비정렬 격자계에서 Block LU-SGS 기법의 개선에 관한 연구

Improvement on Block LU-SGS Scheme for Unstructured Mesh

김주성[1], 권오준[2]

Joo Sung Kim, Oh Joon Kwon

An efficient Gauss-Seidel time integration scheme is developed for solving the Euler and Navier-Stokes equations on unstructured meshes. Roe's FDS is used for the explicit residual computations and van Leer's FVS for evaluating implicit flux Jacobian. To reduce the memory requirement to a minimum level, off-diagonal flux Jacobian contributions are repeatedly calculated during the Gauss-Seidel sub-iteration process. Computational results based on the present scheme show that approximately 15% of CPU time reduction is achieved while maintaining the memory requirement level to 50-60% of the original Gauss-Seidel scheme.

1. 서론

최근 들어 비정렬 격자계에서 내재적 기법에 관한 많은 연구가 진행되고 있다. 대부분의 내재적 기법은 지배방정식에 대해 시간 선형화 (local time linearization) 방법을 적용하며, 결과적인 선형 방정식은 매시간 단계에서 반복 (iteration) 계산 방법을 사용하여 계산되어진다. 많은 연구들이 반복 계산 기법에 필요한 계산시간과 기억 용량(memory)의 절약을 위해 진행되고 있으며, 이러한 반복계산 기법으로는 Gauss-Seidel 기법에서 다양한 예조건자 (preconditioner)를 사용하는 Krylov subspace 방법 등에 이른다. Krylov subspace 방법 중에서 가장 성공적이고 많이 쓰이는 방법으로는 ILU 예조건자를 사용한 GMRES 기법이다. 하지만 이 방법은 예조건자에 필요한 자코비안 (Jacobian) 행렬을 저장하기 위해 많은 기억 용량을 필요로 하는 단점이 있다. 최근에 Luo[1] 등에 의해 개발된 matrix-free GMRES+LU-SGS 기법은 LU-SGS 예조건자를 사용하여 예조건자에 필요한 많은 기억 용량의 요구량을 제거시켰다.

참고 문헌 [2]에서 Anderson 등은 Gauss-Seidel 기법을 사용하여 Navier-Stokes 방정식을 매우 효율적으로 계산하였으며 , GMRES 기법과 비교하여 상응할 수 있는 수렴성을 얻을 수 있음을 보였다[3]. 또한 정렬 격자계에서 Jameson과 Yoon[4]에 의해 개발된 LU-SGS 기법은 여러 연구자들에 의해 비정렬 격자계에서 성공적으로 적용되고 있다[5,6]. LU-SGS 기법에서는 내재적 연산자를 플럭스 자코비안의 spectral radius로 근사화 하는 방법을 사용한다. 결과적으로 내재적 기법에 필요한 많은 기억 용량의 요구량을 필요로 하지 않으며, 반복계산당 걸리는 시간은 Runge-Kutta 외재적 방법에 비해서도 적게 걸리게 된다. 하지만 Wright[7] 등은 내재적 연산자로 spcetral radius를 사용하는 방법은 높은 종횡비의 격자에서는 적합하지 않으며, Navier- Stokes 방정식을 계산하는데 있어서는 이러한 특성에 의해 수렴성의 저하 현상이 발생하게 됨을 보였다. 또한 원래의 플럭스 자코비안을 사용함으로서 이러한 수렴성의 저하 현상을 줄일 수 있음을 보였다. 최근에 Chen[8] 등은 GMRES 기법에 비해 훨씬 더 적은 기억 용량을 필요로 하면서 더 좋은 수렴성을 얻을 수 있는 Block LU-SGS 기법을 개발하였다. 이 기법에서는 플럭스 자코비안 행렬에서 많은 부분을 차지하는 비대각 행렬을 미리 계산하여 저장하지 않고, Gauss-Seidel 반복 계산동안에 비대각 행렬에 해당되는 부분을 계속적으로 구하는 방법을 사용한다. 그러므로 대각 행렬뿐만 아니라 비대각

1) 한국과학기술원 기계공학과 대학원 항공우주공학전공
2) 한국과학기술원 기계공학과 항공우주공학전공

행렬까지 미리 계산한 후, 저장하여 사용하는 기존의 Gauss-Seidel 기법[2]에 비해 50-60% 정도의 기억 용량을 절약하면서, 계산시간은 증가하게 된다.

본 연구에서는 기존의 Gauss-Seidel 기법에 비해 기억용량과 계산 시간을 동시에 줄일 수 있는 Block LU-SGS 기법을 개발하였다. 이러한 특성을 얻기 위해 내재적 연산자로 FVS(Flux Vector Splitting) 기법을 사용하였다. FVS 기법은 FDS 기법과 달리 어떤 면에서의 플럭스의 값이 좌측과 우측의 값으로 분리되는 특성을 갖고 있으며, 이러한 특성을 이용하여 효율적인 Block LU-SGS 기법이 개발된다. 개발된 기법의 효율성을 검증하기 위해 비점성과 점성 유동에 대한 계산을 수행하였다.

2. 유한체적법

임의의 제어 체적 Ω와 제어 체적 경계면 $\partial\Omega$에 대해서 Navier-Stokes 방정식은 다음과 같이 쓸 수 있다.

$$\frac{\partial}{\partial t}\int_{\Omega} Q dA + \oint_{\partial\Omega}[F(Q,\vec{n}) - F_v(Q,\vec{n})]dS = 0 \quad (1)$$

여기서 Q는 보존 변수를 나타내며, $F(Q,\vec{n})$과 $F_v(Q,\vec{n})$는 각각 제어 체적 경계면에서 비점성과 점성 플럭스를 나타낸다. 지배 방정식 (1)을 제어 체적으로 median dual을 사용하는 격자점 중심 기법을 사용하여 차분화 하면 다음과 같이 표현된다.

$$A_i\frac{\partial Q_i}{\partial t} + \sum_{j}^{Nface}[F_{ij} - F_{v,ij}]dS = 0 \quad (2)$$

여기서 $Nface$는 현재의 제어체적 i를 구성하는 제어체적 경계면의 개수를 나타내며, F_{ij}와 $F_{v,ij}$는 수치적인 비점성과 점성 플럭스를 나타낸다. 본 연구에서는 비점성 플럭스에 대해 Roe의 FDS 기법을 사용하였으며, 고차정확도를 얻기 위해 최소자승법(least square method)[2]을 적용하였다. 점성 플럭스에 대해서는 Haselbacher[9]가 제안한 방법을 사용하였다.

3. 내재적 시간 적분법

3.1 Original Gauss-Seidel Scheme

식 (2)에 대해서 시간에 대한 Euler의 후방 차분법을 적용하면 다음과 같이 표현된다.

$$A_i\frac{\Delta Q_i}{\Delta t} + \sum_{j}^{Nface}(F_{ij}^{n+1} - F_{v,ij}^{n+1})dS = O(\Delta t^2) \quad (3)$$

내재적 기법을 적용하는 과정에서 가장 많이 사용되는 근사화 중의 하나는 1차 정확도 플럭스에 대한 선형화를 통해 플럭스 자코비안을 얻는 것이다. 수치적인 플럭스 벡터가 1차 정확도로 계산된다면, F_{ij}^{n+1}는 다음과 같이 쓸 수 있다.

$$F_{ij}^{n+1} = F_{ij}(Q_i^{n+1}, Q_j^{n+1}) \quad (4)$$

식 (4)에 Taylor series 전개를 적용하면, 다음과 같은 식을 얻을 수 있다.

$$F_{ij}(Q_i^{n+1}, Q_j^{n+1}) = F_{ij}(Q_i^n, Q_j^n) + \frac{\partial F_{ij}}{\partial Q_i}\Delta Q_i + \frac{\partial F_{ij}}{\partial Q_j}\Delta Q_j + O(\Delta t^2) \quad (5)$$

식 (4)와 (5)를 식 (3)에 대입하고, 점성 플럭스에 대해 외재적 처리를 하면, 다음과 같은 식을 얻을 수 있다.

$$\left[\frac{A_i}{\Delta t_i} + \sum_{j}^{Nface}\frac{\partial F_{ij}}{\partial Q_i}dS\right]\Delta Q_i + \sum_{j}^{Nface}\left[\frac{\partial F_{ij}}{\partial Q_j}dS\right]\Delta Q_j = -\sum_{j}^{Nface}[F_{ij}^n - F_{v,ij}^n]dS + O(\Delta t^2) \quad (6)$$

식 (6)은 Gauss-Seidel 기법으로 계산할 수 있으며, 이 과정에서 플럭스 자코비안을 미리 계산하여 저장하고 있어야 한다. 2-sweep symmetric Gauss-Seidel 기법을 적용하면 다음과 같이 표현된다.

For k=1, max_sweep Do
Forward sweep:

$$[D_i]^n\Delta Q_i^* + \sum_{j\in Lower}[O_{ij}]^n\Delta Q_j^* + \sum_{j\in Upper}[O_{ij}]^n\Delta Q_j^{k-1} = -Res_i^n \quad (7)$$

Backward sweep:

$$[D_i]^n\Delta Q_i^k + \sum_{j\in Lower}[O_{ij}]^n\Delta Q_j^* + \sum_{j\in Upper}[O_{ij}]^n\Delta Q_j^k = -Res_i^n \quad (8)$$

End Do

여기서 $j\in Lower$와 $j\in Upper$는 각각 행렬의 lower part와 upper part를 나타낸다. 대각 행렬

과 비대각 행렬, $[D_i]^n$과 $[O_{ij}]^n$는 다음과 같다.

$$[D_i]^n = \left[\frac{A_i}{\Delta t_i} + \sum_j^{Nface} \frac{\partial F_{ij}}{\partial Q_i} dS\right] \quad (9)$$

$$[O_{ij}]^n = \frac{\partial F_{ij}}{\partial Q_j} dS \quad (10)$$

잔류항 Res_i^n는 고차정확도로 계산된 값이다.

3.2 Block LU-SGS Scheme

식 (7)과 (8)에서, 플럭스 쟈코비안을 저장하기 위한 대부분의 저장 용량은 비대각 행렬을 저장하기 위해 필요하다. 이러한 비대각 행렬을 저장하기 위한 기억 용량은 각각의 스윕(sweep) 단계에서 Gauss-Seidel sub-iteration 동안에 비대각 행렬에 해당되는 부분을 계속적으로 계산함으로서 없앨 수 있다[8]. 참고 문헌 [8]에서는 비대각 행렬에 해당되는 부분의 계속적인 계산에 의해 발생되는 계산 시간 증가를 최소화하기 위해 다음과 같은 방법을 사용하였다.

$$\frac{\partial F_{ij}}{\partial Q_j}\Delta Q_j = F_{ij}(Q_i^n, Q_j^n + \Delta Q_j^n) - F_{ij}(Q_i^n, Q_j^n) + O(\Delta t^2) \quad (12)$$

식 (12)에서 비대각 행렬과 보존형 변수의 증분 벡터의 곱이 플럭스의 증분으로 표현되었다. 식 (12)를 식 (6)에 대입하면 다음과 같다.

$$[D_i]\Delta Q_i + \sum_j^{Nface}[F_{ij}(Q_i^n, Q_j^n + \Delta Q_j^n) - F_{ij}(Q_i^n, Q_j^n)]dS = -Res_i^n \quad (13)$$

식 (13)에서는 식 (6)과 달리, 더 이상 비대각 행렬을 계산할 필요가 없게 되었다. 식 (13)에 대해 2-sweep symmetric Gauss-Seidel 기법을 적용하면 다음과 같이 표현된다.

For k=1, max_sweep Do
Forward sweep:

$$[D_i]^n\Delta Q_i^* + \sum_{j\in Lower} F_{ij}(Q_i^n, Q_j^n + \Delta Q_j^*)dS + \sum_{j\in Upper} F_{ij}(Q_i^n, Q_j^n + \Delta Q_j^{k-1})dS = \sum_j^{Nface} F_{ij}(Q_i^n, Q_j^n)dS - Res_i^n \quad (14)$$

Backward sweep:

$$[D_i]^n\Delta Q_i^k + \sum_{j\in Lower} F_{ij}(Q_i^n, Q_j^n + \Delta Q_j^*)dS + \sum_{j\in Upper} F_{ij}(Q_i^n, Q_j^n + \Delta Q_j^k)dS = \sum_j^{Nface} F_{ij}(Q_i^n, Q_j^n)dS - Res_i^n \quad (15)$$

End Do

여기서 ΔQ^*, ΔQ^{k-1}, ΔQ^k는 각각의 sub-iteration 단계에서 보존형 변수의 증분을 나타낸다. 식 (14)와 (15)에서 $F_{ij}(Q_i^n, Q_j^n)$는 Gauss- Seidel sub-iteration 동안에 변하지 않으므로 우변으로 이항되었으며, 계산의 초기에 한번만 계산되어진다. 참고 문헌 [8]에서 제안된 Block LU-SGS 기법은 Roe의 FDS 기법을 사용하여 플럭스 쟈코비안을 구성하였다.

3.3 Improvement on Block LU-SGS Scheme

Roe의 FDS 기법을 사용하여 내재적 연산자를 구성하는 방법은 Euler 방정식을 해석하는데 성공적으로 사용되고 있으며, 이 경우 Roe-averaged matrix의 복잡성으로 인해 대부분 Barth[10]가 제안한 근사식을 사용한다. 하지만 Navier-Stokes 방정식을 해석하는데 있어서 Roe의 FDS 기법을 사용한 내재적 연산자는 심각한 안정 제약 조건을 갖게된다. 본 연구에서는 van Leer의 FVS 기법을 사용하여 이러한 문제점을 해결하였다. FVS 기법에서 플럭스 항은 전파되는 파의 방향에 따라 분리되어 차분화되어지며 다음과 같다.

$$F_{ij}(Q_i^n, Q_j^n + \Delta Q_j^n) = F_{ij}^+(Q_i^n) + F_{ij}^-(Q_j^n + \Delta Q_j^n) \quad (16)$$

$$F_{ij}(Q_i^n, Q_j^n) = F_{ij}^+(Q_i^n) + F_{ij}^-(Q_j^n) \quad (17)$$

식 (16)과 (17)을 식 (13)에 대입하면 다음과 같다.

$$[D_i]\Delta Q_i + \sum_j^{Nface}[F_{ij}^-(Q_j^n + \Delta Q_j^n) - F_{ij}^-(Q_j^n)]dS = -Res_i^n \quad (18)$$

위 식에서 양의 플럭스 벡터 F_{ij}^+는 소거되었다. (18)식을 2-sweep symmetric Gauss-Seidel 기법을 적용하면 다음과 같이 표현된다(type-1 Block LU-SGS scheme).

For k=1, max_sweep Do
Forward sweep:

$$[D_i]^n \Delta Q_i^* + \sum_{j\in Lower} F_{ij}^-(Q_j^n + \Delta Q_j^*)\,dS + \sum_{j\in Upper} F_{ij}^-(Q_j^n + \Delta Q_j^{k-1})\,dS = \sum_{j}^{Nface} F_{ij}^-(Q_j^n)\,dS - Res_i^n \quad (19)$$

Backward sweep:

$$[D_i]^n \Delta Q_i^k + \sum_{j\in Lower} F_{ij}^-(Q_j^n + \Delta Q_j^*)\,dS + \sum_{j\in Upper} F_{ij}^-(Q_j^n + \Delta Q_j^k)\,dS = \sum_{j}^{Nface} F_{ij}^-(Q_j^n)\,dS - Res_i^n \quad (20)$$

End Do

식 (19)-(20)을 식 (14)-(15)와 비교하면, 비대각 행렬에 해당하는 계산량이 상당히 감소될 수 있음을 알 수 있다. 이러한 계산량의 감소는 FDS에 비해 FVS가 간단한 점과 식 (18)에서 양의 플럭스의 값이 소거되었기 때문에 발생한다.

식 (19)-(20)에 대해서 LU-SGS 기법[4]을 유도하는 과정에서 사용되었던 방법을 적용하면 좀 더 간단한 식을 얻을 수 있다. 식 (20)에서 식 (19)를 빼면 다음과 같다.

$$[D_i]^n \Delta Q_i^k - [D_i]^n \Delta Q_i^* + \sum_{j\in Upper} [F_{ij}^-(Q_j^n + \Delta Q_j^k) - F_{ij}^-(Q_j^n + \Delta Q_j^{k-1})]\,dS = 0 \quad (21)$$

식 (21)을 식(20) 대신 backward sweep에 사용하면 다음과 같이 쓸 수 있다(type-2 Block LU-SGS scheme).

For k=1, max_sweep Do

Forward sweep:

$$[D_i]^n \Delta Q_i^* + \sum_{j\in Lower} F_{ij}^-(Q_j^n + \Delta Q_j^*)\,dS + \sum_{j\in Upper} F_{ij}^-(Q_j^n + \Delta Q_j^{k-1})\,dS = \sum_{j}^{Nface} F_{ij}^-(Q_j^n)\,dS - Res_i^n \quad (22)$$

Backward sweep

$$[D_i]^n (\Delta Q_i^k - \Delta Q_i^*) + \sum_{j\in Upper} F_{ij}^-(Q_j^n + \Delta Q_j^k)\,dS = \sum_{j\in Upper} F_{ij}^-(Q_j^n + \Delta Q_j^{k-1})\,dS \quad (23)$$

End Do

식 (21)의 backward sweep에서 $\sum_{j\in Upper} F_{ij}^-(Q_j^n + \Delta Q_j^{k-1})\,dS$항은 forward sweep에서 계산한 값과 정확히 같으므로, forward sweep 동안에 계산된 값을 저장하여 사용한다. 이 항을 저장하기 위해서는 $4\times node$의 기억 용량이 요구되며, 이 양은 전체 기억 용량에 비해서는 매우 적은 양이다. 식 (20)과 식 (23)을 비교해 볼 때, 식 (23)에서는 비대각 플럭스 자코비안에 해당하는 부분이 upper matrix 부분만 남아있으므로, 식 (20)에 비해 계산량이 감소될 수 있음을 알 수 있다.

4. 결과 및 고찰

기존의 Gauss-Seidel 기법(original Gauss-Seidel scheme)과 본 논문에서 제안된 type-1과 type-2 Block LU-SGS 기법에 대한 특징을 살펴보기 위해 비점성과 점성 유동에 대한 해석을 수행하였다.

4.1 비점성 유동 해석

첫 번째 검증 경우로 NACA 0012 익형 주위에 대해서 자유류의 마하수가 0.63, 받음각이 2도인 완전한 아음속 유동에 대해 계산하였다. Fig. 1은 Advancing Front 기법을 사용하여 생성한 등방성 삼각형 격자이며, Fig. 2는 마하수 분포를 보여준다. 격자점과 격자 수는 각각 8,820개, 17,392개이며, 208개의 격자점이 익형 주위에 위치되어 있다. CFL 수는 10에서 1,000까지 100번 반복 계산동안 선형적으로 증가시켜서 사용하였다. Fig. 3은 반복 계산 회수 당 수렴 곡선을 보여준다. 세 가지 기법 모두 비슷한 수렴성을 보여주고 있다. Fig. 4는 계산 시간당 수렴곡선이며, type-2 Block LU-SGS 기법이 다른 기법에 비해 약간 빠르게 수렴함을 볼 수 있다. 이러한 결과로부터 type-2 Block LU-SGS 기법이 단위 반복 계산 당 걸리는 시간이 다른 기법에 비해 적게 걸림을 알 수 있다. Fig. 5에서는 단위 반복 계산 당 걸리는 시간을 도표화하였다. 기존의 Gauss-Seidel 기법은 type-2 Block LU-SGS 기법에 비해 약 16% 정도의 시간이 단위 반복 계산 당 더 필요함을 볼 수 있으며, type-1 Block LU-SGS 기법도 약 13% 정도의 시간이 더 소요된다. 3.3 절에서 볼 수 있듯이 type-1과 type-2 Block LU-SGS 기법은 기존의 Gauss-Seidel 기법에 비해 계산량이 증가된다. 하지만 Fig. 5에서의 결과처럼

계산 시간이 감소하는 원인은 type-1과 type-2 Block LU-SGS 기법이 기존의 Gauss-Seidel 기법에 비해 기억 용량의 크기가 현저히 감소하여 캐쉬(cache) 메모리의 효율성이 크게 증가하였기 때문으로 판단된다. Fig. 6은 각각의 기법에 대한 기억 용량 요구량을 나타낸다. Type-1과 type-2 Block LU-SGS 기법은 거의 비슷하며, 기존의 Gauss-Seidel 기법은 type-2 Block LU-SGS 기법에 비해 약 2.5배 정도의 기억 용량을 필요로 함을 볼 수 있다.

4.2 점성 유동 해석

점성 유동에 대한 검증 경우로 NACA 0012 익형 주위에 대해서 자유류의 마하수가 0.5, 받음각이 3도, 레이놀즈 수가 5,000인 층류 유동에 대해 해석하였다. Fig. 7은 계산 격자를 보이고 있으며, C-type 정렬격자를 비정렬 격자 구조로 바꾸어 사용하였다. 격자점은 17,783개, 격자는 17,484개이며, 익형 주위에 192개의 격자점이 분포되어 있다. CFL 수는 비점성 경우와 같이 10에서 1,000 까지 100번 반복 계산동안 선형적으로 증가시켜 사용하였다. Fig. 8은 Fig. 7에서 보인 계산 격자에 대한 마하수 분포를 보여주고 있다. Fig. 9와 Fig. 10은 수렴곡선을 보여주고 있으며, 비점성 경우와 마찬가지로, 반복 계산 회수 당 수렴성은 거의 비슷하나 계산 시간당 수렴성은 type-2 Block LU-SGS 기법이 다른 기법에 비해 약간 우수함을 볼 수 있다. Fig. 11은 단위 반복 계산 당 걸리는 시간을 나타내며, 비점성 경우와 비슷한 경향을 볼 수 있다. Fig. 12는 기억 용량 요구량을 비교한 그림이다. 기존의 Gauss-Seidel 기법은 type-2 Block LU-SGS 기법에 비해 약 2.1배 정도의 기억 용량을 필요로 한다. 사각형 격자가 삼각형 격자에 비해 대각 행렬에 비해 비대각 행렬이 차지하는 비율이 작기 때문에 삼각형 격자에 비해 적은 기억 용량 증가를 나타낸다.

5. 결론

기존의 Gauss-Seidel 기법에 비해, 계산시간과 기억 용량을 절약할 수 있는 기법을 개발하였다. 본 연구에서 제안된 기법을 비점성과 점성 유동 해석에 적용한 결과, 기존의 Gauss-Seidel 기법에 비해, 계산 시간은 약 15%, 기억 용량은 약 50-60% 정도를 절감할 수 있었다.

참고문헌

[1] Luo, H., Baum, J., and Lohner, R., "A Fast, Matrix-free Implicit Method for Compressible Flows on Unstructured Grids," Journal of Computational Physics, Vol. 146, 1998, pp. 664-690.

[2] Anderson, W. K. and Bonhaus, D. L., "An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids," Computers and Fluids, Vol. 23, No. 1, 1994, pp. 1-21.

[3] Anderson, W. K., Rausch, R. D., and Bonhaus, D. L., "Implicit/Multigrid Algorithms for Incompressible Turbulent Flows on Unstructured Grids," AIAA Paper 95-1740, 1995.

[4] Jameson, A. and Yoon, S., "Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations," AIAA Journal, Vol. 25, No. 7, 1987, pp. 929-935.

[5] Soetrisno, M., Imlay, S. T., and Robert, D. W., "A Zonal Implicit Procedure for Hybrid Structured-Unstructured Grids," AIAA Paper 94-0617, 1994.

[6] Sharov, D. and Nakahashi, K., "Reordering of 3-D Hybrid Unstructured Grids for Vectorized LU-SGS Navier-Stokes Computations," AIAA Paper 97-2102, 1997.

[7] Wright, M. J., Candler, G. V., and Prampolini, M., "Data-Parallel Lower-Upper Relaxation Method for the Navier-Stokes Equatons," AIAA Journal, Vol. 34, No. 7, 1996, pp. 1371-1377.

[8] Chen, R. F. and Wang, Z. J., "Fast, Block Lower-Upper Symmetric Gauss-Seidel Scheme for Arbitrary Grids," AIAA Journal, Vol. 38, No. 12, 2000, pp. 2238-2245.

[9] Haselbacher, A. C., McGuirk, J. J., and Page, G. J., "Finite-Volume Discretization Aspects for Viscous Flows on Mixed Unstructured Grids," AIAA Paper 97-1946, 1997.

[10] Barth, T. J., "Analysis of Implicit Local

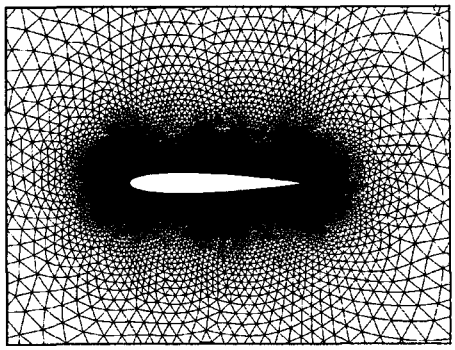

Fig. 1 Isotropic triangular meshes around a NACA 0012 airfoil

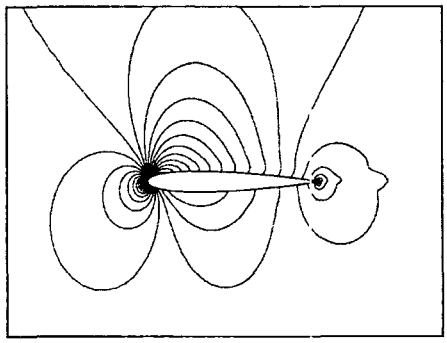

Fig. 2 Mach number contour for inviscid flow around a NACA 0012 airfoil(M=0.63, α =2°)

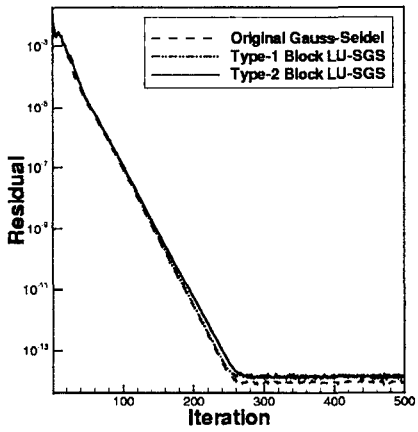

Fig. 3 Comparison of convergence history in terms of iteration

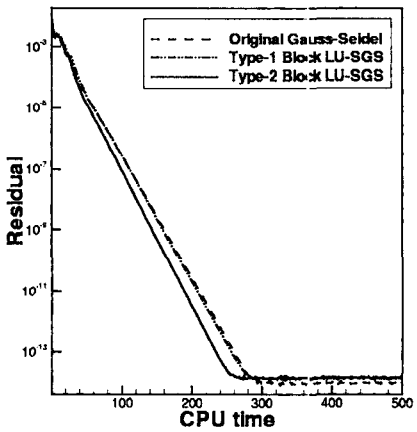

Fig. 4 Comparison of convergence history in terms of CPU time

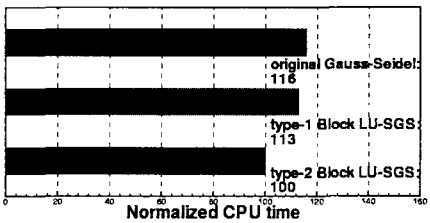

Fig. 5 Percentage comparison of computational time for the simulations of inviscid flow around a NACA 0012 airfoil

Linearization Techniques for Upwind and TVD Algorithms," AIAA Paper 87-0595, 1987.

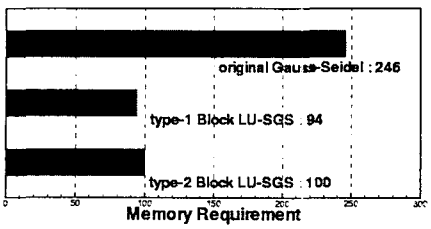

Fig. 6 Percentage comparison of memory requirement for the simulations of inviscid flow around a NACA 0012 airfoil

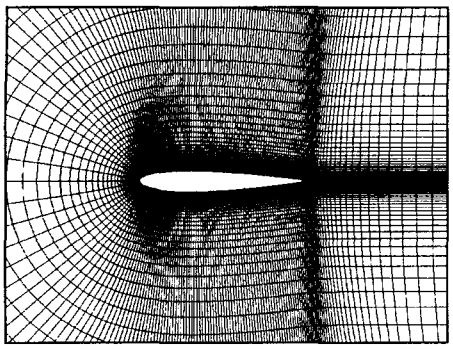

Fig. 7 Quadrilateral meshes around a NACA 0012 airfoil

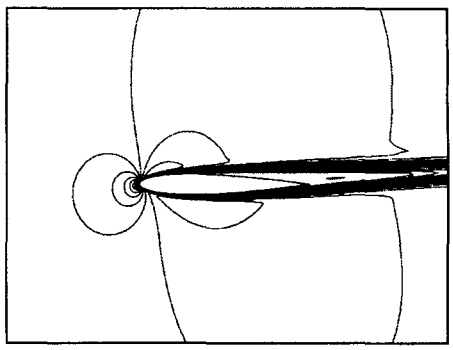

Fig. 8 Mach number contour for laminar viscous flow around a NACA 0012 airfoil(M=0.5, α =3°, Re=5,000)

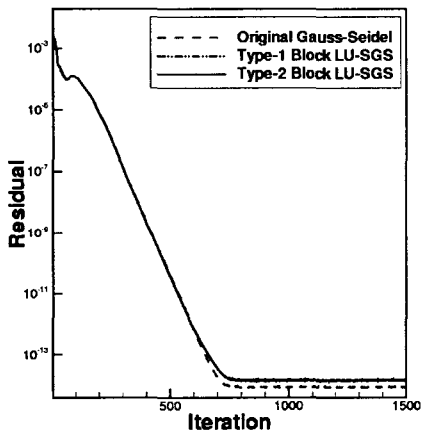

Fig. 9 Comparison of convergence history in terms of iteration

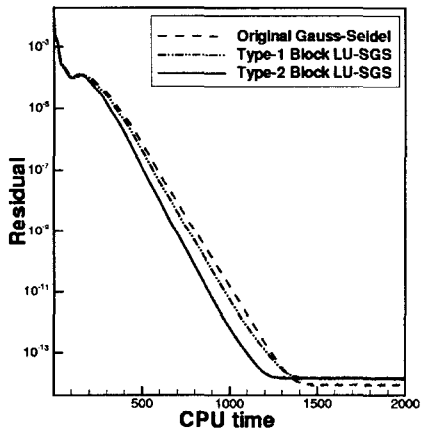

Fig. 10 Comparison of convergence history in terms of CPU time

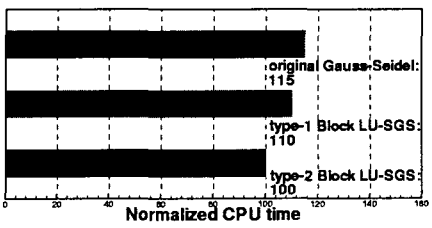

Fig. 11 Percentage comparison of computational time for the simulations of laminar viscous flow around a NACA 0012 airfoil

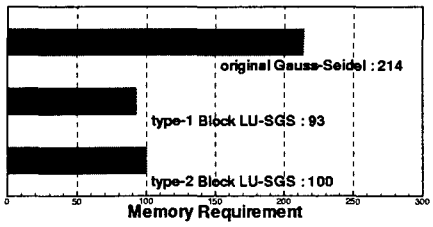

Fig. 12 Percentage comparison of memory requirement for the simulations of laminar viscous flow around a NACA 0012 airfoil