An R-Based Interface to the Google Visualisation API

John Maindonald[1]

[1]Centre for Mathematics & Its Applications, Australian National University

July 19, 2012

googleVis: Linking to the Google Visualisation API

The *googleVis* package allows creation of interactive charts that can be embedded into web pages:

- The output of a *googleVis* function is html code that contains the data and references to JavaScript functions hosted by Google. The data is not uploaded to Google.
- An internal R HTTP server displays the output locally. A browser with Flash and Internet connection is required. The Chrome browser may give best results.

Here, demonstrate Hans Rosling style **Motion Charts**

Use of *googleVis* to Create 'Motion Charts'

Hans Rosling has used Motion Charts very effectively to wow audiences – see, eg, his TED talk:

`http://www.youtube.com/watch?v=ezVk1ahRF78`

What follows is largely based on:

`http://lamages.blogspot.com/2011/09/accessing-and-plotting-world-bank-data.html`

(Markus Gesmann, 24 September 2011)

Code Steps to Display World Bank Data

1. Access database, download data in Javascript Object Notation (JSON) format;
2. Process data as required for *googleVis*
3. Create and plot the graphics object.
   [Create using `gvisMotionChart()`.]

With *googleVis* (and dependencies) installed , type the following to see a motion chart for World Bank data:

```
library(googleVis)
demo(WorldBank)  # This may take a while, especially
                 # with a slowish internet connection
## Note other available demos
demo(package='googleVis')
```

Note – Accessing World Bank data

This uses functions (see the code for the demo) such as:

```
getWorldBankCountries <- function(){
    require(RJSONIO)
    urlbase <- "http://api.worldbank.org/"
    countryInfo  <- paste(urlbase,
                          countries?per_page=12000&format=json",
                          sep="")
  wbCountries <-
    fromJSON(countryInfo)
  wbCountries <- data.frame(t(sapply(wbCountries[[2]], unlist)))
  wbCountries$longitude <- as.numeric(wbCountries$longitude)
  wbCountries$latitude <- as.numeric(wbCountries$latitude)
  levels(wbCountries$region.value) <-
                       gsub(" \\(all income levels\\)",
                            "", levels(wbCountries$region.value))
  return(wbCountries)
}
```

Finally, put data together into a data frame with name subData

Creation of Plot (Javascript Code)

The plot is created thus:

```
M <- gvisMotionChart(subData, idvar="country.name",
                     timevar="year",
                     options=list(width=700, height=600))
## Display the chart your browser
plot(M)
## Now click on the chart ID (to left, below plot) to see
## Javascript code that can be incorporated into a web page.
```