Why R Belongs in the Clouds

In this talk I'll make the case for why R ought to live on the web (or in the Cloud), showcase a web dashboard that I built using R, and finally highlight some nuts and bolts of how it was done.

_____

April 8, 2009 6:30pm
Bay Area Users R Users Group
http://www.meetup.com/R-Users/

Building Web Dashboards with R
 Michael E. Driscoll, Ph.D.
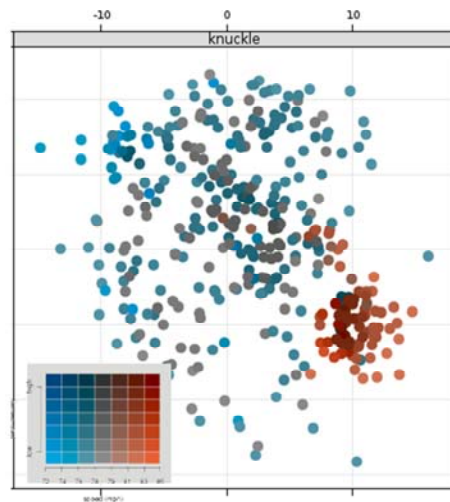 mike@dataspora.com

Follow-on talk by John J. Oram, Ph.D.

(Existing R enthusiasts kindly skip this slide).

What Makes R Special?

While several languages can do what R does – R is unique for combining three capabilities in a single, open source tool:

I.   **data manipulation**: this means connecting to databases like MySQL or Oracle, to slice and dice through large, multivariate data sets. I've programmed in many languages, but I've rarely found a better tool for indexing into data.

II.   **statistical analysis**: this is R's *raison d'être,* and it's most powerful aspect. R can perform:

 *hypothesis testing*: Bayesian analysis or chi-squared tests
  model fitting: general linear models, linear mixed-effects models, least angle regression approaches
  clustering: k-means and others
  machine learning: recursive partitioning, neural networks, support vector machines

Common statistics functions – basic probability distributions – are part of the core language. Less common techniques can be found on CRAN http://**cran.r**-project.org.

III.   **data visualization** – As a visualization nut, this is perhaps my favorite part of R. Visualization is most useful not in testing hypotheses, but in formulating them. nothing helps one understand data than by **looking at it.**
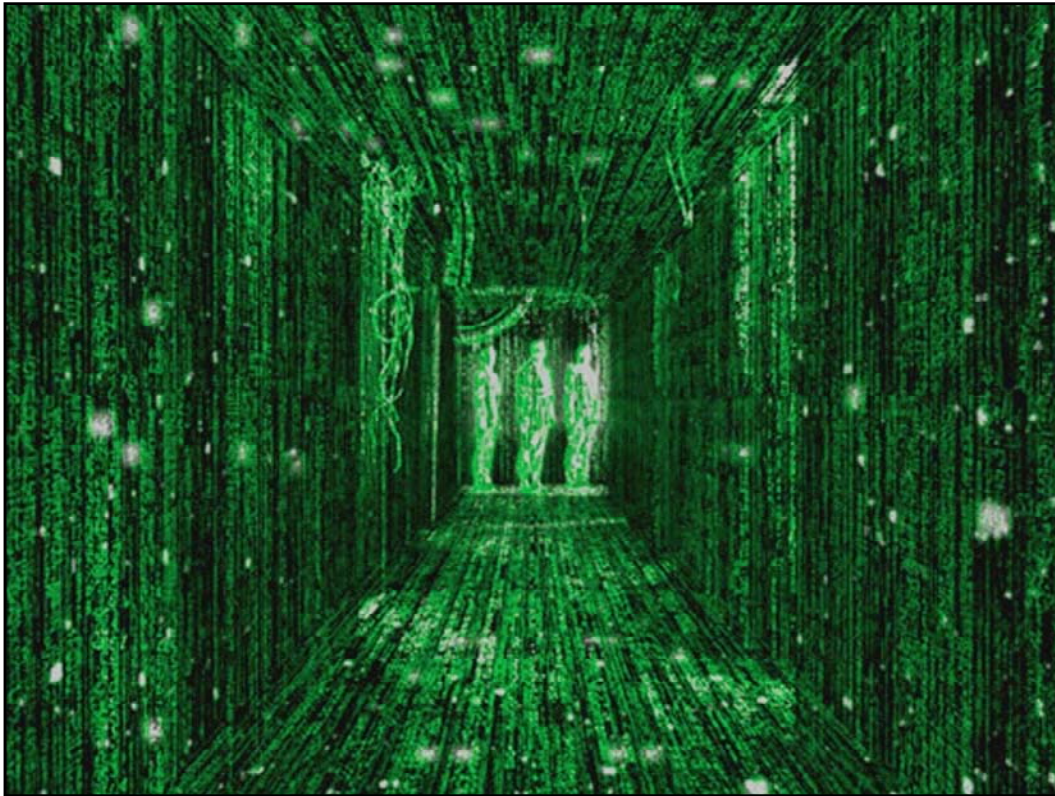
Why the World of Data Needs R

First, I'd like to provide some motivation for why it's critical that R's power be integrated into web applications.

The Industrial Revolution of Data. The world is witnessing what Joe Hellerstein has dubbed "The Industrial Revolution of Data": where machines, not people, are the primary generators of data. In any given minute, databases somewhere are tracking mouse clicks on web sites, point of sale purchases, rider swipes through subway turnstyles, physician prescriptions, digital video recorder rewinds, and the location of every GPS-enabled car and phone on the planet.

The world is streaming billions of data points per minute. This is Big Data. Too big to store on our desktops, or even on our LANs. We need tools to make sense of it.

Big Data Scientists. Facebook calls their data analysts 'data scientists'. I like this term, because it conveys that data analysis involves applying the scientific method: formulating and testing hypotheses. To test hypotheses we need statistics. To do statistics, we use R. And to use R effectively on Big Data, R should live where the data lives: on the web, in the Cloud.

"The sexy job in the next ten years will be statisticians."

– Hal Varian, Chief Economist, Google

"The ability to take data—to be able to understand it, to process it, to extract value from it, to visualize it, to communicate it—that's going to be a hugely important skill."

Interview with Hal Varian – from the McKinsey Quarterly magazine (2009)
http://www.mckinseyquarterly.com/Strategy/Innovation/Hal_Varian_on_how_the_Web_challenges_managers_2286
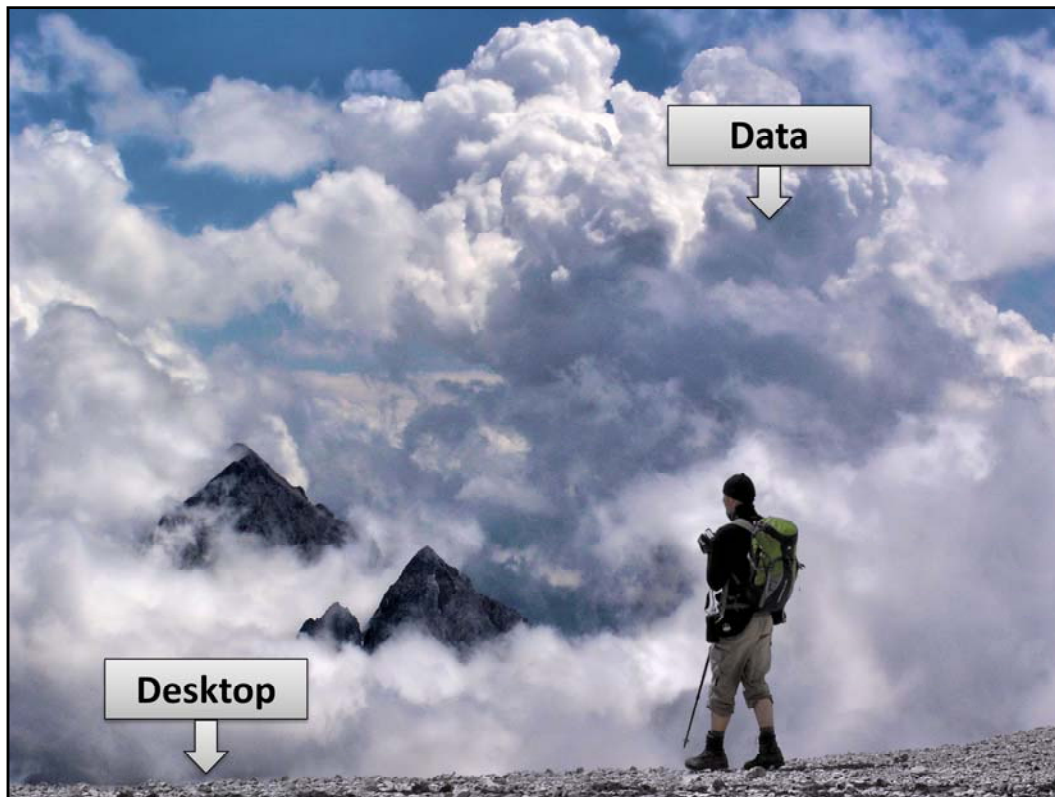
Moving Analytics from the Desktop to the Cloud

The cloud is an enormous, amorphous place with more data than you could possibly conceive.

The 'cloud' is just a useful abstraction, like 'the web.' What's new is the scale and scope: Amazon has opened up their infrastructure, allowing – in effect – any one to rent power on their compute farm, dubbed EC2. Google has done the same, albeit allowing access at a higher level with Google App Engine.

•**I. Data is heavy, software is light**

Data is growing in size and scope, it is getting heavy. Analysis software should "live" near its target data, because of network latencies and storage requirements. For enormous data sets, it's the fastest way to move data is not the fiber, but FedEx – not the internet, but sneaker net (as the late Jim Gray termed it). The key is to move data as little as possible.

•**II. Analytics can't (and shouldn't) be done on the desktop**

In an age of Linked Big Data (c.f. http://blog.ted.com/2009/03/tim_berners_lee_web.php ,
http://dataspora.com/blog/tipping-points-and-big-data/) it's not feasible nor desirable to store terabytes of data on the desktop. Not every firm has hit this breaking point, but many are approaching it.

•**III. CPU power becomes a utility** – like electricity or water, pay as you go. It means that (in theory) web applications – like electrical appliances – can plug into any CPU power grid. And those grids, in turn, have vastly fewer idle cycles. It democratizes access to CPU power and drives the price of commodity CPU computing ever lower.
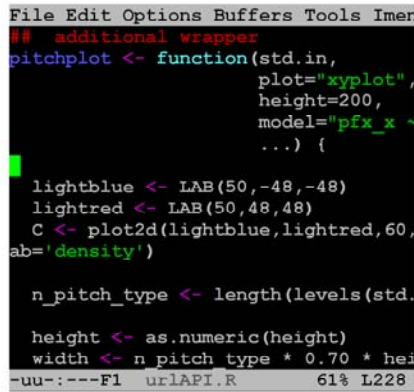
With the cloud, no organization should maintain a cluster that runs at less than 50% capacity (this is effectively every academic research organization in America).

**A web application can wrap complexity in a simple, clickable interface.**

I prefer clicking to coding.

In *Founders at Work,* Evan Williams described how Blogger started as an internal project: he created a web tool for posting thoughts to a web page, and found it easier than hand-coding HTML. People – even coders – prefer clicking to coding.

The problem is that right now, too many of us are repeating the steps in data analysis.
• We struggle to extract data from some online source.
• We fight to format it into a shape we can work with, and import it into our tool of choice.
• We labor to achieve elegant looking graphics.

Rather than forcing us to repeat ourselves, we can build a simple web app that captures the core functions we seek.
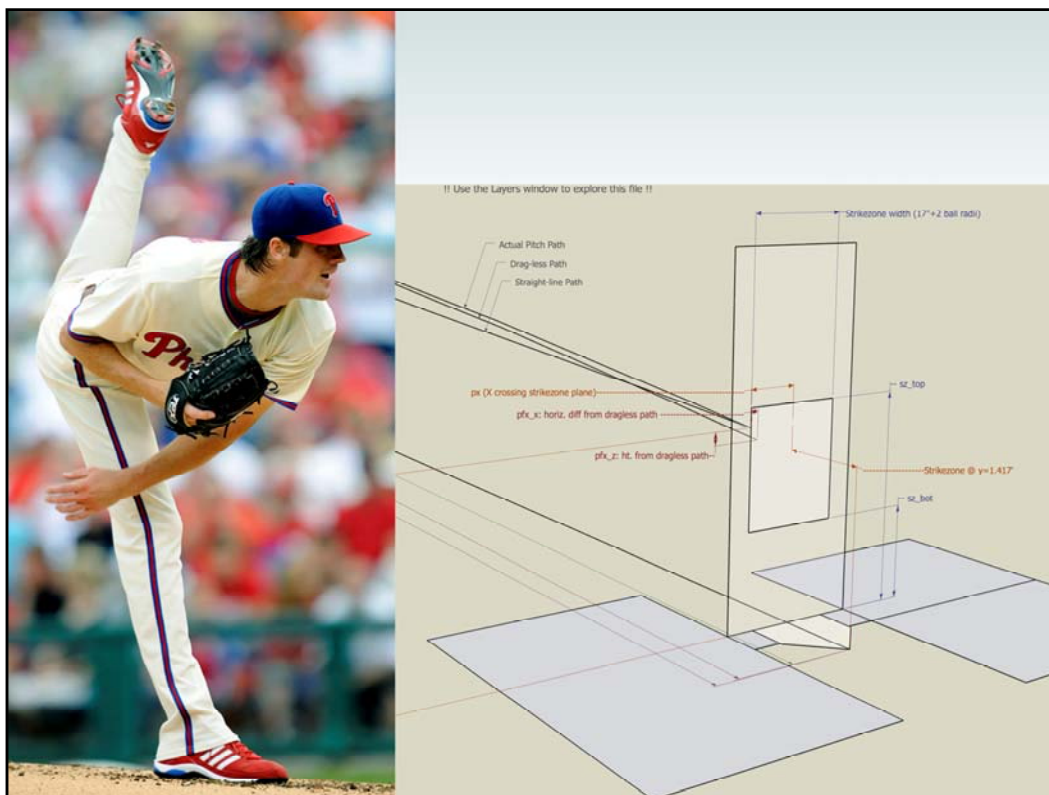
R Web Bashboard: Visualizing MLB Pitchers

On the left is Cole Hamels, who (I'm told) took the Phillies to victory the 2008 World Series.

On the right is a diagram of the PitchFX system, which in the 2008 season, used special cameras to record the speed, position, and many other attributes – as seen in the diagram – of over one million pitches thrown.

What's remarkable is that this data is made publicly available as XML by Major League Baseball.   For example:

  http://gd2.mlb.com/components/game/mlb/year_2008/

We can grab data from here, pull it into R, and crunch it. But why should everyone have to do this? What we'd really like to do is build a dashboard that let's us explore the data, without having to do the painful data munging steps.

Presenting the PitchFX Visualizer…

IV. A Tale of Two Pitchers
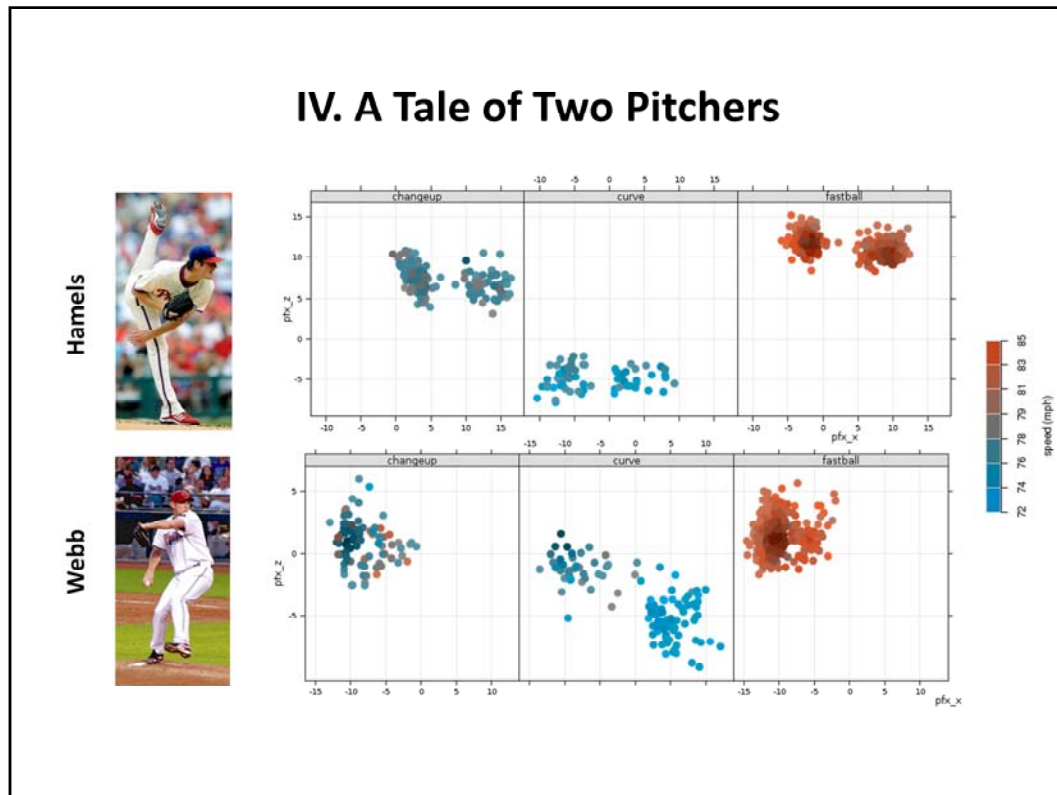
An R Web Dashboard: Visualizing MLB Pitchers

To demonstrate that this data visualization can be useful, let's take a look at two pitchers' data from 2008: the Phillies' Cole Hamels (top) and the Arizona Diamondbacks' Brandon Webb (bottom).

This visualization shows how Hamels and Webb use different approaches to beat batters.

•Cole Hamels is a finesse pitcher, he is able to paint corners; he generally throws his fastballs and change-ups to different places. A batter may know it's a fastball, but not where it will end up.
•Brandon Webb's pitches his fastballs and change-ups to the same location, he varies speed: a batter knows where it will end up, just not how fast.

This example also highlights a method for visualizing six dimensions of data in a single graphic, namely:
 1. and 2. x and y location of the pitch
 3. pitch type
 4. pitch speed
 5. pitch density (lots of pitches make darker luminosity with out changing hue)
 6. pitcher (Cole or Hamels)

http://labs.dataspora.com/gameday
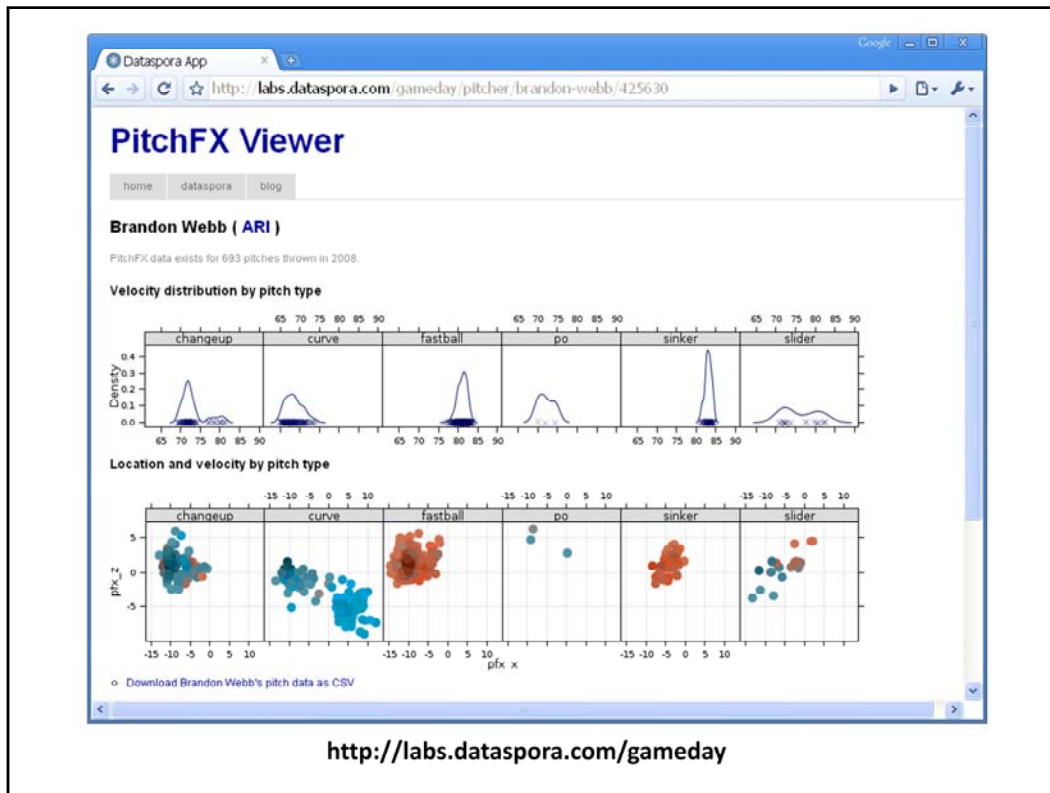
Web Dashboards are Universally Accessible, Live Data Applications

We've realized this in a web dashboard, running at http://labs.dataspora.com/gameday.

More than a toy, putting not just data – but analysis – on the web is an important step for several reasons:

• because R is open source: I can embed R inside the web server without licensing restrictions
• data and the processing both live on the server – important when your data set is large
• when the data changes, the dashboard updates
• accessible from any platform with a web browser, no software installation needed

This dashboard is live at:

 http://labs.dataspora.com/gameday

9
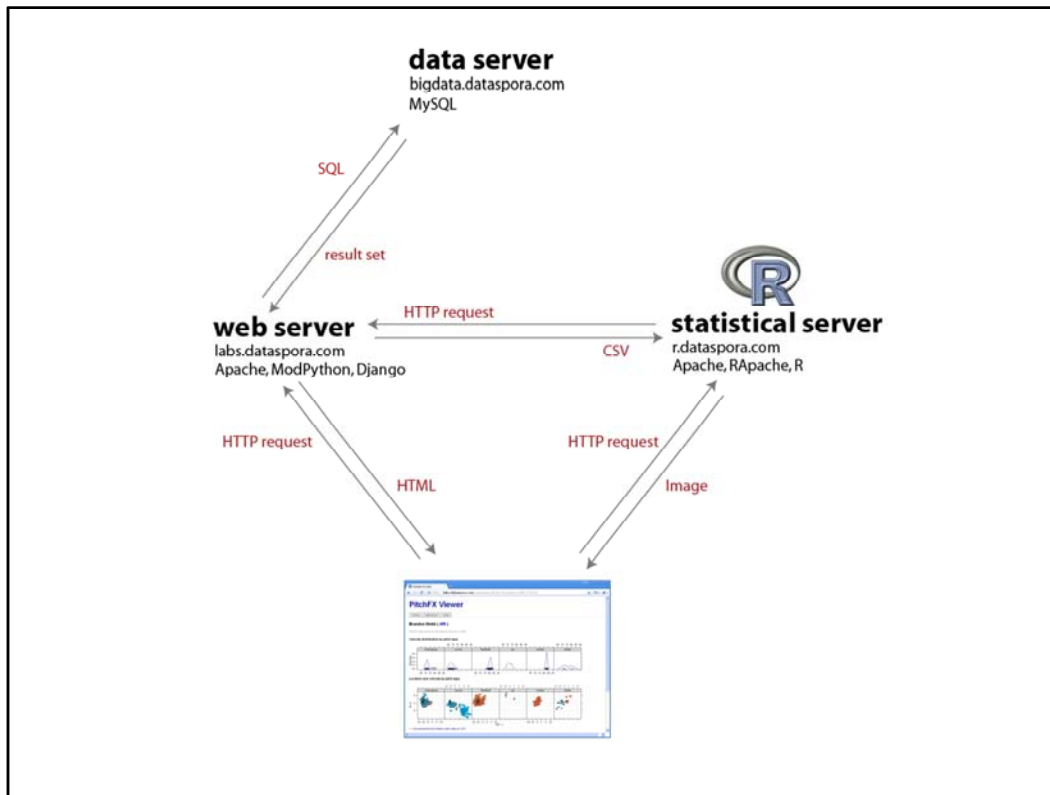
Embedding R into a Web-based Architecture

This is the underlying setup of my system. Putting R in this environment immediately conveys the many benefits of a web architecture, because it is:

 * **Stateless/Scalable** – URL requests can be distributed across one or many servers

 * **Cacheable** - common requests made to the R server can be cached by Apache

 * **Secure** - we can piggyback on existing HTTPS architecture for analysis of sensitive data

rapache: Embedding R within the Apache Server

Our tool of choice is rapache, developed by Jeff Horner at Vanderbilt University.

 http://biostat.mc.vanderbilt.edu/rapache/

Configuring rapache (I)

Ubuntu 8.04, R 2.6.2, Apache 2.2.8-1   **rapache**

1. Download & Install Rapache tarball

```
wget http://biostat.mc.van.../rapache-1.1.7.tar.gz
./configure
make;
make install
```

Configuring rapache #1. Downloading and Install the rapache Tarball

rapache is a standalone package not available at CRAN package, but source is downloadable at:

http://biostat.mc.vanderbilt.edu/rapache/downloads.html

I'll step through how I got it working on my system, but your mileage may vary.

My setup is a Linux box, Ubuntu 8.04, server edition, running a slightly older version of R 2.6.2 with Apache 2.2.8.

The first step is to simply download the tarball and install it (and yes, you must have root/sysadmin privileges for any of this).

Configuring rapache (II)

2. Edit http.conf or apache2.conf   **rapache**

```
LoadModule R_module /usr/lib/apache2/modules/mod_R.so

<Location "/R">
  ROutputErrors
  SetHandler r-script
  RHandler sys.source
  REvalOnStartup "library(Cairo); library(lattice)"
</Location>
```

Configuring rapache #2. Modify your Apache configuration

This requires adding a few lines to your http.conf file, first to load the R module mod_R.so, then to indicate where R scripts will be located, and how they will be handled.

The directive <Location "/R"> in my case points to the directory '/var/www/R' on my webserver.

"RHandler sys.source" defines the function 'sys.source' to be called on files in this directory.

The "REvalOnStartup…" directive allows you to pre-load commonly called libraries when Apache starts.

13

Configuring rapache (III)

3. Write helloworld.R

```
setContentType("text/html")
source("/var/www/Rlib/urlAPI.R")
png("/var/www/hello.png")
plot(sample(100,100),col=1:8,pch=19)
dev.off()
cat("<html>")
cat("<body>")
cat("<h1>hello world</h1>")
cat('<img src="../hello.png"')
cat("</body>")
cat("</html>")
```

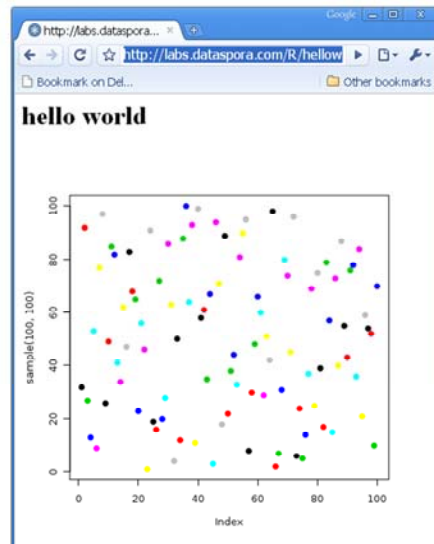**Configuring rapache #3. Write your Hello World script**

In this case, the helloworld.R script would be deposited at /var/www/R/helloworld.R

In this simple example, we write a PNG graphic to a readable location on disk, and then return HTML that references this location.

Configuring rapache #4. Browse to Your Script's Location and Voila

Enjoy the colorful fruits of your labor. Naturally this is just scratching the surface of what **rapache** can do.

An alternative approach to printing HTML directly, is to use a templating system, similar to PHP.

This is available via the R package brew (also developed by Jeffrey Horner), downloadable on CRAN and at:

http://www.rforge.net/brew/

Going Further with rapache

sendBin()

```
setContentType("png")
size = file.info(infile)$size
sendBin(readBin(infile,'raw',n=size)
```

GET, POST and other variables

```
cat("Hello", GET$name)
```

Moving Beyond Toy Examples

Using sendBin() to deliver binary content

The sendBin function allows you to directly send other content types – besides txt/html – directly from R. This is useful for returning small images, such as sparkgraphs, from an R server. This is the approach we take with our Pitch Visualizer App, where URLs return actual images such as:

http://labs.dataspora.com/R/urls.R?data=gameday.pitcher.493137&pitchplot

return dynamically generated PNG files.

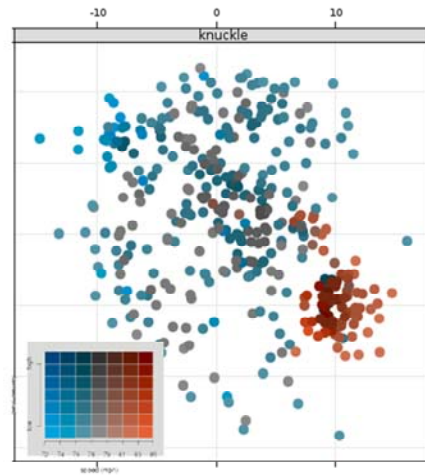**Fetching GET, POST, and other variables**

In RApache, elements in the HTTP query string are accessible via a global 'GET' variable, which is encoded as a list of key-value pairs.

More detailed information on the range of variables that rapache gives can be found at:

http://biostat.mc.vanderbilt.edu/rapache/manual.html#rapache_Variables

Making Beautiful Colors with the Colorspace package

Ross Ihaka's Colorspace package provides access to useful colorspaces beyond RGB, like LAB and HSV. These colorspaces are preferred by artists and designers for their more intuitive properties.
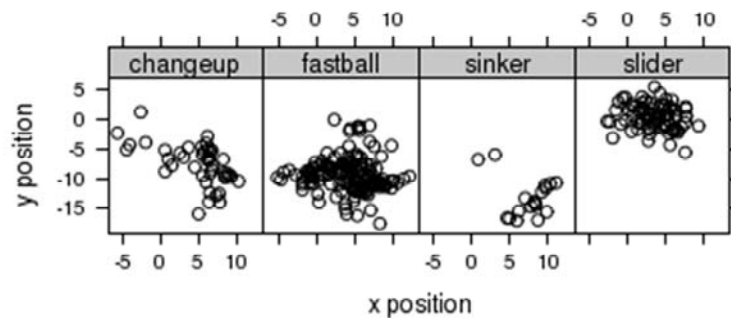
This is the package I used to design the palettes in the PitchFX dashboard.

I've posted further thoughts on using color in data visualizations at:
 http://dataspora.com/blog/how-to-color-multivariate-data/

Panel Plots with R Lattice Package

```
library("Lattice")
xyplot(x ~ y | pitch_type,
data = gameday)
```

Render Statistical Models into Visualizations with the Lattice Package

One of the most powerful visualization tools available is Deepayan Sarkar's Lattice package.

Lattice translates R's model syntax (such as 'x ~ y') into a visual representation.

It is available on CRAN, with great code examples here. http://lmdvr.r-forge.r-project.org/figures/figures.html

Lattice is an R implementation of William Cleveland's Trellis graphics system, developed at Bell Labs.
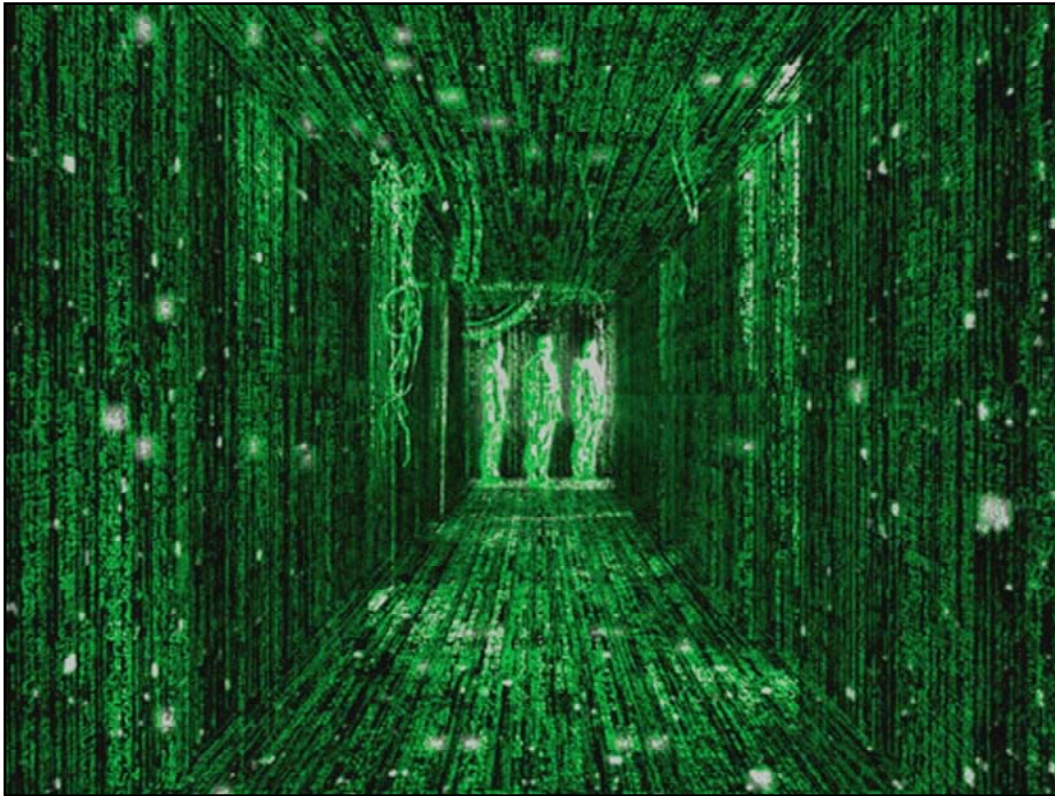
Web Dashboards with R – Bringing Big Data to Many Eyes

To conclude: we live in a world that is exploding with data. But for this data to be useful, it needs to be interpretable. R is a powerful tool for shaping, analyzing, and visualizing data – but it has been limited by two factors:

(i)   confined within a single desktop's storage and computation limits
(ii) accessible only by coders, not clickers

Placing R on the web removes these limitations, by (i) moving it to the where the interesting data lives and cloud computing resources are available (ii) allowing it to be encapsulated into web applications, where anyone who can operate a mouse can explore data.

The most accessible route for our brains to grapple with large data is through our eyes: this is the last step that data takes before a decision is made. Web dashboards deliver data to the eyes of decision-makers, and R is an ideal tool for building these dashboards.

For more information, contact:

Michael E. Driscoll, Ph.D.
mike@dataspora.com
http://www.dataspora.com/blog