brewing with rapache

Jeffrey Horner Vanderbilt University Medical Center, Department of Biostatistics

brew is a new R package for generating reports by "brewing" or mixing together the output of R expressions with text. brew's syntax is very similar to PHP, Ruby's erb module, Java Server Pages, and Python's psp module. rapache is a project that embeds R inside the Apache2 web server. Together, they provide a powerful server-side scripting environment for programmers of all levels of expertise.

novice

```
<%
     myname <- ifelse(is.null(GET$name),'World',GET$name)
%>
<html>
<head><title>A Simple Example</title></head>
<body>
<h1>Hello <%=myname%>!</h1></h1>
<% if (myname=='World'){ %>
          <form method="GET">What is your name?
          <input name="name" type="text">
          <input type="submit" value="Say it!">
          </form>
<% } else { %>
Today is: <%=format(Sys.time())%><br>
<a href="/brew/simple.rhtml">Click me!</a>
<% } %>
</body></html>
```

brew executes all R expresssions located between the "<%" and "%>" markup. in general, these expressions will produce no output, as in the assignment to the variable "myname"; but any print() or cat() statements to stdout will display in the browser. also any text that falls outside of the markup is sent to the browser as is.

expressions between the "<%=" and "%>" markup are also evaluated, while this time the results end up replacing the code. in this example, "<%=myname%>" is replaced with the text string "World" if the user provides no input.

control expressions can span multiple "<%" and "%>" markup pairs.

expert #1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head><title>Brewing with Rapache: useR2007 example</title>
<link rel="stylesheet" type="text/css" href="/useR2007.css" />
<script src="/prototype.js" type="text/javascript"></script>
<script src="/useR2007.js" type="text/javascript"></script>
</head>
<body>
<h2>Power and Sample Size Calculations with SPOWER from Hmisc</h2>
<p>The following example<b>[1]</b> demonstrates the flexibility of
spower and
related functions. We simulate a 2-arm (350 subjects/arm) 5-year follow-up
study for wich the control group's survival distribution is Weibull with 1-
year
survival of .95 and 3-year survival of .7. All subjects are followed at least
one year, and patients enter the study with linearly increasing probability
starting with zero. Assume (1) there is no chance of dropin for the first 6
months, then the probability increases linearly up to .15 at 5 years; (2)
there
is a linearly increasing chance of dropout up to .3 at 5 years; and (3) the
treatment has no effect for the first 9 months, then it has a constant effect
(hazard ratio of .75).
</p>
<p>
[1] Alzola CF, Harrell FE:  An Introduction to S and the Hmisc and Design
Libraries .
Freely available electronic book.
</p>
<% brew('/brew/useR2007plot.rhtml') %>
<p>
Characteristics of control and intervention groups with a lag in the treatment effect
and with non-compliance in two directions.
</p>
<form method="get" name="spower" action="#">
1-year Survival Rate: <input type="text" name="p1" value="<%=p1%>">
3-year Survival Rate: <input type="text" name="p2" value="<%=p2%>">
<input type="button" onclick="RePlot()" value="Re-plot">
<input type="button" onclick="ReSimulate()" value="Power Estimate:">
</body></html>
```

brew templates can call other brew templates. this call is to the template displayed in "expert #2" which generates the plot. brew templates can also be used to provide results from AJAX calls. the javascript function "RePlot()" invokes an AJAX request to the template displayed in "expert #2". the result updates the plot in the browser.

expert #2

```
<%
p1 <- ifelse(is.null(GET$p1),.95,as.numeric(GET$p1))
p2 <- ifelse(is.null(GET$p2),.7,as.numeric(GET$p2))
options(hverbose=FALSE,verbose=FALSE)
library(Hmisc)
library(Cairo)
PLOTDIR='/images'
plotname <- paste('plot.',sprintf('%.4f.%.4f',p1,p2),'.png',sep='')
filename <- file.path(PLOTDIR,plotname)
if (!file.exists(filename)){
     CairoPNG(filename=filename,width=600,height=600)
     sink('/dev/null')
     sc <- Weibull2(c(1,3),c(p1,p2))
     rcens <- function(n) 1 + (5-1) * (runif(n) ^ .5)
     f <- Quantile2(sc,
          hratio=function(x) ifelse(x <= .75, 1, .75),
          dropin=function(x) ifelse(x <= .5, 0, .15 * (x-.5)/(5-.5)),
          dropout=function(x) .3*x/5
     )
     par(mfrow=c(2,2))
     plot(f,'all',label.curves=list(keys='lines'))
     dev.off()
     sink()
}
%>
<img src="/images/<%=plotname%>">
```

rapache does all the heavy lifting when it comes to gathering user input. GET, POST, FILES, and COOKIES are list like variables available to the programmer. this example determines whether or not the user provided any values for the form variables p1 and p2. note that all values are provided as character vectors. it is up to the programmer to convert those to the appropriate type.

additional code and data can be loaded with library(), source(), load(), etc.

graphics devices that don't depend on an X server, like Cairo, are perfect for generating plots on the fly. this example shows CairoPNG() storing the plot to disk in a loctation where the web server can then deliver it to the browser upon request.

output is not just limited to full HTML pages. XML, JSON, partial HTML, CSV, even R binary data format can be dynamically generated. the AJAX call to "expert #2" simply outputs the HTML img tag with the plot URL, while the AJAX call to "expert #3" outputs a single numeric value converted to a string.

expert #3

```
<%
p1 <- ifelse(is.null(GET$p1),.95,as.numeric(GET$p1))
p2 <- ifelse(is.null(GET$p2),.7,as.numeric(GET$p2))
options(hverbose=FALSE,verbose=FALSE)
library(Hmisc)
sink('/dev/null')
sc <- Weibull2(c(1,3),c(p1,p2))
f <- Quantile2(sc,
     hratio=function(x) ifelse(x <= .75, 1, .75),
     dropin=function(x) ifelse(x <= .5, 0, .15 * (x-.5)/(5-.5)),
     dropout=function(x) .3*x/5
)
rcens <- function(n) 1 + (5-1) * (runif(n) ^ .5)
rcontrol <- function(n) f(n,'control')
rinterv <- function(n) f(n,'intervention')
set.seed(211)
x <- spower(rcontrol,rinterv,rcens, nc=350, ni=350, test=logrank, nsim=300)
sink()
%>
<%=format(x,digits=5)%>
```

AJAX is useful for hiding the cost of long computations. the javascript function "ReSimulate()" updates the browser page with a throbbing gif image to notify the user to wait for a bit until the call to spower can complete. once complete, the throbber disappears and the result is displayed in the browser.

www.rforge.net/**brew**

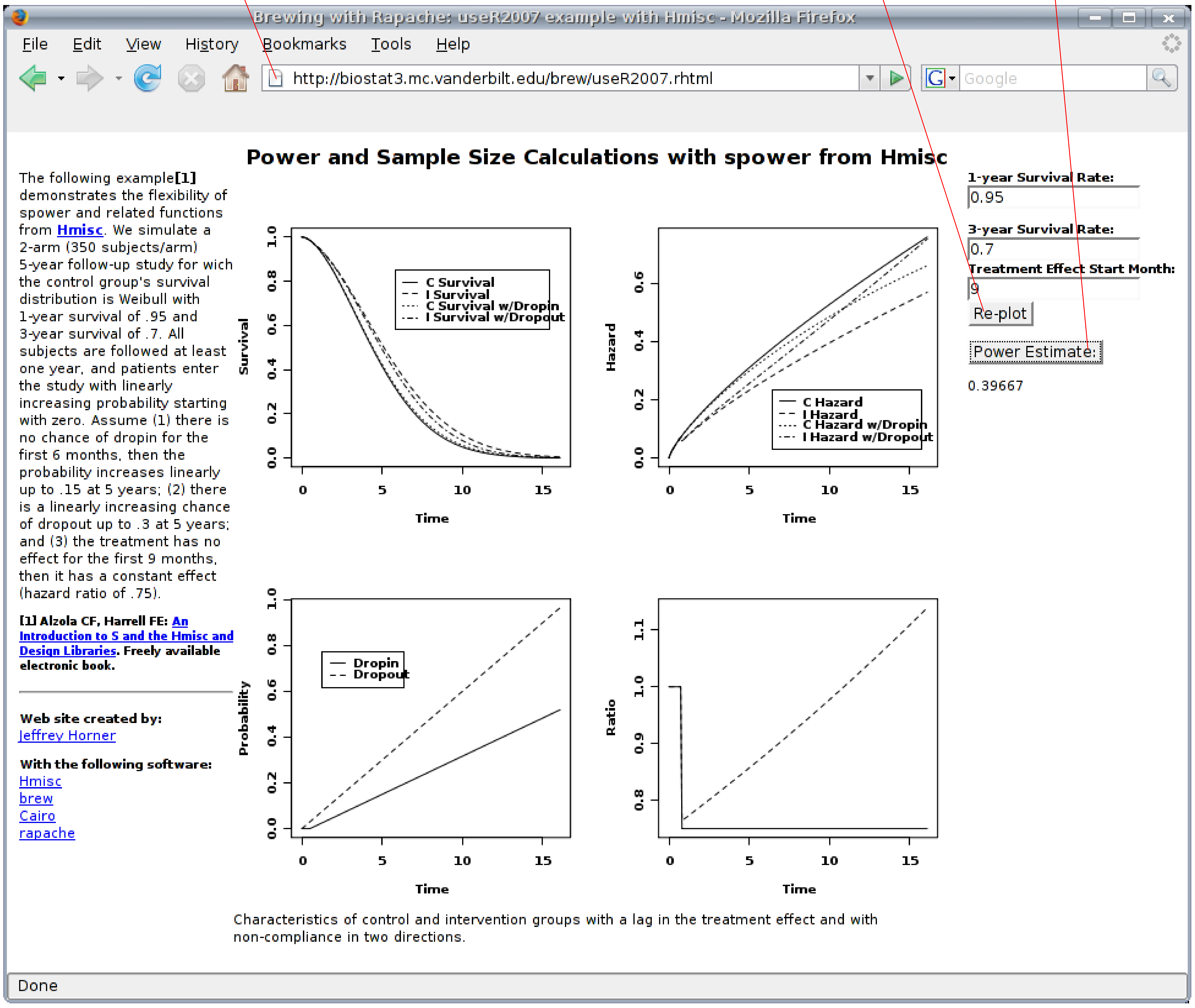
biostat.mc.vanderbilt.edu/**rapache**

biostat.mc.vanderbilt.edu/**JeffreyHorner**

the "expert #1" brew script produces the following page. surf to the url below to test drive.

clicking the "Re-plot" button runs the "expert #2" brew script.

clicking the "Power Estimate" button runs the "expert #3" brew script.



Power and Sample Size Calculations with spower from Hmisc

The following example**[1]** demonstrates the flexibility of spower and related functions from **Hmisc**. We simulate a 2-arm (350 subjects/arm) 5-year follow-up study for wich the control group's survival distribution is Weibull with 1-year survival of .95 and 3-year survival of .7. All subjects are followed at least one year, and patients enter the study with linearly increasing probability starting with zero. Assume (1) there is no chance of dropin for the first 6 months, then the probability increases linearly up to .15 at 5 years; (2) there is a linearly increasing chance of dropout up to .3 at 5 years; and (3) the treatment has no effect for the first 9 months, then it has a constant effect (hazard ratio of .75).

[1] Alzola CF, Harrell FE: An Introduction to S and the Hmisc and Design Libraries. Freely available electronic book.

Web site created by:
Jeffrey Horner

With the following software:
Hmisc
brew
Cairo
rapache

1-year Survival Rate: 0.95

3-year Survival Rate: 0.7

Treatment Effect Start Month: 9

0.39667

Characteristics of control and intervention groups with a lag in the treatment effect and with non-compliance in two directions.

the expert app