Sweave: First Steps Toward Reproducible Analyses

Kevin R. Coombes
Department of Bioinformatics and Computational Biology
Division of Quantitative Sciences
UT M. D. Anderson Cancer Center
kcoombes@mdanderson.org

5 February 2007

The First Problem: Reproducibility

Researcher contacts analyst: "I just read this interesting paper. Can you perform the same analysis on my data?"

Analyst reads paper. Finds algorithms described by biologists in English sentences that occupy minimal amount of space in the methods section.

Analyst gets public data from the paper. Takes wild guesses at actual algorithms and parameters. Is unable to reproduce reported results.

Analyst considers switching to career like bicycle repair, where reproducibility is less of an issue.

Alternate Forms of the Same Problem

1. Remember that microarray analysis you did six months ago? We ran a few more arrays. Can you add them to the project and repeat the same analysis?

2. The statistical analyst who looked at the data I generated previously is no longer available. Can you get someone else to analyze my new data set using the same methods (and thus producing a report I can expect to understand)?

3. Please write/edit the methods sections for the abstract/paper/grant proposal I am submitting based on the analysis you did several months ago.

The Code/Documentation Mismatch

Most of our analyses are performed using R. We can usually find an R workspace in a directory containing the raw data, the report, and one or more R scripts.

There is no guarantee that the objects in the R workspace were actually produced by those R scripts. Nor that the report matches the code. Nor the R objects.

Because R is interactive, unknown commands could have been typed at the command line, or the commands in the script could have been cut-n-pasted in a different order.

This problem is even worse if the software used for the analysis has a fancy modern GUI. It is impossible to document how you used the GUI in such a way that someone else could produce the exact same results—on the same data—six months later.

The Second Problem: Academic Anarchy

Every faculty member and (almost) every statistical analyst has his or her own favorite methods for analyzing each of the many kinds of data that we see in bioinformatics.

This contributes to the reproducibility problem, since everyone uses different methods and different code. But it also causes analyses to take longer and makes it harder to shift resources (i.e., people) around.

The obvious solution is to decide on standard methods for basic tasks. If we can also develop reusable templates that implement these standards, then we can speed up those analyses – and have a chance to produce better documentation as well.

The Solution: Sweave

$$\text{Sweave} = \text{R} + \text{LaTeX}.$$

This talk was prepared using Sweave. So was this standard report.

If you already know both R and LaTeX, then the ten-second version of this talk takes only two slides:

1. Prepare a LaTeX document. Give it an “Rnw” extension instead of “tex”. Say it is called “myfile.Rnw”
2. Insert an R code chunk starting with <<>>=
3. Terminate the R code chunk with an “at” sign (`@`) followed by a space.

Using Sweave

To produce the final document

1. In an R session, issue the command

   ```
   Sweave("myfile.Rnw")
   ```

   This executes the R code, inserts input commands and output computations and figures into a LaTeX file called "myfile.tex".

2. In the UNIX or DOS window (or using your favorite graphical interface), issue the command

   ```
   pdflatex myfile
   ```

   This produces a PDF file that you can use as you wish.

Viewing The Results

Here is a simple example, showing how the R input commands can generate output that is automatically included in the LaTeX output of Sweave.

```
> x <- rnorm(30)
> y <- rnorm(30)
> mean(x)

[1] -0.2074583

> cor(x, y)

[1] -0.2129101
```

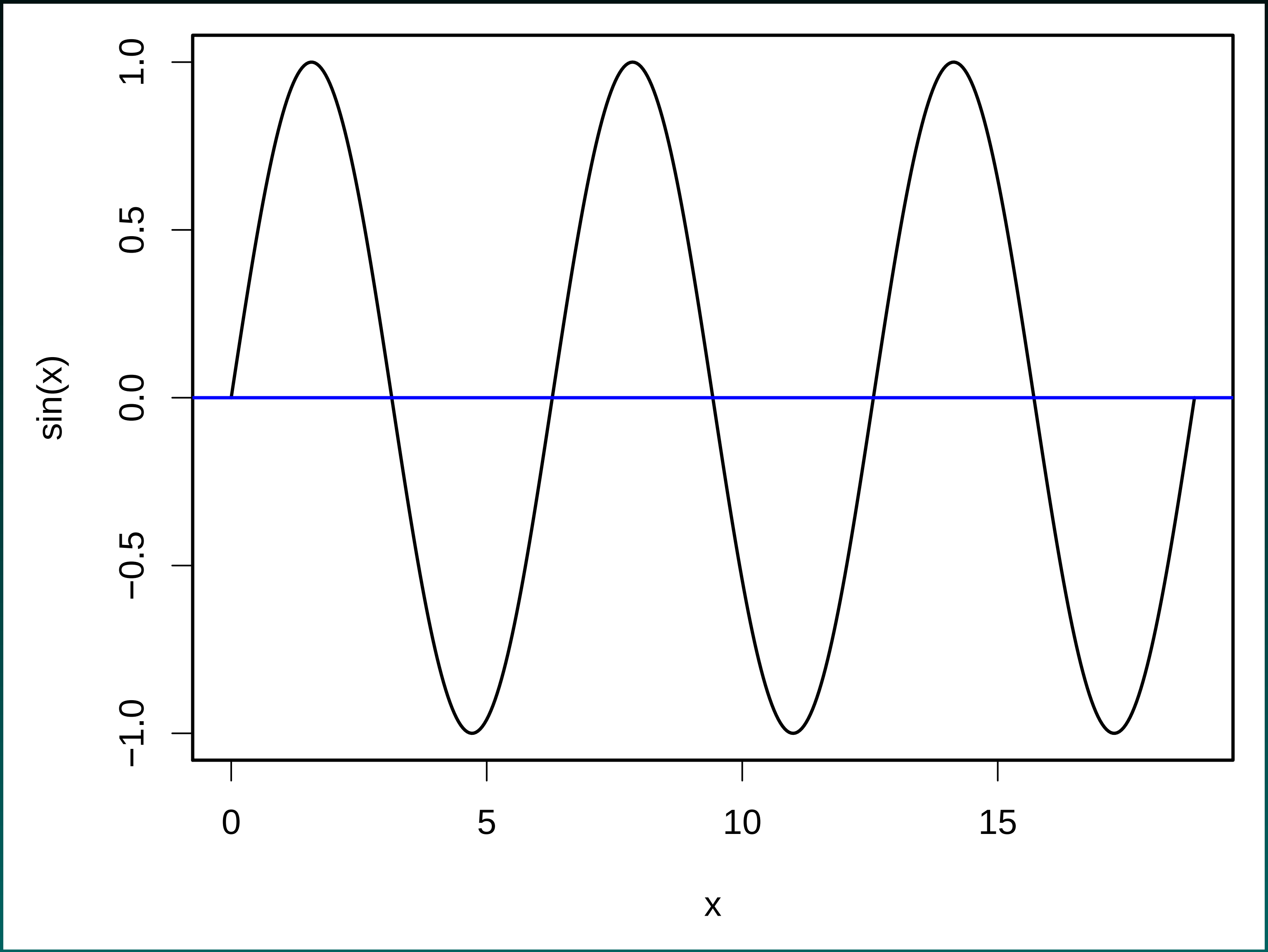
A Figure

Next, we are going to insert a figure. First, we can look at the R commands that are used to produce the figure.

```
> x <- seq(0, 6 * pi, length = 450)
> par(bg = "white", lwd = 2, cex = 1.3, mai = c(1.2,
+     1.2, 0.2, 0.2))
> plot(x, sin(x), type = "l")
> abline(h = 0, col = "blue")
```

On the next slide, we can look at the actual figure. (Part of the point of this example is to illustrate that you can separate the input from the output. You can even completely hide the input in the source file and just include the output in the report.)

Sine Curve



A Table

```
> library(xtable)
> x <- data.frame(matrix(rnorm(12), nrow = 3,
+     ncol = 4))
> dimnames(x) <- list(c("A", "B", "C"), c("C1",
+     "C2", "C3", "C4"))
> tab <- xtable(x, digits = c(0, 3, 3, 3, 3))
> tab
```

	C1	C2	C3	C4
A	0.208	−1.744	−0.826	−0.045
B	−1.115	−1.100	2.265	−0.419
C	−1.088	−0.373	1.089	−0.979

A Table, Repeated

Again, we want to point out that you can show the results—including tables—without showing the commands that generate them.

	C1	C2	C3	C4
A	0.208	−1.744	−0.826	−0.045
B	−1.115	−1.100	2.265	−0.419
C	−1.088	−0.373	1.089	−0.979

Weaving R Into HTML

The `R2HTML` package includes a driver that allows you to weave R commands, output, and figures into HTML documents instead of LaTeX documents. Of course, you must prepare the HTML part of the source appropriately, but the R code is built using the command:

```
Sweave("myfile.Rnw", driver=RweaveHTML)
```

It is worth noting that the `xtable` package can also generate HTML tables, using the commands

```
tab <- xtable(mydata)
print(tab, type="html")
```

Sweave Details

For the remainder of this talk, we will

- Look at the source "Rnw" file that produced this talk.
- Look at the standard report of an analysis of a simple Affymetrix experiment.
- Look at the source files that produced that report, and explain how they can be reused.