Exercício 3.
Nome: Lucas Kou Kinoshita
RM: 2019021557
data: 01/04/2025

Ex.1
Implementação:

Inicialmente, define-se a função correspondente ao treinamento com o modelo adaline de forma simples, sem passo adaptativo e momentum:

```r
adaline <- function(X, y, eta, tol, maxepocas, par) {
  N <- nrow(X) # Número de linhas de X
  n <- ncol(X)
  error_curve <- numeric(maxepocas) # Vetor para armazenar o erro por época

  # Inicializando pesos
  if(par == 1){
    w <- as.matrix(runif(n+1) - 0.5)
    X <- cbind(1,X)
  } else {
    w <- as.matrix(runif(n) - 0.5)
  }

  nepocas <- 0
  erroepoca <- tol + 1 # Inicializa erro acima do limite para entrar no loop

  while (erroepoca > tol && nepocas < maxepocas) {
    xseq <- sample(N) # Embaralha os índices
    ei2 <- 0 # Inicializa erro da época

    for (i in xseq) {
      yhat <- 1*(t(w) %*%  X[i, ])
      erro <- y[i] - yhat
      dw <- eta * erro * X[i, ] # Atualização dos pesos
      w <- w + dw
      ei2 <- ei2 + erro^2 # Acumula erro quadrático
    }

    error_curve[nepocas] <- ei2 / N # Erro médio quadrático por época
    nepocas <- nepocas + 1
  }

  list(weights = w, error = error_curve[1:(nepocas - 1)])
}
```

Em seguida, inicializam-se as variáveis de input para o modelo correspondentes ao *dataset* fornecido pelo professor. E as entradas de teste (X_t) a partir de um sinal senoidal com tempo de amostragem mais rápido do dataset.

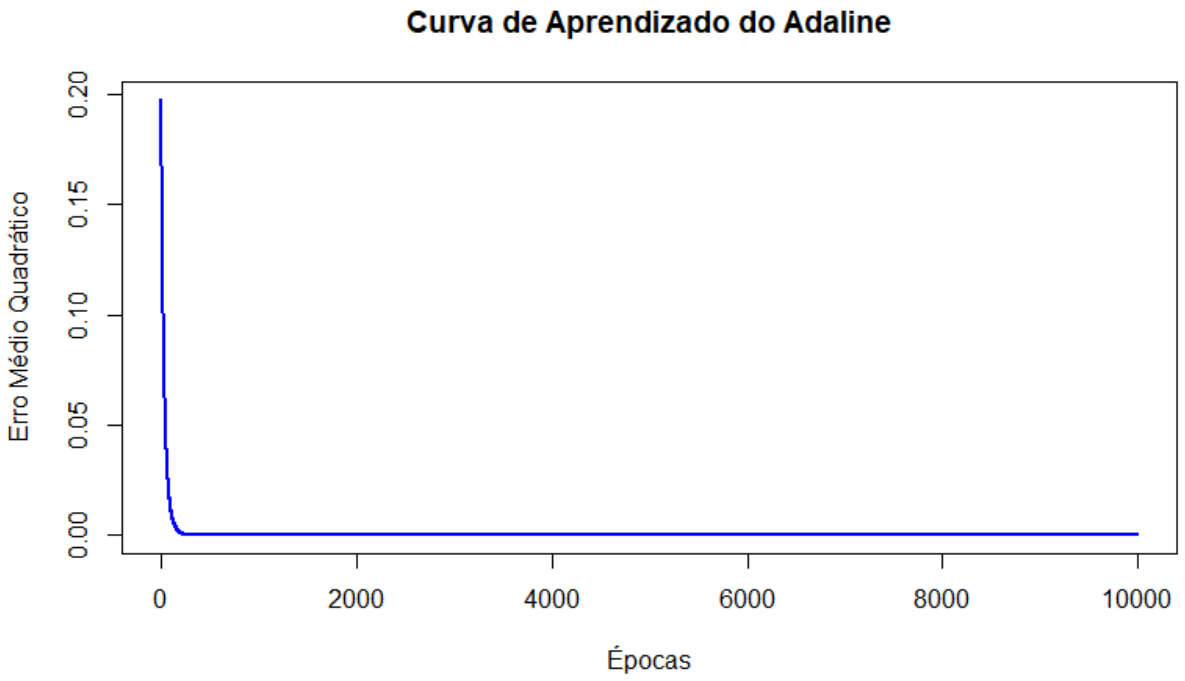
```r
# Leitura dos dados
t <- read.table('dados/Ex1_t', header = FALSE, sep = " ", fill = TRUE)
t <- na.omit(t)
t <- as.matrix(t[,-1])

x <- read.table('dados/Ex1_x', header = FALSE, sep = " ", fill = TRUE)
x <- na.omit(x)
X <- as.matrix(x[,-1])

# Gerando o sinal Xt com amostragem mais densa
t_seq <- seq(min(t), max(t), by = 0.2) # Amostragem mais fina
Xt <- sin((pi/3) * t_seq)  # Sinal seno
Xt <- as.matrix(Xt)

y <- read.table('dados/Ex1_y', header = FALSE, sep = " ", fill = TRUE)
y <- na.omit(y)
Y <- as.matrix(y[,-1])
```

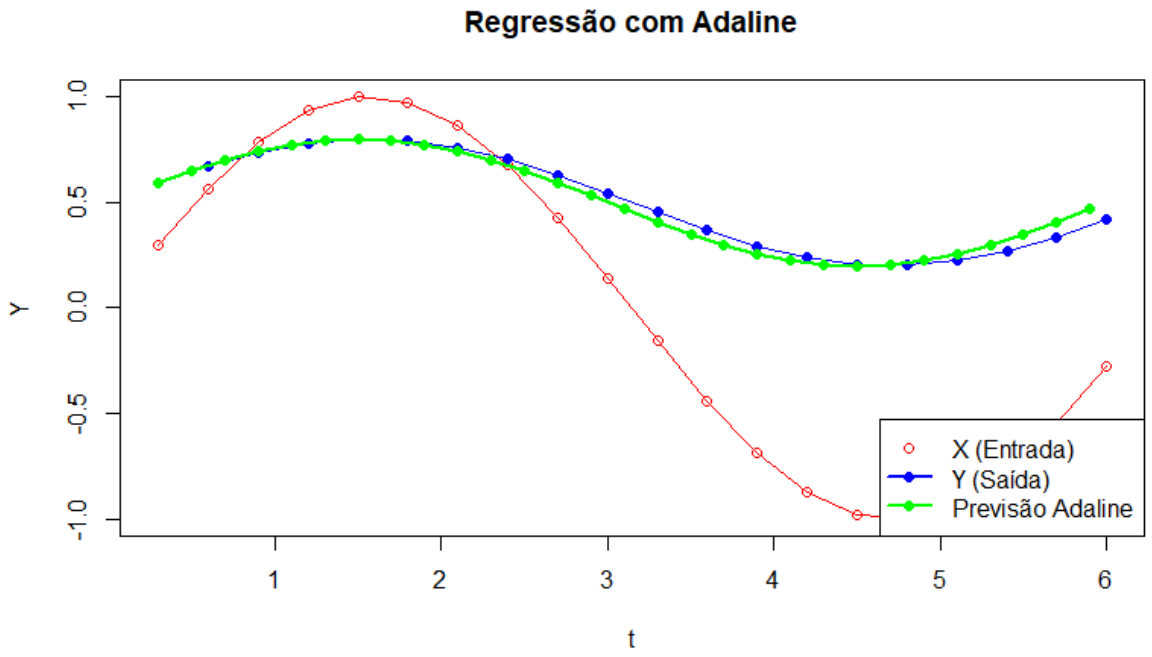
Preparam-se os parâmetros de treinamento para a função com modelo linear (Ht) correspondente à: $y = ax + b$:

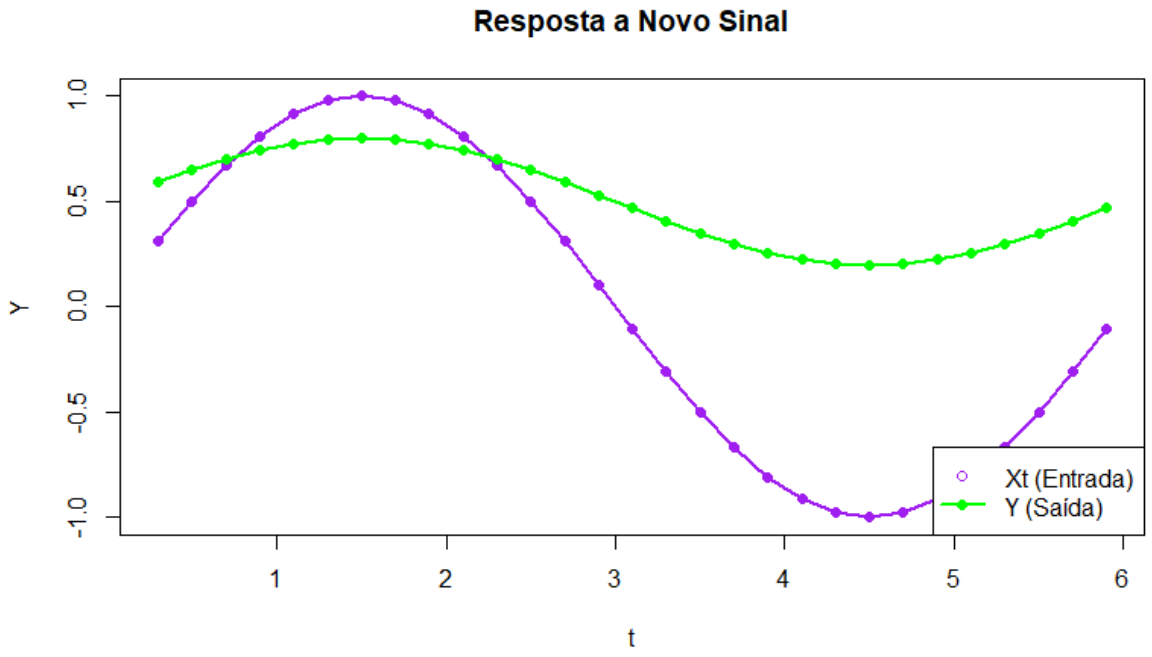
```r
# Preparando as matrizes de entrada
H <- cbind(X, 1)
Ht <- cbind(Xt, 1)

# Parâmetros do Adaline
eta <- 1e-3
tol <- 1e-2
maxepocas <- 10000
adaline_model <- adaline(H, Y, eta, tol, maxepocas, par = 0)

# Extrair resultados
final_weights <- adaline_model$weights
error_curve <- adaline_model$error
```

Por fim, o resultado alcançado pela função aproximada pelo modelo sobre o conjunto de testes (X_t) é comparado com o comportamento do dataset de treino (também plotam-se a curva de aprendizado correspondente ao erro e o conjunto de sinais de entrada/saída gerados durante o teste).

```r
# Plotando a curva de aprendizado
plot(error_curve, type = "l", col = "blue", lwd = 2,
     xlab = "Épocas", ylab = "Erro Médio Quadrático",
     main = "Curva de Aprendizado do Adaline")

# Fazendo previsões com os dados Xt
y_pred <- Ht %*% final_weights

# Plotando os resultados da previsão no gráfico original
plot(t, X, col = "red", pch = 1, xlab = "t",
     ylab = "Y", main = "Regressão com Adaline")
lines(t, X, col = 'red')
points(t, Y, col = "blue", pch = 16)
lines(t, Y, col = "blue", lwd = 1)

# Adicionando as previsões
points(t_seq, y_pred, col = "green", pch = 16)
lines(t_seq, y_pred, col = "green", lwd = 2)

# Adicionando legendas
legend("bottomright",
       legend = c("X (Entrada)", "Y (Saída)", "Previsão Adaline"),
       col = c("red", "blue", "green"),
       pch = c(1, 16, 16), lty = c(1, 1, 1), lwd = c(NA, 2, 2))

# Plotando a avaliação com dados de teste (Xt)
plot(t_seq, Xt, col = "purple", pch = 16, xlab = "t", ylab = "Y", main = "Resposta a Novo Sinal")
lines(t_seq, Xt, col = "purple", lwd = 2)
points(t_seq, y_pred, col = "green", pch = 16)
lines(t_seq, y_pred, col = "green", lwd = 2)
```



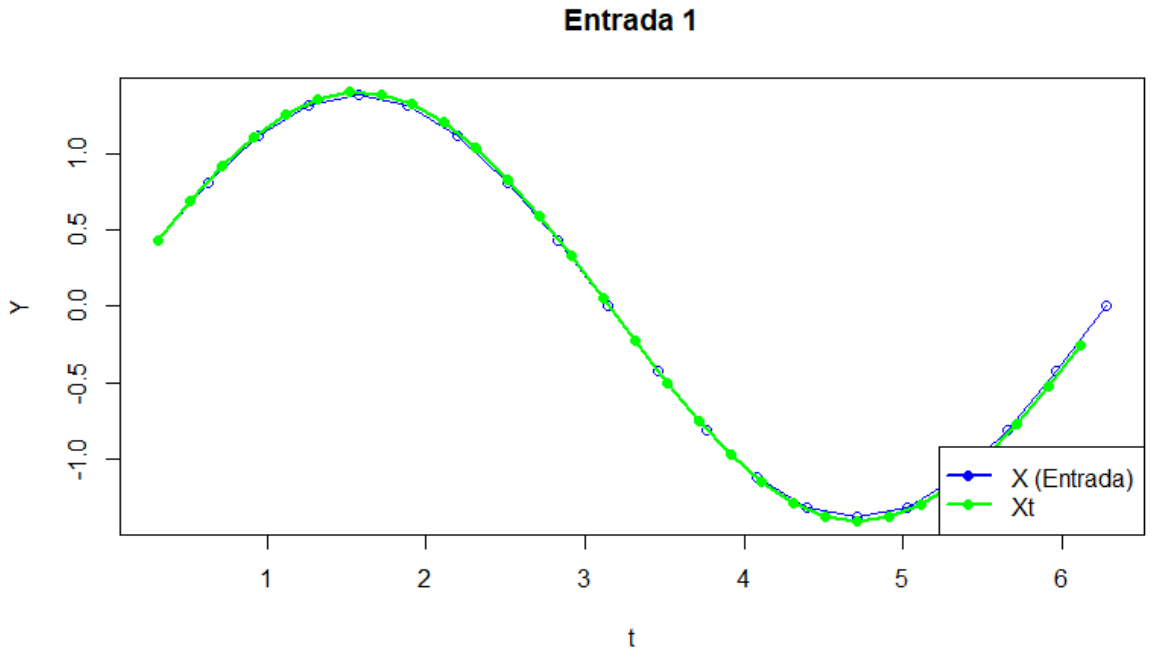

Regressão com Adaline
Y
t
X (Entrada)
Y (Saída)
Previsão Adaline

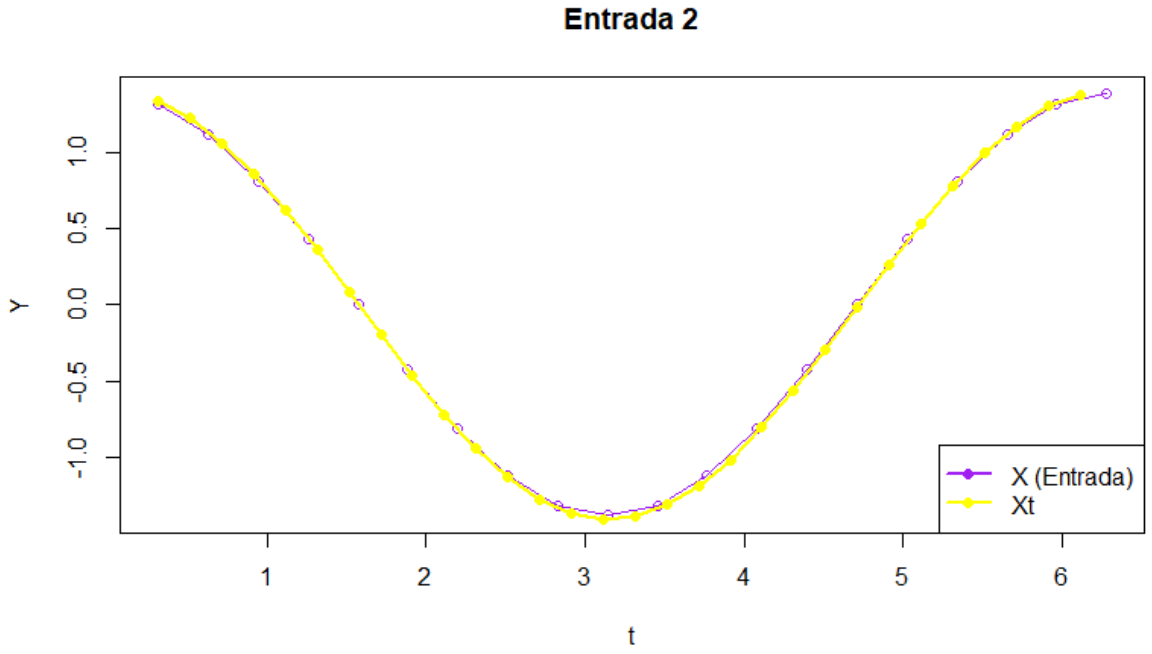

Resposta a Novo Sinal
Y
t
Xt (Entrada)
Y (Saída)

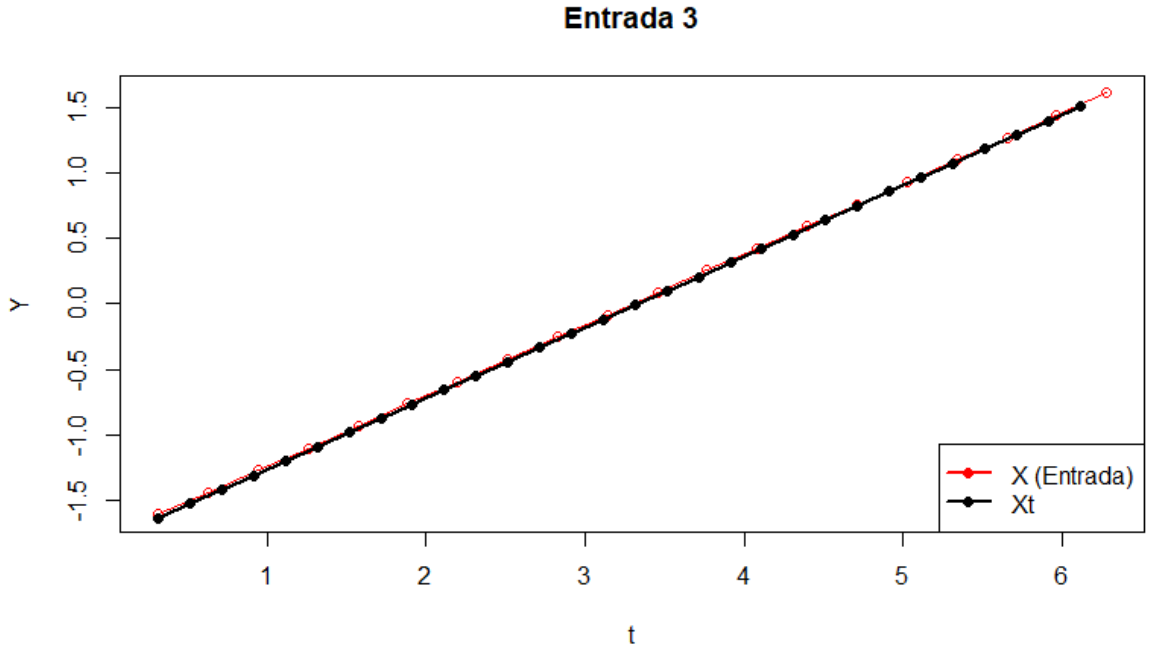
Exercício 2:
Implementação:

De forma semelhante ao primeiro exercício, define-se a função de treinamento do adaline e os dados são extraídos dos arquivos fornecidos pelo professor, em seguida os conjuntos de dados de teste foram aproximados dos dados fornecidos a partir de um tempo de amostragem menor, como pode ser observado nos gráficos de comparação gerados.

```
t <- as.matrix(read.table('dados/t'))
X <- as.matrix(read.table('dados/x'))
Y <- as.matrix(read.table('dados/y'))

t_seq <- seq(min(t), max(t), by = 0.2) # Amostragem mais fina
Xt1 <- 1.4 * sin(pi/3.15 * t_seq)
Xt1 <- as.matrix(Xt1)  # Convertendo para matriz

Xt2 <- 1.4* cos(pi/3.15 * t_seq)
Xt2 <- as.matrix(Xt2)

Xt3 <- 0.54*t_seq - 1.8  # Sinal seno
Xt3 <- as.matrix(Xt3)
```





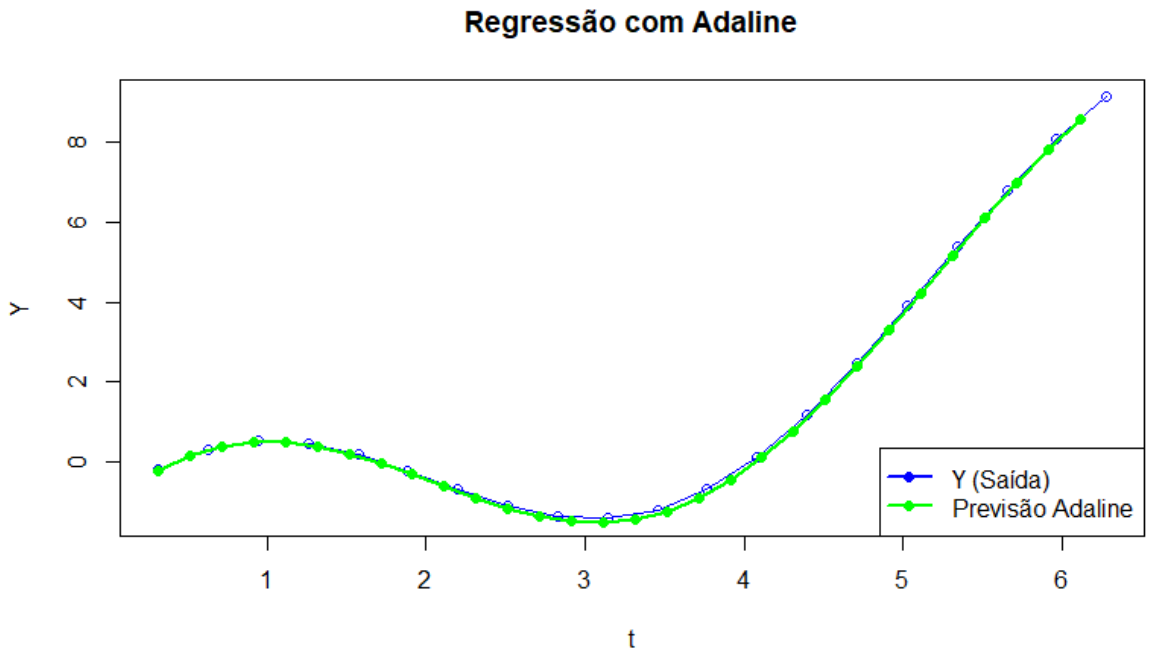


A matriz de entrada é definida a partir da adição da coluna correspondente ao termo polarizante e a matriz de teste é definida como a combinação linear dos conjuntos de teste e da coluna do termo polarizante. Definem-se os parâmetros para a função de treinamento e plota-se a curva de aprendizado.

```r
# Preparando as matrizes de entrada
H <- cbind(X , 1)
Ht <- cbind(Xt1, Xt2, Xt3, 1)

# Parâmetros do Adaline
eta <- 1e-3
tol <- 1e-2
maxepocas <- 10000
adaline_model <- adaline(H, Y, eta, tol, maxepocas, par = 0)

# Extrair resultados
final_weights <- adaline_model$weights
error_curve <- adaline_model$error

# Plotando a curva de aprendizado
plot(error_curve, type = "l", col = "blue", lwd = 2,
     xlab = "Épocas", ylab = "Erro Médio Quadrático",
     main = "Curva de Aprendizado do Adaline")

# Fazendo previsões com os dados Xt
y_pred <- Ht %*% final_weights
```

Por fim, a saída gerada pela função aproximada pelo adaline sobre o conjunto de testes é comparada à saída correspondente à função geradora.

```r
# Plotando os resultados da previsão no grá original
plot(t, Y, col = "blue", pch = 1, xlab = "t",
     ylab = "Y", main = "Regressão com Adaline")
lines(t, Y, col = 'blue')
# Adicionando as previsões
points(t_seq, y_pred, col = "green", pch = 16)
lines(t_seq, y_pred, col = "green", lwd = 2)

# Adicionando legendas
legend("bottomright",
       legend = c("Y (Saída)", "Previsão Adaline"),
       col = c("blue", "green"),
       pch = c(16, 16), lty = c(1, 1), lwd = c(2, 2))
```



Conclusão:

Em ambos os exercícios, através da análise visual, podemos concluir que o treinamento com o uso de neurônios *Adaline* foi capaz de aproximar com sucesso polinômios às funções geradoras.