R in the Cloud with RGoogleDocs and rApache
Easy, fast deployment of R-based Web interfaces.

Timothée Carayol

Qubit Group

12 August 2011
Situations where the development of a simple interface with R (for stats and/or visualization) is desirable abound.

E.g. make R capabilities available to R-illiterate friends/colleagues/students.
Problem: developing a decent-looking and functional Web interface is complicated; typically need to learn at least one other language, possibly several (html, javascript, php..)

Discouraging prospect for many of us.

What if the cost of creating such an interface could be lower than usually perceived?
Introduction

- RGoogleDocs provides a great way to input a configuration; and rApache a great way to query R and visualize its output.
- R can therefore be self-sufficient to build very functional, convenient interfaces, with possibly many user-inputted settings.
rApache (rapache.net) allows an Apache server to use R to handle queries.

R is triggered when a user lands on the URL; R generates the HTML code, including result tables and plots.

documentation

This manual covers installation, configuration, and various uses of the rApache software distribution and supporting packages. This document is intended to contain the most up-to-date information about rApache. Comments, suggestions should be forwarded to the maintainer.

1. Introduction

rApache is a project supporting web application development using the R statistical language and environment and the Apache web server. The current software distribution runs on UNIX/Linux and Mac OS X operating systems. Apache servers with threaded Multi-Processing Modules are now supported, but the Apache Prefork Multi-Processing Module is still recommended (refer to the Multi-Processing Modules chapter from Apache for more about this).

The rApache software distribution provides the Apache module named mod_R that embeds the R interpreter inside the web server to allow access to R via the URL. This module can be configured with several options.
A simple Web interface based on rApache is easy to setup; but input of queries can be somewhat inconvenient, e.g. saving and modifying previous queries.

Advanced web programming skills and large amounts of coding are a solution – but maintained hypothesis is that our statistician is short on time and programming skills.
Google Docs offers online, collaborative spreadsheet software, hosted in the cloud. Accessible from any terminal with a web browser.

The R package RGoogleDocs (by Duncan Temple Lang, on Omegahat) allows R to instantly fetch or push data from / to a Google Spreadsheet, allowing Google Docs to become part of an R workflow.

The RGoogleDocs package

This package is an example of using the RCurl and XML packages written to test RCurl and also to illustrate how to use these tools.
Proposed Interface

- **Idea:** let the input side of the Web application be handled by RGoogleDocs! Set up a spreadsheet and turn it into a configuration file for rApache to fetch each time a user visits the URL.

- **User interface:** click on a cell in GoogleDocs, change its content, refresh the page and see the results change.

- **Alternatively:** a GoogleDocs spreadsheet can contain dynamic data, inputted collaboratively (or via some other software). rApache URL then guarantees up-to-date visualization of results.
Advantages:

- Once you are comfortable with this workflow, you can set up a Web interface with R in a few dozen minutes.
- Keep a history of your queries; possibly have multiple configurations ready to be fired up (by using multiple sheets in the same document).
Drawbacks

- Somewhat inefficient; Apache reluctant to give RAM back, and R is RAM hungry. Some hacks with multicore’s parallel() are possible.
- Many users using the interface at the same time can be problematic – requires some adjustments in the workflow. (e.g. user-specific configuration spreadsheet).
- Not very scalable.
Example: interface for K-means clustering

- Clustering high dimensional data requires many choices – which variables to include? What weight? What transformations? (e.g. taking the logs..) How many clusters? Sample size?
- Objective: allow analysts with no knowledge of R to perform K-means clustering on visitors to a website (based on number of past visits, products purchased etc.).
A google document to configure clustering.

gdoc.param <- GetGDocClustData(
  "your.account@gmail.com",
  "password",
  spreadsheetname
)
- Different reports on the characteristics of the clusters.
- Also – links to the pictures and data that generated them.
- Good practice: also report the gdoc configuration that generated the clusters.
To my knowledge, this is the fastest, easiest way to develop a web interface using R.

Takes advantage of R’s great abilities, along with those of Apache and Google Docs.

Other possible use cases:

- In the classroom – to demonstrate how an algorithm responds to different configuration settings.
- In a commercial environment – for a proof of concept on a prototype, before any further UI / UX considerations.
- For weekend projects involving data visualization responding to some sort of input.
To my knowledge, this is the fastest, easiest way to develop a web interface using R.

Takes advantage of R’s great abilities, along with those of Apache and Google Docs.

Other possible use cases:

- In the classroom – to demonstrate how an algorithm responds to different configuration settings.
- In a commercial environment – for a proof of concept on a prototype, before any further UI / UX considerations.
- For weekend projects involving data visualization responding to some sort of input.

Thanks go to Google Inc., Duncan Temple Lang, the Apache foundation and Jeffrey Horner for making this possible.