Functions from Formulas

R Pruim

January 27, 2012

Contents

1 Creating simple functions easily **1**

2 Plotting functions easily **2**

3 Calculus with functions **2**

4 Future plans **4**

1 Creating simple functions easily

The formula2function() function in the mosaic package makes it easy to create simple "algebraic" functions in R. By algebraic, we mean functions whose body is the evaluation of a mathematical expression built from other functions and the basic arithmetic operators.

Suppose, for example, that you wanted to define a "wave" function like

$$f(x) = a \sin(b * x) \ .$$

While it is straightforward to do this in R, the syntax is a bit heavy for beginners:

```
wave <- function(x, a, b) {
    a * sin(bx)
}
```

This is especially the case for anonymous functions that are created on the fly and passed to other functions for plotting, integrating, and the like.

formula2function() provides an alternative syntax for these simple functions:

```
wave <- formula2function(a * sin(b * x) ~ x & a & b)
```

The resulting function is identical to the one above

```
wave


function (x, a, b)
a * sin(b * x)
<environment: 0x2f46d88>
```

In the example above, we specified all of the inputs in the right side of the formula. The order of the arguments in the function will match the order given in the formula. While this is generally a good idea, it is only required that at least one input be declared in this manner. Other inputs are inferred from the left hand side of the formula an occur after the arguments given explicitly, but the order of these additional arguments is unspecified.

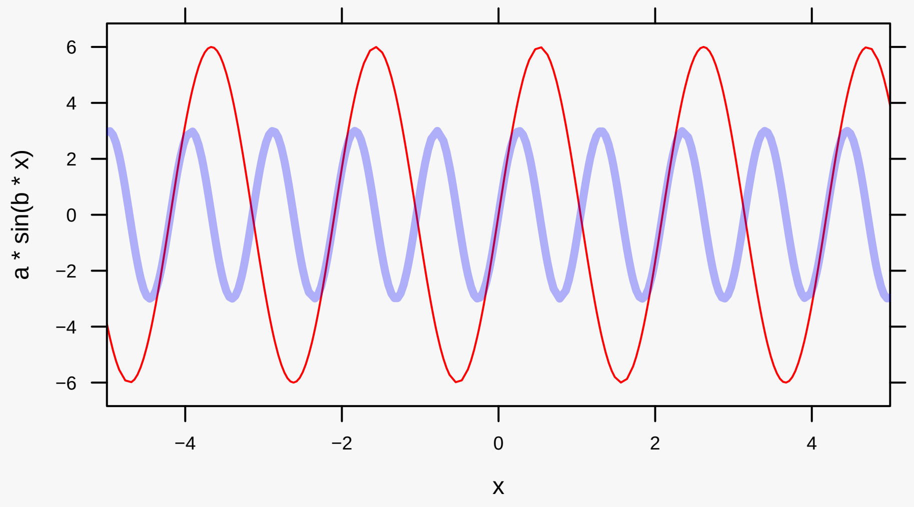
```
g <- formula2function(K * sin(x) * a * cos(y) ~ x & y)
g


function (x, y, K, a)
K * sin(x) * a * cos(y)
<environment: 0x3b71a5c>
```

2 Plotting functions easily

The graphics capabilities in R are tuned for displaying data. Displaying mathematical functions, although possible with a function like `plot()` is not as simple or as flexible as one would like. The `plotFun()` function attempts to remedy this. By borrowing the formula syntax of lattice plots and embedding `formula2function()`, plots of functions can be described quickly and easily.

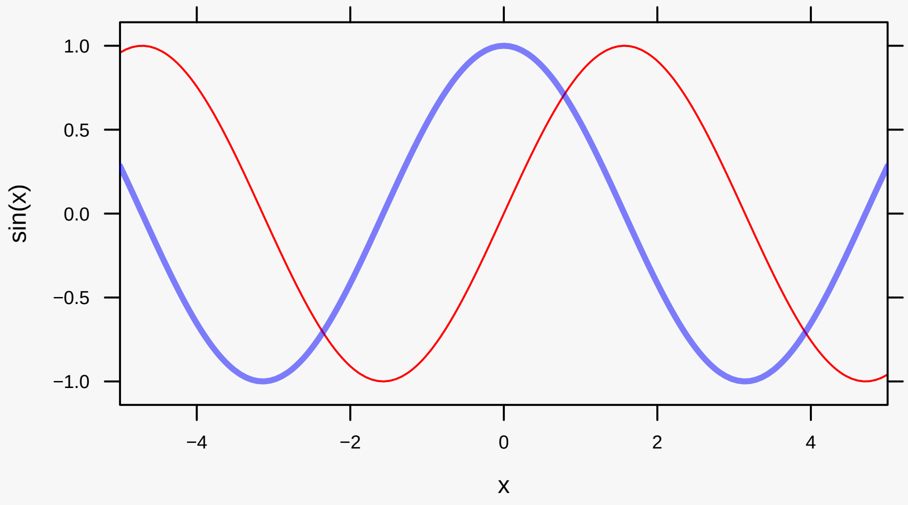
```
plotFun(a * sin(b * x) ~ x, a = 6, b = 3, xlim = range(-5,
    5), col = "red")
plotFun(a * sin(b * x) ~ x, a = 3, b = 6, xlim = range(-5,
    5), col = "blue", alpha = 0.3, lwd = 4, add = TRUE)
```



3 Calculus with functions

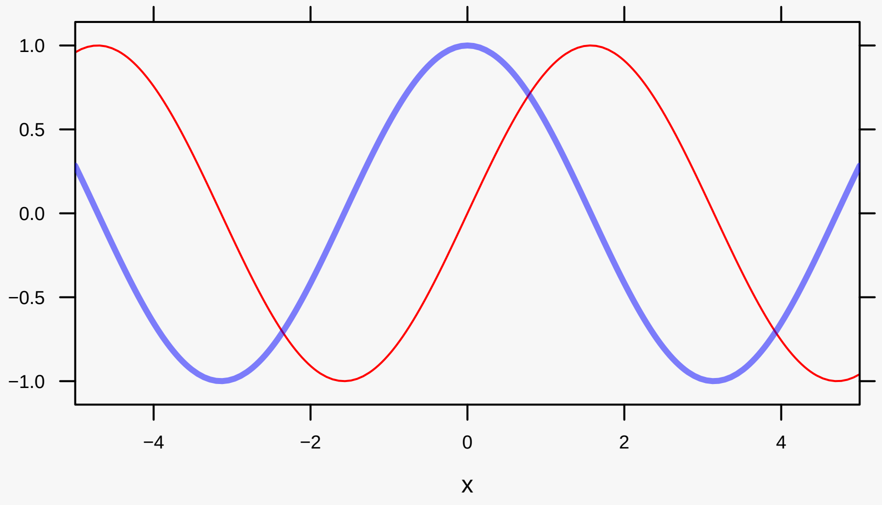
A similar formula interface is used in the differentiation and anti-differentiation operators.

```
plotFun(sin(x) ~ x, xlim = range(-5, 5), col = "red")
plotFun(D(sin(x) ~ x)(x) ~ x, xlim = range(-5, 5), add = TRUE,
    col = "blue", lwd = 3, alpha = 0.5)
```



The previous plotting code would look a bit friendlier with an intervening assignment, and the y-label should be changed or removed:

```
plotFun(sin(x) ~ x, xlim = range(-5, 5), col = "red", ylab = "")
dSin <- D(sin(x) ~ x)
plotFun(dSin(x) ~ x, xlim = range(-5, 5), add = TRUE, col = "blue",
    lwd = 3, alpha = 0.5)
```



4 Future plans

We continue to work on the `mosaic` package and specifically on the functions mentioned in the document. On the to-do list:

1. Use a numerical search for roots and critical points of a function to determine a default plotting region that is likely to show the most interesting aspects of the function.
2. Extend the formula interface to allow for multi-panel plots via the `|` operator.
3. Change the names of `xlim` and `ylim` so that they mention the true names of the arguments. Not all functions use x and y.