Ejecicicos

Gerald Palomino Monge

22 de septiembre de 2017

Ejercicios R Markdown

Ejercicio 1:

operaciones

```
  1+2*(3+4)
```

```
## [1] 15
```

```
  4**3+3**2+1
```

```
## [1] 74
```

```
  sqrt((4+3)*(2+1))
```

```
## [1] 4.582576
```

```
  ((1+2)/(3+4))
```

```
## [1] 0.4285714
```

desviacion estandar

```
  sd(1:100)
```

```
## [1] 29.01149
```

demostracion de simbolos matematicos

```
  demo(plotmath)
```

```
##
##
##  demo(plotmath)
##  ---- ~~~~~~~~
##
## > #  Copyright (C) 2002-2016 The R Core Team
## >
## > require(datasets)
##
## > require(grDevices); require(graphics)
##
## > ## --- "math annotation" in plots :
## >
## > ######
## > # create tables of mathematical annotation functionality
## > ######
## > make.table <- function(nr, nc) {
## +     savepar <- par(mar=rep(0, 4), pty="s")
## +     plot(c(0, nc*2 + 1), c(0, -(nr + 1)),
## +          type="n", xlab="", ylab="", axes=FALSE)
## +     savepar
```

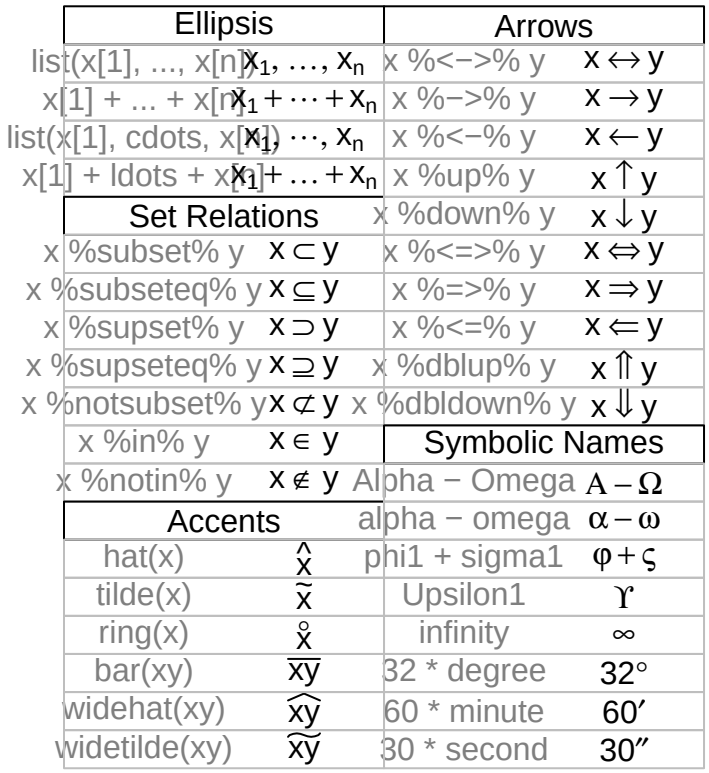
```
## + }
##
## > get.r <- function(i, nr) {
## +     i %% nr + 1
## + }
##
## > get.c <- function(i, nr) {
## +     i %/% nr + 1
## + }
##
## > draw.title.cell <- function(title, i, nr) {
## +     r <- get.r(i, nr)
## +     c <- get.c(i, nr)
## +     text(2*c - .5, -r, title)
## +     rect((2*(c - 1) + .5), -(r - .5), (2*c + .5), -(r + .5))
## + }
##
## > draw.plotmath.cell <- function(expr, i, nr, string = NULL) {
## +     r <- get.r(i, nr)
## +     c <- get.c(i, nr)
## +     if (is.null(string)) {
## +         string <- deparse(expr)
## +         string <- substr(string, 12, nchar(string) - 1)
## +     }
## +     text((2*(c - 1) + 1), -r, string, col="grey50")
## +     text((2*c), -r, expr, adj=c(.5,.5))
## +     rect((2*(c - 1) + .5), -(r - .5), (2*c + .5), -(r + .5), border="grey")
## + }
##
## > nr <- 20
##
## > nc <- 2
##
## > oldpar <- make.table(nr, nc)
```

Arithmetic Operators		Lists	
x + y	$x + y$	list(x, y, z)	x, y, z
x − y	$x - y$	**Relations**	
x * y	xy	x == y	$x = y$
x/y	x/y	x != y	$x \neq y$
x %+−% y	$x \pm y$	x < y	$x < y$
x%/%y	$x \div y$	x <= y	$x \leq y$
x %*% y	$x \times y$	x > y	$x > y$
x %.% y	$x \cdot y$	x >= y	$x \geq y$
−x	$-x$	x %~~% y	$x \approx y$
+x	$+x$	x %=~% y	$x \cong y$
Sub/Superscripts		x %==% y	$x \equiv y$
x[i]	x_i	x %prop% y	$x \propto y$
x^2	x^2	x %~% y	$x \sim y$
Juxtaposition		**Typeface**	
x * y	xy	plain(x)	x
paste(x, y, z)	xyz	italic(x)	*x*
Radicals		bold(x)	**x**
sqrt(x)	$\sqrt{x}$	bolditalic(x)	***x***
sqrt(x, y)	$\sqrt[y]{x}$	underline(x)	$\underline{x}$

```
##
## > i <- 0
##
## > draw.title.cell("Arithmetic Operators", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x + y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x - y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x * y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x / y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %+-% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %/% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %*% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %.% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(-x), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(+x), i, nr); i <- i + 1
##
## > draw.title.cell("Sub/Superscripts", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x[i]), i, nr); i <- i + 1
##
```

```
## > draw.plotmath.cell(expression(x^2), i, nr); i <- i + 1
##
## > draw.title.cell("Juxtaposition", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x * y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(paste(x, y, z)), i, nr); i <- i + 1
##
## > draw.title.cell("Radicals", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(sqrt(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(sqrt(x, y)), i, nr); i <- i + 1
##
## > draw.title.cell("Lists", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(list(x, y, z)), i, nr); i <- i + 1
##
## > draw.title.cell("Relations", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x == y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x != y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x < y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x <= y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x > y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x >= y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %~~% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %=~% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %==% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %prop% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %~% y), i, nr); i <- i + 1
##
## > draw.title.cell("Typeface", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(plain(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(italic(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(bold(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(bolditalic(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(underline(x)), i, nr); i <- i + 1
##
```

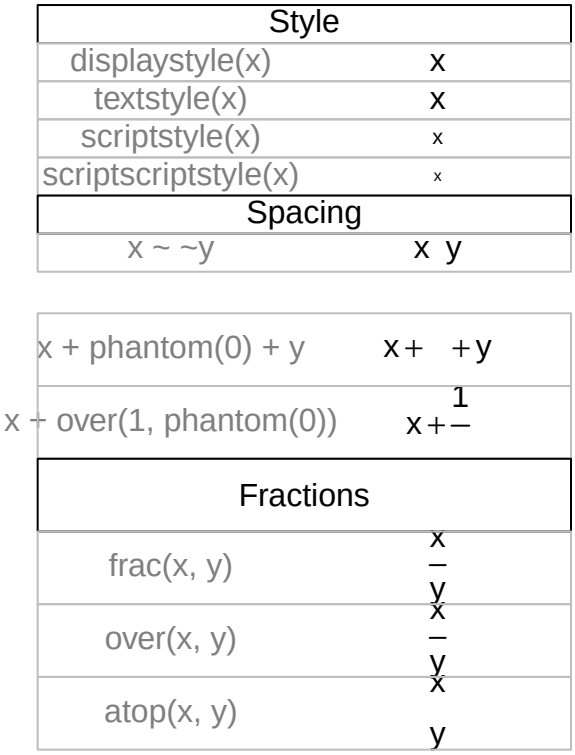
```
## > # Need fewer, wider columns for ellipsis ...
## > nr <- 20
##
## > nc <- 2
##
## > make.table(nr, nc)
```



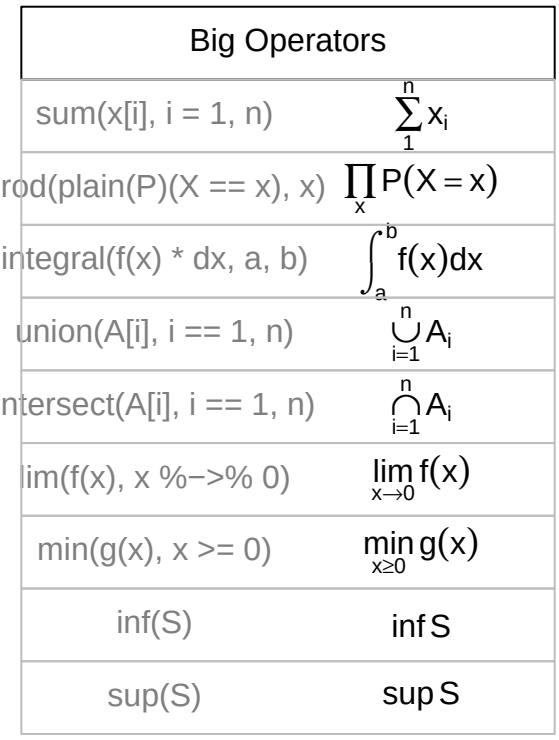

```
## $mar
## [1] 0 0 0 0
##
## $pty
## [1] "s"
##
##
## > i <- 0
##
## > draw.title.cell("Ellipsis", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(list(x[1], ..., x[n])), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x[1] + ... + x[n]), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(list(x[1], cdots, x[n])), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x[1] + ldots + x[n]), i, nr); i <- i + 1
##
## > draw.title.cell("Set Relations", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %subset% y), i, nr); i <- i + 1
##
```

```
## > draw.plotmath.cell(expression(x %subseteq% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %supset% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %supseteq% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %notsubset% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %in% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %notin% y), i, nr); i <- i + 1
##
## > draw.title.cell("Accents", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(hat(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(tilde(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(ring(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(bar(xy)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(widehat(xy)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(widetilde(xy)), i, nr); i <- i + 1
##
## > draw.title.cell("Arrows", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %<->% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %->% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %<-% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %up% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %down% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %<=>% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %=>% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %<=% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %dblup% y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x %dbldown% y), i, nr); i <- i + 1
##
## > draw.title.cell("Symbolic Names", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(Alpha - Omega), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(alpha - omega), i, nr); i <- i + 1
##
```

```
## > draw.plotmath.cell(expression(phi1 + sigma1), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(Upsilon1), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(infinity), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(32 * degree), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(60 * minute), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(30 * second), i, nr); i <- i + 1
##
## > # Need even fewer, wider columns for typeface and style ...
## > nr <- 20
##
## > nc <- 1
##
## > make.table(nr, nc)
## $mar
## [1] 0 0 0 0
##
## $pty
## [1] "s"
##
##
## > i <- 0
##
## > draw.title.cell("Style", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(displaystyle(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(textstyle(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(scriptstyle(x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(scriptscriptstyle(x)), i, nr); i <- i + 1
##
## > draw.title.cell("Spacing", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x ~~ y), i, nr); i <- i + 1
##
## > # Need fewer, taller rows for fractions ...
## > # cheat a bit to save pages
## > par(new = TRUE)
##
## > nr <- 10
##
## > nc <- 1
##
## > make.table(nr, nc)
```

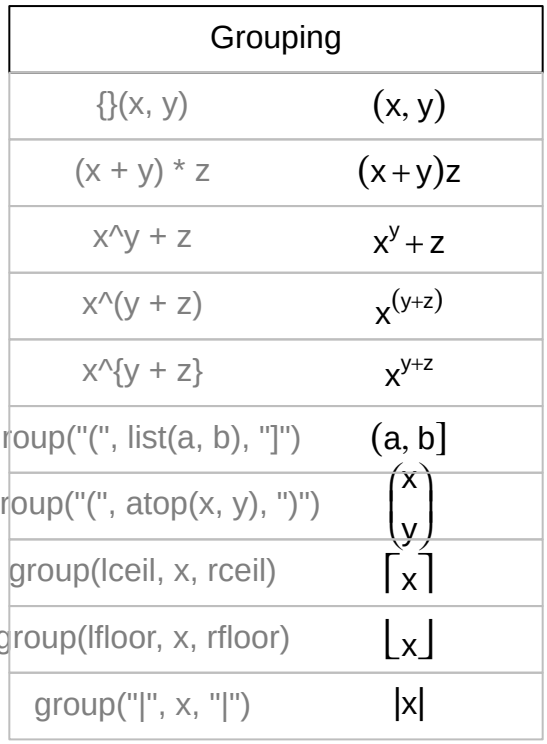



```
## $mar
## [1] 0 0 0 0
##
## $pty
## [1] "s"
##
##
## > i <- 4
##
## > draw.plotmath.cell(expression(x + phantom(0) + y), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x + over(1, phantom(0))), i, nr); i <- i + 1
##
## > draw.title.cell("Fractions", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(frac(x, y)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(over(x, y)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(atop(x, y)), i, nr); i <- i + 1
##
## > # Need fewer, taller rows and fewer, wider columns for big operators ...
## > nr <- 10
##
## > nc <- 1
##
## > make.table(nr, nc)
```




```
## $mar
## [1] 0 0 0 0
##
## $pty
## [1] "s"
##
##
## > i <- 0
##
## > draw.title.cell("Big Operators", i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(sum(x[i], i=1, n)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(prod(plain(P)(X == x), x)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(integral(f(x) * dx, a, b)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(union(A[i], i==1, n)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(intersect(A[i], i==1, n)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(lim(f(x), x %->% 0)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(min(g(x), x >= 0)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(inf(S)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(sup(S)), i, nr); i <- i + 1
##
## > nr <- 11
```

```
##
## > make.table(nr, nc)
```




```
## $mar
## [1] 0 0 0 0
##
## $pty
## [1] "s"
##
##
## > i <- 0
##
## > draw.title.cell("Grouping", i, nr); i <- i + 1
##
## > # Those involving '{ . }' have to be done "by hand"
## > draw.plotmath.cell(expression({}(x , y)), i, nr, string="{}(x, y)"); i <- i + 1
##
## > draw.plotmath.cell(expression((x + y)*z), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x^y + z),   i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x^(y + z)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(x^{y + z}), i, nr, string="x^{y + z}"); i <- i + 1
##
## > draw.plotmath.cell(expression(group("(", list(a, b), "]")), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(bgroup("(", atop(x, y), ")")), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(group(lceil, x, rceil)), i, nr); i <- i + 1
##
```

```
## > draw.plotmath.cell(expression(group(lfloor, x, rfloor)), i, nr); i <- i + 1
##
## > draw.plotmath.cell(expression(group("|", x, "|")), i, nr); i <- i + 1
##
## > par(oldpar)
```

mil masotas

```
  mascotas=sample(c("perro","gato","pollo","pez dorado"),1000,replace = TRUE)
  mascotas[1:10]
```

```
##  [1] "pollo"      "perro"      "pez dorado" "pez dorado" "perro"
##  [6] "pez dorado" "pollo"      "pez dorado" "perro"      "pollo"
```

```
  sum(mascotas=="perro")
```

```
## [1] 277
```

```
  sum(mascotas=="pollo")
```

```
## [1] 248
```

```
  sum(mascotas=="gato")
```

```
## [1] 239
```

```
  sum(mascotas=="pez dorado")
```

```
## [1] 236
```

funciones

```
  f<-function(n){
    if (n%%2==0)
      return(n/2)
    else
      return(3*n+1)
  }

  collats<-function(n){
    cont=0
    while (n>1){
      n=f(n)
      cont=1+cont
    }
    return(cont)
  }


  collats2<-function(n){
    cont=0
    while (n>1){
      n=f(n)
      print(n)
      cont=1+cont
    }
    return(cont)
  }
```

```
  numero=100

  cantidad=collats(numero)

  for (i in 100:200){
      if (collats(i)<cantidad){
        numero=i
        cantidad=collats(i)
      }
  }

  numero
```

```
## [1] 128
```

```
  cantidad
```

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

```
  kk=collats2(numero)
```

```
## [1] 64
## [1] 32
## [1] 16
## [1] 8
## [1] 4
## [1] 2
## [1] 1
```

algorimo de euclides

```
  a<-36
  b<-24
  while (b!=0){
    carry<-a%%b
    a<-b
    b<-carry
  }
  a
```

```
## [1] 12
```

```
  b
```

```
## [1] 0
```