二维定常湍流计算中的 GMRES 算法 [1)]

宁方飞 徐力平
(北京航空航天大学 404 教研室，北京 100083)

摘要 在以前工作的基础上将广义极小残差 (Generalized Minimum RESidual (GMRES)) 算法发展到用于求解二维可压 Favier 平均 Navier-Stokes 方程组. 控制方程经 Newton 线化处理后构成近似的线性系统，然后采用分别耦合了 LUSGS 和 ILU 两种预处理矩阵的 GMRES 算法求解. Spalart-Allmaras 湍流模型被用来封闭流体控制方程组，采用与流体控制方程非耦合的方式，使用 LUSGS 方法求解. 对 GMRES 算法中矩阵 - 向量的乘积采用了有限插分方法，从而避免了精确的左端系数矩阵的计算和存储. 对预处理矩阵的两种使用方法 (左预处理和右预处理) 进行了分析和讨论. 用两个算例对 LUSGS 和 ILU 两种预处理矩阵进行了比较，同时探讨了左预处理和右预处理各自的优缺点. 通过对 Sajben 扩压器和 NACA0012 有攻角流动的计算，表明带有预处理的 GMRES 算法在二维定常跨音黏性流动计算中相比于得到广泛应用的 DDADI 方法具有很大优势，左预处理要优于右预处理.

关键词 Navier-Stokes 方程，广义极小残差算法，预处理， LUSGS 预处理， ILU 预处理

引 言

近二十年来计算机的发展以及人们对求解复杂流动的需求使得计算流体力学得到了长足进展. 其中，对流场的求解效率一直是人们所关注的问题. 一种特别成功的算法是 Newton 迭代，它具有平方敛速 [1]，大大优于其它标准的线性敛速的方法. 对由 Newton 迭代所形成的线性方程组的求解方法基本上有三类：直接解法、迭代方法和共轭梯度法. 其中最有吸引力的是共轭梯度法. 这类方法不需要求解左端系数矩阵的逆，并且在精确求解时只需 N 步就可得到方程的解 (N 为方程的数目)，所以在节省内存的同时，又有很高的效率. 共轭梯度法中，比较有代表性的有： Sonneveld 的共轭梯度平方 (Conjugate Gradient Squared (CGS)) 算法 [2]; Saad 和 Schultz 的广义极小残差 (Generalized Minimum RESidual (GMRES)) 算法 [3] 等，并在流体计算中得到了广泛的应用. 对于 GMRES 算法， Saad 和 Schultz[3] 提出了用于非对称线性系统的求解格式. 之后， Venkatakrishnan 将 GMRES 算法应用于可压缩 Navier-Stokes 方程的求解 [4]; Ajmani 等将 GMRES 算法应用于二维 Navier-Stokes 方程的求解 [5]，并对算法的 Krylov 子空间的数目进行了优化； Nielsen 和 Anderson 将 GMRES 算法应用于非结构网格三维 Euler 方程的求解 [6]，并讨论了矩阵 - 向量乘积的精确解法和有限插分法； Hong 等将 GMRES 算法应用于非结构网格三维 Euler 方程及 Navier-Stokes 方程的求解 [7]. 本文作者在早期工作中 [8]，采用 GMRES 算法求解了 Euler 方程，并且对 LUSGS 和 ILU 两种预处理矩阵进行了比较. 本文进一步将 GMRES 算法发展为用于求解二维可压 Favre 平均 Navier-Stokes 方程，并且对左预处理和右预处理进行了分析，同时对 LUSGS 和 ILU 两种预处理矩阵进行了比较. Spalart-Allmaras

2000-03-05 收到第一稿， 2000-08-04 收到修改稿.
1) 国家自然科学基金资助项目 (59525612).

湍流模型 [9] 被用来封闭流体控制方程组，采用与流体控制方程非耦合的方式，使用 LUSGS 方法求解. 经过对二维 Navier-Stokes 方程的求解的对比计算，表明 GMRES 算法在湍流流场计算中，在相当程度上仍然保持了它在求解无黏流场时的效率. 从计算结果中还可以看到，左预处理要优于右预处理，GMRES 算法中采用基于左预处理的两种预处理矩阵基本上有相同的效果.

1 控制方程的离散

1.1 空间离散

采用有限体积方法对控制方程进行空间离散后成为

$$\frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{U}\cdot\Omega)_{i,j}+(\hat{\boldsymbol{F}}_{i+1/2,j}-\hat{\boldsymbol{F}}_{\nu i+1/2,j})-(\hat{\boldsymbol{F}}_{i-1/2,j}-\hat{\boldsymbol{F}}_{\nu i-1/2,j})+$$
$$(\hat{\boldsymbol{G}}_{i,j+1/2}-\hat{\boldsymbol{G}}_{\nu i,j+1/2})-(\hat{\boldsymbol{G}}_{i,j-1/2}-\hat{\boldsymbol{G}}_{\nu i,j-1/2})=0 \tag{1}$$

其中，$\boldsymbol{U}$ 为守恒变量；Ω 为控制单元面积；$\hat{\boldsymbol{F}},\hat{\boldsymbol{F}}_\nu,\hat{\boldsymbol{G}},\hat{\boldsymbol{G}}_\nu$ 分别为广义坐标系下 ξ,η 方向上的对流通量和扩散通量. 在本文中，扩散通量采用中心差分进行离散，而对流数值通量的求解采用 Roe 的通量差分分裂 (FDS) 格式 [10]

$$\hat{\boldsymbol{F}}_{i+1/2,j}=\frac{1}{2}[\hat{\boldsymbol{F}}(\boldsymbol{U}_L)+\hat{\boldsymbol{F}}(\boldsymbol{U}_R)-|\widetilde{\boldsymbol{A}}(\boldsymbol{U}_L,\boldsymbol{U}_R)|\cdot(\boldsymbol{U}_R-\boldsymbol{U}_L)] \tag{2}$$

其中，$\boldsymbol{U}_L,\boldsymbol{U}_R$ 分别为单元表面 $(i+1/2,j)$ 左右两边的守恒变量；$\widetilde{\boldsymbol{A}}(\boldsymbol{U}_L,\boldsymbol{U}_R)$ 为满足 U 特性的 Roe 平均 Jacobi 系数矩阵. 为了获得空间的高阶精度，单元表面左右两边的流场变量用 MUSCL 公式求取，并采用 Van Albada 限制器来防止解在激波附近的过冲或过膨胀. 另外，为满足熵条件，采用 Hartan 的熵修正方法.

1.2 时间离散

采用 Euler 后差隐式时间离散方法，则 (1) 式成为

$$\left(\frac{\Omega_{i,j}}{\Delta t}\boldsymbol{I}-\frac{\partial\hat{\boldsymbol{R}}^n_{i,j}}{\partial\boldsymbol{U}}\right)\cdot\delta\boldsymbol{U}=\hat{\boldsymbol{R}}^n_{i,j} \tag{3}$$

其中上标代表时间层，$\delta\boldsymbol{U}=\boldsymbol{U}^{n+1}-\boldsymbol{U}^n$, $\boldsymbol{I}$ 为单位矩阵，$\hat{\boldsymbol{R}}$ 为残差向量

$$\hat{\boldsymbol{R}}_{i,j}=-(\hat{\boldsymbol{F}}_{i+1/2,j}-\hat{\boldsymbol{F}}_{\nu i+1/2,j})+(\hat{\boldsymbol{F}}_{i-1/2,j}-\hat{\boldsymbol{F}}_{\nu i-1/2,j})-(\hat{\boldsymbol{G}}_{i,j+1/2}-\hat{\boldsymbol{G}}_{\nu i,j+1/2})+$$
$$(\hat{\boldsymbol{G}}_{i,j-1/2}-\hat{\boldsymbol{G}}_{\nu i,j-1/2}) \tag{4}$$

关于残差向量的 Jacobian 矩阵可以写为

$$\frac{\partial\hat{\boldsymbol{R}}^n_{i,j}}{\partial\boldsymbol{U}}=-\frac{\partial\left(\hat{\boldsymbol{F}}^n_{i+1/2,j}-\hat{\boldsymbol{F}}^n_{\nu i+1/2,j}\right)}{\partial\boldsymbol{U}}+\frac{\partial\left(\hat{\boldsymbol{F}}^n_{i-1/2,j}-\hat{\boldsymbol{F}}^n_{\nu i-1/2,j}\right)}{\partial\boldsymbol{U}}-$$
$$\frac{\partial\left(\hat{\boldsymbol{G}}^n_{i,j+1/2}-\hat{\boldsymbol{G}}^n_{\nu i,j+1/2}\right)}{\partial\boldsymbol{U}}+\frac{\partial\left(\hat{\boldsymbol{G}}^n_{i,j-1/2}-\hat{\boldsymbol{G}}^n_{\nu i,j-1/2}\right)}{\partial\boldsymbol{U}} \tag{5}$$

对于每个控制单元，都可写出如 (3) 式的控制方程，将这些控制方程组合在一起，则可以写为如下矩阵方程组的形式

$$\boldsymbol{A}\cdot\boldsymbol{X}=\boldsymbol{B} \tag{6}$$

其中，$\boldsymbol{A}$ 为 $N\times N$ 阶系数矩阵，这里 N 为计算域中的控制单元数. 对于二维问题，矩阵 $\boldsymbol{A}$ 中每一个元素都是一个 4×4 阶矩阵. $\boldsymbol{X}$ 为解向量 $[\delta\boldsymbol{U}_1,\delta\boldsymbol{U}_2,\cdots,\delta\boldsymbol{U}_{N-1},\delta\boldsymbol{U}_N]$, $\boldsymbol{B}$ 为残差向量 $[\hat{\boldsymbol{R}}_1,\hat{\boldsymbol{R}}_2,\cdots,\hat{\boldsymbol{R}}_{N-1},\hat{\boldsymbol{R}}_N]$. 对于每个时间步，方程 (3) 的左端系数矩阵以及右端的残差向量都为已知，即 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都已知，所以方程组 (6) 成为一个线性系统.

2 GMRES 算法

GMRES 算法是以 Galerkin 原理为基础的一种算法. 对于方程组 (6), GMRES 算法为以下过程 [3]:

(1) 给定线性系统 (6) 解的初始猜值 $\boldsymbol{X}_0$, 然后计算

$$\boldsymbol{r}_0=\boldsymbol{B}-\boldsymbol{A}\cdot\boldsymbol{X}_0 \tag{7}$$

$$\boldsymbol{r}_1=\boldsymbol{M}_L^{-1}\cdot\boldsymbol{B} \tag{8}$$

$$\boldsymbol{v}_1=\frac{\boldsymbol{r}_1}{\|\boldsymbol{r}_1\|} \tag{9}$$

(2) $m=1,2,\cdots,k$ 迭代

$$\boldsymbol{w}=\boldsymbol{M}_L^{-1}\cdot\boldsymbol{A}\cdot\boldsymbol{M}_R^{-1}\cdot\boldsymbol{v}_m \tag{10}$$

$$h_{l,m}=(\boldsymbol{w},\boldsymbol{v}_l),\quad l=1,2,\cdots,m \tag{11}$$

$$\widetilde{\boldsymbol{v}}_{m+1}=\boldsymbol{w}-\sum_{l=1}^{m}h_{l,m}\boldsymbol{v}_l \tag{12}$$

$$h_{m+1,m}=\|\widetilde{\boldsymbol{v}}_{m+1}\| \tag{13}$$

$$\boldsymbol{v}_{m+1}=\frac{\widetilde{\boldsymbol{v}}_{m+1}}{h_{m+1,m}} \tag{14}$$

如果 $\min\limits_{\boldsymbol{y}}\|\beta\cdot\boldsymbol{e}_1-\overline{\boldsymbol{H}}_m\boldsymbol{y}\|/\beta\le\delta$, 则结束迭代.

(3) 求方程 (6) 的近似解

$$\boldsymbol{X}_k=\boldsymbol{M}_R^{-1}\cdot(\boldsymbol{X}_0+\boldsymbol{V}_k\boldsymbol{y}_k) \tag{15}$$

$\boldsymbol{y}_k$ 为以下函数的极小值

$$J(\boldsymbol{y})=\|\beta\cdot\boldsymbol{e}_1-\overline{\boldsymbol{H}}_k\boldsymbol{y}\| \tag{16}$$

首先说明 GMRES 算法过程中各符号的含义. $\boldsymbol{r}_0$, $\boldsymbol{r}_1$, $\boldsymbol{v}_m$ $(m=1,k)$ 和 $\boldsymbol{w}$ 是 $4N$ 阶向量. $\boldsymbol{M}_L$ 和 $\boldsymbol{M}_R$ 分别为左、右预处理矩阵. $\|\cdot\|$ 为向量的 L-2 模, $(\boldsymbol{v}_1,\boldsymbol{v}_2)$ 为向量的内积. (15) 式中 $\boldsymbol{V}_k$ 是由列向量 $\boldsymbol{v}_m$ $(m=1,k)$ 构成的 $k\times 4N$ 阶矩阵, $\boldsymbol{y}$ 为 k 阶向量. 在 (16) 式中, β 为初始残差的 L-2 模 $\beta\equiv\|r_0\|$; $\boldsymbol{e}_1$ 为 $(k+1)\times(k+1)$ 单位矩阵的第一列，即 $[1,0,0,\cdots,0,0]^{\mathrm{T}}$; $\overline{\boldsymbol{H}}_k$ 为一个上 Hessenberg 矩阵加一行向量，此行向量唯一的非零元素为 $h_{k+1,k}$, 位于 $(k+1,k)$.

在 (10) 式中需要求解系数矩阵与向量的乘积 $\boldsymbol{A}\cdot\boldsymbol{u}$. 实际上，由于数值通量计算的复杂性，方程 (6) 左端的系数矩阵相当复杂或不可求. 在本文中，采用了一种不需要给出精确的系数矩阵 $\boldsymbol{A}$ 的矩阵 - 向量乘积的差分近似方法 [6]

$$\boldsymbol{A}\boldsymbol{u}\equiv\frac{\boldsymbol{R}(\boldsymbol{U}+\varepsilon\boldsymbol{u})-\boldsymbol{R}(\boldsymbol{U})}{\varepsilon} \tag{17}$$

其中 $\boldsymbol{R}(\boldsymbol{U}+\varepsilon\boldsymbol{u})$ 为守恒变量 $\boldsymbol{U}$ 加上小扰动量后的残差向量，这里 ε 为一个代数值. 在文献 [6] 中指出，采用如下对 ε 的定义，用 (17) 式计算矩阵 - 向量乘积与精确计算其乘积的效果基本相同

$$\varepsilon\overline{\boldsymbol{u}} \equiv \sqrt{\varepsilon_{\text{machine}}} \tag{18}$$

其中 $\varepsilon_{\text{machine}}$ 是由硬件决定的所谓"机器零"，在本文中取为 10^{-14}. $\overline{\boldsymbol{u}}$ 为向量 $\boldsymbol{u}$ 中元素的均方根值.

3 预处理

对于形如方程组 (6) 的系统，如果左端系数矩阵的条件数较大，方程组是病态的，则 GMRES 算法将没有高速收敛的特性. 要有高的效率，必须对方程组 (6) 采用适当的预处理方法.

如果采用左预处理，设预处理矩阵为 $\boldsymbol{M}_L$，在系统 (6) 的左右两端分别乘以预处理矩阵的逆矩阵 $\boldsymbol{M}_L^{-1}$，成为

$$\boldsymbol{M}_L^{-1}\boldsymbol{A}\cdot\boldsymbol{X}=\boldsymbol{M}_L^{-1}\boldsymbol{B}\Leftrightarrow\widetilde{\boldsymbol{A}}\cdot\boldsymbol{X}=\widetilde{\boldsymbol{B}} \tag{19}$$

求解系统 (6) 现在改为求解左预处理后的系统 (19). 这时， GMRES 计算过程 (7)~(15) 式中的右预处理矩阵 $\boldsymbol{M}_R$ 相应地取为单位矩阵.

如果采用右预处理，设预处理矩阵为 $\boldsymbol{M}_R$，可以将系统 (6) 改写为

$$\left(\boldsymbol{A}\cdot\boldsymbol{M}_R^{-1}\right)\cdot\left(\boldsymbol{M}_R\cdot\boldsymbol{X}\right)=\boldsymbol{B}\Leftrightarrow\widetilde{\boldsymbol{A}}\cdot\widetilde{\boldsymbol{X}}=\boldsymbol{B} \tag{20}$$

求解系统 (6) 现在改为求解右预处理后的系统 (20). 这时， GMRES 计算过程 (7)~(15) 式中的左预处理矩阵 $\boldsymbol{M}_L$ 相应地取为单位矩阵.

在本质上，左预处理和右预处理是相同的，只是左预处理会改变线性系统残差向量的值，从而对 GMRES 内迭代的收敛过程产生影响.

在实际应用中， $\boldsymbol{M}^{-1}$ 并不直接求解. 实际上， $\boldsymbol{M}^{-1}$ 与一个向量 $\boldsymbol{\ell}$ 的乘积与求解如下方程组

$$\boldsymbol{M}\widetilde{\boldsymbol{\ell}}=\boldsymbol{\ell} \tag{21}$$

是等价的. 如果预处理矩阵 $\boldsymbol{M}$ 对改善系统 (6) 的性态有较大作用，并且方程组 (21) 的可解性较好，则此预处理就有可能可以使用在 GMRES 算法中，大大提高 GMRES 算法的效率.

下面将介绍本文所使用的预处理矩阵.

将方程 (5) 中的对流数值通量简化为一阶空间精度 (方程 (3) 右端的残差向量中的数值通量仍采用高阶精度求解)，以 $\hat{\boldsymbol{F}}^n_{i+1/2,j}$ 为例，则有

$$\hat{\boldsymbol{F}}^n_{i+1/2,j}=\frac{1}{2}\left[\hat{\boldsymbol{F}}^n_{i,j}+\hat{\boldsymbol{F}}^n_{i+1,j}-|\boldsymbol{A}(\boldsymbol{U}_{i,j},\boldsymbol{U}_{i+1,j})|^n\cdot(\boldsymbol{U}_{i+1,j}-\boldsymbol{U}_{i,j})^n\right] \tag{22}$$

其中，上标 n 代表时间层. 将上式代入到 (5) 式中，可以得到无黏通量的 Jacobian 矩阵.

关于扩散通量的 Jacobian 矩阵，在许多隐式格式中，都将它简化为一个主对角矩阵，以提高格式的稳定性. 但是计算表明，这种方法应用在 GMRES 算法中时，会降低算法的效率. 所以其 Jacobian 矩阵需要根据扩散通量的离散形式具体进行求解.

最后将 (5) 式代入到方程 (3) 的左端系数矩阵中，求解左端系数矩阵，略去中间推导过程，可得如下五对角矩阵

$$\widetilde{\boldsymbol{M}}=[\boldsymbol{A}_1,0,\cdots,0,\boldsymbol{A}_2,\boldsymbol{A}_3,\boldsymbol{A}_4,0,\cdots,0,\boldsymbol{A}_5] \tag{23}$$

各对角线上的元素分别为

$$\left.\begin{aligned}
&\boldsymbol{a}_1=-\frac{1}{2}(\hat{\boldsymbol{A}}_{i-1,j}-\widetilde{\boldsymbol{A}}^L_{i-\frac{1}{2},j})+\widetilde{A}^L_{\nu i-1/2,j},\quad a_2=-\frac{1}{2}(\hat{\boldsymbol{B}}_{i,j-1}-\widetilde{\boldsymbol{B}}^L_{i,j-1/2})+\widetilde{\boldsymbol{B}}^L_{\nu i,j-1/2}\\
&\boldsymbol{a}_3=\frac{\Omega_{i,j}}{\Delta t}\boldsymbol{I}-\frac{1}{2}(\widetilde{\boldsymbol{A}}^L_{i+1/2,j}-\widetilde{\boldsymbol{A}}^R_{i-1/2,j}+\widetilde{\boldsymbol{B}}^L_{i,j+1/2}-\widetilde{\boldsymbol{B}}^R_{i,j-1/2})-\\
&\qquad\widetilde{\boldsymbol{A}}^L_{\nu i+1/2,j}+\widetilde{\boldsymbol{A}}^R_{\nu i-1/2,j}-\widetilde{\boldsymbol{B}}^L_{\nu i,j+1/2}+\widetilde{\boldsymbol{B}}^R_{\nu i,j-1/2}\\
&\boldsymbol{a}_4=\frac{1}{2}(\hat{\boldsymbol{B}}_{i,j+1}-\widetilde{\boldsymbol{B}}^R_{i,j+1/2})-\widetilde{\boldsymbol{B}}^R_{\nu i,j+1/2},\quad \boldsymbol{a}_5=\frac{1}{2}(\hat{\boldsymbol{A}}_{i+1,j}-\widetilde{\boldsymbol{A}}^R_{i+1/2,j})-\widetilde{\boldsymbol{A}}^R_{\nu i+1/2,j}
\end{aligned}\right\}\tag{24}$$

其中 $\hat{\boldsymbol{A}}_{i,j}$ 和 $\hat{\boldsymbol{B}}_{i,j}$ 分别为广义坐标系下 ξ 方向和 η 方向无黏通量的 Jacobian 矩阵； $\widetilde{\boldsymbol{A}}^{L,R}_{\nu i+1/2,j}$ 和 $\widetilde{\boldsymbol{B}}^{L,R}_{\nu i,j+1/2}$ 分别为 ξ 方向和 η 方向扩散通量的 Jacobian 矩阵； $\widetilde{\boldsymbol{A}}^{L,R}_{i+1/2,j}$ 和 $\widetilde{\boldsymbol{B}}^{L,R}_{i,j+1/2}$ 则分别为 ξ 方向和 η 方向单元表面 FDS 无黏数值通量中人工耗散部分的 Jacobian 矩阵

$$\left.\begin{aligned}
\widetilde{\boldsymbol{A}}^L_{i+1/2,j}=\partial[|\widetilde{\boldsymbol{A}}|\cdot(\boldsymbol{U}_{i+1,j}-\boldsymbol{U}_{i,j})]/\partial\boldsymbol{U}_{i,j}\\
\widetilde{\boldsymbol{A}}^R_{i+1/2,j}=\partial[|\widetilde{\boldsymbol{A}}|\cdot(\boldsymbol{U}_{i+1,j}-\boldsymbol{U}_{i,j})]/\partial\boldsymbol{U}_{i+1,j}
\end{aligned}\right\}\tag{25}$$

$$\left.\begin{aligned}
\widetilde{\boldsymbol{B}}^L_{i,j+1/2}=\partial[|\widetilde{\boldsymbol{B}}|\cdot(\boldsymbol{U}_{i,j+1}-\boldsymbol{U}_{i,j})]/\partial\boldsymbol{U}_{i,j}\\
\widetilde{\boldsymbol{B}}^R_{i,j+1/2}=\partial[|\widetilde{\boldsymbol{B}}|\cdot(\boldsymbol{U}_{i,j+1}-\boldsymbol{U}_{i,j})]/\partial\boldsymbol{U}_{i,j+1}
\end{aligned}\right\}\tag{26}$$

其中各式右端中的 $\widetilde{\boldsymbol{A}}$ 和 $\widetilde{\boldsymbol{B}}$ 分别为 ξ 方向和 η 方向上的 Roe 平均矩阵. 对于 $\widetilde{\boldsymbol{A}}^{L,R}_{i+1/2,j}$ 和 $\widetilde{\boldsymbol{B}}^{L,R}_{i,j+1/2}$，可以根据 FDS 数值通量的求解公式采用链式法则进行求解.

为使系统 (21) 容易求解，还需要对矩阵 $\widetilde{\boldsymbol{M}}$ 做进一步的近似. 本文在以构造成的左端系数矩阵 $\widetilde{\boldsymbol{M}}$ 的基础上，采用了以下两种不同的预处理矩阵.

(1) LUSGS 预处理矩阵

将矩阵 (23) 分解为 $\widetilde{\boldsymbol{M}}=\boldsymbol{L}+\boldsymbol{D}+\boldsymbol{U}$, 其中 $\boldsymbol{L}$ 为不包含主对角元素的下三角阵； $\boldsymbol{D}$ 为对角阵； $\boldsymbol{U}$ 为不包含主对角元素的上三角阵. 然后对矩阵进行如下因子化

$$\boldsymbol{M}=(\boldsymbol{L}+\boldsymbol{D})\cdot\boldsymbol{D}^{-1}\cdot(\boldsymbol{D}+\boldsymbol{U})\tag{27}$$

将它作为预处理矩阵，则方程组 (21) 的求解可由两步完成

$$(\boldsymbol{L}+\boldsymbol{D})\cdot\widetilde{\boldsymbol{l}}^*=\boldsymbol{l}\tag{28}$$

$$(\boldsymbol{D}+\boldsymbol{U})\cdot\widetilde{\boldsymbol{l}}=\boldsymbol{D}\cdot\widetilde{\boldsymbol{l}}^*\tag{29}$$

与 Jameson 提出的 LUSGS 方法的求解过程相同 [11].

(2) ILU 预处理矩阵

这里的 ILU (Incomplete Lower-Upper Decompsition) 未完全 LU 分解在对矩阵 $\widetilde{\boldsymbol{M}}$ 做不完全 Crout 分解后，不会在矩阵中增加新的非零元素，这也被称为所谓的 ILU (0) 方法 [12]. 具体方法如下：

将简化的左端系数矩阵 $\widetilde{\boldsymbol{M}}$ 分解为

$$\widetilde{\boldsymbol{M}}=\boldsymbol{M}+\boldsymbol{R}\tag{30}$$

其中 M 为 $\widetilde{M}$ 的不完全 Crout 分解， $M=LD^{-1}U$, L 为主对角元素不为零的下三角阵； D 为对角阵； U 为主对角元素不为零的上三角阵； R 为分解误差. 矩阵 L,D 及 U 满足以下条件:

(i) 矩阵 L 与矩阵 U 的主对角元素都等于 D 的相应元素;

(ii) L 和 U 的非主对角元素等于对应的 $\widetilde{M}$ 中的元素;

(iii) $\mathrm{diag}\,(LD^{-1}U)=\mathrm{diag}\,(A)$.

由以上条件，容易求得对角阵 D 的元素为

$$d_i=a_{i,3}-a_{i,2}d_{i-1}^{-1}a_{i-1,4}-a_{i,1}d_{i-m}^{-1}a_{i-m,5} \tag{31}$$

将矩阵 $M=LD^{-1}U$ 作为预处理矩阵，同样可以采用如 (28) 和 (29) 式的两步方法来求解方程组 (21).

4 内存需求及时间步

GMRES 算法的内存需求量相对来说较大. 首先，需要保存预处理矩阵，对于二维问题，其矩阵元素的数量为 $5\times I_{\max}\times J_{\max}$，而每个矩阵元素是一个 4×4 矩阵，其中 $I_{\max}$ 和 $J_{\max}$ 分别是 I 方向和 J 方向的单元数. 其次，需要保存初始残差向量 r_0 以及向量 v_m $(m=1,k)$，对于二维问题，它们是元素个数为 $4\times I_{\max}\times J_{\max}$ 的向量. 无黏计算和黏性计算在内存需求上基本相同. 实际上，对流通量和扩散通量的 Jacobian 矩阵可以保存在同一个数组中，所以黏性计算需要增加的内存只限于扩散通量计算中的内存需求.

在 GMRES 算法中，迭代的次数越大，则相应的线性问题的近似解越精确. 但是如果迭代的次数过多，则向量 v_m 的个数会很多，内存需求量是相当大的. 由于我们针对的是 Navier-Stokes 方程的定常解，在中间时间步上我们并不关心解的精确性，所以没有必要在每一时间步上做过多的迭代次数. 所以在本文中，将迭代步数设为 5. 另外还采用了当地时间步长方法.

如果当前的流场与最终的流场解相差很大，则大的 CFL 数可能会使 GMRES 计算中断. 在本文中， CFL 数是与残差的大小成反比的

$$CFL=CFL_{\min}\times\left(\frac{RES_0}{RES_n}\right) \tag{32}$$

其中 $CFL_{\min}$ 为初始 CFL 数，这里取为 10; RES_0 和 RES_n 分别为流场的初始残差和当前第 n 时间步的残差的 L-2 模.

另外，在数值实验中还发现，如果将 GMRES 迭代过程中的收敛公差 δ 取为常值，则收敛速度或者会变慢，或者计算会发散. 试验表明，收敛公差在流场与最终流场解相差较大时应比较“宽松”，即公差值较大；而在时间的推进过程中，可以逐渐减小公差值，这样才能使 GMRES 算法有高的效率. 从另一个角度上分析，因为采用了当地时间步长方法，如果在大致的流场结构还没有形成时就使用较小的公差值，则不同控制单元的流场解在时间方向上推进的速度是不一致的，这就有可能会导致计算的发散. 基于以上分析， GMRES 迭代过程中的收敛公差 δ 应与流场残差成一定的比例. 而 CFL 数正反映了流场残差的演化，所以对应于 CFL 数的变化，在 GMRES 迭代过程 (2) 中的收敛准则 δ 采用文献 [5] 中的方法，取为

$$\delta=\frac{1}{\log_{10}(CFL^2)} \tag{33}$$

以上的处理方法与无黏计算相同.

5 计算结果与比较

这里用两个算例对 GMRES 算法以及两种预处理矩阵进行了分析和验证，同时与常用的主对角占优的近似因子分解 (DDADI) 方法进行了比较. 对于 DDADI 方法，对 (5) 式中无黏通量的 Jacobian 矩阵用如下系数矩阵分裂来近似

$$\hat{\boldsymbol{A}}^{\pm} = \hat{\boldsymbol{R}}^{-1}\hat{\boldsymbol{\Lambda}}^{\pm}\hat{\boldsymbol{R}}, \quad \hat{\boldsymbol{\Lambda}}_i^{\pm} = \frac{(\hat{\lambda}_i \pm |\hat{\lambda}_i|)}{2} \tag{34}$$

其中，$\hat{\boldsymbol{R}}$ 为由 Jacobian 矩阵 $\hat{\boldsymbol{A}}$ 的左特征行向量构成的特征矩阵，$\hat{\boldsymbol{\Lambda}}^{\pm}$ 为对角矩阵，$\hat{\lambda}_i$ $(i=1,4)$ 为矩阵 $\hat{\boldsymbol{A}}$ 的特征值. 而黏性通量的 Jacobian 矩阵只加到系数矩阵的主对角矩阵元素上，以增加系数矩阵的主对角占优性. 另外在 DDADI 方法中也采用了当地时间步长方法，并且在下面的计算中 CFL 数取为 10.

对方程中湍流黏性的计算这里采用了 Spalart-Allmaras[9] 一方程模型. 由于本文采用了有限体积方法离散 Navier-Stokes 方程，所以这里将原始的模型方程转化为了守恒形式，然后进行离散求解.

5.1 Sajben 扩压器流动

Sajben 扩压器流动的实验结果见文献 [13]. 本文采用壁面正交网格，网格数为 161×61. 扩压器下壁面为平板，上壁面为一光滑曲线，喉道高度为 44 mm, 出口与喉道的面积比为 1.5. 计算工况是：入口总温为 292 K, 总压 135 000 Pa; 出口静压与入口总压之比为 0.735. 在此工况下扩压器上壁面将出现激波诱导的边界层分离 [13]. 边界条件的给法为：入口给定总温、总压以及气流方向为 X 方向，气流速度大小外推；出口给定静压，其它变量外推；固体表面采用无滑移条件，密度和压力一阶外推. 初场给为零速度场. 湍流模型方程中的涡黏性函数的边界条件为：入口给为分子黏性的 1/10, 出口外推，在壁面上取为 0; 另外将转捩点取在位于入口处的上、下壁面处. 涡黏性函数的初场取为其值等于入口值的均匀场.

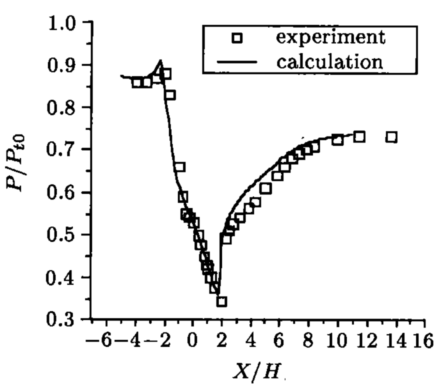

图 1 Sajben 扩压器上表面压力分布

Fig.1 Pressure distributions at upper surface of Sajben's diffuser

扩压器上表面压力沿流向的分布见图 1. 从图中可以看到，压力分布与实验基本吻合，只是在入口部分压力有一点过冲，另外在激波后的分离区部分有一些差异，这与湍流模型的准确性有关. 图 2 和图 3 则分别是残差的计算步数收敛史和 CPU 时间收敛史，图中说明部分中的 “-L” 和 “-R” 分别代表左预处理和右预处理. 由图 2 和图 3 可以看到，无论在计算步数上还是在 CPU 时间上，GMRES 算法的几种形式的收敛速度都大大优于 DDADI 方法. 另外从图 2 还可以看到，在计算步数上，采用右预处理的两种预处理矩阵都比左预处理下的两种预处理矩阵快. 这是因为右预处理不影响系统 (6) 的残差向量值，而左预处理

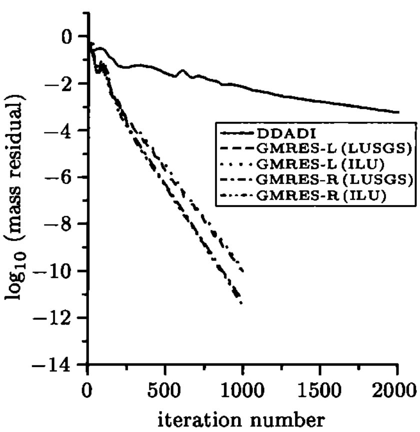

图 2 扩压器流场计算步数收敛史

Fig.2 Residual history for the iteration number of diffuser flow

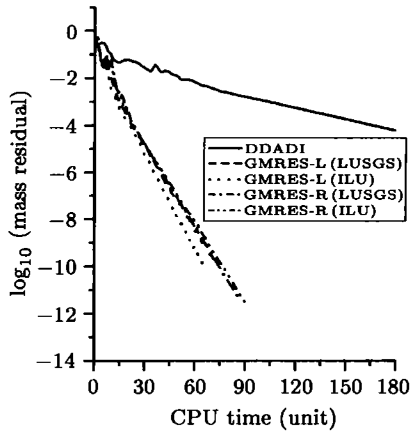

图 3 扩压器流场 CPU 时间收敛史

Fig.3 Residual history for CPU time of diffuser flow

在改变了系统的残差向量值的同时，引入了预处理矩阵的分解误差而造成的. 但是对于 CPU 时间却不是这样，见图 3. 这是因为采用右预处理时，在 GMRES 算法的内迭代中需要更多的计算量. 由于我们关心的是计算所占用 CPU 的时间，由图 3 可以看到，基于左预处理的 ILU 预处理矩阵具有最优的收敛速度.

5.2 NACA0012 有攻角跨音流动

采用 C 型网格，网格数取为 201×71. 绕流 NACA0012 翼型的来流工况是：来流马赫数为 0.799, 来流攻角为 2.26°. 边界条件的给法为：对于计算域的外边界，根据边界线的方向区分入口边界和出口边界. 对于入口边界，给定总温、总压和速度方向，外插速度大小；对于出口边界，给定静压，其它变量外插得到. 对于计算域的内边界，有一部分是在翼型表面上，其余的部分相互重合. 在翼型表面上的边界处给定无滑移条件，密度和压力外插得到. 对于重合的边界，因为本文采用的是网格中心方法，所以将与当前边界相对的两层单元的流场变量作为对应虚网格单元的流场变量，这里用两层虚网格可以保证在边界处仍有二阶空间精度. 初场给为来流工况下的均匀场. 湍流模型方程中的涡黏性函数的边界条件给法与扩压器流动计算中的方法类似.

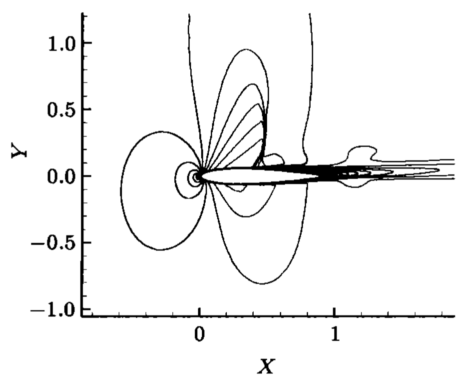

图 4 NACA0012 等马赫线分布

Fig.4 Mach contour of NACA0012 flow

NACA0012 有攻角超临界流动的等马赫线分布见图 4. 图 5 和图 6 分别是残差的计算步数收敛史和 CPU 时间收敛史. 可以看到，GMRES 算法仍然保持了很高的优势. 另外，GMRES 算法的左预处理略优于右预处理. 相对来说，基于左预处理的 LUSGS 预处理矩阵具有最优的收敛速度.

本文两个算例的计算都是在 Pentuim II (233 MHz) 单 CPU 微机上完成的. 对于 DDADI 方

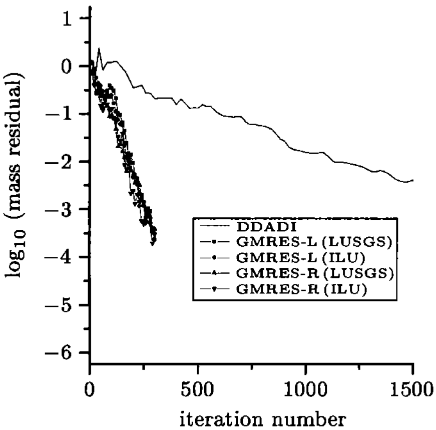

图 5 NACA0012 计算步数收敛史

Fig.5 Residual history for iteration number of NACA0012 flow

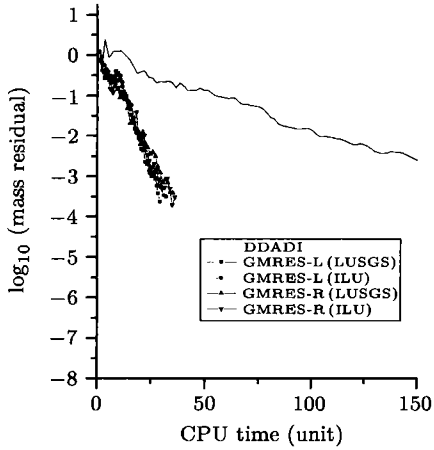

图 6 NACA0012 CPU 时间收敛史

Fig.6 Residual history for CPU time of NACA0012 flow

法，每个控制单元每计算一个时间步约需 3.7×10^{-4} s. 对于左预处理下 LUSGS 预处理矩阵的 GMRES 算法，扩压器流动约需要 4.63×10^{-4} s, NACA0012 流动约需要 4.14×10^{-4} s. 就单点单步所需的 CPU 时间而言， GMRES 算法需要较长的时间． GMRES 算法在两种流动计算中单点单步所需时间不同的原因是因为其内迭代的次数不同.

6 结 论

在早期工作中，我们已经证明 GMRES 算法在求解 Euler 方程上有很高的效率．由计算结果也可以看到，本文发展的预处理下的 GMRES 算法在求解定常可压缩黏性流动方面仍然有很大的优势．无论从计算步数上或从计算时间上来看， GMRES 算法都比常用的 DDADI 方法要快得多．同时，对预处理矩阵的左预处理和右预处理两种使用方法进行了分析和讨论．从收敛速度上看，左预处理比右预处理要好．另外，本文的两种预处理矩阵都取得了很好的效果.

将 GMRES 算法用于求解三维 Navier-Stokes 方程在本质上与二维情况相同，初步的计算表明，它仍然保持了较高的效率.

参 考 文 献

1 Orkwis PD, George JH. A comparison of CGS preconditioning methods for Newton's method solvers. AIAA Paper, 93-3327, 1993

2 Sonneveld P. CGS, A fast Lanczos-type solver for nonlinear systems. *SIAM Journal of Scientific Statistics and Computing*, 1987, 10: 350~356

3 Saad Y, Schultz MH. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 1986, 7: 856~869

4 Venkatakrishnan V. Preconditioned conjugate gradient methods for the compressible Navier-Stokes equations. *AIAA Journal*, 1991, 29(7): 1092~1100

5 Ajmani K, Ng WF, Liou MS. Preconditioned conjugate gradient methods for the Navier-Stokes equations. *Journal of Computational Physics*, 1994, 110: 68~81

6 Nielsen EJ, Anderson WK, Walters RW, et al. Application of Newton-Krylov methodology to a three-dimensional unstructured Euler code. AIAA Paper, 95-1733, 1995

7 Hong L, Joseph DB, Rainald L. A fast, matrix-free implicit method for compressible flows on unstructured grids. AIAA Paper, 99-0936, 1999

8 宁方飞, 徐力平. GMRES 算法在二维定常无黏流计算中的应用. 计算物理, 2000, 17(5): 537~547 (Ning Fangfei, Xu Liping. Application of GMRES algorithm in predictions of two-dimensional inviscid steady flows. *Chinese Journal of Computational Physics*, 2000, 17(5): 537~547 (in Chinese))

9 Spalart PR, Allmaras SR. A one-equation turbulence model for aerodynamic flows. AIAA Paper, 92-0439, 1992

10 Roe PL. Approximate Riemann solver, parameter vector, and difference schemes. *Journal of Computational Physics*, 1981, 433: 357~372

11 Jameson A, Yoon S. Lower-upper implicit schemes with multiple grids for the Euler equations. *AIAA Journal*, 1987, 25(7): 929~935

12 Meijerink JA, Van der Vorst HA. Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur. *Journal of Computional Physics*, 1981, 44: 134~155

13 Bogar TJ, Sajben M, Kroutil JC. Characteristic frequencies of transonic diffuser flow oscillations. *AIAA Journal*, 1983, 21(9): 1232~1240

APPLICATION OF GMRES ALGORITHM IN THE PREDICTIONS OF TWO-DIMENSIONAL TURBULENT STEADY FLOWS [1)]

Ning Fangfei　　Xu Liping

(*Faculty* 404, *Beijing University of Aeronautics & Astronautics, Beijing* 100083, *China*)

Abstract　A GMRES (Generalized Minimum RESidual) algorithm is developed for solving two-dimensional compressible Favier-averaged Navier-Stokes equations based on authors' previous work concentrated on Euler equations. An approximate linear system is constructed resulting from Newton linearization of governing flow equations and then solved by GMRES algorithm coupled respectively with two preconditioning matrices, i.e., LUSGS and ILU. Spalart-Allmaras turbulence model is used to enclose flow equations and solved using LUSGS algorithm in an uncoupled manner with flow equations. The matrix-vector multiplications which emerge in GMRES algorithm is solved using finite-difference approach, thus the calculations of exact left-hand-side coefficient matrix is avoided and computer memory is saved. Some analysis and discussions are made concentrated on two approaches of usage of preconditioning matrix, i.e., left preconditioning and right preconditioning. Based upon the computations of two typical turbulent flow cases, the comparisons of these two types of preconditionings, i.e., LUSGS and ILU, are presented. In addition, the performances of left preconditioning and right preconditioning are discussed. With the predictions of Sajben's diffuser flow and the transonic flow over NACA0012 airfoil, it is shown that GMRES algorithm coupled with appropriate preconditioning has superior advantage over widely used so-called DDADI method in convergence rate, left preconditioning is better than right preconditioning.

Key words　Navier-Stokes equations, GMRES algorithm, preconditioning, LUSGS preconditioning, ILU preconditioning

Received 5 March 2000, revised 4 August 2000.

1) The project supported by the National Natural Science Foundation of China (59525612).