Nonlinear Dynamics and Systems Theory, **12** (3) (2012) 215–236

Numerical Solutions of System of Non-linear ODEs by Euler Modified Method

B. S. Desale * and N. R. Dasre

School of Mathematical Sciences, North Maharashtra University, Jalgaon 425001, India

Received: June 29, 2011; Revised: June 19, 2012

Abstract: In this paper, we have proposed Euler's modified method for solving the six coupled system of non-linear ordinary differential equations (ODEs), which are aroused in the reduction of stratified Boussinesq equations. This method can also be called as revised Euler's modified method for solving two simultaneous ODEs. We have obtained the numerical solutions on stable and unstable manifolds. The error between the numerical solution and exact solution is of order 10^{-20} to 10^{-6}. We have coded this programme in C-language.

Keywords: *stratified Boussinesq equation, Euler modified method, integrable systems.*

Mathematics Subject Classification (2010): 34A09, 65L05, 65L99.

1 Introduction

The stratified Boussinesq equations form a system of Partial Differential Equations (PDEs) modelling the movements of planetary atmospheres. It may be noted that literature also refers to Boussinesq approximation as Oberbeck–Boussinesq approximation. For this, one may refer to an interesting article by Rajagopal et al [1] which provides a rigorous mathematical justification for perturbations of the Navier-Stokes equations. Majda & Shefter [2] have chosen certain special solutions of this system of ODEs to demonstrate the onset of instability when the Richardson number is less than 1/4. Majda and Shefter [3] have shown that the analysis, in the special cases considered, reduces to the solutions of Hamiltonian system. These reductions form an interesting coupled system of six non-linear ODEs. Shrinivasan et al [4] have also tested the system for complete integrability by use of first integrals. Further, Desale [6] has incorporated the effect

* Corresponding author: mailto:bsdesale@nmu.ac.in

of rotation in the same system in the context of basin scale dynamics, while Desale and Sharma [7] have given special solutions of rotating stratified Boussinesq equations. Desale and Patil [8] have tested the system of six coupled nonlinear ODEs by Painleve Test. Burton and Zhang [9] have given the periodic solutions for singular integral equations. Biswas et al [10] have studied the behavior of soliton solutions in the form of KdV partial differential equation in the fiber optics solitons theory in communication engineering.

In this paper, we have given the C-code to find and to test the initial values which lie on the invariant surface given by equation (4). We have implemented Euler Modified method to find the numerical solution of the system (1) passing through the initial values on invariant surface (4). We have discussed the use of this method in the subsection (3.1). We have given the codes for solutions on stable and unstable manifolds of invariant surface which is obtained by four first integrals.

2 Preliminaries

Shrinivasan et al [4] have tested the system (1) as given below for complete integrability. Also, Deasle and Shrinivasan [5] have shown that in the general case, the problem of integration reduces to the integrations of the system of six coupled autonomous ODE's

$$\left.\begin{array}{rcl} \dot{\mathbf{w}} & = & \frac{g}{\rho_b}\hat{\mathbf{e_3}} \times \mathbf{b}, \\ \dot{\mathbf{b}} & = & \frac{1}{2}\mathbf{w} \times \mathbf{b}, \end{array}\right\} \tag{1}$$

where $\mathbf{w} = (w_1, w_2, w_3)^T$, $\mathbf{b} = (b_1, b_2, b_3)^T$ and $\frac{g}{\rho_b}$ is a non-dimensional constant as mentioned by Desale [11] in his Ph. D. thesis.

The above system can be written component-wise as below

$$\left.\begin{array}{rcl} \dot{w_1} & = & -\frac{g}{\rho_b}b_2,\ \dot{w_2} = \frac{g}{\rho_b}b_1,\ \dot{w_1} = 0, \\ \dot{b_1} & = & \frac{1}{2}(w_2b_3 - w_3b_2),\ \dot{b_2} = \frac{1}{2}(w_3b_1 - w_1b_3),\ \dot{b_3} = \frac{1}{2}(w_1b_2 - w_2b_1). \end{array}\right\} \tag{2}$$

The system (1) admits the following four first integrals

$$\left.\begin{array}{lll} 1)\ \ |\mathbf{b}|^2 & = & c_1, \\ 2)\ \ \mathbf{w}\cdot\mathbf{b} & = & c_2, \\ 3)\ \ \hat{\mathbf{e_3}}\cdot\mathbf{w} & = & c_3, \\ 4)\ \ \frac{|\mathbf{w}|^2}{2} + \frac{2g}{\rho_b}\hat{\mathbf{e_3}}\cdot\mathbf{b} & = & c_4, \end{array}\right\} \tag{3}$$

with non zero values of c_1, c_2, c_3 and c_4. The possible critical points of the system (1) are $(\pm\hat{\mathbf{e_3}}, \pm\hat{\mathbf{e_3}})$. For $c_1 = 1$ and $w = \pm\hat{\mathbf{e_3}}$, c_3 may assume the values ± 1 (not both). Now we take $c_3 = 1$, so that the possible critical points are $(\hat{\mathbf{e_3}}, \pm\hat{\mathbf{e_3}})$. At the rest points $(\hat{\mathbf{e_3}}, \pm\hat{\mathbf{e_3}})$, the value of c_2 is ± 1.

Remark 2.1 The case $c_2 = -1$ will be surface disjoint from $\mathbf{w}\cdot\mathbf{b} = 1$ and the similar analysis will be carried out if we take $c_2 = -1$. Right now we take $c_1 = 1$, $c_2 = 1$ and $c_3 = 1$. But this forces $\mathbf{b} = \hat{\mathbf{e_3}}$ at a critical point, so with our specific conditions we have only one rest point $(\hat{\mathbf{e_3}}, \hat{\mathbf{e_3}})$ on the invariant surface (3). At this critical point fourth first integral assumes the value $c_4 = \frac{1}{2} + \frac{2g}{\rho_b}$.

With the above specification, we have following four first integrals

$$|\mathbf{b}|^2 = 1, \mathbf{w}\cdot\mathbf{b} = 1, \hat{\mathbf{e_3}}\cdot\mathbf{w} = 1, \frac{|\mathbf{w}|^2}{2} + \frac{2g}{\rho_b}\hat{\mathbf{e_3}}\cdot\mathbf{b} = \frac{1}{2} + \frac{2g}{\rho_b}. \tag{4}$$

A critical point $(\hat{\mathbf{e_3}}, \hat{\mathbf{e_3}})$ lies on invariant surface and (b_1, b_2, b_3) is on the surface $|\mathbf{b}|^2 = 1$. Therefore we have

$$\left.\begin{aligned} w_1 &= \frac{-b_2 k}{1-b_3} + \frac{b_1}{1+b_3}, \\ w_2 &= \frac{b_1 k}{1-b_3} + \frac{b_2}{1+b_3}, \\ w_3 &= 1. \end{aligned}\right\} \tag{5}$$

where k is a function of b_3, given by the following equation

$$k^2 = \frac{(1-b_3)^2}{(1+b_3)^2}\Big[\frac{4g(1+b_3)-\rho_b}{\rho_b}\Big]. \tag{6}$$

One may refer [4, 5] for more details of this analysis. Since $|\mathbf{b}|^2 = 1$, we can use spherical-polar co-ordinates

$$b_1 = \cos\theta\sin\phi,\ b_2 = \sin\theta\sin\phi,\ b_3 = \cos\phi. \tag{7}$$

Hence,

$$k^2 = \tan^4(\frac{\phi}{2})\Big[\frac{8g}{\rho_b}\cos^2(\frac{\phi}{2}) - 1\Big]. \tag{8}$$

For k to be real , Shrinivasan et al [5] have put up the restriction to ϕ as $0 \le \phi \le 2\cos^{-1}(\sqrt{\frac{\rho_b}{8g}})$. With this limitation k takes the values negative, positive and zero. With these possible choices of k, the invariant surface will be the union of disjoint manifolds corresponding to $k > 0$, is unstable manifold, $k < 0$, is stable manifold and $k = 0$, is a center manifold. Regarding these manifolds, readers are advised to refer to Shrinivasan et al [5].

Now for $k > 0$, the unstable manifold is given by

$$\left.\begin{aligned} w_1 &= \tan(\tfrac{\phi}{2})\Big[\cos\theta - \sin\theta\sqrt{\tfrac{8g}{\rho_b}\cos^2(\tfrac{\phi}{2}) - 1}\ \Big], \\ w_2 &= \tan(\tfrac{\phi}{2})\Big[\cos\theta + \sin\theta\sqrt{\tfrac{8g}{\rho_b}\cos^2(\tfrac{\phi}{2}) - 1}\ \Big], \\ w_3 &= 1, \\ b_1 &= \cos\theta\sin\phi, \\ b_2 &= \sin\theta\sin\phi, \\ b_3 &= \cos\phi, \\ with & \\ k &= \tan^2(\frac{\phi}{2})\Big[\frac{8g}{\rho_b}\cos^2(\frac{\phi}{2}) - 1\Big]. \end{aligned}\right\} \tag{9}$$

On this surface, system (1) reduces to

$$\left.\begin{aligned} \frac{d\phi}{dt} &= \tfrac{1}{2}\tan(\tfrac{\phi}{2})\sqrt{\tfrac{8g}{\rho_b}\cos^2(\tfrac{\phi}{2})-1}, \\ \frac{d\theta}{dt} &= \tfrac{1}{4}\sec^2(\tfrac{\phi}{2}), \end{aligned}\right\} \tag{10}$$

where as for $k < 0$, the stable manifold is given by

$$\left.\begin{aligned} w_1 &= \tan(\tfrac{\phi}{2})\Big[\cos\theta + \sin\theta\sqrt{\tfrac{8g}{\rho_b}\cos^2(\tfrac{\phi}{2})-1}\ \Big], \\ w_2 &= \tan(\tfrac{\phi}{2})\Big[\cos\theta - \sin\theta\sqrt{\tfrac{8g}{\rho_b}\cos^2(\tfrac{\phi}{2})-1}\ \Big], \\ w_3 &= 1, \\ b_1 &= \cos\theta\sin\phi, \\ b_2 &= \sin\theta\sin\phi, \\ b_3 &= \cos\phi, \\ \mathit{with} & \\ k &= -\tan^2(\tfrac{\phi}{2})\Big[\tfrac{8g}{\rho_b}\cos^2(\tfrac{\phi}{2})-1\Big]. \end{aligned}\right\} \tag{11}$$

On this surface, system (1) reduces to

$$\left.\begin{aligned} \frac{d\phi}{dt} &= -\tfrac{1}{2}\tan(\tfrac{\phi}{2})\sqrt{\tfrac{8g}{\rho_b}\cos^2(\tfrac{\phi}{2})-1}, \\ \frac{d\theta}{dt} &= \tfrac{1}{4}\sec^2(\tfrac{\phi}{2}). \end{aligned}\right\} \tag{12}$$

3 Numerical Solution

In their studies, Shrinivasan et al [5] have shown that the system (1) is completely integrable and solutions exist on invariant surface (3) for all the time. So we are looking for the numerical solution of the system (1) on the invariant surface (3). We find the initial values which satisfy the four first integrals given by (4) and consequently we can find the solutions of system (1) passing through these initial values. We use the following programme to find the initial values so that they satisfy the four first integrals. We use the following programme to test finitely many points.

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{ FILE *fp;
  double b10,b20,b30,phi0,theta0;
  double eps=0.0000001,G=39.2;
  double g=9.8,rho_b=2;
  long int i,j,k;
```

```
  double y1,w10,w20,w30;
  double int1,int2,int3L,int3R;
  double diff1,diff2,diff3;
   clrscr();
   fp=fopen("new_02a1.xls","w+");

   fprintf(fp,"\n\t PROGRAMME FOR INITIAL SOLUTIONS
                   SATISFYING FIRST FOUR INTEGRALS");
 b10=0.000001;
 b20=0.000001;
 b30=0.000001;
 printf("\n\t PROGRAMME FOR INITIAL SOLUTIONS
              SATISFYING FIRST FOUR INTEGRALS");
    fprintf(fp,"\n\tb10\tb20\tb30\ttheta0\tphi0\n");
     printf("\nb10\tb20\tb30\ttheta0\tphi0\n");
     for(k=0;k<1000;k++) //b30 loop
       {
          for(j=0;j<1000;j++)//b20 loop
         {
            for(i=0;i<1000;i++) //b10 loop
           {
        theta0=atan(b20/b10);
        phi0=atan(sqrt(b10*b10+b20*b20)/b30);
        y1=sqrt(39.2*cos(phi0/2.0)*cos(phi0/2.0)-1.0);
        w10=tan(phi0/2.0)*(cos(theta0)-(sin(theta0)*y1));
        w20=tan(phi0/2.0)*(sin(theta0)+(cos(theta0)*y1));
        w30=1.000000;
        int1=b10*b10+b20*b20+b30*b30;
        int2=b10*w10+b20*w20+b30*w30;
   int3L=w10*w10+w20*w20+w30*w30+((4.0*g*b30)/rho_b);
   int3R=1.0+((4.0*g)/rho_b);
   diff1=fabs(int1-1.0);
   diff2=fabs(int2-1.0);
   diff3=fabs(int3L-int3R);
  if(diff1<eps)
    {
   if(diff2<eps)
    {
     if(diff3<eps)
  {
fprintf(fp,"\n\t%.10lf\t%.10lf\t%.10lf\t%.10lf\t%.10lf",
              b10,b20,b30,theta0,phi0);
printf("\n%.10lf\t%.10lf\t%.10lf\t%.10lf\t%.10lf",
            b10,b20,b30,theta0,phi0);
 } } }
 b10=b10+0.000001;
 if(b10>=1.000001) b10=0.000001; }
 b20=b20+0.000001;
```

```
if(b20>=1.000001) b20=0.000001; }
b30=b30+0.000001;
if(b30>=1.000001) b30=0.000001; }
getch(); }
```

With the help of the above programme we get the initial value. After getting the initial value, we decide on which manifold the initial value lies on – that is whether it is stable, unstable or central manifold. Using the above programme, we get the initial value $b_0 = (b_{10}, b_{20}, b_{30})$. From this initial value $b_0 = (b_{10}, b_{20}, b_{30})$, we calculate the value of k, then we conclude whether the initial value is on stable, unstable or on center manifold. Once we confirm, our initial value is either on stable or unstable surface, accordingly we find the numerical solution by Euler modified method. In the following subsection (3.1), we implement the method to calculate the numerical solution. Further, we write the algorithm and encode the programme.

3.1 Implementation of Euler modified method for the numerical solution

We start with the initial condition $t = 0$ and the initial point $\mathbf{b}_0 = (b_{10}, b_{20}, b_{30})$. We calculate the initial value of (ϕ_0, θ_0) as

$$\theta_0 = \tan^{-1}\left(\frac{b_2}{b_1}\right), \quad \phi_0 = \tan^{-1}\left(\frac{\sqrt{b_1^2 + b_2^2}}{b_3}\right). \tag{13}$$

Now, we calculate the value of ϕ_1 and θ_1 by Predictor Formula as

$$\phi_1 = \phi_0 + hf_1(t_0, \phi_0, \theta_0), \quad \theta_1 = \theta_0 + hf_2(t_0, \phi_0, \theta_0), \tag{14}$$

where h is a step size, $f_1 = \frac{1}{2}\tan(\frac{\phi}{2})\sqrt{\frac{8g}{\rho_b}\cos^2(\frac{\phi}{2}) - 1}$, $f_2 = \frac{1}{4}\sec^2(\frac{\phi}{2})$. Since there is an error in ϕ_1 and θ_1, we refine or try to get more accurate values of ϕ_1 and θ_1 by Corrector Formula as below,

$$\phi_1^{(1)} = \phi_0 + \frac{h}{2}[f_1(t_0, \phi_0, \theta_0) + f_1(t_0 + h, \phi_1, \theta_1)]. \tag{15}$$

In the above step the error can be reduced to the desired accuracy. Here we have considered the accuracy of 10^{-20}. The error is reduced by repeating the corrector formula as below,

$$\phi_1^{(n+1)} = \phi_0 + \frac{h}{2}[f_1(t_0, \phi_0, \theta_0) + f_1(t_0 + h, \phi_1^{(n)}, \theta_1)]. \tag{16}$$

As we get the most correct value of ϕ, we use this value of ϕ for calculating the correct value of θ with the accuracy of 10^{-20} as

$$\theta_1^{(1)} = \theta_0 + \frac{h}{2}[f_2(t_0, \phi_0, \theta_0) + f_2(t_0 + h, \phi_1, \theta_1)], \tag{17}$$

$$\theta_1^{(n+1)} = \theta_0 + \frac{h}{2}[f_2(t_0, \phi_0, \theta_0) + f_2(t_0 + h, \phi_1, \theta_1^{(n)})], \tag{18}$$

and so on. This gives us the corrected values of θ and ϕ. The exact solutions of (10) are

$$\left.\begin{aligned} \phi(t) &= 2\sin^{-1}\Big[\frac{2k_1.\sqrt{\frac{G-1}{G}}.e^{-\frac{t}{4}\sqrt{G-1}}}{1 + k_1^2.e^{-\frac{t}{2}\sqrt{G-1}}}\Big], \\ \theta(t) &= \frac{t}{4} + \tan^{-1}\Big[\frac{\sqrt{G}}{k_1}.e^{\frac{t}{4}\sqrt{G-1}} - \sqrt{G-1}\Big] \\ &\quad - \tan^{-1}\Big[\frac{\sqrt{G}}{k_1}.e^{\frac{t}{4}\sqrt{G-1}} + \sqrt{G-1}\Big] + k_2, \end{aligned}\right\} \tag{19}$$

where k_1, k_2 are constants and $G = 8g/\rho_b$. Now for our calculations, we took $G = 39.2$ with $g = 9.8$ and $\rho_b = 2$. We have compared the corrected values with the exact solutions and we got the minimum error of 10^{-20} and maximum up to 10^{-6}. Now by using the method of back substitution we have obtained the values of $\mathbf{b}(b_1, b_2, b_3)$ and $\mathbf{w}(w_1, w_2, w_3)$.

3.2 Algorithm for numerical solution

Here we give the algorithm for numerical solution by Euler's Modified Method [14, 15]. The details of the algorithm are as given below:

Step 1: Enter the initial values of $t_0, \phi_0, \theta_0, t, g, \rho_b$ and h (step size).

Step 2:Calculate the values of $b_{10}, b_{20}, b_{30}, w_{10}, w_{20}, w_{30}, k_1, k_2$ and k. Here we have obtained the initial values.

Step 3: Calculate the values of ϕ_1 and θ_1 by using Euler's Predictor Formula.

Step 4: Calculate the value of ϕ_1 up to the desired accuracy by using Euler's Corrector Formula.

Step 5: Calculate the value of θ_1 up to the desired accuracy by using Euler's Corrector Formula.

Step 6: Calculate the values of b_1, b_2, b_3, w_1, w_2 and w_3 by using equation (7).

Step 7: Calculate the exact values of ϕ and θ by using equation (9) then calculate the exact values of b_1, b_2, b_3, w_1, w_2 and w_3 by using equation (7).

Step 8: Print the required exact and calculated numerical values.

Step 9: Replace ϕ_1 by ϕ_0, θ_1 by θ_0 and t_0 by $t + h$ and go to Step 3, until the value of ϕ is reached to its maximum for the given unstable manifold.

Step 10: Plot the graphs to see the difference.

Step 11: End.

3.3 Numerical solution on unstable manifold

On this manifold, we have $k > 0$ and the system (1) reduces to (10). Now we use the following programme to find the solution on the unstable manifold.

```
 #include<stdio.h>
 #include<stdlib.h>
 #include<conio.h>
 #include<math.h>
 #include<sys\stat.h>
 void main()
      {
  double f(double p);
  FILE *fp;
  double phi0,phi1,phi10,theta0,theta1,theta10,er_theta,er_phi;
  double h,t,t0,t1,b1,b2,b3,w1,w2,w3,b10,b20,b30,w10,w20,w30;
  double eb1,eb2,eb3,ew1,ew2,ew3,be1,be2,be3,we1,we2;
  double x,y0,y1,z0,z1,diff1,diff2,eps=0.01;
  double etheta,ephi,G=39.2,u,u1,u2,k1,k2,k;
  int i,n; /* g=9.8 , rho_b=2,*/
  clrscr();
printf("\n\n\t\t PROGRAMME FOR EULER MODIFIED METHOD");
```

```
fp=fopen("nrd001.xls","w+"); t0=0.0; t=6.0; h=0.001;
    printf("\n\n\t\t Enter the value of phi0= ");
    scanf("%lf",&phi0);
 printf("\n\n\t\t Enter the value of theta0= ");
    scanf("%lf",&theta0);
    fprintf(fp,"\n The value of phi0=%lf ",phi0);
 fprintf(fp,"\n The value of theta0=%lf ",theta0);
    b10=cos(theta0)*sin(phi0);
    b20=sin(theta0)*sin(phi0);
 b30=cos(phi0);
 y1=sqrt(39.2*cos(phi0/2.0)*cos(phi0/2.0)-1);
 w10=tan(phi0/2.0)*(cos(theta0)-sin(theta0)*y1);
 w20=tan(phi0/2.0)*(sin(theta0)+cos(theta0)*y1);
 w30=1.000000;
 fprintf(fp,"\n The value of b10=%lf ",b10);
 fprintf(fp,"\n The value of b20=%lf ",b20);
 fprintf(fp,"\n The value of b30=%lf ",b30);
 fprintf(fp,"\n The value of w10=%lf ",w10);
 fprintf(fp,"\n The value of w20=%lf ",w20);
 fprintf(fp,"\n The value of w30=%lf ",w30);

 /*calculating k1 and k2 for exact solution and
             k for initial solution  */
 u=sin(phi0/2.0);
 k1=(sqrt((G-1.0)/G)+sqrt(((G-1)/G)-u*u))/u;

 k2=theta0-atan((sqrt(G)/k1)-(sqrt(G-1.0)))
          +atan((sqrt(G)/k1)+(sqrt(G-1.0)));
 k=(tan(phi0/2)*tan(phi0/2))*sqrt(G*cos(phi0/2)*cos(phi0/2)-1);
 printf("\n\n\tThe value of k1=%.8f \n\n\tThe value of
         k2=%.8f",k1,k2);
    printf("\n\n\tThe value of k=%.8f ",k);
    fprintf(fp,"\nThe value of k1=%.8f ",k1);
    fprintf(fp,"\nThe value of k2=%.8f ",k2);
    fprintf(fp,"\nThe value of k=%.8f ",k);
         i=0;
printf("\n\n\tPress 'ENTER' to get step by step");
fprintf(fp,"\n t\t b1\t b2\t b3\t w1\t w2\t w3\t theta\tphi
          \tk\t ephi\t etheta");
printf("\n\n\t Error in Theta\t\t Error in Phi \t Value of K");
while(t0<t)
 {
      i++;
      t1=t0+h;
      y0=sqrt(39.2*cos(phi0/2.0)*cos(phi0/2.0)-1);
      phi1=phi0+(h/2.0)*tan(phi0/2.0)*y0;
      phi10=phi1;
      theta1=theta0+(0.25*h*f(phi0));
```

```
      theta10=theta1;

  /* Calculation of phi by modified formula */
      do{
        y0=sqrt(39.2*cos(phi0/2.0)*cos(phi0/2.0)-1);
        y1=sqrt(39.2*cos(phi10/2.0)*cos(phi10/2.0)-1);
phi1=phi0+(0.25*h)*((tan(phi0/2.0)*y0)+(tan(phi10/2.0)*y1));
       diff1=fabs(phi1-phi10);
       phi10=phi1;
       }while(diff1>eps);
k=(tan(phi1/2)*tan(phi1/2)) *sqrt(G*cos(phi1/2)*cos(phi1/2)-1);

  /* Calculation of theta by modified formula */
    do{
    theta1=theta0+(0.125*h)*(f(phi0)+f(phi1));
    diff2=fabs(theta10-theta1);
      }while(diff2>eps);

 /* Calculation of an approximate solution what we need */
     b1=cos(theta1)*sin(phi1);
     b2=sin(theta1)*sin(phi1);
     b3=cos(phi1);
     y1=sqrt(39.2*cos(phi1/2.0)*cos(phi1/2.0)-1);
     w1=tan(phi1/2.0)*(cos(theta1)-sin(theta1)*y1);
     w2=tan(phi1/2.0)*(sin(theta1)+cos(theta1)*y1);
     w3=1.000000;

 /* calculation  of exact solution */
ephi=2*asin((2*k1*sqrt((G-1)/G)*exp(-(t1/4)*sqrt(G-1)))
       /(1+k1*k1*exp(-(t1/2)*sqrt(G-1))));

   u1=atan((sqrt(G)*exp((t1/4)*sqrt(G-1)))/k1-sqrt(G-1));
   u2=atan((sqrt(G)*exp((t1/4)*sqrt(G-1)))/k1+sqrt(G-1));
   etheta=(t1/4)+u1-u2+k2;
   k=(tan(etheta/2)*tan(etheta/2))
         *sqrt(G*cos(etheta/2)*cos(etheta/2)-1);

 /* calculation of error in theta and phi*/
     er_theta=fabs(theta1-etheta);
     er_phi=fabs(phi1-ephi);

 /*   calculation of B and W */
  be1=cos(etheta)*sin(ephi);
  be2=sin(etheta)*sin(ephi);
  be3=cos(ephi);
  y1=sqrt(39.2*cos(ephi/2.0)*cos(ephi/2.0)-1);
  we1=tan(ephi/2.0)*(cos(etheta)-sin(etheta)*y1);
  we2=tan(ephi/2.0)*(sin(etheta)+cos(etheta)*y1);
```

```
  fprintf(fp,"\n%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf
           \t%lf\t%lf\t%lf", t1, b1, b2, b3, w1, w2, w3,
           theta1, phi1, k, etheta, ephi);
printf("\n\n\t%.20lf\t%.20lf\t%lf",er_theta,er_phi,k);
        phi0=phi1;
        theta0=theta1;
        t0=t1;
            getch();
         }}
    double f(double p)
       { double p_dash;
          p_dash=(1.0/cos(p/2.0))*(1.0/cos(p/2.0));
          return(p_dash);
       }
```

3.4 Numerical solution on stable manifold

On this manifold, we have $k < 0$ and the system (1) reduces to (12). Now we use the following programme to find the solution on the stable manifold.

```
 #include<stdio.h>
 #include<stdlib.h>
 #include<conio.h>
 #include<math.h>
 #include<sys\stat.h>
 void main()
 {
 double f(double p);
 FILE *fp;
 double phi0,phi1,phi10,theta0,theta1,theta10,er_theta,er_phi;
 double h,t,t0,t1,b1,b2,b3,w1,w2,w3,b10,b20,b30,w10,w20,w30;
 double eb1,eb2,eb3,ew1,ew2,ew3,be1,be2,be3,we1,we2;
 double x,y0,y1,z0,z1,diff1,diff2,eps=0.01;
 double etheta,ephi,G=39.2,u,u1,u2,k1,k2,k;
 int i,n; /* g=9.8 , rho_b=2,*/
clrscr(); printf("\n\n\t\t PROGRAMME FOR EULER MODIFIED METHOD");
fp=fopen("nrd001.xls","w+"); t0=0.0; t=6.0; h=0.001;
printf("\n\n\t\t Enter the value of phi0= ");
scanf("%lf",&phi0);
printf("\n\n\t\t Enter the value of theta0= ");
scanf("%lf",&theta0);
fprintf(fp,"\n The value of phi0=%lf ",phi0);
fprintf(fp,"\n The value of theta0=%lf ",theta0);
   b10=cos(theta0)*sin(phi0);
   b20=sin(theta0)*sin(phi0);
   b30=cos(phi0);
 y1=sqrt(39.2*cos(phi0/2.0)*cos(phi0/2.0)-1);
 w10=tan(phi0/2.0)*(cos(theta0)+sin(theta0)*y1);
 w20=tan(phi0/2.0)*(sin(theta0)-cos(theta0)*y1);
```

```
 w30=1.000000;
  fprintf(fp,"\n The value of b10=%lf ",b10);
  fprintf(fp,"\n The value of b20=%lf ",b20);
  fprintf(fp,"\n The value of b30=%lf ",b30);
  fprintf(fp,"\n The value of w10=%lf ",w10);
  fprintf(fp,"\n The value of w20=%lf ",w20);
  fprintf(fp,"\n The value of w30=%lf ",w30);

  /*calculating k1 and k2 for exact solution
    and k for initial solution  */
 u=sin(phi0/2.0);
 k1=(sqrt((G-1.0)/G)+sqrt(((G-1)/G)-u*u))/u;
 k2=theta0-atan((sqrt(G)/k1)-(sqrt(G-1.0)))
          +atan((sqrt(G)/k1)+(sqrt(G-1.0)));
 k= - (tan(phi0/2)*tan(phi0/2))
           *sqrt(G*cos(phi0/2)*cos(phi0/2)-1);
 printf("\n\n\tThe value of k1=%.8f \n\n\tThe value of
          k2=%.8f",k1,k2);
 printf("\n\n\tThe value of k=%.8f ",k);
 fprintf(fp,"\nThe value of k1=%.8f ",k1);
 fprintf(fp,"\nThe value of k2=%.8f ",k2);
 fprintf(fp,"\nThe value of k=%.8f ",k);
     i=0;
printf("\n\n\tPress 'ENTER' to get step by step");
fprintf(fp,"\n t\t b1\t b2\t b3\t w1\t w2\t w3\t theta\t phi
         \tk\t ephi\t etheta");
printf("\n\n\t Error in Theta\t\t Error in Phi \t Value of K");
while(t0<t)
    {
         i++;
         t1=t0+h;
         y0=sqrt(39.2*cos(phi0/2.0)*cos(phi0/2.0)-1);
         phi1=phi0-(h/2.0)*tan(phi0/2.0)*y0;
         phi10=phi1;
         theta1=theta0+(0.25*h*f(phi0));
         theta10=theta1;

  /* Calculation of phi by modified formula */
  do{
   y0=sqrt(39.2*cos(phi0/2.0)*cos(phi0/2.0)-1);
   y1=sqrt(39.2*cos(phi10/2.0)*cos(phi10/2.0)-1);
   phi1=phi0-(0.25*h)*((tan(phi0/2.0)*y0)+(tan(phi10/2.0)*y1));
   diff1=fabs(phi1-phi10);
   phi10=phi1;
    }while(diff1>eps);
 k= - (tan(phi1/2)*tan(phi1/2))
          *sqrt(G*cos(phi1/2)*cos(phi1/2)-1);
```

```
  /* Calculation of theta by modified formula */
    do{
       theta1=theta0+(0.125*h)*(f(phi0)+f(phi1));
       diff2=fabs(theta10-theta1);
       }while(diff2>eps);

 /* Calculation of an approximate solution what we need */
    b1=cos(theta1)*sin(phi1);
    b2=sin(theta1)*sin(phi1);
    b3=cos(phi1);
    y1=sqrt(39.2*cos(phi1/2.0)*cos(phi1/2.0)-1);
    w1=tan(phi1/2.0)*(cos(theta1)+sin(theta1)*y1);
    w2=tan(phi1/2.0)*(sin(theta1)-cos(theta1)*y1);
    w3=1.000000;

 /* calculation  of exact solution */
ephi=2*asin((2*k1*sqrt((G-1)/G)*exp(-(t1/4)*sqrt(G-1)))
      /(1+k1*k1*exp(-(t1/2)*sqrt(G-1))));
u1=atan((sqrt(G)*exp((t1/4)*sqrt(G-1)))/k1-sqrt(G-1));
u2=atan((sqrt(G)*exp((t1/4)*sqrt(G-1)))/k1+sqrt(G-1));
etheta=(t1/4)+u1-u2+k2;
k=-(tan(etheta/2)*tan(etheta/2))
   *sqrt(G*cos(etheta/2)*cos(etheta/2)-1);

 /* calculation of error in theta and phi*/
    er_theta=fabs(theta1-etheta);
    er_phi=fabs(phi1-ephi);

 /*   calculation of B and W */
     be1=cos(etheta)*sin(ephi);
     be2=sin(etheta)*sin(ephi);
     be3=cos(ephi);
     y1=sqrt(39.2*cos(ephi/2.0)*cos(ephi/2.0)-1);
     we1=tan(ephi/2.0)*(cos(etheta)+sin(etheta)*y1);
     we2=tan(ephi/2.0)*(sin(etheta)-cos(etheta)*y1);
fprintf(fp,"\n%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf
            \t%lf\t%lf\t%lf\t%lf\t%lf",t1,b1,b2,b3,w1,w2,
            w3, theta1, k, etheta, ephi);
 printf("\n\n\t%.20lf\t%.20lf\t%lf",er_theta,er_phi,k);
     phi0=phi1;
     theta0=theta1;
     t0=t1;
      getch();
         } }
 double f(double p)
         { double p_dash;
           p_dash=(1.0/cos(p/2.0))*(1.0/cos(p/2.0));
           return(p_dash);  }
```

4 Experimental Results

We have written the code for the above algorithm in C-programming. We have plotted the graphs by using Matlab. Here we have considered the initial solution as $\phi_0 = 0.100$ and $\theta_0 = 0.000$ for $k > 0$. Since at $\phi = 2.820649$ the value of k becomes negative, we have considered $\phi_0 = 2.820649$ and $\theta_0 = 0.000$ for $k > 0$.

In each figure, the first graph shows the numerical value calculated by us, the second graph shows the exact solution and the third graph shows the comparison of the first and the second graphs as shown in Figure 1 to Figure 16.

4.1 Figures for numerical solution on unstable manifold

Here we consider $k > 0$. Here are Figures from 1 to 8.

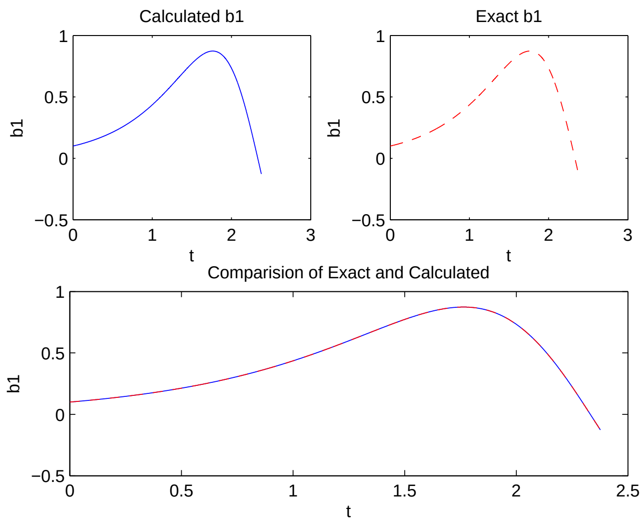

Figure 1: Graphs for b_1.

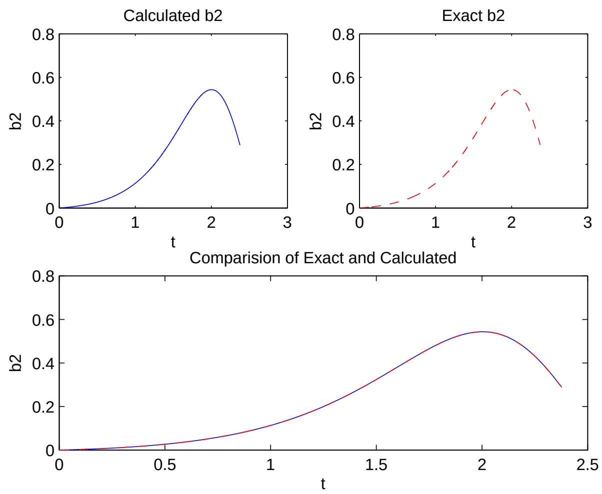

Figure 2: Graphs for b_2.

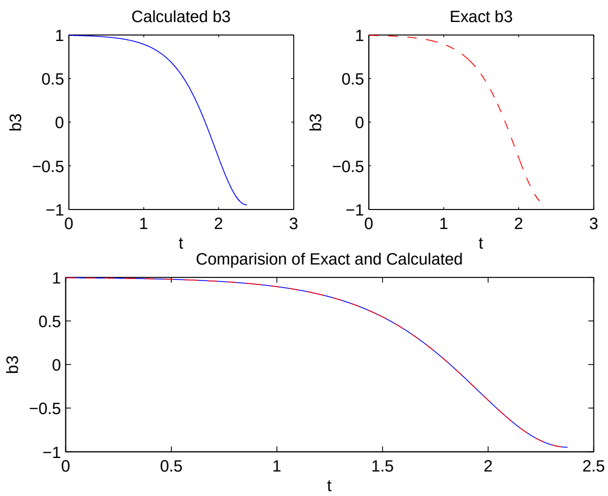

Figure 3: Graphs for b_3.

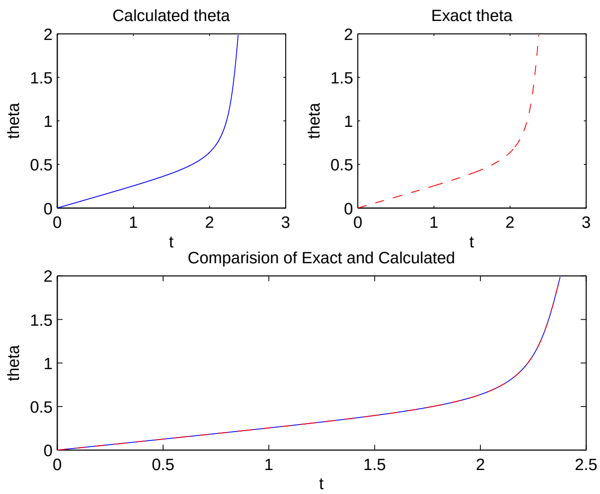

Figure 4: Graphs for θ.

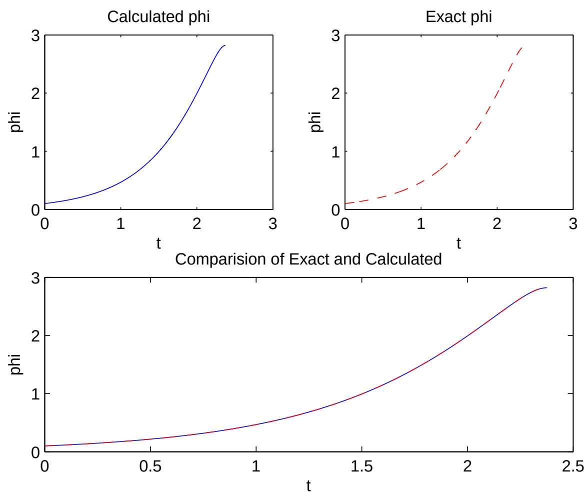

Figure 5: Graphs for ϕ.

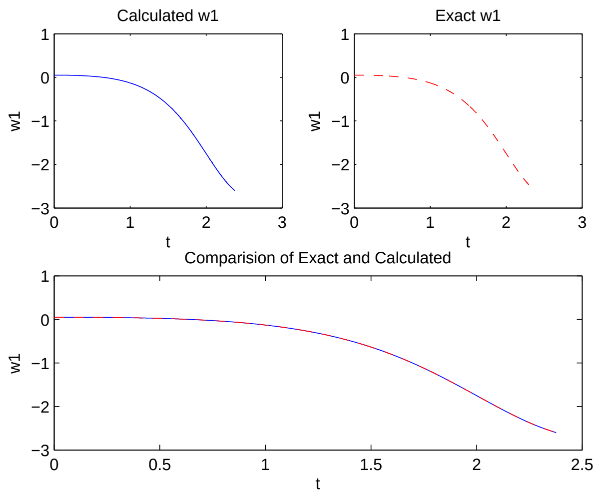

Figure 6: Graphs for w_1.

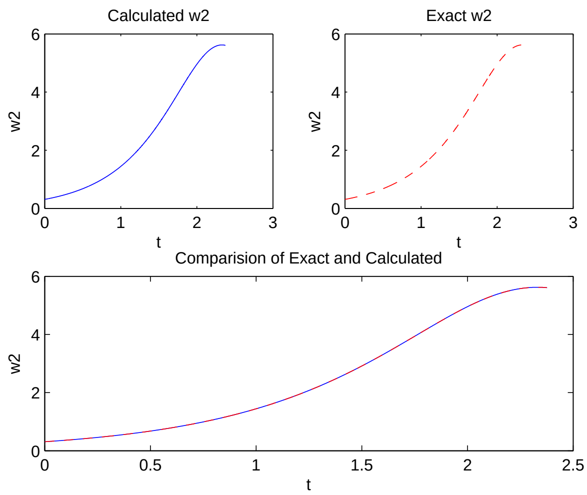

Figure 7: Graphs for w_2.

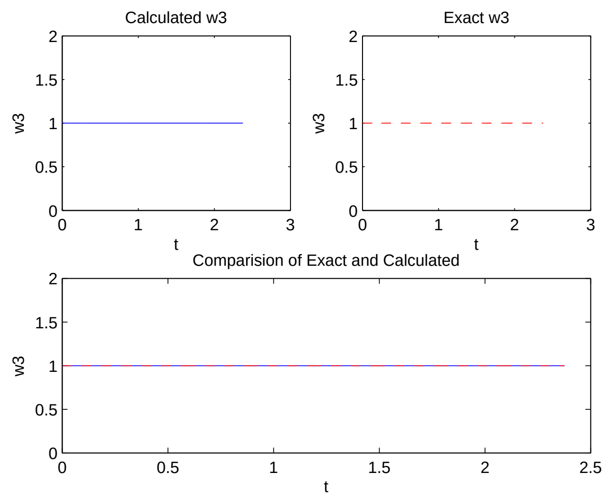

Figure 8: Graphs for w_3.

4.2 Numerical solution on stable manifold

Here we consider $k < 0$. Here are Figures from 9 to 16.

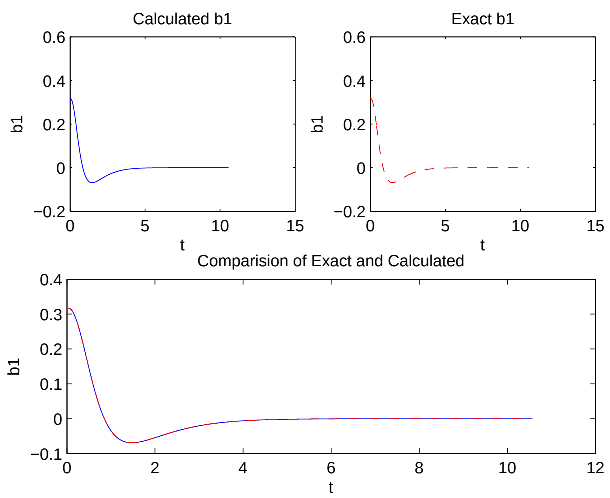

Figure 9: Graphs for b_1.

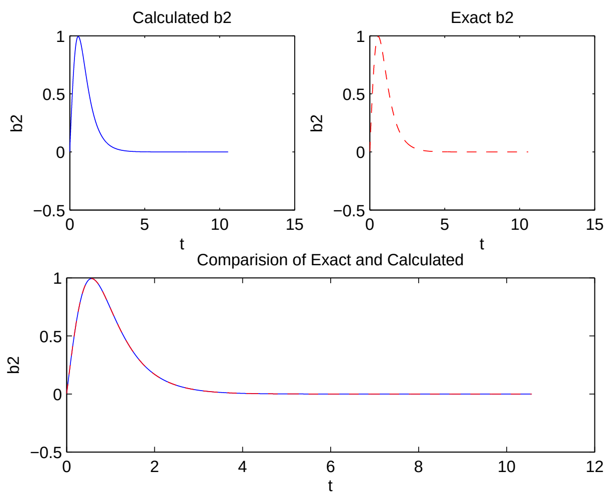

Figure 10: Graphs for b_2.

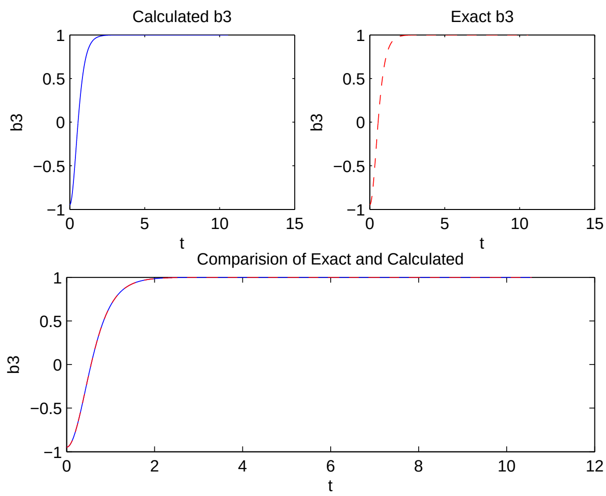

Figure 11: Graphs for b_3.

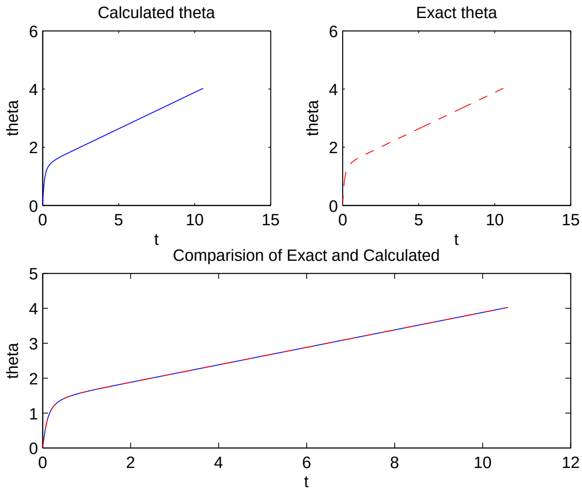

Figure 12: Graphs for θ.

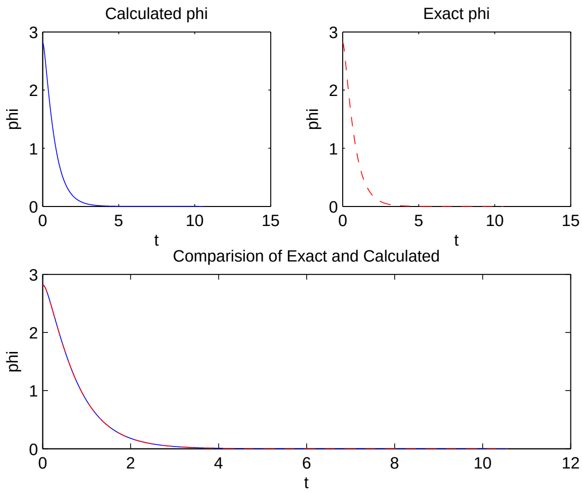

Figure 13: Graphs for ϕ.

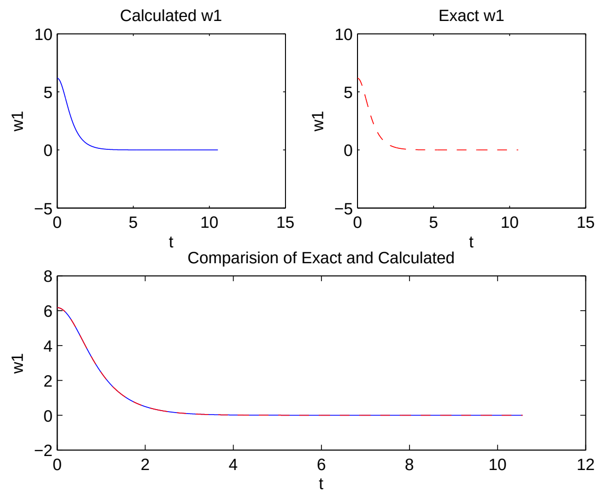

Figure 14: Graphs for w_1.

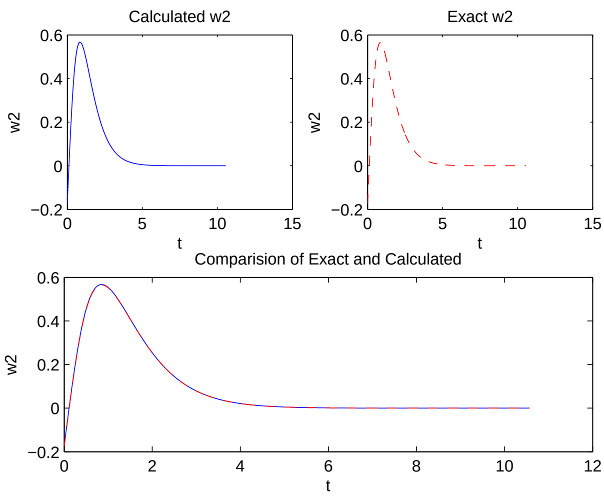

Figure 15: Graphs for w_2.

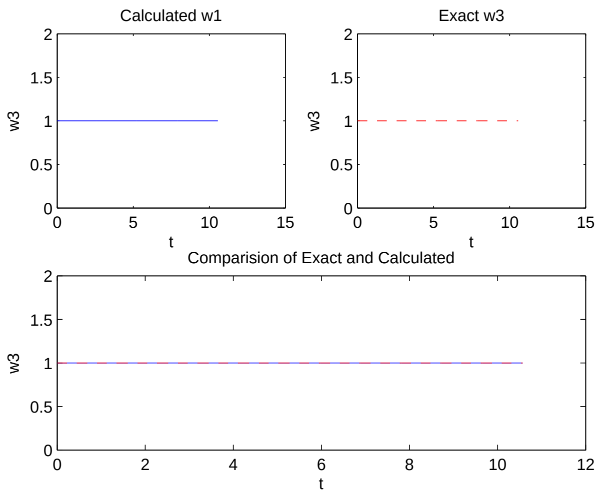

Figure 16: Graphs for w_3.

5 Conclusion

Here we have presented the scheme of Euler Modified Method for the numerical solution of the system of non-linear six coupled ODE's (1), with the error of 10^{-6}. Initially we have an error of 10^{-20} in the solution. It can be reduced as we reduce the step size. This error increases but it is up to 10^{-6} which is the upper bound. In future we will attempt to minimize the error and sharpen the accuracy of the solution.

Acknowledgment

We are thankful to Mr. Uday Shende, Director and Dr. S. R. Devane, the principal, Ramrao Adik Institute of Technology, Sector-07, Nerul, Navi Mumbai-400 706, INDIA for their valuable support throughout the work.

References

[1] Rajagopal, K. R., Ruzicka, M. and Srinivasa, A. R. On the Oberbeck-Boussinesq Approximation. *Mathematical Models and Methods in Applied Sciences* **6** (1996) 1157–1167.

[2] Majda, A. J. and Shefter, M. G. Elementary stratified flows with instability at large Richardson number. *J. Fluid Mechanics* **376** (1998) 319–350.

[3] Majda, A. J. *Introduction to PDEs and Waves for the Atmosphere and Ocean*. Courant Lecture Notes in Mathematics 9. American Mathematical Society, Providence, Rhode Island, 2003.

[4] Srinivasan, G.K., Sharma, V. D. and Desale, B. S. An integrable system of ODE reductions of the stratified Boussinesq equations. *Computers and Mathematics with Applications* **53** (2007) 296–304.

[5] Desale, B. S. and Srinivasan, G. K. Singular Analysis of the System of ODE reductions of the Stratified Boussinesq Equations. *IAENG International Journal of Applied Mathematics* **38** (4) (2008) 184–191.

[6] Desale, B. S. Complete Analysis of an Ideal Rotating Uniformly Stratified System of ODEs. *Nonlinear Dynamics and Systems Theory* **9**(3) (2009) 263–275.

[7] Desale, B. S. and Sharma, V. Special Solutions to Rotating Stratified Boussinesq Equations. *Nonlinear Dynamics and Systems Theory* **10**(1) (2010) 29–38.

[8] Desale, B. S. and Patil, K. D. Painleve' Test to a Reduced System of Six Coupled Nonlinear ODEs. *Nonlinear Dynamics and Systems Theory* **10**(4) (2010) 349–361.

[9] Burton, T. A. and Zhang, B. Periodic solutions of Singular Integral Equations. *Nonlinear Dynamics and Systems Theory* **11**(1) (2011) 113–123.

[10] Biswas, M. H. A., Rahman, M. A. and Das, T. Optical Soliton in Nonlinear Dynamics and It's Graphical Representation. *Nonlinear Dynamics and Systems Theory* **11**(4) (2011) 383–396.

[11] Desale, B. S. An integrable system of reduced ODEs of Stratified Boussinesq Equations. *Ph. D. thesis, Indian Institute of Technology Bombay.* Powai, Mumbai, 2007.

[12] Mathews, J. H. *Numerical Methods for Mathematics, Science and Engineering*, 2^{nd} Edition. Prentice Hall of India, New Delhi, 1994.

[13] Kreyszig, E. *Advanced Engineering Mathematics*, 9^{th} Edition. Wiley India (P) Ltd., New Delhi, 2010.

[14] Jain, M. K. and Iyengar, S. R. K. *Numerical Methods for Scientific and Engineering Computation*, 3^{rd} Edition. New Age International (P) Ltd., Mumbai, 2001.

[15] Sastry, S. S. *Introductory Methods of Numerical Analysis*. Prentice Hall of India (P) Ltd., New Delhi, 2001.