The **pgfSweave** Package

Cameron Bracken and Charlie Sharpsteen

January 25, 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**. This package is built directly upon **cacheSweave** and therefore also **Sweave**.

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**[1] and the command line utility **eps2pgf**[2].

When we refer to the "caching" of an graphic we mean that if the code chunk which generated the graphic is unchanged, an image is read from a file rather than regenerated from the code. The TeX package **pgf**[3] provides the ability to "externalize graphics." The externalization chapter in the **pgf/Ti*k*Z** manual is extremely well written, so please look there for more information.

2 Usage

We assume a familiarity with the usage of **Sweave**, for more information see the **Sweave** manual[4]. This section will explain the usage of the `pgf` and `external` options and then provide a complete example.

[1] `http://sourceforge.net/projects/pgf/`
[2] `http://sourceforge.net/projects/eps2pgf/`
[3] Latest CVS available at `http://sourceforge.net/cvs/?group_id=142562`
[4] `http://www.stat.uni-muenchen.de/~leisch/Sweave/Sweave-manual.pdf`

2.1 The pgf option

The first new code chunk option **pgf**, acts the same as the **pdf** or **eps** options but instead of resulting in an \includegraphics{} statement the result is an \input{} statement. For example the text and Consider the following code:

Input:

```
\begin{figure}[ht]
 <<pgf-option,fig=T,pgf=T,echo=F>>=
     x <- rnorm(100)
     plot(x)
 @
\caption{caption}
\label{fig:pgf-option}
\end{figure}
```

Output:

```
\begin{figure}[ht]
\input{pgf-option.pgf}
\caption{caption}
\label{fig:pgf-option}
\end{figure}
```

The .pgf file is generated with the **eps2pgf** utility. The postscript graphics device is used first to generate a .eps file. Then the command

```
$ java -jar /path/to/eps2pgf.jar -m directcopy graphic.eps
```

is run on every code chunk that has fig=TRUE and pgf=TRUE. When using **pgfSweave** the pgf option is set to TRUE by default.

2.2 The external option

Input:

```
\begin{figure}[ht]
 <<external,fig=T,pgf=T,external=T,echo=F>>=
     x <- rnorm(100)
     plot(x)
 @
\caption{caption}
\label{fig:pgf-option}
\end{figure}
```

Output:

```
\begin{figure}[ht]
\beginpgfgraphicnamed{external}
\input{external.pgf}
\endpgfgraphicnamed
\caption{caption}
\label{fig:external}
\end{figure}
```

2.3 A complete example

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

3 Consistency in style between graphics and text

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.

```
> hist(a)
```

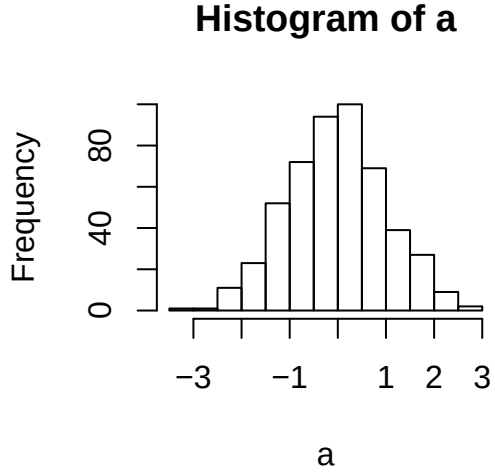

Figure 1: This is normal **Sweave**.

```
> hist(a)
```

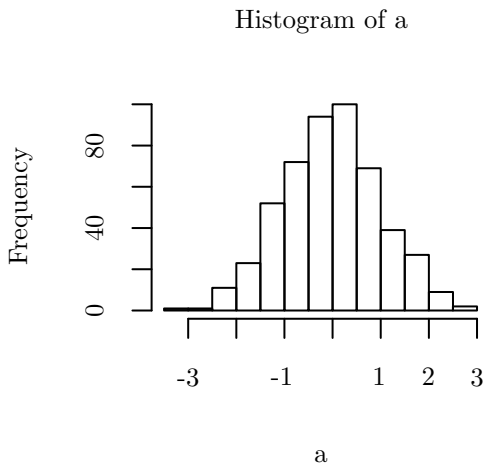

Figure 2: This is from **pgfSweave**.

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 = "$\\alpha\\beta\\gamma\\delta\\epsilon\\Re \\longrightarrow\\ell\\hbar$")
> title(ylab = "{\\color{green!40!blue}\\scshape \\Large Teal Label in Small Caps}")
> title(main = "{\\large This is a plot of $\\displaystyle\\int_a^b \\frac{x}{y}dx$}")
> abline(fit)
```

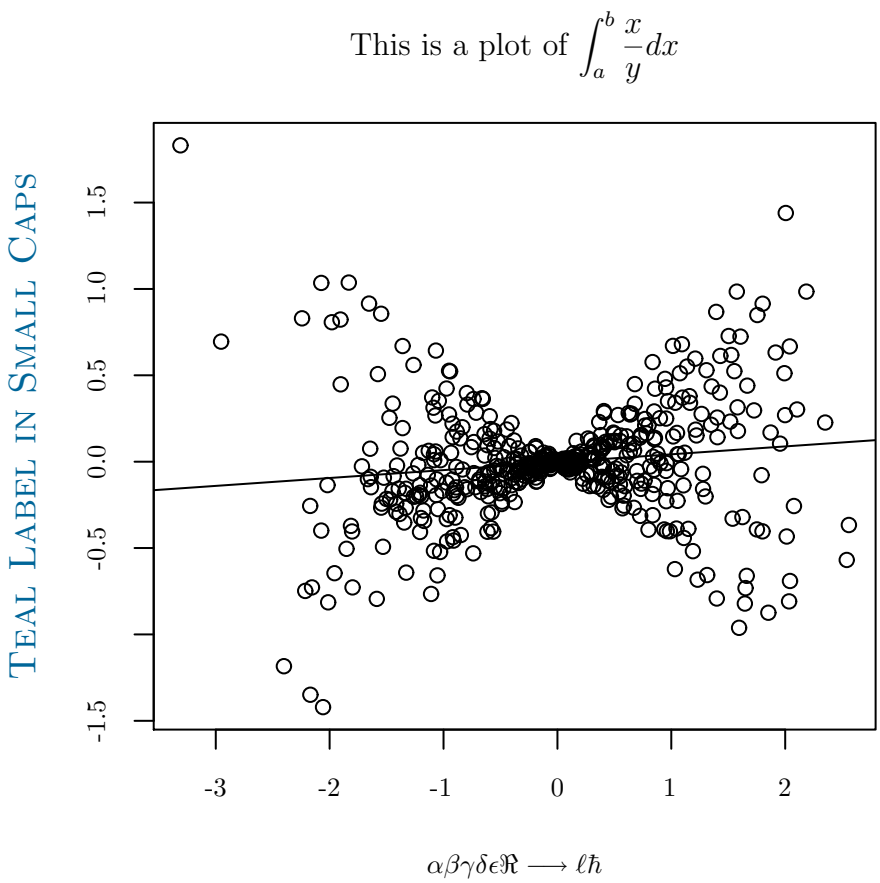

Figure 3: Large size plot but still consistant. Also LaTeX can be put directly into the titles of the plot which matches the style of your paper. All \'s must be escaped using this method.

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.

4 Frequently Asked Questions

Can pgfSweave be run from the command line?

Sure! Use: (HOPEFULLY THIS WILL WORK IT DOES NOT YET!)

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

OR use the shell script provided in the **pgfSweave** package source in the `exec/` directory Save the script somewhere in your `PATH` and run

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

How do I set subdirectories for figures and caches?

This is strait out of the **Sweave** and **cacheSweave** manuals (nothing new here). For a figures subdirectory [5] use the `prefix.string` option:

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

For a caching subdirectory use a code chunk at the beginning or your document like:

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

Why are the width and height options being ignored?

This is another one from **Sweave**. You must use the `nogin` option in `Sweave.sty` for the width and height parameters to actually affect the size of the image in the document:

```
\usepackage[nogin]{Sweave}
```

latex/pdflatex is not found in R.app (Mac OS X) and [Possibly] R.exe (Windows)

Your latex program is not in the default search path. Put a line such as:

```
Sys.setenv("PATH" = paste(Sys.getenv("PATH"),"/usr/texbin",sep=":"))
```

in your `.Rprofile` file.

[5] make sure to create the directory first!