R examples

Edwin Ruiz

April 3, 2024

```r
#<- is the same as = but it is conventional for R

#Basic data type in R is vector, example
x <- 1:5

#adding to each element in the vector:
x + 100
```

```
## [1] 101 102 103 104 105
```

```r
#recycling:
x + c(100, 1)
```

```
## Warning in x + c(100, 1): longer object length is not a multiple of shorter
## object length
```

```
## [1] 101   3 103   5 105
```

```r
#Notes:
#  c() function in R is used to combine or concatenate its argument


#Adding an additional element to vector:
1:6 + c(100, 1)
```

```
## [1] 101   3 103   5 105   7
```

```r
#Creating string vectors:
y <- c("Edwin", "Neuro", "ITS", "UCSD")
y[2]
```

```
## [1] "Neuro"
```

```r
#Error when adding numerical to string:
#y + 100

#Appending more string values to already defined vector string:
paste(y, "loves R")
```

```
## [1] "Edwin loves R" "Neuro loves R" "ITS loves R"   "UCSD loves R"
```

```r
#defining logicals:
z <- c(T, F, T, F)
z
```

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

```
#Notes:
#  data types in R:
#     numeric == integer in python
#     character == string in python
#     logicals == Boolean in python

#Adding to logicals
z + 100
```

```
## [1] 101 100 101 100
```

```
#Returning logicals for vectors:
x > 3
```

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

```
#summing up logicals logic:
sum(x > 3)
```

```
## [1] 2
```

```
#misinterpreting a vector of intergers to character:
y <- c(5, 10, 1, "edwin")
y
```

```
## [1] "5"     "10"    "1"     "edwin"
```

```
#turning logicals to numericals:
y <- c(5, 10, 1, F, T)
y
```

```
## [1]  5 10  1  0  1
```

```
#Turning logicals into characters:
y <-  c("edwin", F, T)
y
```

```
## [1] "edwin" "FALSE" "TRUE"
```

```
#Notes:
#  Data types precedence:
#   1.  Character
#   2.  Numeric
# 3.    Logicals

#Creating dataframe:
df <- data.frame(numbs=1:5, chars=letters[1:5], logical=c(T,T,F,T,F))
df
```

```
##   numbs chars logical
## 1     1     a    TRUE
## 2     2     b    TRUE
## 3     3     c   FALSE
## 4     4     d    TRUE
## 5     5     e   FALSE
```

```
#Accessing columns for each
df$logical
```

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

```r
df$chars
```

```
## [1] "a" "b" "c" "d" "e"
```

```r
df$numbs
```

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

```r
#Accessing a specific column (here is column 2):
df[,2]
```

```
## [1] "a" "b" "c" "d" "e"
```

```r
#accessing a specific value from the df:
#syntax: df[row, column]

#trying to get c
df[3,2]
```

```
## [1] "c"
```

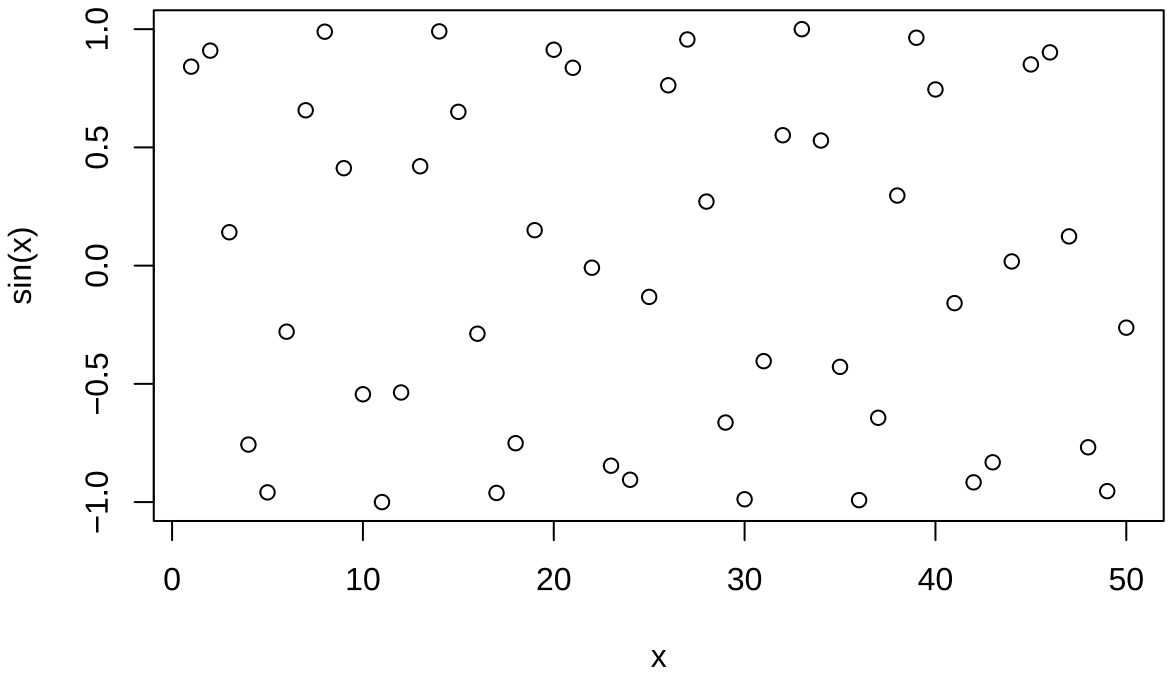
```r
#can also use:
df$chars[3]
```

```
## [1] "c"
```

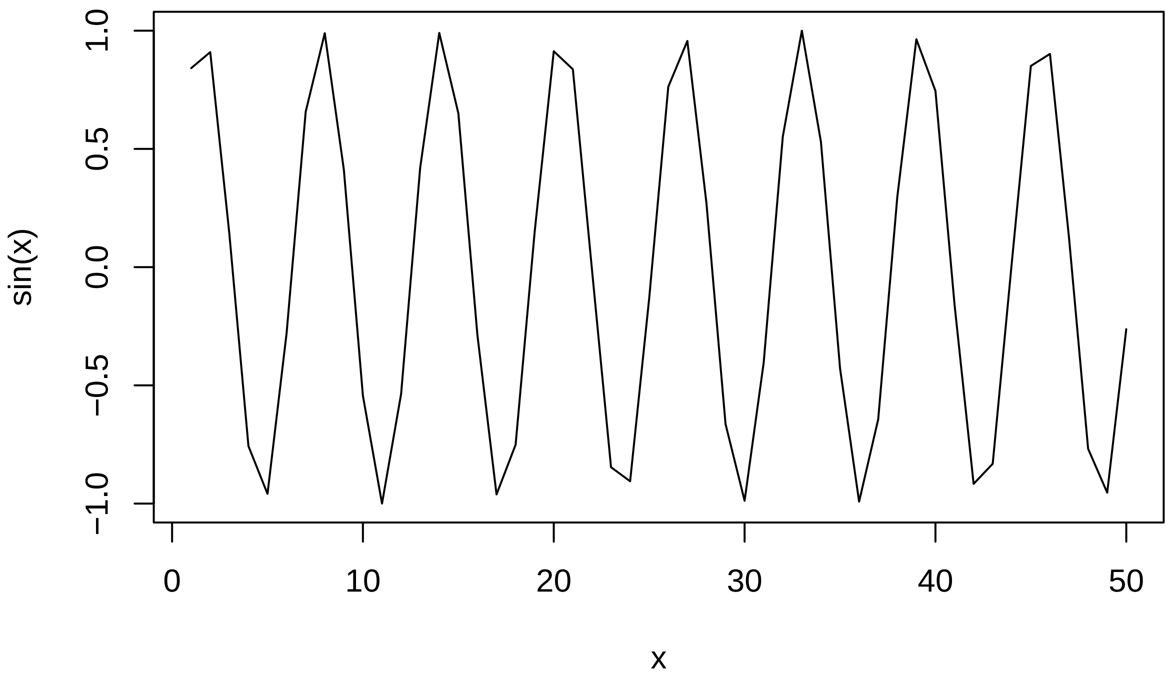
```r
#taking sin() of a vector:
x <- 1:50
sin(x)
```

```
##  [1]  0.841470985  0.909297427  0.141120008 -0.756802495 -0.958924275
##  [6] -0.279415498  0.656986599  0.989358247  0.412118485 -0.544021111
## [11] -0.999990207 -0.536572918  0.420167037  0.990607356  0.650287840
## [16] -0.287903317 -0.961397492 -0.750987247  0.149877210  0.912945251
## [21]  0.836655639 -0.008851309 -0.846220404 -0.905578362 -0.132351750
## [26]  0.762558450  0.956375928  0.270905788 -0.663633884 -0.988031624
## [31] -0.404037645  0.551426681  0.999911860  0.529082686 -0.428182669
## [36] -0.991778853 -0.643538133  0.296368579  0.963795386  0.745113160
## [41] -0.158622669 -0.916521548 -0.831774743  0.017701925  0.850903525
## [46]  0.901788348  0.123573123 -0.768254661 -0.953752653 -0.262374854
```

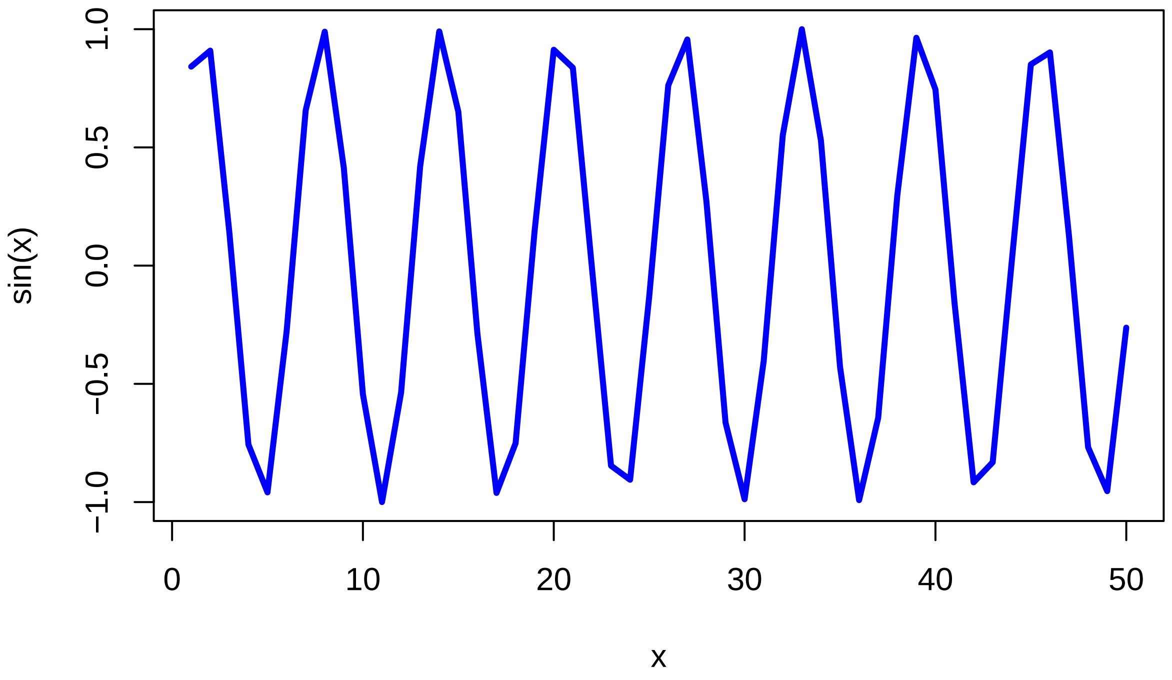
```r
#Default is points ( plot(x, sin(x), typ="p") ):
plot(x, sin(x))
```



```
#Changing it to a line plot:
plot(x, sin(x), typ="l")
```



```
#Changing it to thicker line and color blue:
plot(x, sin(x), typ="l", col="blue", lwd=3)
```



```
#Using logs:
log(10)
```

```
## [1] 2.302585
```

```
log(10, base=10)
```

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

```
#Notes:
#  Plotting defaults:
#  ?plot.default
```