Package 'NLRoot'

February 19, 2015

Type Package

Title searching for the root of equation

Version 1.0

Date 2012-07-05

Author Zheng Sengui,Lu Xufen,Hou Qiongchen,Zheng Jianhui

Maintainer Zheng Sengui<1225620446@qq.com>

Description This is a package which can help you search for the root of a equation.

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-07-06 16:21:52

NeedsCompilation no

R topics documented:

`BFfzero` *Bisection Method*

Description

Bisection Method to Find the Root of Nonlinear Equation

Usage

```
BFfzero(f, a, b, num = 10, eps = 1e-05)
```

Arguments

`f`	the objective function which we will use to solve for the root
`a`	mininum of the interval which cantains the root from Bisection Method
`b`	maxinum of the interval which cantains the root from Bisection Method
`num`	the number of sections that the interval which from Bisection Method
`eps`	the level of precision that |x(k+1)-x(k)| should be satisfied in order to get the idear real root. eps=1e-5 when it is default

Details

Be careful to choose a & b. If not we maybe fail to find the root

Value

a root of the objective function which between the interwal from a to b

Note

Maintainer:Zheng Sengui<1225620446@qq.com>

Author(s)

Zheng Sengui,Lu Xufen,Hou Qiongchen,Zheng Jianhui

References

Luis Torgo (2003) Data Mining with R:learning by case studies. LIACC-FEP, University of Porto

See Also

`NDHfzero`,`NIMfzero`,`SMfzero`

Examples

```
f<-function(x){x^3-x-1};f1<-function(x){3*x^2-1};

    BFfzero(f,0,2)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (f, a, b, num = 10, eps = 1e-05)
{
    h = abs(b - a)/num
    i = 0
    j = 0
    a1 = b1 = 0
    while (i <= num) {
        a1 = a + i * h
```

```
            b1 = a1 + h
            if (f(a1) == 0) {
                print(a1)
                print(f(a1))
            }
            else if (f(b1) == 0) {
                print(b1)
                print(f(b1))
            }
            else if (f(a1) * f(b1) < 0) {
                repeat {
                    if (abs(b1 - a1) < eps)
                      break
                    x <- (a1 + b1)/2
                    if (f(a1) * f(x) < 0)
                      b1 <- x
                    else a1 <- x
                }
                print(j + 1)
                j = j + 1
                print((a1 + b1)/2)
                print(f((a1 + b1)/2))
            }
            i = i + 1
        }
        if (j == 0)
            print("finding root is fail")
        else print("finding root is successful")
      }
```

`NDHfzero` *Newton Downhill Method*

Description

Newton Downhill Method to Find the Root of Nonlinear Equation

Usage

```
NDHfzero(f, f1, x0 = 0, num = 1000, eps = 1e-05, eps1 = 1e-05)
```

Arguments

`f`	the objective function which we will use to solve for the root
`f1`	the derivative of the objective function (say f)
`x0`	the initial value of Newton iteration method or Newton downhill method
`num`	num the number of sections that the interval which from Brent's method devide into. num=1000 when it is default

`eps`	the level of precision that |x(k+1)-x(k)| should be satisfied in order to get the idear real root. eps=1e-5 when it is default
`eps1`	the level of precision that |f(x)| should be satisfied, where x comes from the program. when it is not satisified we will fail to get the root

Details

eps1 of precision that |f(x)| should be satisfied, where x comes from the program. when it is not satisified we will fail to get the root

Value

a root of the objective function

Note

Maintainer:Zheng Sengui<1225620446@qq.com>

Author(s)

Zheng Sengui,Lu Xufen,Hou Qiongchen,Zheng Jianhui

References

Luis Torgo (2003) Data Mining with R:learning by case studies. LIACC-FEP, University of Porto

See Also

`BFfzero`,`NIMfzero`,`SMfzero`

Examples

```
f<-function(x){x^3-x-1};f1<-function(x){3*x^2-1};
NDHfzero(f,f1,2)

##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (f, f1, x0 = 0, num = 1000, eps = 1e-05, eps1 = 1e-05)
{
    a = x0
    b = a - f(a)/f1(a)
    i = 0
    while ((abs(b - a) > eps)) {
        c = 1
        j = 0
        while (abs(f(b)) >= abs(f(a))) {
            b = a - c * f(a)/f1(a)
            j = j + 1
            c = 1/(2^j)
        }
        a = b
        b = a - f(a)/f1(a)
        c = 1
        j = 0
        while (abs(f(b)) >= abs(f(a))) {
            b = a - c * f(a)/f1(a)
            j = j + 1
            c = 1/(2^j)
        }
        i = i + 1
    }
    print(b)
    print(f(b))
    if (abs(f(b)) < eps1) {
        print("finding root is successful")
    }
    else print("finding root is fail")
  }
```

`NIMfzero` *Newton iteration method*

Description

Newton iteration method to Find the Root of Nonlinear Equation.

Usage

```
NIMfzero(f, f1, x0 = 0, num = 100, eps = 1e-05, eps1 = 1e-05)
```

Arguments

`f`	the objective function which we will use to solve for the root
`f1`	the derivative of the objective function (say f)
`x0`	the initial value of Newton iteration method or Newton downhill method
`num`	the number of sections that the interval which from Brent's method devide into. num=100 when it is default
`eps`	the level of precision that |x(k+1)-x(k)| should be satisfied in order to get the idear real root. eps=1e-5 when it is default
`eps1`	the level of precision that |f(x)| should be satisfied, where x comes from the program. when it is not satisified we will fail to get the root

Details

the root we found out is based on the x0. So it is better to choose x0 carefully

Value

the root of the function

Note

Maintainer:Zheng Sengui<1225620446@qq.com>

Author(s)

Zheng Sengui,Lu Xufen,Hou Qiongchen,Zheng Jianhui

References

Luis Torgo (2003) Data Mining with R:learning by case studies. LIACC-FEP, University of Porto

See Also

BFfzero,NDHfzero,SMfzero

Examples

```
f<-function(x){x^3-x-1};f1<-function(x){3*x^2-1};
NIMfzero(f,f1,0)

##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as


function (f, f1, x0 = 0, num = 100, eps = 1e-05, eps1 = 1e-05)
{
    a = x0
    b = a - f(a)/f1(a)
    i = 0
    while ((abs(b - a) > eps) & (i < num)) {
        a = b
        b = a - f(a)/f1(a)
        i = i + 1
    }
    print(b)
    print(f(b))
    if (abs(f(b)) < eps1) {
        print("finding root is successful")
    }
    else print("finding root is fail")
  }
```

`SMfzero` *Secant Method*

Description

Secant Method to Find the Root of Nonlinear Equation.

Usage

```
SMfzero(f, x1, x2, num = 1000, eps = 1e-05, eps1 = 1e-05)
```

Arguments

`f`	the objective function which we will use to solve for the root
`x1`	the initial value of Secant Method
`x2`	the initial value of Secant Method
`num`	the number of sections that the interval which from Brent's method devide into. num=1000 when it is default
`eps`	the level of precision that |x(k+1)-x(k)| should be satisfied in order to get the idear real root. eps=1e-5 when it is default
`eps1`	the level of precision that |f(x)| should be satisfied, where x comes from the program. when it is not satisified we will fail to get the root

Details

Be careful to choose x1 & x2.if not we maybe fail to get the root

Value

the root of the function

Note

Maintainer:Zheng Sengui<1225620446@qq.com>

Author(s)

Zheng Sengui,Lu Xufen,Hou Qiongchen,Zheng Jianhui

References

Luis Torgo (2003) Data Mining with R:learning by case studies. LIACC-FEP, University of Porto

See Also

`BFfzero`,`NDHfzero`,`NIMfzero`

Examples

```
f<-function(x){x^3-x-1};f1<-function(x){3*x^2-1};
SMfzero(f,0,2)


##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as


function (f, x1, x2, num = 1000, eps = 1e-05, eps1 = 1e-05)
{
    i = 0
    while ((abs(x1 - x2) > eps) & (i < num)) {
        c = x2 - f(x2) * (x2 - x1)/(f(x2) - f(x1))
        x1 = x2
        x2 = c
        i = i + 1
    }
    print(x2)
    print(f(x2))
    if (abs(f(x2)) < eps1) {
        print("finding root is successful")
    }
    else print("finding root is fail")
  }
```

Index