Chief Editor
Dr. A. Singaraj, M.A., M.Phil., Ph.D.
Editor
Mrs.M.Josephin Immaculate Ruba

Editorial Advisors

1. Dr.Yi-Lin Yu, Ph. D
   Associate Professor,
   Department of Advertising & Public Relations,
   Fu Jen Catholic University,
   Taipei, Taiwan.
2. Dr.G. Badri Narayanan, PhD,
   Research Economist,
   Center for Global Trade Analysis,
   Purdue University,
   West Lafayette,
   Indiana, USA.
3. Dr. Gajendra Naidu.J., M.Com, LL.M., M.B.A., PhD. MHRM
   Professor & Head,
   Faculty of Finance, Botho University,
   Gaborone Campus, Botho Education Park,
   Kgale, Gaborone, Botswana.
4. Dr. Ahmed Sebihi
   Associate Professor
   Islamic Culture and Social Sciences (ICSS),
   Department of General Education (DGE),
   Gulf Medical University (GMU), UAE.
5. Dr. Pradeep Kumar Choudhury,
   Assistant Professor,
   Institute for Studies in Industrial Development,
   An ICSSR Research Institute,
   New Delhi- 110070.India.
6. Dr. Sumita Bharat Goyal
   Assistant Professor,
   Department of Commerce,
   Central University of Rajasthan,
   Bandar Sindri, Dist-Ajmer,
   Rajasthan, India
7. Dr. C. Muniyandi, M.Sc., M. Phil., Ph. D,
   Assistant Professor,
   Department of Econometrics,
   School of Economics,
   Madurai Kamaraj University,
   Madurai-625021, Tamil Nadu, India.
8. Dr. B. Ravi Kumar,
   Assistant Professor
   Department of GBEH,
   Sree Vidyanikethan Engineering College,
   A.Rangampet, Tirupati,
   Andhra Pradesh, India
9. Dr. Gyanendra Awasthi, M.Sc., Ph.D., NET
   Associate Professor & HOD
   Department of Biochemistry,
   Dolphin (PG) Institute of Biomedical & Natural Sciences,
   Dehradun, Uttarakhand, India.
10. Dr. D.K. Awasthi, M.SC., Ph.D.
    Associate Professor
    Department of Chemistry, Sri J.N.P.G. College,
    Charbagh, Lucknow,
    Uttar Pradesh. India

ISSN (Online) : 2455 - 3662
SJIF Impact Factor :3.967

EPRA International Journal of

Multidisciplinary Research

Monthly Peer Reviewed & Indexed International Online Journal

Volume: 2 Issue: 12 December 2016

Published By : EPRA Journals

CC License

EPRA International Journal of Multidisciplinary Research (IJMR) ISSN (Online): 2455-3662

SJIF Impact Factor: 3.967 Volume: 2 | Issue: 12 |December 2016

SOLVING NONLINEAR EQUATIONS: RECIPES IN R

Sreeja K[1]

[1]Assistant professor,
C M S College,
Kottayam,
Kerala, India.

Sindhu Thomas[2]

[2]Assistant professor,
C M S College,
Kottayam, Kerala, India

ABSTRACT

Nonlinear equations have got numerous statistical applications in various fields such as biology, physics, and electronics. Here we have discussed different iterative methods helpful in finding approximations of roots of such equations. Comparisons of some of these methods along with their implementation in R language has been demonstrated. Packages and functions available in R for this purpose have been discussed.

KEY WORDS: *Nonlinear Equations, R Programming Language, Bisection Method, Newton-Raphson Method, Fixed Point Iteration Method*

INTRODUCTION

Non linear equations- importance of study

Nonlinear equations are helpful in describing fundamental phenomena in physical and biological systems and are of immense applications in communication systems as well. Many problems in engineering areas can be expressed as finding roots of some non linear equations. Unfortunately, there are no exhaustive methods for finding all the roots of such equations in general. In this context, study of numerical methods and implementation of these methods in various programming languages have become more relevant.

R programming language

R is the open-source version of the language S known for its applications in statistical analysis and graphics. R is a high-level language which performs complicated calculations and makes quality graphics.

R has numerous functions which can be applied on matrices, and is suitable for numerical integration, and implement different statistical tools, performdatavisualisation, model problems mathematically.

Iterative vs analytical methods for root finding

In root finding problems, direct methods or analytical methods attempt to solve the equation by delivering exact root. The process will be done by a finite number of operations. However, in general, non linear equations can not be solved using analytical methods. Iterative methods are implemented in such situations. These methods use an initial value to generate better approximations to a solution. Iterative methods are useful even for linear equations involving large number of variables where it will be time consuming to implement analytical methods.

Bisection Method: the famous bracketing method

The Bisection method is the simplest method to find a root of an equation, linear or nonlinear.The method is done by halving the interval in which the root lies. So before using this method, we have to identify the initial interval which will contain a root of the given equation.The method systematically reduces the interval by halving the interval and one half is selected after performing a simple test.The procedure is repeated till therequired interval length is obtained.

If the function f(x) is continuous on [a,b] and f(a) and f(b) have opposite signs, Bisection method presents a smaller interval that is half of the current interval such that the function has opposite signs at the end points of the interval. The procedure is repeated to reduce the length of the interval. The algorithm is based on the assumptions that f(x) is continuous on [a,b] and $f(a)\, f(b) < 0$

The Newton-Raphson method

Newton–Raphson method is named after Isaac Newton and Joseph Raphson. It is a method for obtaining better approximations to the roots of f(x)=0, where f(x) is a real valued function.

The Newton–Raphson method for one unknown can be explained as follows:

Given a function f on R and its derivative f', we begin with a first approximation x_0 for a root of the equation f(x) = 0.a better approximation x_1 is given by the formula

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_{0)}}$$

Geometrically, (x_1, 0) is the intersection with the x-axis of the tangent to f at (x_0, $f(x_0)$).

The process is repeated as $xx_{n+1} = x_n - \frac{f(x_n)}{f'(x_{n)}}x$until a sufficiently accurate value is reached.

The method can also be extended to systems of equations.

Fixed Point Method

This method tries to find the roots of an equation by finding fixed point of a function obtained from the given equation. In this method we first express an equation f(x)=0 in the form x=g(x) and tries to find the fixed points of g(x). The function g(x) is known as the iteration function.

Comparison of Root finding Methods

Method	Advantages	Disadvantages
Bisection Method	●Simple and easy to implement ●function is evaluated only once in each iteration ●method reduces the size of the interval containing the zero in each step. ●The function need not be differentiable	●convergence might be slow ●Good intermediate approximations may be discarded
Fixed Point Method	●existence of convergence criterion.	●The convergence depends on the choice of iteration function ●The method may not converge at all. ●The convergence may be slow.
Newton Raphson Method	●It is a convenient method if the derivative can be obtained analytically ●Rate of convergence is quadratic	●It does not always converge ●There is no convergence criterion

Implementing Algorithms Using R

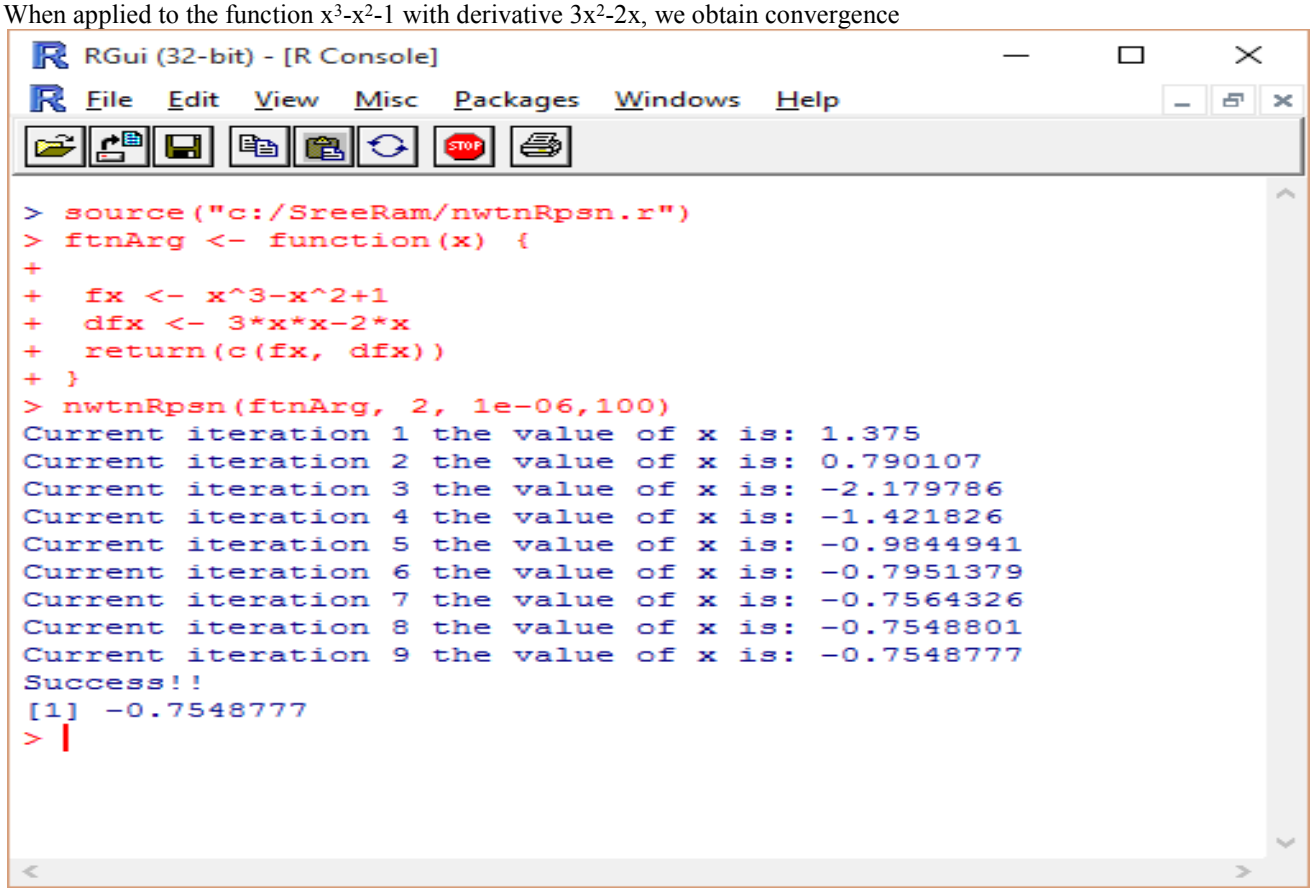
Newton raphson method

Algorithm is applied using a function nwtnRpsn.

```
nwtnRpsn<- function(fn, a, err, maxIt) {
 x <- a
 fx<- fn(x)
 i<-  0

 while ((abs(fx[1]) >err) && (i<maxIt)) {
  x <- x - fx[1]/fx[2]
  fx<- fn(x)
  i<- i + 1
  cat("Current iteration", iteration, "the value of x is:", x, "\n")
 }

 if (abs(fx[1]) >err) {
```

```
  cat("Failed\n")
  return(NULL)
 } else {
  cat("Success!!\n")
  return(x)
 }
}
```

When applied to the function x^3-x^2-1 with derivative $3x^2-2x$, we obtain convergence

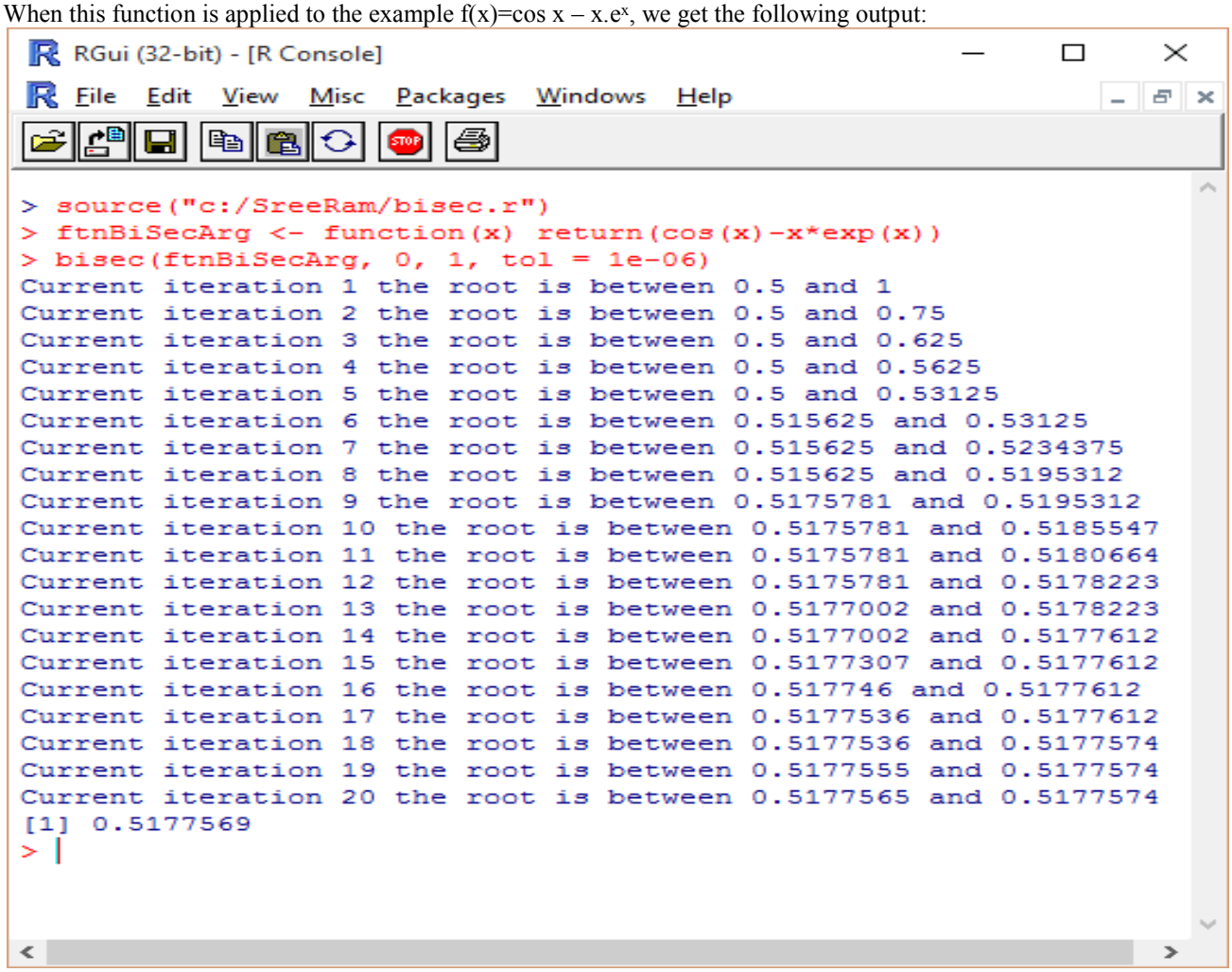


Implementing Bisection method

```
bisec<- function(ftn, a, b, tolerence = 1e-9) {

 # check inputs
 if (a >= b) {
  cat("problem : a >= b \n")
  return(NULL)
 }
 fa<- ftn(a)
 fb<- ftn(b)
 if (fa == 0) {
  return(a)
 } else if (fb == 0) {
  return(b)
 } else if (fa * fb > 0) {
  cat("problem: ftn(a) * ftn(b) > 0 \n")
  return(NULL)
 }

 # successively refine a and b
 n <- 0
 while ((b - a) >tolerence) {
```

```
  m <- (a + b)/2
  fm<- ftn(m)
  if (fm == 0) {
    return(xm)
  } else if (fa * fm< 0) {
    b <- m
    fb<- fm
  } else {
    a <- m
    fa<- fm
  }
  n <- n + 1
  cat("Current iteration", n, "the root is between", a, "and", b, "\n")
 }


 return((a + b)/2)
}
```

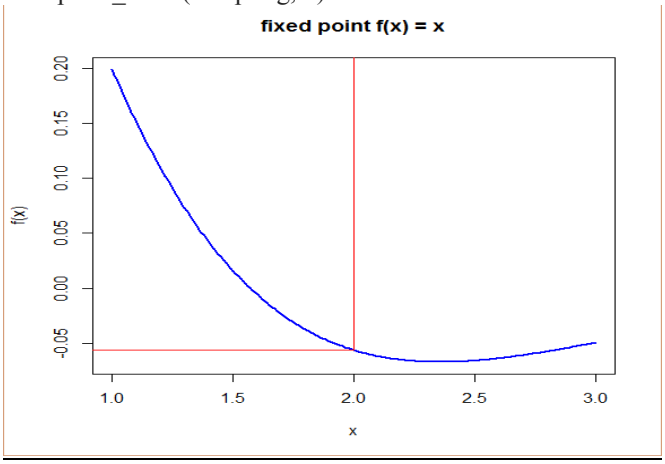
When this function is applied to the example f(x)=cos x – x.e^x, we get the following output:



Locating Roots Using Packages In R

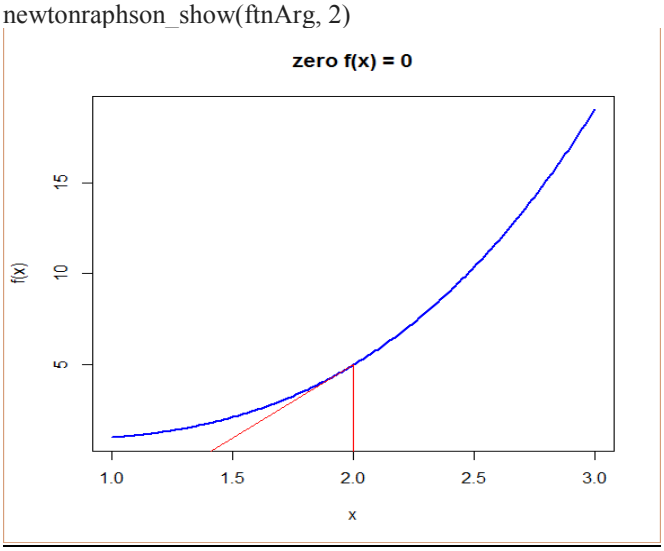
Using spuRs Package
Fixedpoint_Show Function:

```
ftnFpArg<- function(x) return(cos(x)/exp(x))
fixedpoint_show(ftnFpArg, 2)
```



Newtonraphson_show function:

```
ftnArg<- function(x) {
fx<- x^3-x^2+1
dfx<- 3*x*x-2*x
return(c(fx, dfx))
}
newtonraphson_show(ftnArg, 2)
```

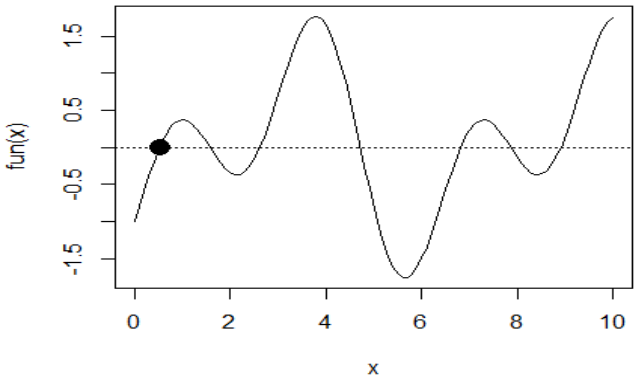


Using Rootsolve Package

This package contains root finding algorithms for solving nonlinear equations using Newton Raphson Method. It contains an extesnsion uniroot.all of the function uniroot from the base package. The function uniroot obtains only one root of an equation whereas polyroot helps to find complex roots of a polynomial.

To locate root of the equation f(x)=sin 2x- cos x, in the interval [0,10] and plot the curve, we write:

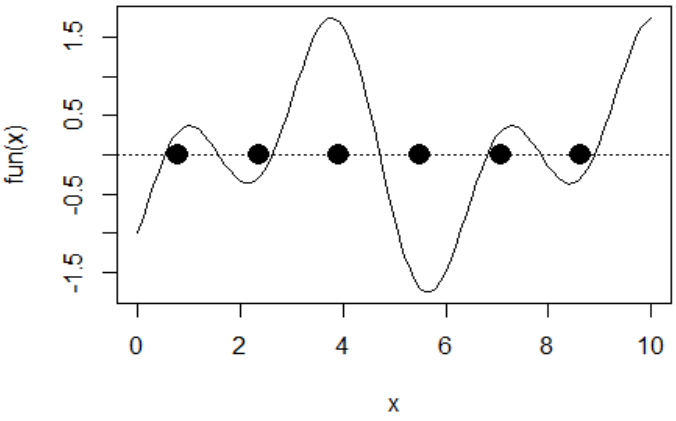
```
fun<- function (x) sin(2*x)-cos(x)
curve(fun(x), 0, 10)
abline(h = 0, lty = 3)
uni<- uniroot(fun, c(0, 10))$root
points(uni, 0, pch = 16, cex = 2)
```



Although the figure shows the presence of more zeroes in the interval [0,10], uniroot gives only one zero.

The function uniroot.all is an extension of uniroot which extracts many zeroes in the interval.

```
fun<- function (x) sin(2*x)-cos(x)
curve(fun(x), 0, 10)
abline(h = 0, lty = 3)
all<- uniroot.all(fun, c(0, 10))
points(All, y = rep(0, length(All)), pch = 16, cex = 2)
```



uniroot.all applies the function uniroot to subdivisions of given interval to locate roots.

This function may not be successful in extracting all roots in the given interval and so cannot be regarded as a full proof method.

Polyroot function

This function can be used to find zeros of a real or complex polynomial.
The argument for this function is a vector whose coordinates are the coefficients of the terms in the polynomial. The terms of the polynomial must be considered in the descending order of power.

```
> x <- c(4,21,3,8)
> y <- c(1,2,3,4)
> z <- c(1,0,0,1)
>polyroot(x)
[1] -0.1930595+0.000000i -0.0909702+1.606736i -0.0909702-1.606736i
>polyroot(y)
[1] -0.0720852+0.6383267i -0.6058296+0.0000000i -0.0720852-0.6383267i
>polyroot(z)
```

[1] 0.5+0.8660254i -1.0+0.0000000i 0.5-0.8660254i

CONCLUSION

All numerical algorithms possess certain limitations in addition to their utility in various problems. Until now there exist no algorithm which give guarantee to find all the solutions of nonlinear equations. Most of them are able to give at most one root of most equations. It requires prior information to implement these algorithms.

The numerical algorithms find effective demonstrations in dealing with complicated problems for which analytical methods cannot be applied or hand calculations cannot be done. The use of any computational algorithm, analytically or numerical, without the proper understanding of the limitations and shortcomings will not take us to the correct results.

REFERENCES

1. *Esch, L., Kieffer, R., Lopez, T., Berbé, C., Damel, P., Debay, M., &Hannosset, J. F. Numerical Methods for Solving Nonlinear Equations. Asset and Risk Management: Risk Oriented Finance, 375-381.*
2. *Jones, O., Maillardet, R., & Robinson, A. (2014). Introduction to scientific programming and simulation using R. CRC Press.*
3. *Jones, O., Maillardet, R., & Robinson, A. (2014).Package 'spuRs'*
4. *Soetaert, K. (2014). Package rootSolve: roots, gradients and steady-states in R.*
5. *Yuan, Y. X. (2011). Recent advances in numerical methods for nonlinear equations and nonlinear least squares. Numerical algebra, control and optimization, 1(1), 15-34.*