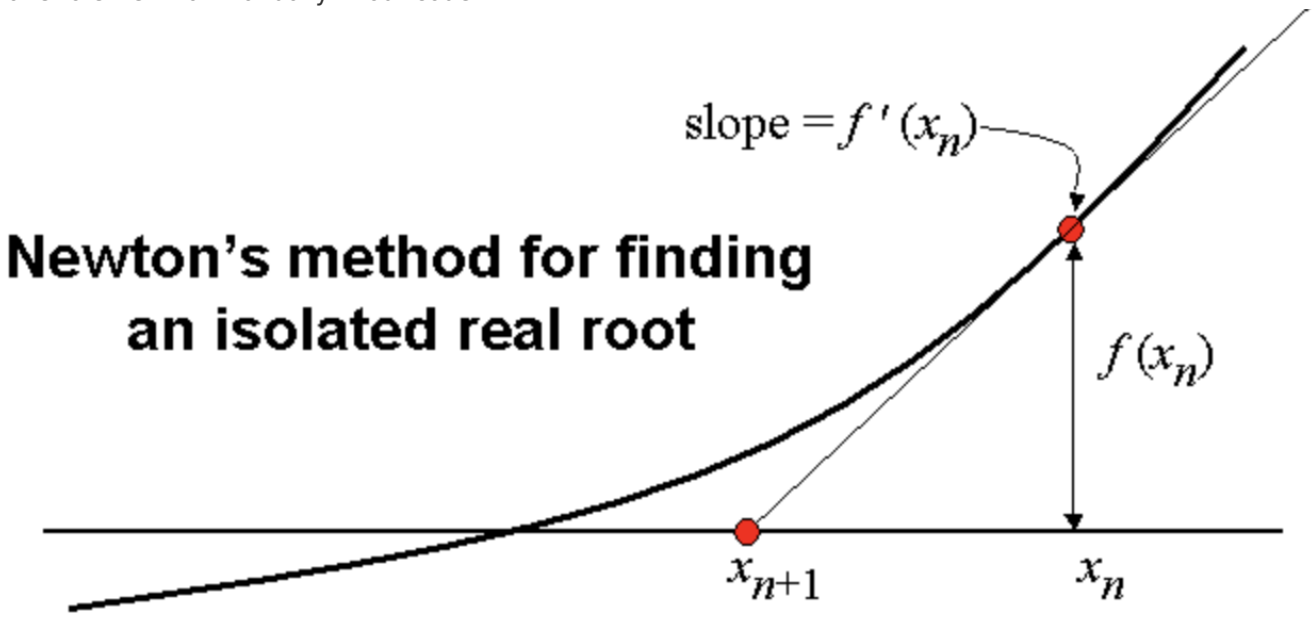
Newton-Raphson Method, is a Numerical Method, used for finding a root of an equation.
The method requires the knowledge of the derivative of the equation whose root is to be determined. So we would have to enter that manually in our code.



$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Newton-Raphson Method may not always converge, so it is advisable to ask the user to enter the maximum number of iteration to be performed in case the algorithm doesn't converge to a root. Moreover, since the method requires division by the derivative of the function, one should add a condition that prevents division by zero.

The following is a simple version of the program that finds the root, and tabulates the different values at each iteration. Just like any other numerical method bisection method is also an iterative method, so it is advised to tabulate values at each iteration.

PROGRAM(Simple Version):

```
/*******************************
 ****NEWTON RAPHSON METHOD*****
  2017 (c) Manas Sharma - http://bragitoff.com
 *******************************/
#include<stdio.h>
#include<math.h>

/*Function whose root is to be determined*/
double f(double x){
  return 3*x+sin(x)-exp(x);
}
```

```
/*Derivative of the function whose root is to be determined*/
double df(double x){
  return 3-cos(x)-exp(x);
}

int main(){
  double x,eps,x1;
  int maxSteps;
  printf("Enter the initial guess:\n");
  scanf("%lf",&x1);
  printf("Enter the desired accuracy:\n");
  scanf("%lf",&eps);
  printf("Enter the max. number of steps:\n");
  scanf("%d",&maxSteps);
  int iter=1;
  /*Newton-Raphson Method begins that tabulates the various values at each iteration*/
printf("_____________________________________________________________________________
_____\n");
  printf("x\tf(x)\t\tf'(x)\t\tx1\t\t|x-x1|\t\tf(x1)\n");
printf("_____________________________________________________________________________
_____\n");
  do{
    x=x1;
    /* IF-Condition to prevent division by zero[To be done: Check for infinite values
too]*/
    if(fabs(df(x))>=0.000000001&&df(x)!=NAN){
      /*New value of x using the NR Expression */
      x1=x-f(x)/df(x);
      printf("%d.\t%lf\t%lf\t%lf\t%lf\t%lf\n",iter,f(x),df(x),x1,fabs(x-x1),f(x1));
      iter++;
    }
    else{
      printf("Sorry! The slope is 0 for one of the iterations.n NR Method failed.\n");
      return 0;
    }
  }while(fabs(x-x1)>=eps&&iter<=maxSteps);
printf("_____________________________________________________________________________
________\n\nOne of the roots of the given equation is:\n\n%lf\n\n\n",x1);
}
```

The better version of the above code uses a function called ‘rootNR’ to perform the NR task and return the root. However, this function won’t tabulate the values at each iteration.
So in the following program I have also provided another function called ‘printNR’ that would return the root as well as print the various values at each iteration.

PROGRAM(Better Version):

```
/*****************************
 ****NEWTON RAPHSON METHOD*****
  2017 (c) Manas Sharma - http://bragitoff.com
 *****************************/
```

```
#include<stdio.h>
#include<math.h>

/*Function whose root is to be determined*/
double f(double x){
  return x*x*x-27;
}

/*Derivative of the function whose root is to be determined*/
double df(double x){
  return 3*x*x;
}

/*Function that returns the root from Newton-Raphson Method*/
double rootNR(double f(double x),double df(double x),double x1,double eps,double
maxSteps){
  double x;
  int i=1;
  do{
    x=x1;
    if(fabs(df(x))>=0.000000001){
      x1=x-f(x)/df(x);
      i++;
    }
  }while(fabs(x-x1)>=eps&&i<=maxSteps);
  return x1;
}

/*Newton-Raphson Method Function that tabulates the values at each iteration*/
double printNR(double f(double x),double df(double x),double x1,double eps,double
maxSteps){
  double x;
  int iter=1;
printf("_____________________________________________________________________________
_____________________\n");
  printf("iter\tx\t\tf(x)\t\tf'(x)\t\tx1\t\t|x-x1|\t\tf(x1)\n");
printf("_____________________________________________________________________________
_____________________\n");
  do{
    x=x1;
    if(fabs(df(x))>=0.000000001){
      x1=x-f(x)/df(x);
      printf("%d.\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\n",iter,x,f(x),df(x),x1,fabs(x-
x1),f(x1));
      iter++;
    }
  }while(fabs(x-x1)>=eps&&iter<=maxSteps);
  return x1;
}
main(){
  double x,eps,x1;
  int maxSteps;
```

```
  printf("Enter the initial guess:\n");
  scanf("%lf",&x);
  printf("Enter the desired accuracy:\n");
  scanf("%lf",&eps);
  printf("Enter the max. number of steps:\n");
  scanf("%d",&maxSteps);
printf("_____________________________________________________________________
_____________\n\nOne of the roots of the given equation
is:\n\n%lf\n\n\n",printNR(f,df,x,eps,maxSteps));
}
```

OUTPUT:

For x^3-27:

```
Enter the desired accuracy:
0.000001
Enter the max. number of steps:
30
_____________________________________________________________________________
x       f(x)            f'(x)           x1              |x-x1|          f(x1)
_____________________________________________________________________________
1.      702.000000      243.000000      6.111111        2.888889        201.223594
2.      201.223594      112.037037      4.315066        1.796045        53.345632
3.      53.345632       55.859379       3.360067        0.954999        10.935330
4.      10.935330       33.870154       3.037207        0.322860        1.017094
5.      1.017094        27.673875       3.000454        0.036753        0.012258
6.      0.012258        27.008171       3.000000        0.000454        0.000002
7.      0.000002        27.000001       3.000000        0.000000        0.000000
_____________________________________________________________________________

One of the roots of the given equation is:

3.000000
```

For 3x+sin(x)-exp(x):

```
Enter the initial guess:
5
Enter the desired accuracy:
0.00001
Enter the max. number of steps:
20
_____________________________________________________________________________
x       f(x)            f'(x)           x1              |x-x1|          f(x1)
_____________________________________________________________________________
1.      -134.372083     -145.696821     4.077728        0.922272        -47.583339
2.      -47.583339      -55.418346      3.219107        0.858621        -15.425905
3.      -15.425905      -21.008793      2.484848        0.734259        -3.934210
4.      -3.934210       -8.207312       2.005494        0.479354        -0.506282
5.      -0.506282       -4.008625       1.879195        0.126298        0.042173
6.      0.042173        -3.244701       1.892193        0.012997        -0.008526
7.      -0.008526       -3.318008       1.889623        0.002570        0.001598
8.      0.001598        -3.303422       1.890107        0.000484        -0.000304
9.      -0.000304       -3.306165       1.890015        0.000092        0.000058
10.     0.000058        -3.305643       1.890033        0.000017        -0.000011
11.     -0.000011       -3.305742       1.890029        0.000003        0.000002
_____________________________________________________________________________

One of the roots of the given equation is:

1.890029
```

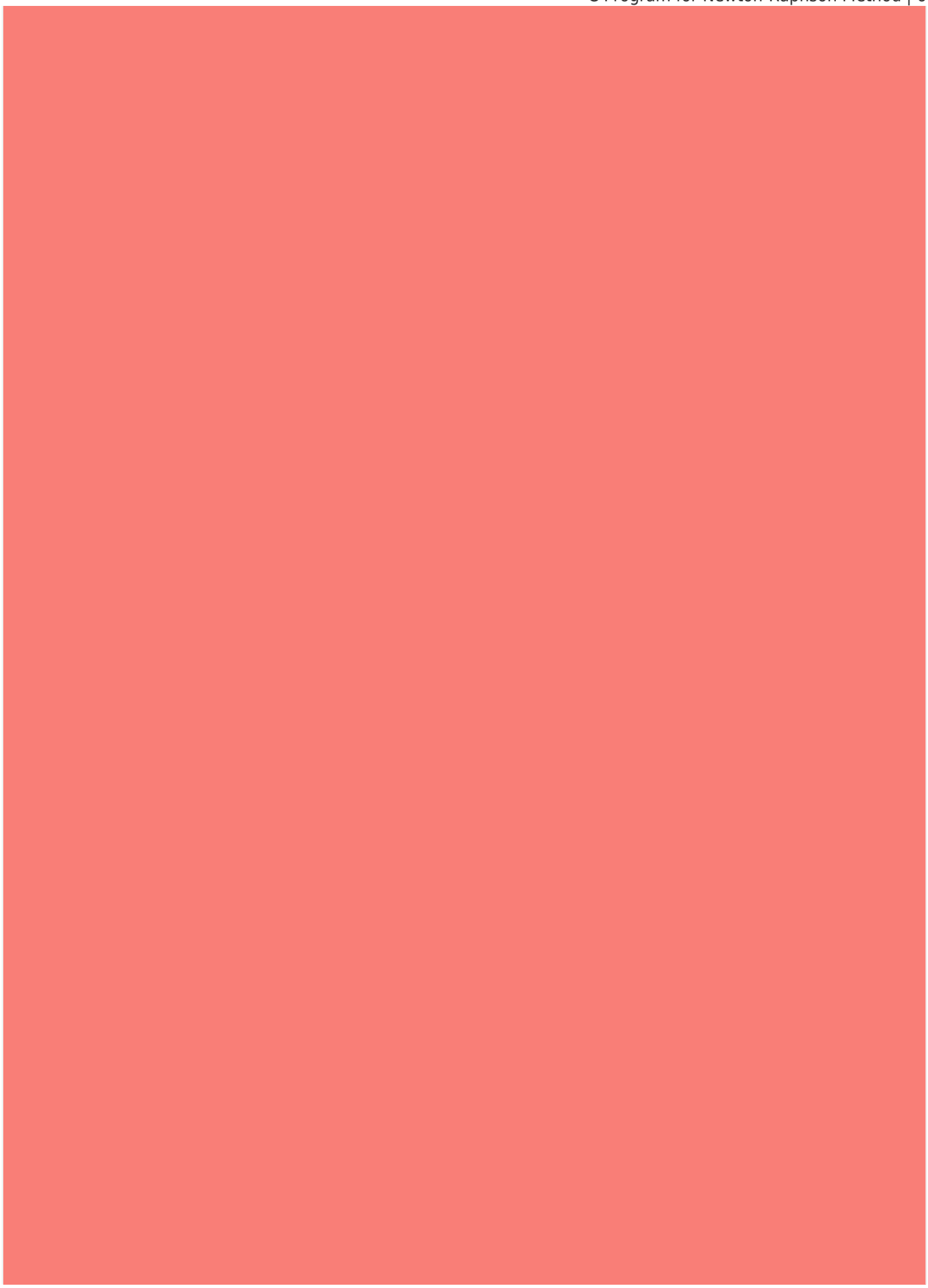
Related Posts:

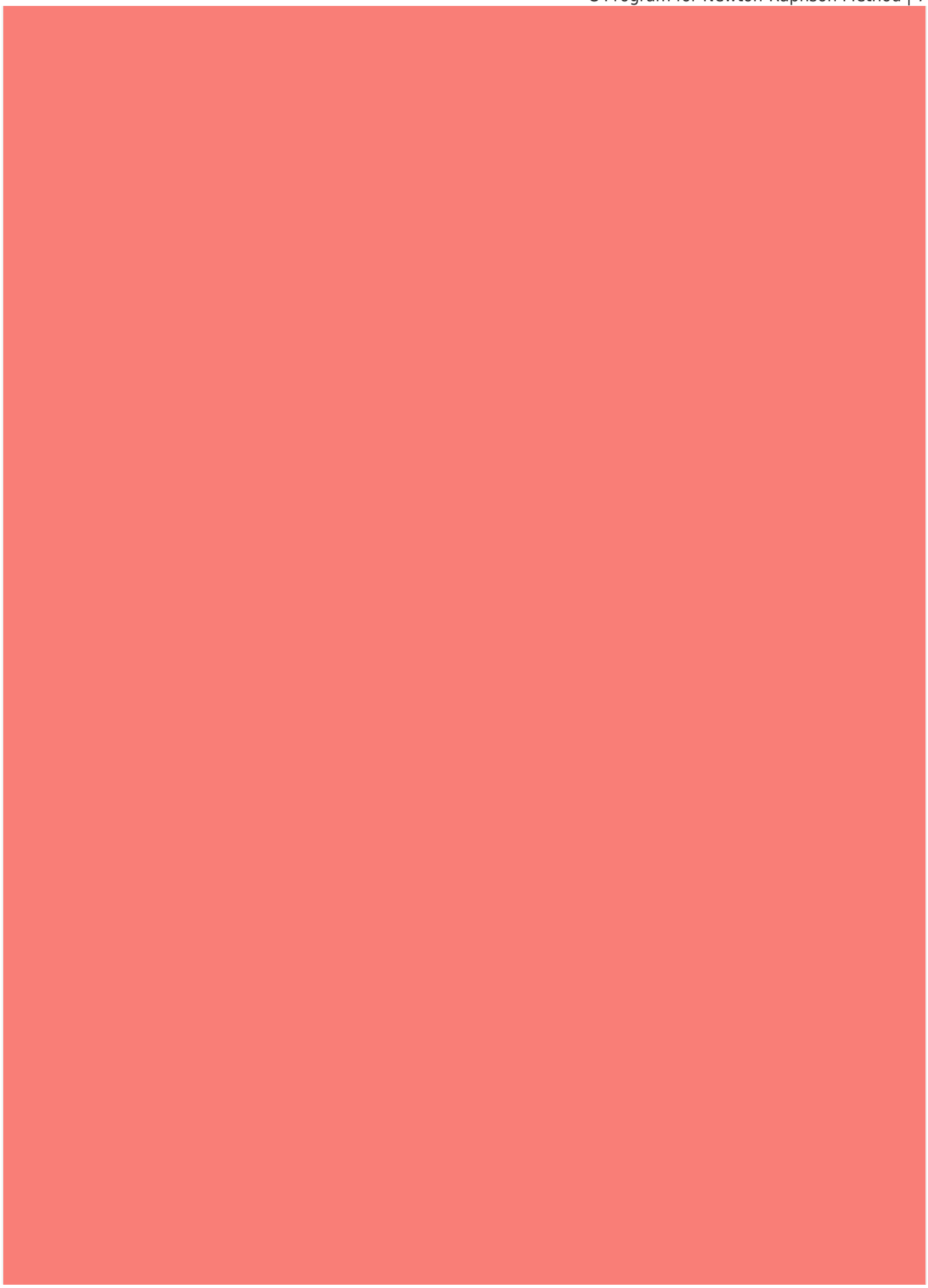
Newton-Raphson C++ Program
Newton-Raphson Lab Manual(Contains Flow-chart and algorithm)

Manas Sharma

I'm a physicist specializing in computational material science with a PhD in Physics from Friedrich-Schiller University Jena, Germany. I write efficient codes for simulating light-matter interactions at atomic scales. I like to develop Physics, DFT, and Machine Learning related apps and software from time to time. Can code in most of the popular languages. I like to share my knowledge in Physics and applications using this Blog and a YouTube channel.

manas.bragitoff.com/





Share this:

Click to share on Facebook (Opens in new window)

Click to share on Twitter (Opens in new window)

Click to share on WhatsApp (Opens in new window)

Click to share on Pinterest (Opens in new window)

Click to share on Reddit (Opens in new window)

Click to share on LinkedIn (Opens in new window)

Click to email a link to a friend (Opens in new window)

Click to print (Opens in new window)

Click to share on Tumblr (Opens in new window)

Click to share on Pocket (Opens in new window)

Click to share on Telegram (Opens in new window)

[wpedon id="7041" align="center"]