Iranaian J. Numerical Analysis and Optimization
Vol. 3, No. 1, (2013), 13-30

A level set moving mesh method in static form for one dimensional PDEs

Maryam Arab Ameri

Abstract

In this paper, we propose an adaptive mesh approach for time dependent parial differential equations, based on a so-called moving mesh PDE(MMPDE) and level set method. It means that the velocity of mesh nodes is calculated by MMPDE and is employed as veocity in the level set equation. Then, at each time level, the mesh points are considered as the level contours of the level set function. Finally the method is merged with local time step technique.

Keywords: Adaptive grid; Level set function; Level contours; Moving mesh; Local time stepping refinement; MMPDE.

1 Introduction

In this paper, we discuss a class of adaptive mesh algorithms for solving time-dependent partial differential equations(PDEs) whose solutions have large variations over a given physical domain, such as shock waves, boundary layer and interior layer.

This method is based on the level set concept. the level set methods are powerful numerical techniques for tracking the evolution of interfaces moving in complex ways. The level set methods were used for the first time to represent the evolution of surfaces implicity by Osher and Sethian [11]. Due to its many advantages, this approach has been used for many cases [1, 5, 9, 13, 19], for example, it has been used for mesh generation around a body(inside or outside) by the level set function [12]. Liao et al. presented some points about using the level set functions for moving grid based on deformation method [6]. In this paper, we also formulate an adaptive mesh method which is based on the level set method. This method is combined with a class of moving mesh methods which employs a moving mesh partial differential equation, MMPDE, to perform mesh adaptation. The MMPDE is formulated in terms of coordinate transformation or mapping,[3, 4, 8, 16, 17]. Several moving mesh equations(MMPDEs) based upon equidistribution principle have been

Maryam Arab Ameri
Department of Mathematics, University of Sistan and Baluchestan, Zahedan, Iran. e-mail: arabameri@math.usb.ac.ir

derived in [3], where in Section 2, we briefly review some of them. Finally after introducing the new method, we present local time stepping refinement technique based on the slope of the level set function for increasing the efficiency of the moving mesh method. The mentioned method is used to solve scalar field satisfying

$$u_t(x,t) = L(u), \tag{1}$$

where L is a differential operator defined on physical domain Ω, $a \le x \le b$ and $0 \le t \le T_f$.

2 Moving Mesh Methods

In this section, we develop a new moving mesh method based on the level set concept. We start with a review of the equidistribution principle(EP)[3, 15] and derive MMPDEs from that principle in subsection 2.1,then a description of adaptive grid based on level set method is given in the subsection 2.2.

2.1 MMPDE

The evolution of moving computational grid can be viewed as a discretization of a one-to-one time dependent coordinate mapping. Let x and ξ denote the physical and computational coordinates, with domains Ω and Ω_c, respectively. Without loss of generality, both of them are assumed to be in $[0,1]$. Define a coordinate transformation by:

$$x = x(\xi,t),\ \ \xi \in [0,1],\ \ x(0,t) = 0,\quad x(1,t) = 1. \tag{2}$$

The computational coordinates is discretized on a uniform mesh given by

$$\xi_j = \frac{j}{N},\quad j = 0,1,\cdots,N,$$

where N is a certain positive integer and the corresponding mesh in x denoted by

$$0 = x_0 < x_1(t) < x_2(t) < \cdots < x_{N-1}(t) < x_N = 1.$$

A major factor of moving mesh approach is the monitor function, $\rho(x,t)$, which is chosen to be somehow a measure of the solution error. For a given monitor function, the mesh point locations, $x_j(t)$, could be required to satisfy the following equidistribution principle (EP) for all the values of time t [3]:

$$\int_{x_{j-1}(t)}^{x_j(t)} \rho(x,t)dx = \frac{1}{N}\int_0^1 \rho(x,t)dx = \frac{1}{N}\theta(t),$$

or equivalently

$$\int_0^{x_j(t)} \rho(x,t)dx = \frac{j}{N}\theta(t) = \xi_j\theta(t). \tag{3}$$

By differentiating (3) with respect to ξ once and twice,, we obtain two differential forms of EP,

$$\rho(x(\xi,t))\frac{\partial}{\partial\xi}x(\xi,t) = \theta(t), \tag{4}$$

$$\frac{\partial}{\partial\xi}(\rho(x(\xi,t),t)\frac{\partial}{\partial\xi}x(\xi,t)) = 0. \tag{5}$$

These EPs,(3),(4) and (5), which do not contain the node speed $\dot{x}(\xi,t)$,are called quasi-static EPs(QSEPs). Related to these QSEPs, various MMPDEs were derived in [3] by taking the mesh to satisfy the above EP (Eq. (5)) at a later time $t+\tau$ instead of at t. In this case the mesh should satisfy

$$\frac{\partial}{\partial\xi}\rho(x(\xi,t+\tau),t+\tau)\frac{\partial}{\partial\xi}x(\xi,t+\tau) = 0 \tag{6}$$

where the parameter τ is called a relaxation time. By expanding the term $\frac{\partial}{\partial\xi}x(\xi,t+\tau)$ in Taylor series and dropping certain higher order terms, various MMPDEs can be obtained. Two of them which are used in our work are MMPDE5 and MMPDE6:

$$MMPDE5: \quad -\dot{x} = -\frac{1}{\tau}\frac{\partial}{\partial\xi}(\rho\frac{\partial x}{\partial\xi}), \tag{7}$$

$$MMPDE6: \quad \frac{\partial^2\dot{x}}{\partial\xi^2} = -\frac{1}{\tau}\frac{\partial}{\partial\xi}(\rho\frac{\partial x}{\partial\xi}). \tag{8}$$

These MMPDEs and also the rest of them not only force the mesh ,$x(\xi,t)$, toward equidistribution principle but also prevent the mesh from crossing. More specifically, the term $-(\frac{1}{\tau})(\frac{\partial}{\partial\xi}(\rho\frac{\partial \mathrm{x}}{\partial\xi}))$ plays the fundamental role of a correction term to make the mesh equidistribute the monitor function and it is stabilizing term for the mesh trajectories.

The monitor function, ρ, is chosen such that the mesh points are concentrated in regions where more accuracy is needed, and so ρ is usually taken to be some measure of the solution error estimated from discrete solution values. A commonly used monitor function is $\rho = \sqrt{1+\alpha u_x^2}$ (α is regularizing factor), which equidistributes the arclength of the solution u.

In this work, we discretize MMPDE5 and MMPDE6 by centered finite difference in space, which yields for MMPDE5

$$\dot{x}_j = \frac{E_j}{\tau}, \quad j = 0,1,\ldots,N$$

and for MMPDE6

$$\dot{x}_{j+1} - 2\dot{x}_j + \dot{x}_{j-1} = -\frac{E_j}{\tau}, \quad j = 0, 1, \ldots, N.$$

The quantity E_j represents a centered approximation to the term on the right hand side of MMPDE5 and MMPDE6 given by

$$E_j = M_{j+1/2}(x_{j+1} - x_j) - M_{j-1/2}(x_j - x_{j-1}), \quad j = 0, 1, \ldots, N$$

where $M_{j+1/2} = \frac{1}{2}(M_j + M_{j+1}), M_j = M(u_j)$, and $u_j \approx u(x_j, t)$ is an approximation of the solution at grid point x_j.

2.2 Moving Mesh Based on Level Set Approach

In this section, at first a new moving grid method is formulated, and then the algorithm of this method is presented.
Essential point in all applications of the level set method is using the implicit representation. This point is also used in grid generation in such a way that the mesh nodes are represented implicitly by the level contours of the level set function. In this method, at each time level we construct the level set function,$\psi(x, t)$, and the mesh points $x_j, j = 0, 1, \ldots, N$, are obtained by the level contours of $\psi(x, t)$, that means the mesh points are:

$$\{x_j \mid \psi(x, t) = c_j,\ j = 1, 2, \ldots, N\} \tag{9}$$

where c_j is j-th component of a constant vector $C = (0, \frac{1}{N}, \frac{2}{N}, \ldots, 1)$ and N is the number of mesh points.
In fact, in each time, we seek a level set function, $\xi = \psi(x, t) : \Omega \to \Omega_c$, which maps x_j to $c_j = \frac{j}{N},\ j = 0, 1, \ldots, N$ and satisfies the well-known equidistribution principle

$$J\rho = \frac{\sigma}{|\Omega_c|}, \tag{10}$$

where J is the Jacobian of the coordinate transformation $x = \psi^{-1}(\xi, t) : \Omega_c \to \Omega$, and $\rho = \rho(x, t)$ is a given monitor function, and $\sigma = \sigma(t) = \int_\Omega \rho(x, t)dx$.
For updating the level set function or equivalently for updating the position of mesh nodes, the well-known level set equation is applied,

$$\psi_t(x, t) + v\psi_x(x, t) = 0. \tag{11}$$

The ideal initial condition for the above equation is $\psi(x, t) = x$, because in the most mesh adaptation algorithms, the purpose is to convert a uniform mesh at initial time to an equidistributed mesh at the next time levels. The mentioned initial condition gives a uniform mesh at $t = 0$. Also the following boundary conditions are considered for the level set equation,

$$\psi(0,t)=0, \psi(1,t)=1.$$

As we mentioned at each time level, any

$$\psi(x,t)=a \tag{12}$$

has a correspondence point on x-axis which is one of the new mesh nodes. By differentiating (12) with respect to t, we get

$$\psi_t(x,t)+\psi_x(x,t)\dot{x}=0, \tag{13}$$

and when compared with (11) and (13), we get

$$\dot{x}=v, \tag{14}$$

that means, the nodes velocity is equal to v.
So for calculating the nodes velocity in the level set equation, the MMPDE5 or MMPDE6 should be applied which generates an equidistributed mesh through this algorithm.
At each time level, after determining new nodes, the solution of PDE should be determined. For this purpose, let

$$\tilde{U}(\psi(x,t),t)=u(x,t), \tag{15}$$

then

$$\tilde{U}_t=u_x\dot{x}+u_t$$

where $u_t=L(u)$ by (1) and $\dot{x}=v$ by (14). The derivatives in $L(u)$, such as u_x, u_{xx} are also transformed. For example, from (15), we have

$$\tilde{U}_\psi=u_x x_\psi \Rightarrow u_x=\frac{\tilde{U}_\psi}{x_\psi}=\frac{\tilde{U}_\psi}{\frac{\sigma}{\rho|\Omega_c|}}=\frac{\rho|\Omega_c|}{\sigma}\tilde{U}_\psi,$$

$$\tilde{U}_{\psi\psi}=u_{xx}(x_\psi)^2+u_x x_{\psi\psi} \Rightarrow u_{xx}=\frac{\tilde{U}_{\psi\psi}-u_x x_{\psi\psi}}{(x_\psi)^2}=(\frac{\rho|\Omega_c|}{\sigma})^2(\tilde{U}_{\psi\psi}-u_x x_{\psi\psi}).$$

The higher derivatives can be derived similarly. The transformed equation for $\tilde{U}(\psi,t)$ takes the form of

$$\tilde{U}_t=\tilde{L}(\tilde{U}), \tag{16}$$

where $\tilde{L}$ is a differential operator in ψ. Finally (16) will be solved on a uniform grid.

2.3 Algorithm of Adaptive Level Set Method(ALSM)

In this section, we provide an algorithm to solve the given PDE (1) in a moving grid based on the level set function.
Step1: Enter the initial time, t_0, the final time, T_f, the length of time step, Δt, and the number of mesh points, N.
Step2: Set $i = 0$.
Step3: If $t_0 = 0$, set

$$\begin{aligned} X^{(t_0)} &= \{x_j^0 = \frac{j}{N}, j = 0, 1, \ldots, N\}, \\ \psi^{(t_0)} &= X^{(t_0)}, \\ U^{(t_0)} &= u^{(t_0)} = u(X^{(t_0)}, t_0), \\ \tilde{U}^{(t_0)} &= U^{(t_0)} \end{aligned}$$

Step4: Determine $\rho(u(x, t_i))$ by the solution u being calculated.
Step5: Compute mesh velocity, $v = \dot{x}$, either by MMPDE5:

$$\dot{x}_j = \frac{E_j}{\tau}, \quad j = 0, 1, \ldots, N$$

or by MMPDE6:

$$\dot{x}_{j+1} - 2\dot{x}_j + \dot{x}_{j-1} = -\frac{E_j}{\tau}. \quad j = 0, 1, \ldots, N$$

Step6: Update the level set function by the following equation:

$$\psi_t(x, t) + \psi_x(x, t).v = 0,$$

which by forward finite difference discretization in time converts to:

$$\psi(t_{i+1}) = \psi(t_i) + \Delta t(v\psi_x)_{t_i},$$

with the boundary conditions $\psi_0(t_i + 1) = 0$ and $\psi_N(t_i + 1) = 1$.
Step7: Define the inverse of $\psi(x, t)$ for updating the new nodes, $X(t_{i+1})$, in current time.
Step8: Determine the values of u on the current time level by the described procedure in previous subsection and solve (16) or

$$\tilde{U}(t_{i+1}) = \tilde{U}(t_i) + \Delta t.\tilde{L}(\tilde{U}(t_i)),$$

Step9: Set $i = i + 1$.
Step10: If $t_i \leq T_f$ go to Step3, else break.

3 Local Time Step Refinement

Local time stepping for one-dimensional conservation laws was first proposed by Osher and Sanders [10]. Tan et. al. proposed variable time stepping for one and two dimension for conservation laws, which uses semi-dynamic moving mesh method such that the CFL condition is used to obtain time interval refinement to compute the solution [18]. Also, Soheili and Salahshour used this approach for the blow-up problems [14].
In this section, we also present the details of the local time stepping refinement(LTSR) which is performed based on the level set function.
Let initial time steps of the problem have the form

$$t_0 = 0 \to t_1 = t_0 + \Delta t \to \ldots \to t_n = t_0 + n\Delta t \to t_{n+1} = t_0 + (n+1)\Delta t \to \ldots \to T_f,$$

where Δt is a specified value for the time step and is constant through using this procedure. On the first interval $[t_0, t_1]$, set $\Delta t_0 = \frac{t_1 - t_0}{k_0}$ where $k_0 \in N$ is constant and depends on the slope of the level set function, (the method of determining k_0 will be described). We have

$$t_{0+(k_0-i)\Delta t_0} = t_0 + (k_0 - i)\Delta t_0,\ i = k_0, k_0 - 1, \ldots, 0$$

so the time integration on $[t_0, t_1]$ involves k_0 sub-steps such that

$$t_0 = 0 \to t_0 + \Delta t_0 \to \ldots \to t_0 + k_0\Delta t_0 = t_1.$$

Generally, suppose that we are at time level $t = t_n$ and we want to move towards $t = t_{n+1}$. Similarly consider $\Delta t_n = \frac{t_{n+1} - t_n}{k_n}$ such that

$$t_{n+(k_n-i)\Delta t_n} = t_n + (k_n - i)\Delta t_n,\ i = k_n, k_n - 1, \ldots, 0$$

that means, on interval $[t_n, t_{n+1}]$, we have

$$t_n \to t_n + \Delta t_n \to \ldots \to t_n + k_n\Delta t_n = t_{n+1}.$$

This process continues up to T_f. Now, $k_0, k_1, \ldots, k_n, \ldots$ are determined by the following process. At the first step, we start integrate the problem from t_0 to $t_1 = t_0 + \Delta t$ without any LTSR, we call this process prediction step, then we determine the level set function, the new mesh nodes of adaptive mesh and solution of PDE at t_1. According to the property of the mentioned adaptive mesh method in previous section, "for larger slope of the level set function, more mesh nodes are concentrated". Thus according to slope of the level set function, we define the interval $[t_0, t_1]$. So the slope of $\psi_j^{(1)}$ on $[x_j, x_{j+1}],\ j = 0, 1, \ldots, N$ is calculated, then for having more efficient solution, $[t_0, t_1]$ is subdivided to k_0 parts. Judicious choice for k_0 can be

$$k_0 = \min\{\max_j[slope(\psi_j^{(1)}), 10]\}, \quad j = 0, 1, \ldots, N,$$

and the time integration is done on this sub-interval again but with $\Delta t_0 = \frac{t_1 - t_0}{k_0}$. After obtaining the solution at t_1 we act on $[t_1, t_2]$ similarly and determine the solution at t_2.
Generally, after calculating the solution at t_n, at first we have a prediction process concluding time integration from t_n to t_{n+1} with pre-determined value of Δt as time step for knowing the slope of $\psi_j^{(n+1)}$ on $[x_j, x_{j+1}], j = 0, 1, \ldots, N$ and subsequently $k_n = \min\{\max_j[slope(\psi_j^{(n+1)})], 10\}$. Then the time integration is performed on this sub-interval k_n times with local time step Δt_n.

3.1 Algorithm of Adaptive Level Set Method with LTSR

Step1: Enter the initial time, t_0, the final time, T_f, the length of time step, Δt, and the number of mesh points, N.
Step2: Set $i = 0$.
Step3: If $t_0 = 0$, set

$$X^{(t_0)} = \{x_j^0 = \frac{j}{N}, j = 0, 1, \ldots, N\},$$

$$\psi^{(t_0)} = X^{(t_0)},$$

$$\tilde{U}^{(t_0)} = U^{(t_0)} = u^{(t_0)} = u(X^{(t_0)}, t_0).$$

Step4: Run ALSM one time with $t_0 = t_i, T_f = t_{i+1}$ and Δt, then calculate $\psi^{(t_{i+1})}, X^{(t_{i+1})}$ and $\tilde{U}^{(t_{i+1})}$.
Step5: Calculate the slope of $\psi^{(i+1)} = \psi^{(t_{i+1})}$ on $[x_j, x_{j+1}], \quad j = 0, 1, \ldots, N$ and subsequently $k_i = \min\{\max_j[slope(\psi_j^{(i+1)})], 10\}$.
Step6: Set $\Delta t_i = \frac{t_{i+1} - t_i}{k_i}$.
Step7: Run ALSM k_i times with $t_0 = t_i$, $T_f = t_{i+1}$, and $\Delta t = \Delta t_i$, then again calculate $\psi^{(t_{i+1})}, X^{(t_{i+1})}$ and $\tilde{U}^{(t_{i+1})}$.
Step8: Set $i = i + 1$.
Step9: If $t_i \leq T_f$ go to step4, else break.

4 Numerical Experiment

In this section, we implement the present work for two numerical examples where $u(x, t)$ is a given function. We use the arclength monitor function. In order to obtain an accurate and non-oscillatory solution, it is necessary to smooth the mesh points. Following [5], we have applied smoothed monitor function as below:

$$\tilde{\rho}_i = \frac{\sum_{k=i-ip}^{i+ip} \rho_k (\frac{\gamma}{\gamma+1})^{|k-i|}}{\sum_{k=i-ip}^{i+ip} (\frac{\gamma}{\gamma+1})^{|k-i|}},$$

where ip is a nonnegative integer and γ is a positive constant. In this paper we select the mentioned formula for smoothing monitor function with $ip = 1$ and $\gamma = 1$. We apply MMPDE5 and MMPDE6 in the evolution equation of the level set function for obtaining nodes velocity.

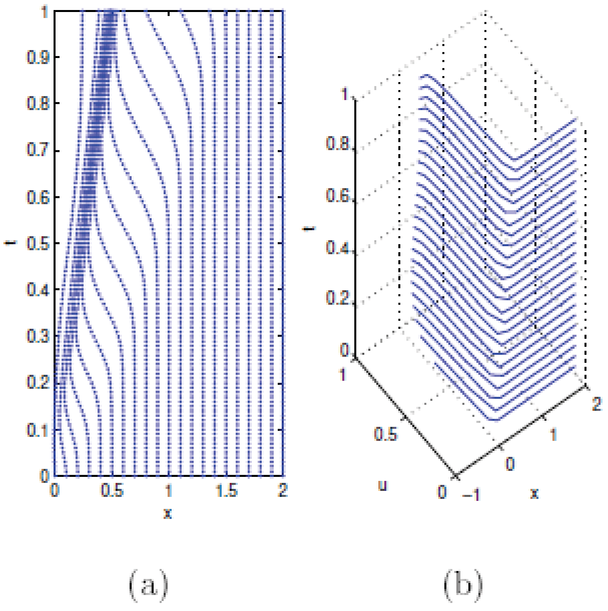

Fig. 1: The mesh trajectory and solution of the first example for $0 \leq t \leq 1.0$ with adaptive level set method(MMPDE5) of 21 mesh point,($\mu = 0.5, \lambda = 0.4, \varepsilon = 0.01, \beta = 0$).

Example 4.1. Consider the Burger's equation as first example,

$$u_t + uu_x - \varepsilon u_{xx} = 0,$$

where its exact solution is:

$$u(x,t) = \frac{\mu + \lambda + (\mu - \lambda) e^{\frac{\lambda}{\varepsilon}(x - \mu t - \beta)}}{1 + e^{\frac{\lambda}{\varepsilon}(x - \mu t - \beta)}}.$$

We select $\mu = 0.5, \lambda = 0.4, \varepsilon = 0.01$ and $\beta = 0$. This problem is characterized by moving discontinuities(specially when ε is very small), that means the discontinuities move in time, and so the solution at a particular point in space

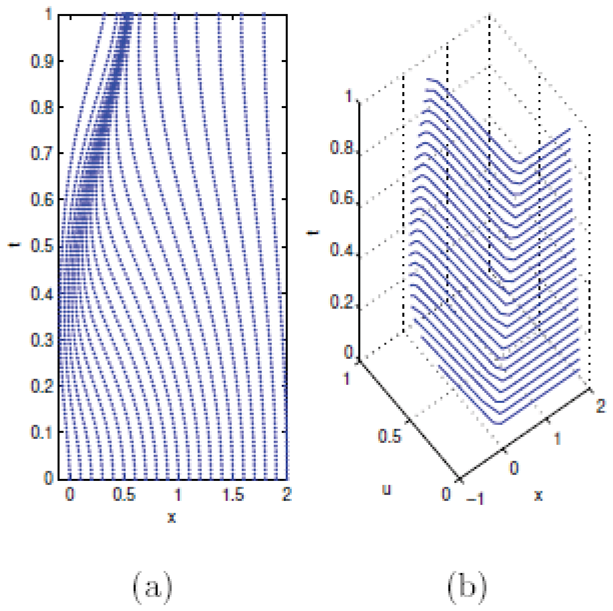

Fig. 2: The mesh trajectory and solution of the first example for $0 \le t \le 1.0$ with adaptive level set method(MMPDE6) of 21 mesh point,($\mu = 0.5, \lambda = 0.4, \varepsilon = 0.01, \beta = 0$).

Table 1: The error of ALSM(with MMPDE5) for the first example at $t = 1.0$, obtained by the arclength monitor function with $\alpha = 0$(uniform mesh) and $\alpha = 2$(moving mesh)without LTSR and with LTSR, ($\mu = 0.5, \lambda = 0.4, \varepsilon = 0.01$ and $\beta = 0$).

MMPDE	Time stepping	Type of mesh	N	L^∞-error	CPU time
			21	0.1924	0.504
5	without LTSR	fixed mesh	31	0.0782	0.612
		($\alpha = 0$)	41	0.0564	0.8807
			21	0.0711	0.7548
5	without LTSR	Moving mesh	31	0.0602	0.8315
		($\alpha = 2$)	41	0.0587	0.9812
			21	0.0510	0.917
5	with LTSR	Moving mesh	31	0.0347	1.019
		($\alpha = 2$)	41	0.0298	1.079

can change very rapidly. The solution of such a problem on a fixed uniform spatial mesh, needs very small time step in order to have sufficient accuracy, but using the adaptive mesh for finding the solution of this problem improves both the accuracy and the efficiency.

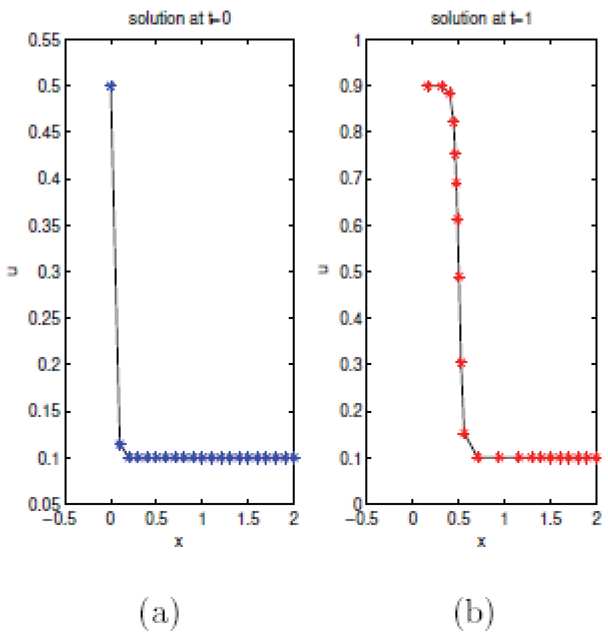

Fig. 3: The solution at $t = 0$ with uniform mesh and at $t = 1$ with adaptive level set method(MMPDE5) of 21 mesh point for the first example, ($\mu = 0.5, \lambda = 0.4, \varepsilon = 0.01, \beta = 0$).

Table 2: The error of ALSM(with MMPDE6) for the first example at $t = 1.0$, obtained by the arclength monitor function with $\alpha = 0$ (uniform mesh) and $\alpha = 2$(moving mesh)without LTSR and with LTSR, ($\mu = 0.5, \lambda = 0.4, \varepsilon = 0.01$ and $\beta = 0$).

MMPDE	Time stepping	Type of mesh	N	L^∞-error	CPU time
			21	0.0645	0.8815
6	without LTSR	fixed mesh	31	0.0578	0.9205
		($\alpha = 0$)	41	0.0512	0.9876
			21	0.0441	0.031
6	without LTSR	Moving mesh	31	0.0315	1.108
		($\alpha = 2$)	41	0.0274	1.230

For this reason, we have used new adaptive mesh for this problem and in order to demonstrate the efficiency of the adaptive level set method (ALSM)(which has combined with MMPDE5), and also to compare ALSM with or without LTSR, some results are presented in Table1. These results show L^∞-error and CPU times for different number of mesh points. These results certify higher accuracy of the mentioned method(ALSM) in comparison with using uniform mesh. Besides, it shows that using the adaptive method

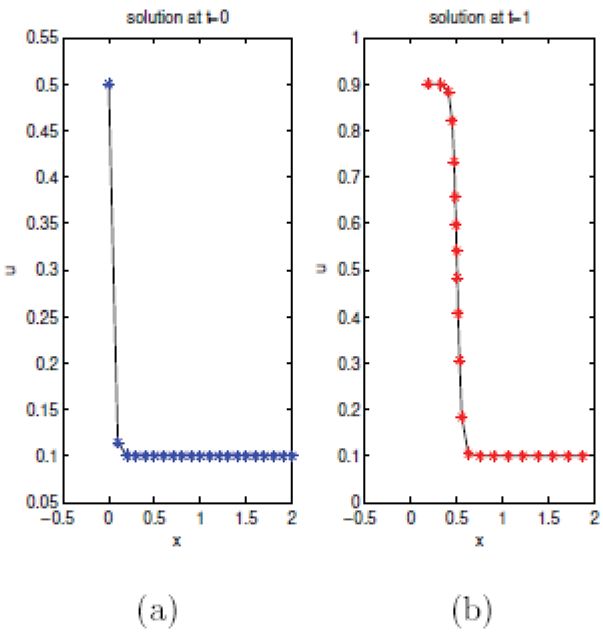

Fig. 4: The solution at $t = 0$ with uniform mesh and at $t = 1$ with adaptive level set method(MMPDE6) of 21 mesh point for the first example, $(\mu = 0.5, \lambda = 0.4, \varepsilon = 0.01, \beta = 0)$.

with LTSR gives better results. Similar information, are given in Table 2 when the MMPDE6 is applied for calculating nodes velocity. Figure 1(a) shows mesh trajectories for $0 \le x \le 2, 0 \le t \le 1.0$ which has been derived by using the new method with MMPDE5, also the solution of PDE, u, has been plotted on the moving mesh for $0 \le t \le 1.0$ in Figure 1(b). The spatial domain is divide into 21 mesh points. Like above, mesh trajectories and solution of Example 4.1 have been plotted by using the new method with MMPDE6 in Figure 2. Also in Figure 3 we plotted the solution of PDE at $t = 0$ on a uniform initial mesh and at $t = 1$ on a moving mesh with MMPDE5 and similarly, the solution at $t = 0$ and $t = 1$ has been plotted with MMPDE6 in Figure 4.
For demonstrating the efficiency of our method we compare this method with another moving mesh method. For this purpose, we consider moving element free Petrov-Galerkin viscous method, MEFP-GVP, where introduced in [2], for comparing under equal conditions we consider the parameters $\mu, \lambda, \varepsilon$ and β in the Burger's equation according to [2], $(\mu = 0.5, \lambda = 0.4, \varepsilon = 1/15$ and $\beta = 0.16)$. Also, we solve this equation for $x \in [0, 1]$ and $t \in [0, 0.51]$ again. With above conditions and with $N = 7$ grid points the L^∞-error is 0.03 by

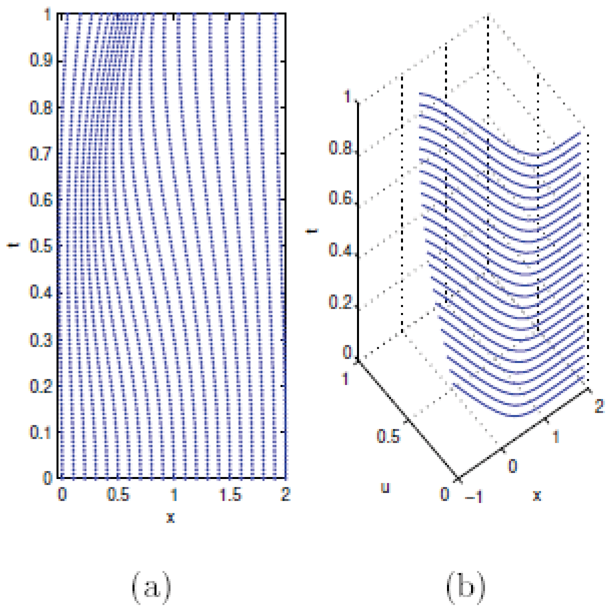

Fig. 5: The mesh trajectory and solution of the first example for $0 \leq t \leq 1.0$ with adaptive level set method(MMPDE5) of 21 mesh point,($\mu = 0.5, \lambda = 0.4, \varepsilon = 1/15, \beta = 0$).

MEFP-GVM but in ALSM we have L^∞- error= 0.01. This result shows the preference of our method. In addition, the CPU time in our method is very smaller than the MEFP-GVM. Also we plotted grid motion and the solution for new conditions for 21 grid points in Figure 5.

Table 3: The error of ALSM(with MMPDE5) for the second example at $t = 1.0$, obtained by the arclength monitor function ($\alpha = 2$) without LTSR and with LTSR.

MMPDE	Time stepping	N	L^∞-error	CPU time
		21	0.0060	0.7213
5	without LTSR	31	0.0052	0.8507
		41	0.0050	0.9759
		21	0.0051	0.906
5	without LTSR	31	0.0032	1.015
		41	0.0029	1.076

Example 4.2. Consider the equation

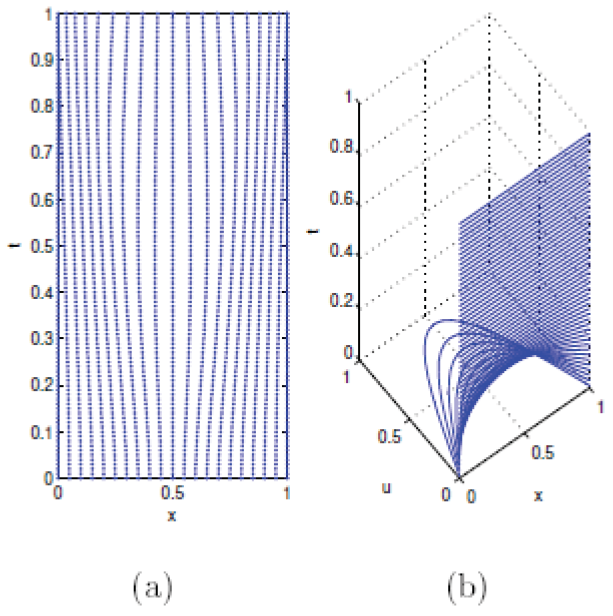

Fig. 6: The mesh trajectory and solution of the second problem for $0 \leq x \leq 1$ and $0 \leq t \leq 1.0$ with adaptive level set method(MMPDE5) of 21 mesh point.

Table 4: The error of ALSM(with MMPDE6) for the second example at $t = 1.0$, obtained by the arclength monitor function ($\alpha = 2$) without LTSR and with LTSR.

MMPDE	Time stepping	N	L^∞-error	CPU time
		21	0.0023	0.8706
6	without LTSR	31	0.0015	0.9844
		41	0.0009	1.029
		21	0.0048	1.025
6	without LTSR	31	0.0030	1.098
		41	0.0024	1.221

$$u_t = u_{xx},$$

with the exact solution:

$$u(x,t) = e^{-\pi^2 t} \sin \pi x$$

This problem has been used in [3] as a numerical example and also in [15] to study the moving mesh with variable relaxation time.
Since $u_x(x,t) \to 0$ in the limit as $t \to +\infty$, then for typical arclength monitor function

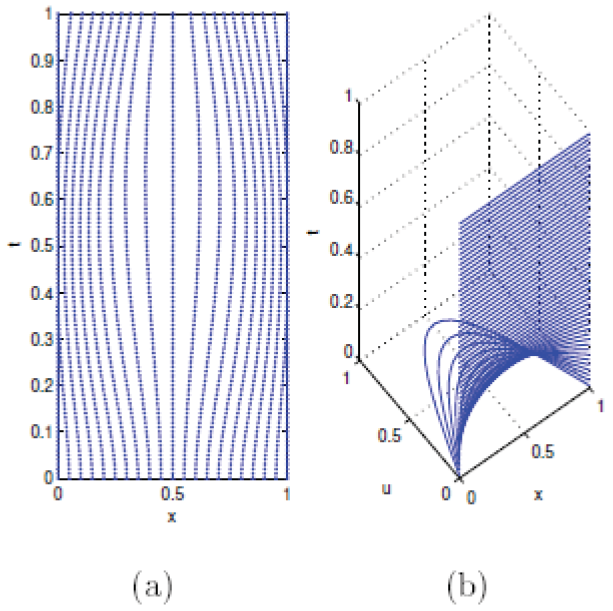

Fig. 7: The mesh trajectory and solution of the second problem for $0 \leq x \leq 1$ and $0 \leq t \leq 1.0$ with adaptive level set method(MMPDE6) of 21 mesh point.

$$M = \sqrt{1 + \alpha u_x^2}$$

we have $M(x,t) \to 1$ as $t \to +\infty$ and so the equidistributed mesh should tend to a uniform mesh in space.
Similar to Tables 1 and 2, some results about the second example are given in Tables 3 and 4 which certify the quality of the new mesh adaptive algorithm. Figure 6(a) shows the mesh trajectories for the above problem using adaptive level set method with MMPDE5 and in Figure 6(b), the numerical solution of PDE is plotted for $0 \leq t \leq 1.0$ by the new method with MMPDE6 have been plotted. We also plotted the solution at $t = 0$ on uniform initial mesh and at the times $t = 0.3, 0.5$ and $t = 1$ on moving mesh in Figure 8. As we expect, the adaptive mesh at $t = 1$ tends towards uniform mesh.
In the second example, because of the smoothness of the solution there is not any major difference between the uniform and the adaptive mesh and also between the adaptive mesh with MMPDE5 and MMPDE6. In this example, ALSM with LTSR does not yield any different result from ALSM without LTSR.

Example 4.3. We use the following Burger's equation as the third example,

$$u_t + uu_x - \varepsilon u_{xx} = 0, \quad 0 < x < x_R.$$

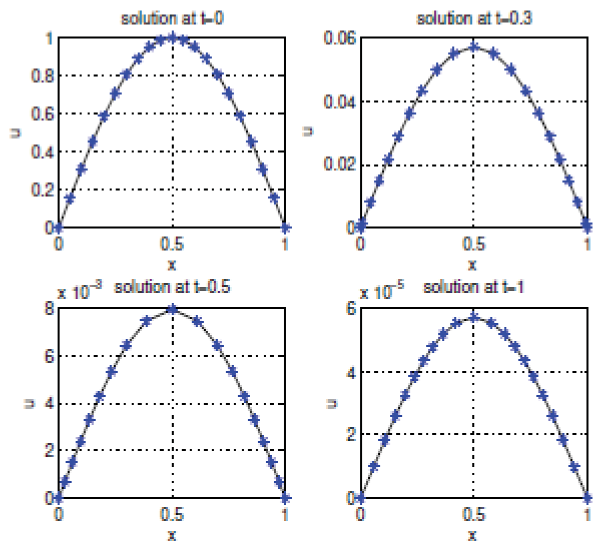

Fig. 8: The solution of the second PDE at $t = 0, 0.3, 0.5$ and $t = 1$ with its corresponding mesh nodes.

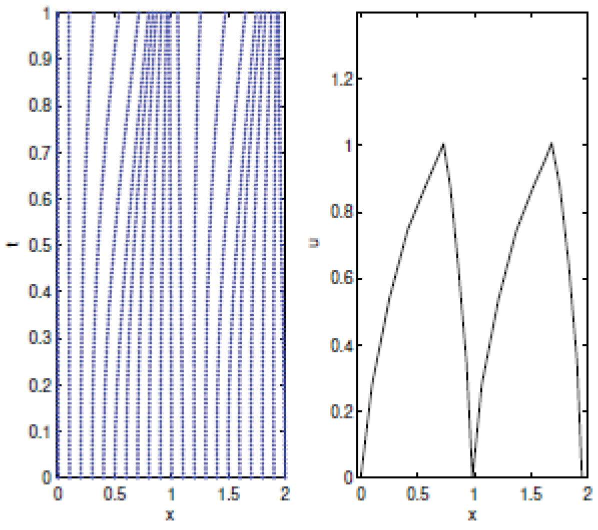

Fig. 9: The mesh trajectory and the computed solution of the third example up to $t = 1$ for $0 < x < 2$ with adaptive level set method(MMPDE5) of 21 mesh points.

The initial condition is considered as follows [7]:

$$u(x,0) = |\sin(\frac{2\pi x}{x_R})|.$$

The boundary conditions are chosen in the form

$$u(0,t) = u(x_R,t) = 0.$$

We select $\varepsilon = 10^{-5}$ and $x_R = 2$ in this example. This example has been solved by the presented method up to $t = 1$.
The mesh trajectory has been plotted along with the nodes of moving grid with $N = 21$ mesh points in different times in Figure 9. Also the computed solution of the third example has been plotted in Figure 9. It is clear that the mesh moves correctly, because the solution has large variations about $x = 0.5$ and $x = 2$ and the mesh also concentrates around these areas.

5 Conclusion

In this paper, we have developed a static moving mesh method based on the level set approach for solving one dimensional time dependent PDEs, such that for representing the nodes of adaptive mesh, the level set equation is used. Also, we proposed a strategy for local time step refinement where the local time step is selected by the level set function. This adaptive method is among the static moving mesh. We plan to investigate the dynamical moving mesh method based on the level set function.

References

1. Aslam, T., *A level set algorithm for tracking discontinuities in hyperbolic conservation laws*,I:*Scalar equations*, J. Comput. Phys. **167**, (2001), 413-438.
2. Ghorbani, M. and Soheili, A.R.,*Moving element free Petrov Galerkian viscous method,special issue on meshless methodes*, Journal of the Chinese Institue of engineers, **27**, (2004), 473-479.
3. Huang, W., Ren, Y. and Russell, R., *Moving mesh partial differential equations(MMPDEs)based on the equiditribution principle*, SIAM J.Number. Anal. **31**, (1994), 709-730.
4. Huang, W., Ren, Y. and Russell, R., *Moving mesh methods based on moving mesh partial differential equations*, J. Compute. Phys., **113**, (1994), 279-290.
5. Jin, S. and Osher, S.,*A level set method for the computation of multivalued solutions to quasi-linear hyperbolic PDEs and Hamilton-Jacobi equations*, SIAM .Numer. Analys. **28**, (1991), 907-921.
6. Liao, G., Liu, F., Pena, G.D., peng, D. and Osher, S., *Level-set-based deformation methods for adaptive grids*, J.Compute. Phys., **159**, (2000), 103-122.

7. Mazhukin, A.V., *Dynamic adaptation in convection-diffusion equations*, Comput. method. Appl. Math, **8**, (2008), 171-186.
8. Mazhukin, A.V. and Chuiko, M.M., *Solution of the multi-interface Stefan problem by the method of dynamic adaptation*, Compute. method. Appl. Math, **2**, (2002), 283-294.
9. Osher, S. and Fedkiw, R., *Level set methods and dynamic implicit surfaces*, Springer-Verlag, NY, (2002)
10. Osher, S. and Sanders, R., *Numerical approxiamtion to nonlinear conservation laws with locally varying time and spaces grid* , Math. Compute. **41**, (1983), 321-336.
11. Osher, S. and Sethian, J., *Front propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations*, J. Compute. Phys. **79**, 12 (1988).
12. Sethian, J.A.*Level set methods and fast marching methods*, New York: Cambridge University Press,(1999).
13. Soheili, A.R. and Ameri, M.A., *Adaptive grid based on geometric conservation law level set method for time dependent PDE* , Numerical methods for PDEs, **25**, (2009), 582-597.
14. Soheili, A.R. and Salahshour, S., *Moving mesh methods with local time step refinement for blow-up problems*, Appl.Math.Copmpute. **195**, (2008), 76-85.
15. Soheili, A.R. and Stockie, J.M.,*A moving mesh method with variable mesh relaxation time*, Appl.Number.Math. **58**, (2008), 249-263.
16. Stockie, J.M., Mackenzie, J.A. and Russell, R., *A moving mesh method for one dimensional hyperbolic conservation laws*, SIAM. J.S.Compute. **22**, (2001), 1791-1831.
17. Tan, H.Z. and Tang, T., *Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws*, SIAM. J.Numer. Anal. **41**, (2003), 487-515.
18. Tan, Z., Zhang, Z., Huang, Y. and Tang, T.,*Moving mesh methods with locally varying time step*, J. Comp. Phys. **200**, (2004), 347-367.
19. Tasi, Y.H.R. and Giga, Y. and Osher, S.,*A level set approach for computing discontinuous solutions of Hamilton-Jacobi equations*, J. Math. Compute. **72**, (2002), 159-181.