Solutions to Assignment 9

Rongfei Jin

April 4, 2025

1,2,3,5,6,7,13

Exercise 1

(a)

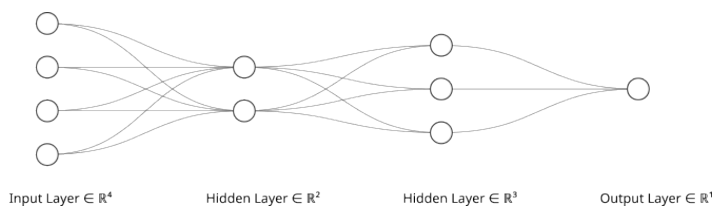

Figure 1

(b)

First hidden layer

$$A_k^{(1)} = (w_{k0}^{(1)} + \sum_{j=1}^{4} w_{kj}^{(1)} X_j)_+$$

$$\begin{aligned}
A_l^{(2)} &= (w_{l0}^{(2)} + \sum_{k=1}^{2} w_{lk}^{(2)} A_k^{(1)})_+ \\
A_l^{(2)} &= (w_{l0}^{(2)} + \sum_{k=1}^{2} w_{lk}^{(2)} (w_{k0}^{(1)} + \sum_{j=1}^{4} w_{kj}^{(1)} X_j)_+)_+
\end{aligned}$$

$$\begin{aligned}
Z &= (\beta_0 + \sum_{l=1}^{3} \beta_l A_l^{(2)})_+ \\
&= (\beta_0 + \sum_{l=1}^{3} \beta_l (w_{l0}^{(2)} + \sum_{k=1}^{2} w_{lk}^{(2)} (w_{k0}^{(1)} + \sum_{j=1}^{4} w_{kj}^{(1)} X_j)_+)_+)_+
\end{aligned}$$

$$
\begin{aligned}
f(X) &= \beta_0 + (\sum_{l=1}^{3} \beta_l A_l^{(2)})_+ \\
&= \beta_0 + (\sum_{l=1}^{3} \beta_l \sum_{j=1}^{2})(\beta_j)+
\end{aligned}
$$

(c)

```r
set.seed(42)
w_1 <- matrix(rnorm(4*2), nrow=4)
b_1 <- matrix(rnorm(2), ncol=2) # bias for the first layer, 2 neurons

w_2 <- matrix(rnorm(2*3), nrow=2)
b_2 <- matrix(rnorm(3), ncol=3)

z <- matrix(rnorm(3), nrow=3)
b_3 <- matrix(rnorm(1), ncol=1)

X <- matrix(c(1,2,3,4), ncol=4)

g <- function(x) {
  return(pmax(0, x))
}

a1 <- g(X %*% w_1 + b_1)

a1

a2 <- g(a1 %*% w_2 + b_2)

z <- g(a2 %*% z + b_3)

z
```

(d)

The total number of parameters in the network is the sum of the parameters in each layer.

$$(4+1)\cdot 2 + (2+1)\cdot 3 + (1+3) = 23$$

Exercise 2

(a)

$$
\begin{aligned}
\frac{e^{Z_m+c}}{\sum_{l=0}^{L} e^{Z_l+c}} &= \frac{e^c}{e^c} \cdot \frac{e^{Z_m}}{\sum_{l=0}^{L} e^{Z_l}} \\
&= \frac{e^{Z_m}}{\sum_{l=0}^{L} e^{Z_l}}
\end{aligned}
$$

(b)

$$
\begin{aligned}
\frac{e^{c_1+\cdots+c_p+\beta_{k0}+\beta_{k_1x_1}+\cdots+\beta_{k_px_p}}}{\sum_{l=1}^{K} e^{c_1+\cdots+c_p+\beta_{l0}+\beta_{l_1x_1}+\cdots+\beta_{k_px_p}}} &= \frac{e^{c_1+\cdots+c_p}\cdot e^{\beta_{k0}+\beta_{k_1x_1}+\cdots+\beta_{k_px_p}}}{e^{c_1+\cdots+c_p}\cdot\sum_{l=1}^{K} e^{\beta_{l0}+\beta_{l_1x_1}+\cdots+\beta_{l_px_p}}} \\
&= \frac{e^{\beta_{k0}+\beta_{k_1x_1}+\cdots+\beta_{k_px_p}}}{\sum_{l=1}^{K} e^{\beta_{l0}+\beta_{l_1x_1}+\cdots+\beta_{k_px_p}}}
\end{aligned}
$$

Exercise 3

$$
-\sum_{i=1}^{n}\sum_{i=1}^{2}\log(f_m(x_i))
$$

$$
\begin{aligned}
\ell(\beta_0,\beta_1) &= \prod_{i:y_1=1} p(x_i)\prod_{i:y_1=0}(1-p(x_i)) \\
\log(\ell(\beta_0,\beta_1)) &= \sum_{i:y_1=1}\log(p(x_i)) + \sum_{i:y_1=0}\log(1-p(x_i))
\end{aligned}
$$

Let $f_1 = p(x), f_2 = (1-p(x))$, y_{i1}, y_{i0} be the indicator function for $y_i = 1$ and $y_i = 0$ respectively. Then we can rewrite the log-likelihood as:

$$
\log(\ell(\beta_0,\beta_1)) = \sum^{i} y_{i1}\log(p(x_i)) + \sum^{i} y_{i0}\log(1-p(x_i))
$$

This is equivalent to the negative multinomial log-likelihood function

Exercise 5

$$
R^2 = 1 - \frac{\text{SSE}}{\text{SST}} = 1 - \frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2}
$$

$$
\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i-\hat{y}_i|
$$

MAE does not square the error, so it is less sensitive to outliers than MSE. In contrast, R^2 is a relative measure of fit that compares the variance explained by the model to the total variance in the data.

Exercise 6

(a)

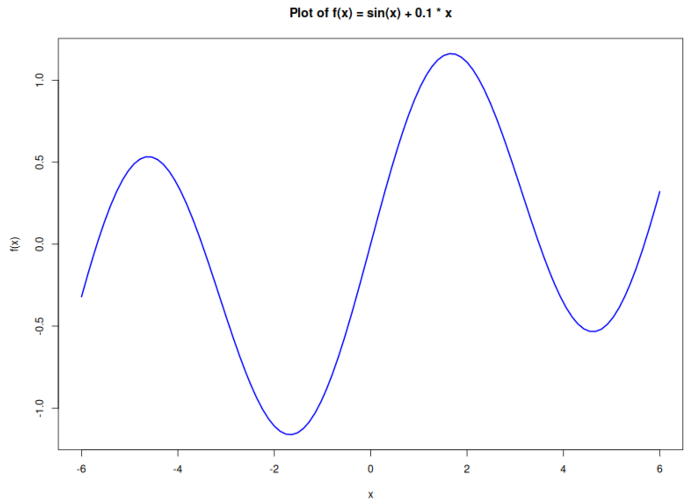

Figure 2

(b)

$$cos(\beta) + \frac{1}{10}$$

(c) and (d)

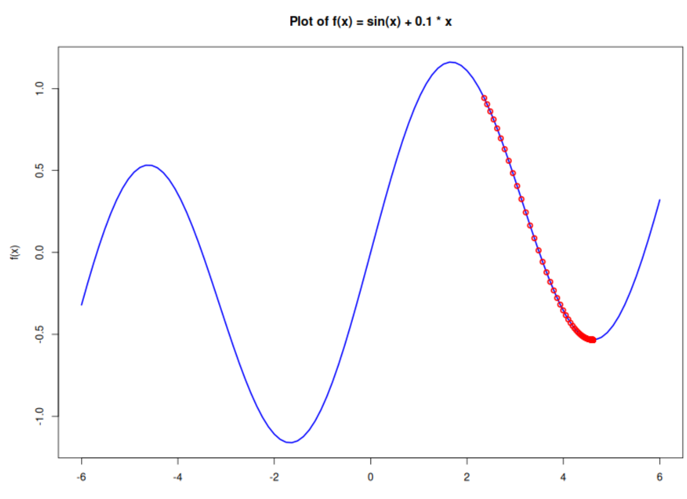

Figure 3

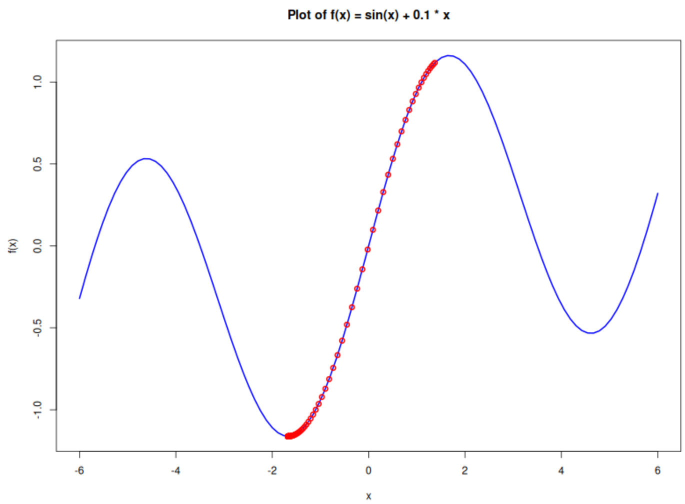

Figure 4

```
library(png)

f <- function(x) {
  sin(x) + 0.1 * x
}

f_prime <- function(x) {
  return(cos(x) + 0.1)
}

png("assignment9/figs/q6a.png", width=800, height=600)
x_vals <- seq(-6, 6, length.out=100)
y_vals <- f(x_vals)
plot(x_vals, y_vals, type='l', col='blue', lwd=2,
     main='Plot of f(x) = sin(x) + 0.1 * x',
     xlab='x', ylab='f(x)')
dev.off()

gradient_descent <- function(f, f_prime, x0, learning_rate=0.01, tolerance=1e-6, max_iter=1000) {
  hist <- c() # To store the history of x values
  x <- x0
  for (i in 1:max_iter) {
    x_new <- x - learning_rate * f_prime(x)
    hist <- c(hist, x_new)

    if (abs(x_new - x) < tolerance) {
      cat("Converged after", i, "iterations.\n")
      return(list(
        x = x,
        hist = hist
      ))
    }

    x <- x_new
  }

  cat("Max iterations reached without convergence.\n")
  return(list(
```

```
    x = x,
    hist = hist
  ))
}

sample_points <- function(f, x, percentage = 0.2) {
  # evenly sample percentage of points in x
  n <- length(x)
  if (n <= 1) {
    return(x)
  }

  # Calculate the number of points to sample
  num_points <- max(1, round(n * percentage))

  # Sample indices
  indices <- seq(1, n, length.out = num_points)

  # Return the sampled points
  sampled_x <- x[indices]
  return(data.frame(
    x = sampled_x,
    y = f(sampled_x)
  ))
}

d1 <- gradient_descent(
  f = f,
  f_prime = f_prime,
  x0 = 2.3, # Starting point
  learning_rate = 0.1,
  tolerance = 1e-6,
  max_iter = 1000
)

png("assignment9/figs/q6c.png", width=800, height=600)
# Plot the history of x values
plot(x_vals, y_vals, type='l', col='blue', lwd=2,
     main='Plot of f(x) = sin(x) + 0.1 * x',
     xlab='x', ylab='f(x)')
points(sample_points(f, d1$hist, 1), col='red', lwd=2,
     main='Convergence of Gradient Descent',
     xlab='Iteration', ylab='x value')
abline(h=d1$x, col='blue', lty=2)
text(x=0, y=d1$x-0.1,
     labels=paste("Converged to x =", round(d1$x, 4)), pos=4, col='blue')
dev.off()



d2 <- gradient_descent(
  f = f,
  f_prime = f_prime,
  x0 = 1.4, # Starting point
  learning_rate = 0.1,
  tolerance = 1e-6,
  max_iter = 1000
)
```

```r
png("assignment9/figs/q6d.png", width=800, height=600)
# Plot the history of x values scatter
plot(x_vals, y_vals, type='l', col='blue', lwd=2,
     main='Plot of f(x) = sin(x) + 0.1 * x',
     xlab='x', ylab='f(x)')
points(sample_points(f, d2$hist, 1), col='red', lwd=2,
     main='Convergence of Gradient Descent',
     xlab='Iteration', ylab='x value')
abline(h=d2$x, col='blue', lty=2)
text(x=0, y=d2$x-0.1,
     labels=paste("Converged to x =", round(d2$x, 4)), pos=4, col='blue')
dev.off()
```