R Notebook

Code

Load Libraries

Hide

Hide

```
library(GenSA)
library(ggplot2)
library(GA)
```

Hide

Hide

```
#Load Libraries
install.packages('GenSA')
```

```
trying URL 'https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.4/GenSA_1.1.7.tg
z'
Content type 'application/x-gzip' length 215011 bytes (209 KB)
==================================================
downloaded 209 KB


The downloaded binary packages are in
    /var/folders/1n/wm_v8jtd0d720rc2ymvkfhg80000gn/T//RtmpETl71f/downloaded_packages
```

Hide

Hide

```
install.packages('GA')
```

```
trying URL 'https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.4/GA_3.1.1.tgz'
Content type 'application/x-gzip' length 2178951 bytes (2.1 MB)
==================================================
downloaded 2.1 MB


The downloaded binary packages are in
    /var/folders/1n/wm_v8jtd0d720rc2ymvkfhg80000gn/T//RtmpETl71f/downloaded_packages
```

Hide

Hide

```
library('ggplot2')
```

```
package ‘ggplot2’ was built under R version 3.4.4
```

Hide

Hide

```
library('quantmod')
```

```
package ‘quantmod’ was built under R version 3.4.4Loading required package: xts
package ‘xts’ was built under R version 3.4.4Loading required package: zoo
package ‘zoo’ was built under R version 3.4.4
Attaching package: ‘zoo’

The following objects are masked from ‘package:base’:

    as.Date, as.Date.numeric

Loading required package: TTR
package ‘TTR’ was built under R version 3.4.4Version 0.4-0 included new data defaults
. See ?getSymbols.
Learn from a quantmod author: https://www.datacamp.com/courses/importing-and-managing
-financial-data-in-r
```

Hide

Hide

```
library('GA')
```

```
package ‘GA’ was built under R version 3.4.4Loading required package: foreach
package ‘foreach’ was built under R version 3.4.3Loading required package: iterators
package ‘iterators’ was built under R version 3.4.4  ____    _
 / ___|  / \     Genetic
| |  _  / _ \    Algorithms
| |_| |/ ___ \
 \____/_/   \_\  version 3.1.1
Type 'citation("GA")' for citing this R package in publications.
```

Hide

Hide

```
library('GenSA')
```

```
package ‘GenSA’ was built under R version 3.4.3
```

Hide

Hide

```
library('PerformanceAnalytics')
```

```
package 'PerformanceAnalytics' was built under R version 3.4.3
Package PerformanceAnalytics (1.5.2) loaded.
Copyright (c) 2004-2018 Peter Carl and Brian G. Peterson, GPL-2 | GPL-3
https://github.com/braverock/PerformanceAnalytics


Attaching package: 'PerformanceAnalytics'

The following object is masked from 'package:graphics':

    legend
```

Hide

Hide

```
#Feeding real-time stock data into the system
tickers <- c("GM", "AMZN", "BMWYY", "LVMUY", "KO")
getSymbols(tickers, from = "2010-12-01", to = "2018-11-05")
```

```
[1] "GM"    "AMZN"  "BMWYY" "LVMUY" "KO"
```

Hide

Hide

```
P <- NULL
for(ticker in tickers) {
  tmp <- Cl(to.monthly(eval(parse(text = ticker))))
  P <- cbind(P, tmp)
}
colnames(P) <- tickers
R <- diff(log(P))
R <- R[-1,]
mu <- colMeans(R)
sigma <- cov(R)
library("PerformanceAnalytics")
pContribCVaR <- ES(weights = rep(0.2, 5), method = "gaussian", portfolio_method = "co
mponent", mu = mu, sigma = sigma)$pct_contrib_ES
obj <- function(w) {
  fn.call <<- fn.call + 1
  if (sum(w) == 0) { w <- w + 1e-2 }
  w <- w / sum(w)
  CVaR <- ES(weights = w, method = "gaussian", portfolio_method = "component", mu = m
u, sigma = sigma)
  tmp1 <- CVaR$ES
  tmp2 <- max(CVaR$pct_contrib_ES - 0.225, 0)
  out <- tmp1 - 1e+3 * tmp2
  return(out)
}
obj1 <- function(w) {
  fn.call <<- fn.call + 1
  if (sum(w) == 0) { w <- w + 1e-2 }
  w <- w / sum(w)
  CVaR <- ES(weights = w, method = "gaussian", portfolio_method = "component", mu = m
u, sigma = sigma)
  tmp1 <- CVaR$ES
  tmp2 <- max(CVaR$pct_contrib_ES - 0.225, 0)
  out1 <- tmp1 + 1e+3 * tmp2
  return(out1)
}
```

Hide

Hide

```
# Solution is the weights assigned to each of the stocks for optimum portfolio
set.seed(1234)
fn.call <<- 0
gap <- ga(type = "real-valued",fitness=obj,lower=rep(0,5),upper=rep(1,5), popSize = 5
0,maxiter = 150, pcrossover = 0.75, pmutation = 0.1)
```

```
GA | iter = 1 | Mean = -220.05129 | Best =  -54.86347
GA | iter = 2 | Mean = -170.70494 | Best =  -40.82414
GA | iter = 3 | Mean = -138.64441 | Best =  -40.82414
GA | iter = 4 | Mean = -99.67271 | Best = -26.04375
```

```
GA | iter = 5 | Mean = -71.09705 | Best = -26.04375
GA | iter = 6 | Mean = -62.53484 | Best = -25.32443
GA | iter = 7 | Mean = -54.24932 | Best = -23.35230
GA | iter = 8 | Mean = -47.33546 | Best = -20.77581
GA | iter = 9 | Mean = -40.64160 | Best = -15.90522
GA | iter = 10 | Mean = -40.88976 | Best = -15.90522
GA | iter = 11 | Mean = -44.20312 | Best = -15.90522
GA | iter = 12 | Mean = -42.27071 | Best = -15.90522
GA | iter = 13 | Mean = -34.72313 | Best = -15.90522
GA | iter = 14 | Mean = -37.48621 | Best = -15.90522
GA | iter = 15 | Mean = -29.74813 | Best = -15.90522
GA | iter = 16 | Mean = -30.88109 | Best = -15.90522
GA | iter = 17 | Mean = -30.75004 | Best = -15.90522
GA | iter = 18 | Mean = -31.73937 | Best = -15.90522
GA | iter = 19 | Mean = -27.90865 | Best = -15.90522
GA | iter = 20 | Mean = -29.06460 | Best = -15.52273
GA | iter = 21 | Mean = -24.26602 | Best = -13.55976
GA | iter = 22 | Mean = -25.05263 | Best = -13.55976
GA | iter = 23 | Mean = -26.83828 | Best = -12.81688
GA | iter = 24 | Mean = -27.98375 | Best = -12.81688
GA | iter = 25 | Mean = -35.16641 | Best = -12.14503
GA | iter = 26 | Mean = -26.79523 | Best = -12.14503
GA | iter = 27 | Mean = -24.70203 | Best = -12.14503
GA | iter = 28 | Mean = -27.97698 | Best = -10.46231
GA | iter = 29 | Mean = -25.02368 | Best = -10.46231
GA | iter = 30 | Mean = -20.02017 | Best = -10.46231
GA | iter = 31 | Mean = -25.72428 | Best = -10.46231
GA | iter = 32 | Mean = -18.75614 | Best = -10.46231
GA | iter = 33 | Mean = -25.27058 | Best = -10.46231
GA | iter = 34 | Mean = -23.26855 | Best = -10.46231
GA | iter = 35 | Mean = -19.99589 | Best = -10.38601
GA | iter = 36 | Mean = -20.52644 | Best = -10.38601
GA | iter = 37 | Mean = -20.15578 | Best = -10.38601
GA | iter = 38 | Mean = -18.09100 | Best = -10.38601
GA | iter = 39 | Mean = -19.92437 | Best = -10.38601
GA | iter = 40 | Mean = -19.14357 | Best = -10.38601
GA | iter = 41 | Mean = -21.766103 | Best =  -8.996975
GA | iter = 42 | Mean = -15.114655 | Best =  -8.996975
GA | iter = 43 | Mean = -18.231048 | Best =  -8.996975
GA | iter = 44 | Mean = -24.000764 | Best =  -8.996975
GA | iter = 45 | Mean = -20.537755 | Best =  -8.996975
GA | iter = 46 | Mean = -22.302709 | Best =  -8.996975
GA | iter = 47 | Mean = -17.675847 | Best =  -8.996975
GA | iter = 48 | Mean = -12.735995 | Best =  -8.922921
GA | iter = 49 | Mean = -15.140967 | Best =  -8.922921
GA | iter = 50 | Mean = -18.053713 | Best =  -8.676531
GA | iter = 51 | Mean = -13.803133 | Best =  -8.676531
GA | iter = 52 | Mean = -17.555392 | Best =  -8.676531
GA | iter = 53 | Mean = -21.242481 | Best =  -8.676531
GA | iter = 54 | Mean = -19.464074 | Best =  -8.676531
```

```
GA | iter = 55 | Mean = -14.433340 | Best =  -7.847027
GA | iter = 56 | Mean = -18.597566 | Best =  -7.847027
GA | iter = 57 | Mean = -17.332378 | Best =  -7.847027
GA | iter = 58 | Mean = -15.877485 | Best =  -7.847027
GA | iter = 59 | Mean = -18.027928 | Best =  -7.847027
GA | iter = 60 | Mean = -15.076578 | Best =  -7.847027
GA | iter = 61 | Mean = -17.474123 | Best =  -7.847027
GA | iter = 62 | Mean = -20.021801 | Best =  -7.847027
GA | iter = 63 | Mean = -14.415924 | Best =  -7.847027
GA | iter = 64 | Mean = -13.534898 | Best =  -7.847027
GA | iter = 65 | Mean = -15.625316 | Best =  -7.847027
GA | iter = 66 | Mean = -11.635145 | Best =  -7.091211
GA | iter = 67 | Mean = -15.934945 | Best =  -7.091211
GA | iter = 68 | Mean = -18.717324 | Best =  -7.091211
GA | iter = 69 | Mean = -14.922012 | Best =  -7.091211
GA | iter = 70 | Mean = -15.473479 | Best =  -7.091211
GA | iter = 71 | Mean = -11.909650 | Best =  -7.091211
GA | iter = 72 | Mean = -18.261884 | Best =  -5.940762
GA | iter = 73 | Mean = -17.904952 | Best =  -5.940762
GA | iter = 74 | Mean = -15.834922 | Best =  -5.940762
GA | iter = 75 | Mean = -16.659397 | Best =  -5.940762
GA | iter = 76 | Mean = -14.190657 | Best =  -5.940762
GA | iter = 77 | Mean = -16.883429 | Best =  -5.940762
GA | iter = 78 | Mean = -19.875008 | Best =  -5.940762
GA | iter = 79 | Mean = -17.834207 | Best =  -5.940762
GA | iter = 80 | Mean = -16.858031 | Best =  -5.940762
GA | iter = 81 | Mean = -16.780854 | Best =  -5.940762
GA | iter = 82 | Mean = -17.041730 | Best =  -5.940762
GA | iter = 83 | Mean = -18.694801 | Best =  -5.940762
GA | iter = 84 | Mean = -18.525042 | Best =  -5.940762
GA | iter = 85 | Mean = -19.781231 | Best =  -5.940762
GA | iter = 86 | Mean = -20.518230 | Best =  -5.940762
GA | iter = 87 | Mean = -16.892118 | Best =  -5.742114
GA | iter = 88 | Mean = -16.259743 | Best =  -5.742114
GA | iter = 89 | Mean = -20.150040 | Best =  -5.742114
GA | iter = 90 | Mean = -12.582836 | Best =  -5.579834
GA | iter = 91 | Mean = -16.375683 | Best =  -5.579834
GA | iter = 92 | Mean = -12.016939 | Best =  -5.579834
GA | iter = 93 | Mean = -13.189190 | Best =  -5.579834
GA | iter = 94 | Mean = -16.504596 | Best =  -5.579834
GA | iter = 95 | Mean = -16.025251 | Best =  -5.579834
GA | iter = 96 | Mean = -17.644222 | Best =  -5.490242
GA | iter = 97 | Mean = -15.152526 | Best =  -5.490242
GA | iter = 98 | Mean = -17.879772 | Best =  -5.490242
GA | iter = 99 | Mean = -11.743010 | Best =  -5.490242
GA | iter = 100 | Mean = -13.891530 | Best =  -5.490242
GA | iter = 101 | Mean = -18.751676 | Best =  -5.490242
GA | iter = 102 | Mean = -16.417500 | Best =  -5.490242
GA | iter = 103 | Mean = -9.733724 | Best = -5.490242
GA | iter = 104 | Mean = -13.554079 | Best =  -5.490242
```

```
GA | iter = 105 | Mean = -15.255389 | Best =  -4.242993
GA | iter = 106 | Mean = -14.821367 | Best =  -4.232044
GA | iter = 107 | Mean = -15.771833 | Best =  -3.713437
GA | iter = 108 | Mean = -9.482741 | Best = -3.713437
GA | iter = 109 | Mean = -12.761215 | Best =  -3.113008
GA | iter = 110 | Mean = -15.278978 | Best =  -3.113008
GA | iter = 111 | Mean = -13.954098 | Best =  -3.113008
GA | iter = 112 | Mean = -15.791069 | Best =  -3.113008
GA | iter = 113 | Mean = -13.558644 | Best =  -3.113008
GA | iter = 114 | Mean = -16.873784 | Best =  -3.113008
GA | iter = 115 | Mean = -18.296013 | Best =  -3.113008
GA | iter = 116 | Mean = -15.009758 | Best =  -3.113008
GA | iter = 117 | Mean = -16.679559 | Best =  -3.113008
GA | iter = 118 | Mean = -12.665602 | Best =  -3.113008
GA | iter = 119 | Mean = -14.882788 | Best =  -3.113008
GA | iter = 120 | Mean = -12.087280 | Best =  -1.244082
GA | iter = 121 | Mean = -11.268033 | Best =  -1.244082
GA | iter = 122 | Mean = -10.026164 | Best =  -1.244082
GA | iter = 123 | Mean = -12.716964 | Best =  -1.244082
GA | iter = 124 | Mean = -11.916685 | Best =  -1.244082
GA | iter = 125 | Mean = -18.537473 | Best =  -1.244082
GA | iter = 126 | Mean = -9.977300 | Best = -1.244082
GA | iter = 127 | Mean = -11.867164 | Best =  -1.244082
GA | iter = 128 | Mean = -10.740783 | Best =  -1.244082
GA | iter = 129 | Mean = -11.401530 | Best =  -1.244082
GA | iter = 130 | Mean = -13.465393 | Best =  -1.244082
GA | iter = 131 | Mean = -8.958469 | Best = -1.244082
GA | iter = 132 | Mean = -10.336899 | Best =  -1.244082
GA | iter = 133 | Mean = -7.641691 | Best = -1.210087
GA | iter = 134 | Mean = -13.268918 | Best =  -1.210087
GA | iter = 135 | Mean = -13.003052 | Best =  -1.210087
GA | iter = 136 | Mean = -17.1203751 | Best =  -0.9408666
GA | iter = 137 | Mean = -9.9292040 | Best = -0.9408666
GA | iter = 138 | Mean = -14.7808807 | Best =  -0.9408666
GA | iter = 139 | Mean = -6.9451916 | Best = -0.9408666
GA | iter = 140 | Mean = -8.5888664 | Best = -0.9408666
GA | iter = 141 | Mean = -12.0541364 | Best =  -0.9408666
GA | iter = 142 | Mean = -8.6190551 | Best = -0.9408666
GA | iter = 143 | Mean = -6.5696511 | Best = -0.8855573
GA | iter = 144 | Mean = -8.0034805 | Best = -0.8855573
GA | iter = 145 | Mean = -7.6201253 | Best = -0.7273805
GA | iter = 146 | Mean = -11.5436991 | Best =  -0.7273805
GA | iter = 147 | Mean = -10.0673968 | Best =  -0.6083845
GA | iter = 148 | Mean = -8.6495918 | Best = -0.6083845
GA | iter = 149 | Mean = -9.4843359 | Best = -0.6083845
GA | iter = 150 | Mean = -6.1852469 | Best = -0.6083845
```

Hide

Hide

```
nsol <- gap@solution
nsol <- nsol / sum(nsol)
fn.call.gap <- fn.call
nsol
```

```
            x1        x2       x3        x4        x5
[1,] 0.1744992 0.2241486 0.135447 0.1770054 0.2888998
```

Hide

Hide

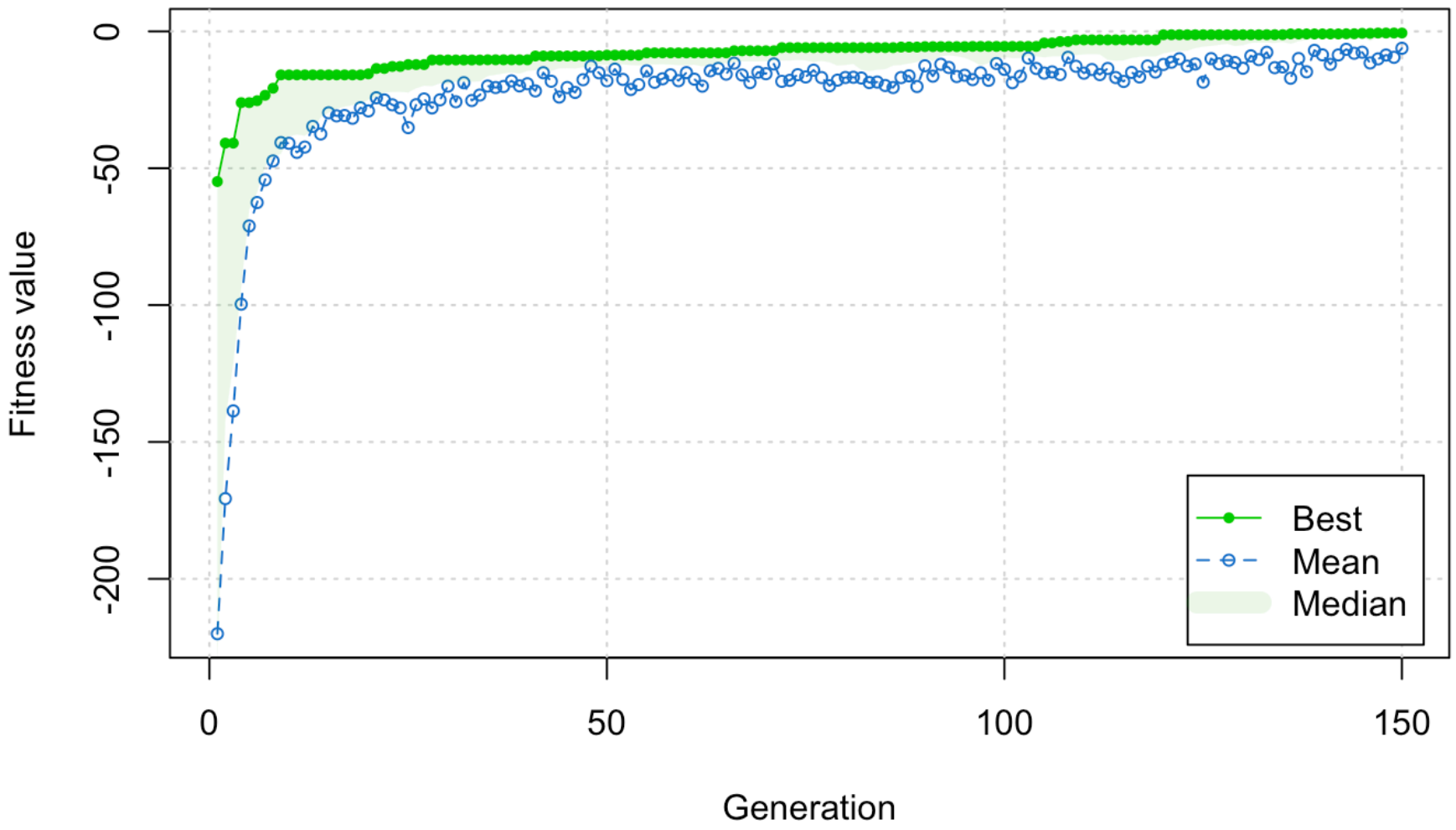
```
summary(gap)
```

```
── [1mGenetic Algorithm[22m ──────────────────

GA settings:
Type                  =  real-valued
Population size       =  50
Number of generations =  150
Elitism               =  2
Crossover probability =  0.75
Mutation probability  =  0.1
Search domain =
      x1 x2 x3 x4 x5
lower  0  0  0  0  0
upper  1  1  1  1  1

GA results:
Iterations             = 150
Fitness function value = -0.6083845
Solution =
            x1        x2        x3        x4        x5
[1,] 0.5943835 0.7635006 0.4613628 0.6029201 0.9840577
```

Hide

Hide

```
plot(gap)
```



Hide

Hide

```
#Applying GenSA to assign weights to the stocks for optimum portfolio using max objec
tive
set.seed(1234)
fn.call <<- 0
out.GenSA <- GenSA(fn = obj1, lower = rep(0, 5), upper = rep(1, 5), control = list(sm
ooth = FALSE, max.call = 3000))
fn.call.GenSA <- fn.call
out.GenSA$value
```

```
[1] 0.07357047
```

Hide

Hide

```
out.GenSA$counts
```

```
[1] 5515
```

Hide

Hide

```
cat("GenSA call functions", fn.call.GenSA, "times.\n")
```

```
GenSA call functions 5515 times.
```

Hide

Hide

```
wstar.GenSA <- out.GenSA$par
wstar.GenSA <- wstar.GenSA / sum(wstar.GenSA)
rbind(tickers, round(100 * wstar.GenSA, 2))
```

```
        [,1]    [,2]    [,3]    [,4]    [,5]
tickers "GM"    "AMZN"  "BMWYY" "LVMUY" "KO"
        "16.17" "19.62" "5.93"  "16.13" "42.15"
```

Hide

Hide

```
100 * (sum(wstar.GenSA * mu) - mean(mu))
```

```
[1] 0.03499312
```

Hide

Hide

```
wstar.GenSA
```

```
[1] 0.16169230 0.19624096 0.05927314 0.16133505 0.42145855
```

Hide

Hide

```
#comparing the weights as per both the algorithms
b <- matrix(c(nsol, wstar.GenSA), nrow = 5, ncol = 2)
rownames(b) = c("GM", "AMZN", "BMWYY", "LVMUY", "KO")
colnames(b) = c("GAPortfolio", "GenSAPortfolio")
b
```

```
      GAPortfolio GenSAPortfolio
GM      0.1744992     0.16169230
AMZN    0.2241486     0.19624096
BMWYY   0.1354470     0.05927314
LVMUY   0.1770054     0.16133505
KO      0.2888998     0.42145855
```

Hide

Hide

```
TickerSymbol <- (c("GM", "AMZN", "BMWYY", "LVMUY", "KO","_______________","TOTAL"))
Company <- (c("General Motors","Amazon","BMW","Louis Vuitton","Coca-Cola","_____________","""))
GA_percent  <- (c(17,22,14,18,29,"_______________ ",100))
GenSA_percent <- (c(16,20,6,16,42,"_______________ ",100))
TData <- data.frame(TickerSymbol, Company, GA_percent, GenSA_percent)
TData
```

TickerSymbol <fctr>	**Company** <fctr>	**GA_percent** <fctr>	**GenSA_percent** <fctr>
GM	General Motors	17	16
AMZN	Amazon	22	20
BMWYY	BMW	14	6
LVMUY	Louis Vuitton	18	16
KO	Coca-Cola	29	42
_____	_____	_____	_____
TOTAL		100	100

7 rows

Hide

Hide

```
#Question number 2
x <- c(1, 3, 7, 9, 5, 2, 4, 8, 6, 22, 11, 33)
y <- c(50, 60, 70, 80, 90, 100, 66, 18, 32, 28, 69, 44)
a <- sum((y - mean(y)) ^ 2)
a1 <- sum((x - mean(x)) * (y - mean(y)))
a2 <- sum((x - mean(x)) ^ 2)
b1 <- a1/a2
b0 <- mean(y) - b1 * mean(x)
print(b1)
```

```
[1] -1.074035
```

Hide

Hide

```
print(b0)
```

```
[1] 68.85149
```

Hide

Hide

```
linear_model <- lm(y ~ x)
summary(linear_model)
```

```

Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-42.259 -17.361   5.056  14.176  33.297

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  68.8515    10.1362   6.793 4.78e-05 ***
x            -1.0740     0.7893  -1.361    0.203
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 24.36 on 10 degrees of freedom
Multiple R-squared:  0.1562,    Adjusted R-squared:  0.07186
F-statistic: 1.852 on 1 and 10 DF,  p-value: 0.2035
```

Hide

Hide

```
obj <- function (r) {
    fn <- function (b0, b1) {
        return (sum(y - (b0 + b1 * x)) ^ 2)
    }
    return (fn(r[1], r[2]))
}
ubound <- c(2, 1)
lbound <- c(0, 0)
```

Hide

Hide

```
#GA
ga_search <- ga(
    type = "real-valued",
    fitness = function (x) -obj(x),
    lower = lbound,
    upper = ubound,
    popSize = 50
)
```

```
GA | iter = 1 | Mean = -403892.1 | Best = -335421.3
GA | iter = 2 | Mean = -377150.0 | Best = -333919.3
GA | iter = 3 | Mean = -370129.6 | Best = -333919.3
GA | iter = 4 | Mean = -362864.2 | Best = -333919.3
GA | iter = 5 | Mean = -362282.5 | Best = -333919.3
GA | iter = 6 | Mean = -363824.0 | Best = -333919.3
GA | iter = 7 | Mean = -357286.4 | Best = -333919.3
GA | iter = 8 | Mean = -353436.8 | Best = -333919.3
GA | iter = 9 | Mean = -354898.5 | Best = -333919.3
GA | iter = 10 | Mean = -354040.8 | Best = -333919.3
GA | iter = 11 | Mean = -351485.2 | Best = -333919.3
GA | iter = 12 | Mean = -351565.2 | Best = -333919.3
GA | iter = 13 | Mean = -356063.5 | Best = -333919.3
GA | iter = 14 | Mean = -347128.4 | Best = -333919.3
GA | iter = 15 | Mean = -349145.6 | Best = -333919.3
GA | iter = 16 | Mean = -343803.0 | Best = -333919.3
GA | iter = 17 | Mean = -347402.4 | Best = -333919.3
GA | iter = 18 | Mean = -347163.9 | Best = -333919.3
GA | iter = 19 | Mean = -342834.7 | Best = -332281.4
GA | iter = 20 | Mean = -340475.9 | Best = -332281.4
GA | iter = 21 | Mean = -342022.8 | Best = -332281.4
GA | iter = 22 | Mean = -340873.2 | Best = -332281.4
GA | iter = 23 | Mean = -342108.1 | Best = -332281.4
GA | iter = 24 | Mean = -344330.4 | Best = -332281.4
GA | iter = 25 | Mean = -343726.3 | Best = -332281.4
GA | iter = 26 | Mean = -341465.1 | Best = -332281.4
GA | iter = 27 | Mean = -341483.0 | Best = -332281.4
GA | iter = 28 | Mean = -338121.9 | Best = -332281.4
GA | iter = 29 | Mean = -341453.9 | Best = -332281.4
GA | iter = 30 | Mean = -339789.8 | Best = -332281.4
GA | iter = 31 | Mean = -342868.2 | Best = -332281.4
GA | iter = 32 | Mean = -343080.4 | Best = -332281.4
GA | iter = 33 | Mean = -346534.3 | Best = -332281.4
GA | iter = 34 | Mean = -340127.2 | Best = -332281.4
GA | iter = 35 | Mean = -339756.0 | Best = -332281.4
GA | iter = 36 | Mean = -338848.3 | Best = -332281.4
GA | iter = 37 | Mean = -336189.7 | Best = -330283.5
GA | iter = 38 | Mean = -337493.3 | Best = -330283.5
```

```
GA | iter = 39 | Mean = -337052.4 | Best = -330283.5
GA | iter = 40 | Mean = -341067.8 | Best = -330283.5
GA | iter = 41 | Mean = -338055.6 | Best = -330283.5
GA | iter = 42 | Mean = -338749.9 | Best = -330283.5
GA | iter = 43 | Mean = -339646.8 | Best = -330283.5
GA | iter = 44 | Mean = -340985.4 | Best = -330283.5
GA | iter = 45 | Mean = -337562.0 | Best = -330283.5
GA | iter = 46 | Mean = -338880.6 | Best = -330283.5
GA | iter = 47 | Mean = -344157.3 | Best = -330283.5
GA | iter = 48 | Mean = -343735.9 | Best = -330283.5
GA | iter = 49 | Mean = -338466.7 | Best = -330283.5
GA | iter = 50 | Mean = -335228.3 | Best = -330283.5
GA | iter = 51 | Mean = -335926.5 | Best = -330283.5
GA | iter = 52 | Mean = -334966.7 | Best = -330283.5
GA | iter = 53 | Mean = -334931.3 | Best = -330283.5
GA | iter = 54 | Mean = -343858.1 | Best = -330283.5
GA | iter = 55 | Mean = -341752.0 | Best = -330283.5
GA | iter = 56 | Mean = -339282.8 | Best = -330283.5
GA | iter = 57 | Mean = -340306.3 | Best = -330283.5
GA | iter = 58 | Mean = -344794.6 | Best = -330283.5
GA | iter = 59 | Mean = -343859.1 | Best = -330283.5
GA | iter = 60 | Mean = -339531.1 | Best = -330283.5
GA | iter = 61 | Mean = -341775.4 | Best = -330283.5
GA | iter = 62 | Mean = -339617.2 | Best = -330283.5
GA | iter = 63 | Mean = -343398.9 | Best = -330283.5
GA | iter = 64 | Mean = -340969.4 | Best = -330283.5
GA | iter = 65 | Mean = -338236.5 | Best = -330283.5
GA | iter = 66 | Mean = -335959.0 | Best = -330283.5
GA | iter = 67 | Mean = -334600.6 | Best = -330283.5
GA | iter = 68 | Mean = -335563.8 | Best = -330283.5
GA | iter = 69 | Mean = -339584.7 | Best = -330283.5
GA | iter = 70 | Mean = -338229.7 | Best = -330283.5
GA | iter = 71 | Mean = -339000.0 | Best = -330283.5
GA | iter = 72 | Mean = -341541.7 | Best = -330283.5
GA | iter = 73 | Mean = -340092.1 | Best = -330283.5
GA | iter = 74 | Mean = -340178.6 | Best = -329092.6
GA | iter = 75 | Mean = -334147.4 | Best = -329092.6
GA | iter = 76 | Mean = -333630.5 | Best = -329092.6
GA | iter = 77 | Mean = -335727.1 | Best = -329092.6
GA | iter = 78 | Mean = -337175.8 | Best = -329092.6
GA | iter = 79 | Mean = -338815.5 | Best = -329092.6
GA | iter = 80 | Mean = -337000.0 | Best = -329092.6
GA | iter = 81 | Mean = -337607.0 | Best = -329092.6
GA | iter = 82 | Mean = -334785.7 | Best = -329047.1
GA | iter = 83 | Mean = -334470.7 | Best = -329047.1
GA | iter = 84 | Mean = -335528.8 | Best = -329047.1
GA | iter = 85 | Mean = -331863.6 | Best = -329011.7
GA | iter = 86 | Mean = -339104.1 | Best = -328872.7
GA | iter = 87 | Mean = -332051.2 | Best = -328872.7
GA | iter = 88 | Mean = -333586.3 | Best = -328872.7
```

```
GA | iter = 89 | Mean = -335494.7 | Best = -328872.7
GA | iter = 90 | Mean = -336700.1 | Best = -328872.7
GA | iter = 91 | Mean = -333298.6 | Best = -328872.7
GA | iter = 92 | Mean = -333040.3 | Best = -328872.7
GA | iter = 93 | Mean = -334963.9 | Best = -328872.7
GA | iter = 94 | Mean = -337678.7 | Best = -328872.7
GA | iter = 95 | Mean = -341479.6 | Best = -328872.7
GA | iter = 96 | Mean = -335546.5 | Best = -328872.7
GA | iter = 97 | Mean = -337048.7 | Best = -328872.7
GA | iter = 98 | Mean = -336001.5 | Best = -328872.7
GA | iter = 99 | Mean = -337561.5 | Best = -328649.5
GA | iter = 100 | Mean = -337931.5 | Best = -328649.5
```

Hide

Hide

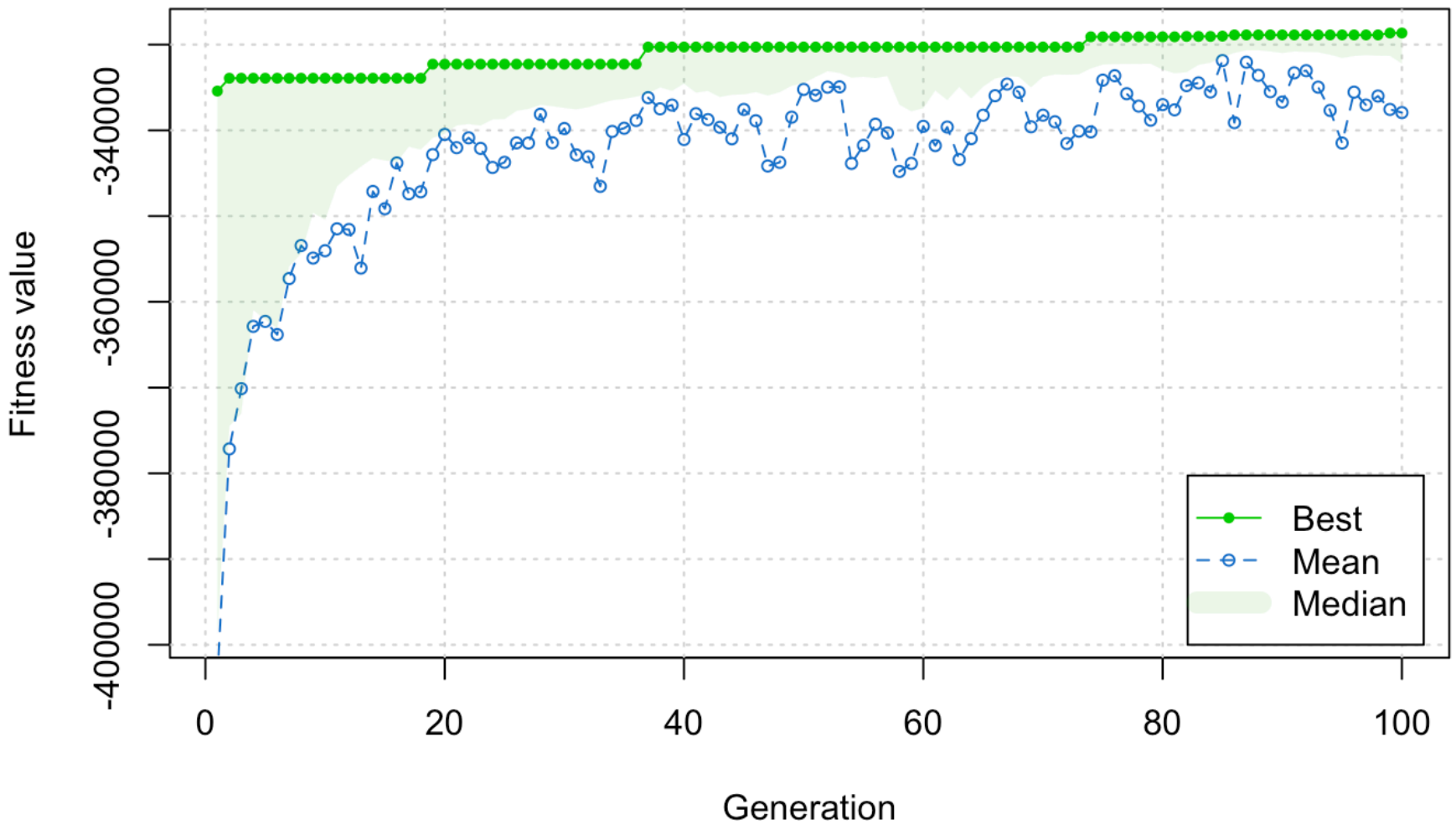
```
summary(ga_search)
```

```
── [1mGenetic Algorithm[22m ──────────────────

GA settings:
Type                  =  real-valued
Population size       =  50
Number of generations =  100
Elitism               =  2
Crossover probability =  0.8
Mutation probability  =  0.1
Search domain =
      x1 x2
lower  0  0
upper  2  1

GA results:
Iterations             = 100
Fitness function value = -328649.5
Solution =
          x1        x2
[1,] 1.93741 0.9952385
```

Hide

Hide

```
plot(ga_search)
```



Hide

```
#SA
par <- c(1, 0)
sa_search <- GenSA(
    par = par,
    lower = lbound,
    upper = ubound,
    fn = obj
)
```