有限差分法和有限体积法在计算流体中的应用

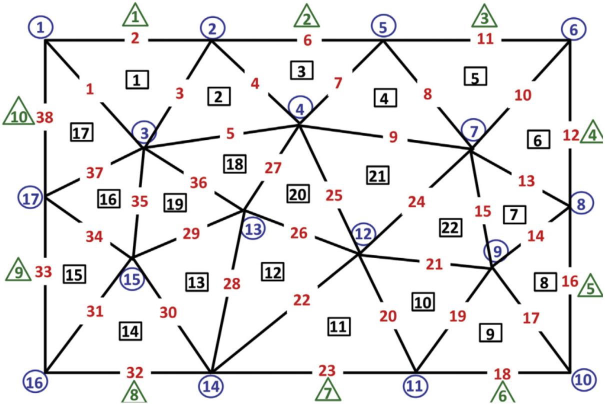
非结构化网格

汪洋

武汉理工大学交通学院

2021 年 2 月

二维三角形划分


1 1 2 2 2 6 5 3 11 6
1 1 3 2 4 3 7 4 8 5 10
10 38 17 3 5 4 9 7 6 12 4
37 36 18 27 20 25 21 24 13
17 16 35 19 15 7 8
34 29 13 26 12 22 9 14
9 33 15 15 13 28 12 21 8 16 5
31 30 14 22 11 20 10 19 9 17
16 32 8 14 23 7 11 18 6 10

连接信息

- 每个体有唯一编号，全局编号。
- 每个面有唯一编号，全局编号。
- 每个节点有唯一编号，全局编号。
- 每个面有局部编号 (围成的体)，所以一个三角形有局部面编号 $1, 2, 3$。
- 每个体有局部编号 (共享的面)，所以每个面有两个局部单元编号 $1, 2$
- 每个边界面有唯一编号，全局编号。

这些局部编号和全局编号的空间关系称为连接性

体到面

`cell-to-face connectivity`，通过二维整形数组存储即可。

- $link\_cell\_to\_face(20,1) = 25$
- $link\_cell\_to\_face(20,2) = 27$
- $link\_cell\_to\_face(20,3) = 26$

面的局部坐标 1,2,3 是网格生成工具定义的，只要一致即可。

面到体

face-to-cell connectivity，通过二维整形数组存储即可。

- $link\_face\_to\_cell(25, 1) = 20$
- $link\_face\_to\_cell(25, 2) = 21$

体的局部坐标 1, 2 是网格生成工具定义的，只要一致即可。考虑下面法向

为什么要这两个信息

因为需要计算两个内容:

- 计算每个体的法向通量。divergence
- 通过体心值计算每个面的值。scalar

计算通量

cell to face 的演示

```
for icell = 1 : ncells
    divergence(icell) = 0
    for ifc = 1 : nface(icell)
        gface = link_cell_to_face(icell, ifc)
        divergence(icell) = divergence + nflux(
           gface) * areaf(gface)
    end
end
```

计算面上的值

face to cell 的演示

```
for iface = 1 : nfaces
    iglobal_cell1 = link_face_to_cell(iface,1)
    iglobal_cell2 = link_face_to_cell(iface,2)
    face_value = interpolation(cell_value(iglobal_cell1), cell_value(iglobal_cell2))
end
```

体到节点，面到节点

存储体到节点、面到节点，与前面类似。

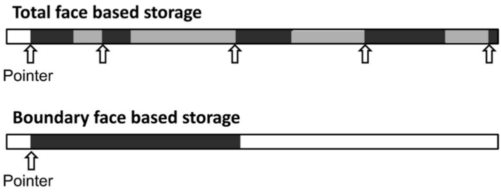
- $link\_face\_to\_node(25,1) = 4$
- $link\_face\_to\_node(25,2) = 12$
- $link\_cell\_to\_node(20,1) = 4$
- $link\_cell\_to\_node(20,2) = 13$
- $link\_cell\_to\_node(20,3) = 12$

在计算插值函数时会用到。

边界面

对于非结构化网格，边界面通常会分开存续。有以下两点好处：

- 并不是所有信息与内部面有关，有些信息仅仅存在边界面。
- 从内存读取数据角度



因此要提前处理边界面的编号信息

- $link\_face\_to\_bface(33) = 9$
- $link\_bface\_to\_face(9) = 33$

计算面上的值

基于face-based data structure 的演示

```
sumphif = 0
sumarea = 0
for ifc = 1 : nfaces
    if (bface(ifc) == 1)
        sumphif = sumphif + phif(ifc) * areaf(
           ifc)
        sumarea = sumarea + areaf(ifc)
    endif
end
average = sumphif / sumarea
```

计算面上的值

基于boundary-based data structure 的演示

```
sumphif = 0
sumarea = 0
for ibfc = 1 : nbfaces
    gfc = link_bface_to_face(ibfc)
    sumphif = sumphif + phif(ibfc) * areaf(gfc)
    sumarea = sumarea + areaf(gfc)
end
average = sumphif / sumarea
```

优缺点

基于 face-based 数据结构

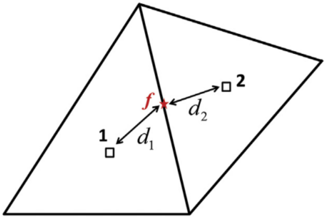
- 增加许多逻辑判断，这样会导致分支预测失败 (branch mispredictions)
- 指针会移动来移动去，会造成无效

interpolation

插值

- 为了从体心得到面心和节点值，只能选用合理的插值方法
- 一般采用距离权重插值方法

$$\phi_f = \frac{\phi_1/d_1 + \phi_2/d_2}{1/d_1 + 1/d_2}$$



插值另外一个公式

平方插值

$$\phi_f = \frac{\phi_1/d_1^2 + \phi_2/d_2^2}{1/d_1^2 + 1/d_2^2}$$

插值函数:

$$\phi_f = \frac{1/d_1}{1/d_1 + 1/d_2}\phi_1 + (1 - \frac{1/d_1}{1/d_1 + 1/d_2})\phi_2$$

则插值函数为:

$$w_f = \frac{1/d_1}{1/d_1 + 1/d_2}$$

$$\phi_f = w_f\phi_1 + (1 - w_f)\phi_2$$

插值函数的处理

- 计算 w_f 只需要几何信息。d_1, d_2 很容易通过坐标计算出来。
- 可以在前处理阶段将插值函数存储下来。节约大量计算时间。

体到节点插值

体心到节点插值函数

$$w_{v,i} = \frac{1/d_i}{\sum_{i=1}^{N} 1/d_i}$$

则节点值计算:

$$\phi_v = \frac{\phi_i/d_i}{\sum_{i=1}^{N} 1/d_i} = \sum_{i=1}^{N} w_{v,i}\phi_i$$

体积计算

假设向量 $\mathbf{q} = x\hat{\mathbf{i}}$。

$$\nabla \cdot \mathbf{q} = \nabla \cdot (x\hat{\mathbf{i}}) = 1$$

所以可以:

$$V_O = \int_{V_O} dV = \int_{V_O} \nabla \cdot \mathbf{q} dV$$

使用散度定理:

$$V_O = \int_{V_O} \nabla \cdot \mathbf{q} dV = \int_S \mathbf{q} \cdot \hat{\mathbf{n}} dA = \int_S x\hat{\mathbf{i}} \cdot \hat{\mathbf{n}} dA$$

$$V_O = \sum_{f=1}^{N_{f,O}} n_{x,f} \int_{S_f} x dA = \sum_{f=1}^{N_{f,O}} n_{x,f} x_f A_f$$

体积计算公式

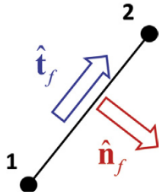
为了防止高度奇异，skewness

$$V_O = \frac{1}{2}\left(\sum_{f=1}^{N_{f,O}} n_{x,f}x_f A_f + \sum_{f=1}^{N_{f,O}} n_{y,f}y_f A_f\right)$$

$$V_O = \frac{1}{3}\left(\sum_{f=1}^{N_{f,O}} n_{x,f}x_f A_f + \sum_{f=1}^{N_{f,O}} n_{y,f}y_f A_f + \sum_{f=1}^{N_{f,O}} n_{z,f}z_f A_f\right)$$

面积计算和法向量

通常网格工具会给出面积和法向量等信息。



面法向可以通过面切向计算。

$$t_{x,f} = \frac{x_2 - x_1}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} = \frac{x_2 - x_1}{A_f}$$

$$t_{y,f} = \frac{y_2 - y_1}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} = \frac{y_2 - y_1}{A_f}$$

二维

两个关系:

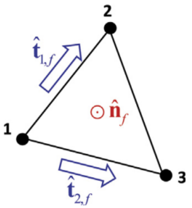
$$\widehat{\mathbf{n}}_f \cdot \widehat{\mathbf{t}}_f = 0$$

$$\widehat{\mathbf{n}}_f \times \widehat{\mathbf{t}}_f = \widehat{\mathbf{k}}$$

可以求得:$n_{y,f} = -t_{x,f}, n_{x,f} = t_{y,f}$

三维

三维图形:



$$A_f = A_{123} = \frac{1}{2}|\mathbf{t_1} \times \mathbf{t_2}| = \frac{1}{2}|n|$$

$$\mathbf{n} = \begin{bmatrix} \widehat{\mathbf{i}} & \widehat{\mathbf{j}} & \widehat{\mathbf{k}} \\ \widehat{\mathbf{t}}_{1x} & \widehat{\mathbf{t}}_{1y} & \widehat{\mathbf{t}}_{1z} \\ \widehat{\mathbf{t}}_{2x} & \widehat{\mathbf{t}}_{2y} & \widehat{\mathbf{t}}_{2z} \end{bmatrix}$$

计算每个面上的法线方向

```
for icell = 1: ncells
    for ifc = 1 : nface(icell)
        gface = link_cell_to_face(icell, ifc)
        ic1 = link_face_to_cell(gface, 1)
        if (icell = ic1)
            snsign(icell, ifc) = 1
        else
            sngign(icell, ifc) = -1
        endif
    end
end
```

计算每个单元的体积

```
for icell = 1: ncells
    volume(icell) = 0
    for ifc = 1 : nface(icell)
        volume(icell) = volume(icell) + sn(
            gface, 1) * snsign(icell, ifc) * xf(
            gface) * areaf(gface)
    end
end
```

OpenFOAM