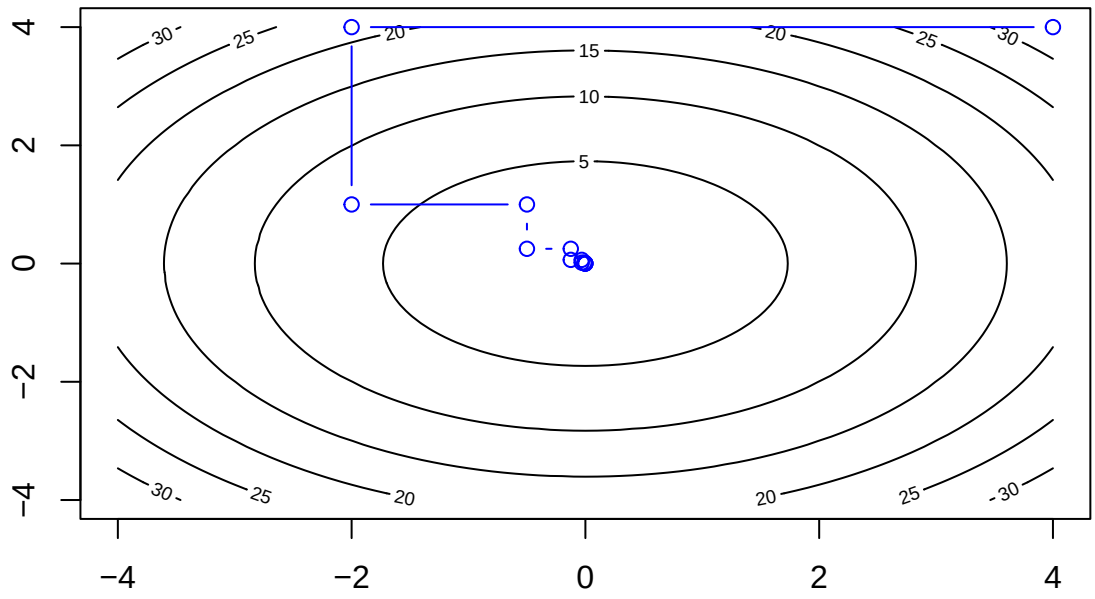
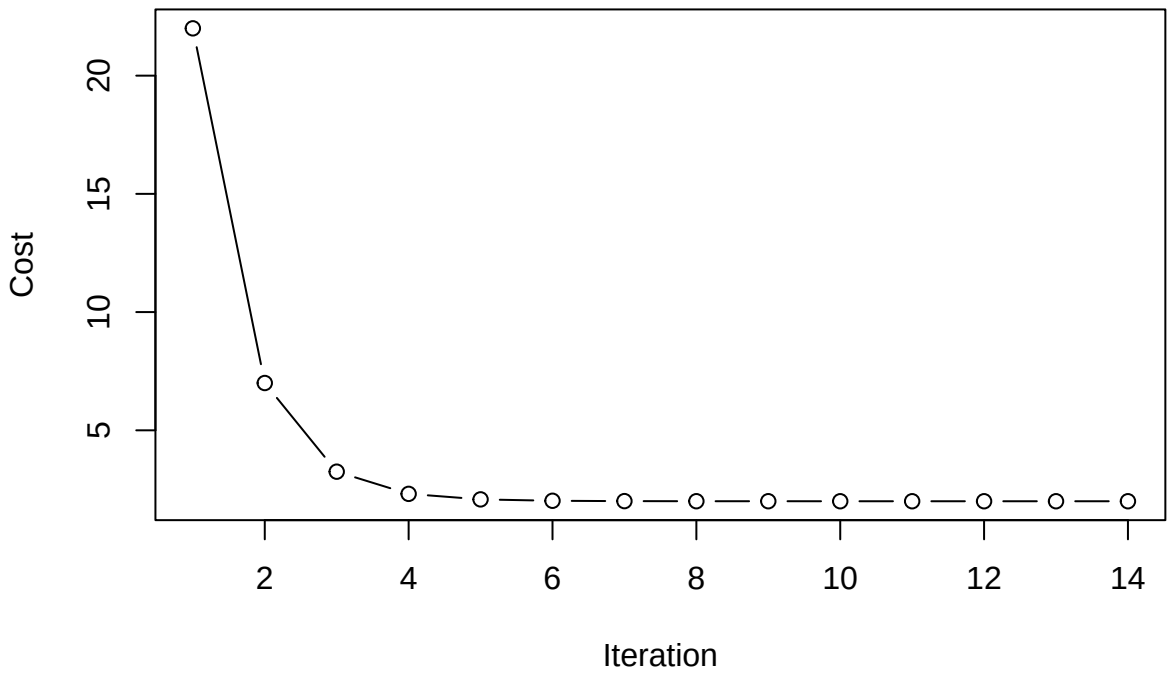
Optimzaiton 4

```
g <- function(w1, w2) {
  w1^2 + w2^2 + 2
}
coordinate_descent <- function(g, alpha_choice = 1, max_its = 100, w0, tol = 10^-5) {

  w <- matrix(w0, nrow = 1)
  weight_history <- w
  cost_history <- numeric()

i <- 1
 repeat {

  for (j in 1:2) {
    i <- i + 1
    w <- rbind(w, w[i - 1, ])
    if (j == 1) {
      w[i, j] <- -w[i - 1, 2] / 2
    } else {
      w[i, j] <- -w[i, 1] / 2
    }

      weight_history <- rbind(weight_history, w[i, ])
      cost_history <- c(cost_history, g(w[i, 1], w[i, 2]))
    }

  #if ((g(w[i, 1], w[i, 2]) - g(w[i - 1, 1], w[i - 1, 2])) >= tol || i > max_its)
   if ( g(w[i - 1, 1], w[i - 1, 2])- (g(w[i, 1], w[i, 2])) < tol || i > max_its)
    break
}


  result <- list(
    weight_history = weight_history,
    cost_history = cost_history
  )

  return(result)
}



w0 <- c(4, 4)
result <- coordinate_descent(g, alpha_choice = 1, max_its = 100, w0 = w0)
xgrid <- ygrid <- seq(-4, 4, 0.1)
g_out <- outer(xgrid, ygrid, FUN = g)
contour(x = xgrid, y = ygrid, z = g_out)
```

```
points(result$weight_history[, 1], result$weight_history[, 2], type = "b", col = "blue")
```

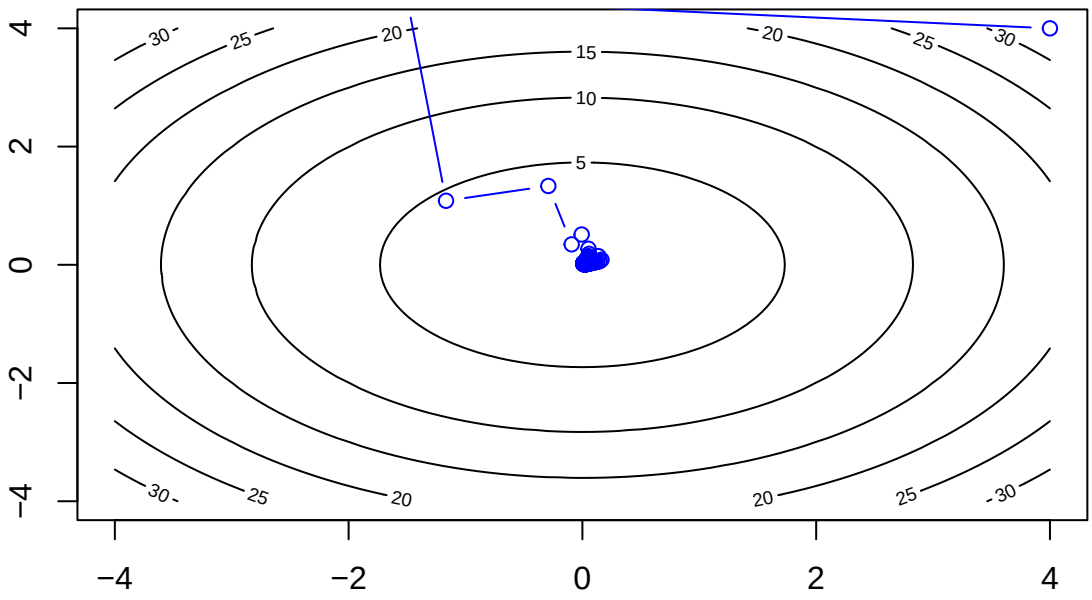



```
plot(result$cost_history, type = "b", xlab = "Iteration", ylab = "Cost")
```

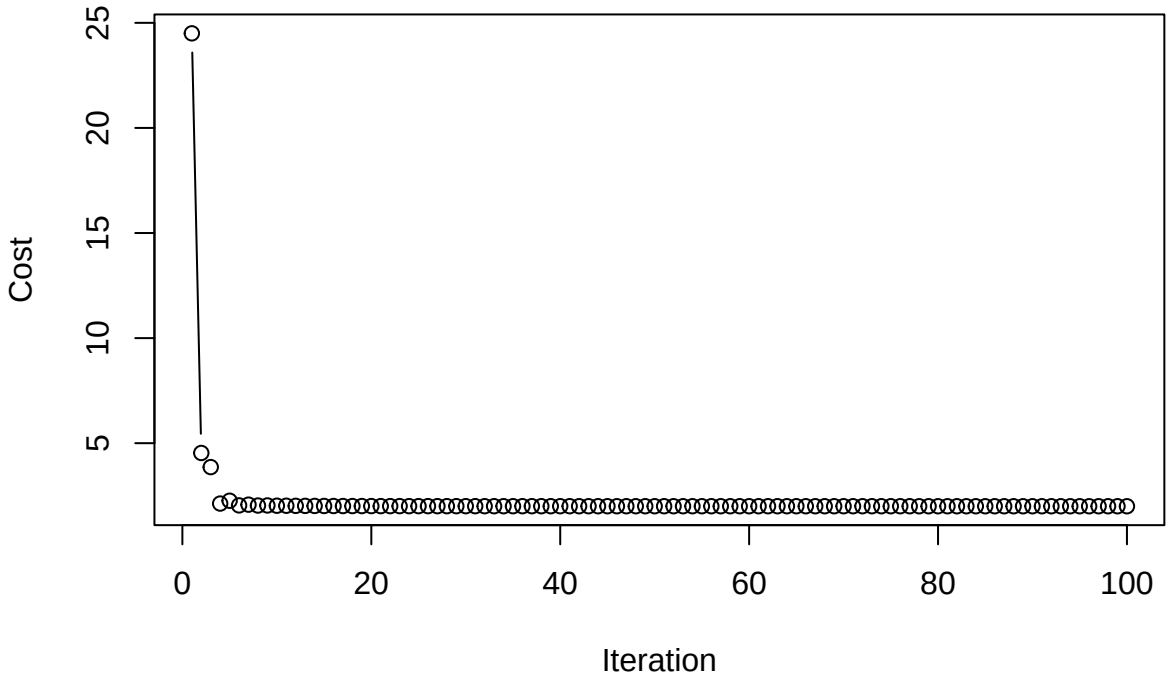



```
g <- function(w1, w2) {
  w1^2 + w2^2 + 2
}

coordinate_descent <- function(g, alpha_choice = 1, max_its = 100, w0, tol = 10^-5) {

  w <- matrix(w0, nrow = 1)
  weight_history <- w
  cost_history <- numeric()

  alpha0 <- 1


i <- 1
 repeat {

  for (j in 1:2) {
    i <- i + 1
    w <- rbind(w, w[i - 1, ])
    if (j == 1) {
      w[i, j] <- -w[i - 1, 2] / 2
    } else {
      w[i, j] <- -w[i, 1] / 2
    }
    alpha <- alpha0 *1/i
    w <- w + alpha
```

```
      weight_history <- rbind(weight_history, w[i, ])
      cost_history <- c(cost_history, g(w[i, 1], w[i, 2]))
    }

  #if ((g(w[i, 1], w[i, 2]) - g(w[i - 1, 1], w[i - 1, 2])) >= tol || i > max_its)
   if ( g(w[i - 1, 1], w[i - 1, 2])- (g(w[i, 1], w[i, 2])) < tol || i > max_its)
    break
}

  result <- list(
    weight_history = weight_history,
    cost_history = cost_history
  )

  return(result)
}


w0 <- c(4, 4)

result <- coordinate_descent(g, alpha_choice = "diminishing", max_its = 100, w0 = w0)

xgrid <- ygrid <- seq(-4, 4, 0.1)
g_out <- outer(xgrid, ygrid, FUN = g)
contour(x = xgrid, y = ygrid, z = g_out)
points(result$weight_history[, 1], result$weight_history[, 2], type = "b", col = "blue")
```



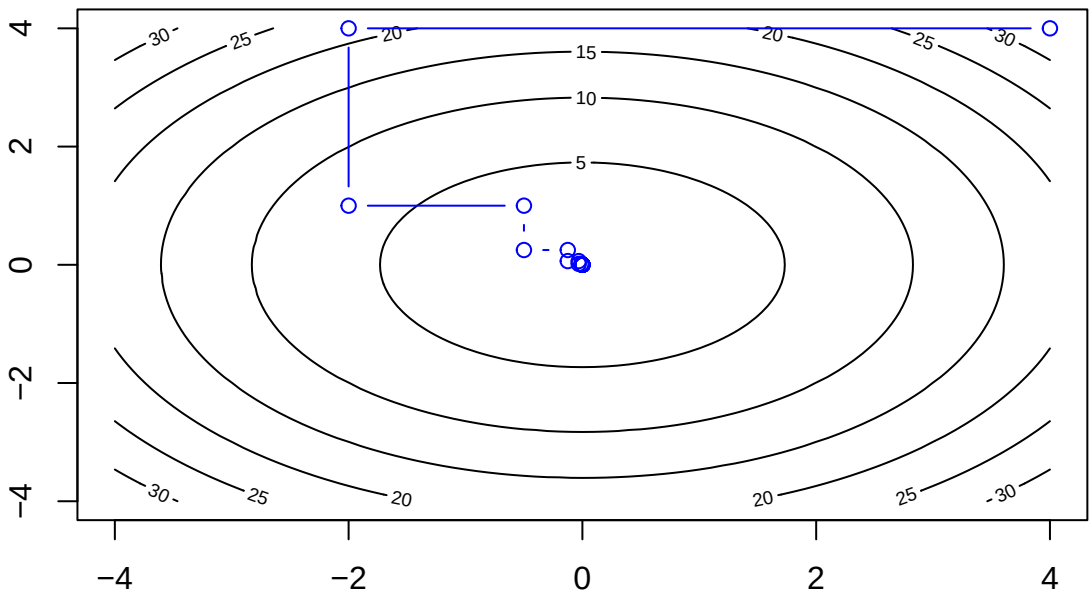

```
plot(result$cost_history, type = "b", xlab = "Iteration", ylab = "Cost")
```



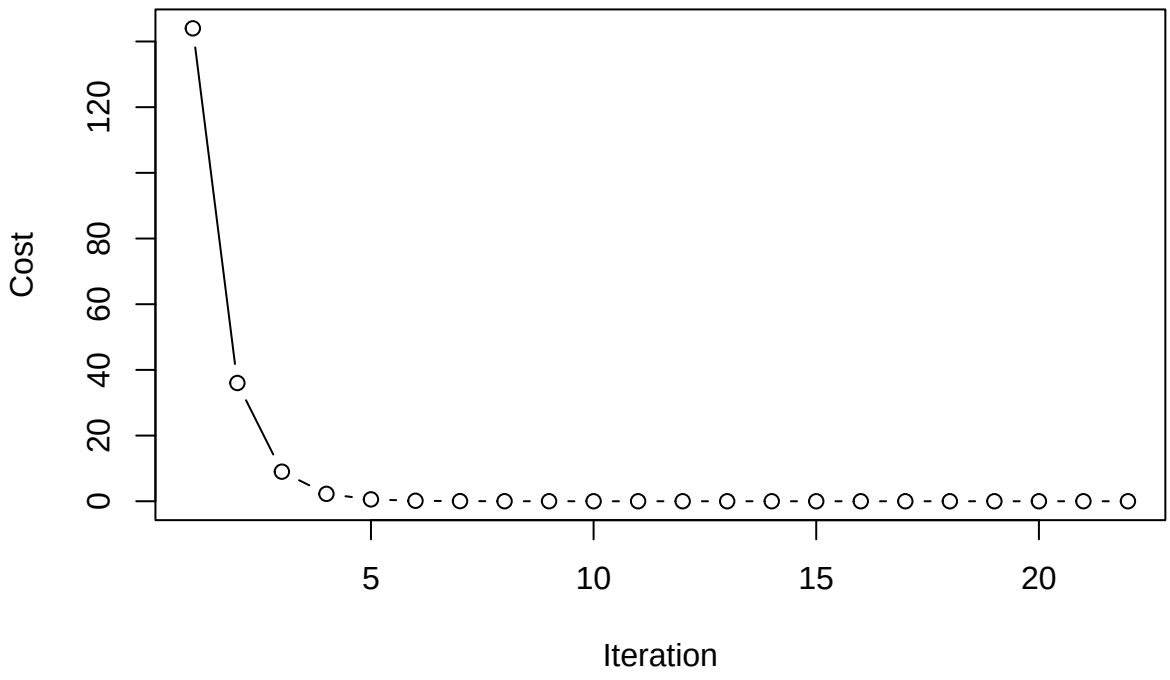

```
g <- function(w1, w2) {
  0.26*(w1^2 + w2^2) - 0.48*w1*w2
}

#Global minimum of the given cost function is:
#g_prime_w1 <-0.52*w1 - 0.48*w2
#g_prime_w2 <- 0.52*w2 - 0.48*w1

#0.52w1 - 0.48w2 = 0
#0.52w2 - 0.48w1 = 0

#0.2704w1 - 0.2496w2 = 0
#0.2496w2 - 0.2304w1 = 0

#0.2704w1 - 0.2304w1 - 0.2496w2 + 0.2496w2 = 0

#0.04w1 = 0

#w1 = 0
#w2 = 0

#Therefore, the global minimum of the given cost function is at w1 = 0 and w2 = 0.
```

```
#First order coordinate descent algorithm as a local optimization scheme:
#We leverage the given function's first derivative to calculate a descent direction, hence the first or
```

```r
g <- function(w1, w2) {
  return(c(4 * w1 + 2 * w2, 2 * w1 + 4 * w2))
}

coordinate_descent <- function(g, alpha_choice = 1, max_its = 100, w0, tol = 1e-5) {
  w <- matrix(w0, nrow = 1)
  weight_history <- w
  cost_history <- numeric()
  i <- 1
  repeat {
    for (j in 1:2) {
      i <- i + 1
      w <- rbind(w, w[i - 1, ])
      if (j == 1) {
        w[i, j] <- -w[i - 1, 2] / 2
      } else {
        w[i, j] <- -w[i, 1] / 2
      }
      weight_history <- rbind(weight_history, w[i, ])
      cost_history <- c(cost_history, sum(g(w[i, 1], w[i, 2])^2))
    }
    if (abs(g(w[i - 1, 1], w[i - 1, 2]) - g(w[i, 1], w[i, 2])) < tol || i > max_its)
      break
  }
  result <- list(
    weight_history = weight_history,
    cost_history = cost_history
  )
  return(result)
}

w0 <- c(4, 4)
result <- coordinate_descent(g, alpha_choice = 1, max_its = 100, w0 = w0)

xgrid <- seq(-4, 4, 0.1)
ygrid <- seq(-4, 4, 0.1)
#g_out <- outer(xgrid, ygrid, FUN = (function(x, y) sum(g(x, y)^2)))

contour(x = xgrid, y = ygrid, z = g_out)
points(result$weight_history[, 1], result$weight_history[, 2], type = "b", col = "blue")
```

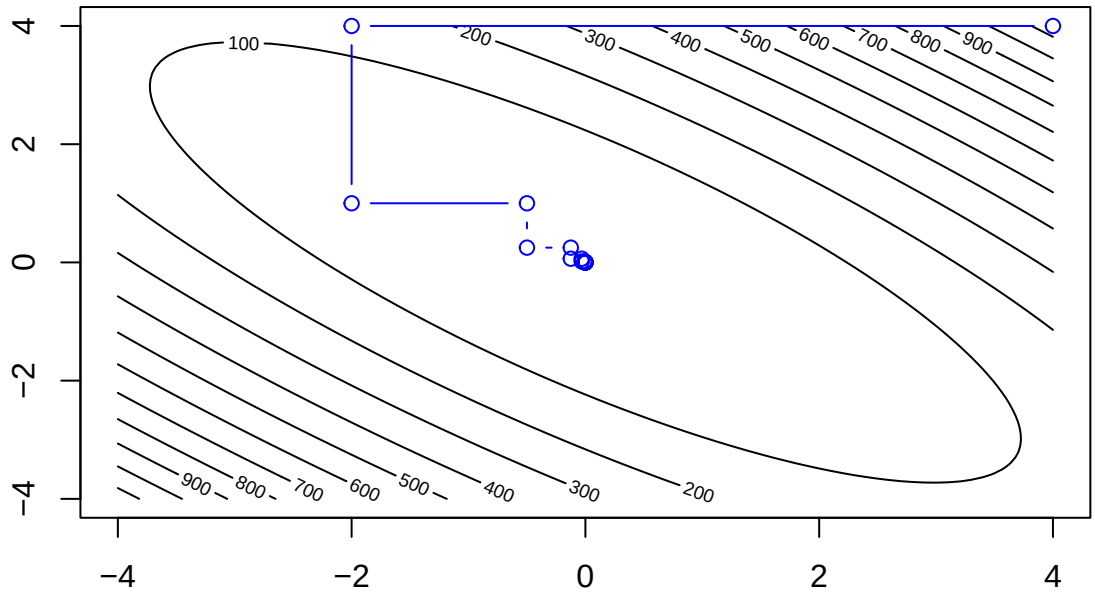



```
plot(result$cost_history, type = "b", xlab = "Iteration", ylab = "Cost")
```

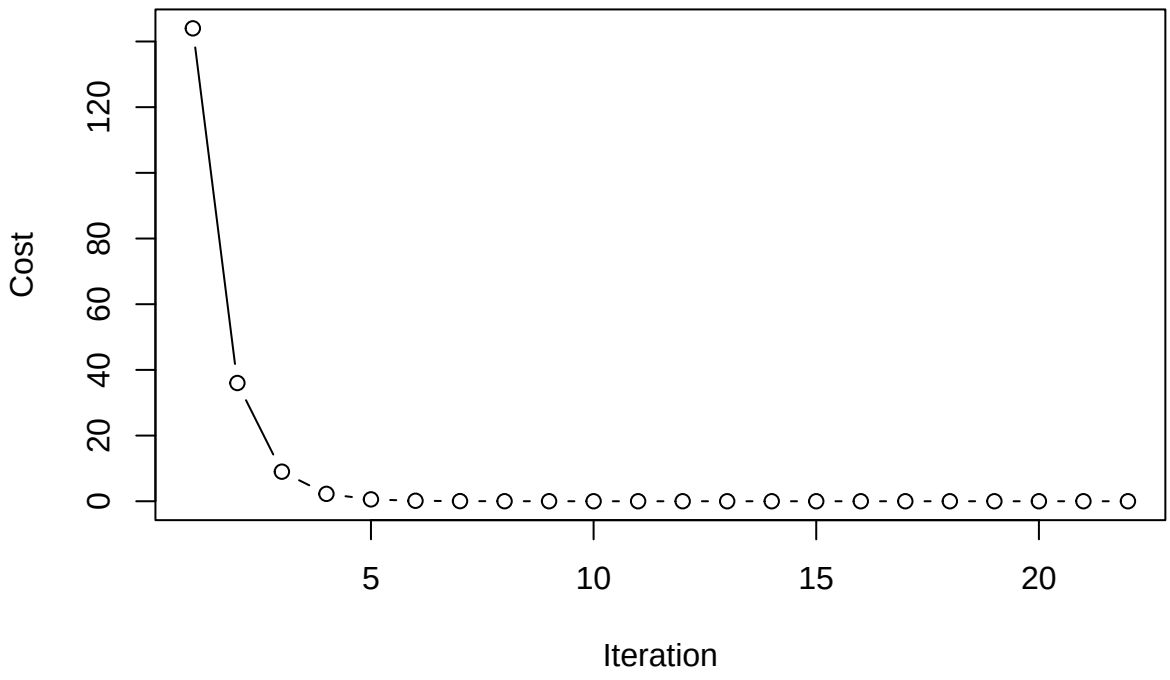



```
g <- function(w1, w2) {
  return(c(4 * w1 + 2 * w2, 2 * w1 + 4 * w2))
}

coordinate_descent <- function(g, alpha_choice = 1, max_its = 100, w0, tol = 1e-5) {
  w <- matrix(w0, nrow = 1)
  weight_history <- w
  cost_history <- numeric()
  i <- 1
  alpha0 <- 1
  repeat {
    for (j in 1:2) {
      i <- i + 1
      w <- rbind(w, w[i - 1, ])
      if (j == 1) {
        w[i, j] <- -w[i - 1, 2] / 2
      } else {
        w[i, j] <- -w[i, 1] / 2
      }
      weight_history <- rbind(weight_history, w[i, ])
      cost_history <- c(cost_history, sum(g(w[i, 1], w[i, 2])^2))
      if (alpha_choice == "diminishing") {
        alpha <- alpha0 / i
        w[i, ] <- w[i, ] + alpha * g(w[i, 1], w[i, 2])
      }
    }
```

```
    if (abs(g(w[i - 1, 1], w[i - 1, 2]) - g(w[i, 1], w[i, 2])) < tol || i > max_its)
      break
  }
  result <- list(
    weight_history = weight_history,
    cost_history = cost_history
  )
  return(result)
}

w0 <- c(4, 4)
result <- coordinate_descent(g, alpha_choice = "diminishing", max_its = 100, w0 = w0)

xgrid <- seq(-4, 4, 0.1)
ygrid <- seq(-4, 4, 0.1)
g_out <- outer(xgrid, ygrid, FUN = Vectorize(function(x, y) sum(g(x, y)^2)))

contour(x = xgrid, y = ygrid, z = g_out)
points(result$weight_history[, 1], result$weight_history[, 2], type = "b", col = "blue")
```



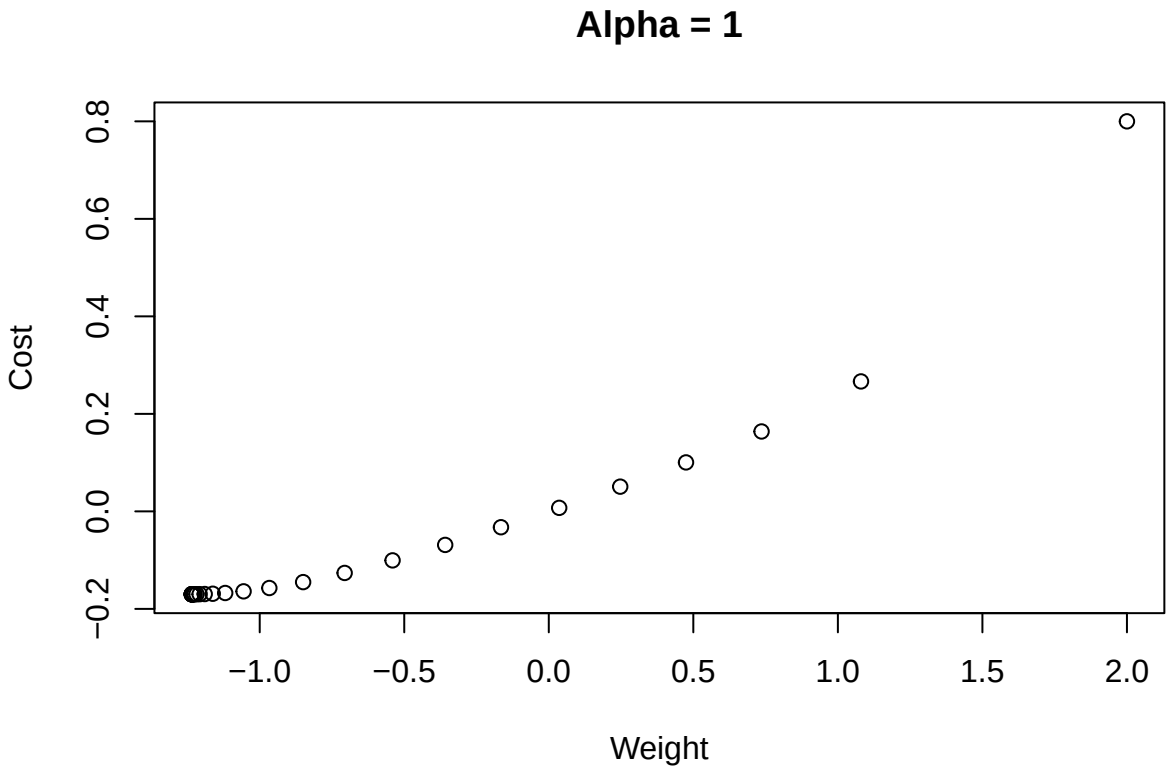

```
plot(result$cost_history, type = "b", xlab = "Iteration", ylab = "Cost")
```

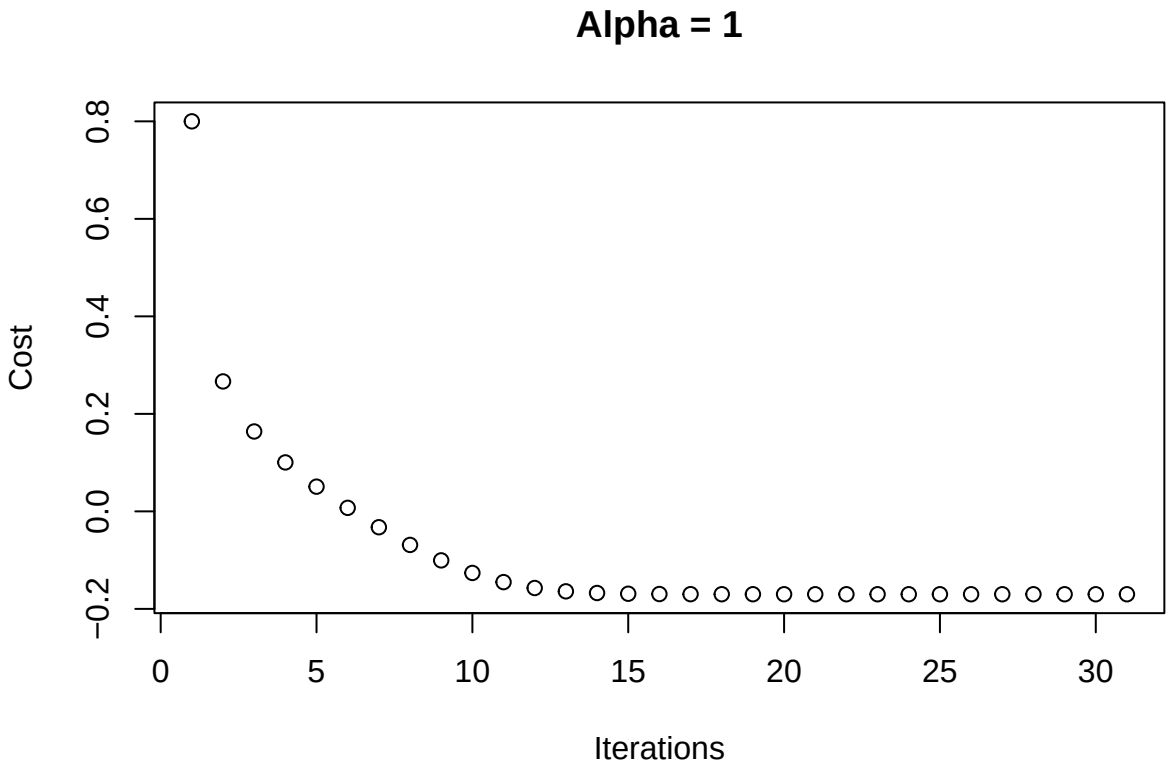


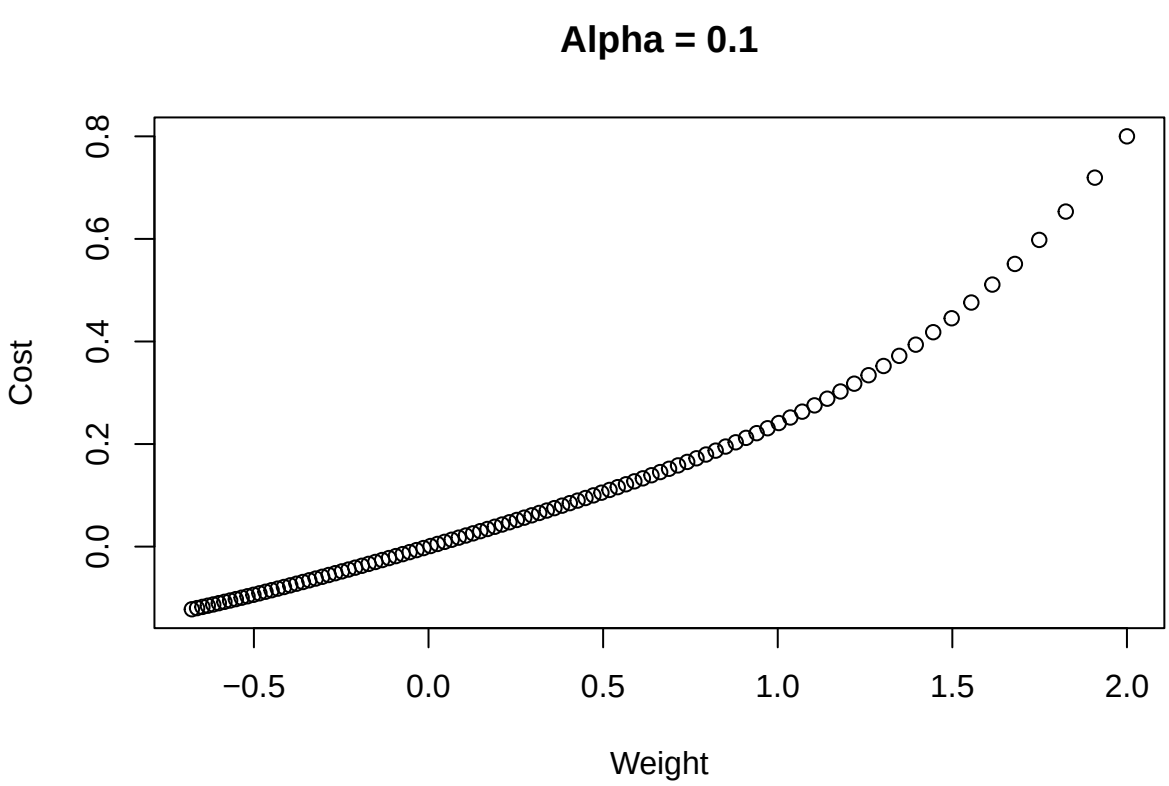

```
#1. The sum of convex functions is always convex and 2. The maximum of convex functions is convex
#Given: f(x) and g(x); h(x) = f(x) + g(x); thus for any 0 <= lamba <= 0; f(lamba(x) +(1-lamba)y) <= lam
```

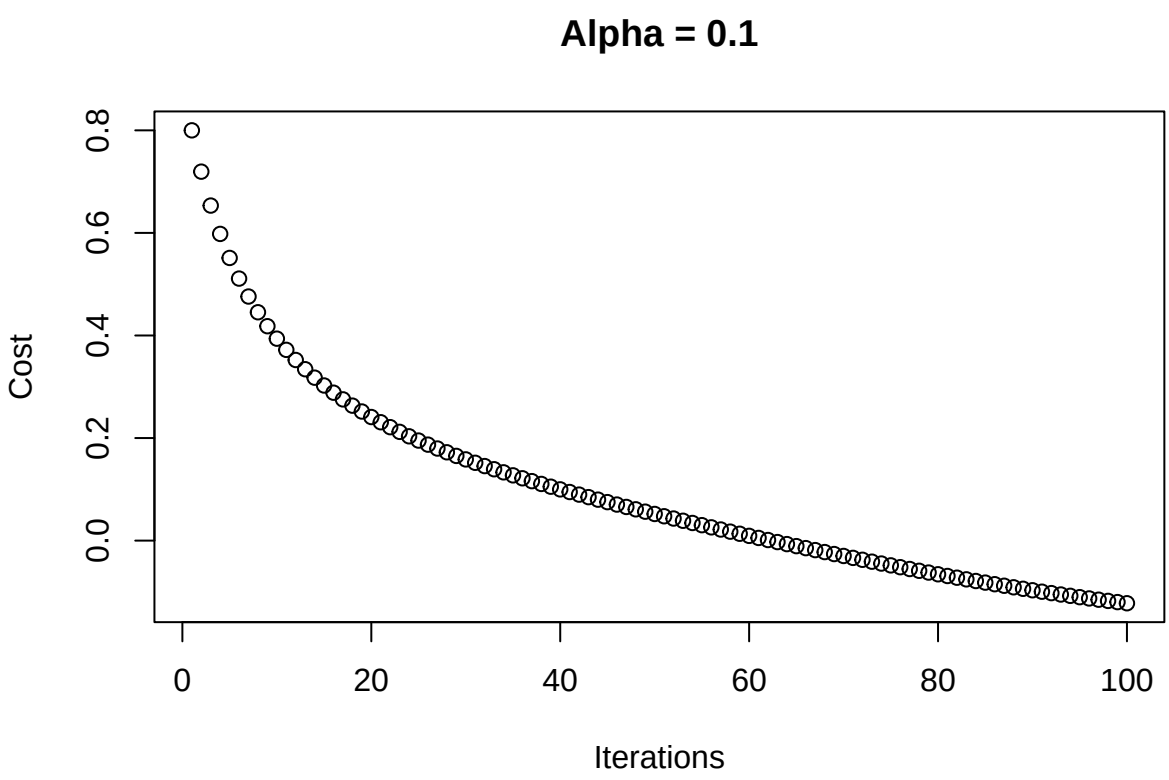
```
g <- function(w) {
  1/50 * (w^4 + w^2 + 10*w)
}

g_prime <- function(w) {
  1/50 * (4*w^3 + 2*w + 10)
}

gd <- function(g, g_prime, alpha_choice = 1, max_its = 100, w0 = 0, tol = 1e-5) {
  alpha <- alpha_choice
  k <- 1
  w <- w0
  cost_history <- numeric(max_its)
  weight_history <- matrix(0, nrow = length(w), ncol = max_its)
  cost_history[1] <- g(w)
  weight_history[, 1] <- w

  while (k < max_its & sqrt(sum(g_prime(w)^2)) > tol) {
    k <- k + 1
    if (alpha_choice == "diminishing") {
      alpha <- alpha / sqrt(k)
    }
    w <- w - alpha * g_prime(w)
```

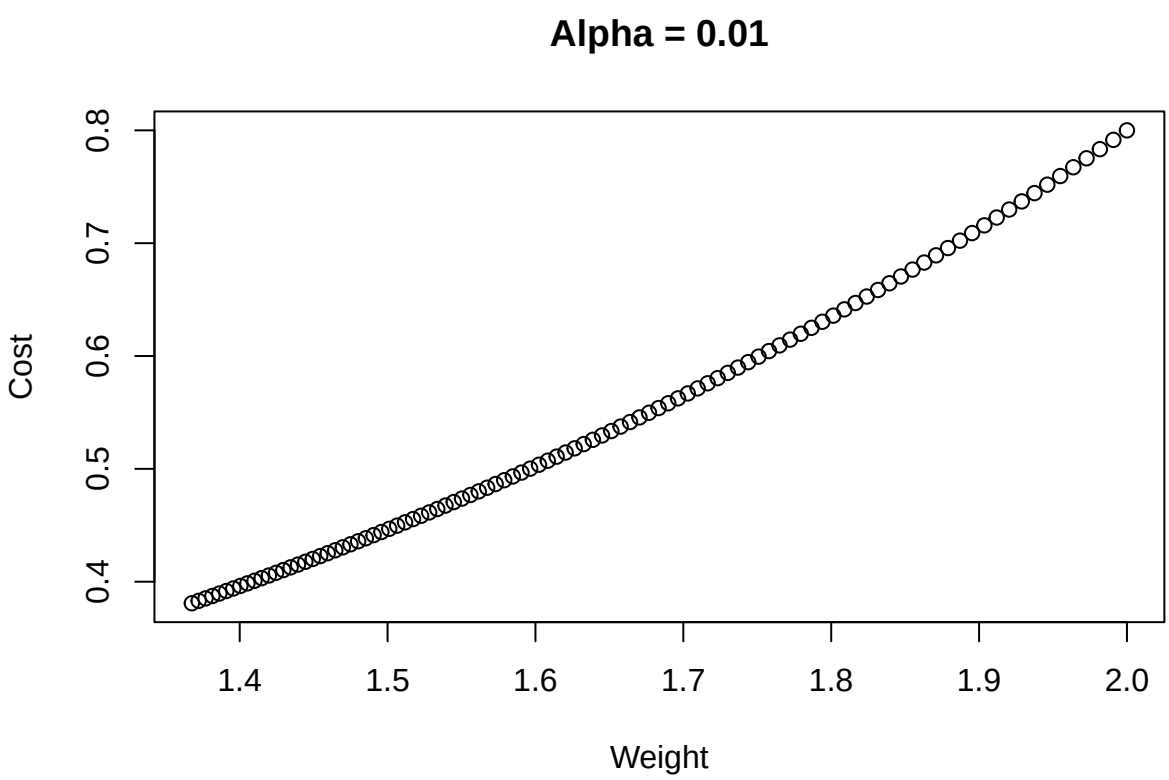
```
    cost_history[k] <- g(w)
    weight_history[, k] <- w
  }

  list(weight_history = weight_history[, 1:k], cost_history = cost_history[1:k])
}

g <- function(w) {
  1/50 * (w^4 + w^2 + 10 * w)
}

g_prime <- function(w) {
  1/50 * (4 * w^3 + 2 * w + 10)
}

alpha_choices <- c(1, 0.1, 0.01)
initial_point <- 2

for (i in 1:length(alpha_choices)) {
  alpha_choice <- alpha_choices[i]
  result <- gd(g, g_prime, alpha_choice, w0 = initial_point)

  plot(result$weight_history, result$cost_history,
       xlab = "Weight", ylab = "Cost", main = paste("Alpha =", alpha_choice))

  plot(result$cost_history,
       xlab = "Iterations", ylab = "Cost", main = paste("Alpha =", alpha_choice))
}
```

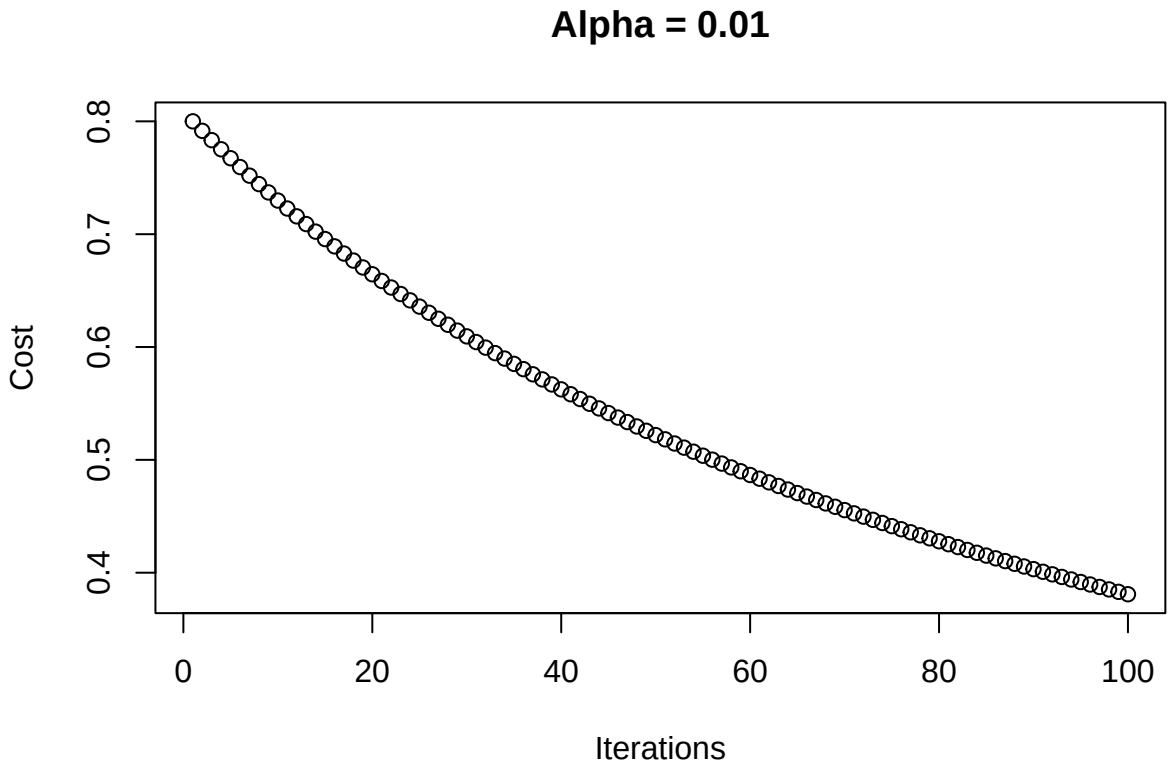

Alpha = 1
Cost
−0.2
0.0
0.2
0.4
0.6
0.8
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Weight


Alpha = 1
Cost
−0.2
0.0
0.2
0.4
0.6
0.8
0
5
10
15
20
25
30
Iterations


Alpha = 0.1
Cost
0.0
0.2
0.4
0.6
0.8
−0.5
0.0
0.5
1.0
1.5
2.0
Weight

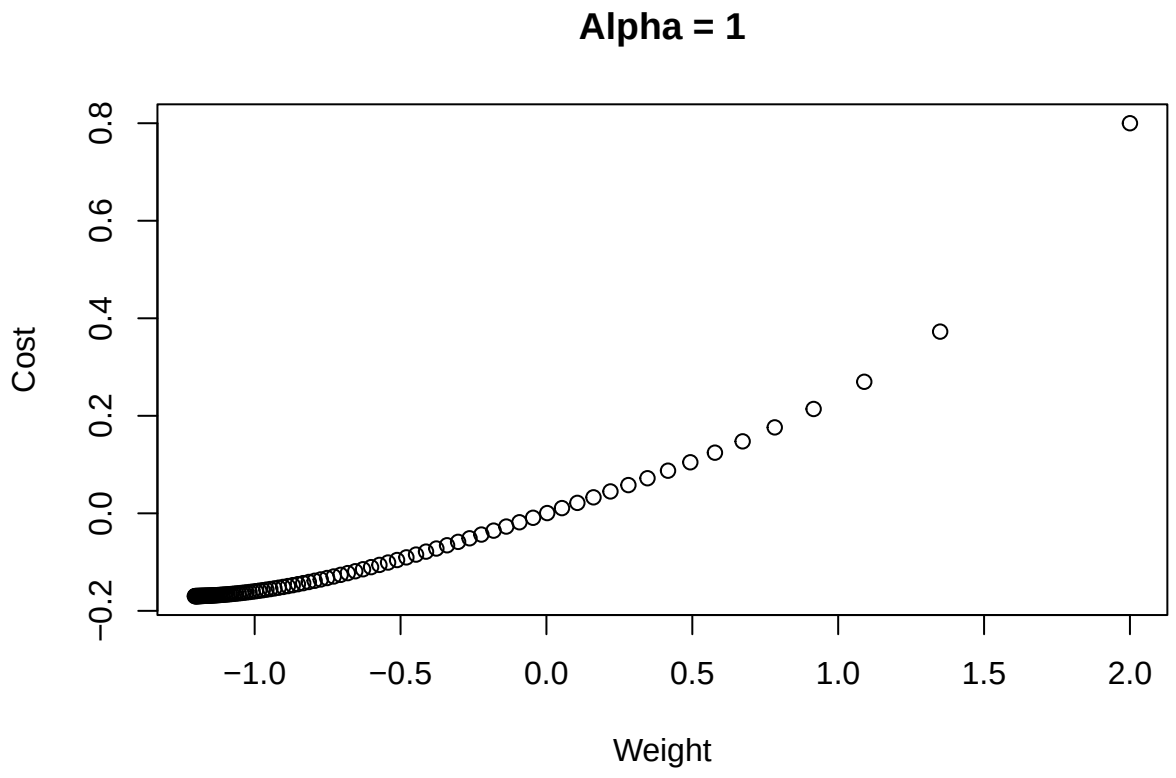

Alpha = 0.1
Cost
0.0
0.2
0.4
0.6
0.8
0
20
40
60
80
100
Iterations

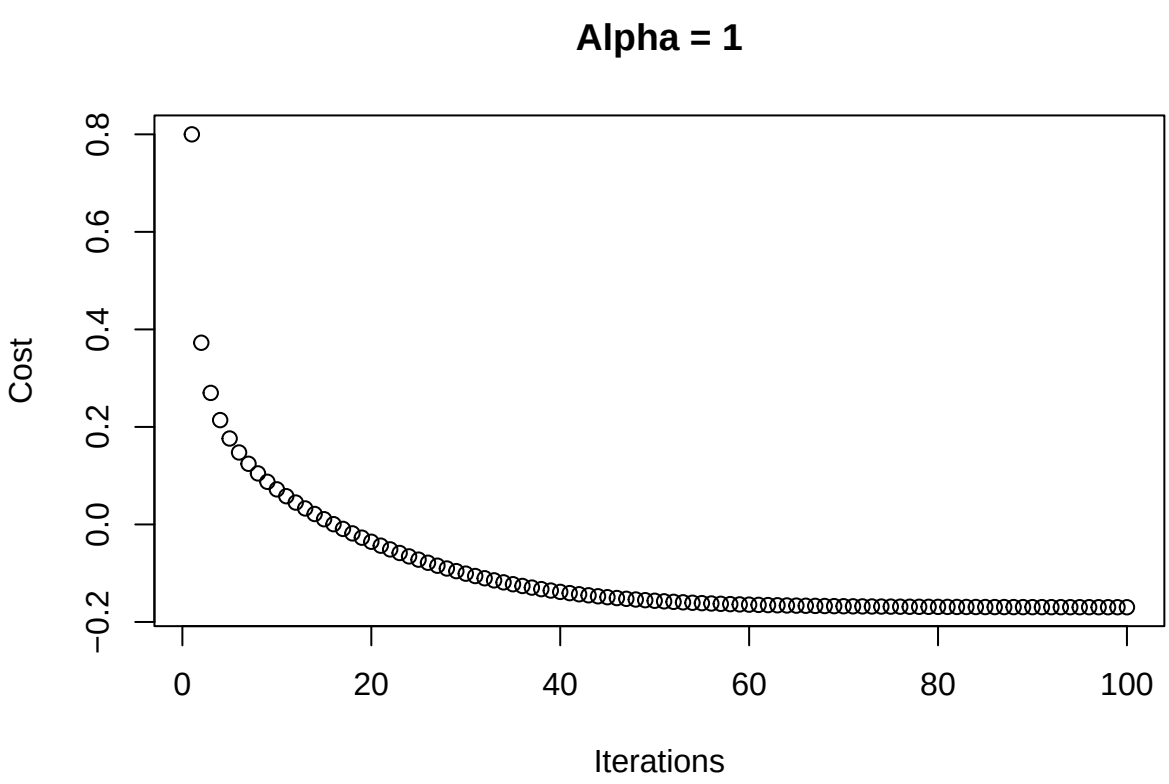

Alpha = 0.01
Cost
0.4
0.5
0.6
0.7
0.8
1.4
1.5
1.6
1.7
1.8
1.9
2.0
Weight

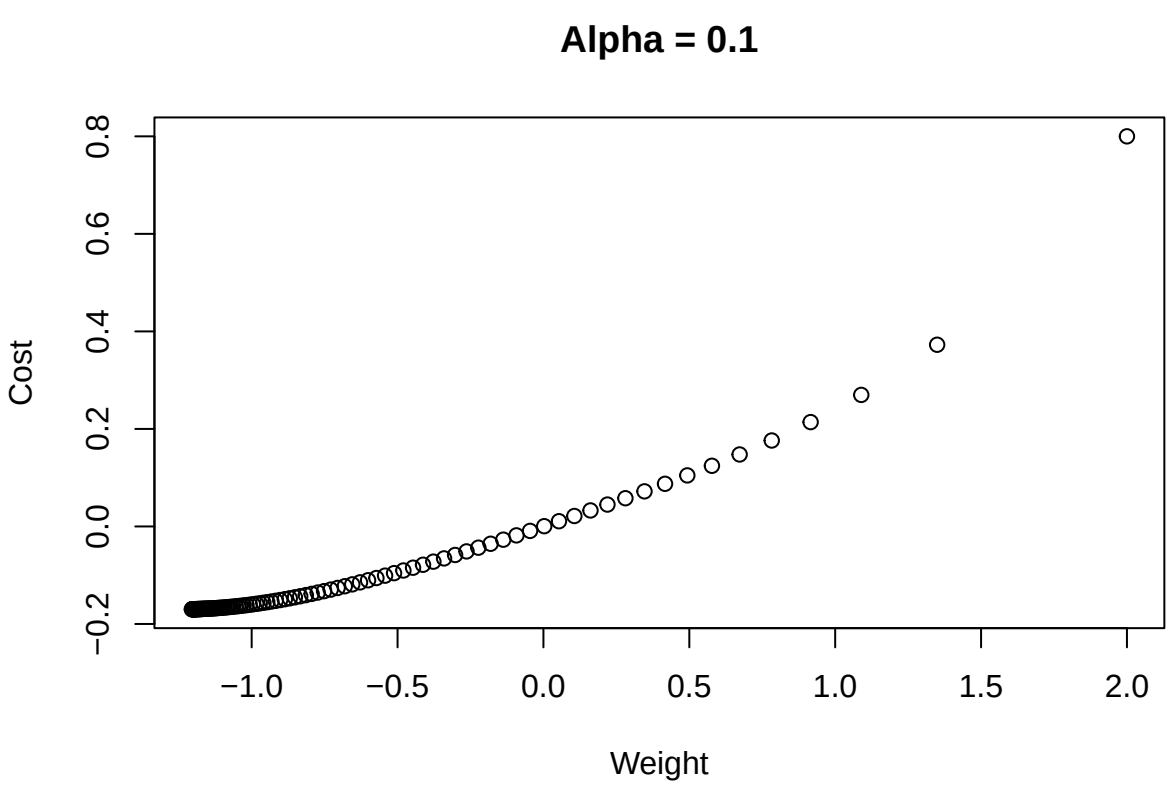


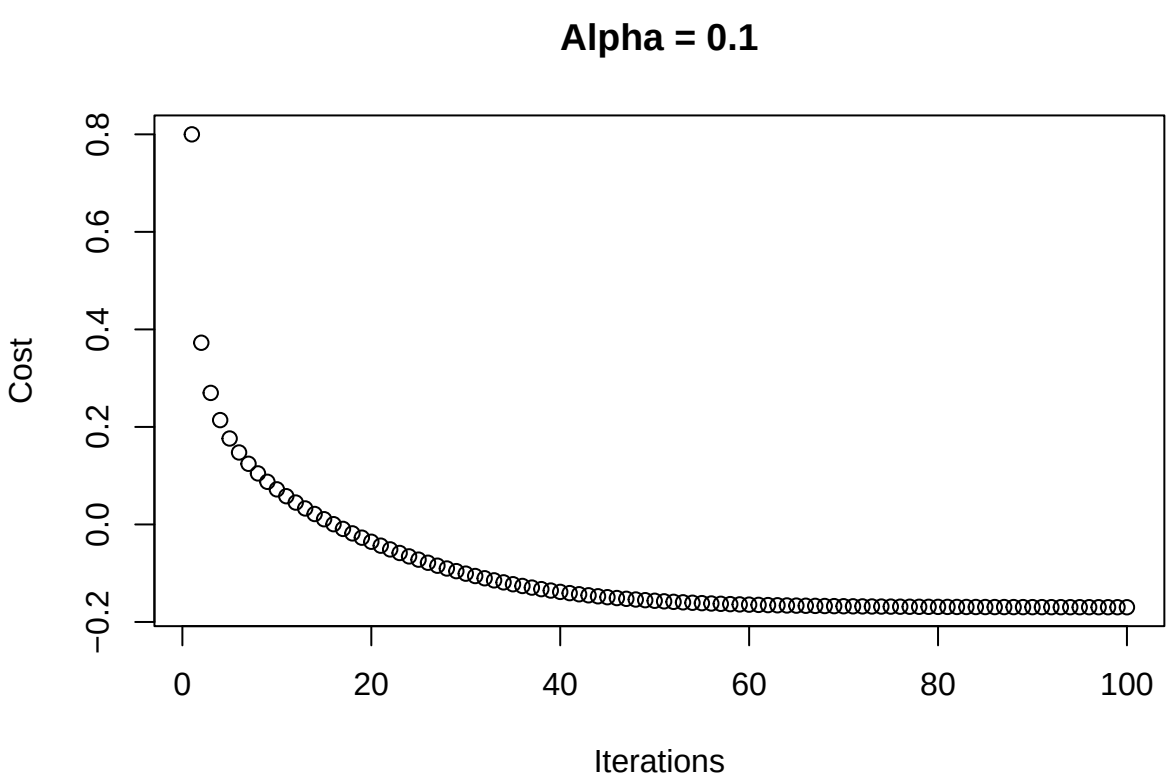

```r
g <- function(w) {
  1/50 * (w^4 + w^2 + 10*w)
}

g_prime <- function(w) {
  1/50 * (4*w^3 + 2*w + 10)
}

gd <- function(g, g_prime, alpha_choice = 1, max_its = 100, w0 = 0, tol = 1e-5) {
  alpha <- alpha_choice
  k <- 1
  w <- w0
  cost_history <- numeric(max_its)
  weight_history <- matrix(0, nrow = length(w), ncol = max_its)
  cost_history[1] <- g(w)
  weight_history[, 1] <- w
  alpha0 <- 1
  while (k < max_its & sqrt(sum(g_prime(w)^2)) > tol) {
    k <- k + 1
    if (alpha_choice == "diminishing") {
      alpha <- alpha0 / sqrt(k)
    }
    w <- w - alpha * g_prime(w)
    cost_history[k] <- g(w)
    weight_history[, k] <- w
  }
```

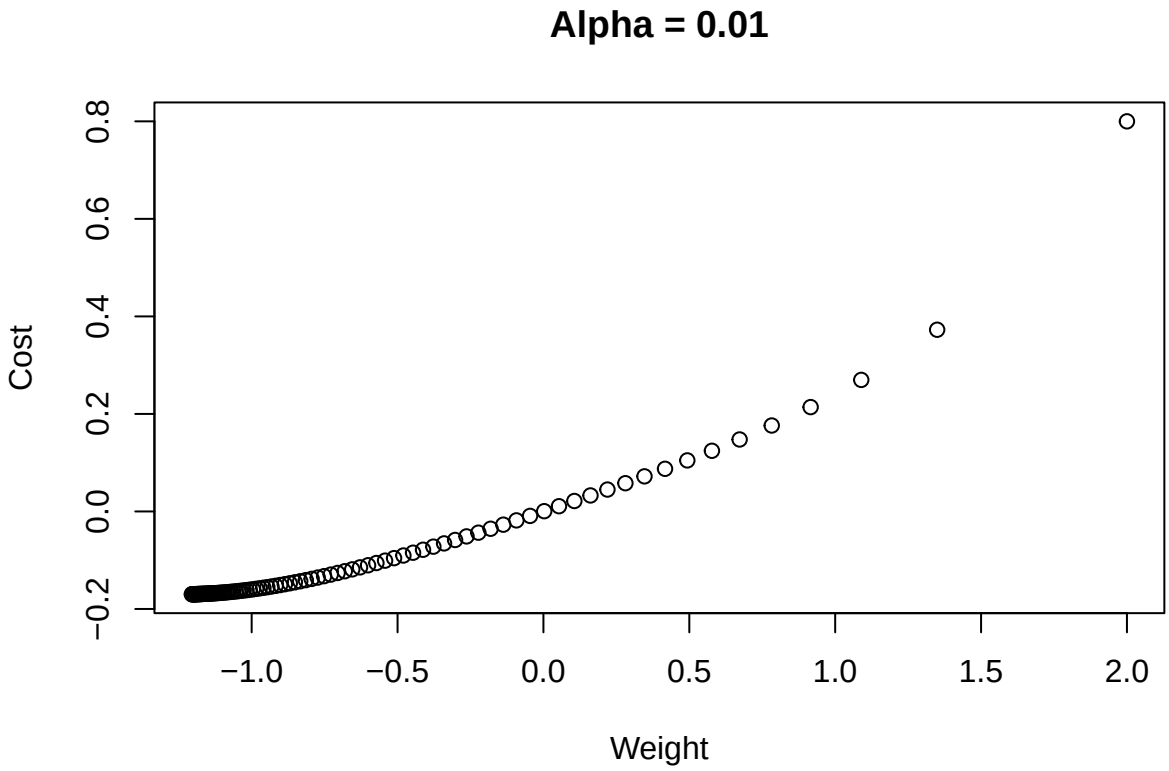
```
  list(weight_history = weight_history[, 1:k], cost_history = cost_history[1:k])
}

g <- function(w) {
  1/50 * (w^4 + w^2 + 10 * w)
}

g_prime <- function(w) {
  1/50 * (4 * w^3 + 2 * w + 10)
}

alpha_choices <- c(1, 0.1, 0.01)
initial_point <- 2

for (i in 1:length(alpha_choices)) {
  alpha_choice <- alpha_choices[i]
  result <- gd(g, g_prime, alpha_choice = "diminishing", w0 = initial_point)

  plot(result$weight_history, result$cost_history,
       xlab = "Weight", ylab = "Cost", main = paste("Alpha =", alpha_choice))
  plot(result$cost_history,
       xlab = "Iterations", ylab = "Cost", main = paste("Alpha =", alpha_choice))
}
```

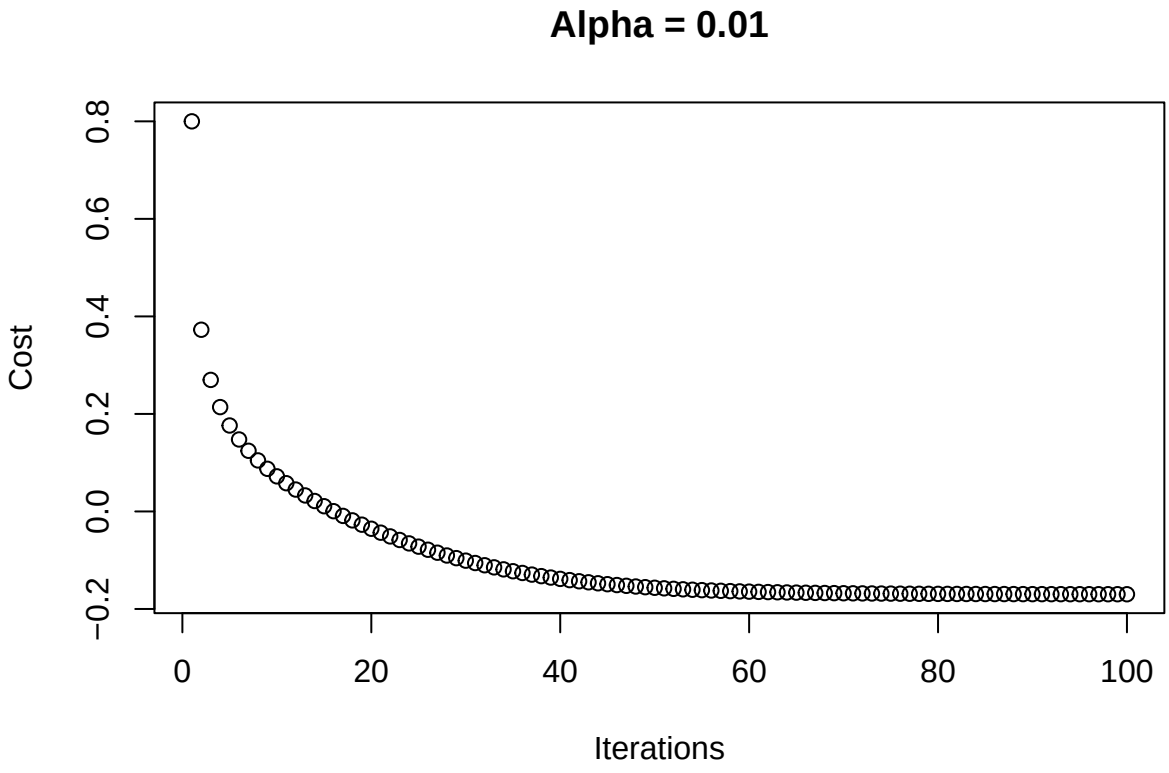



Alpha = 1
Cost
−0.2
0.0
0.2
0.4
0.6
0.8
0
20
40
60
80
100
Iterations


Alpha = 0.1
Cost
−0.2
0.0
0.2
0.4
0.6
0.8
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Weight

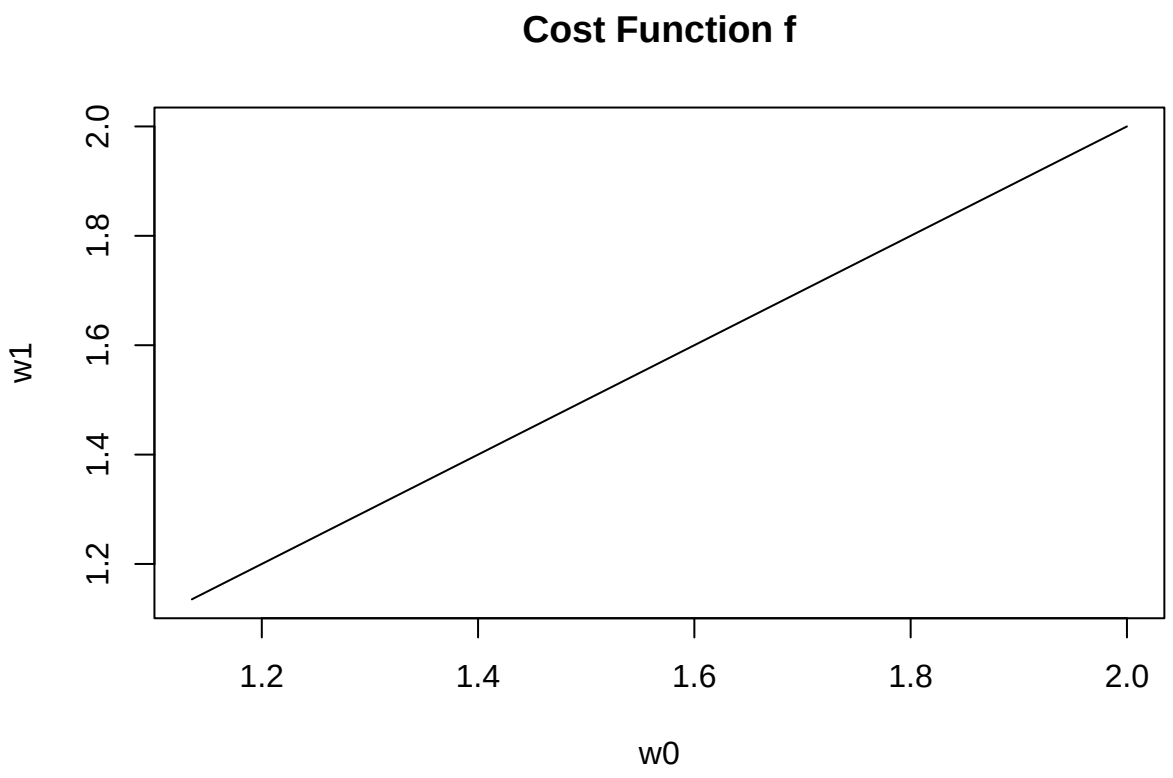

Alpha = 0.1
Cost
−0.2
0.0
0.2
0.4
0.6
0.8
0
20
40
60
80
100
Iterations


Alpha = 0.01
Cost
−0.2
0.0
0.2
0.4
0.6
0.8
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Weight

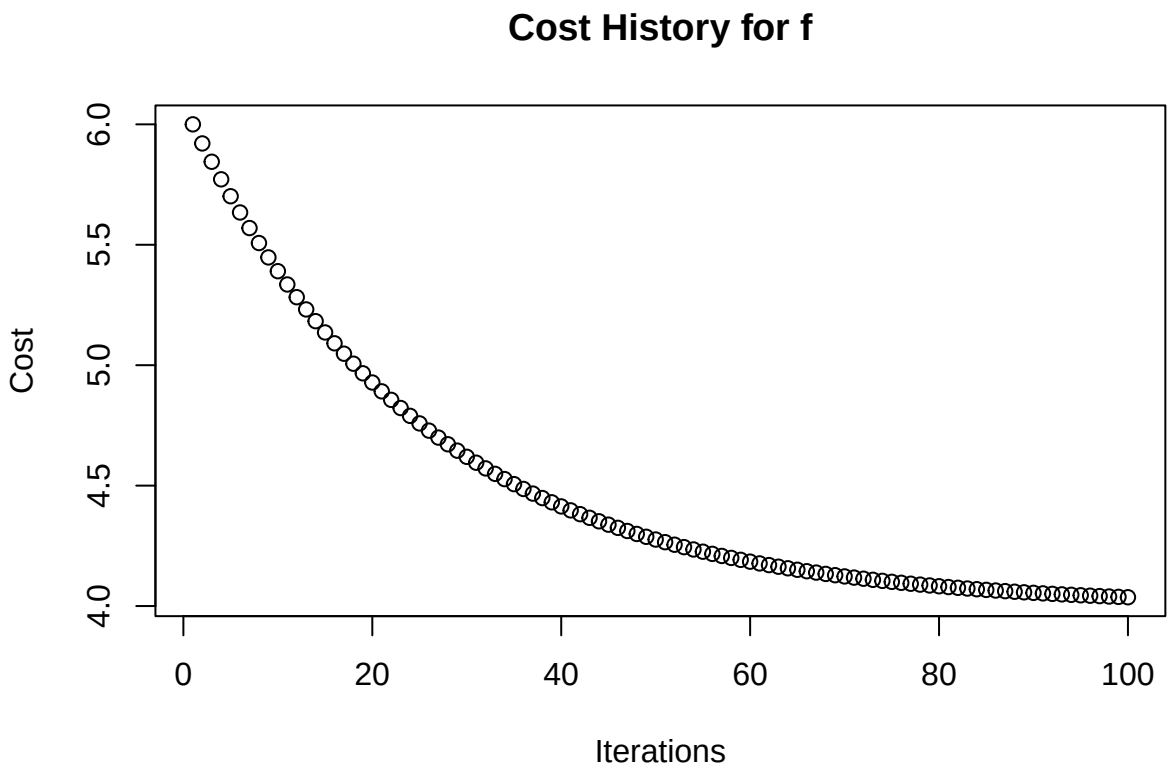



```r
gd <- function(g, g_prime, alpha_choice = 1, max_its = 100, w0 = c(0, 0), tol = 1e-5) {
  alpha <- alpha_choice
  k <- 1
  w <- w0
  cost_history <- numeric(max_its)
  weight_history <- matrix(0, nrow = length(w), ncol = max_its)
  cost_history[1] <- g(w)
  weight_history[, 1] <- w
  alpha0 <- 1
  while (k < max_its & sqrt(sum(g_prime(w)^2)) > tol) {
    k <- k + 1
    if (alpha_choice == "diminishing") {
      alpha <- alpha0 / sqrt(k)
    }
    w <- w - alpha * g_prime(w)
    cost_history[k] <- g(w)
    weight_history[, k] <- w
  }

  list(weight_history = weight_history[, 1:k], cost_history = cost_history[1:k])
}
f <- function(w) {
  w[1]^2 + w[2]^2 - 2*w[1] - 2*w[2] + 6
}

f_prime <- function(w) {
```

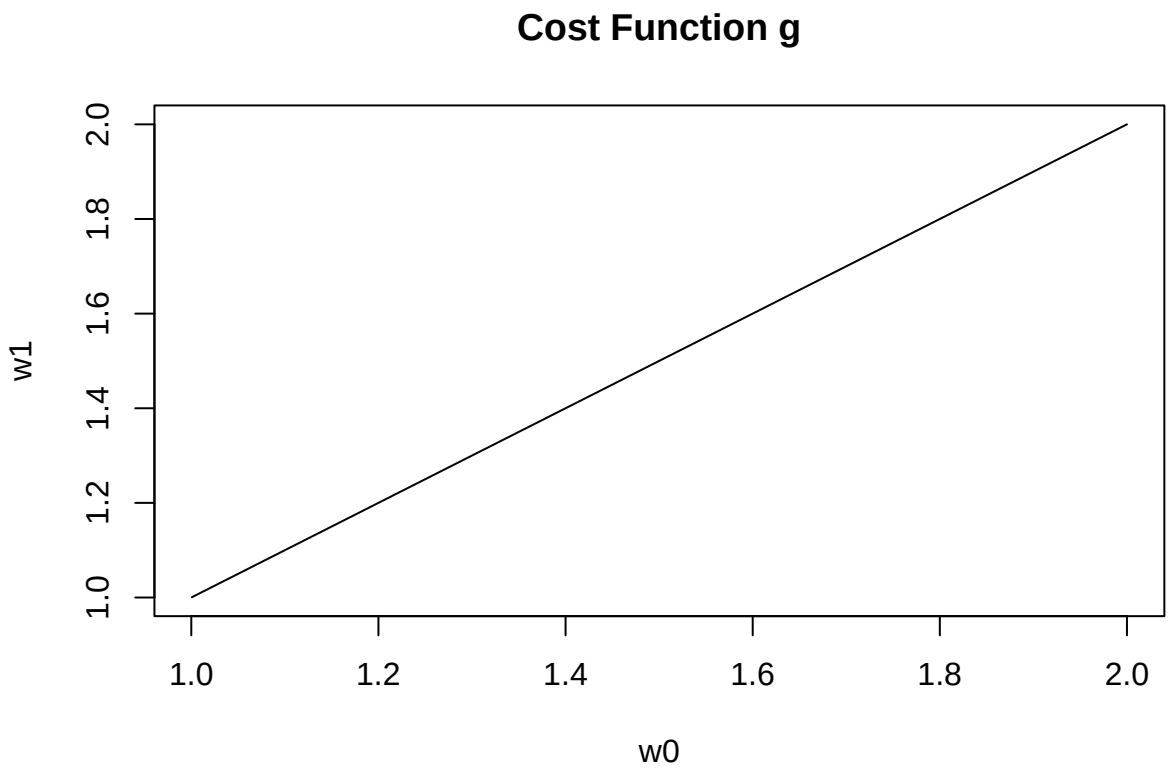
```
  c(2*w[1] - 2, 2*w[2] - 2)
}

g <- function(w) {
  ((w[1]^4 + w[2]^4)/4) * ((w[1]^3 + w[2]^3)/3) - w[1]^2 - w[2]^2 + 4
}

g_prime <- function(w) {
  c((w[1]^3 + w[2]^3)*w[1] - 2*w[1], (w[1]^3 + w[2]^3)*w[2] - 2*w[2])
}
alpha_choice <- 0.01
initial_point <- c(2, 2)
result_f <- gd(f, f_prime, alpha_choice = alpha_choice, w0 = initial_point)
plot(result_f$weight_history[1, ], result_f$weight_history[2, ],
     type = "l", xlab = "w0", ylab = "w1", main = "Cost Function f")
```



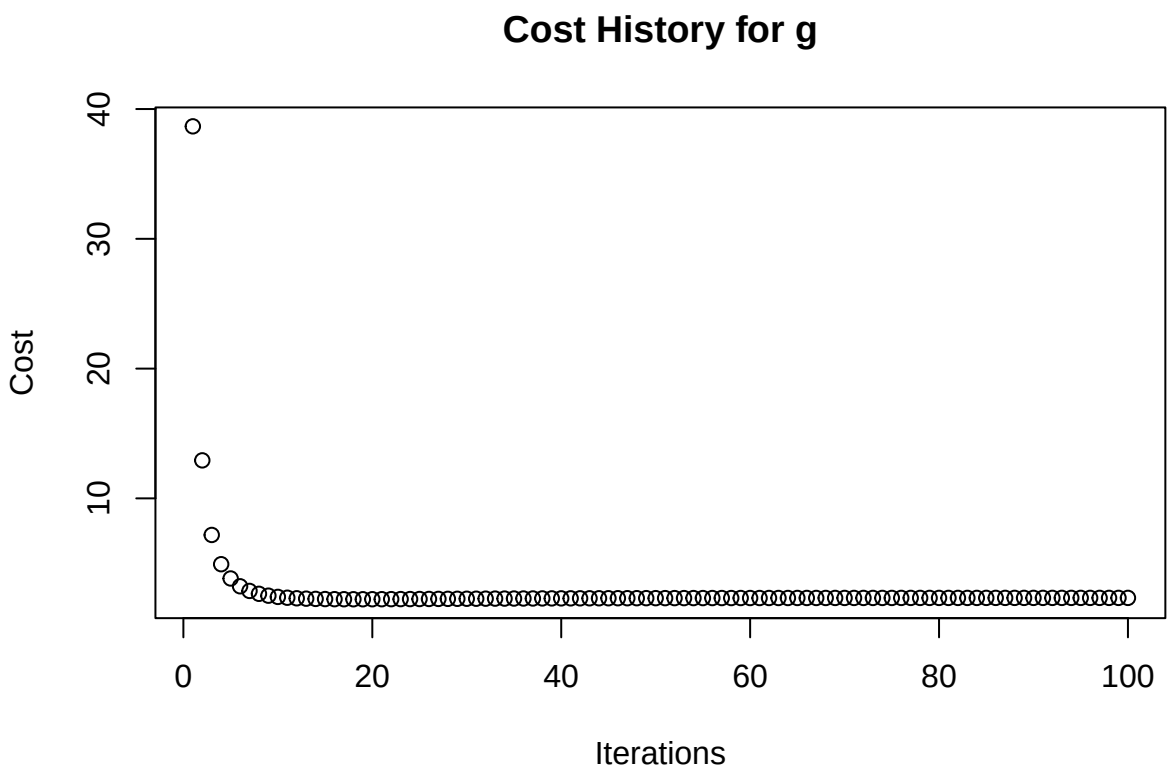

```
plot(result_f$cost_history,
     xlab = "Iterations", ylab = "Cost", main = "Cost History for f")
```




```
alpha_choice <- 0.01
initial_point <- c(2, 2)
result_g <- gd(g, g_prime, alpha_choice = alpha_choice, w0 = initial_point)

plot(result_g$weight_history[1, ], result_g$weight_history[2, ],
     type = "l", xlab = "w0", ylab = "w1", main = "Cost Function g")
```

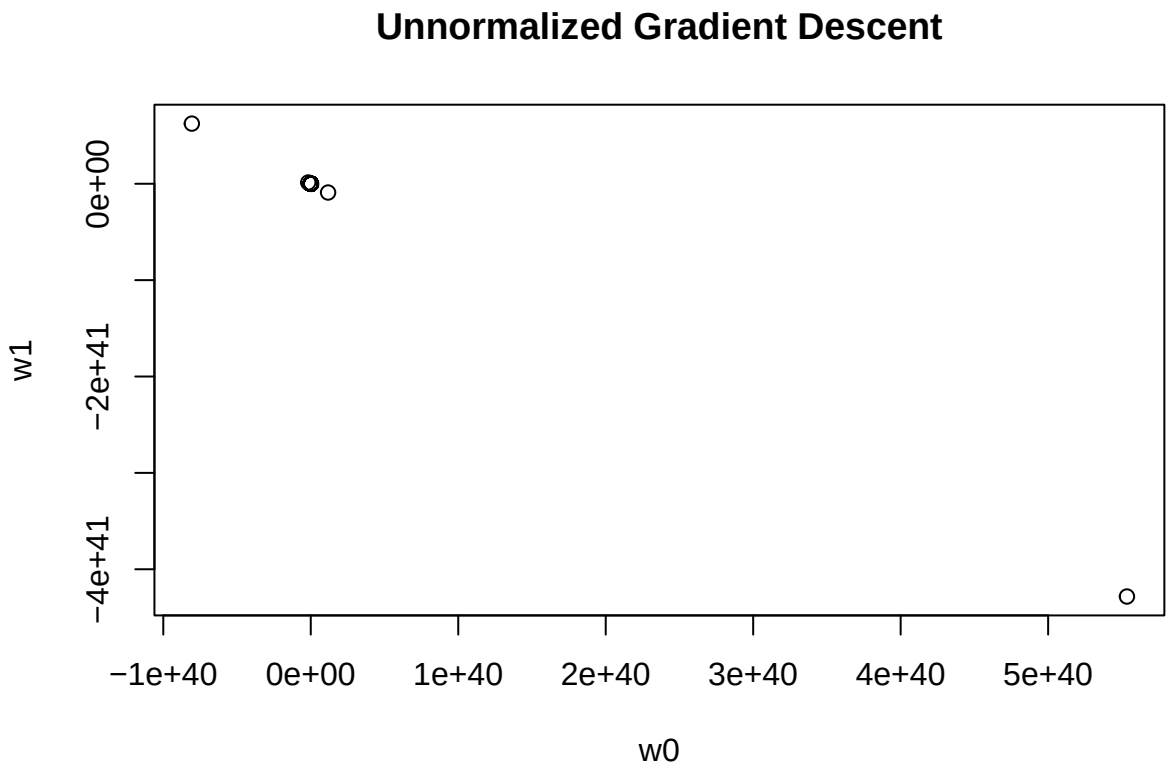



```
plot(result_g$cost_history,
     xlab = "Iterations", ylab = "Cost", main = "Cost History for g")
```

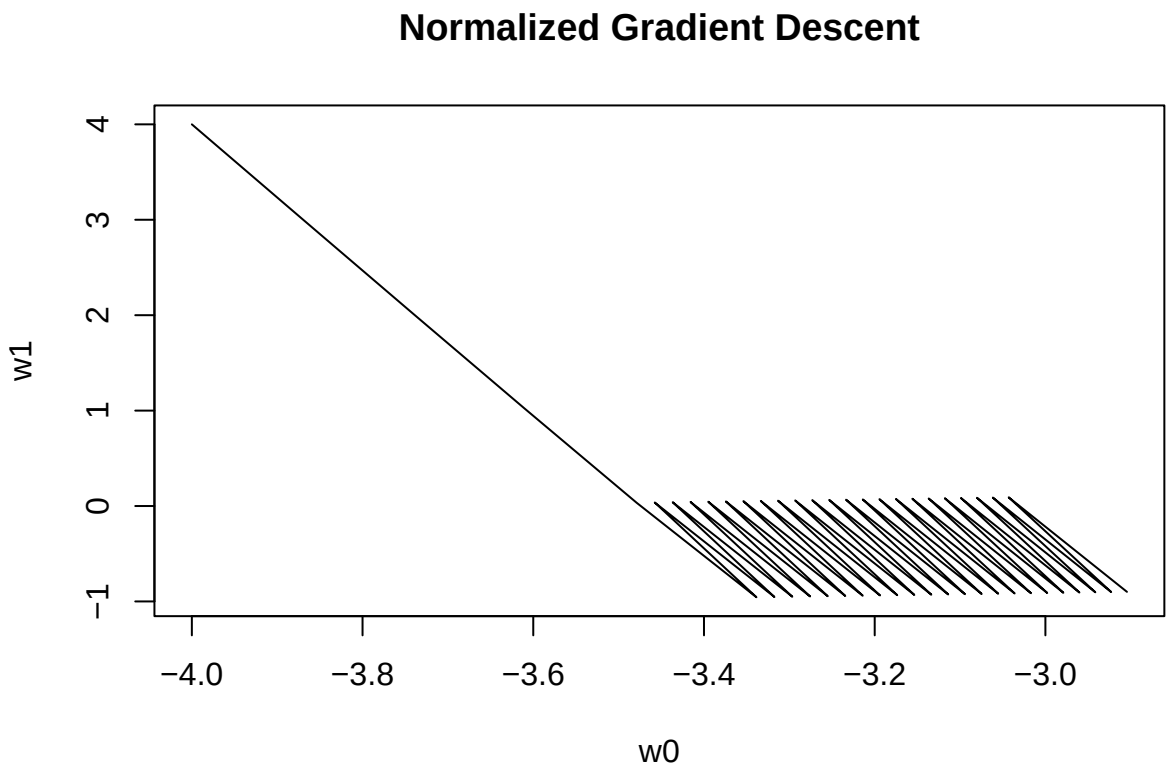



```
g  <- function(w) {
  (1/4 * w[1] - 2 * w[2])^2 + (1/4 * w[1] - 1/2 * w[2])^2
}

g_prime <- function(w) {
  c(2 * (1/4 * w[1] - 2 * w[2]) * (1/4), 2 * (1/4 * w[1] - 2 * w[2]) * (-2) +
       2 * (1/4 * w[1] - 1/2 * w[2]) * (1/4))
}

gd_modified <- function(g, g_prime, alpha_choice = 1, max_its = 100, w0 = 0, tol = 1e-5, normalized = 1
  alpha <- alpha_choice
  k <- 1
  w <- w0
  cost_history <- numeric(max_its)
  weight_history <- matrix(0, nrow = length(w), ncol = max_its)
  cost_history[1] <- g(w)
  weight_history[, 1] <- w
  alpha0 <- 1

  while (k < max_its & sqrt(sum(g_prime(w)^2)) > tol) {
    k <- k + 1
    if (alpha_choice == "diminishing") {
      alpha <- alpha0 / sqrt(k)
    }
    if (normalized == 1) {
      g_prime_norm <- g_prime(w) / sqrt(sum(g_prime(w)^2))
```

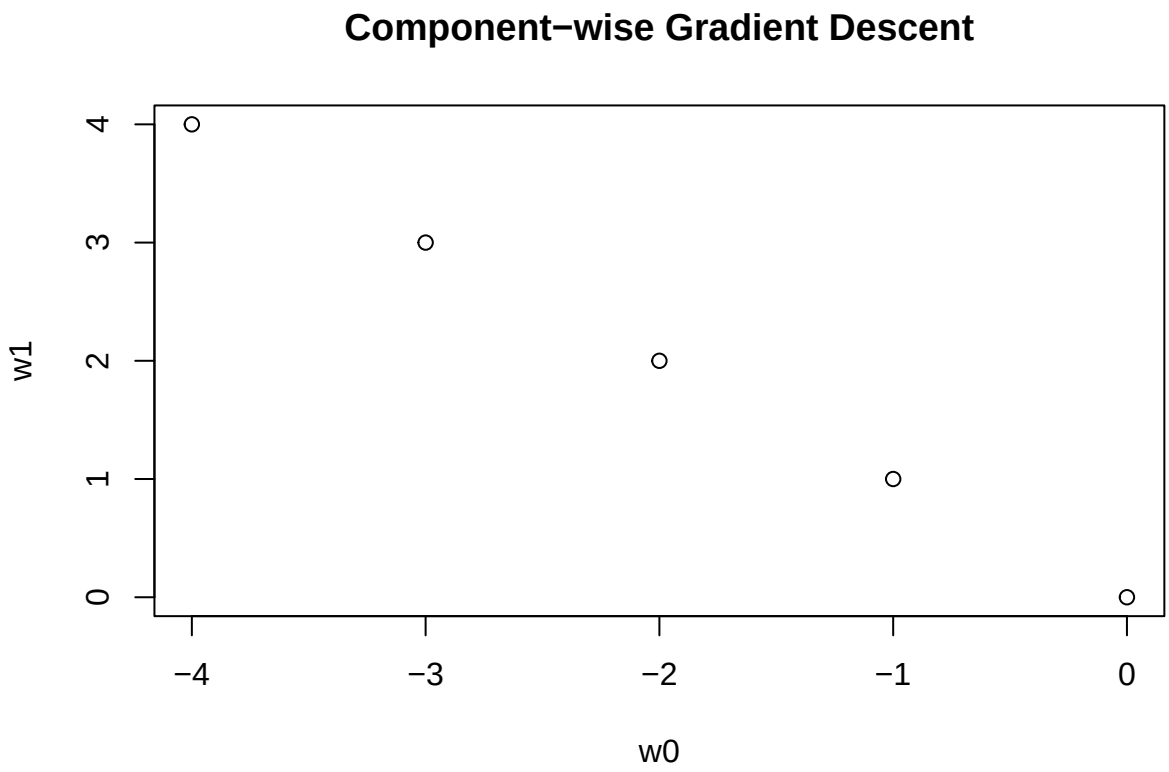
```
    } else if (normalized == -1) {
      g_prime_norm <- g_prime(w) / abs(g_prime(w))
    } else {
      g_prime_norm <- g_prime(w)
    }
    w <- w - alpha * g_prime_norm
    cost_history[k] <- g(w)
    weight_history[, k] <- w
  }

  list(weight_history = weight_history[, 1:k], cost_history = cost_history[1:k])
}


w0 <- c(-4, 4)
max_steps <- 50

result_standard <- gd_modified(g, g_prime, max_its = max_steps, w0 = w0, normalized = 0)
result_full_normalized <- gd_modified(g, g_prime, max_its = max_steps, w0 = w0, normalized = 1)
result_component_normalized <- gd_modified(g, g_prime, max_its = max_steps, w0 = w0, normalized = -1)

plot(result_standard$weight_history[1, ], result_standard$weight_history[2, ], xlab = "w0", ylab = "w1"
```




```
plot(result_full_normalized$weight_history[1, ], result_full_normalized$weight_history[2, ],type= "l", x
```




```
plot(result_component_normalized$weight_history[1, ], result_component_normalized$weight_history[2, ], t
```


Component−wise Gradient Descent
w1
0
1
2
3
4
−4
−3
−2
−1
0
w0