University of Reading

School of Mathematics, Meteorology & Physics

Accuracy of a Moving Mesh Numerical Method applied to the Self-similar Solution of Nonlinear PDEs

Kam Wong

Supervisor: Prof M.J.Baines

August 2010

This dissertation is submitted to the Department of Mathematics in partial fulfillment of the requirements for the degree of Master of Science

"I confirm that this is my own wohh[illegible]

Abstract

Many problems involving non-linear differential equations have similarity present. In this project, I will describe two problems, a blow-up problem and the porous medium equation, which are invariant under similarity transformations of variables. The special solutions for these problems are called self-similar solutions. The purpose of the project is to study the evolution of the self-similar solutions of these partial

a numerical method using

mesh movement can approximately preserve this property to give good results when solving problems with self-similarity.

solutions (SSS). In chapter 2 I introduce how to find scale invariance and SSS for some differential equations. In section 2.1 and 2.2, I show the method for finding the scale invariant and SSS for a blow-up equation and the PME. For a more practical problem, in section 2.3, I consider the scale invariance and SSS for a more general form of PME. Also we can find the SSS exactly for this problem. In section 2.4, I discuss the exact solution of this more general PME when time varies.

In chapter 3, I introduce the numerical method. In section 3.1, I describe the numerical approximation for moving boundary problems based on conservation

1.2 Numerical methods

Numerical methods are approximate calculation methods which are methods of solving problems by using computers. Many differential equations cannot be solved exactly, therefore we need to find approximate solutions, and then check that the numerical method was a good method to be applied. Numerical approximation does not find the exact answer, it usually obtains an approximate solution while maintaining a reasonable range of errors. The approximate solution can be quite accurate if it is stable and convergent.

For example, we can use the numerical approximation on the numerical weather prediction because we cannot predict the exact weather report.

In general, numerical methods are not designed specifically to approximate the self-similarity of PDEs. In fact, some of the general numerical methods can give a worse performance for long time behaviour.

1.3 Scale invariance

1.4 Why use a moving mesh Numerical method on scale invariant solutions

The reason for seeking a numerical method for scale invariant problems is to be able to preserve the qualitative properties of differential equation, and to keep the error as small as possible. It is a technique to give an approximate solution of general scale-invariant non linear problems. The approximate solution can be quite accurate. If a moving mesh method has been used, the error will remain really small (acceptable) provided that the method is stable. Some differential equations have to solve over an infinite

2.1.2 Porous medium equation

or

Similar to the blow-up equation, scale all the variables t, x and u, and make the equation be invariant. Assume that

Using Euler scheme to find the self-similar solutions, we substitute back the initial conditions to equation (17) and (18), and assume Solved by C++ program

We found the self similar solution as showed below:

From the data, when .

Next we use Runge-Kutta 4 scheme to find the self-similar solutions when time equal

Now, the next step is to increase the accuracy of the approximation. We have applied the shooting method to find better results.

Use Shooting Method
Start at with guess ,
Step forward in z using

until reaching . If value of g at is not zero

. p-7(c)6(r)9.25 0 0 1 0 0dTJcrave

Finding the self-similar solution

For more general form of porous medium equation

From equation (8), we find

By substituting and

Finding the ODE form for solving :

2.4 Exact solution of Porous medium equation for different m values when time varies:

From these figures, we can see the evolution of self-similar solutions for the Porous Medium Equation

and so on. When m is equal to 4, and the time also changes from 1 second to 4 seconds, we can6(w)15957B4ew(m)] TJETBT1 0 0 1 1.8025 725.17 Tm0(th)-7(a)1t4( )] TJETBT1 0 0 1 216

Chapter 3 Numerical method

3.1 Numerical method for moving boundary problems

A.) Mapping (or transformation) maps moving boundary to fixed region

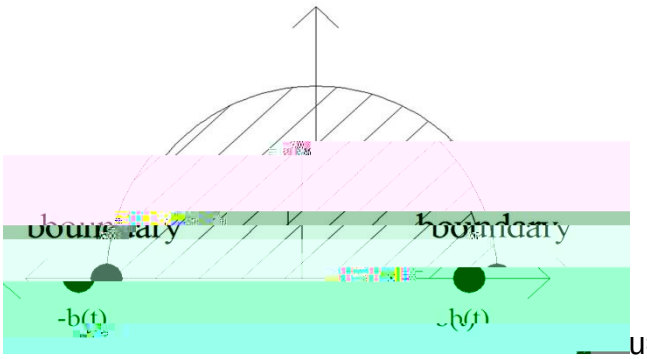
How to map?

B.) Generate a velocity which moves points

What is velocity?

Need a strategy to either map or move nodes,

Consider a velocity approach, for the more general Porous medium equation



___u=0 at boundaries

Area =

— — ——— ——— ——— ———

where —

Therefore the area is constant.

The Motivation for using Numerical Method, from reference [7], is given in the Theorem in the Appendix.

3.1.1 Generate velocities using Conservation and Leibnitz Rule

For a numerical method, take area of a section to be constant (conserved)

as in [8], from which we can generate velocities using Leibnitz Rule

—

where are velocities. Hence

We know , put i = 1

gives ,

Then put i = 21 90es

3.2 Example of fi

For i=N, the last point

if m=3, we have general formula:

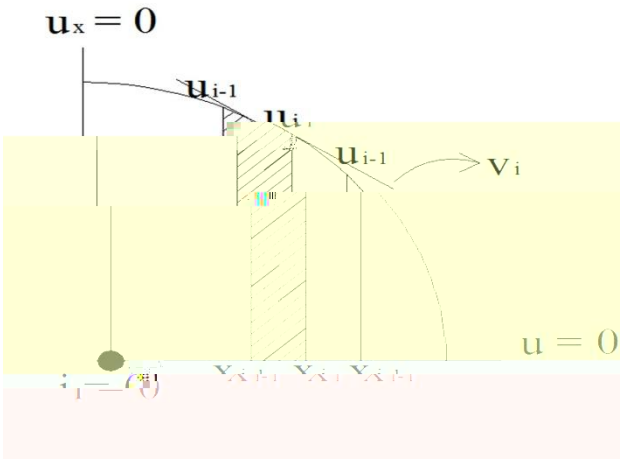
For i=0,

For i=1,2,.....,N-1, using the central difference formula to generate .

For i=N, the last point

3.3 Formula to find the velocities for general m

The formula for finding velocities for general m is



For i=0,

For i=1,2,.....,N-1, using the central difference formula to generate .

For i=N, the last point

3.4 Numerical method of calculating the x values and u values when the time changes

Firstly, calculate the new x value from the velocity using explicit Euler scheme

3.5 Error calculations between the exact solution and numerical solution

1. Calculate the error in boundary position

   Error in boundary

   which is equal to the difference between the value of using numerical method and the value of which is the exact self-similar solution boundary point.

2. Calculate the error in solution
   a. " error "

   b. " error "

   c. " error "

   where are the approximate solution values using the numerical method and are the corresponding values of the exact solution.

The reason of calculating those errors

First, we want to determine the accuracy of getting the approximate solution using the numerical method. Secondly, we try to find out a way to improve the accuracy of the result by changing the parameters in the numerical method.

Method to increase the solutions accuracy

In the next chapter, I show two different ways of improving the results. First, I increase the numbers of points and compare the accuracy of the results from different numbers of points.

Second, I decrease the size of the time steps to check what will happen to the solution accuracy with different numbers of time steps.

Chapter 4. Error calculation with different points step and time steps

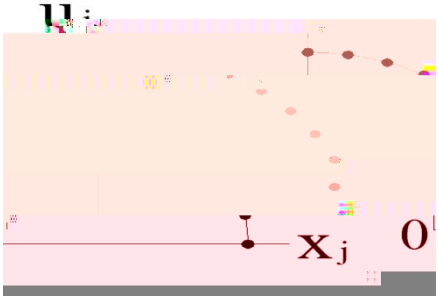
4.1 Error calculation with 10 points for time step equal to 0.01

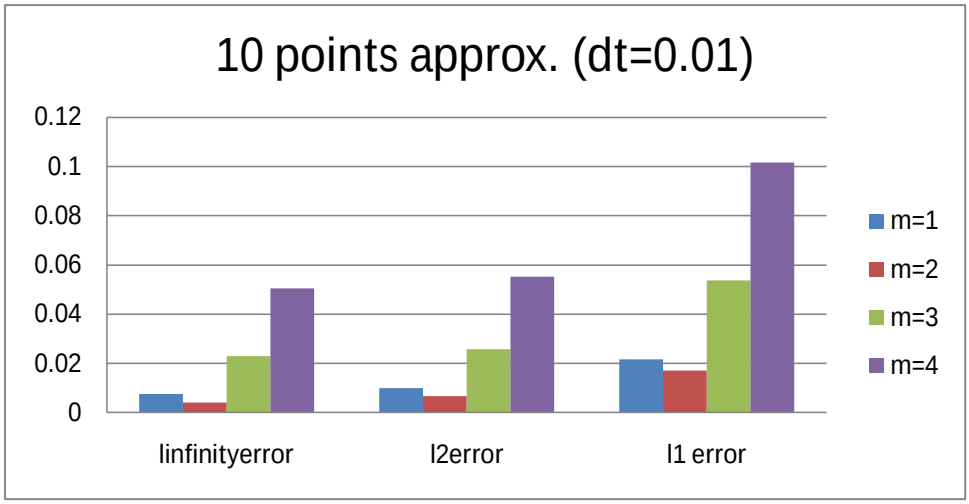
Starting with the time step equal to 0.01 and the number of points equal to 10, calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

How to apply the numerical method to generate the result

At the beginning, we start with the exact solution when time equal to 1. Then we apply the numerical method to approximate the solution with different time step until the time is equal to 2.

In the first case, the time steps are equal to 0.01, which means we need to run the program for 100 times to reach time equal to 2. We take the number of points to be equal to 10. Then we compare the error between the solution using numerical calculation and the exact solution for different values of m.





10 points approx. (dt=0.01)				
	linfinityerror	l2error	l1 error	boundary error
m=1	0.00760356	0.00995198	0.0216289	0.00720774
m=2	0.00403803	0.00673991	0.0171885	0.00495405
m=3	0.0229165	0.0258753	0.053699	0.0270282
m=4	0.0504221	0.055299	0.101573	0.047171

Error calculation with 20 points for time step equal to 0.01

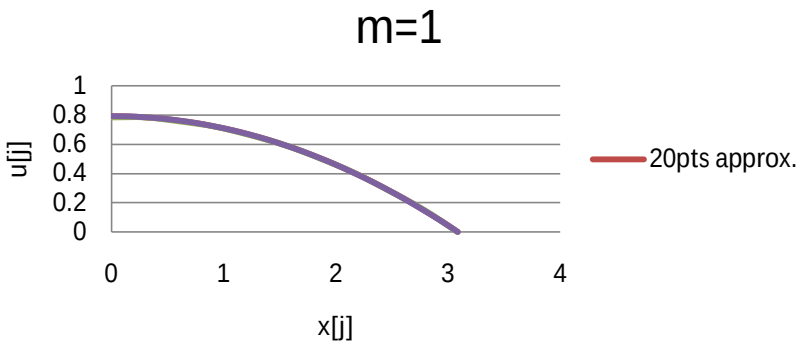
Now we keep the time steps equal to 0.01 and change the number of points to 20. Then calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

The time steps are equal to 0.01, so we need to run 100 steps to reach time equal to 2.

20 points approx. (dt=0.01)				
	linfinityerror	l2error	l1 error	boundary error
m=1	0.00209553	0.00364496	0.0115186	0.000793082
m=2	0.00119655	0.00270182		

Approximate solutions for different points of approximation for time step 0.01

The diagram below show the solutions for different points of approximation when time equal to 2 with the time step equal to 0.01 and compare with the exact solution when time equal to 2.

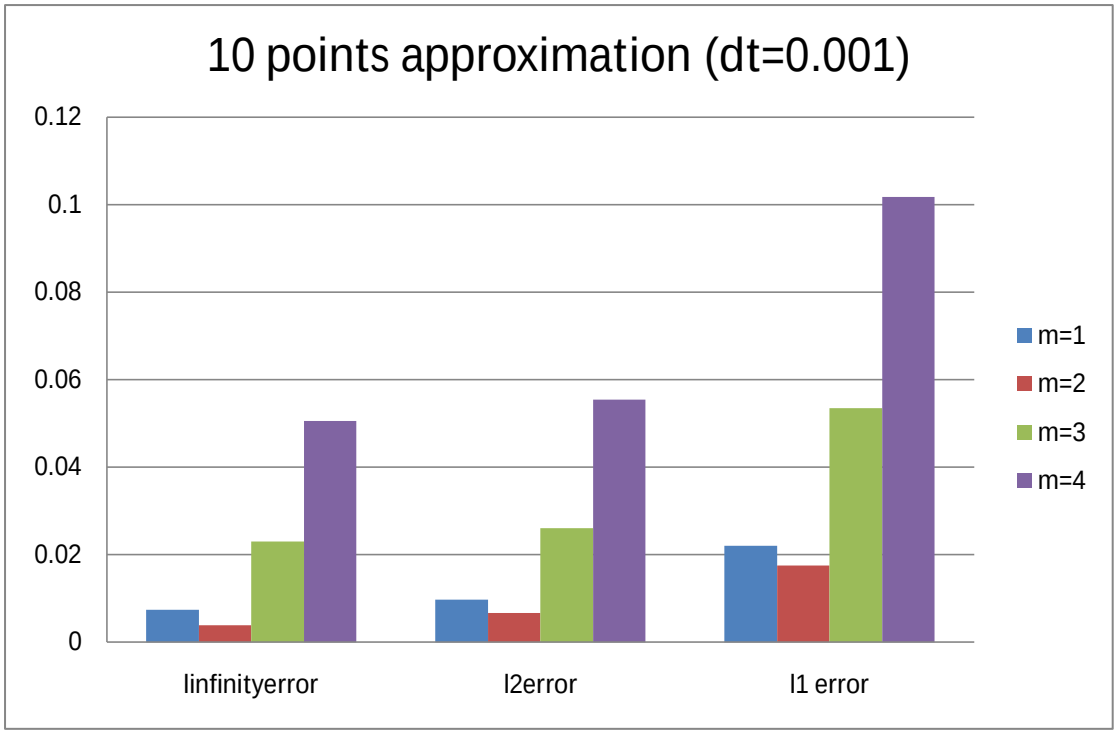


4.2 Error calculation with 10 points for time step equal to 0.001

Because we have an unstable solution when the number of points is 50 and the time step equal to 0.01. We decrease the time step equal to 0.001 to keep away of the unstable solution.

Now we change the time steps equal to 0.001 and the number of points equal to 10. Then calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

The time steps are equal to 0.001, so we need to run 1000 steps to reach time equal to 2.

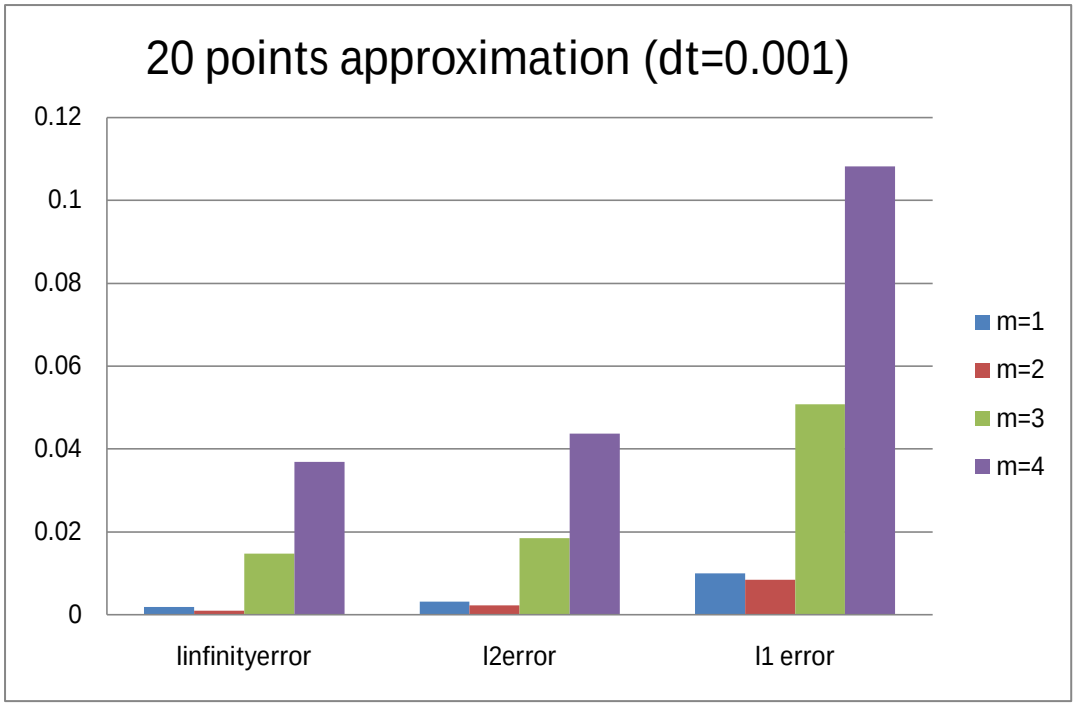


10 step approximation (dt=0.001)				
	linfinityerror	l2error	l1 error	boundary error
m1	0.00743814	0.009751	0.022011	0.00829078
m2	0.00386414	0.00659	0.017518	0.00565947
m3	0.0230331	0.025979	0.053528	0.0275071
m4	0.0505132	0.055411	0.101788	0.0474908

Error calculation with 20 points for time step equal to 0.001

Now we keep the time steps equal to 0.001 and change the number of points to 20. Then calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

The time steps are equal to 0.001, so we need to run 1000 steps to reach time equal to 2.



20 points approximation (dt=0.001)				
	linfinityerror	l2error	l1 error	boundary error
m1	0.00188328	0.003178	0.009989	0.00187345
m2	0.00099176	0.00228	0.008471	0.00141046
m3	0.0147765	0.018468	0.050825	0.0138743
m4	0.036926	0.043755	0.108232	0.0258484

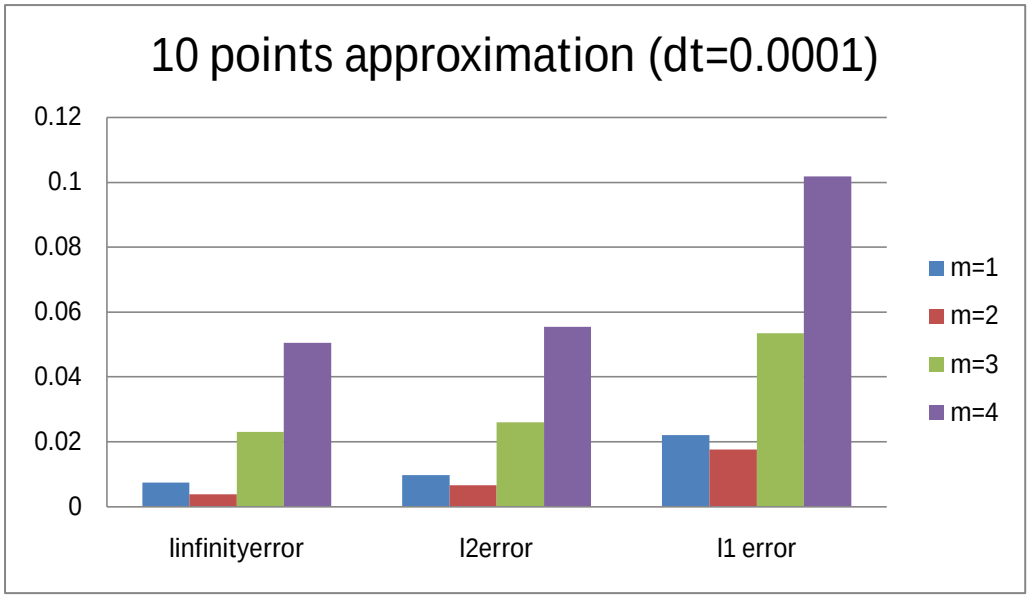
Error calculation with 50 points for time step equal to 0.001
Now we keep

4.3 Error calculation with 10 points for time step equal to 0.0001

We have a stable solution for the time step equal to 0.001 and the numbers of point's approximation equal to 10, 20 and 50. Next, we try to decrease the time step equal to 0.0001 to generate a more accurate solution.

Now we change the time steps equal to 0.0001 and the number of points equal to 10. Then calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

The time steps are equal to 0.0001, so we need 10000 numbers of steps to reach time equal to 2.

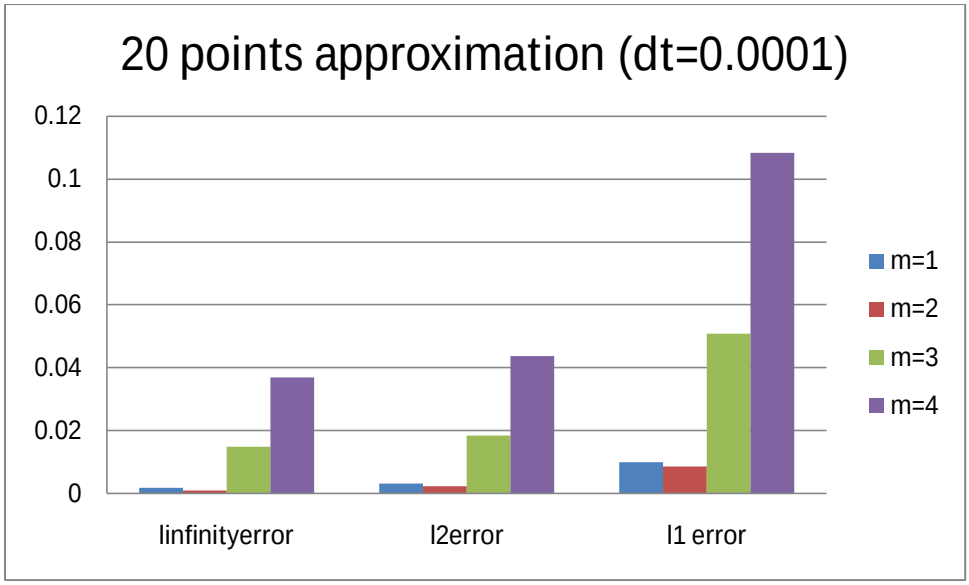


10 points approx.				
	linfinityerror	l2error	l1 error	boundary error
m=1	0.00742051	0.00973175	0.0220496	0.00839903
m=2	0.00384594	0.00657707	0.017551	0.00573011
m=3	0.0230448	0.02599	0.0535083	0.0275551
m=4	0.0505223	0.0554225	0.101808	0.0475228

Error calculation with 20 points for time step equal to 0.0001

Now we keep the time steps equal to 0.0001 and change the number of points to 20. Then calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

The time steps are equal to 0.0001, so we need to run 10000 steps to reach time equal to 2.

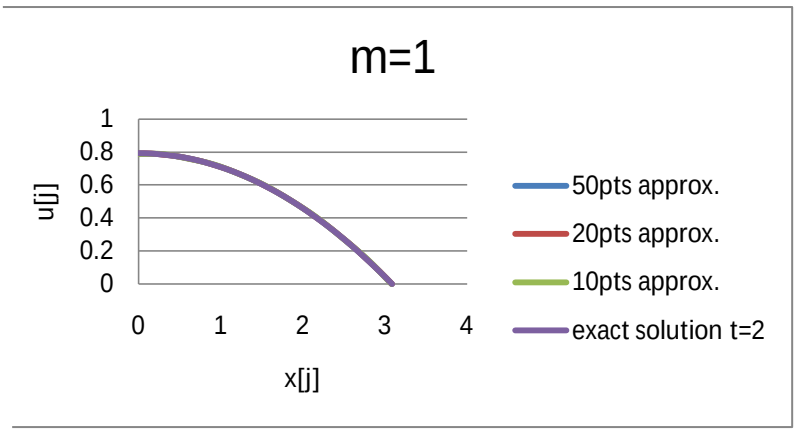


20 points approximation (dt=0.0001)				
	l infinity error	l2error	l1 error	boundary error
m=1	0.00186101	0.003139	0.0100273	0.00198134
m=2	0.000970518	0.002251	0.00850487	0.0014809
m=3	0.0147896	0.018484	0.0507885	0.0139262
m=4	0.0369427	0.043777	0.108301	0.0258883

Error calculation with 50 points for time step equal to 0.0001

Approximate solutions for different points of approximation when dt=0.0001

The diagram below show the solutions for different points of approximation when time equal to 2 with the time step equal to 0.0001 compared with the exact solution when time equal to 2.



150 ref*145.8 478.2 123 1 0 0 1 131.25 4

4.4 Discussions on the results

At the beginning, we start the numerical approximation with

the only difference is for the 50 points approximation the time steps equal to 0.001 which is stable. Next, we try to decrease the time step equal to 0.0001 to generate more accurate solution.

In section 4.3, the time steps are equal to 0.0001, which means the

5 Conclusions and future work

In chapter 1, I introduced the objective of the project which was to investigate whether the accuracy of a moving mesh numerical approximation based on conservation gave a good prediction. And also I defined what are Numerical

In chapter 4, I found the results of the error calculation with different numbers of points and time steps. And I brief discussed on the errors and the approximate solutions. The method gives good approximations if the solution was stable. If we have more point then the solution will be more accurate if the solution is stable. And also if we increased the number of time steps, it will ensure the stability of the solutions. Normally, if we increase the time step and keep the number of points unchanged, the results should be getting closer and closer to the exact solution and should converge to the exact solution. But for the results we found, did not show this situation. The solution does not get closer and closer to the perfect solution and converge. The reasons for this are due to the round-off errors from the computer.

The idea of the dissertation was to investigate the suggestion in the Theorem in the Appendix that it is possible to get good numerical solutions to problems with self-similar solutions by using the conservation principle ( ). We have shown that approximations found in this way are good approximations to self-similar solutions.

For future work, I want to find out the self-similar solution for other Partial differential equations which can provide an exact solution. Then, I can apply the numerical approximation again to investigate whether or not the method will provide good approximations to the sss. And also I could find out a method to improve the error. For example, try to change the time steps and points step jointly in some ratio. Possibly this will be provided better results. Because of time I cannot work out the self-similar solution for blow-up time using RK4 scheme. Furthermore, we can find the order of accuracy p.

Method to find the order of accuracy p

where

6. Appendix

Theorem:
Given that is a positive solution of a scale-invariant mass-conserving problem governed by the PDE

in the interior of the interval , if

1. satisfies the local conservation principle

for all and all

2. at the endpoints either

or

is the similarity velocity,

for all

3. the initial condition coincides with a self-similar scaling solution at time

then

remains self-similar for all and all

is the similarity velocity — for all and all

In other words, initial data which coincides with a self-similar solution at time is propagated as a self-similar solution with the self-similar velocity for all time.