Selected R functions

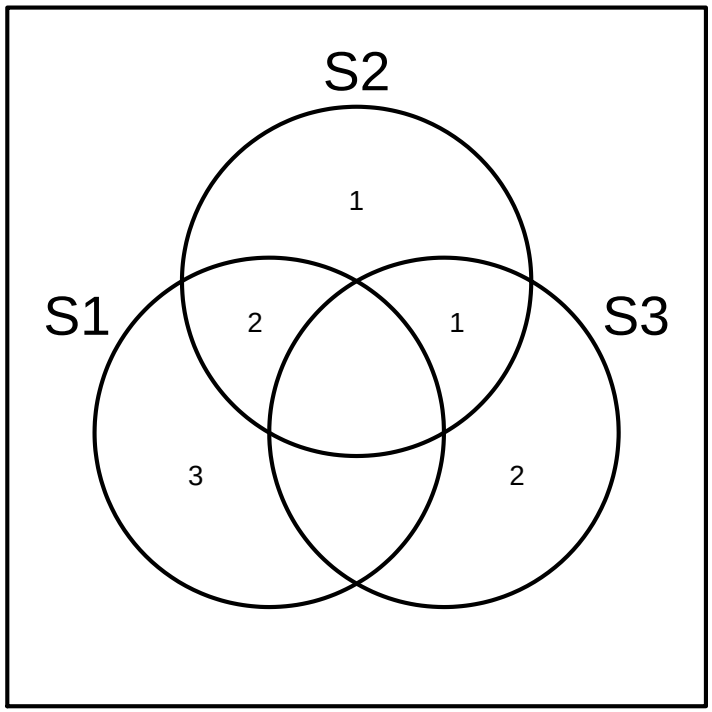
Marcin Kierczak

18 Nov 2016

Set operations

In R, working with sets is very natural. Let us begin by defining three sets and plotting Venn diagram of these.

```
library(venn)
S1 = c(1:5); S2=(4:7); S3=c(7:9)
venn(list(S1=S1, S2=S2, S3=S3))
```



Set operations cted.

$S_1 = 1, 2, 3, 4, 5; S_2 = 4, 5, 6, 7; S_3 = 7, 8, 9$

```
union(S1, S2) # Union
```

```
## [1] 1 2 3 4 5 6 7
```

```
intersect(S2, S3) # Intersection
```

```
## [1] 7
```

```
setdiff(S3, S2) # Difference
```

```
## [1] 8 9
```

Set operations cted.

$S_1 = 1, 2, 3, 4, 5; S_2 = 4, 5, 6, 7; S_3 = 7, 8, 9$

```
setequal(S1, S3) # Equality rel.
```

```
## [1] FALSE
```

```
is.element(2, S3) # Is element rel.
```

```
## [1] FALSE
```

Selected math functions – polynomials

To be able to work with polynomials, such as $x^3 + x^2 + 2x + 7$, we need to install and load the `polynom` package. Once this is done, let us define two polynomials, the one above and $3x^2 + 5x - 3$.

```
library(polynom)
poly1 <- polynomial(c(7,2,1,1))
poly1
```

```
## 7 + 2*x + x^2 + x^3
```

```
poly2 <- polynomial(c(-3, 5,3))
poly2
```

```
## -3 + 5*x + 3*x^2
```

Polynomials – defining, alternative 2

Polynomials can be also defined in terms of their zeros, i.e. the points where their value equals zero. Instead of zeros, one can provide coords of the points the polynomial has to contain.

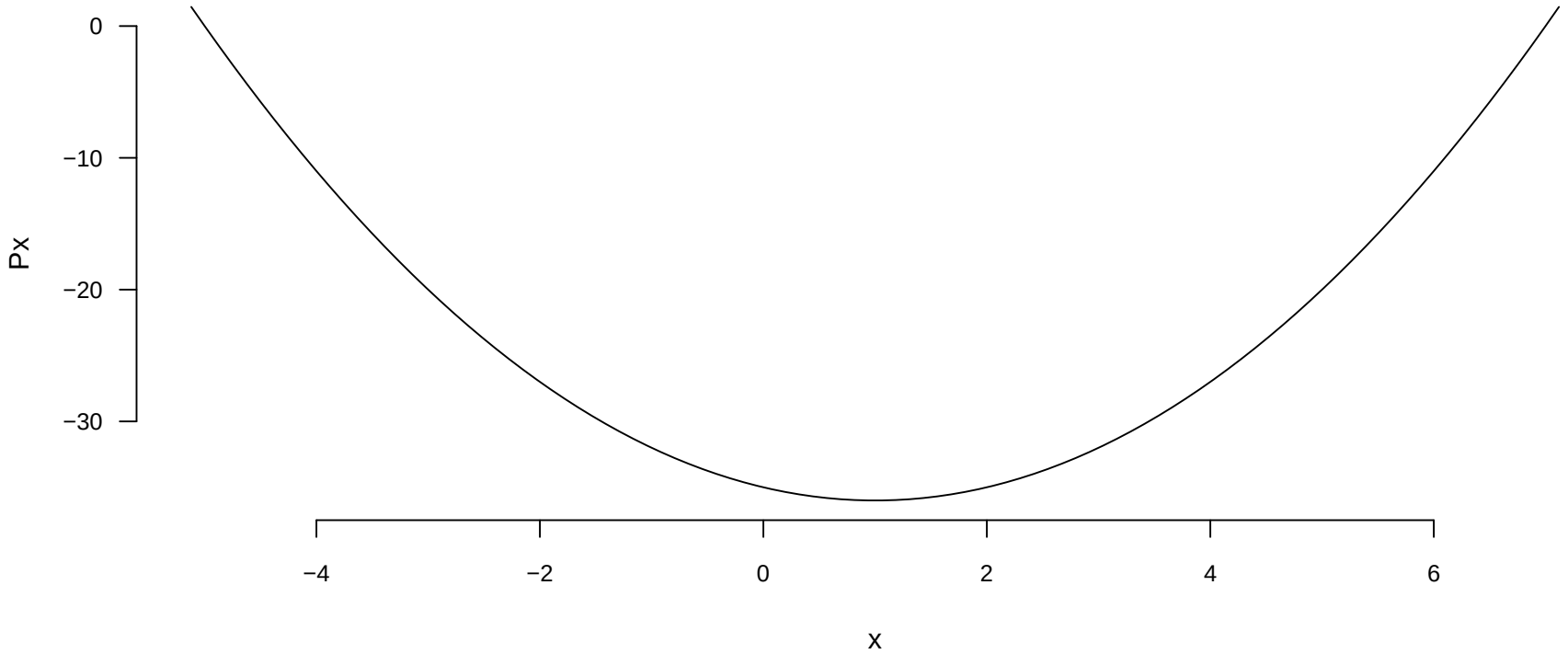
```
poly3 <- poly.calc(c(-5,7)) # zeros

# points A(1,-1); B(4,4); C(9,5)
poly.calc(c(1,4,9), c(-1,4,5))
```

```
## -3.4 + 2.583333*x - 0.1833333*x^2
```

Polynomials – plot

Let us visualize one of the defined polynomials and see whether its zeros are where they should be...

```
poly3 <- poly.calc(c(-5,7)) # zeros
p <- plot(poly3, bty='n', las=1, cex.axis=.8)
```



Polynomials – basic operations

Polynomials can be added, subtracted, multiplied and divided using standard operators:

```
poly1 + poly2
```

```
## 4 + 7*x + 4*x^2 + x^3
```

```
poly1 / poly2
```

```
## -0.2222222 + 0.3333333*x
```

Polynomials – calculus

We can also compute derivatives and integrals of polynomials:

```
# integrate on the [-2,2] interval
integral(poly1, c(-2,2))
```

```
## [1] 33.33333
```

```
deriv(poly2)
```

```
## 5 + 6*x
```

Polynomials – GCD and LCM

It is also easy to find the Greatest Common Divisor and the Least Common Multiple of two polynomials:

```
# integrate on the [-2,2] interval
GCD(poly1, poly3)
```

```
## 1
```

```
LCM(poly1, poly2)
```

```
## -21 + 29*x + 28*x^2 + 8*x^3 + 8*x^4 + 3*x^5
```

Math functions

R can also be used to examine functions: find their extrema and zeros. Say, we want to examine the $y = (x - 9)^2 - 5x - 2$ function. Let's define it first:

```
f <- function(x) {
  y <- (x - 9)^2 - 5*x - 2
  return(y)
}
f(2) # compute its value at x=2
```

```
## [1] 37
```

Math functions - zeros

To find zeros of the $y = (x-9)^2 - 5x - 2$ function we us the `uniroot` function. Note, we have to define the interval on which we are searching.

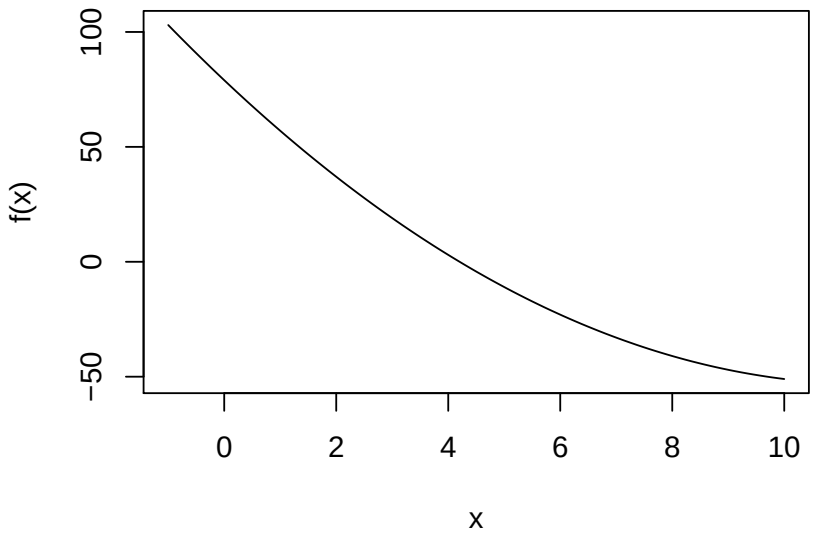
```
uniroot(f, lower=-1, upper=10)
```

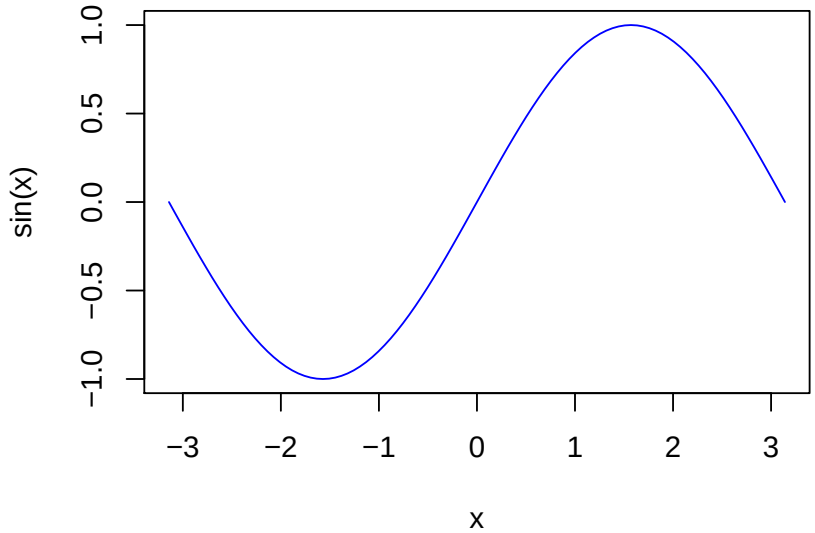
```
## $root
## [1] 4.20274
##
## $f.root
## [1] -1.518598e-06
##
## $iter
## [1] 6
##
## $init.it
## [1] NA
##
```

Math functions - plotting

Sure, one can plot a function, such as: $y = (x-9)^2 - 5x - 2$ or $sin(x)$.

```
par(mfrow=c(1,2))
curve(f, from = -1, to=10)
curve(sin, from=-pi, to=pi, col='blue')
```





Math formulas in R

In R, one can use the so-called *formulas*. A formula is a symbolic representation of a relationship between variables. Typically, on the left side of a formula, we have one variable, the dependent variable and on the right side, we have one or more explanatory variables that are forming an *expression*. For instance, we can write **y** $\sim$ **x** which means that y depends on x. The former is the *dependent variable* while the latter is the *independent* or *explanatory variable*.

- We can have more independent variables forming an expression, e.g.: **y** $\sim$ **x^2 + 2z + 3b**
- We also can have interactions between variables: **y** $\sim$ **a + b + a:b** which can also be written as: **y** $\sim$ **a*b**

Math formulas 2

- Writing: **y ~ (a + b) %in% c** is equivalent to: **y ~ a:c + b:c**,
- Similarly, the division sign **y ~ a/b** yields **y ~ a + a:b**,
- The minus sign excludes the variable: **y ~ a * b - a** equals to: **y ~ b + a:b**,
- To add the constant term, one can write +1, e.g.: **y ~ a + b + 1**,
- To remove the constant term, one can use either +0 or -1: **y ~ +0 + a + b**.

The formulas are very useful when defining *linear models* and in *calculus*.

Math – calculus

The symbolic calculus is implemented in package *stats* and pre-loaded automaticaly when you start R session. Derivation is implemented in two functions, *D()* and *deriv()* that differ in the form in which they take arguments: the former takes an expression as argument, the latter a formula with left side undifined. Let us do some derivation on one of the polynomials we have defined:

```
poly1
```

```
## 7 + 2*x + x^2 + x^3
```

```
deriv(poly1, 'x')
```

```
## 2 + 2*x + 3*x^2
```

Math – calculus cted.

To do the same using the *D()* function:

```
poly1
```

```
## 7 + 2*x + x^2 + x^3
```

```
p1 <- expression(x^3 +x^2 + 2*x)
deriv.call <- deriv(p1, 'x')
deriv.call
```

```
## expression({
##     .expr2 <- x^2
##     .expr4 <- 2 * x
##     .value <- x^3 + .expr2 + .expr4
##     .grad <- array(0, c(length(.value), 1L), list(NULL,
##     .grad[, "x"] <- 3 * .expr2 + .expr4 + 2
##     attr(.value, "gradient") <- .grad
##     .value
```

Math – calculus cted.

Let us now evaluate this derivative in points $x = \{1, \ldots, 5\}$:

```
x <- 1:5
eval(deriv.call)
```

```
## [1]   4  16  42  88 160
## attr(,"gradient")
##       x
## [1,]  7
## [2,] 18
## [3,] 35
## [4,] 58
## [5,] 87
```

Math – calculus, integration.

Symbolic integration is a bit more tricky, let's look at the numerical integration in R. We are, say, interested in the area under the $N(0, 1)$ on the $-.5, .5$ interval:

```
integrate(dnorm, lower = -.5, upper=.5, mean = 0, sd = 1)
```

```
## 0.3829249 with absolute error < 4.3e-15
```

And here is the graphical illustration of the area integrated in the above example: