**上机作业二**

1. 写出用二分法解一元非线性方程的程序，并完成P101第1题。

# P101-1 dichotomy

dizero <- function(f, a, b, eps = 1e-6){

  if(f(a)\*f(b) > 0)

    list(fail = "finding root is fail!")

  else{

    repeat{

      if(abs(b-a) < eps) break;

      x <- (a+b)/2;

      if(f(a)\*f(x) < 0) b <- x else a <- x

    }

    list(root = x, fun = f(x))

  }

}

f1 <- function(x) 2\*x^3 - 6\*x - 1;

f2 <- function(x) exp(x-2) + x^3 - x;

f3 <- function(x) 1 + 5\*x - 6\*x^3 - exp(2\*x);

> u <- seq(-2, 2, 0.05);

> v <- f1(u);

> plot(u, v, type = "l")

> # [-2,-1],[-1,0],[1,2]

> uniroot(f1, c(-2,-1), tol = 1e-6)$root

[1] -1.641784

> dizero(f1, -2, -1, 1e-6)$root

[1] -1.641784

> uniroot(f1, c(-1,0), tol = 1e-6)$root

[1] -0.1682544

> dizero(f1, -1, 0, 1e-6)$root

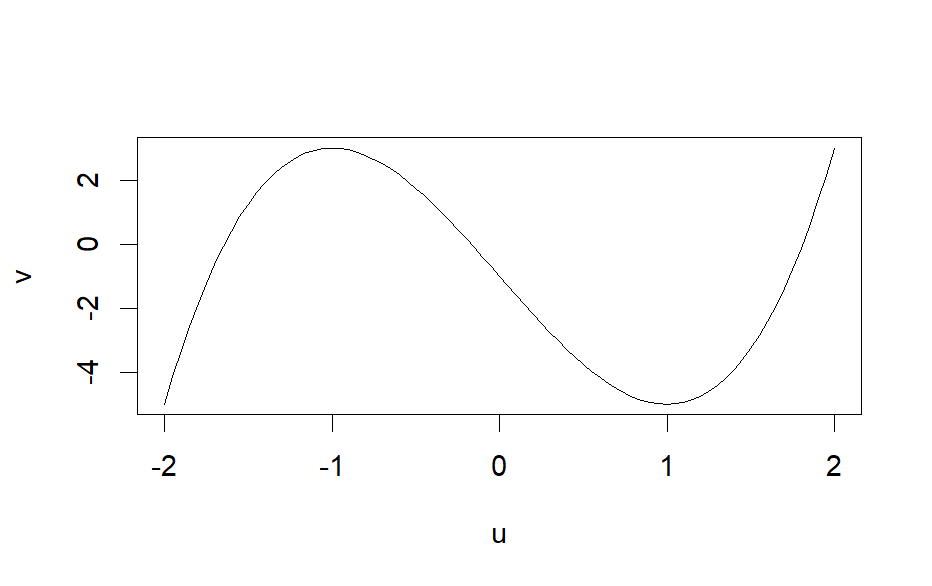
[1] -0.1682539

> uniroot(f1, c(1,2), tol = 1e-6)$root

[1] 1.810038

> dizero(f1, 1, 2, 1e-6)$root

[1] 1.810039



> u <- seq(-1.5, 1.5, 0.05);

> v <- f2(u);

> plot(u, v, type = "l")

> # [-1.5,-0.5],[-0.5,0.5],[0.5,1.5]

> uniroot(f2, c(-1.5,-0.5), tol = 1e-6)$root

[1] -1.023482

> dizero(f2, -1.5, -0.5, 1e-6)$root

[1] -1.023482

> uniroot(f2, c(-0.5,0.5), tol = 1e-6)$root

[1] 0.1638222

> dizero(f2, -0.5, 0.5, 1e-6)$root

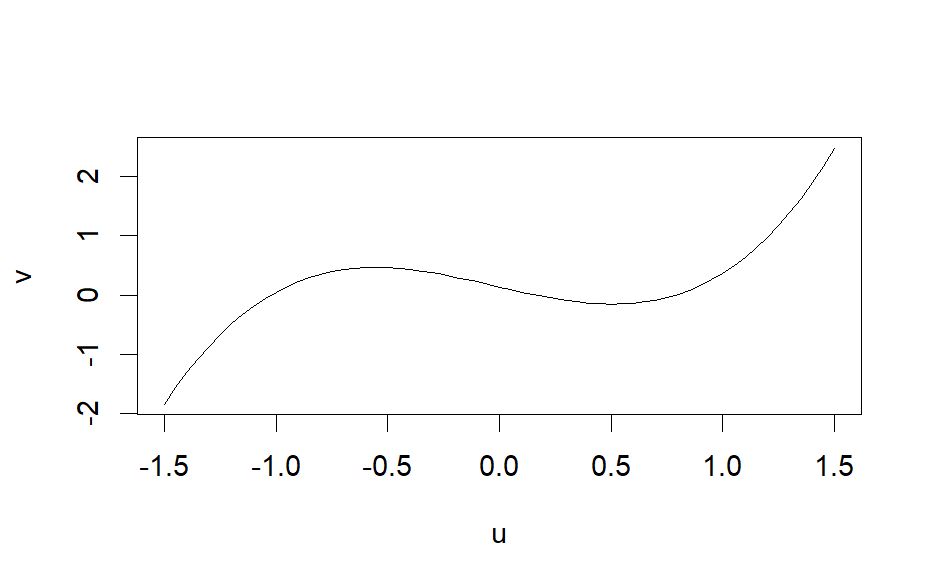
[1] 0.1638231

> uniroot(f2, c(0.5,1.5), tol = 1e-6)$root

[1] 0.7889413

> dizero(f2, 0.5, 1.5, 1e-6)$root

[1] 0.7889414



> u <- seq(-1, 1, 0.05);

> v <- f3(u);

> plot(u, v, type = "l")

> # [-1.5,-0.5],[-0.6,0.4],[0.5,1.5]

> uniroot(f3, c(-1.5,-0.5), tol = 1e-6)$root

[1] -0.8180937

> dizero(f3, -1.5, -0.5, 1e-6)$root

[1] -0.8180933

> uniroot(f3, c(-0.6,0.4), tol = 1e-6)$root

[1] 5.06246e-09

> dizero(f3, -0.6, 0.4, 1e-6)$root

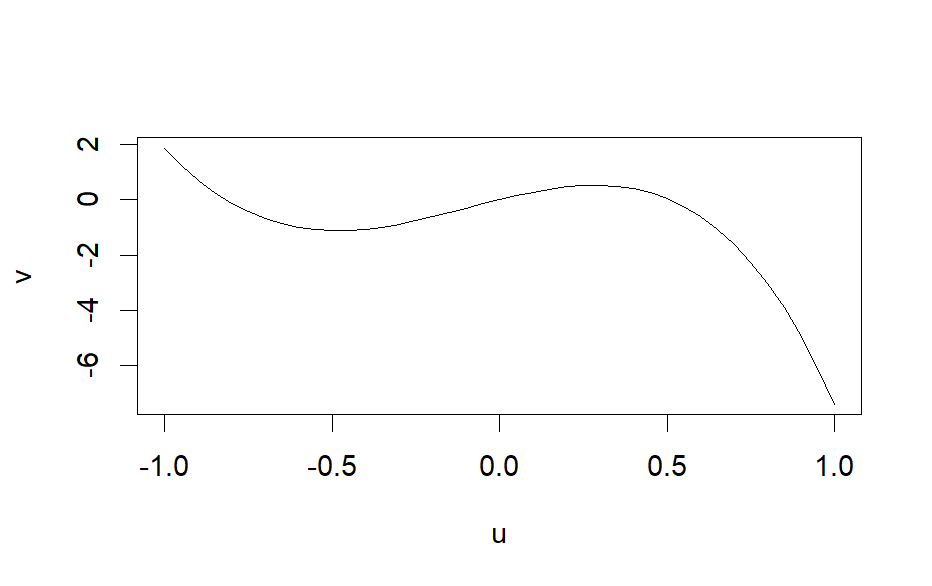
[1] -5.722046e-07

> uniroot(f3, c(0.5,1.5), tol = 1e-6)$root

[1] 0.5063083

> dizero(f3, 0.5, 1.5, 1e-6)$root

[1] 0.5063086



1. 写出用Newton法解多元非线性方程的程序，并完成P101第3题。

# P101-3 Newton

Newtons <- function(f, x0, eps, max\_iter = 50){

  x <- x0

  for (i in 1:max\_iter){

    result <- f(x)

    fx <- result$f

    if (max(abs(fx)) < eps){

      return(list(root = x, iter = i))

    }

    x <- x - fx%\*%solve(result$J)

}

return(list(root = x, iter = i))

}

funs <- function(x){

  f <- c(x[1]^2 + x[2]^2 - 1,

         x[1]^3 - x[2])

  J <- matrix(c(2\*x[1], 2\*x[2],

              3\*x[1]^2, -x[2]), nrow = 2, byrow = T)  # derivative

  list(f = f, J = J)

}

Newtons(funs, c(-0.8, 0.6), 1e-3)

> Newtons(funs, c(-0.8, 0.6), 1e-3)

$root

[,1] [,2]

[1,] 0.8264179 0.5634237

$iter

[1] 26

1. 完成P101第6题。

注意：Newton法初始值不能离解太远

# P101-6 binomial distribution

n <- 10; p <- 0.2;

samples <- rbinom(100, n, p);

funs <- function(n, p, moments){

  f <- c(n\*p - moments[1],

         n\*p\*(1-p) - moments[2])  # origin Moment

  J <- matrix(c(p, n,

                p\*(1-p), n\*(1-2\*p)),

              nrow = 2, byrow = T)  # derivative

  list(f = f, J = J)

}

Newtons <- function(f, x0, epsn, epsp, max\_iter = 10){

  x <- x0

  for (i in 1:max\_iter){

    result <- f(x)

    fx <- result$f

    if (abs(fx[1]) < epsn && abs(fx[2]) < epsp){

      return(list(root = x, iter = i))

    }

    x <- x - solve(result$J)%\*%fx

  }

  return(list(root = x, iter = i))

}

moments <- c(mean(samples), var(samples));

funs\_moment <- function(x) funs(x[1], x[2], moments);

Newtons(funs\_moment, c(10, 0.2), 1e-3, 1e-5)

> Newtons(funs\_moment, c(10, 0.2), 1e-3, 1e-5)

$root

[,1]

[1,] 9.5511463

[2,] 0.2109485

$iter

[1] 4

1. 用R编程求出含有数字6却不能被6整除的5位数有多少个。

sum <- 0;

for(i in 10000:99999){

  flag <- FALSE;

  n <- i;

  for(j in 1:5){

    if(n%%10 == 6){

      flag <- TRUE;

      break

    }

    n <- n%/%10;

  }

  if(flag == TRUE){

    if(i%%5 != 0) sum <- sum + 1;

  }

}

sum

[1] 31176

1. 在一副标准的52张扑克中,4种花色个13张。用模拟的方法求出从这副扑克中随机取出两张，这两张是同一种花色的概率。

fetch <- function(){

  a <- sample(1:52, 1)

  b <- sample(1:52, 1)

  if((a-1)%/%13 == (b-1)%/%13) return(TRUE)

  else return(FALSE)

}

N <- 10000;

sum <- 0;

for(i in 1:N){

  if(fetch() == TRUE) sum = sum + 1;

}

p <- sum/N; p

[1] 0.2536