `data.frame` and `idvar` and `timevar` specify the column names of the id variable and time variable for the plot, while display options are set in an optional list. The options and data requirements follow those of the Google Visualisation API and are documented in the help pages, see `?gvisMotionChart`.

The output of a `googleVis` function is a list of lists containing information about the chart type, chart id and the html code in a sub-list with header, chart, caption and footer.

The idea behind this concept is that users can get on one hand side a complete web page, but on the other hand can extract specific parts, such as the chart. This is particular helpful if the package functions are used in solutions where the user wants to feed the visualisation output into other sites, or would like to embed them into rsp-pages, see page 14.

The output of a `googleVis` function will be of class `'gvis'` and `'list'`. Generic print (`print.gvis`) and plot (`plot.gvis`) functions exist to ease the handle of such objects.

To illustrate the concept we shall create a motion chart using the Fruits sample data set.

2.3 Motion Chart Example

Following the documentation of the Google Motion Chart API we need a data set which has at least four columns; one identifying the variable we would like to plot, one time variable and at least two numerical variables. Further numerical and character columns are allowed.

As an example we shall use the `Fruits` data set:

```
R> data(Fruits)
R> Fruits
```

```
    Fruit Year Location Sales Expenses Profit       Date
1  Apples 2008     West    98       78     20 2008-12-31
2  Apples 2009     West   111       79     32 2009-12-31
3  Apples 2010     West    89       76     13 2010-12-31
4 Oranges 2008     East    96       81     15 2008-12-31
5 Bananas 2008     East    85       76      9 2008-12-31
6 Oranges 2009     East    93       80     13 2009-12-31
7 Bananas 2009     East    94       78     16 2009-12-31
8 Oranges 2010     East    98       91      7 2010-12-31
9 Bananas 2010     East    81       71     10 2010-12-31
```

In this case we will use the columns `'Fruit'` and `'Year'` as id and time variable respectively. However we could heave used also `'Date'` instead of `'Year'`.

```
R>  M <- gvisMotionChart(Fruits, idvar="Fruit", timevar="Year")
```

The structural output of `gvisMotionChart` is a list of list as described above

```
R>  str(M)
```

```
List of 3
 $ type   : chr "MotionChart"
 $ chartid: chr "MotionChart_2010-11-29-22-06-18"
 $ html   :List of 4
  ..$ header : chr "\n<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0
  ..$ chart  : chr "<!-- MotionChart generated in R 2.12.0 by goog
  ..$ caption: chr "Data: Fruits, Chart ID: MotionChart_2010-11-29
  ..$ footer : chr "\n<%@include file=\"../src/simpleFooter.rsp\"%
 - attr(*, "class")= chr [1:2] "gvis" "list"
```

The first two items of the list contain information about the chart type used and an individual chart id generated at run time from the chart type and date:

```
R> M$type
```

```
[1] "MotionChart"
```

```
R> M$chartid
```

```
[1] "MotionChart_2010-11-29-22-06-18"
```

The html output list is then split into header, chart, caption and footer. This allows the user to extract only certain parts of the page, or to create a complete html page.

The header part of the html page has only basic html tags and includes two rsp-files to provide a simple layout framework.

```
R> cat(M$html$header)
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
 "http://www.w3.org/TR/REC-html40/loose.dtd">
<html>
<%@include file="../src/simpleHead.rsp"%>
<body>
<%@include file="../src/simpleHeader.rsp"%>
```

The actual Google Visualisation is stored in the `chart` item of the html list.

```
R> cat(M$html$chart)
```

```
<!-- MotionChart generated in R 2.12.0 by googleVis 0.2.1 package -->
<!-- Mon Nov 29 22:06:18 2010 -->


<script type="text/javascript" src="http://www.google.com/jsapi">
</script>
<script type="text/javascript">
google.load("visualization", "1", { packages:["motionchart"] });
google.setOnLoadCallback(drawChart);
function drawChart() {
var data = new google.visualization.DataTable();
var datajson = [
 [
```

```
 "Apples",
  2008,
"West",
    98,
    78,
    20,
"2008-12-31"
],
[
 "Apples",
  2009,
"West",
   111,
    79,
    32,
"2009-12-31"
],
[
 "Apples",
  2010,
"West",
    89,
    76,
    13,
"2010-12-31"
],
[
 "Oranges",
  2008,
"East",
    96,
    81,
    15,
"2008-12-31"
],
[
 "Bananas",
  2008,
"East",
    85,
    76,
     9,
"2008-12-31"
],
[
 "Oranges",
```

```
  2009,
"East",
    93,
    80,
    13,
"2009-12-31"
],
[
 "Bananas",
  2009,
"East",
    94,
    78,
    16,
"2009-12-31"
],
[
 "Oranges",
  2010,
"East",
    98,
    91,
     7,
"2010-12-31"
],
[
 "Bananas",
  2010,
"East",
    81,
    71,
    10,
"2010-12-31"
]
];
data.addColumn('string','Fruit');
data.addColumn('number','Year');
data.addColumn('string','Location');
data.addColumn('number','Sales');
data.addColumn('number','Expenses');
data.addColumn('number','Profit');
data.addColumn('string','Date');
data.addRows(datajson);
var chart = new google.visualization.MotionChart(
   document.getElementById('MotionChart_2010-11-29-22-06-18')
);
```

```
var options ={};
options["width"] =    600;
options["height"] =    500;
chart.draw(data,options);
}
</script>
<div id="MotionChart_2010-11-29-22-06-18" style="width: 600px; height: 500px;">
</div>
```

A basic chart caption and html footer are the final items of the html list:

```
R> cat(M$html$caption)
```

```
Data: Fruits, Chart ID: MotionChart_2010-11-29-22-06-18
<BR>
R version 2.12.0 Patched (2010-10-16 r53339),
<a href="http://code.google.com/apis/visualization/terms.html">
Google Terms of Use</a>
<BR>
<BR>
```

```
R> cat(M$html$footer)
```

```
<%@include file="../src/simpleFooter.rsp"%>
</body>
</html>
```

2.4 Displaying `gvis` objects

To display the page locally just type:

```
R> plot(M)
```

The plot method for `gvis` object will automatically create by default a rsp-file in the `rsp/myAnalysis` folder of the `googleVis` package using the type and chart id information of the object, and it will display the output using the local web server of the `R.rsp` package.

The following R-command will show you the path to this folder:

```
R> system.file(file.path("rsp", "myAnalysis"), package="googleVis")
```

If you would like to write the output into a different directory, e.g. your web server repository, then you can set the arguments for `filename` and `repos` (web server address of the repository) individually.

Further examples are part of the googleVis demo, including one example demonstrating how the output of several visualisations can be incorporated into a single page.

2.5 Using googlVis with rsp

The R.rsp package allows the user to integrate R code into html-code, which together with the R.rsp web server are executed at run time.

As mentioned above the R.rsp package allows us to dynamically generate documents into static content using R Server Pages. This means we can mix html and R code to create content on the fly. As an example, we can embed the above motion chart into a rsp-page:

```
<html>
<body>
<%=gvisMotionChart(Fruits, idvar="Fruit", timevar="Year")$html$Chart%>
</body>
</html>
```

You find a few examples a part of the googleVis package. Those examples can be displayed via the following R command:

```
R> browseRsp("http://127.0.0.1:8074/library/googleVis/rsp/")
```

The actual rsp-file is located within the googleVis package directory, and again R allows you to find the file with the following R command:

```
R> filePath(system.file("rsp", package = "googleVis"), "index.rsp")
```

Please read also the documentation of the R.rsp package.

3 Contact

3.1 Collaboration

Obviously, the package is work in progress and there are many other functions of the Google Visualisation API which are still untouched.

Please feel free to send us an email rvisualisation@gmail.com if you would like to be kept informed of new versions, or if you have any feedback, ideas, suggestions or would like to collaborate.

3.2 Citation

Please cite R and/or googleVis if you use it in your work or publications. Use

```
R> citation()
```

or

```
R> citation("googleVis")
```

for information on how to cite the software.

3.3 Training and consultancy

Please contact us if you would like to discuss tailored training or consultancy: rvisualisation@gmail.com

References

[Ben09] Henrik Bengtsson. R.rsp: R server pages. http://CRAN.R-project.org/package=R.rsp, 2009. R package version 0.3.6.

[DKM10] MacKenzie Smith (MIT Libraries) David Karger (MIT CSAIL). Simile: Semantic Interoperability of Metadata and Information in unLike Environments. http://simile.mit.edu/, 2010.

[Fou10a] Gapminder Foundation. Gapminder. http://www.gapminder.org, 2010.

[Fou10b] Open Source Geospatial Foundation. Openlayers: Free maps for the web. http://www.openlayers.org/, 2010.

[FR10] Ben Fry and Casey Reas. Processing an open source programming language and environment to create images, animations, and interactions. http://processing.org/, 2010.

[GdC10] Markus Gesmann and Diego de Castillo. googleVis: Using the Google Visualisation API with R. http://code.google.com/p/google-motions-chart-with-r/, 2010. R package version 0.2.0.

[Inca] Google Inc. Google Visualization API Terms of Service. http://code.google.com/apis/visualization/terms.html.

[Incb] Adobe Systems Incorporated. Adobe Flash Player. http://get.adobe.com/flashplayer/.

[Inc10a] Google Inc. Google Annotated Time Line API. http://code.google.com/apis/visualization/documentation/gallery/annotatedtimeline.html, 2010.

[Inc10b] Google Inc. Google Geo Map API. http://code.google.com/apis/visualization/documentation/gallery/geomap.html, 2010.

[Inc10c] Google Inc. Google Geo Map API. http://code.google.com/apis/visualization/documentation/gallery/map.html, 2010.

[Inc10d] Google Inc. Google Motion Chart API. http://code.google.com/apis/visualization/documentation/gallery/motionchart.html, 2010.

[Inc10e] Google Inc. Google Public Data Explorer. http://www.google.com/publicdata/home, 2010.

[Inc10f] Google Inc. Google Table API. http://code.google.com/apis/visualization/documentation/gallery/table.html, 2010.

[Inc10g] Google Inc. Google Tree Map API. http://code.google.com/apis/visualization/documentation/gallery/treemap.html, 2010.

[Inc10h] Google Inc. Google Visualisation Reference. http://code.google.com/apis/visualization/documentation/reference.html, 2010.

[Inc10i] Google Inc. Google Visualization API. http://code.google.com/apis/visualization/documentation/gallery.html, 2010.

[JG10] George Neusse John Glazebrook, Guenther Harrasser. Open flash chart. http://teethgrinder.co.uk/open-flash-chart/, 2010.

[JSO06] JSON.org. JSON. http://www.json.org/, 2006. RFC 4627 application/json.

[Lab10] UC Berkeley Visualization Lab. flare: Data visualisation for the web. http://flare.prefuse.org, 2010.

[Lan10] Duncan Temple Lang. RJSONIO: Serialize R objects to JSON, JavaScript Object Notation. http://www.omegahat.org/RJSONIO/, 2010. R package version 0.3-0.

[Lei02] Friedrich Leisch. Sweave: Dynamic generation of statistical reports using literate data analysis. In Wolfgang Härdle and Bernd Rönz, editors, *Compstat 2002 — Proceedings in Computational Statistics*, pages 575–580. Physica Verlag, Heidelberg, 2002. ISBN 3-7908-1517-9.

[Ros06] Hans Rosling. TED Talk: Hans Rosling shows the best stats you've ever seen. http://www.ted.com/talks/hans_rosling_shows_the_best_stats_you_ve_ever_seen.html, 2006.

[RtICsg10] IBM Research and the IBM Cognos software group. Many eyes. http://services.alphaworks.ibm.com/manyeyes/page/Create_a_Visualization.html, 2010.

[Saa10] Sebastián Pérez Saaibi. *R/RMETRICS Generator Tool for Google Motion Charts*. https://www.rmetrics.org/, 2010. Meielisalp, Lake Thune Switzerland, June 27 - July 1, 2010.