Week 2

Irlanda Ayon-Moreno

3/4/2018

Arrays and Factos

Matrices & Arrays

You can transform a vector into a n-dimensional array by giving it a **dimensions** attribute The dimensions attribute is a numeric vector with as many elements as desired dimensions

```
x <- 1:8
dim(x) <- c(2,4)
# dim(x) <- c(2,2,2)
x
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
```

To have more control over how a matrix is filled, we use the function `matrix()`

```
a <- 1:8
A <- matrix(a, nrow = 2, ncol = 4)
A
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
```

About Matrices
- R stores matrices as vectors
- matrices are also atoic
- Matrices in R are stored column-major - If you want to fill a matrix by rows, use `byrow = TRUE`

```
b <- 1:8
B <- matrix(b, nrow = 2, ncol = 4, byrow = TRUE)
B
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
```

Factors

A factor is designed to handle **categorical** data. (Especially data with an "ordinal" scale)
Factors are internally stored as vectors of integers To create a factor, pass a vector to `factor()`

```r
size <- c("sm", "md", "lg", "md")
size <- factor(size)
size
```

```
[1] sm md lg md
Levels: lg md sm
```

Lists

A list is the most general data structure in R
Lists can contain any other type of data structure (even other lists)
Lists are a special type of vector `lst <- vector(mode = "list")`
Lists are vectors in the sense of being a one-dimensional object
Lists are **not** atomic structures

Subsetting and Indexing

Use the bracket notation system
use [] to extract values from a list

```r
lst <- list(
  c(1,2,3),
  matrix(1:9, nrow = 3, ncol = 3),
  list(1:2, c(TRUE, FALSE), c("a","b"))
)
lst
```

```
[[1]]
[1] 1 2 3

[[2]]
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

[[3]]
[[3]][[1]]
[1] 1 2

[[3]][[2]]
[1]  TRUE FALSE

[[3]][[3]]
[1] "a" "b"
```

```r
#access an element
lst[3]
```

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

[[1]][[2]]
[1]  TRUE FALSE

[[1]][[3]]
[1] "a" "b"
```

```
# access object of list element
lst[[3]]
```

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

[[2]]
[1]  TRUE FALSE

[[3]]
[1] "a" "b"
```

```
# access objects inside the element
lst[[3]][1]
```

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

```
lst[[3]][[1]][1]
```

```
[1] 1
```

$ to access list named element(s)

list$name

```
# giving names to elements in the list
lst2 <- list(
  vec = c(1,2,3),
  mat = matrix(1:9, nrow = 3, ncol = 3),
  lis = list(1:2, c(TRUE, FALSE), c("a","b"))
)
lst2
```

```
$vec
[1] 1 2 3

$mat
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
$lis
$lis[[1]]
[1] 1 2

$lis[[2]]
[1]  TRUE FALSE

$lis[[3]]
[1] "a" "b"
```

```
lst2$vec
```

```
[1] 1 2 3
```

```
# giving names to elements in the list
names(lst) <- c("A", "B", "C")
names(lst)
```

```
[1] "A" "B" "C"
```

```
lst
```

```
$A
[1] 1 2 3

$B
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

$C
$C[[1]]
[1] 1 2

$C[[2]]
[1]  TRUE FALSE

$C[[3]]
[1] "a" "b"
```

```
vec = c(1,2,3)
mat = matrix(1:9, nrow = 3, ncol = 3)
lis = list(1:2, c(TRUE, FALSE), c("a","b"))
lst3 = list(vec, mat, lis)
lst3
```

```
[[1]]
[1] 1 2 3
```

```
[[2]]
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

[[3]]
[[3]][[1]]
[1] 1 2

[[3]][[2]]
[1]  TRUE FALSE

[[3]][[3]]
[1] "a" "b"
```

```
lst4 = list(vec = vec, mat = mat, lis = lis)
lst4
```

```
$vec
[1] 1 2 3

$mat
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

$lis
$lis[[1]]
[1] 1 2

$lis[[2]]
[1]  TRUE FALSE

$lis[[3]]
[1] "a" "b"
```

Graphics

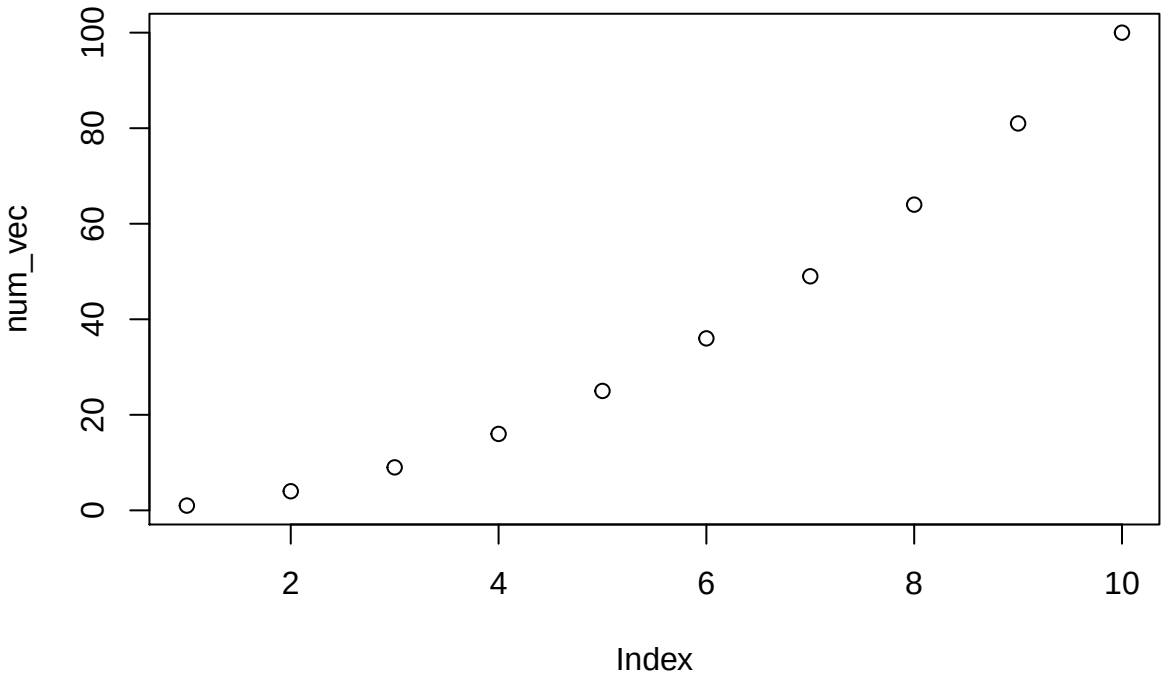
- "graphics" and "grid" are the two main graphics systems in R
- "graphics" is the traditional system, also referred to as base graphics
- high-level : functions produce complete plots
- low-level : functions add further output to an existing plot

- "grid" prodives low-level functions for programming plotting functions

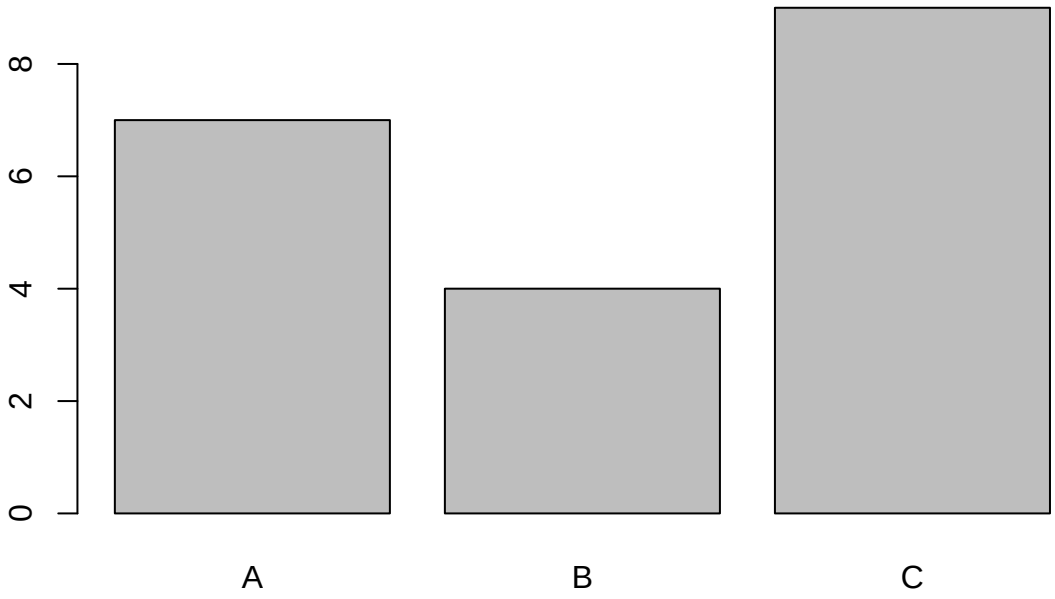
The plot() Function

- most important high-level function
- the 1st argument provides the data to plot
- the data can take diff. formsL vectors, factors, matrices, data frames
- one / two / multiple variables
- you can create your own `plot()` method function

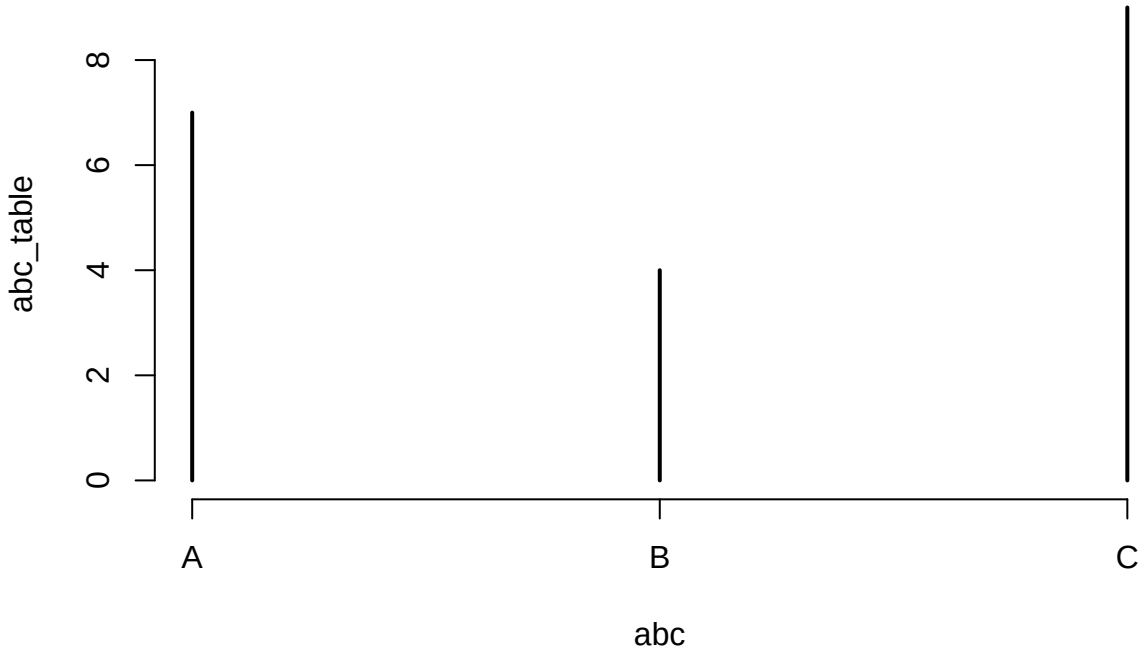
```
num_vec <- (c(1:10))^2
plot(num_vec)
```



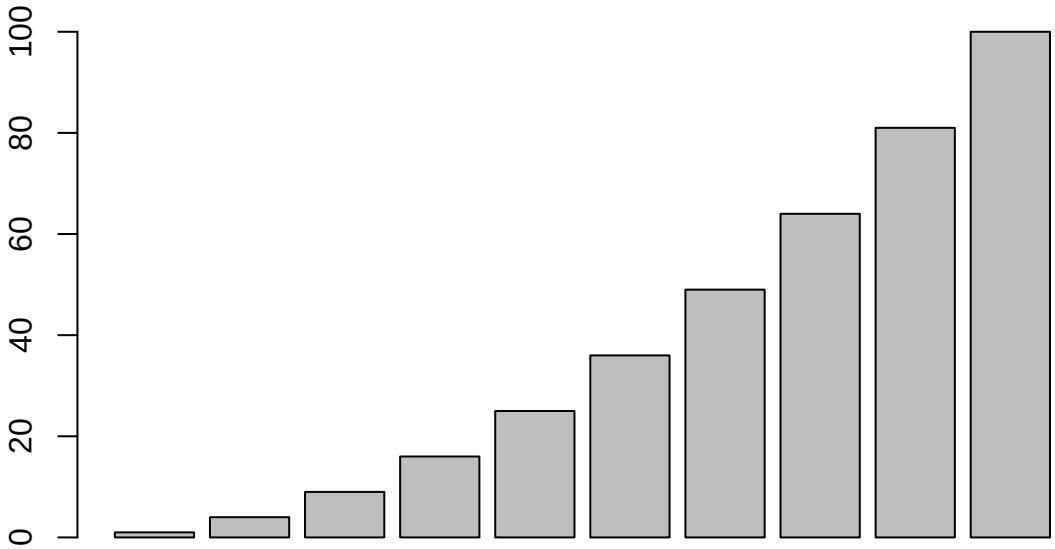

```
set.seed(4)
abc <- factor(sample(c('A', 'B', 'C'), 20, replace = TRUE))
plot(abc)
```



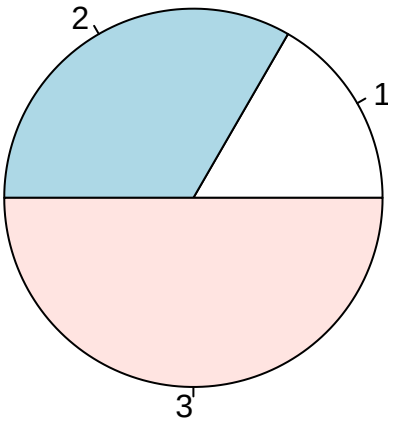

```
abc_table <- table(abc)
plot(abc_table)
```



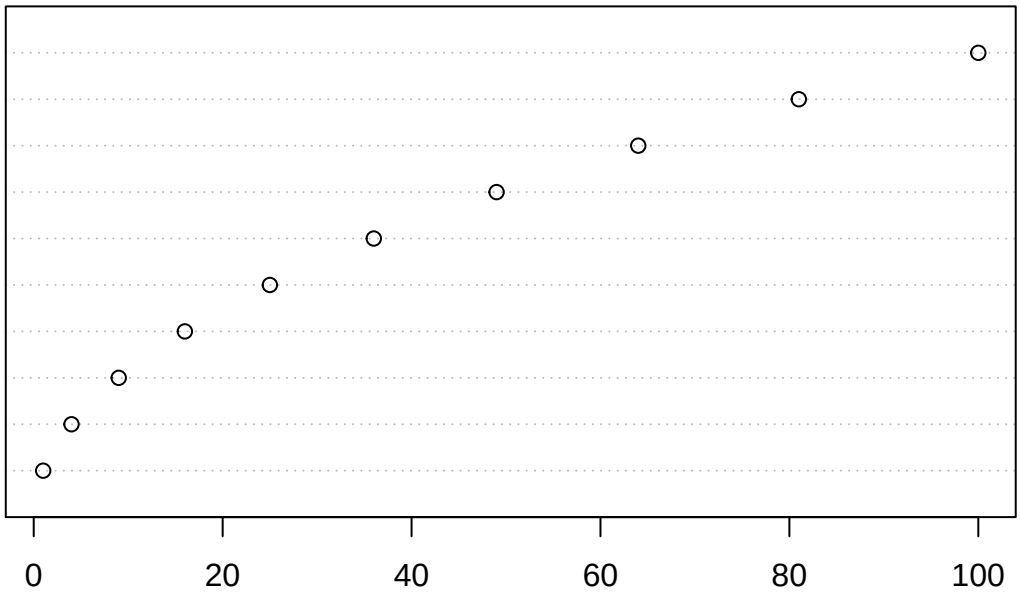

```
barplot(num_vec)
```



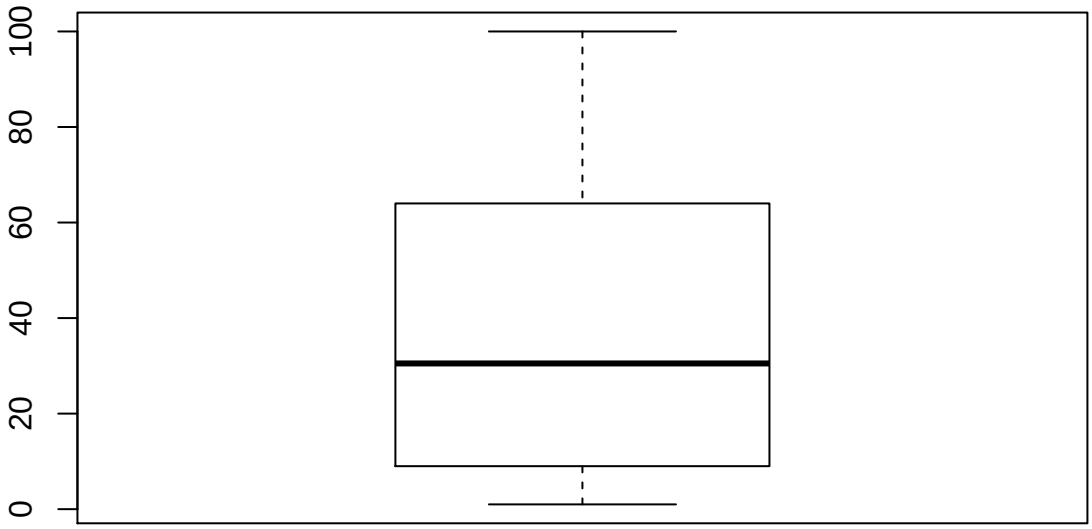

```
pie(1:3)
```



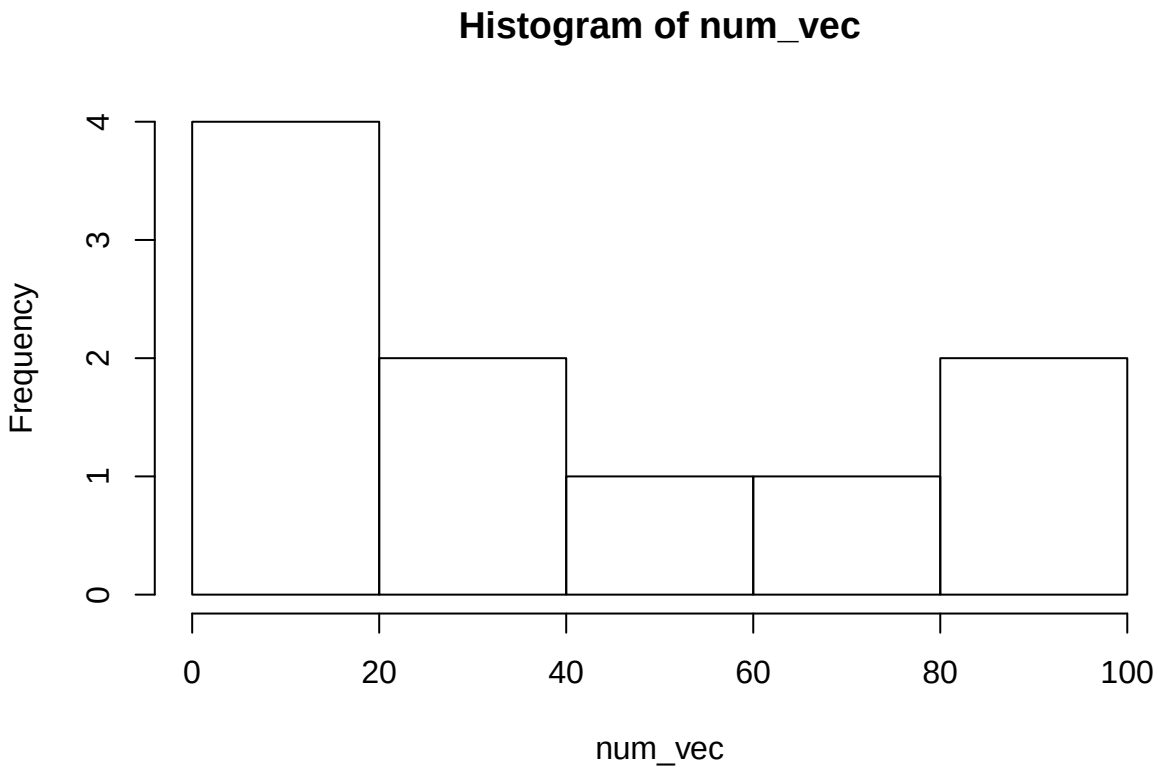

```
dotchart(num_vec)
```



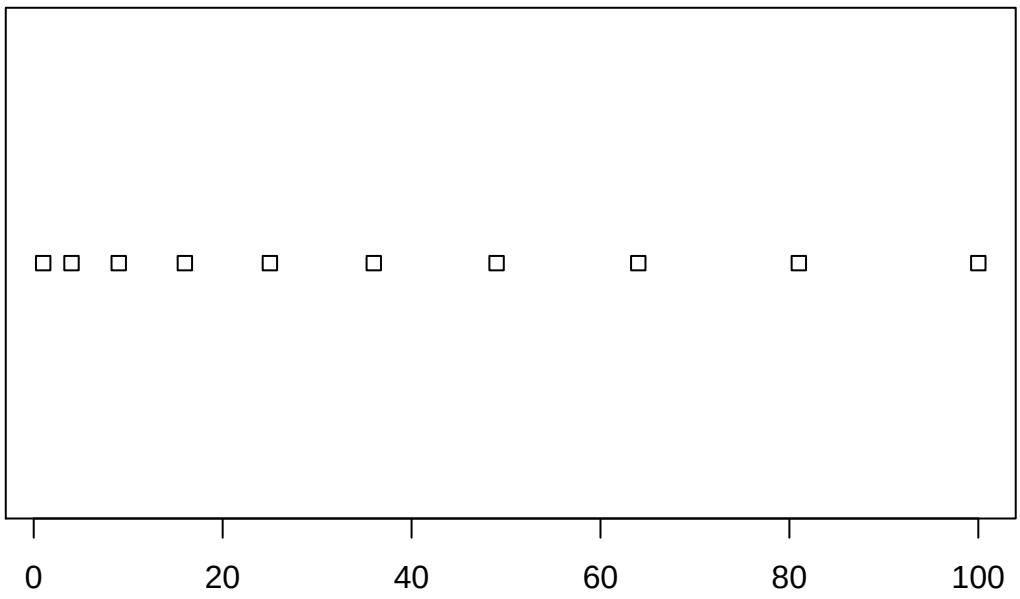
```r
boxplot(num_vec)
```




```r
hist(num_vec)
```




```r
stripchart(num_vec)
```




```
stem(num_vec)
```

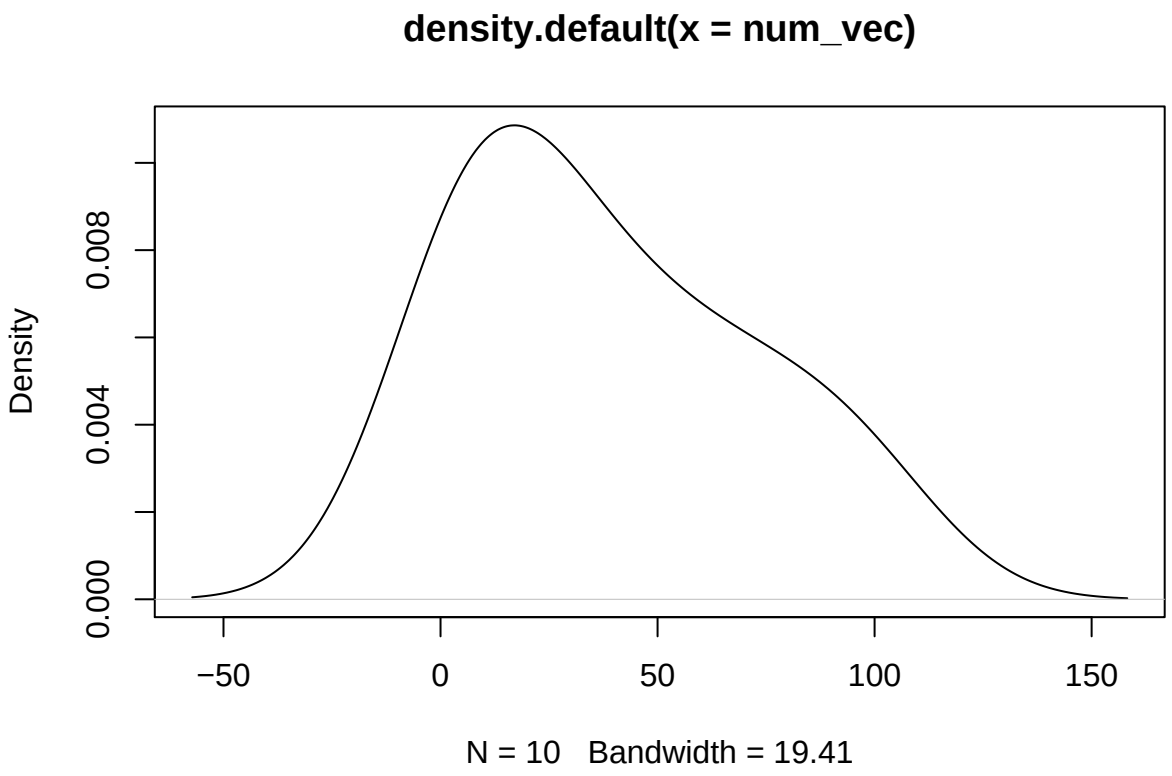
```
  The decimal point is 1 digit(s) to the right of the |

   0 | 1496
   2 | 56
   4 | 9
   6 | 4
   8 | 1
  10 | 0
```

Kernel Density Curve

We can pass a "density" object to plot() in order to get a density curve

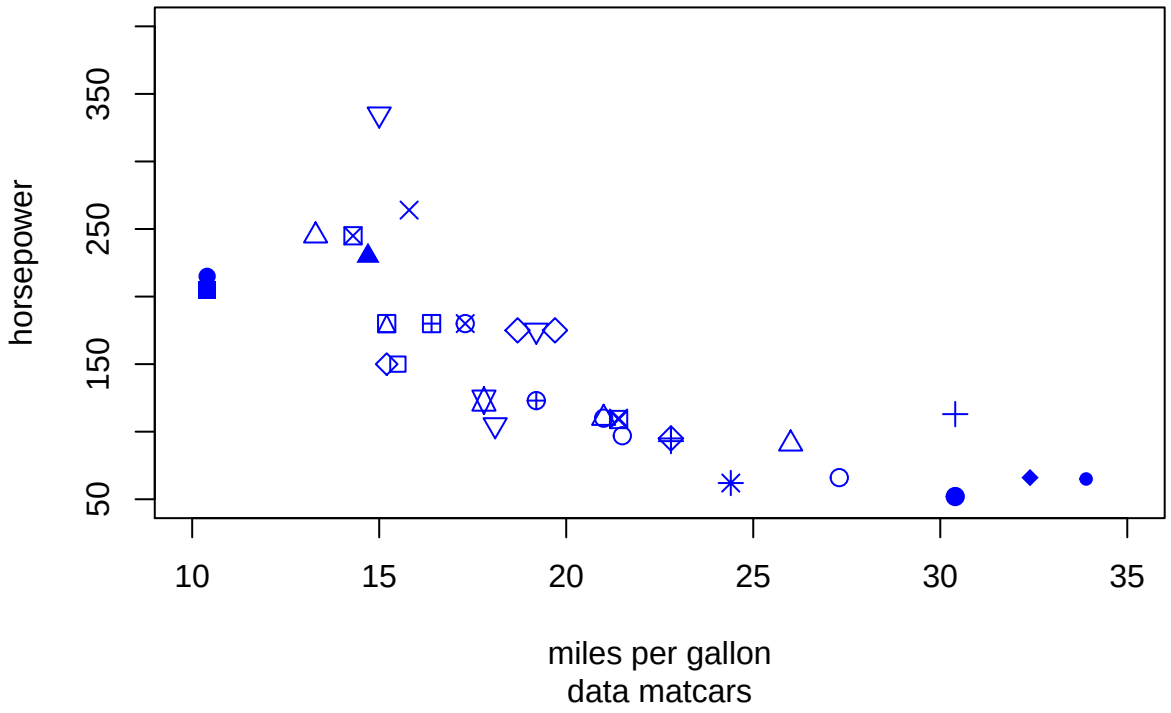
```
dens <- density(num_vec)
plot(dens)
```



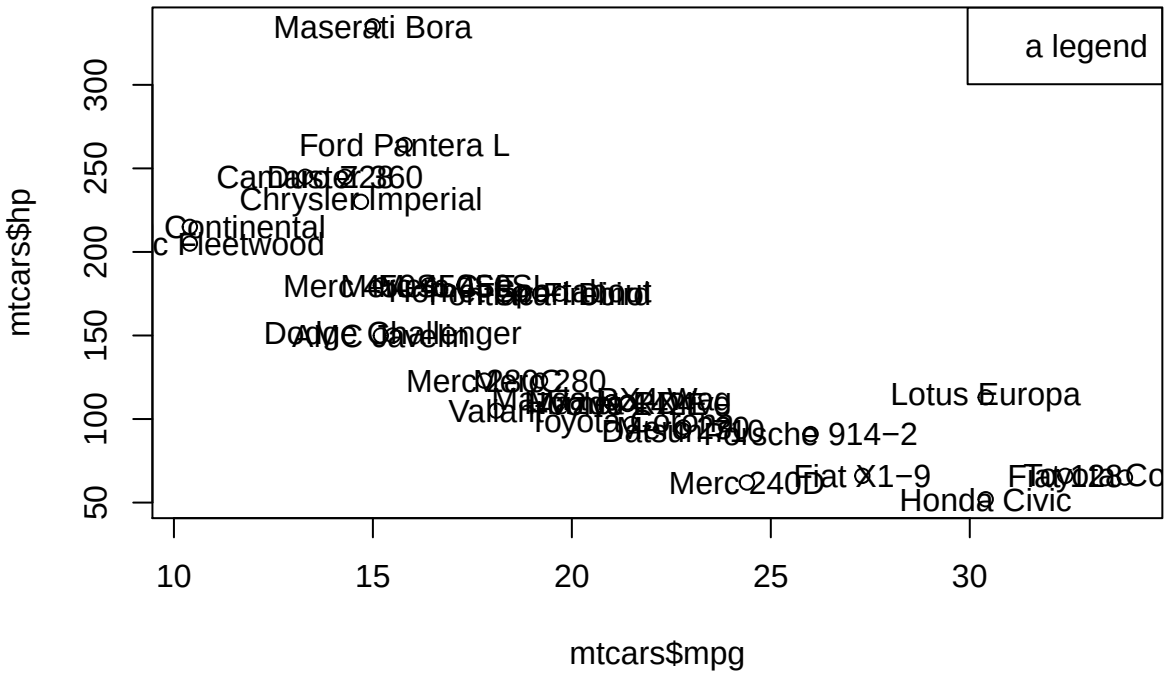

```
head(mtcars)
```

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
plot(mtcars$mpg, mtcars$hp,
     xlab = "miles per gallon", ylab = "horsepower", # axis labels
     main = "Simple Scatterplot", # title
     sub = "data matcars", # subtitle
     xlim = c(10, 35), ylim = c(50,400), # x & y coordinate ranges
     cex = 1.2, # character expansion
     pch = 1:25, # point character
     col = "blue" # color
)
```




```
plot(mtcars$mpg, mtcars$hp)
text(mtcars$mpg, mtcars$hp, labels = row.names(mtcars))
legend("topright", legend = "a legend")
title("Miles Per Gallon -vs- Horsepower")
```



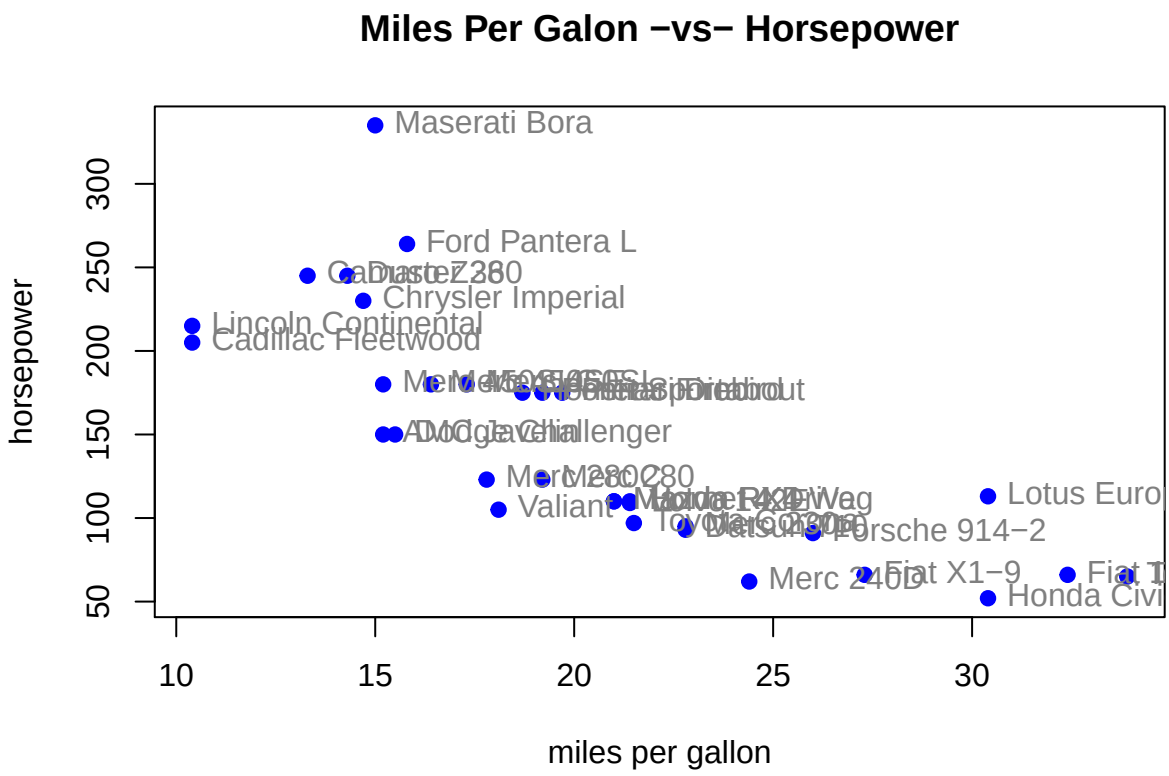

```
# simple scatter-plot
plot(mtcars$mpg, mtcars$hp, type = "n", xlab = "miles per gallon", ylab = "horsepower")
# grid lines
abline(v = seq(from = 10, to = 30, by = 5), col = gray) # vertical lines
```

```
Warning in int_abline(a = a, b = b, h = h, v = v, untf = untf, ...):
supplied color is neither numeric nor character
```

```
abline(h = seq(from = 50, to = 300, by = 50), col = gray) # horizontal linesfunct
```

```
Warning in int_abline(a = a, b = b, h = h, v = v, untf = untf, ...):
supplied color is neither numeric nor character
```

```
# plot points
points(mtcars$mpg, mtcars$hp, pch = 19, col = "blue")
# plot text
text(mtcars$mpg, mtcars$hp, labels = rownames(mtcars), pos = 4, col = "gray50")
# graphic title
title("Miles Per Galon -vs- Horsepower")
```


Miles Per Galon −vs− Horsepower
horsepower
miles per gallon
50
100
150
200
250
300
10
15
20
25
30
Maserati Bora
Ford Pantera L
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Valiant
Lotus Euro
Porsche 914−2
Merc 240D
Fiat X1−9
Honda Civi