randomwalk.R

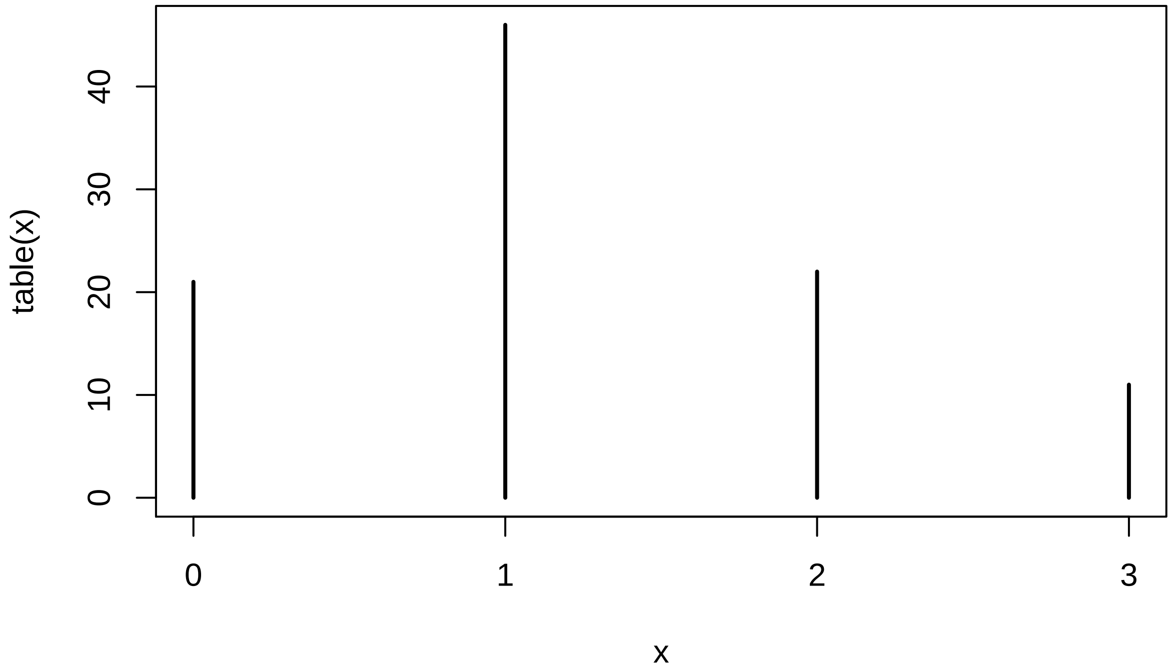
Tech

2021-05-20

```r
true_p <- 0.4

set.seed( 2487 )

x = rbinom( 10, 1, true_p )

typeof( x )
```

```
## [1] "integer"
```

```r
p_hat = mean( x )

n = 100
x = rbinom( n, size = 3, prob = true_p )

plot( table(x) )
```

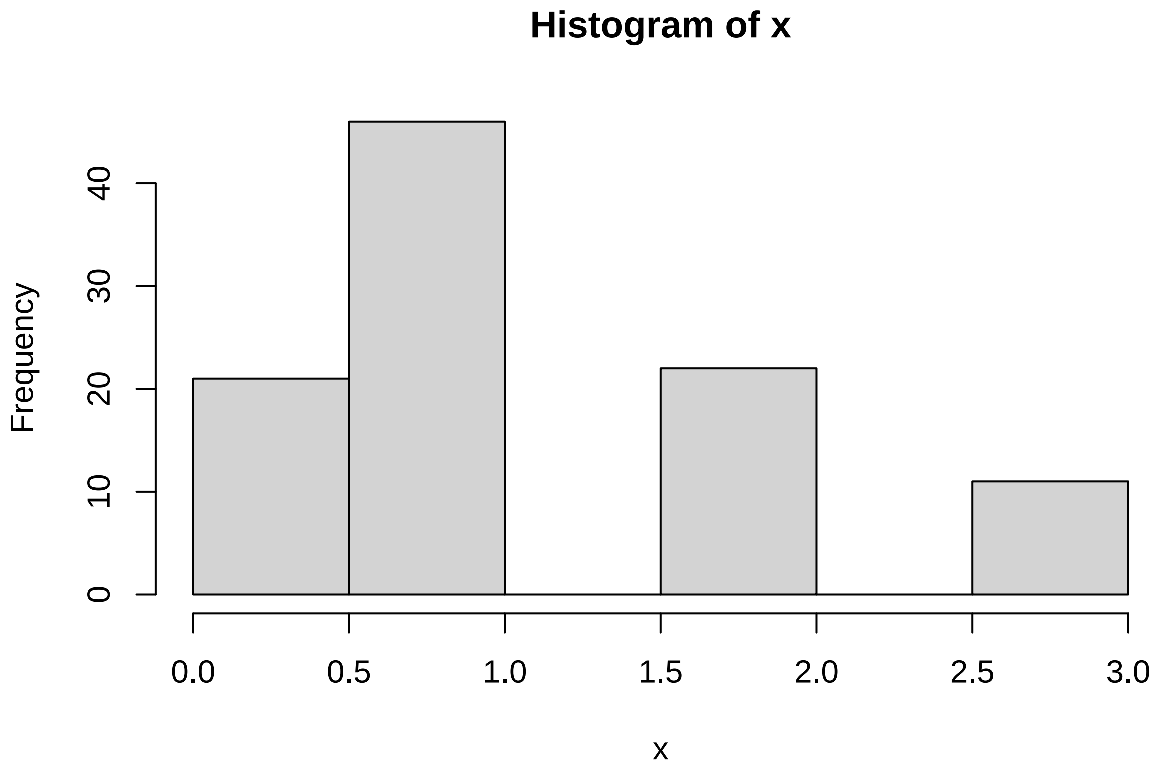


```
h = hist( x )

str( h )
```

```
## List of 6
##  $ breaks  : num [1:7] 0 0.5 1 1.5 2 2.5 3
##  $ counts  : int [1:6] 21 46 0 22 0 11
##  $ density : num [1:6] 0.42 0.92 0 0.44 0 0.22
##  $ mids    : num [1:6] 0.25 0.75 1.25 1.75 2.25 2.75
##  $ xname   : chr "x"
##  $ equidist: logi TRUE
##  - attr(*, "class")= chr "histogram"
```

```
library( ggplot2 )
```

```
## Warning: package 'ggplot2' was built under R version 4.0.3
```



```
x = 1:10
y <- runif( 10 )

y_without_last2 <- y[ -c( 9, 10 ) ]
y_without_last2
```

```
## [1] 0.7117063 0.8311779 0.3244556 0.5992146 0.1001617 0.9946698 0.1652718
## [8] 0.6055093
```

```
# Type coercion
z = c( 1, 2L, 3.14159, "hello" )

z[2] = NA
z2 = NA

is.na( z2 )
```

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

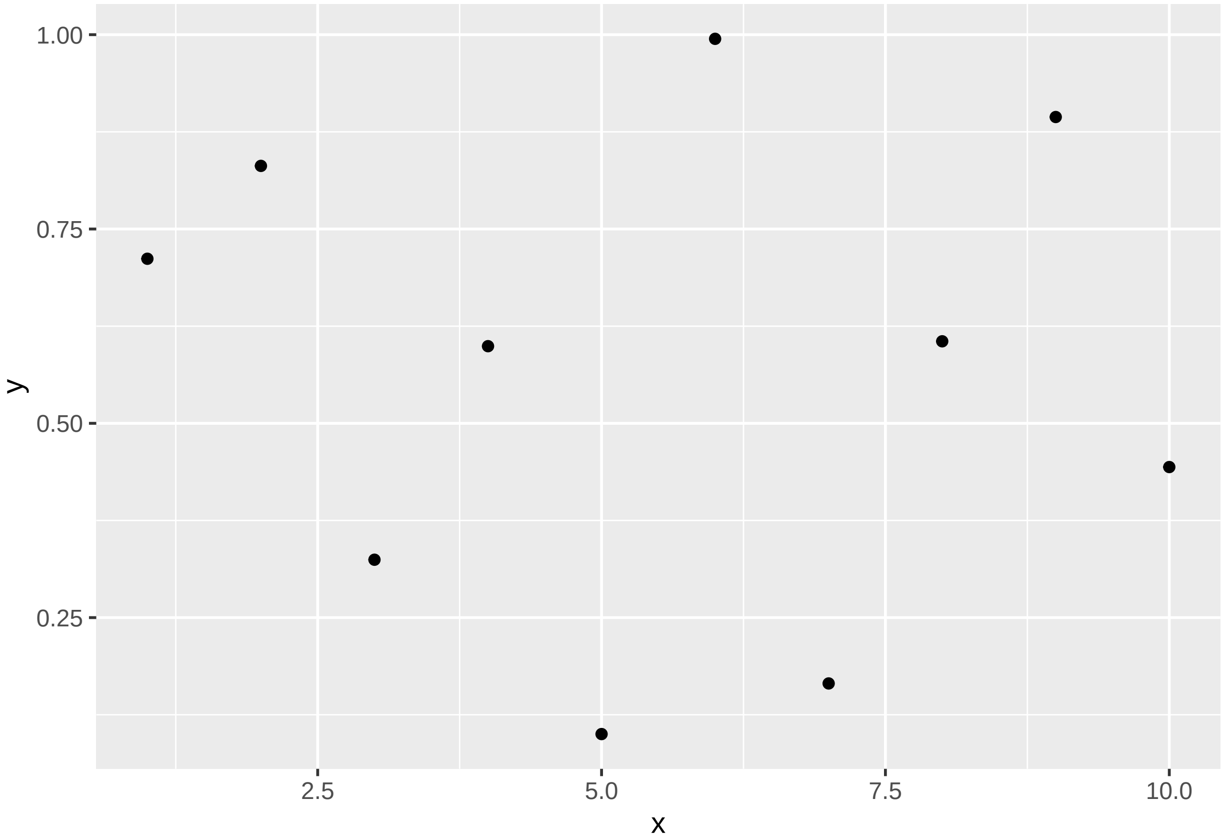
```
is.na( z )
```

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

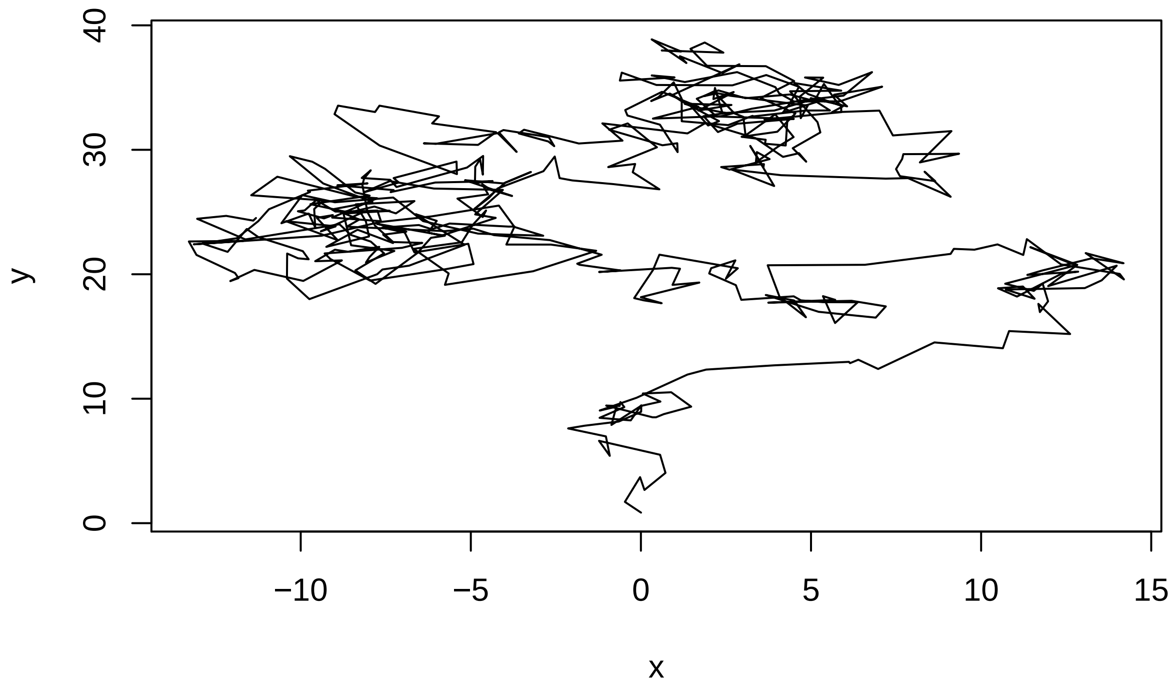
```
length( y_without_last2 )
```

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

```
qplot( x, y )
```



```
# Random stuff.
set.seed( 542893 )

# Random walk
random_walk <- function( n = 100 )
{
  x0 = rnorm( n )
  x = cumsum( x0 )

  y0 = rnorm( n )
  y <- cumsum( y0 )

  plot( x, y, type = "l" )
}

random_walk( 500 )
```



```
rm( list = ls() )

directions <- c( "north", "east", "south" , "west" )

step <- function( position, directions )
{
  if(direction == "north"){
    position$y = position$y + 1L
  } else if(direction == "east"){
    position$x = position$x + 1L
  } else if(direction == "south"){
    position$y = position$y - 1L
  } else if(direction == "west"){
    position$x = position$x - 1L
  }
  position
}

# Plot a single step
plot_step = function(from, to)
{
  arrows(from$x, from$y, to$x, to$y, length = 0.1)
}

# Run an entire simulation
#simulate = function(position = list(x = 0, y = 0),pause = 0.2, bounds = 10, nsteps = bounds * 10)
```

```
#{
#  b = c(-bounds, bounds)
#  plot(b, b, type = "n")
#  walk = sample(directions, nsteps, replace = TRUE)
#  to = position
#  for(d in walk){
#    from = to
#    to = step(from, d)
#    plot_step(from, to)
#    Sys.sleep(pause)
#  }
#  from
#}

#set.seed(4378)
#simulate()

#simulate()
```