Chapter 16
Finite Volume Methods

In the previous chapter we have discussed finite difference methods for the discretization of PDEs. In developing finite difference methods we started from the differential form of the conservation law and approximated the partial derivatives using finite difference approximations. In the finite volume method we will work directly with the integral form of the conservation law 75.

57 Self-Assessment

Before reading this chapter, you may wish to review...

- Integral form of conservation laws 11
- Convection Equation 11
- Upwinding 13

After reading this chapter you should be able to...

- Understand the difference between finite difference and finite volume methods
-

Relevant self-assessment exercises: [LIST SELF-ASSESSMENT EXERCISES HERE]

58 Finite Volume Method in 1-D

The basis of the finite volume method is the integral conservation law. The essential idea is to divide the domain into many control volumes (or cells) and approximate the integral conservation law on each of the control volumes. Figure 28 shows an example of a partition of a one-dimensional domain into cells. By convention cell i lies between the points $x_{i-\frac{1}{2}}$ and $x_{i+\frac{1}{2}}$. (Notice that the cells do not need to be of equal size.)

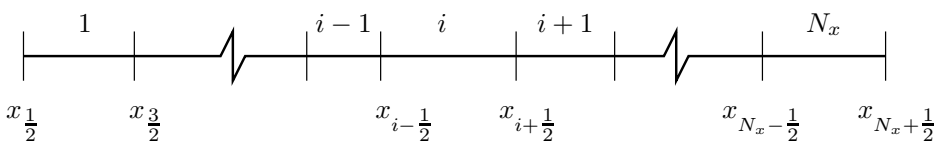

Fig. 28 Mesh and notation for one-dimensional finite volume method.

Recall the integral form of the conservation law (i.e. Equation 75). The one-dimensional form of Equation 75 is

$$\frac{d}{dt}\int_{x_L}^{x_R} U\,dx + F(U)|_{x_R} - F(U)|_{x_L} = \int_{x_L}^{x_R} S(U,t)\,dx. \tag{121}$$

Thus, applying this to control volume i gives, and for now assuming $S = 0$,

$$\frac{d}{dt}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} U\,dx + F(U)|_{x_{i+\frac{1}{2}}} - F(U)|_{x_{i-\frac{1}{2}}} = 0. \tag{122}$$

Next, we define the mean value of U in control volume i as,

$$U_i \equiv \frac{1}{\Delta x_i}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} U\,dx, \quad \text{where} \quad \Delta x_i \equiv x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}. \tag{123}$$

Then, Equation 122 becomes,

$$\Delta x_i \frac{dU_i}{dt} + F(U)|_{x_{i+\frac{1}{2}}} - F(U)|_{x_{i-\frac{1}{2}}} = 0. \tag{124}$$

At this point, no approximations have been made thus Equation 124 is exact. Now, we make the first approximation. Specifically, we assume that the solution in each control volume is constant,

$$U(x,t) = U_i(t) \quad \text{for} \quad x_{i-\frac{1}{2}} < x < x_{i+\frac{1}{2}}. \tag{125}$$

Thus, the finite volume approximation will be piecewise constant as shown in Figure 29.

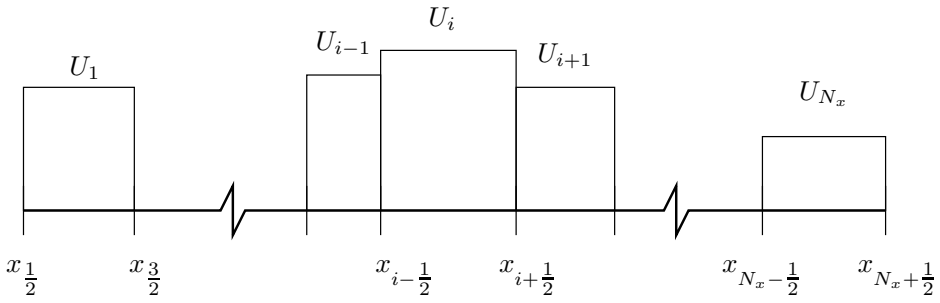

Fig. 29 Piecewise constant solution for one-dimensional finite volume method.

With this assumed form of the solution, the next issue is to determine the flux at $x_{i\pm\frac{1}{2}}$ at time t. Notice that the finite volume approximation is discontinuous at the interface between any two cells. Namely, as we approach $x_{i+\frac{1}{2}}$ from the left our numerical solution gives $\lim_{\varepsilon\to 0^-} U(t)|_{x_{i+\frac{1}{2}}+\varepsilon} = U_i(t)$, but if we approach from the right then we have $\lim_{\varepsilon\to 0^+} U(t)|_{x_{i+\frac{1}{2}}+\varepsilon} = U_{i+1}(t)$. In order for our numerical scheme to be successful we must ensure that the numerical flux we use is consistent with the behavior of the underlying PDE problem. In particular we want to choose F as the instantaneous flux obtained from solving the conservation law exactly with initial condition at time t corresponding to our numerical approximation.

Consider first the case of one-dimensional convection equation. Recall from Chapter 11 that the solution convects with velocity $u(t)$. Thus, for the initial instant after t (which we denote as $t^+ = t + \varepsilon$ where ε is an infinitesimal, positive number):

$$U(x_{i+\frac{1}{2}}, t^+) = \begin{cases} U_i(t) & \text{if } u(t) > 0 \\ U_{i+1}(t) & \text{if } u(t) < 0 \end{cases} \tag{126}$$

The corresponding flux can be calculated directly from this value of U,

$$F(x_{i+\frac{1}{2}},t^+) = \begin{cases} u(t)U_i(t) & \text{if } u(t) > 0 \\ u(t)U_{i+1}(t) & \text{if } u(t) < 0 \end{cases} \tag{127}$$

An alternative way to write this flux which is valid regardless of the sign of $u(t)$ is,

$$F(x_{i+\frac{1}{2}},t^+) = \frac{1}{2}u(t)\left(U_{i+1}(t)+U_i(t)\right) - \frac{1}{2}|u(t)|\left(U_{i+1}(t)-U_i(t)\right). \tag{128}$$

This flux, which uses the upstream value of U to determine the flux, is known as an 'upwind' flux.

The final step in arriving at a full-discrete approximation for one-dimensional convection is to discretize Equation 124 in time. This can be done choosing any of the ODE integration methods we studied previously. For simplicity, we choose the forward Euler method so that the final fully-discrete form of the finite volume method is,

$$\Delta x_i \frac{U_i^{n+1}-U_i^n}{\Delta t} + F^n_{i+\frac{1}{2}} - F^n_{i-\frac{1}{2}} = 0, \tag{129}$$

where we use the notation,

$$F^n_{i+\frac{1}{2}} = \frac{1}{2}u^n\left(U^n_{i+1}+U^n_i\right) - \frac{1}{2}|u^n|\left(U^n_{i+1}-U^n_i\right). \tag{130}$$

Example 1 (Finite Volume Method applied to 1-D Convection).

The following Matlab script solves the one-dimensional convection equation using the finite volume algorithm given by Equation 129 and 130. The problem is assumed to be periodic so that whatever leaves the domain at $x = x_R$ re-enters it at $x = x_L$.

```
% This Matlab script solves the one-dimensional convection
% equation using a finite volume algorithm.  The
% discretization forward Euler in time.

clear all;
close all;

% Number of points
Nx = 50;
x  = linspace(0,1,Nx+1);
dx = 1/Nx;

% Calculate midpoint values of x in each control volume
xmid = 0.5*(x(1:Nx) + x(2:Nx+1));

% Set velocity
u = 1;

% Set final time
tfinal = 1;

% Set timestep
CFL = 0.5;
dt = CFL*dx/abs(u);

% Set initial condition to U0 = exp(-x^2)
% Note: technically, we should average the initial
% distribution in each cell but I chose to just set
% the value of U in each control volume to the midpoint
% value of U0.
U = 0.75*exp(-((xmid-0.5)/0.1).^2)';
t = 0;

% Loop until t > tfinal
while (t < tfinal),

```

```
  Ubc = [U(Nx); U; U(1)]; % This enforces the periodic bc

  % Calculate the flux at each interface
  F =   0.5*    u *( Ubc(2:Nx+2) + Ubc(1:Nx+1)) ...
      - 0.5*abs(u)*( Ubc(2:Nx+2) - Ubc(1:Nx+1));

  % Calculate residual in each cell
  R = F(2:Nx+1) - F(1:Nx);

  % Forward Euler step
  U = U - (dt/dx)*R;

  % Increment time
  t = t + dt;

  % Plot current solution
  stairs(x,[U; U(Nx)]);
  axis([0, 1, -0.5, 1.5]);
  grid on;
  drawnow;

end

% overlay exact solution
U = 0.75*exp(-((xmid-0.5)/0.1).^2)';
hold on;
stairs(x,[U; U(Nx)], 'r-');
```

58.1 Finite Volume Method in 2-D

The finite volume discretization can be extended to higher-dimensional problems. Suppose the physical domain is divided into a set of triangular control volumes, as shown in Figure 30. Application of Equation 75 to control volume

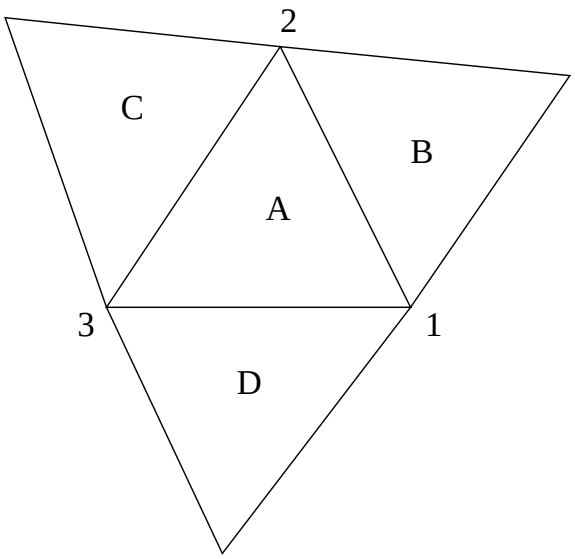

Fig. 30 Triangular mesh and notation for finite volume method.

A gives,

$$\frac{d}{dt}\int_{\Omega_A} U\,dA + \int_{\partial\Omega_A} \mathbf{F}(U)\cdot\mathbf{n}\,ds = \int_{\Omega_A} S(U,t)\,dA, \tag{131}$$

where Ω_A is the interior and $\partial\Omega_A$ is the boundary of control volume A. As in the one-dimensional case, we define the cell average,

$$U_A \equiv \frac{1}{A_A}\int_{\Omega_A} U\,dA, \tag{132}$$

where A_A is the area of control volume A. Thus, Equation 131 becomes,

$$A_A\frac{dU_A}{dt} + \int_{\partial\Omega_A} \mathbf{F}(U)\cdot\mathbf{n}\,ds = \int_{\Omega_A} S(U,t)\,dA. \tag{133}$$

In the case of convection, we again assume $S = 0$. Also, we expand the surface integral into the contributions for the three edges,

$$A_A\frac{dU_A}{dt} + \int_1^2 H(U,\mathbf{n}_{AB})\,ds + \int_2^3 H(U,\mathbf{n}_{AC})\,ds + \int_3^1 H(U,\mathbf{n}_{AD})\,ds = 0, \tag{134}$$

where $H(U,\mathbf{n}) = \mathbf{F}(U)\cdot\mathbf{n}$ and $\mathbf{n}_{AB}$ is the unit normal pointing from cell A to cell B, and similarly for $\mathbf{n}_{AC}$ and $\mathbf{n}_{AD}$.

As in one-dimensional case, we assume that the solution everywhere in the control volume is equal to the cell average value. Finally, the flux at each interface is determined by the 'upwind' value using the velocity component normal to the face. For example, at the interface between cell A and B,

$$H(U,\mathbf{n}_{AB}) \approx \hat{H}(U_A,U_B,\mathbf{n}_{AB}) \equiv \frac{1}{2}\mathbf{v}_{AB}\cdot\mathbf{n}_{AB}\,(U_B+U_A) - \frac{1}{2}|\mathbf{v}_{AB}\cdot\mathbf{n}_{AB}|\,(U_B-U_A)\,, \tag{135}$$

where $\mathbf{v}_{AB}$ is the velocity between the control volumes. Thus, when $\mathbf{v}_{AB}\cdot\mathbf{n}_{AB} > 0$, the flux is determined by the state from cell A, i.e. U_A. Likewise, when $\mathbf{v}_{AB}\cdot\mathbf{n}_{AB} < 0$, the flux is determined by the state from cell B, i.e. U_B. The velocity, $\mathbf{v}_{AB}$ is usually approximated as the velocity at the midpoint of the edge (note: $\mathbf{v}$ can be a function of $\mathbf{x}$ in two-dimensions even though the velocity is assumed to be divergence free, i.e. $\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$). We use the notation $\hat{H}$ to indicate that the flux is an approximation to the true flux when $\mathbf{v}$ is not constant. Thus, the finite volume algorithm prior to time discretization would be given by,

$$A_A\frac{dU_A}{dt} + \hat{H}(U_A,U_B,\mathbf{n}_{AB})\Delta s_{AB} + \hat{H}(U_A,U_C,\mathbf{n}_{AC})\Delta s_{AC} + \hat{H}(U_A,U_D,\mathbf{n}_{AD})\Delta s_{AD} = 0. \tag{136}$$

The final step is to integrate in time. As in the one-dimensional case, we might use a forward Euler algorithm which would result in the final fully discrete finite volume method,

$$A_A\frac{U_A^{n+1}-U_A^n}{\Delta t} + \hat{H}(U_A^n,U_B^n,\mathbf{n}_{AB})\Delta s_{AB} + \hat{H}(U_A^n,U_C^n,\mathbf{n}_{AC})\Delta s_{AC} + \hat{H}(U_A^n,U_D^n,\mathbf{n}_{AD})\Delta s_{AD} = 0. \tag{137}$$

Example 2 (Finite Volume Method for 2-D Convection on a Rectangular Mesh). The following Matlab script solves the two-dimensional convection equation using a two-dimensional finite volume algorithm on rectangular cells. The algorithm is the extension of Equation 137 from triangular to rectangular cells. The problem is assumed to be periodic and have a constant velocity.

```
% This Matlab script solves the one-dimensional convection
% equation using a finite volume algorithm.  The
% discretization forward Euler in time.

clear all;
close all;
```

```
7
8  % Number of points
9  Nx = 50;
10 x  = linspace(0,1,Nx+1);
11 dx = 1/Nx;
12
13 % Calculate midpoint values of x in each control volume
14 xmid = 0.5*(x(1:Nx) + x(2:Nx+1));
15
16 % Set velocity
17 u = 1;
18
19 % Set final time
20 tfinal = 1;
21
22 % Set timestep
23 CFL = 0.5;
24 dt = CFL*dx/abs(u);
25
26 % Set initial condition to U0 = exp(-x^2)
27 % Note: technically, we should average the initial
28 % distribution in each cell but I chose to just set
29 % the value of U in each control volume to the midpoint
30 % value of U0.
31 U = 0.75*exp(-((xmid-0.5)/0.1).^2)';
32 t = 0;
33
34 % Loop until t > tfinal
35 while (t < tfinal),
36
37   Ubc = [U(Nx); U; U(1)]; % This enforces the periodic bc
38
39   % Calculate the flux at each interface
40   F =   0.5*    u *( Ubc(2:Nx+2) + Ubc(1:Nx+1)) ...
41       - 0.5*abs(u)*( Ubc(2:Nx+2) - Ubc(1:Nx+1));
42
43   % Calculate residual in each cell
44   R = F(2:Nx+1) - F(1:Nx);
45
46   % Forward Euler step
47   U = U - (dt/dx)*R;
48
49   % Increment time
50   t = t + dt;
51
52   % Plot current solution
53   stairs(x,[U; U(Nx)]);
54   axis([0, 1, -0.5, 1.5]);
55   grid on;
56   drawnow;
57
58 end
59
60 % overlay exact solution
61 U = 0.75*exp(-((xmid-0.5)/0.1).^2)';
62 hold on;
63 stairs(x,[U; U(Nx)], 'r-');
```

58.2 Finite Volume Method for Nonlinear Systems

The basic finite volume approach can be extended to nonlinear systems of equations such as the Euler equations (see Example 2). The main issue in this extension is how to calculate an upwind flux when there is a system of equations. In one dimension, the basic finite volume discretization remains the same as given by Equation 130,

$$\Delta x_i \frac{U_i^{n+1} - U_i^n}{\Delta t} + F^n_{i+\frac{1}{2}} - F^n_{i-\frac{1}{2}} = 0. \tag{138}$$

The flux, however, must upwind (to some degree) all of the states in the equation. One relatively simple way in which this can be done is using what is known as the local Lax-Friedrichs flux. In this case, the flux is given by,

$$F_{i+\frac{1}{2}}(U_i, U_{i+1}) = \frac{1}{2}\left[F(U_{i+1}) + F(U_i)\right] - \frac{1}{2} s_{\max}\left(U_{i+1} - U_i\right), \tag{139}$$

where $s_{\max}$ is the maximum speed of propagation of any small disturbance for either state U_i or U_{i+1}.

Example 3 (Lax-Friedrichs Flux for 1-D Euler Equations).

For the one-dimensional Euler equations, there are three equations which are approximated, i.e. conservation of mass, conservation of x-momentum, and conservation of energy. A small perturbation analysis can be performed which shows that the three speeds of propagation for this set of equations are u, $u-a$, and $u+a$ where u is the flow velocity and a is the speed of sound. Thus, the maximum speed will always be $|u|+a$ and the corresponding value of $s_{\max}$ for the Lax-Friedrichs flux is,

$$s_{\max} = \max\left(|u|_i + a_i,\, |u|_{i+1} + a_{i+1}\right). \tag{140}$$