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

Declaration

"I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged"

Signature

_____

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 differential equation equations (PDE's) and estimate how a numerical method using mesh movement can approximately preserve this property to give good results when solving problems with self-similarity.

Contents

1.) Background

1.1 Introduction

Objective:

In this project, I will construct self-similar solutions for two different partial differential equations (PDEs) and approximate them using a moving mesh Numerical method. The aim of doing this is to predict an approximate solution, because many differential equations are not able to be solved exactly. Before we apply the numerical method, we quote a Theorem which states that movement based on conservation principle preserves. We have to investigate that numerical methods will give us a good self-similar solutions.

In this project, I will describe two problems, a blow-up equation and the porous medium equation. For the blow-up equation, it is not possible to solve for the self-similar solution exactly. Therefore I have used an Euler scheme and Runge-Kutta 4 scheme to solve the self-similar solutions approximately. For the Porous medium equation(PME), I can solve the solution exactly, so approximation is not necessary. However we want to check the accuracy of the numerical approximation. Therefore, in the dissertation, I will also compare the numerical approximation with the PME exact solution.

This project introduces the method of finding self-similar solutions. I begin in Section 1.2 and 1.3, explaining what scale invariance and self-similar solutions are. In Section 1.4, I have pointed out the reason for using a moving mesh Numerical method based on conservation on scale invariant solutions.

In order to easily understand the concept of scale invariant and self-similar

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. Firstly, I generate velocities using Leibnitz Rule. Secondly, I move the boundaries with velocities using Euler scheme. Lastly, I find the solutions using the mid-point rule. To better understand how to find the velocities, in section 3.2, I have given an example to show the method. In section 3.3, I find a formula for solving the velocities for the more general PME. In section 3.4, I point out how to use Euler method and mid-point rule to calculate the x values and u values when the time changes. In section 3.5, I show the method to calculate the error between exact solution and the numerical solution.

In chapter 4, I show the results of the error calculation with different numbers of points and time steps. In section 4.4, I briefly discuss the errors and the approximate solutions.

In chapter 5, I sum up the project by means of conclusions and recommendations for future work. In chapter 6 appendix, I state the theorem which suggests that good numerical results to self-similar problems can be obtained using the conservation principle. During the project, I read references [1],[2],[3] and [4] improve the ideas.

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

In mathematics, scale invariance refers to the invariance of a quality or state, called scaling symmetry. Scale invariance is a function of the object or the laws that does not change if length scales (or time scales) are multiplied by a power of a common factor. The quality or state is invariant when the scale has changed, either larger or smaller. Dilatation (also known as the dilation) is the technical terms for this transformation.

For example, a function f(x) is called scale invariant if a fixed amount of scaling of x does not change the shape of the function. In mathematics, the property of scale invariance is written as: f(Yx) = h f(x) for fixed numbers Y and h.

For example the length of coastline varies with size but after scale transformation the coastline of fractal dimension is unchanged.

C++ programming

Type of computer programming that can do simultaneous updates in discrete time steps according to given rule and have infinite update.

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 domain, by using numerical method it will be automatic and able to work with initial data or boundary conditions.

Also, the problem can be solved by a computing program. Therefore we can be input large parameter ranges and solve by the computer system infinitely often. Also, the results can be tested for accuracy, as we can have a good prediction for the self-similar solutions. Using the program the system will be faster and easier to predict when different conditions have been input.

Chapter 2

2.1 Scale Invariance

2.1.1 Blow up equation

$$u_t = u_{xx} + u^2 \qquad (1)$$

or can be written as the form

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + u^2$$

Scale all the variables t, x and u, to make the equation invariant.

Assume that

$$t = \lambda t'$$
$$x = \lambda^{\beta} x'$$
$$u = \lambda^{\gamma} u'$$

where γ and β constants to be determined.

Therefore we have,

$$\frac{\partial u}{\partial t} = \frac{\lambda^{\gamma}}{\lambda}\frac{\partial u'}{\partial t'} = \lambda^{\gamma-1}\frac{\partial u'}{\partial t'}$$
$$\frac{\partial^2 u}{\partial x^2} = \frac{\lambda^{\gamma}}{\lambda^{2\beta}}\frac{\partial^2 u'}{\partial x'^2} = \lambda^{\gamma-2\beta}\frac{\partial^2 u'}{\partial x'^2}$$
$$u^2 = \lambda^{2\gamma} u'^2$$

Substituting back into the blow up equation, we get

$$\lambda^{\gamma-1}\frac{\partial u'}{\partial t'} = \lambda^{\gamma-2\beta}\frac{\partial^2 u'}{\partial x'^2} + \lambda^{2\gamma} u'^2 \qquad (2)$$

For the equation (1) to be unchanged, need all the powers of λ to be equal.

We need to make $\lambda^{\gamma-1}$, $\lambda^{\gamma-2\beta}$ and $\lambda^{2\gamma}$ are equal, therefore we need

$$\gamma - 1 = \gamma - 2\beta = 2\gamma$$

so $\gamma = -1$ and $\beta = \frac{1}{2}$ will make the blow up equation invariant.

The similarity transformation

$$t = \lambda t'$$
$$x = \lambda^{\frac{1}{2}} x'$$
$$u = \lambda^{-1} u'$$

will keep the original equation unchanged.

2.1.2 Porous medium equation

$$u_t = (uu_x)_x$$

or

$$\frac{\partial u}{\partial t} = \frac{\partial\left(u\frac{\partial u}{\partial x}\right)}{\partial x}$$

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

$$t = \lambda t' , \quad x = \lambda^{\beta} x' , \qquad u = \lambda^{\gamma} u'$$

Then we have

$$\frac{\partial u}{\partial t} = \frac{\lambda^{\gamma}\,\partial u'}{\lambda \quad \partial t'} = \lambda^{\gamma-1}\frac{\partial u'}{\partial t'}$$

$$\frac{\partial u}{\partial x} = \frac{\lambda^{\gamma}\,\partial u'}{\lambda^{\beta}\,\partial x'} = \lambda^{\gamma-\beta}\frac{\partial u'}{\partial x'}$$

$$u = \lambda^{\gamma} u'$$

$$u\frac{\partial u}{\partial x} = \lambda^{2\gamma-\beta} u'\frac{\partial u'}{\partial x'}$$

$$\frac{\partial\left(u\frac{\partial u}{\partial x}\right)}{\partial x} = \frac{\lambda^{2\gamma-\beta}\,\partial u'\frac{\partial^2 u'}{\partial x'^2}}{\lambda^{\beta}\,\partial x'} = \lambda^{2\gamma-2\beta}\frac{\partial^3 u'}{\partial x'^3}$$

Substituting back into the porous medium equation, we get

$$\lambda^{\gamma-1}\frac{\partial u'}{\partial t'} = \lambda^{2\gamma-2\beta}\frac{\partial^3 u'}{\partial x'^3}$$

For this equation to be unchanged, we need all the powers of λ be equal, therefore we need

$$\gamma - 1 = 2\gamma - 2\beta$$

$$\gamma = 2\beta - 1$$

We have additional information, that u will equal to zero on the boundary (a,b), then by integrating both sides of the porous medium equation, we have:

$$\int_a^b u_t \, dx = \int_a^b (uu_x)_x \, dx$$

$$\frac{d}{dt}\int_a^b u \, dx = uu_x|_a^b = 0$$

Therefore $\int_a^b u \, dx$ will be constant.

From the extra information, total integral $\int u\,dx = \text{constant}$.

By the transformation, we have

$$x = \lambda^{\beta} x' \ , \qquad u = \lambda^{\gamma} u'$$

then

$$\int u\,dx = \lambda^{\gamma+\beta} \int u'\,dx'$$

But we have proved in above that the total integral is constant. Therefore, $\lambda^{\gamma+\beta}$ should be equal to 1. Then we have

$$\gamma + \beta = 0$$

By combining the result $\gamma = 2\beta - 1$ and the result we have from the extra information. We know that if $\gamma = -\frac{1}{3}$ and $\beta = \frac{1}{3}$ will be given the porous medium equation be invariant.

2.2 Self-similar solutions

Under the similarity transformation, solutions which are invariant are special solutions called Self-similar Solutions.

For Blow-up equation and Porous medium equation

By similarity transformation, we have

$$t = \lambda t' \qquad (3)$$

$$x = \lambda^{\beta} x' \qquad (4)$$

$$u = \lambda^{\gamma} u' \qquad (5)$$

where β and γ are known values

Next, we find x and u in term of the time t. From equation (3), we have

$$\lambda = \frac{t}{t'} \qquad (6)$$

By substituting equation (6) into equations (4) and (5), we get

$$x = \frac{t^{\beta}}{t'^{\beta}} x'$$

$$\frac{x}{t^{\beta}} = \frac{x'}{t'^{\beta}}$$

and

$$u = \frac{t^{\gamma}}{t'^{\gamma}} u'$$

$$\frac{u}{t^{\gamma}} = \frac{u'}{t'^{\gamma}}$$

These variables z = $\frac{x}{t^{\beta}}$ and w = $\frac{u}{t^{\gamma}}$ are invariant under the similarity transformation.

For the invariant self-similar solution, let

$$w = f(z) \qquad \text{also invariant for any f} \qquad (7)$$

By substituting the variables z and w into equation (7),

$$\frac{u}{t^{\gamma}} = f(\frac{x}{t^{\beta}})$$

$$\text{So } u = t^{\gamma} f\left(\frac{x}{t^{\beta}}\right) \qquad \text{also invariant for any f} \qquad (8)$$

Equation (8) is the Self-similar solution.

Finding the self-similar solution

$$u = t^{\gamma} f\left(\frac{x}{t^{\beta}}\right) \tag{8}$$

If we can solve for f then we can get the self-similar solution.

2.2.1 Blow-up equation

$$u_t = u_{xx} + u^2 \tag{1}$$

From (8)

$$u_t = \gamma t^{\gamma-1} f\left(\frac{x}{t^{\beta}}\right) + t^{\gamma} f'\left(\frac{x}{t^{\beta}}\right)(-\beta) x t^{-\beta-1} \tag{9}$$

$$u_x = \frac{t^{\gamma}}{t^{\beta}} f'\left(\frac{x}{t^{\beta}}\right)$$

$$u_{xx} = \frac{t^{\gamma}}{t^{2\beta}} f''\left(\frac{x}{t^{\beta}}\right) \tag{10}$$

$$u^2 = t^{2\gamma} \left(f\left(\frac{x}{t^{\beta}}\right)\right)^2 \tag{11}$$

By substituting equation (9), (10) and (11) back to the blow-up equation, we have:

$$\gamma t^{\gamma-1} f\left(\frac{x}{t^{\beta}}\right) - \beta t^{\gamma} f'\left(\frac{x}{t^{\beta}}\right) x t^{-\beta-1} = \frac{t^{\gamma}}{t^{2\beta}} f''\left(\frac{x}{t^{\beta}}\right) + t^{2\gamma} \left(f\left(\frac{x}{t^{\beta}}\right)\right)^2$$

We have variables $z = \frac{x}{t^{\beta}}$ and try to write this into a simple form, we have:

$$\gamma t^{\gamma-1} f - \beta t^{\gamma-1} f' z = \frac{t^{\gamma}}{t^{2\beta}} f'' + t^{2\gamma} f^2$$

For the blow up equation to be invariant, we have $\gamma = -1 \text{ and } \beta = \frac{1}{2}$.

$$t^{\gamma-1} = \frac{t^{\gamma}}{t^{2\beta}} = t^{2\gamma} = t^{-2}$$

Therefore $t^{\gamma-1}$, $\frac{t^{\gamma}}{t^{2\beta}}$ and $t^{2\gamma}$ are equal and can be cancelled from the equation, giving

$$\gamma f - \beta f' z = f'' + f^2$$

Since $\gamma = -1 \text{ and } \beta = \frac{1}{2}$ the simplest ODE form for finding $f(z)$ is written as:

$$-f - \frac{1}{2} f' z = f'' + f^2 \tag{12}$$

ODE system for solving $f(z)$:

$$-f-\frac{1}{2}f'z=f''+f^2 \qquad (12)$$

Firstly, define

$$g=f' \qquad (13)$$

then substitute equation (13) back to the ODE, we have

$$-f-\frac{1}{2}gz=g'+f^2$$

Therefore, we have the system

$$f'=g \qquad (14)$$

$$g'=-f-\frac{1}{2}gz-f^2 \qquad (15)$$

Next step, we can solve these equations by using numerical schemes, for example the Euler scheme and Runge-Kutta 4 schemes. Let $y=\begin{pmatrix}f\\g\end{pmatrix}$ and $F=\begin{pmatrix}f\\g\end{pmatrix}'$.

(A) Euler scheme

$$y_{n+1}=y_n+\Delta z\,F(y_n) \qquad (16)$$

If we substitute equation (14) and (15) back to the Euler scheme, we have:

$$f_{n+1}=f_n+\Delta z\,g_n \qquad (17)$$

$$g_{n+1}=g_n+\Delta z\left(-f-\frac{1}{2}gz-f^2\right) \qquad (18)$$

By advancing f_n and g_n by equations (17) and (18) using a computer program written in C++, we can find $f(z)$ and substitute back to find the self similar solution.

(B) Runge-Kutta 4 scheme

$$y_{n+1}=y_n+\frac{h}{6}(K_1+2K_2+2K_3+K_4) \qquad (19)$$

$$K_1=F(t_n,y_n)$$

$$K_2=F\left(t_n+\frac{h}{2},y_n+\frac{h}{2}K_1\right)$$

$$K_3=F\left(t_n+\frac{h}{2},y_n+\frac{h}{2}K_2\right)$$

$$K_4=F(t_n+h,y_n+hK_3)$$

Similarly, advancing y_n by a computer program in C++, we can find $f(z)$ and substitute back to find the self similar solution.

Numerical evaluation of f for the blow-up equation

$$u_t = u_{xx} + u^2 \qquad (1)$$

The self similar solution for solving the equation is written as below:

$$u = t^{\gamma} f\left(\frac{x}{t^{\beta}}\right)$$

where $\beta = \frac{1}{2}$, $\gamma = -1$

At t = 1 , $u = f(x)$, $z = x$ and $u_x = f'(x) = g(x)$, say

$$f = u$$

$$g = u_x$$

Also, the simplest ODE form for solving $f\left(\frac{x}{t^{\beta}}\right)$ is written as:

ODE system

$$f' = g$$

$$g' = -f - f^2 - \frac{1}{2}gz$$

For example, take initial conditions on f, g to be

$$f(x) = u = \sin \pi x$$

$$g(x) = u_x = \pi \cos \pi x$$

$$z(x) = -1$$

Then we have the initial conditions:

$$f(-1) = 0$$

$$g(-1) = \pi$$

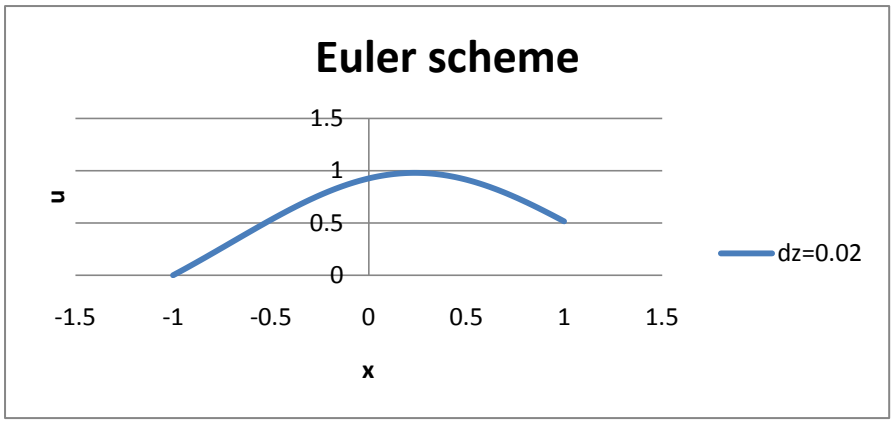
$$z(0) = -1$$

We know that the solution for u will be symmetric, so

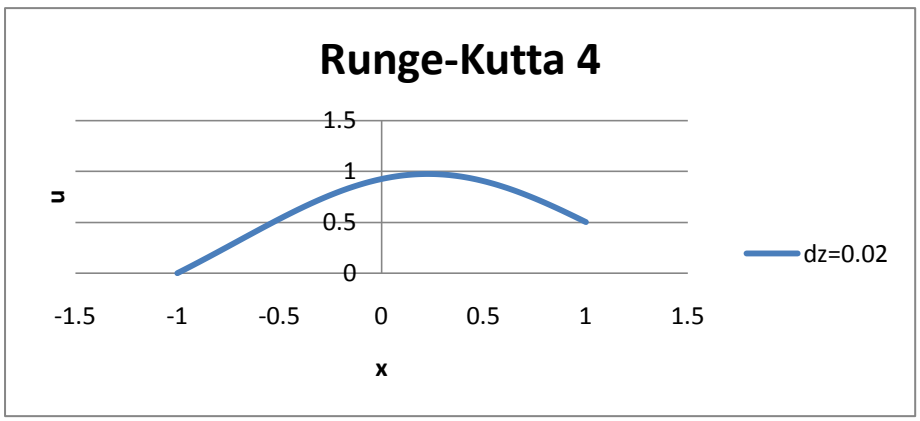
$$-1 \leq z \leq 1$$

Using Euler scheme to find the self-similar solutions, we substitute back the initial conditions to equation (17) and (18), and assume $\Delta z = 0.02$. Solved by C++ program We found the self similar solution as showed below:

From the data, when $x = 0, u = 0.92305, g(0) = 0.425778.$



Next we use Runge-Kutta 4 scheme to find the self-similar solutions when time equal to 1, we substitute back the initial conditions to equation (19), and assume $\Delta z = 0.02$. Solved by C++ program. We found the self similar solution as showed below:



From the data, when $x = 0, u = 0.923826, g(0) = 0.405289$

By compare with these two solutions, it given that using RK 4 scheme for finding the sss will give more approximate than Euler scheme.

where f, g should be satisfy the conditions (at different points)

$$f(-1) = 0$$

$$g(0) = 0$$

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 $z = -1$ with guess $g = \alpha = \alpha_1$,

Step forward in z using $z = j\,dz$

$$f_{j+1} = f_j + dz\, f'_j$$
$$g_{j+1} = g_j + dz\, g'_j$$

until reaching $z = 0$. If value of g at $z = 0$ is not zero,

then try another value of α, let $\alpha = \alpha_2$ and do the same procedure

If g(0) still not equal to zero, then try α_3, and so on.

By using the shooting method, change α until g(0) equal to zero. The values of α should converge and then f is required solution.

The process of finding α can be made automatic. For example, using the bisection method.

<u>Apply shooting method on Euler scheme</u>

Assume $\Delta z = 0.02$ and the time is 1. We guess $\alpha_1 = 2.7$.

then we get $g(0) = -0.02382$ and $f(1) = -0.0789$.

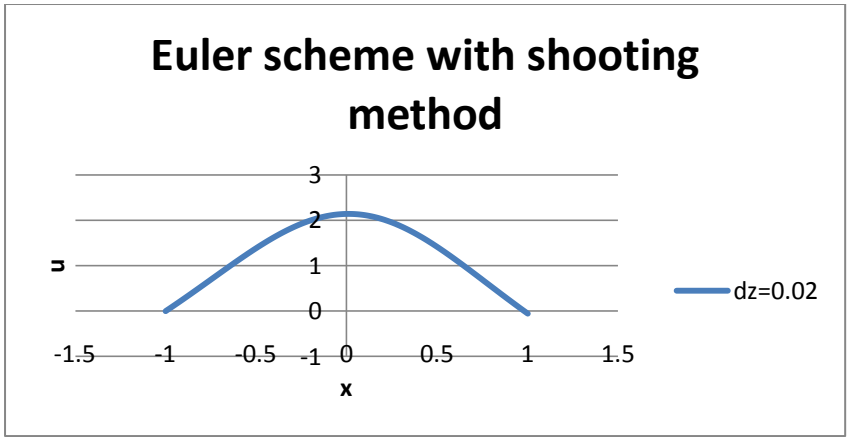
Because the value of g(0) not equal to zero, we guess $\alpha_2 = 2.66$.

then we get $g(0) = 0.000617$ and $f(1) = -0.05487$.

Because the value of g(0) not equal to zero, we keep using the bisection method forα

	α_2	α_1	α_3	α_4
	2.66	2.7	2.68	2.67
g(0)	0.000617	-0.02382	-0.01153	-0.00544

Corresponding to 2 d.p. When $\alpha = \alpha_2 = 2.66$, $f(1) = -0.05487$

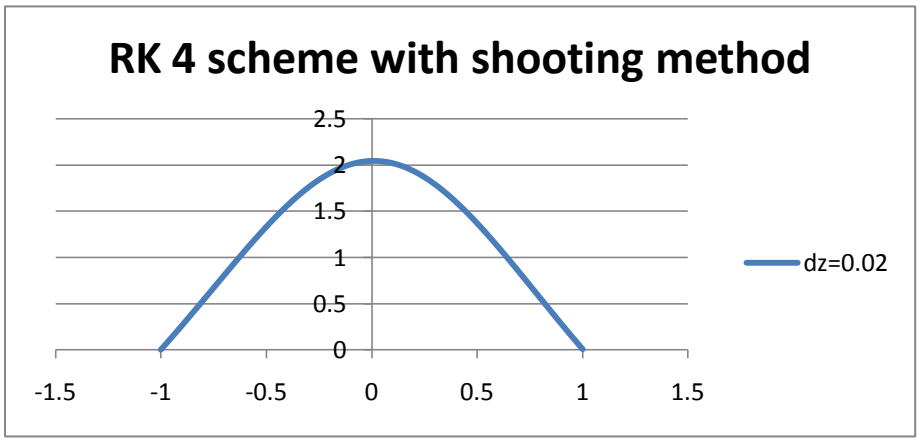


Apply shooting method on Runge-Kutta 4 scheme

Same approaches as applying shooting method on Euler scheme

	α_1	α_2	α_3	α_4	α_5
	2.5	2.6	2.55	2.53	2.54
g(0)	0.022039	-0.03664	-0.00686	0.004809	-0.00101

Corresponding to 2 d.p. When $\alpha = \alpha_5 = 2.54,\ f(1) = -0.004564$



where f, g should be satisfy the conditions (at different points)

$$f(1) = 0$$

$$g(0) = 0$$

When time is equal to 1,

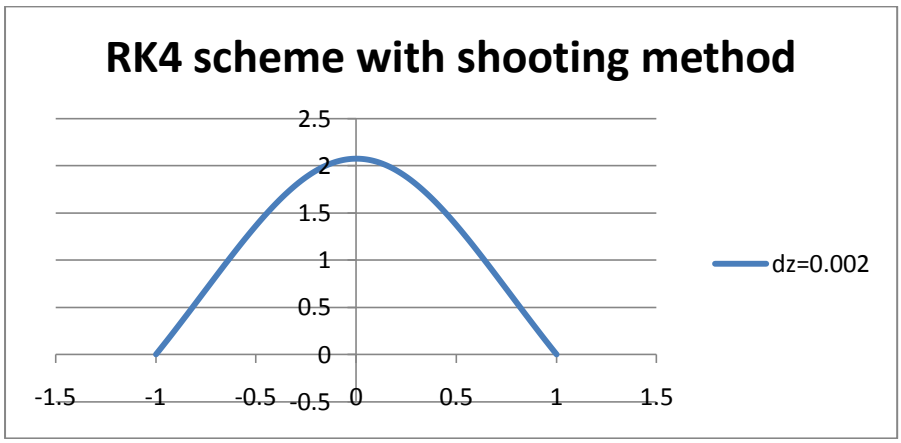
Compared the results of two schemes with shooting method. We can investigate that RK 4 scheme has more accurate prediction on self-similar solution then Euler scheme. Therefore, we have selected RK 4 method to approximate the sss for blow-up equation. But is there any methods can increase the accuracy of the solution.

Increase the numbers of points on the RK4 prediction

Assume $\Delta z = 0.002$ which means the numbers of points on the approximation has increased to 1000 and the time again equal to 1.

	α_1	α_2	α_3	α_4
	2.6	2.65	2.63	2.64
g(0)	0.021802	-0.00687	0.0047	**-0.00044**

Corresponding to 2 d.p. When $\alpha = \alpha_4 = 2.64,\ f(1) = -0.00044$



By increasing the number of points on the approximation, the results are more accurate. Given that for 100 points step, $f(1) = -0.004564$. When increases the points step to 1000, $f(1) = -0.00044$ which is much closer to the boundary conditions $f(1) = 0$.

How to solve the sss for blow-up equation when time changed

The self similar solution for solving the equation is written as below:

$$u = t'^{\gamma} f\left(\frac{x}{t'^{\beta}}\right)$$

where $\beta = \frac{1}{2}$, $\gamma = -1,\ t' = (T - t)$

The transformation from $z[j]$ into $x[j]$,

$$x[j] = z[j]\sqrt{t'}$$

Finding Self-Similar solution

2.2.2 Porous medium equation

$$u_t = (uu_x)_x \tag{20}$$

From (8)

$$u_t = \gamma t^{\gamma-1} f\left(\frac{x}{t^\beta}\right) + t^\gamma f'\left(\frac{x}{t^\beta}\right)(-\beta)xt^{-\beta-1} \tag{21}$$

$$u_x = \frac{t^\gamma}{t^\beta} f'\left(\frac{x}{t^\beta}\right)$$

$$uu_x = \frac{t^{2\gamma}}{t^\beta} f\left(\frac{x}{t^\beta}\right) f'\left(\frac{x}{t^\beta}\right)$$

$$(uu_x)_x = \frac{t^{2\gamma}}{t^\beta}\frac{1}{t^\beta} f'\left(\frac{x}{t^\beta}\right) f'\left(\frac{x}{t^\beta}\right) + \frac{t^{2\gamma}}{t^\beta} f\left(\frac{x}{t^\beta}\right)\frac{1}{t^\beta} f''\left(\frac{x}{t^\beta}\right)$$

$$= t^{2\gamma-2\beta} f'\left(\frac{x}{t^\beta}\right) f'\left(\frac{x}{t^\beta}\right) + t^{2\gamma-2\beta} f\left(\frac{x}{t^\beta}\right) f''\left(\frac{x}{t^\beta}\right) \tag{22}$$

By substituting equation (21) and (22) back into the porous medium equation, we have:

$$\gamma t^{\gamma-1} f\left(\frac{x}{t^\beta}\right) + t^\gamma f'\left(\frac{x}{t^\beta}\right)(-\beta)xt^{-\beta-1} = t^{2\gamma-2\beta} f'\left(\frac{x}{t^\beta}\right) f'\left(\frac{x}{t^\beta}\right) + t^{2\gamma-2\beta} f\left(\frac{x}{t^\beta}\right) f''\left(\frac{x}{t^\beta}\right)$$

We have variables $z = \frac{x}{t^\beta}$ and try to write this into a simple form, we have:

$$\gamma t^{\gamma-1} f - \beta t^{\gamma-1} f' z = t^{2\gamma-2\beta} f' f' + t^{2\gamma-2\beta} f f''$$

Given the porous medium equation with zero boundary conditions be invariant, we have $\gamma = -\frac{1}{3}$ and $\beta = \frac{1}{3}$.

$$t^{\gamma-1} = t^{2\gamma-2\beta} = t^{-\frac{4}{3}}$$

Therefore $t^{\gamma-1}$ and $t^{2\gamma-2\beta}$ are equal and can be cancelled from the equation, giving

$$\gamma f - \beta f' z = f' f' + f f''$$

In this case we can solve the ODE exactly.

$$-\frac{1}{3} f - \frac{1}{3} f' z = f' f' + f f''$$

$$-\frac{1}{3}(f + f' z) = f' f' + f f''$$

$$-\frac{1}{3}(\,\mathrm{zf})' = (\mathrm{f'f})'$$

Integrate both sides, we get

$$-\frac{1}{3}\mathrm{zf} = \mathrm{f'f} + \mathrm{C}$$

$$-\frac{1}{3}\mathrm{z} = \mathrm{f'} + \mathrm{C}$$

We have the boundary condition that when $\mathrm{z} = 0,\ \mathrm{f'} = 0$ by symmetry. Therefore we know that C is zero. So

$$-\frac{1}{3}\mathrm{z} = \mathrm{f'}$$

$$\mathrm{f} = -\frac{1}{6}\mathrm{z}^2 + \mathrm{D}$$

By the boundary condition, $\mathrm{f} = 0$ at $\mathrm{z} = 1$, we know that D equal to 1.

$$\mathrm{f} = -\frac{1}{6}\mathrm{z}^2 + 1$$

therefore

$$\mathrm{f(z)} = -\frac{1}{6}\mathrm{z}^2 + 1 \qquad (23)$$

which is positive for $-\sqrt{6} \le \mathrm{z} \le \sqrt{6}$.

[This solution was first obtained by Barenblatt [6] and Pattle[5]]

Next, we substitute the solution of f(z) back to the formula to find the self similar solution. We get the particular solution

$$\mathrm{u} = t^{\gamma}\,(1 - \frac{1}{6}\left(\frac{\mathrm{x}}{t^{\beta}}\right)^2)$$

2.3 More general form of porous medium equation

$$u_t = (u^m u_x)_x \qquad (24)$$

or

$$\frac{\partial u}{\partial t} = \frac{\partial\left(u^m \frac{\partial u}{\partial x}\right)}{\partial x}$$

Similarity

For scale invariance, assume that

$$t = \lambda t' , \quad x = \lambda^\beta x' , \qquad u = \lambda^\gamma u'$$

Then we have

$$\frac{\partial u}{\partial t} = \frac{\lambda^\gamma \partial u'}{\lambda \quad \partial t'} = \lambda^{\gamma-1}\frac{\partial u'}{\partial t'}$$

$$\frac{\partial\left(u^m \frac{\partial u}{\partial x}\right)}{\partial x} = \frac{\lambda^{\gamma m+\gamma-\beta}\,\partial u' \frac{\partial^2 u'}{\partial x'^2}}{\lambda^\beta\,\partial x'} = \lambda^{\gamma m+\gamma-2\beta}\frac{\partial^3 u'}{\partial x'^3}$$

And substitute back to the porous medium equation (24), we get

$$\lambda^{\gamma-1}\frac{\partial u'}{\partial t'} = \lambda^{\gamma m+\gamma-2\beta}\frac{\partial^3 u'}{\partial x'^3}$$

For the equation to be unchanged, need all the powers of λ to be equal. We need to make $\lambda^{\gamma-1}$ and $\lambda^{\gamma m+\gamma-2\beta}$ are equal, therefore we need

$$\gamma - 1 = \gamma m + \gamma - 2\beta$$
$$\gamma m = 2\beta - 1 \qquad (25)$$

Combine with the extra information that the total integral is constant (similar to what we have show previously) that

$$\gamma + \beta = 0$$
$$\gamma = -\beta \qquad (26)$$

Substitute the solution (26) from the extra information back to the equation (25), we have

$$-\beta m = 2\beta - 1$$
$$\beta(2 + m) = 1$$
$$\beta = \frac{1}{2+m} \quad \text{and} \quad \gamma = -\frac{1}{2+m}$$

Finding the self-similar solution

For more general form of porous medium equation

$$u_t = (u^m u_x)_x \qquad (24)$$

From equation (8), we find

$$u_t = \gamma t^{\gamma-1} f\left(\frac{x}{t^\beta}\right) + t^\gamma f'\left(\frac{x}{t^\beta}\right)(-\beta) x t^{-\beta-1}$$

$$u^m = t^{m\gamma}\left(f\left(\frac{x}{t^\beta}\right)\right)^m$$

$$u^m u_x = \frac{t^{m\gamma+\gamma}}{t^\beta}\left(f\left(\frac{x}{t^\beta}\right)\right)^m f'\left(\frac{x}{t^\beta}\right)$$

$$(u^m u_x)_x = \frac{t^{m\gamma+\gamma}}{t^\beta}\left\{m\left(f\left(\frac{x}{t^\beta}\right)\right)^{m-1} f'\left(\frac{x}{t^\beta}\right)\frac{1}{t^\beta} f'\left(\frac{x}{t^\beta}\right) + \left(f\left(\frac{x}{t^\beta}\right)\right)^m f''\left(\frac{x}{t^\beta}\right)\frac{1}{t^\beta}\right\}$$

By substituting u_t and $(u^m u_x)_x$ back to the porous medium equation, we have:

$$\gamma t^{\gamma-1} f - \beta t^{\gamma-1} f' z = \frac{t^{m\gamma+\gamma}}{t^\beta}\left\{m f^{m-1} f' \frac{1}{t^\beta} f' + f^m f'' \frac{1}{t^\beta}\right\}$$

$$\gamma t^{\gamma-1} f - \beta t^{\gamma-1} f' z = \frac{t^{m\gamma+\gamma}}{t^{2\beta}}\{m f^{m-1} f' f' + f^m f''\}$$

Given the porous medium equation be invariant, with $\gamma = -\frac{1}{2+m}$ and $\beta = \frac{1}{2+m}$.

$$t^{\gamma-1} = t^{m\gamma+\gamma-2\beta} = t^{\frac{-3-m}{2+m}}$$

Therefore $t^{\gamma-1}$and $t^{2\gamma-2\beta}$ are equal and can be cancelled from the equation, giving

$$\gamma f - \beta f' z = m f^{m-1} f' f' + f^m f''$$

Finding the ODE form for solving $f(z)$:

$$-\frac{1}{2+m}(f+f'z) = (f^m f')'$$

$$-\frac{1}{2+m}(zf)' = (f^m f')'$$

Integrate both sides, we get

$$-\frac{1}{2+m}zf = f^m f' + C$$

We have the boundary conditions that when z = 0, $f' = 0$. And also when z = 0, f = 1. Therefore we know that C is zero. So

$$-\frac{1}{2+m}zf = f^m f'$$

$$-\frac{1}{2+m}z = f^{m-1}f'$$

$$-\frac{1}{2+m}z = \frac{(f^m)'}{m}$$

$$-\frac{m}{2+m}z = (f^m)'$$

$$f^m = -\frac{mz^2}{2(2+m)} + D$$

By the boundary conditions $f = 0$ at $z = 1$, we know that D equal to 1. Therefore

$$f(z) = \left(1 - \frac{mz^2}{2(2+m)}\right)^{\frac{1}{m}}$$

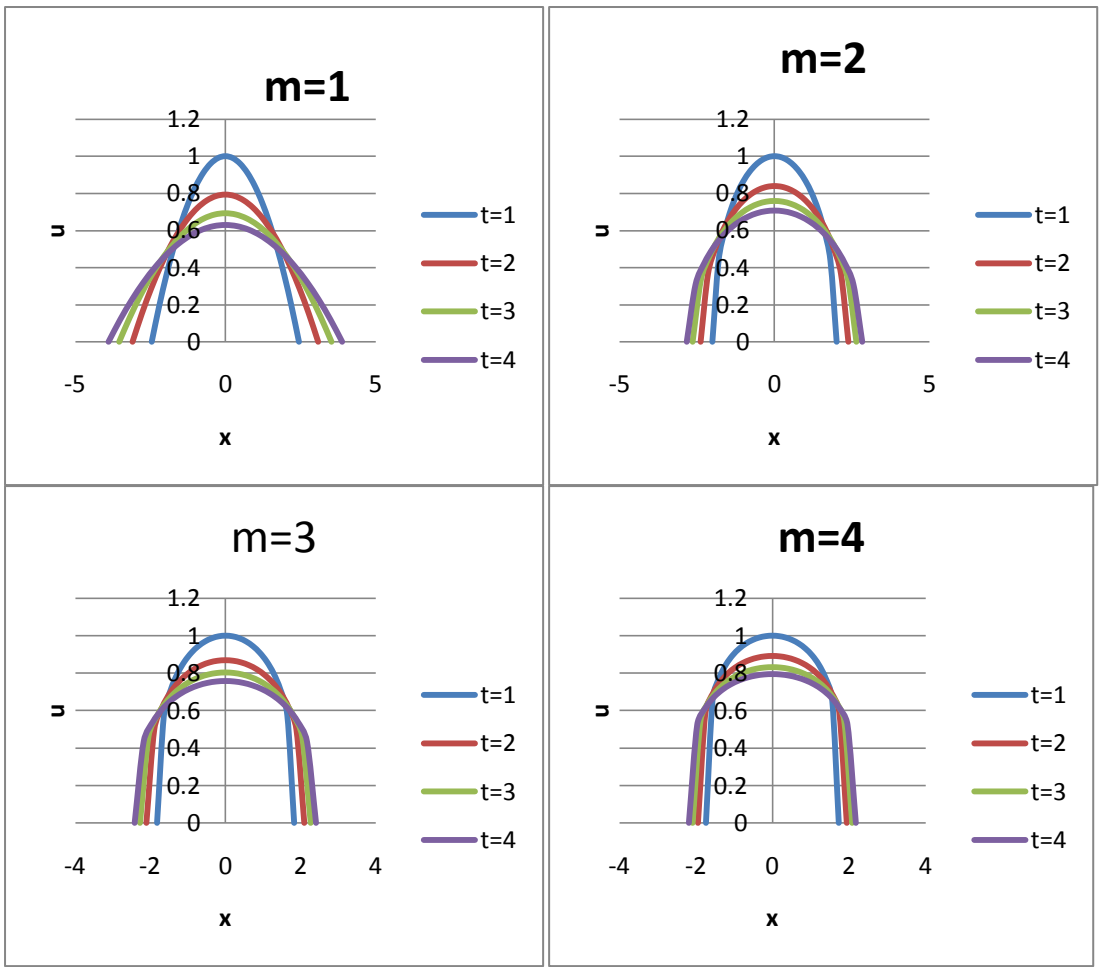
which is positive for $-\sqrt{\frac{4+2m}{m}} \le z \le \sqrt{\frac{4+2m}{m}}$.

Next, we substitute the solution of f(z) back to the formula solving the self similar solution. We get the particular solution

$$u = t^{\gamma}\left(1 - \frac{mz^2}{2(2+m)}\right)^{\frac{1}{m}}$$

(also first found by Barenblatt [6] and Pattle[5].)

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 when time is changed. The boundaries move.

When m is equal to 1, and the time is changed from 1 second to 4 seconds, we can see the values of u become smaller, and the distance between the x become larger, which means that the values of x will become larger. When m is equal to 2, and the time also change from 1 second to 4 seconds, we can see that some of the values of u are larger than in the m equal to 1 case, and the values of x are smaller than that when m equal to 1. When m is equal to 3, and the time also change from 1 second to 4 seconds, we can see the time is longer, but the difference between the values of u and the values of x are smaller than that when m is equal to 1 and m equal to 2 cases,

and so on. When m is equal to 4, and the time also changes from 1 second to 4 seconds, we can see that the values of u are almost the same and the values of x also almost the same. In conclusion, we can see that when m is getting larger, the effect of the values u and the values x will be smaller in terms of the change of time. When m gets larger and time remains the same, the support of x becomes narrower.

Chapter 3 Numerical method

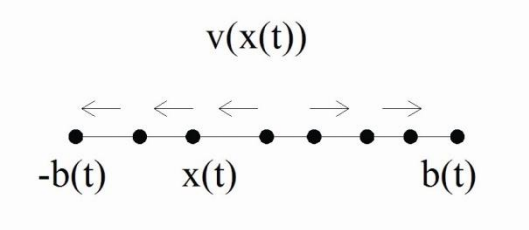
3.1 Numerical method for moving boundary problems

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

How to map?

B.) Generate a velocity which moves points

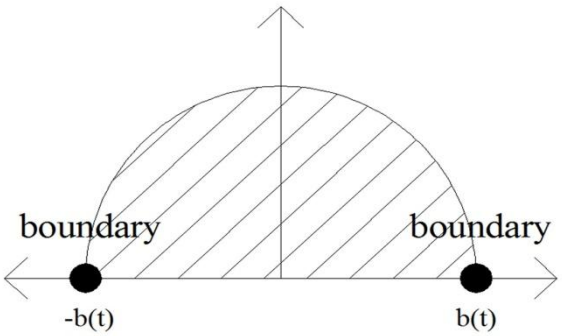
What is velocity?



Need a strategy to either map or move nodes,

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

$$u_t = (u^m u_x)_x$$



u=0 at boundaries

$$\text{Area} = \int_{-b(t)}^{b(t)} u(x,t)dx = \int_0^{b(t)} u(x,t)\,dx + \int_{-b(t)}^{0} u(x,t)\,dx$$

$$\frac{d}{dt}(\text{Area}) = \int_{-b(t)}^{b(t)} \left(\frac{\partial u}{\partial t}\right) dx + \frac{d\,b(t)}{dt} \times \frac{d}{d\,b(t)} \int_0^{b(t)} u\,dx - \frac{d\,b(t)}{dt} \times \frac{d}{d\,b(t)} \int_{-b(t)}^{0} u\,dx$$

$$= \int_{-b(t)}^{b(t)} (uu_x)_x\,dx + \dot{b}\,u|_{b(t)} - \dot{b}\,u|_{-b(t)}$$

where $\dot{b} = \frac{d\,b}{dt}$

$$= uu_x|_{-b(t)}^{b(t)} + 0 - 0$$

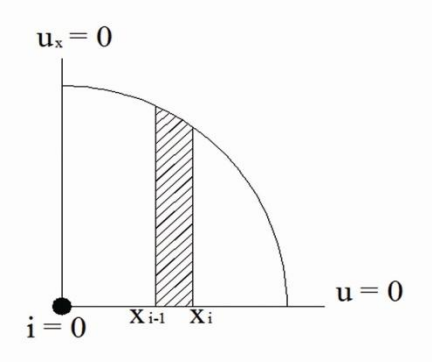
$$= 0$$

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)



$$\int_{x_{i-1}(t)}^{x_i(t)} u\,dx = \text{constant} \qquad (*)$$

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

$$\frac{d}{dt}\int_{x_{i-1}(t)}^{x_i(t)} u dx = 0 = \int_{x_{i-1}(t)}^{x_i(t)} (uu_x)_x dx + \dot{x_i}u_i - \dot{x_{i-1}}u_{i-1}$$

where $\dot{x_i}$, $\dot{x_{i-1}}$ are velocities. Hence

$$0 = uu_x|_{x_{i-1}(t)}^{x_i(t)} + \dot{x_i}u_i - \dot{x_{i-1}}u_{i-1} \qquad \text{true for } i = 1, \dots, N$$

We know $\dot{x_0} = 0$, put i = 1

$$uu_x|_{x_0(t)}^{x_1(t)} + \dot{x_1}u_1 - \dot{x_0}u_0 = 0$$

$$uu_x|_{x_0(t)}^{x_1(t)} + \dot{x_1}u_1 = 0$$

$$\dot{x_1} = -u_x|_{x_0(t)}^{x_1(t)}$$

gives $\dot{x_1}$,

Then put i = 2,

$$uu_x|_{x_1(t)}^{x_2(t)} + \dot{x_2}u_2 - \dot{x_1}u_1 = 0$$

gives $\dot{x_2}$, and continue to get all velocities, until i = N.

Alternative formula to find the velocities

Again, taking the area of section is constant in time

$$\int \mathrm{u}\,\mathrm{dx} = \text{constant in time}$$

for any interval. Then,

$$\frac{\mathrm{d}}{\mathrm{dt}}\int \mathrm{u}\,\mathrm{dx} = 0$$

Using Leibnitz Rule, we get

$$\int \left[\frac{\partial \mathrm{u}}{\partial \mathrm{t}} + \frac{\partial}{\partial \mathrm{x}}\left(\mathrm{u}\frac{\mathrm{dx}}{\mathrm{dt}}\right)\right]\mathrm{dx} = 0 \qquad (**)$$

For Porous medium equation, we have

$$\frac{\partial \mathrm{u}}{\partial \mathrm{t}} = \frac{\partial}{\partial \mathrm{x}}\left(\mathrm{u}^{\mathrm{m}}\frac{\partial \mathrm{u}}{\partial \mathrm{x}}\right)$$

Substitute the PME back to the equation$(**)$,

$$\int \left[\frac{\partial}{\partial \mathrm{x}}\left(\mathrm{u}^{\mathrm{m}}\frac{\partial \mathrm{u}}{\partial \mathrm{x}}\right) + \frac{\partial}{\partial \mathrm{x}}\left(\mathrm{u}\frac{\mathrm{dx}}{\mathrm{dt}}\right)\right]\mathrm{dx} = 0$$

Then, since this integral is zero for any interval,

$$\mathrm{u}^{\mathrm{m}}\frac{\partial \mathrm{u}}{\partial \mathrm{x}} + \mathrm{u}\frac{\mathrm{dx}}{\mathrm{dt}} = 0$$

$$\frac{\mathrm{dx}}{\mathrm{dt}} = -\mathrm{u}^{\mathrm{m}-1}\frac{\partial \mathrm{u}}{\partial \mathrm{x}} \quad \text{or}$$

$$= -\frac{1}{\mathrm{m}}\frac{\partial}{\partial \mathrm{x}}(\mathrm{u}^{\mathrm{m}})$$

Therefore the general formula for finding velocities is

$$\frac{\mathrm{dx}}{\mathrm{dt}} = -\mathrm{u}^{\mathrm{m}-1}\frac{\partial \mathrm{u}}{\partial \mathrm{x}} \quad \text{or}$$

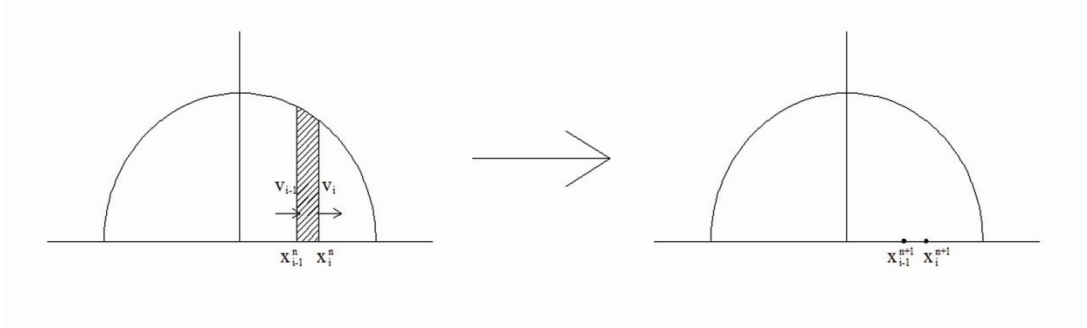
$$= -\frac{1}{\mathrm{m}}\frac{\partial}{\partial \mathrm{x}}(\mathrm{u}^{\mathrm{m}})$$

3.1.2 Move boundaries with velocity using Euler scheme

Move nodes with velocities $\dot{x}_i$ using

$$x_i^{n+1} = x_i^n + \Delta t\, \dot{x}_i \qquad \text{(Euler scheme)}$$

which gives positions of x_i at next time step. This is a first order approximation, in time.



3.1.3 Finding new values of u using mid-point rule

Need to find u^{n+1} , from (*) value of the integral is same as for the initial time t=1,

$$\int_{x_{i-1}(t^{n+1})}^{x_i(t^{n+1})} u\, dx = \int_{x_{i-1}(1)}^{x_i(1)} u\, dx \ ,$$

where we know the value of $\int_{x_{i-1}(1)}^{x_i(1)} u\, dx$ from the self-similar solution when t=1,

using for example the mid-point rule. Then we get

$$\int_{x_{i-1}(t^{n+1})}^{x_{i+1}(t^{n+1})} u\, dx = \int_{x_{i-1}(1)}^{x_{i+1}(1)} u\, dx$$

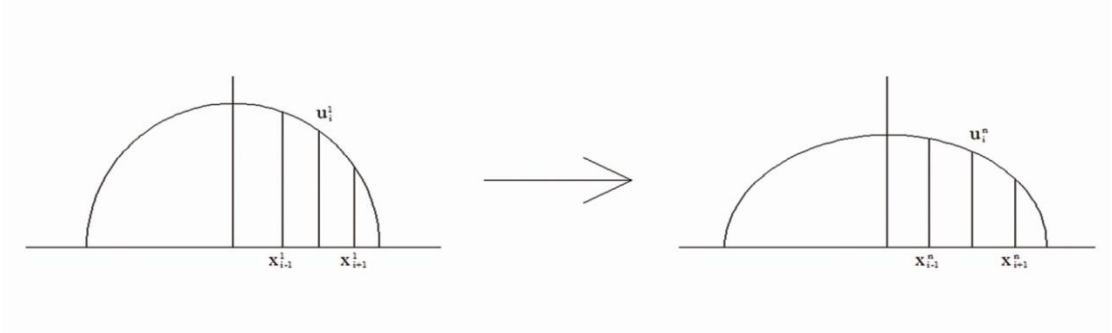
We also know area of $\int_{x_{i-1}(1)}^{x_{i+1}(1)} u\, dx$, so we get

$$[x_{i+1}(t^{n+1}) - x_{i-1}(t^{n+1})]u_i^{n+1} = \int_{x_{i-1}(1)}^{x_{i+1}(1)} u\, dx \ ,$$

leading to

$$u_i^{n+1} = \frac{[x_{i+1}(1) - x_{i-1}(1)]u_i}{[x_{i+1}(t^{n+1}) - x_{i-1}(t^{n+1})]}$$

which gives approximate values of u_i at next time step. This is a second order approximation in space for equal spacing.



3.2 Example of finding the velocities

Because the self similar solution is symmetric, therefore we start at the middle point where x=0, $\left.\frac{dx}{dt}\right|_0 = 0$, for each m.

if m=1, we have general formula:

$$\left.\frac{dx}{dt}\right|_i = -\frac{\partial u}{\partial x} \qquad \text{or} = -\frac{\partial}{\partial x}(u\ )$$

For i=0,

$$\left.\frac{dx}{dt}\right|_0 = 0$$

For i=1,2,……,N-1, using the central difference formula to generate $\left.\frac{dx}{dt}\right|_i$.

$$\left.\frac{dx}{dt}\right|_i = -\left(\frac{u_{i+1} - u_{i-1}}{x_{i+1} - x_{i-1}}\right)$$

for i=N, the last point

$$\left.\frac{dx}{dt}\right|_N = -\left(\frac{u_N - u_{N-1}}{x_N - x_{N-1}}\right)$$

if m=2, we have general formula:

$$\left.\frac{dx}{dt}\right|_i = -u\frac{\partial u}{\partial x} \qquad \text{or} = -\frac{1}{2}\frac{\partial}{\partial x}(u^2)$$

For i=0,

$$\left.\frac{dx}{dt}\right|_0 = 0$$

For i=1,2,……,N-1, using the central difference formula to generate $\left.\frac{dx}{dt}\right|_i$.

$$\left.\frac{dx}{dt}\right|_i = -\frac{1}{2}\left(\frac{u_{i+1}^2 - u_{i-1}^2}{x_{i+1} - x_{i-1}}\right)$$

For i=N, the last point

$$\left.\frac{dx}{dt}\right|_N = -\left(\frac{u_N - u_{N-1}}{2}\right)\left(\frac{u_N - u_{N-1}}{x_N - x_{N-1}}\right)$$

if m=3, we have general formula:

$$\left.\frac{dx}{dt}\right|_i = -u^2\frac{\partial u}{\partial x} \qquad \text{or} = -\frac{1}{3}\frac{\partial}{\partial x}(u^3)$$

For i=0,

$$\left.\frac{dx}{dt}\right|_0 = 0$$

For i=1,2,……,N-1, using the central difference formula to generate $\left.\frac{dx}{dt}\right|_i$.

$$\left.\frac{dx}{dt}\right|_i = -\frac{1}{3}\left(\frac{u_{i+1}^3 - u_{i-1}^3}{x_{i+1} - x_{i-1}}\right)$$

For i=N, the last point

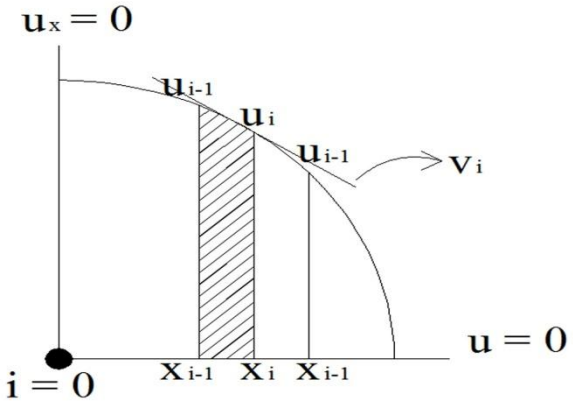
$$\left.\frac{dx}{dt}\right|_N = -\left(\frac{u_N - u_{N-1}}{2}\right)^2\left(\frac{u_N - u_{N-1}}{x_N - x_{N-1}}\right)$$

and so on.

3.3 Formula to find the velocities for general m

The formula for finding velocities for general m is

$$\frac{dx}{dt} = -u^{m-1}\frac{\partial u}{\partial x} \quad \text{or} \quad = -\frac{1}{m}\frac{\partial}{\partial x}(u^m)$$



For i=0,

$$\left.\frac{dx}{dt}\right|_0 = 0$$

For i=1,2,……,N-1, using the central difference formula to generate $\left.\frac{dx}{dt}\right|_i$.

$$\left.\frac{dx}{dt}\right|_i = -\frac{1}{m}\frac{\partial}{\partial x}(u^m)$$

$$\left.\frac{dx}{dt}\right|_i = -\frac{1}{m}\left(\frac{u_{i+1}^m - u_{i-1}^m}{x_{i+1} - x_{i-1}}\right)$$

For i=N, the last point

$$\left.\frac{dx}{dt}\right|_i = -u^{m-1}\frac{\partial u}{\partial x}$$

$$\left.\frac{dx}{dt}\right|_N = -\left(\frac{u_N - u_{N-1}}{2}\right)^{m-1}\left(\frac{u_N - u_{N-1}}{x_N - x_{N-1}}\right)$$

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

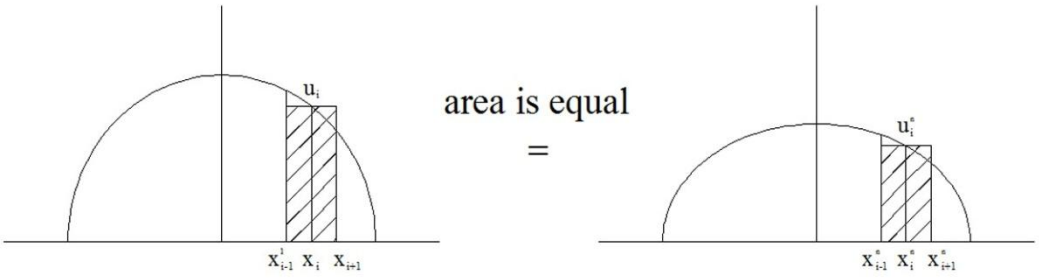
$$x_i^{n+1} = x_i^n + \Delta t\, \dot{x}_i$$

where $\dot{x}_i$ is the velocity at the point x_i, Δt is the time step and x_i^{n+1} is the position of x_i at the next time step.

Next, by applying a mid-point approximation to $\int u\, dx = \int u\ dx|_{t=1}$, the general formula to calculate u value in next time step is

$$u_i^{n+1} = \frac{[x_{i+1}(1) - x_{i-1}(1)]u_i}{[x_{i+1}(t^{n+1}) - x_{i-1}(t^{n+1})]}$$

where $[x_{i+1}(1) - x_{i-1}(1)]u_i$ is the area at time equal to 1. (i=1,2,……N-1) and $[x_{i+1}(t^{n+1}) - x_{i-1}(t^{n+1})]\, u_i^{n+1}$ is the area at the new time t^{n+1}.



This general formula keeps the area of solutions constant when the time changes.

How we apply this general formula in the computer program

For i=0, we have to use the trapezium rule to calculate the area because we cannot apply the mid-point rule in the first and the last intervals.

$$u_i^{n+1} = \frac{(x_{i+1} - x_i) * (u_i + u_{i+1})}{2} \frac{1}{(x_{i+1}^{n+1} - x_i^{n+1})}$$

For i=1,2,……,N-1, we can apply the mid-pt rule to the formula

$$u_i^{n+1} = \frac{[x_{i+1} - x_{i-1}]u_i}{(x_{i+1}^{n+1} - x_{i-1}^{n+1})}$$

For i=N, the last point u_N is always equal to zero

$$u_N^{n+1} = u_N^n = u_N^1 = 0$$

3.5 Error calculations between the exact solution and numerical solution

1. Calculate the error in boundary position

$$\text{Error in boundary} \quad = |x_N - x_N^{sss}|$$

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

2. Calculate the error in solution
   a. " l_1 error "

$$\sum_i |u_i - u_i^{sss}|$$

   b. " l_2 error "

$$\sqrt{\sum_i (u_i - u_i^{sss})^2}$$

   c. " l_∞ error "

$$\max_i |u_i - u_i^{sss}|$$

where u_i are the approximate solution values using the numerical method and u_i^{sss} 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

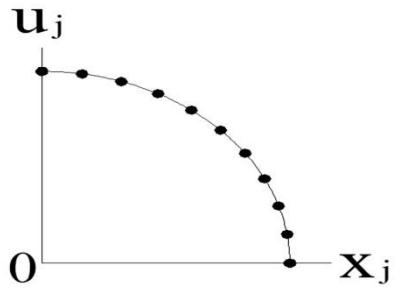
4.1 Error calculation with 10 points for time step equal to 0.01

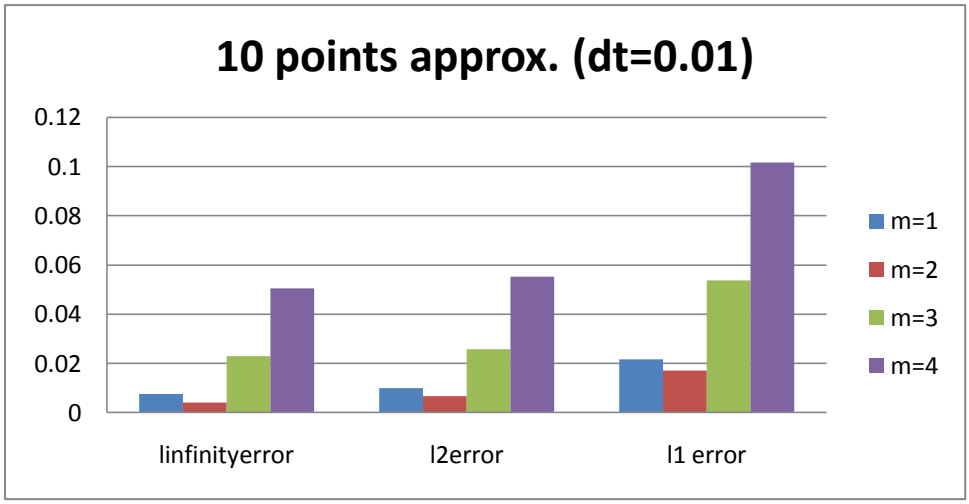
Starting with the time step equal to 0.01 and the number of points equal to 10, calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

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.



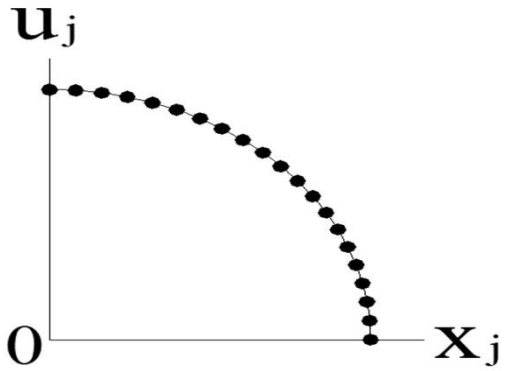


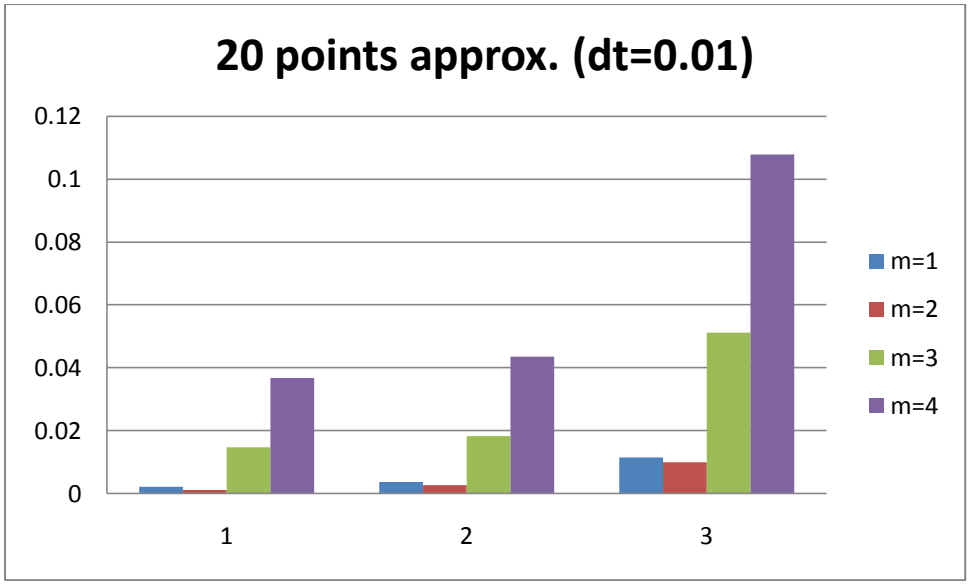
10 points approx. (dt=0.01)				
	linfinityerror	l2error	l1 error	boundary error
m=1	0.00760356	0.00995198	0.0216289	0.00720774
m=2	0.00403803	0.00673991	0.0171885	0.00495405
m=3	0.0229165	0.0258753	0.053699	0.0270282
m=4	0.0504221	0.055299	0.101573	0.047171

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.



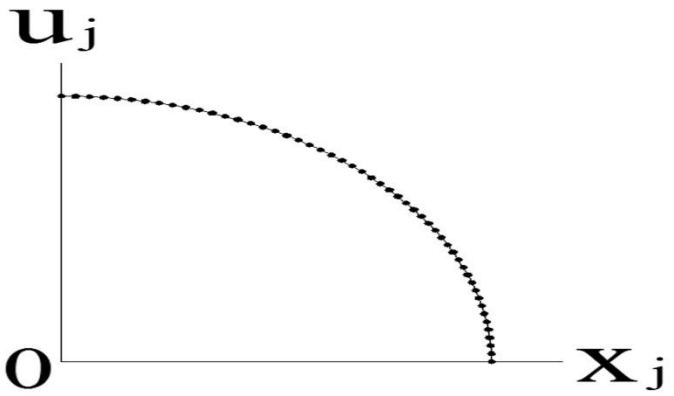


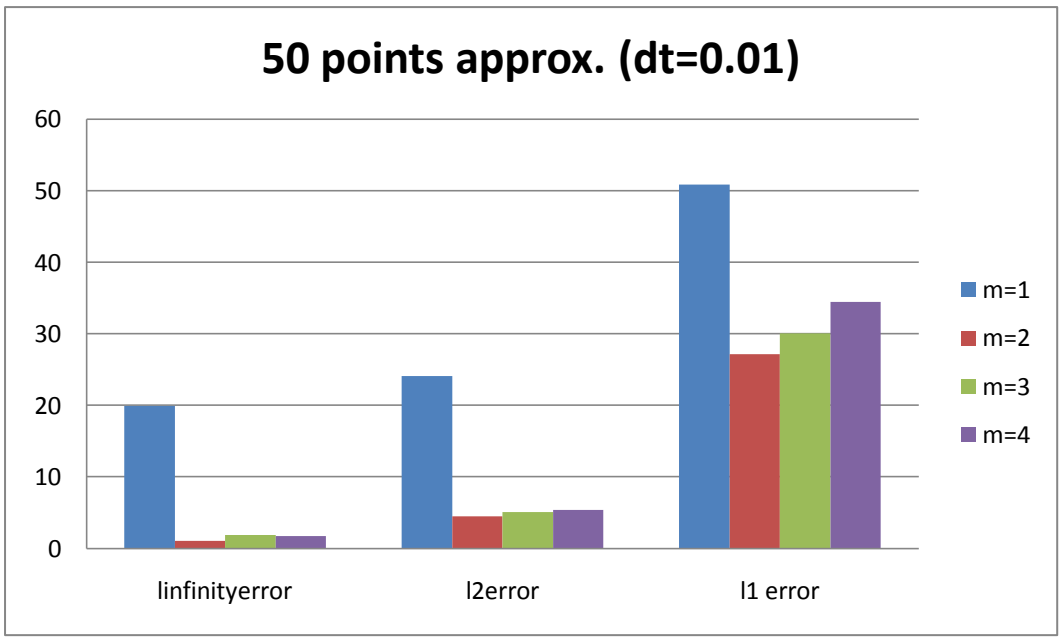
20 points approx. (dt=0.01)				
	linfinityerror	l2error	l1 error	boundary error
m=1	0.00209553	0.00364496	0.0115186	0.000793082
m=2	0.00119655	0.00270182	0.0099368	0.000706295
m=3	0.0146465	0.0183382	0.051108	0.0133553
m=4	0.0367602	0.0435501	0.107889	0.0254492

Error calculation with 50 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 50. 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.

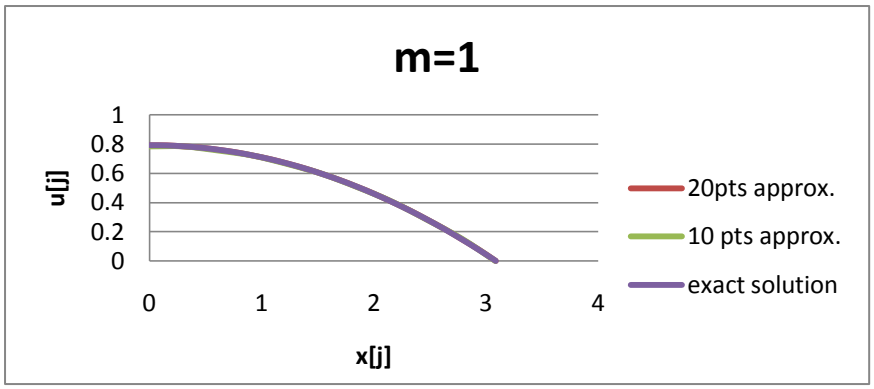


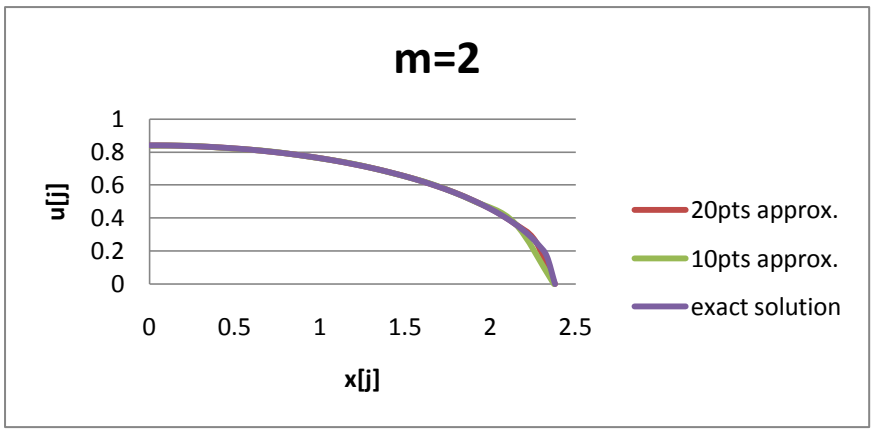


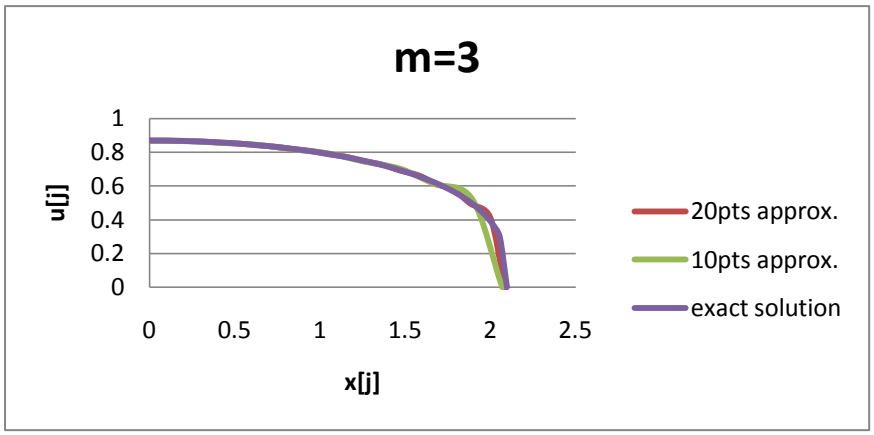
50 points approx. (dt=0.01)				
	linfinityerror	l2error	l1 error	boundary error
m=1	19.9158	24.0537	50.8696	0.0288201
m=2	1.05169	4.48732	27.1773	0.0442264
m=3	1.89115	5.06183	30.0269	0.0574284
m=4	1.71073	5.41063	34.4655	0.0810229

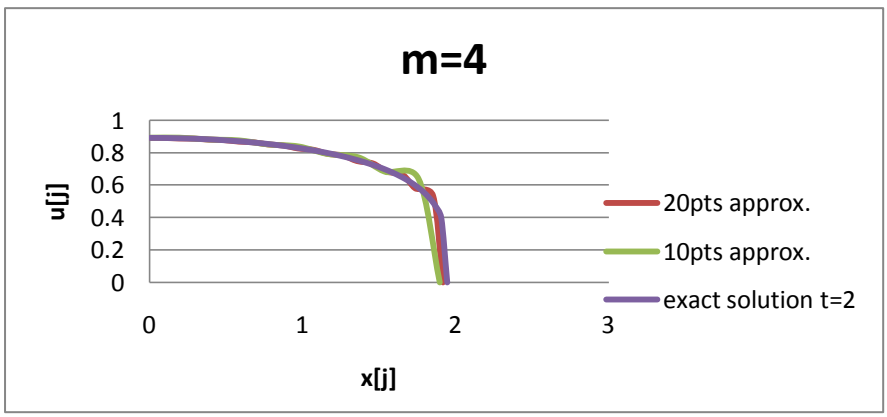
Approximate solutions for different points of approximation for time step 0.01

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.







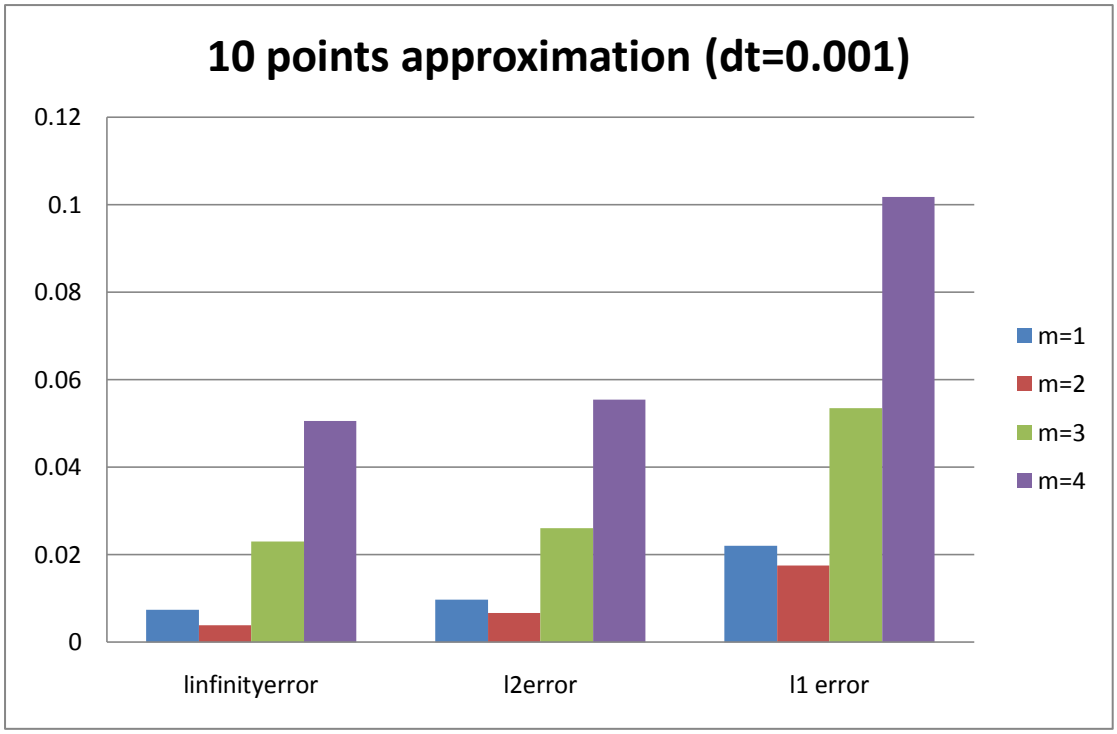


4.2 Error calculation with 10 points for time step equal to 0.001

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.

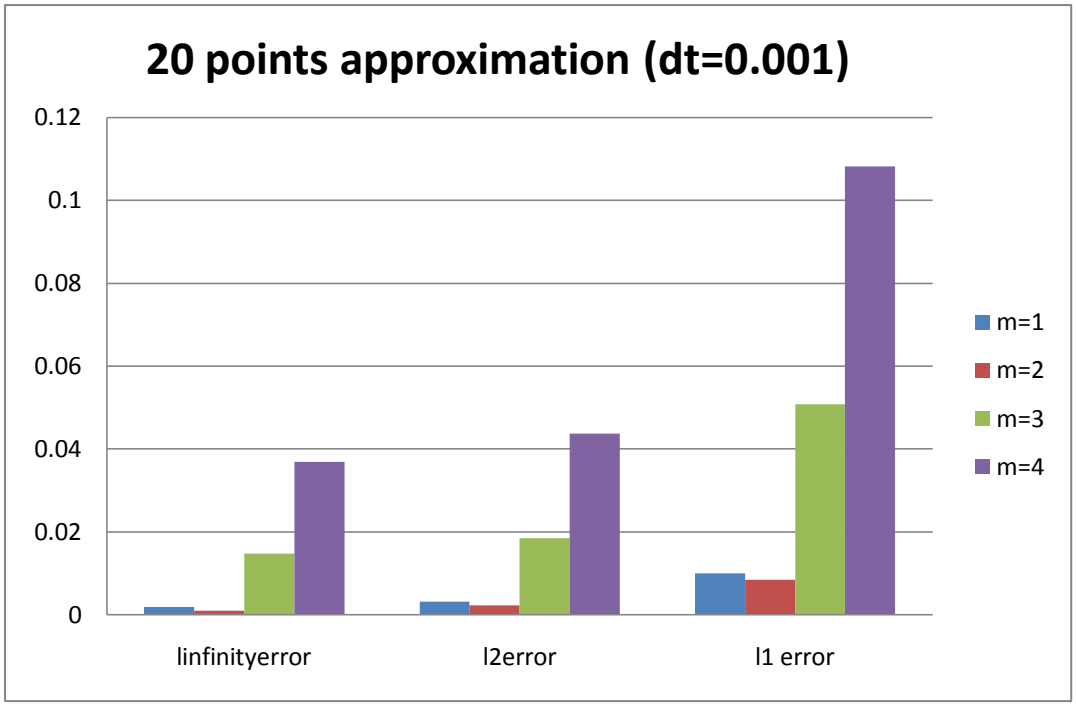


10 step approximation (dt=0.001)				
	linfinityerror	l2error	l1 error	boundary error
m1	0.00743814	0.009751	0.022011	0.00829078
m2	0.00386414	0.00659	0.017518	0.00565947
m3	0.0230331	0.025979	0.053528	0.0275071
m4	0.0505132	0.055411	0.101788	0.0474908

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.

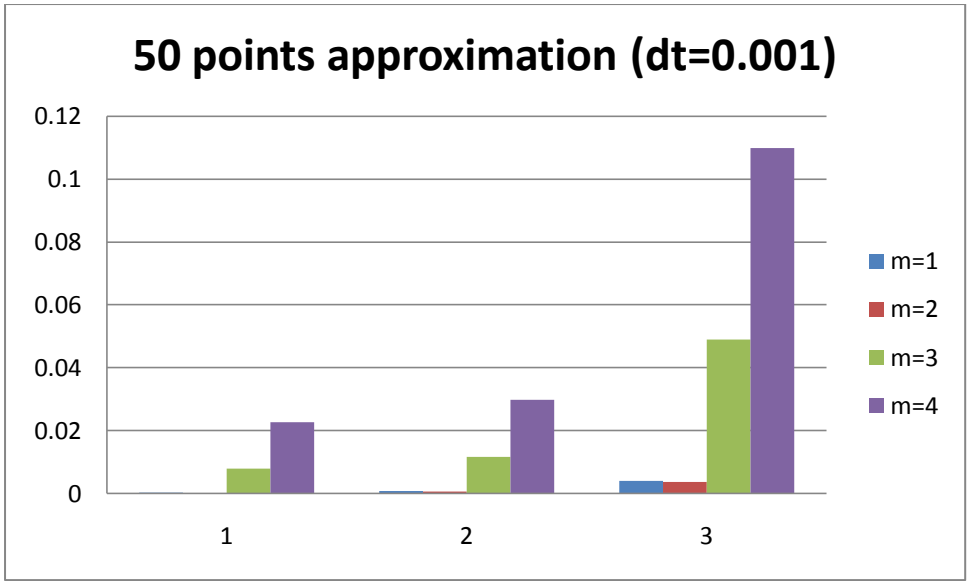


20 points approximation (dt=0.001)				
	linfinityerror	l2error	l1 error	boundary error
m1	0.00188328	0.003178	0.009989	0.00187345
m2	0.00099176	0.00228	0.008471	0.00141046
m3	0.0147765	0.018468	0.050825	0.0138743
m4	0.036926	0.043755	0.108232	0.0258484

Error calculation with 50 points for time step equal to 0.001

Now we keep the time steps equal to 0.001 and change the number of points to 50. Then calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

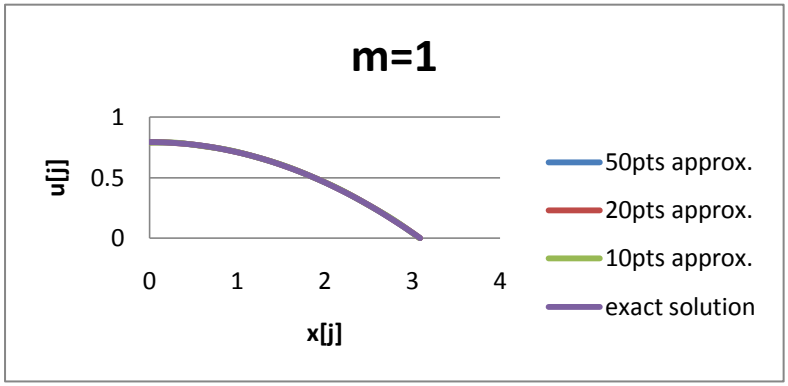
The time steps are equal to 0.001, so we need to run 1000 steps to reach time equal to 2.

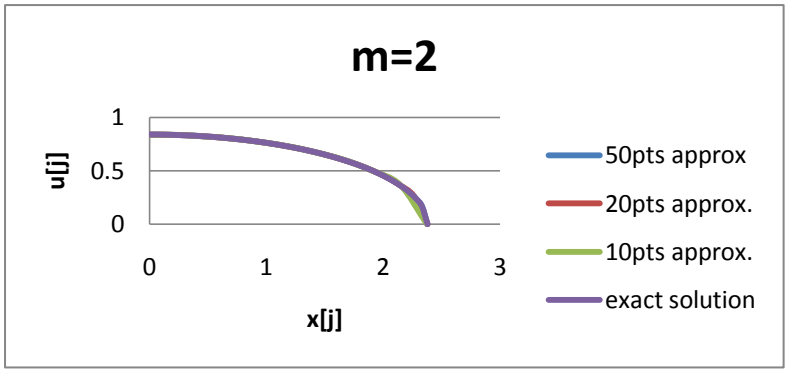


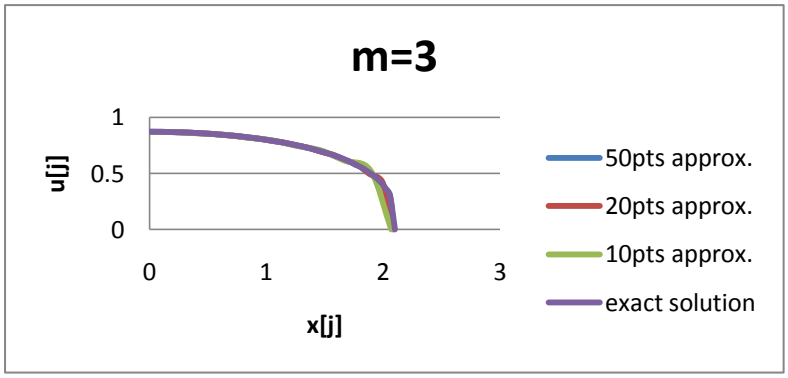
50 points approximation (dt=0.001)				
	l infinity error	l2error	l1 error	boundary error
m1	0.00032574	0.000837	0.004013	0.000192799
m2	0.00018124	0.000632	0.00358	0.000162486
m3	0.00788961	0.011627	0.048987	0.00544837
m4	0.0227005	0.029784	0.109927	0.0106087

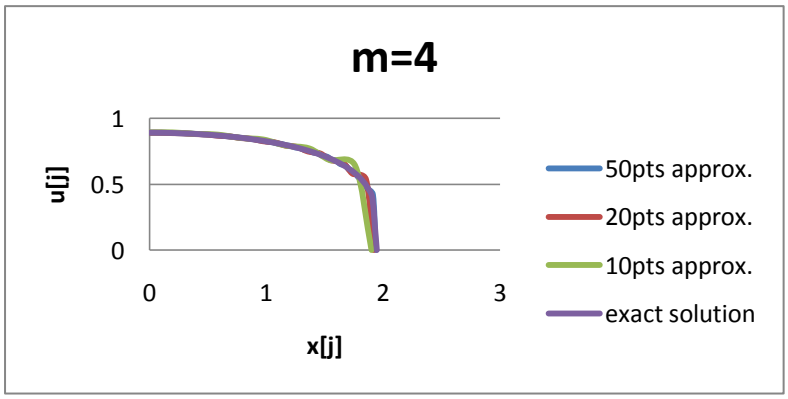
Approximate solutions for different points of approximation for time step 0.001

The diagram below shows the solutions for different points of approximation when time equal to 2 with the time step equal to 0.001 compared with the exact solution when time equal to 2.







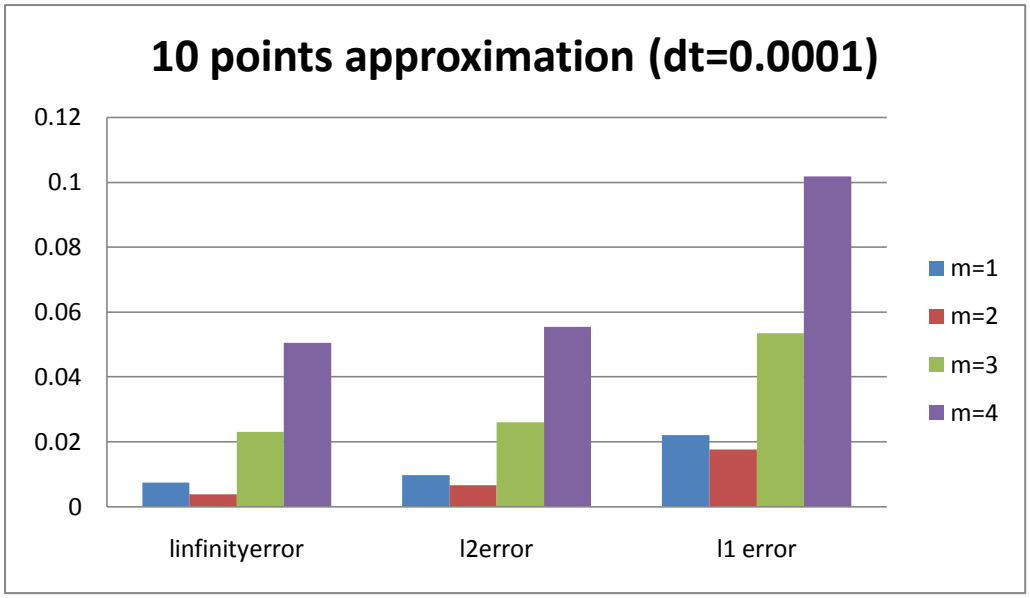


4.3 Error calculation with 10 points for time step equal to 0.0001

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.

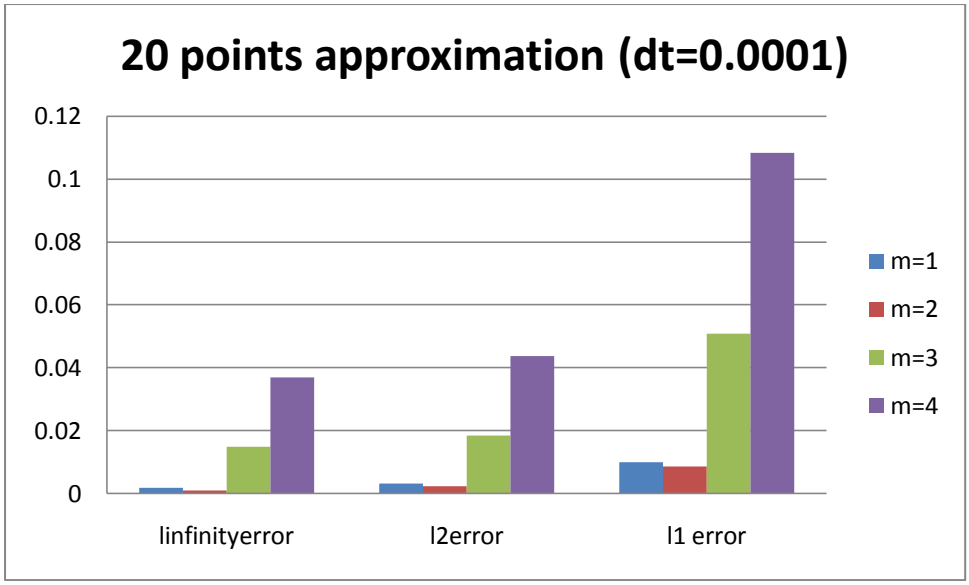


10 points approx.				
	linfinityerror	l2error	l1 error	boundary error
m=1	0.00742051	0.00973175	0.0220496	0.00839903
m=2	0.00384594	0.00657707	0.017551	0.00573011
m=3	0.0230448	0.02599	0.0535083	0.0275551
m=4	0.0505223	0.0554225	0.101808	0.0475228

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.

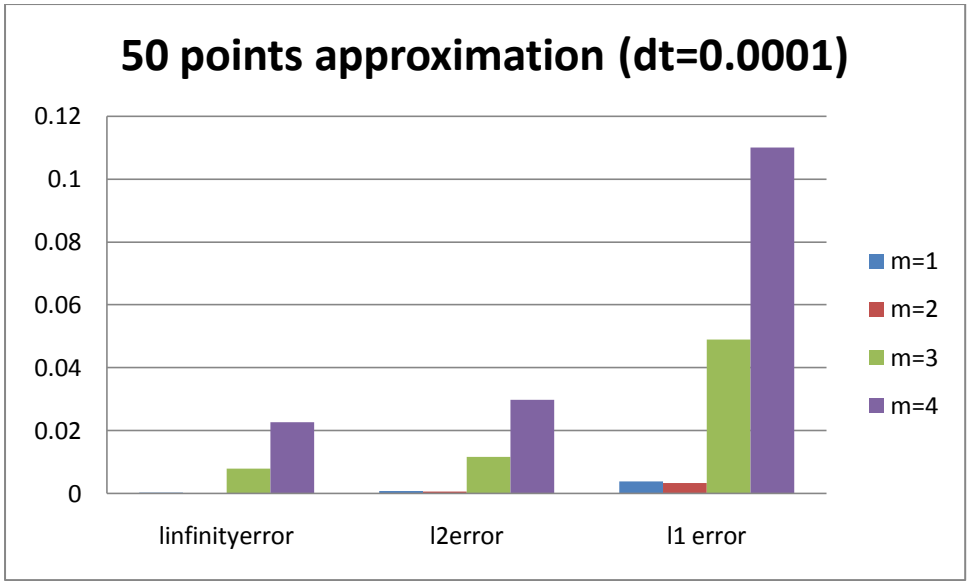


20 points approximation (dt=0.0001)				
	l infinity error	l2error	l1 error	boundary error
m=1	0.00186101	0.003139	0.0100273	0.00198134
m=2	0.000970518	0.002251	0.00850487	0.0014809
m=3	0.0147896	0.018484	0.0507885	0.0139262
m=4	0.0369427	0.043777	0.108301	0.0258883

Error calculation with 50 points for time step equal to 0.0001

Now we keep the time steps equal to 0.0001 and change the number of points to 50. Then calculate the error between the solution using numerical method and the exact solution when time is equal to 2.

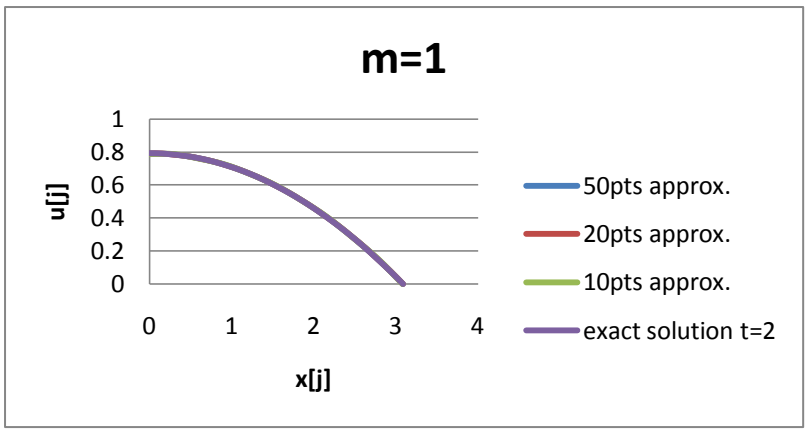
The time steps are equal to 0.0001, so we need to run 10000 steps to reach time equal to 2.

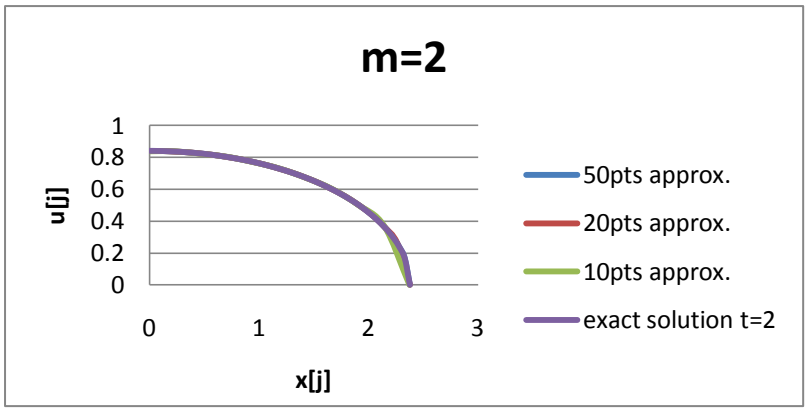


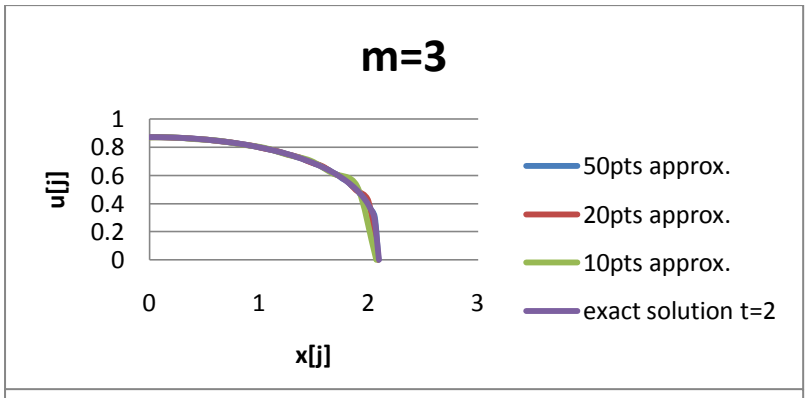
50 points approximation (dt=0.0001)				
	l infinity error	l2error	l1 error	boundary error
m=1	0.000300427	0.000755125	0.00377834	0.000300016
m=2	0.00015806	0.000561686	0.00333322	0.000232391
m=3	0.00789915	0.0116413	0.049029	0.00550081
m=4	0.0227141	0.0298064	0.110076	0.0106506

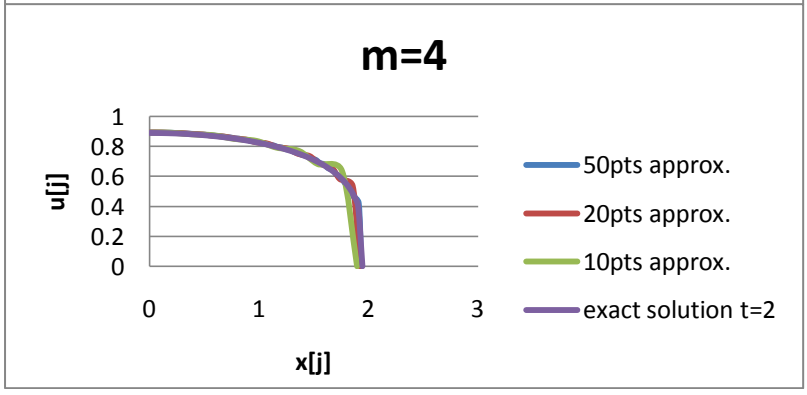
Approximate solutions for different points of approximation when dt=0.0001

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.









4.4 Discussions on the results

At the beginning, we start the numerical approximation with values from the exact solution when time equal to 1. With different time steps approach, if the time equal to 2, it stops.

In section 4.1, the time steps are equal to 0.01, which means the equation needs to run for 100 times to reach time equal to 2. Starting with 10 points we carried out the approximation for m equal 1 to 4. The result showed that when m equal to 1, all the errors are really small and the errors are greater than when m equal to 2, but when m equal to 3 and 4, all the errors are increase at a faster rate compared with m equal to 1 and 2 but all the errors are still really small. Then we keep the same time steps and change the number of points to 20. Then we can compare the results with 10 points and 20 points approximation. As we saw from the results, we can say that all the errors are smaller with more number of points. Also with smaller m, the ratio of the errors decrease more rapidly. Next, we keep the same time steps and increase the number of points to 50. Then we have an unstable result, because we have a large time step with a large number of points and the Euler method goes unstable. Therefore in the next section, we decreased the time steps to 0.001. Then we get a smaller time steps, so stability of the solutions ensured.

In section 4.2, the time steps are equal to 0.001, which means the equation needs to run for 1000 times to reach time equal to 2. We compared the 10 points, 20 points and 50 points approximations with m equal 1 to 4. All the errors keep decreasing when the number of points increased. Also if m is smaller, the decreasing ratio is faster. The errors different from the exact solution are keeping really small in comparison with the errors with the time steps equal to 0.01. The errors are similar,

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 equation needs to run for 10000 times to reach time equal to 2. We compare the results with the time steps equal to 0.001. The results are similar and remain really small.

Conclusion with these three examples, we can say if we have more points then the solution will be more accurate provided that the solution is stable. And also if we increased the number of time steps, it will ensure the stability of the solutions.

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 methods, the meaning of scale invariance and similarity, and the advantages of using moving mesh Numerical methods on scale invariant solutions.

In chapter 2, I showed how to scale the variables to keep the equation invariant and find the similarity transformation for the blow-up equation and Porous medium equation. Also, I found the general form of self-similar solutions for blow-up equation and PME. For the blow-up equation, we cannot solve the SSS exactly. Therefore we applied two numerical schemes to solve the equation. In the project, I applied the explicit Euler scheme and the Runge-Kutta 4 scheme to solve the equation by a computer program written in C++. And using the shooting method to get the approximate solution. On the other hand, for the PME, we solved the SSS exactly. Therefore we have a particular solution for sss. Moreover, we found out the self-similar solution for a more general form of the PME. At the end of the chapter 2, I plotted the exact solution of more general form of PDE with different m values when time varies. And I discussed the diagram.

In chapter 3, I described a numerical method for the Porous medium equation. I generated the velocities for moving the points using a conservation principle. At the end of chapter 3, I pointed out the method of error calculation and reasons for doing the error calculation by which the accuracy of the solution using numerical method was investigated.

.

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

$$\text{error}_N = CN^p$$

$$\text{error}_{2N} = C(2N)^p$$

where N is the number of points, C is constant, p is the order of accuracy

$$\frac{\text{error}_N}{\text{error}_{2N}} = \frac{CN^p}{C(2N)^p}$$

$$\frac{\text{error}_N}{\text{error}_{2N}} = \frac{1}{2^p}$$

$$2^p = \frac{\text{error}_{2N}}{\text{error}_N}$$

$$p = \log\left(\frac{\text{error}_{2N}}{\text{error}_N}\right)\frac{1}{\log 2}$$

For example, get data from section 4.3. For m=1,

l_1 error for 10 points =0.0220496

l_1 error for 20 points =0.0100273

Therefore,

$$p = \log\left(\frac{0.0100273}{0.0220496}\right)\frac{1}{\log 2}$$

$$p = -1.13682$$

we need to keep on compare l_1 error for 40 points with l_1 error for 20 points.

Getting a new value of p, and so on. Until the order of accuracy p converges.

6. Appendix

Theorem:

Given that $u(t,x)$ is a positive solution of a scale-invariant mass-conserving problem governed by the PDE

$$u_t = \mathcal{L}u \qquad (1)$$

in the interior of the interval $\big(a(t), b(t)\big)$, if

1. $u(t,x)$ satisfies the local conservation principle

$$\int_{a(t)}^{\hat{x}(t)} u(t,x)dx = \text{ constant in time} \qquad (2)$$

for all t and all $\hat{x}(t) \in (a(t), b(t))$,

2. at the endpoints either

- $u = 0$ or
- υ is the similarity velocity,

for all t,

3. the initial condition $u(t_0, x)$ coincides with a self-similar scaling solution at time $t = t_0$,

then

- $u(t,x)$ remains self-similar for all t and all x
- $\upsilon(t,x)$ is the similarity velocity $\frac{\beta x}{t}$ for all t and all x

In other words, initial data which coincides with a self-similar solution at time $t = t_0$ is propagated as a self-similar solution with the self-similar velocity for all time.

7. References

[1] C.J.Budd, W.Huang and R.D Russell, "Moving mesh methods for problems with blow-up" (1996) p.305-327.

[2] C.J Budd, G J Collins, "Symmetry based numerical methods for partial differential equations", Numerical analysis (1997), Addison Wesley Longman Limited(1998) p.16-33

[3] C.J Budd, M.D.Piggott, "Geometric Integration and Its Applications", University of Bath p.1-9, 74-76

[4] G.Bluman and S. Kumei, "Symmetries and differential equations", (1989) Springer, New York

[5]R E Pattle, Quarterly Journal of Mechanics and Applied Mathematics, p.407-409, (1959)

[6] G.I.Barenblatt, Prik. Mat. I Mekh, p.67-78, (1952)

[7] M.J.Baines, Private Communication

[8] M.J.Baines, M.E.Hubbard, P.K.Jimack. A Moving Mesh Finite Element Algorithm for the Adaptive Solution of Time-Dependent Partial Differential Equations with Moving Boundaries, Applied Numerical Mathematics, (2005), p.450-469