Finite Volume Method and Its Convergence

Zexuan Sun

Major: Information and Computational Science

Student ID: 2016301000038

1 Finite Volume Method

Based on the conservation law, we derive the difference scheme of the finite volume method as follows:

$$u_i^{n+1} = u_i^n + \Delta t F_i^n + \frac{\nu \Delta t}{(\Delta x)^2}(u_{i+1}^n - 2u_i^n + u_{i-1}^n).$$

Typically, we have two choices concerning the grids partition, namely, vertex centered grids and cell centered grids. There two choices can have different performances under different boundary conditions.

1.1 Neumann Boundary Conditions

Here we only consider the case where the left boundary is the Neumann boundary condition, namely $\partial_x v(0,t) = g(t), \forall t > 0$, for the case where the right boundary is the Neumann boundary condition is similar. Using the finite volume method and take the unit center method, we get the following format at the boundary point:

$$u_1^{n+1} = u_1^n + \Delta t F_1^n + \frac{\nu \Delta t}{\Delta x^2}(u_2^n - u_1^n) - \frac{\nu \Delta t}{\Delta x} g^n,$$

Usually for the convenience of programming, we add a ghost point so that the end points are updated in the same scheme as the interior, i.e.

$$u_0^n = u_1^n - \Delta g^n.$$

In the case of vertex centered grids, we get the scheme as below:

$$u_0^{n+1} = u_0^n + \Delta t F_0^n + \frac{2\nu \Delta t}{\Delta x^2}(u_1^n - u_0^n) - \frac{2\nu \Delta t}{\Delta x} g^n,$$

Similarly, in the case of cell centered grids, using ghost point, we set

$$u_{-1}^n = u_1^n - 2\Delta g^n.$$

From the schemes of these ghost points using different grids, we can say that, theoretically, we can get higher accuracy for Neumann boundary conditions using cell centered grids.

1.2 Dirichlet Boundary Conditions

As above, we only consider the case where the left boundary is the Dirichlet boundary condition. Use the finite volume method and choose the cell centered grids, we get the following scheme at the end points:

$$u_1^{n+1} = u_1^n + \Delta t F_1^n + \frac{\nu \Delta t}{\Delta x^2}(u_2^n - 3u_1^n + 2g^n).$$

For vertex centered grids, we get the scheme as below:

$$u_0^{n+1} = g^{n+1}.$$

Obviously, for the Dirichlet boundary condition using the vertex centered grids we can get the better solution.

1.3 Examples

In order to test the actual performance of using cell centered and vertex centered grids under different boundary conditions, we conducted numerical experiments on the following instance:

$$\begin{cases} v_t = \nu v_{xx} + F(x,t), & x \in (0,1), t > 0 \\ v(x,0) = f(x), & x \in (0,1) \\ v_x(0,t) = a(t), v(1,t) = b(t), & t \geq 0 \end{cases}$$

where $\nu = 0.1, f(x) = x(1-x), a(t) = 10sint, b(t) = 4sin6t, F(x,t) = sin2\pi xsin4\pi t$. We tested the following three schemes:

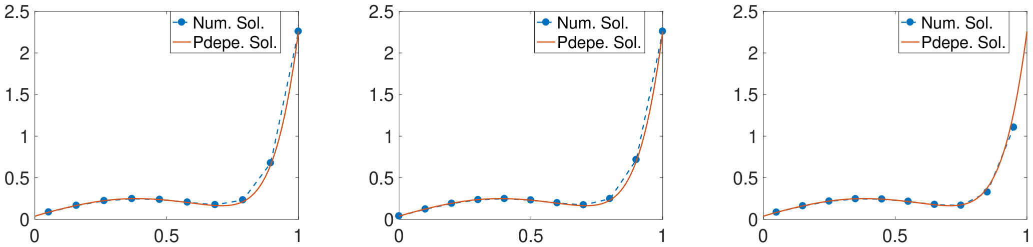
- left end point is cell centered, right is vertex centered, which we refer as lcrb.

- both end points are vertex centered, which we refer as lbrb.
- both end points are cell centered, which we refer as lcrc.

In order to compare the accuracy of solutions we get, we use the solution obtained by Matlab's pdepe function as a baseline. At the same time, we implemented the matlab_sol function encapsulating the pdepe function for the convience of programming later. We also wrote there functions lcrb(Dt,Tend,choice), lbrb(Dt,Tend,choice), lcrc(Dt,Tend,choice), which implement the aforementioned three schemes, individually. The Dt is time step and Tend is the end time. The fuctions plot the figure of the solutions drived from current scheme and pdepe when choice is 1. And the functions plot the figure of error, i.e. the absolute error between the solution derived from the current scheme and the solution obtained by pdepe. Also, these three functions would output the average of the error on each grid point, i.e. the sum of all errors divided by the number of grid points.

At first, we plotted the image of the solution derived from the three schemes and the solution obtained by pdepe, separately, when the end time t is $0.1, 0.9, 2.0$ and the time step is 0.01. The result is shown in Fig.1-3.

From the image we can see that all the three schemes have done a good job, except that lcrc has an obvious error at the right end point, which is obviously caused by the use of the cell center format. In order to better compare the accuracy of the three schemes, we draw the error of the three schemes using the solution obtained by pdepe, when $t = 0.1, 0.9, 2.0$ and time step is 0.01. The result is exhibited in Fig.4-6.

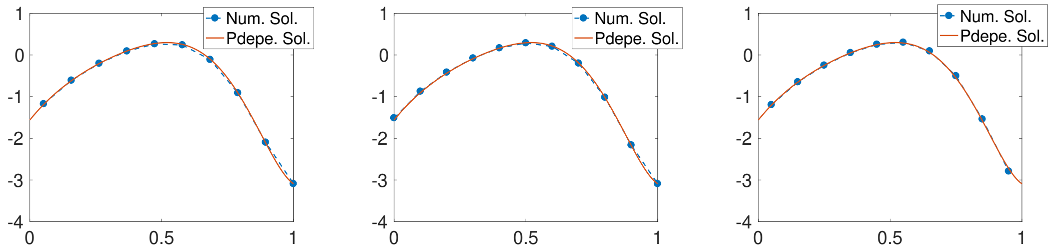


(a) $t = 0.1$, lcrb

(b) $t = 0.1$, lbrb

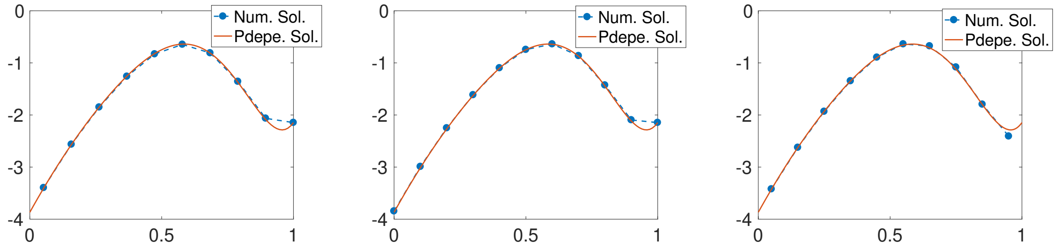
(c) $t = 0.1$, lcrc

Figure 1: The comparison of the three schemes with pdepe, when $t = 0.1$ and time step is 0.01



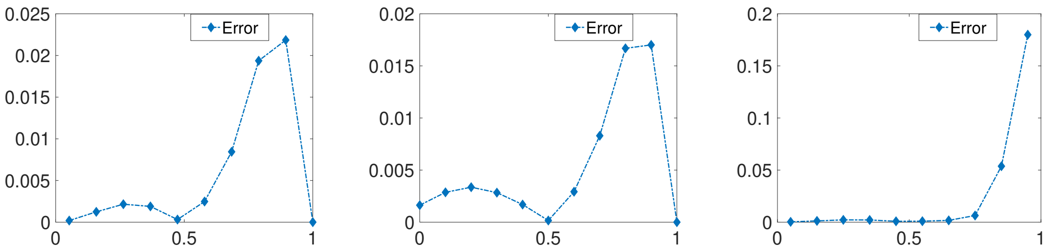
(a) lcrb (b) lbrb (c) lcrc

Figure 2: The comparison of the three schemes with pdepe, when $t = 0.9$ and time step is 0.01



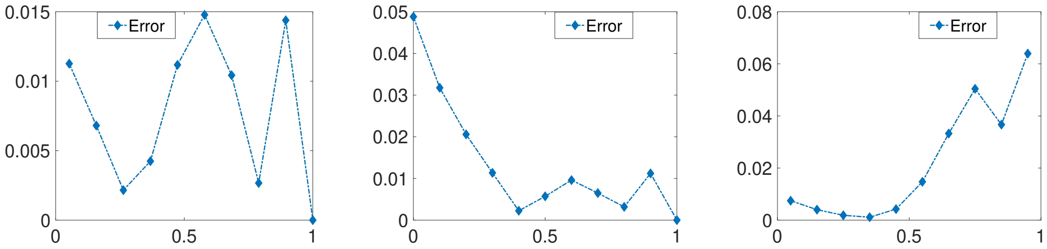
(a) lcrb (b) lbrb (c) lcrc

Figure 3: The comparison of the three schemes with pdepe, when $t = 2.0$ and time step is 0.01



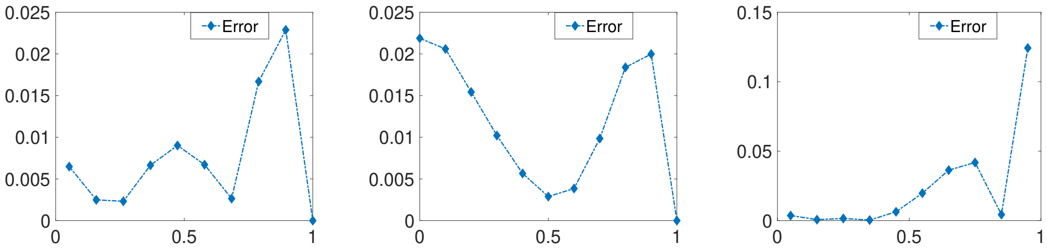
(a) lcrb (b) lbrb (c) lcrc

Figure 4: The error of the three schemes when $t = 0.1$ and time step is 0.01



(a) lcrb (b) lbrb (c) lcrc

Figure 5: The error of the three schemes when $t = 0.9$ and time step is 0.01



(a) lcrb (b) lbrb (c) lcrc

Figure 6: The error of the three schemes when $t = 2.0$ and time step is 0.01

From the figures above, we can see that when $t = 0.1$, the maximum value of the error in both the lcrb and lbrb is about the same, about 0.02. But, lcrb has a smaller error than lbrb at the left end point. The lcrc has a larger value of error at the last point near the right boundary due to the cell centered scheme at the right boundary. When $t = 0.9$, lcrb performed best among the three schemes, with all errors smaller than 0,015. But lbrb has relatively large value of error at left boundary and lcrc has even larger value of error at the right boundary, about 0.07. When $t = 2.0$, lcrb and lbrb have similar performances, but lcrb had smaller error at left boundary. And the large error of lcrc at the right boundary makes the overall performace of lcrc worse compared with other two schemes.

To better illustrate the overall error value when $t = 0.1, 0.9, 2.0$ and time step is 0.01, We calculated the average error values of grid points using the three formats under different conditions. And the result is shown in Table.1.

From the table, we can see that lcrb performed the best among the three schemes except when $t = 0.1$. And lcrc performed the worst.

Scheme	$t = 0.1$	$t = 0.9$	$t = 2.0$
lcrb	0.0058	0.0078	0.0076
lbrb	0.0052	0.0137	0.0117
lcrc	0.0249	0.0218	0.0239

Table 1: The average error values using the three schemes

1.4 Conclusions

From the numerical experiments above, we can come to the conclusion that for this instance, lcrb is the best scheme, lcrc is the wrost, which is consistent with our theoretical analysis. That is, we should adopt the cell centered scheme for Neumann boundary conditions, and for Dirichlet boundary conditions we should adopt the vertex centered scheme.

2 Convergence of Finite Volume Method

Typically, we say difference equation $L_i^n u_i^n = F_i^n$ converges pointwise to the solution of the differential equation $Lv = F$, if for any x and t,

$$\lim_{\substack{\triangle x \to 0, \triangle t \to 0 \\ (k\triangle x,(n+1)\triangle t)\to(x,t)}} u_i^n = v(x,t)$$

We have proved that for initial boundary value problems of partial differential equation as below:

$$\begin{cases} v_t = \nu v_{xx}, & x \in (0,1), t > 0, \\ v(x,0) = f(x), & x \in [0,1], \\ v(0,x) = v(1,t) = 0, & t \geq 0 \end{cases}$$

where $\nu = 0.1, f(x) = sin2\pi x$. When we choose the following scheme:

$$\begin{cases} u_i^{n+1} = (1-2r)u_i^n + r(u_{i+1}^n + u_{i-1}^n), & n \geq 0, i = 1,2,...,N-1 \\ u_0^{n+1} = u_N^{n+1}, & n \geq 0 \\ u_i^0 = f(i\triangle x) \end{cases}$$

where $r = \nu\triangle/\triangle x^2, 0 \leq r \leq \frac{1}{2}$, then the difference scheme converges pointwise to the solution of the differential equation. Next, we would like to verify this fact by numerical experiments.

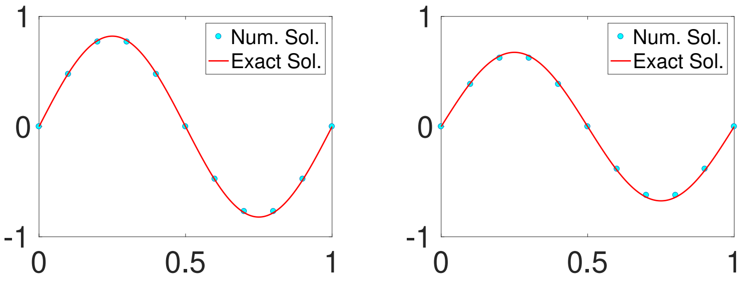
2.1 Numerical Experiments

We only calculate the numerical results when $t = 0.05 and 0.1$. And we only implement tests on the following three types of the combinations of space step and time step:

- $\triangle x = 0.1, \triangle t = 0.05$
- $\triangle x = 0.05, \triangle t = 0.00125$
- $\triangle x = 0.1, \triangle t = 0.0005$

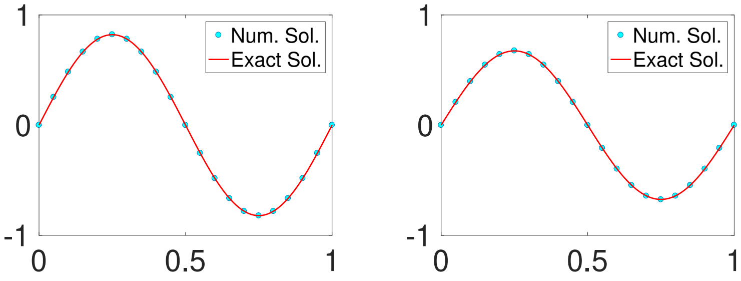
We wrote function FDM _eg3(DX,DT,TEND,choice) to implement the difference method, when DX is space step, DT is time step and TEND is the end time. When choice is set to 1, the function plot the figure of numerical solution and analytical solution. And when choice is set to 2, the function plot the figure of error. At the same time, the function would output the average of the error on each grid point, i.e. the sum of all errors divided by the number of grid points.

At first, we make comparisons of the numerical solutions and analytical solutions in three kinds of steps when $t = 0.05$ and 0.1. The results are shown in Fig.7-9.



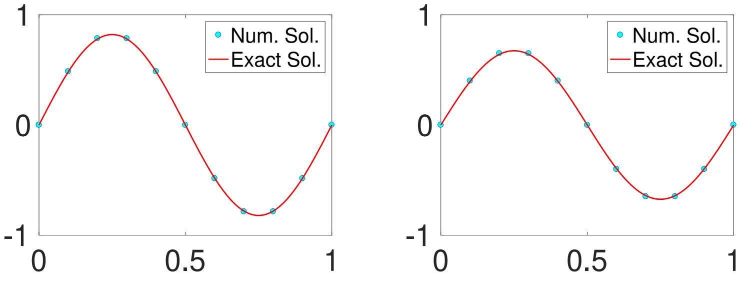
(a) $t = 0.05$ (b) $t = 0.1$

Figure 7: $\triangle x = 0.1, \triangle t = 0.05$, the comparison of numerical solutions and analytical solutions



(a) $t = 0.05$ (b) $t = 0.1$

Figure 8: $\triangle x = 0.05, \triangle t = 0.00125$， the comparison of numerical solutions and analytical solutions

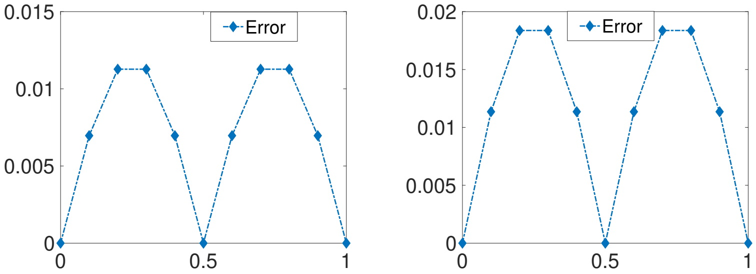


(a) $t = 0.05$

(b) $t = 0.1$

Figure 9: $\triangle x = 0.1, \triangle t = 0.0005$，the comparison of numerical solutions and analytical solutions

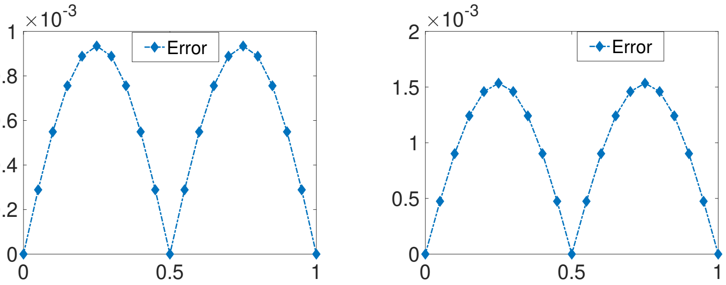
From the figures, we can see that numerical solutions are quite accurate in the three situations. Then we draw the figures of error to better illustrate the accuracy of the numerical solutions. The results are exhibited in Fig.10-12.



(a) $t = 0.05$

(b) $t = 0.1$

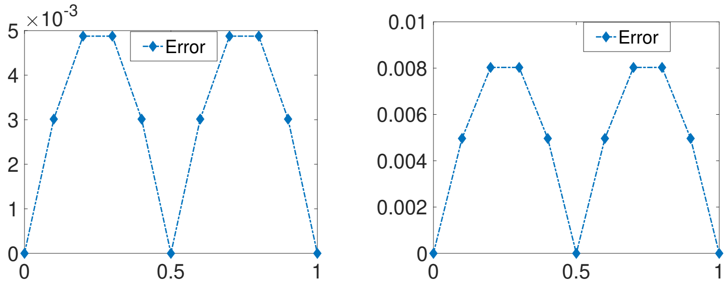
Figure 10: $\triangle x = 0.1, \triangle t = 0.05$, the figure of error



(a) $t = 0.05$

(b) $t = 0.1$

Figure 11: $\triangle x = 0.05, \triangle t = 0.00125$, the figure of error



(a) $t = 0.05$

(b) $t = 0.1$

Figure 12: $\triangle x = 0.1, \triangle t = 0.0005$, the figure of error

We can see that as time goes by, the error increases. For the second type of step, since it has smaller step in both time and space, the error of which is lower. As for the third type of step, although it just has smaller time step, it is also more accurate. And it is obvious that the second type of step is more accurate than the third one. Finally, we calculate the average error value of the three step we chose, the results are shown below: We can draw the same conclusion from the table. That is, the second type of stop is the best, the first is the worst.

Step Type	$t = 0.05$	$t = 0.1$
$\triangle x = 0.1, \triangle t = 0.05$	0.0066	0.0108
$\triangle x = 0.05, \triangle t = 0.00125$	5.62e-04	9.23e-04
$\triangle x = 0.1, \triangle t = 0.0005$	0.0029	0.0047

Table 2: The average error value in three steps, when $t = 0.05$ and 0.1

2.2 Conclusion

According to the numerical experiments we have implemented, the difference equation would converge pointwise to the solution of differential equation when $\triangle x \to 0, \triangle t \to 0$.