The pgfSweave Package

Cameron Bracken and Charlie Sharpsteen

January 13, 2009

The pgfSweave provides capabilities for "caching" graphics generated with Sweave. This document highlights the features and usage of pgfSweave. pgfSweave provides a new driver for Sweave, pgfSweaveDriver and new chunk options pgf and external on top of the cache option provided by cacheSweave

1 Motivation and Background

Sweave is a tool for generating "reproducible" documents by embedding R or S "code chunks" directly into a LaTeX document. Two main drawbacks to this approach are:

1. Code chunks with lengthy computations and plotting commands are executed every time a document is compiled
2. Consistency in style (font, point size) in automatically generated graphics is difficult to achieve.

The cacheSweave package addresses the first issue of lengthy computations by storing the result of computations in a filehash databases. This provides significant speedup of certain computations, namely those which create objects in the global environment. Unfortunately most plotting commands do not create objects which can be cached. This is the first issue addressed by pgfSweave. So called "caching" of plots is achieved with the help of two tools: the TeX package pgf (http://sourceforge.net/projects/pgf/) and the command line utility eps2pgf (http://sourceforge.net/projects/eps2pgf/).

The pgf package provides the ability to "externalize graphics." MORE ABOUT THE EXTERNALIZATION PROCESS AND EPS2PGF.

This package is built directly upon the cacheSweave and therefore also Sweave.

2 Usage

We assume a familiarity with the usage Sweave, for more information see the Sweave manual (http://www.stat.uni-muenchen.de/~leisch/Sweave/Sweave-manual.pdf). We first suggest setting a subdirectory for the cached code chunks with a code chunk like:

```
<<setup,echo=F>>=
setCacheDir("cache")
@
```

We also suggest using a separate subdirectory[1] for your figures with the Sweave option prefix.string like:

```
\SweaveOpts{prefix.string=figs/fig}
```

[1] make sure to create the directory first!

At this point we will provide a complete example. The example from the `Sweave` manual is used to highlight the differences. The two frame below show the input Sweave file `example.Rnw` and the resulting tex file `example.tex`.

pgfSweave-example-Rnw.in

```
\documentclass{article}

\usepackage{pgf}
\pgfrealjobname{example}
\title{pgfSweave Example}
\author{Cameron Bracken}

\begin{document}
<<setup,echo=F>>=
setCacheDir("cache")
@
\maketitle
In this example we embed parts of the examples from the \texttt{kruskal.test}
help page into a \LaTeX{} document:

<<data,cache=T,pgf=T,external=T>>=
data(airquality)
library
kruskal.test(Ozone ~ Month, data = airquality)
@
which shows that the location parameter of the Ozone distribution varies
significantly from month to month. Finally we include a boxplot of the data:

\begin{center}
<<plot,fig=TRUE,echo=FALSE,pgf=T,external=T>>=
boxplot(Ozone ~ Month, data = airquality)
@
\end{center}

\end{document}
```

pgfSweave-example-Rnw.in

On the input file run:

```
R> library(pgfSweave)
R> pgfSweave('example.Rnw',pdf=T)
```

And we get:

pgfSweave-example-tex.in

```
\documentclass{article}

\usepackage{pgf}
\pgfrealjobname{example}
\title{pgfSweave Example}
\author{Cameron Bracken}

\usepackage{/Library/Frameworks/R.framework/Resources/share/texmf/Sweave}
\begin{document}
\maketitle
```

```
In this example we embed parts of the examples from the \texttt{kruskal.test}
help page into a \LaTeX{} document:

\begin{Schunk}
\begin{Sinput}
> data(airquality)
> library
> kruskal.test(Ozone ~ Month, data = airquality)
\end{Sinput}
\end{Schunk}
which shows that the location parameter of the Ozone distribution varies
significantly from month to month. Finally we include a boxplot of the data:

\begin{center}
\beginpgfgraphicnamed{pgfSweave-example-plot}
\input{pgfSweave-example-plot.pgf}
\endpgfgraphicnamed
\end{center}

\end{document}
```
pgfSweave-example-tex.in

```
> hist(a)
```

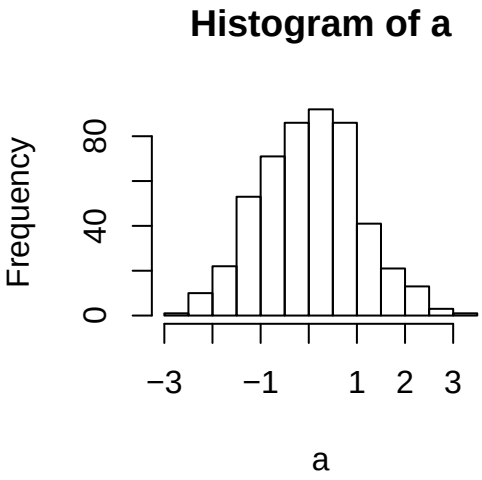

Figure 1: This is normal `Sweave`.

```
> hist(a)
```

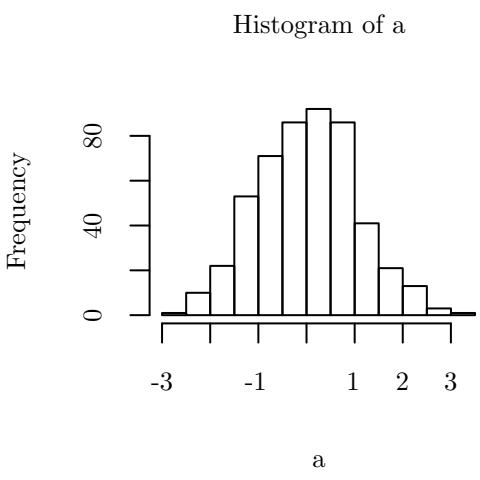

Figure 2: This is from `pgfSweave`.

In figure **??** Notice the inconsistency in font and size between the default R output and the default LaTeX output. Fonts and font sizes can be changed from R but it is hard to be precise. What if you decide to change the font and and point size of your entire document? In figure **??** The text is consistent with the rest of the document.

The example below illustrates some of the power of `pgfSweave`. LaTeX code can be directly input into captions. This sort of thing is already available in R but again consistency in font and text size is difficult to achieve.

```
> plot(a, b, xlab = "", ylab = "")
> title(xlab = "$\\alpha\\beta\\gamma\\delta\\epsilon\\Re\\ell\\hbar\\odot\\otimes\\oplus$")
> title(ylab = "\\color{red}{\\scshape Red Label in Small Caps}")
> title(main = "{\\large This is a plot of $\\displaystyle\\int_a^b \\frac{x}{y}dx$}")
> abline(fit)
```

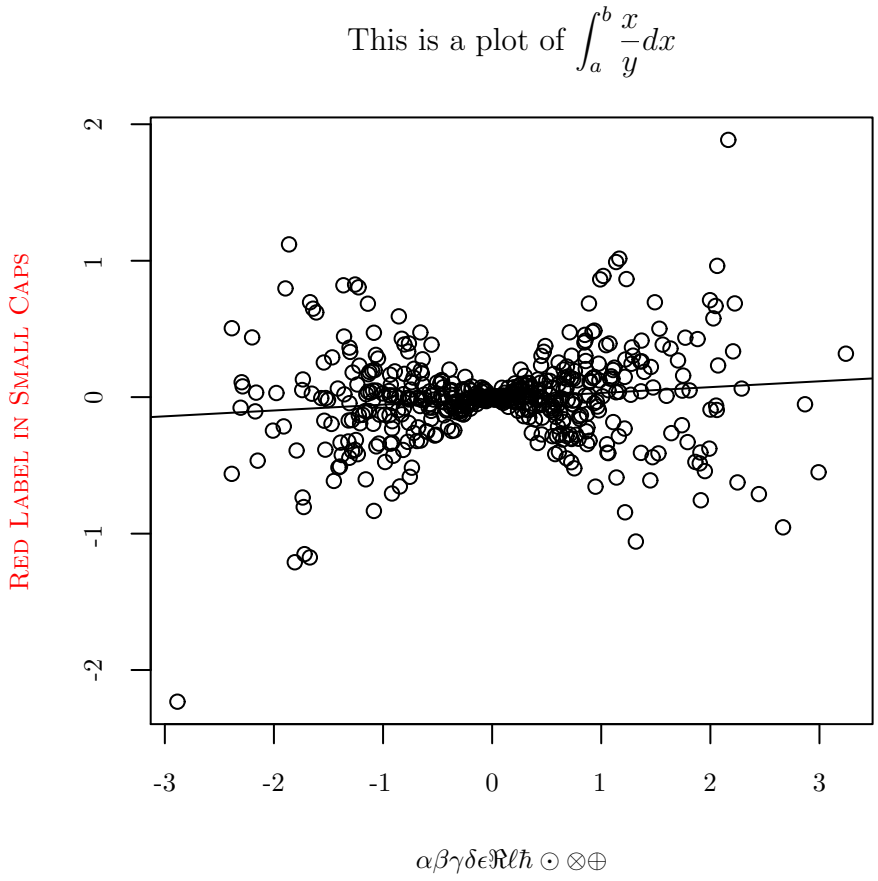

Figure 3: Large size plot but still consistant. Also LaTeX can be put directly into the titles of the plot which matches the style of your paper. All \'s must be escaped using this method.

Every part of `Sweave` and `cacheSweave` work the same but do not cache explicit print statements or they will not show up on a second compile.

3 Frequently Asked Questions

Can `pgfSweave` be run from the command line?

Sure! Use:

```
R CMD pgfSweave <myfile>.Rnw
```

OR use this shell script:

```
#!/usr/bin/Rscript

library(pgfSweave)
args <- commandArgs(T)
pgfSweave(args[1],pdf=TRUE,quiet=FALSE)
```

Just save the script above as `pgfsweave` then run

```
$> pgfsweave <yourfile>.Rnw
```