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 .......................................................... 1
- NDHfzero ......................................................... 3
- NIMfzero ........................................................... 5
- SMfzero ............................................................. 7

Index

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 minimum of the interval which contains the root from Bisection Method
b maximum of the interval which contains 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 ideal 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 interval from a to b

Note

Maintainer:Zheng Sengui<1225620446@qq.com>

Author(s)

Zheng Sengui,Lu Xufen,Hou Qiongchen,Zheng Jianhui

References


See Also

NDHfzero,NIMfzero,SMfzero

Examples

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

Bffzero(f,0,2)

```r
# 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
```
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 ideal real root. \( \text{eps}=1e-5 \) 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 satisfied 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 satisfied 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**


**See Also**

`bffzero`, `nimfzero`, `smfzero`

**Examples**

```r
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 
        }
        i = i + 1 
    }
    return(b)
}
```
nimfzero

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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>f</td>
<td>the objective function which we will use to solve for the root</td>
</tr>
<tr>
<td>f1</td>
<td>the derivative of the objective function (say f)</td>
</tr>
<tr>
<td>x0</td>
<td>the initial value of Newton iteration method or Newton downhill method</td>
</tr>
<tr>
<td>num</td>
<td>the number of sections that the interval which from Brent’s method devide into. num=100 when it is default</td>
</tr>
<tr>
<td>eps</td>
<td>the level of precision that</td>
</tr>
<tr>
<td>eps1</td>
<td>the level of precision that</td>
</tr>
</tbody>
</table>

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


See Also

bffzero, ndhfzero, smfzero

Examples

```r
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 satisfied 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


See Also

BFFzero,NDHfzero,NIMfzero
Examples

\[
\begin{align*}
&f\leftarrow \text{function}(x)(x^3-x-1); f1\leftarrow \text{function}(x)(3x^2-1); \\
&\text{SMfzero}(f,0,2)
\end{align*}
\]

```r
# Should be DIRECTLY executable !! MMMM
# 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

*Topic \textasciitilde kwd1
  Bffzero, 1
  NDHfzero, 3
  NIMfzero, 5
  SMfzero, 7
*Topic \textasciitilde kwd2
  Bffzero, 1
  NDHfzero, 3
  NIMfzero, 5
  SMfzero, 7

Bffzero, 1, 4, 6, 7

NDHfzero, 2, 3, 6, 7
NIMfzero, 2, 4, 5, 7

SMfzero, 2, 4, 6, 7