```
time = c(0.1, 0.2, 0.1, 0.5, 0.02, 0.06)
temp = c(100, 100, 200, 200, 300, 300)
ca = c(0.98, 0.983, 0.544, 0.225, 0.566, 0.034)
# makes a matrix of the data
data = cbind(ca, time, temp)
# makes a linear model of the data
model = lm(ca ~time + temp)
# displays a summary of the model
summary(model)
# writes the summary to a different variable
summary_final <- summary(model)
# t-values
summary_final$coefficients[,3]
# f values
summary_final$fstatistic
# standard error
summary_final$coefficients[, 2]
# predicting values
predicted_ca = predict.lm(model)
# plotting predicted and actual values
plot(ca, predicted_ca, xlab = "Actual values of C_A", ylab = "Predicted Values of Ca",
type = "o")
View(Reactor_Data)
# making matrix from given excel data
model_reactor = cbind(Reactor_Data$`Inlet Temp`, Reactor_Data$Pressure,
Reactor_Data$`H2/HC ratio`, Reactor_Data$`Feed Flow`, Reactor_Data$C6A)
# dimensions of the data
dim(model_reactor)
# finding data type of a particular variable
typeof(Reactor_Data$`Inlet Temp`)
# doing pairs analysis on data
pairs(Reactor_Data)
# finding statistical correlation b/w all the variables in model
cor(Reactor_Data)
dim(model_reactor)
# making new matrix from existing data
selected_data = model_reactor[1:10, 1:5]
cor(Reactor_Data)
# making linear multivariable model
reactor_data_model = lm(Reactor_Data$C6A ~ Reactor_Data$`Inlet Temp` +
Reactor_Data$Pressure + Reactor_Data$`H2/HC ratio` + Reactor_Data$`Feed Flow`)
# anova analysis of data model
anova(reactor_data_model)
# coefs among all variables of the data model
coef(reactor_data_model)
# non linear model development:
ca= c(0.98, 0.983, 0.544, 0.225, 0.566, 0.034)
time= c(0.1, 0.2, 0.1, 0.5, 0.02, 0.06)
temp = c(100, 100, 200, 200, 300, 300)
df = cbind(ca, time, temp)
# importing external library
library(Metrics)
# log10(time), exp(temp) backchodi mein likha h, kuch bhi ho sakta h
# depends on model
model_nonlinear=nls(ca ~ c1*log10(time)+c2*exp(temp), start = list(c1=1, c2=0))
summary(model_nonlinear)
coef(model_nonlinear)
predict(model_nonlinear)
# final expression of NL model
ca_model_nl = -8.93*0.1*log10(time)-5.172*10^-131*exp(temp)
ca_model_nl
plot(ca, ca_model_nl, type = "o")
rmse(ca, ca_model_nl)   # error analysis using rmse method
mape(ca, ca_model_nl)   # error analysis using mape analysis
data_nonlinearmodel=cbind(ca, ca_model_nl, (ca-ca_model_nl))   # exporting results
write.csv(data_nonlinearmodel, file= "data_nonlinearmodel.csv")
# model optimization using R for parameter optimization
# case 1: single-parameter (coefficient: c1) optimization of developed model
```

```
OF = function(c1) {                                  # created objective function (OF)
which is (model-experiment)^2
  ca_model11 = ((1.017*time-c1*temp) -ca)^2
  return(ca_model11)
}
c1= 0
d=nlminb(c1, OF, lower = -Inf, upper = Inf)    #use nlminb function for optimization
# get the optimised value of c1
d$par
# get the optimised value of objective function
d$objective
# optimal number of iterations to solve
d$iterations
# Case 2: multi-parameters (coefficients: c1 and C2) optimization of the developed model
OF = function(C) {
  ca_model12 = (C[1]*time-C[2]*temp - ca)^2
  return(ca_model12)
}
C= c(0,0)
d=nlminb(C, OF, lower = -Inf, upper = Inf)
d$par
d$objective
# ode solving
library(deSolve)
CSTR <- function(t, state, parameters) {
  with(as.list(c(state, parameters)), {
    dh =  a + b*h^0.5
    list(c(dh))
  })
}
parameters <- c(a = 3, b = -1)
state      <- c(h = 0)
time      <- seq(0, 200, by = 10)
Sol_cstr <- ode(t = time, y = state, parms = parameters, func = CSTR)
plot(Sol_cstr)
Sol_cstr
Sol_cstr[11,2] # height of the tank at 90 minutes is 9 m
# multiple ODEs at once
library(deSolve)
CSTR_series <- function(t, state, parameters) {
  with(as.list(c(state, parameters)), {
    dca1 = a+ b*ca1+c*(ca1*cb1)                  # ODE form for tank-1, A-component
    dcb1 = d+ b*cb1+c*(ca1*cb1)                 # ODE form for tank-2, B-component
    dca2 = -b*ca1+ b*ca2+c*(ca2*cb2)             # ODE form for tank-1, A-component
    dcb2 = -b*cb1+ b*cb2+c*(ca2*cb2)             # ODE form for tank-2, B-component
    list(c(dca1, dcb1, dca2, dcb2))
  })
}
parameters <- c(a = 2, b = -0.5, c= -0.3, d= 2)   # a, b, c, d are the parameters
calculated for given data (i.e. V, Caf, Ff, k)
state      <- c(ca1 = 2, cb1=2, ca2= 0, cb2= 0)   # initial conditions for tank-1 (Ca1,
cb1) and tank-2 (ca2, cb2)
time      <- seq(0, 40, by = 1)
Sol_cstr_series <- ode(t = time, y = state, parms = parameters, func = CSTR_series)
plot(Sol_cstr_series)
Sol_cstr_series
plot(Sol_cstr_series)
Sol_cstr_series[11, 2]   # predict the concentration of "Ca1" at 10th minutes.
Sol_cstr_series[11, 4]
```