Stats 102B HW4

Tori Wang

2023-05-17

Problem 1

Consider the following function:

$$f(x) = f(x_1, x_2) = 4x_1^2 + 2x_2^2 + 4x_1x_2 + 5x_1 + 2x_2$$

a)

We find the theoretical minimum:

$$\nabla f(x) = \begin{pmatrix} 8x_1 + 4x_2 + 5 \\ 4x_2 + 4x_1 + 2 \end{pmatrix}$$

We solve

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 8x_1 + 4x_2 + 5 \\ 4x_2 + 4x_1 + 2 \end{pmatrix}$$

to find:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -3/4 \\ 1/4 \end{pmatrix}$$

We check that this is indeed a minimum because the Hessian:

$$H = \begin{pmatrix} 8 & 4 \\ 4 & 4 \end{pmatrix}$$

Since H is positive definite, the function is convex and the minimum we found is the global minimum.

b)

```
library(ggplot2)
```

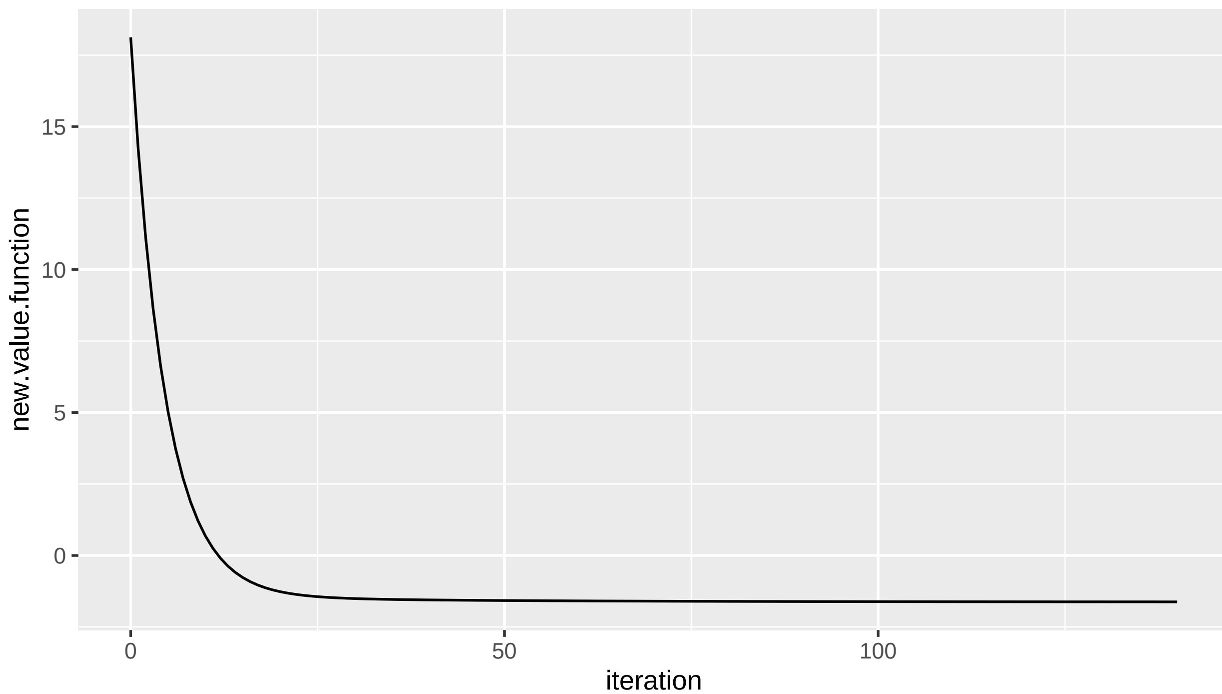
Use the following values for the tolerance parameter: 0.0001, 0.000001, 0.00000001 and for the step size 0.01, 0.05, 0.1.

```
# gradient descent for 2D differentiable functions

#define objective function to be optimized
objective.function = function(x) {
          4*x[1]^2 + 2*x[2]^2 + 4*x[1]*x[2] + 5*x[1] + 2*x[2]
    }
```

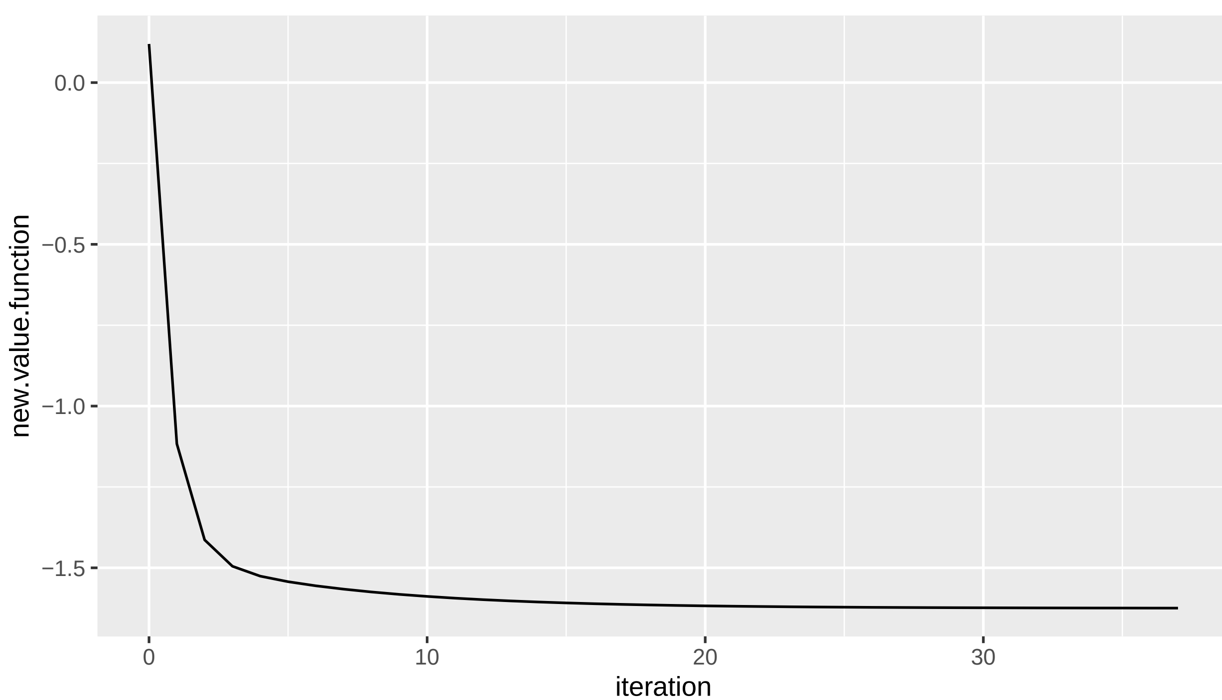
```r
# define the derivative for a 2D objective function
derivative = function(x) {
  c(8*x[1]+4*x[2] + 5, 4*x[2]+4*x[1] + 2)
}

# here we define the step size
mystepsize=c(0.01, 0.05, 0.1)
# here we define the tolerance of the convergence criterion
mytol = c(0.0001, 0.000001, 0.00000001)
# starting point
mystartpoint = c(1,2)

gradientDesc = function(obj.function, startpoint,
                        stepsize, conv_threshold, max_iter) {

  old.point = startpoint
  gradient = derivative(old.point)
  new.point = c(old.point[1] - stepsize*gradient[1],
                old.point[2] - stepsize*gradient[2])


  old.value.function = obj.function(new.point)

  converged = F
  iterations = 0

  while(converged == F) {
    ## Implement the gradient descent algorithm
    old.point = new.point
    ##print(old.point)

    gradient = derivative(old.point)
    new.point = c(old.point[1] - stepsize*gradient[1],
                  old.point[2] - stepsize*gradient[2])

    new.value.function = obj.function(new.point)


    if( abs(old.value.function - new.value.function) <= conv_threshold) {
      converged = T
    }

    data.output = data.frame(iteration = iterations,
                             old.value.function = old.value.function,
                             new.value.function = new.value.function,
                             old.point=old.point, new.point=new.point
                             )

    if(exists("iters")) {
      iters <- rbind(iters, data.output)
    } else {
      iters = data.output
    }
```

```r
    iterations = iterations + 1
    old.value.function = new.value.function

    if(iterations >= max_iter) break
  }
  return(list(converged = converged,
              num_iterations = iterations,
              old.value.function = old.value.function,
              new.value.function = new.value.function,
              coefs = new.point,
              iters = iters))
}

#
# for(tolerance in mytol){
#   for(stepsize in mystepsize){
#     results = gradientDesc(objective.function, mystartpoint, stepsize,
#                            tolerance, 30000)
#     paste("Stepsize = ", stepsize, ", Tolerance: ", tolerance)
#     results$num_iterations
#     results$new.value.function
#     results$coefs
#     ggplot(data = results$iters, mapping = aes(x = iteration, y = new.value.function)) +     geom_lin
#   }
# }

for(tolerance in mytol){
  for(step in mystepsize){
    print("tolerance and step size:")
    print(c(tolerance, step))

    results = gradientDesc(objective.function, mystartpoint, step,
                           tolerance, 30000)
        print("Number of iterations: ")
    print(results$num_iterations)
    print("Minimum found: ")
    print(results$coefs)
    plot = ggplot(data = results$iters, mapping = aes(x = iteration, y = new.value.function)) +geom_lin
    print(plot)


  }
}
```

```
## [1] "tolerance and step size:"
## [1] 1e-04 1e-02
## [1] "Number of iterations: "
## [1] 141
## [1] "Minimum found: "
## [1] -0.7835792  0.3043330
```

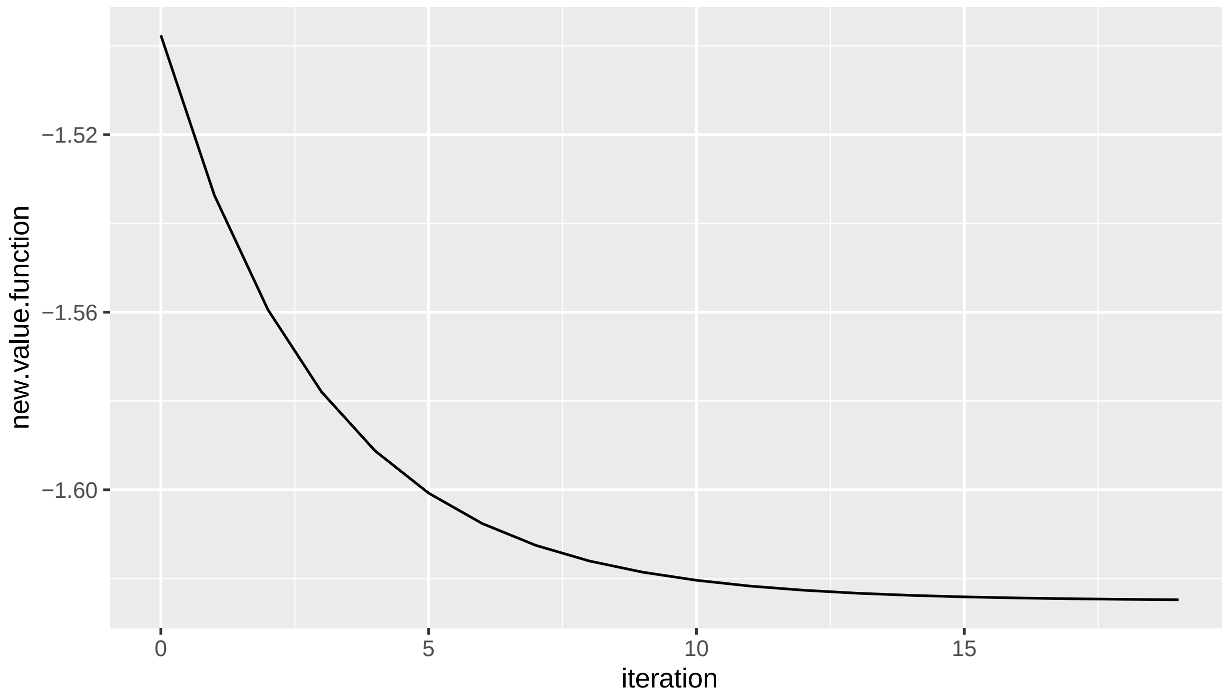



```
## [1] "tolerance and step size:"
## [1] 1e-04 5e-02
## [1] "Number of iterations: "
## [1] 38
## [1] "Minimum found: "
## [1] -0.7634764  0.2718054
```

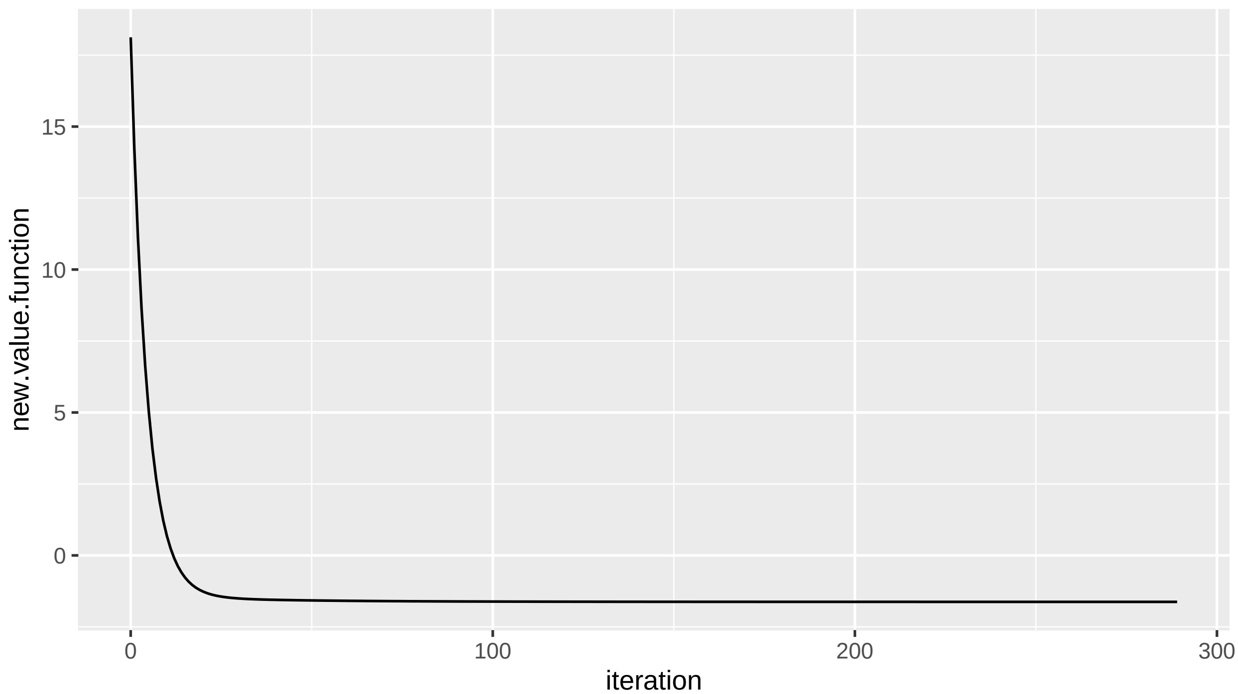



```
## [1] "tolerance and step size:"
```

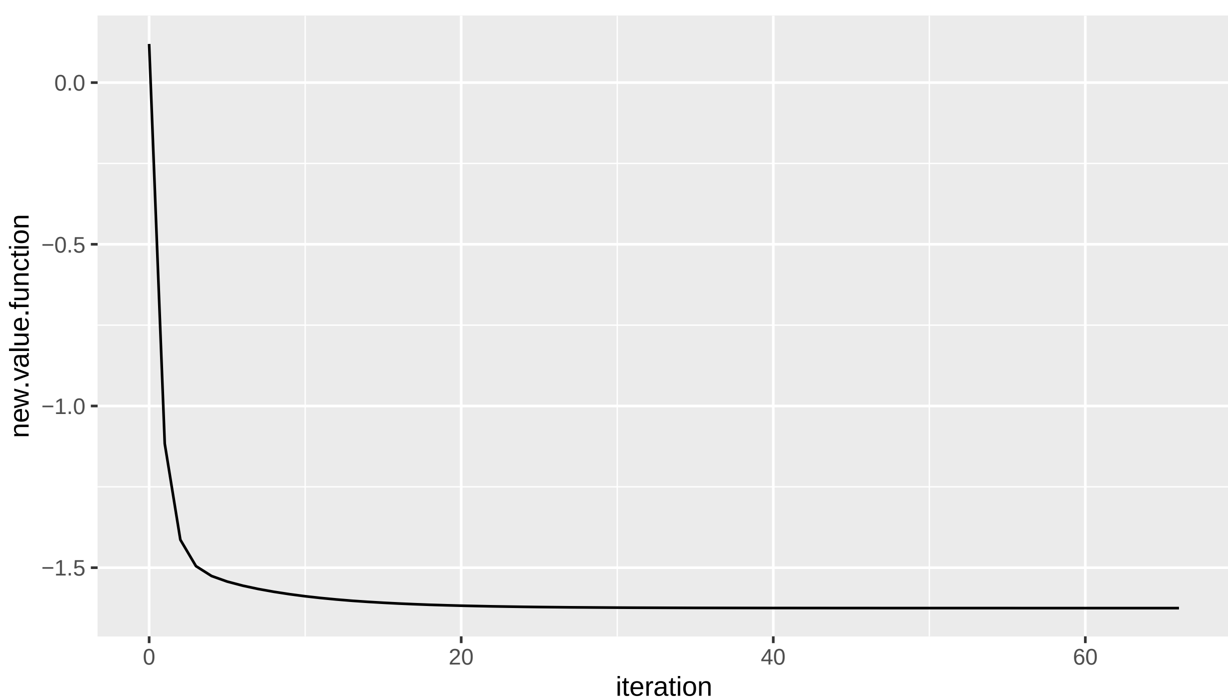
```
## [1] 1e-04 1e-01
## [1] "Number of iterations: "
## [1] 20
## [1] "Minimum found: "
## [1] -0.7591924  0.2648736
```




```
## [1] "tolerance and step size:"
## [1] 1e-06 1e-02
## [1] "Number of iterations: "
## [1] 290
## [1] "Minimum found: "
## [1] -0.7533866  0.2554796
```

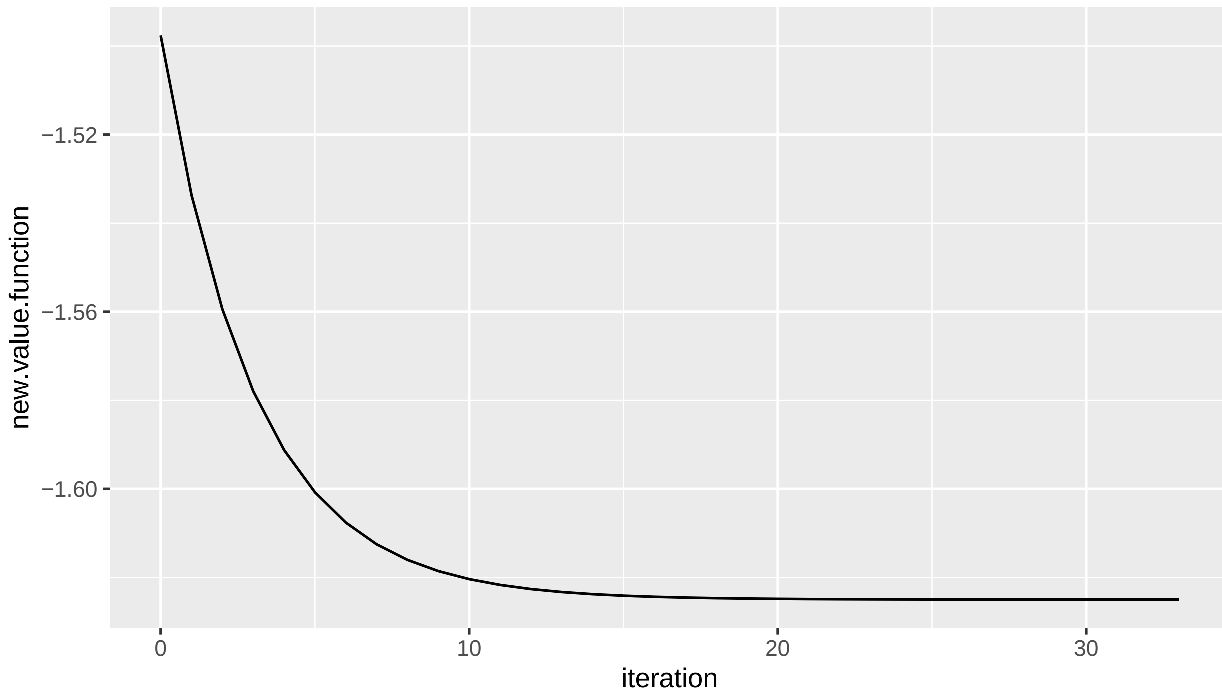



```
## [1] "tolerance and step size:"
## [1] 1e-06 5e-02
## [1] "Number of iterations: "
## [1] 67
## [1] "Minimum found: "
## [1] -0.7513449  0.2521762
```

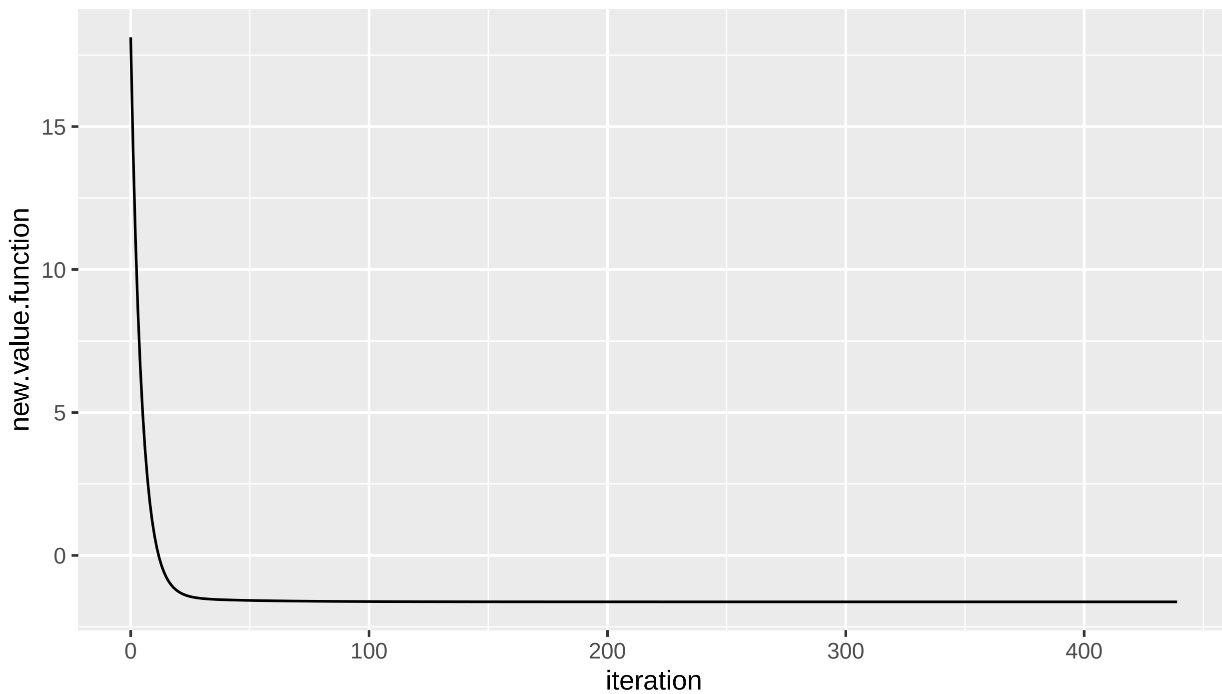



```
## [1] "tolerance and step size:"
```

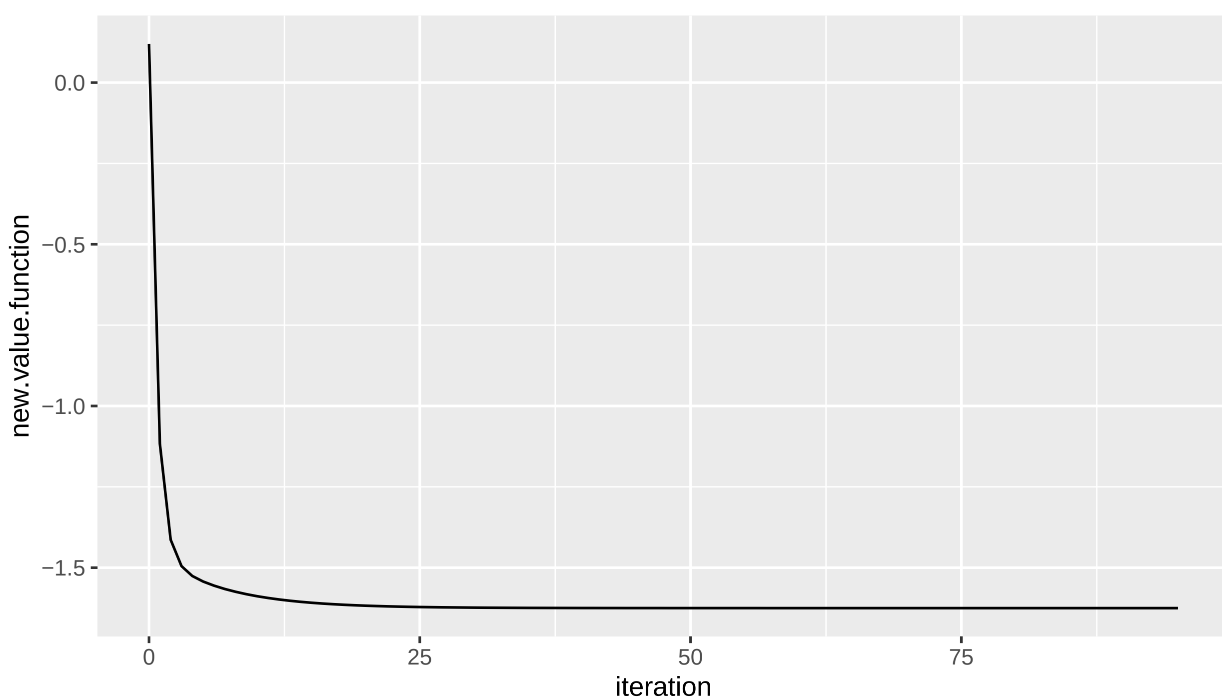
```
## [1] 1e-06 1e-01
## [1] "Number of iterations: "
## [1] 34
## [1] "Minimum found: "
## [1] -0.7509023  0.2514599
```




```
## [1] "tolerance and step size:"
## [1] 1e-08 1e-02
## [1] "Number of iterations: "
## [1] 440
## [1] "Minimum found: "
## [1] -0.7503363  0.2505442
```

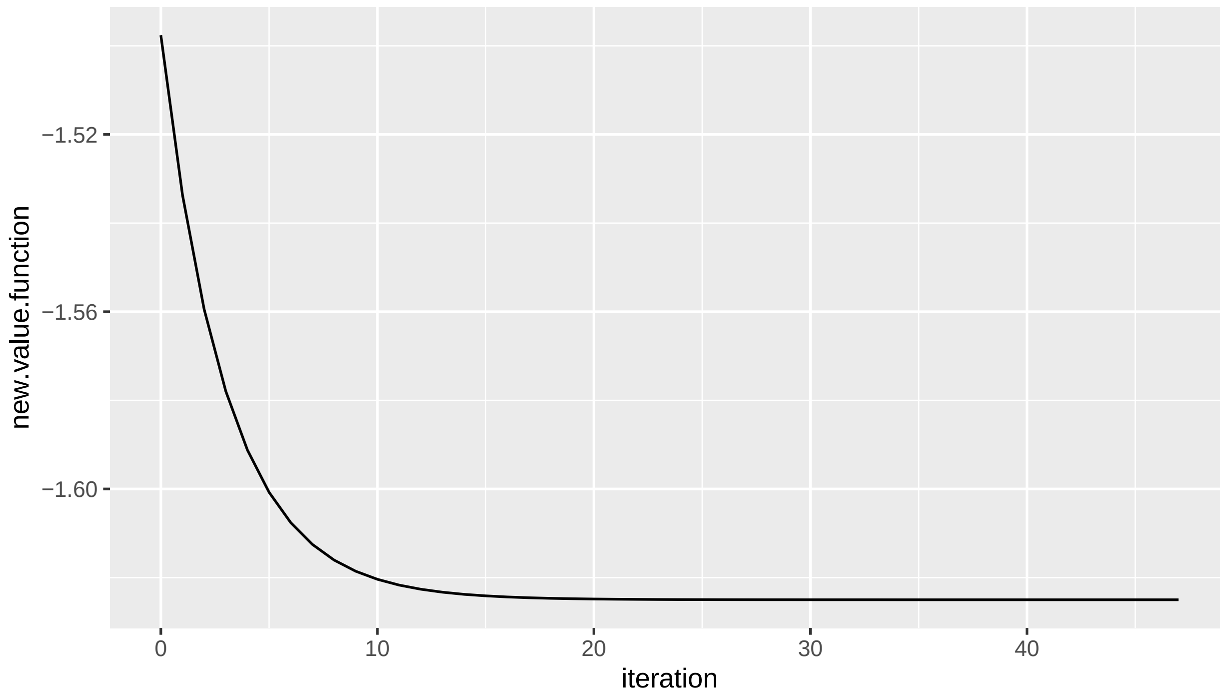



```
## [1] "tolerance and step size:"
## [1] 1e-08 5e-02
## [1] "Number of iterations: "
## [1] 96
## [1] "Minimum found: "
## [1] -0.7501342  0.2502172
```




```
## [1] "tolerance and step size:"
```

```
## [1] 1e-08 1e-01
## [1] "Number of iterations: "
## [1] 48
## [1] "Minimum found: "
## [1] -0.7500886  0.2501433
```



Comment on the accuracy of the calculated minimum as a function of the tolerance parameter, and on the number of iterations as a function of the step size.

For each set step size, the accuracy of the calculate minimum increases as tolerance decreases. Thus the accuracy and tolerance of the Gradient Descent function have an inverse relationship function.

We can see that for each tolerance setting, the number of iterations decreases as the step size increases. Thus the number of iterations and the step size have an inverse relationship function.