ACCELERATION OF GMRES CONVERGENCE FOR SOME CFD PROBLEMS: preconditioning and stabilization techniques

Azzeddine Soulaimani[*]
Département de Génie Mécanique, École de technologie supérieure,
1100 Notre-Dame Ouest, Montréal (Québec), H3C 1K3, Canada
email: asoulaimani@mec.etsmtl.ca

Nizar Ben Salah
Laboratoire de Mécanique, Matériaux et Procédés
ESSTT, 5 Avenue Taha Hussein, Tunis, Tunisia

Yousef Saad
Department of Computer Science and Engineering,
University of Minnesota,4-192EE/CSci Building,
200 Union Street S.E., Minneapolis, MN 55455

Abstract: Large CFD problems are often solved using iterative methods. Preconditioning is mandatory to accelerate the convergence of iterative methods. This paper presents new results using several preconditioning techniques. These preconditoners are non-standard in the CFD community. Several numerical tests were carried out for solving three-dimensional incompressible, compressible and magneto-hydrodynamic (MHD) problems. A selection of numerical results is presented showing in particular that the Flexible GMRES algorithm preconditioned with ILUT factorization provides a very robust iterative solver.

1 Introduction

The solution of partial differential equations of fluid dynamics by any time and space discretization method often leads to solving a non-linear problem in the Euclidian space $\Re^n$:

$$F(x) = 0. \tag{1}$$

[*]Corresponding author,

A well known strategie to solve (1) is the Newton iterative method: let $x_0 \in \Re^n$, then

$$x_{i+1} = x_i + J^{-1}(x_i)F(x_i) \qquad i \geq 0,$$

where $J(x_i)$ is the Jacobian associated with F and evaluated at x_i.
Thus, at each iteration i one has to solve a linear problem

$$J(x_i)y = F(x_i). \tag{2}$$

For a relatively large dimension n, iterative methods such as GMRES or BiCGSTAB are prefered over direct solvers such as Gauss elimination method.

To summarize, a solution strategy often used to solve CFD problems consists of tow nested loops, the first one over time and the second one over Newton's iterations. This can be sketched as follows

ALGORITHM **1.1** *General solution strategy*

```
1. Do i=1, number − time − steps
2. Compute and factorize the Jacobian
   matrix (or an approximation) when necessary
3. Do k=1,number − Newton − iterations
4.       Compute the residual $F(x_i)$ and
         solve the linearized system (2)
5.       Check for convergence
6.       EndDo
7. EndDo
```

At each time step, one is then concerned oby the convergence of Newton's iterations and the cost of solving the linearized problem (2). For steady problems, one is also concerned by the global convergence. In practice, two different difficulties may be encountred, the divergence of Newton's iterations and the stagnation of the global loop. Possible cures can be found in the use of one or a combination of the following actions:

- (a) reduction of the time step,
- (b) a judicious selection of the initial solution,
- (c) add some robustness to the solution algorithm for the linearized problem: increase the Krylov-space dimension, use of a better preconditioning...
- (d) use of stabilization techniques for discretization,
- (e) use of multigrid methods or adaptive meshing techniques, etc.

In this paper, some techniques related to options (c) and (d) above are proposed and numerically tested. For simplicity, we rewrite the system (2), using linear algebra notations, as: $Ax = b$. Preconditioning is the way by to accelerate the convergence of iterative methods. Generally, a new system equivalent to (2) is built using a preconditioning matrix M:

$$AM^{-1}Mx = b,$$

where M is generally an approximation of A. Roughly speaking, M is an approximation of A such that the conditioning number of AM^{-1} is close to unity. The convergence of the iterative method is strongly related to the choice of M. It has been a big challenge in iterative methods to develop robust and wide-range use preconditioners. However, it is not possible to know which one performs best in all situations. Very often, one method can perform well in a particular class of problems but may be expensive, slow or even diverge in other problems. Thus, it is very important to have at hand a large range of iterative methods in order to choose the best suited one for a particular problem. This paper contributes to testing and validating several non-standard preconditioners on incompressible, compressible and MHD flow problems. These preconditioners include the ILUT factorization of the Jacobian matrix, a higher order update of the Jacobian using the Flexible GMRES method, a lower order update of the Jacobian using the Broyden method and the Deflated GMRES algorithm. The test problems chosen are particularly hard to solve with classical iterative methods. Useful stabilized finite element techniques for space discretization are also discussed. The space discretization plays an important role not only to obtain stable and accurate solution but it affects also the convergence of iterative solvers. By stabilizing the standard Galerkin finite element method for , the eigenvalues of the stiffness matrix are shifted to the right and the conditioning is improved.

The outline of this paper is as follows. First, we present several choices of building the preconditioner. In the second section, we briefly describe the CFD model problems used for the validations. A selection of numerical results for three-dimensional incompressible, compressible and MHD problems are shown. Finally, some concluding comments are drawn.

2 Preconditioners

First, the preconditioners used in this work are briefly presented. More details are given in the references. We begin with a review of the right-preconditioned GMRES algorithm [1,2].

Algorithm **2.1** *Right preconditioned GMRES*

```
1. Compute $r_0 = b - Ax_0$, $\beta := ||r_0||_2$,
   $v_1 := r_0/\beta$.
2. Define the $(m+1) \times m$ matrix $\overline{H}_m = \{h_{i,j}\}_{1 \leq i \leq m+1, 1 \leq j \leq m}$. Set $\overline{H}_m = 0$.
3. For $j = 1, 2, ..., m$ Do:
4.    Compute $w_j := AM^{-1}v_j$
5.       For $i = 1, ..., j$ Do:
6.          $h_{ij} := (w_j, v_i)$
7.          $w_j := w_j - h_{ij}v_i$
8.       EndDo
9.    $h_{j+1,j} = ||w_j||_2$. If $h_{j+1,j} = 0$ set
      $m := j$, goto 12
10.   $v_{j+1} = w_j/h_{j+1,j}$
11. EndDo
12. Compute $y_m$ the minimizer of
    $||\beta e_1 - \overline{H}_m y||_2$ and $x_m = x_0 + V_m y_m$.
13. If satisfied Stop, else set $x_0 := x_m$ goto 1.
```

Remark. In exact Newton method, A represents the true Jacobian matrix. However, it is not always pssible to compute it analytically or it may be expensive to recompute it at every iteration, especially for CFD problems. It is preferable to compute an approximation of the Jacobian and freeze it for a prescribed number of time steps and Newton iterations. This matrix will be used to construct the preconditioner M. On the other hand, the action of the Jacobian on a vector (i.e. the matrix-by-vector product in line 4 (i.e. Az_j)) is approximated using a finite difference quotient such as: $[F(x_0 + \epsilon z_j) - F(x_0)]/\epsilon$, where ϵ is an appropriate small number. All the problems considered in this paper are solved using this *non-linear version* of GMRES.

2.1 ILUT preconditioner

We now describe three preconditioning techniques. The first one is the incomplete factorization $ILUT(\tau, p)$ of A. One of the most common ways to define the preconditioning matrix M is through Incomplete LU factorizations. In essence, an ILU factorization is simply an approximate Gaussian elimination. When Gaussian Elimination is applied to a sparse matrix A, a large number of nonzero elements may appear in locations originally occupied by zero elements. These fill-ins are often small elements and may be dropped to obtain Incomplete LU factorizations. So ILU is in essence a Gaussian Elimination procedure in which fill-ins are dropped.

The simplest of these procedures is ILU(0) which is obtained by performing the standard LU factorization of A and dropping all fill-in elements that are generated during the process. In other words, the L and U factors have the same pattern as the lower and upper triangular parts of A (respectively). More accurate factorizations denoted by ILU(k) and IC (k) have been defined which drop fill-ins according to their 'levels'. Level-1 fill-ins for example are generated from level-zero fill-ins (at most). So, for example, ILU(1) consists of keeping all fill-ins that have level zero or one and dropping any fill-in whose level is higher.

Another class of preconditioners is based on dropping fill-ins according to their numerical values. One of these methods is ILUT (ILU with Threshold). This procedure uses basically a form of Gaussian elimination which generates the rows of L and U one by one. Small values are droped during the elimination using a parameter τ. A second parameter, p, is then used to keep the largest p entries in each of the rows of L and U. This procedure is denoted by $ILUT(\tau, p)$ of A. More details on ILUT and other preconditioners can be found in [?]. Practical values of p and τ are $p = NNZ/NEQ + lfil$, with NNZ the number of non-zero entries of the original matrix, NEQ the number of unknowns and $lfil$ is a positive or negative integer, and $10^{-3} \leq \tau \leq 10^{-9}$.

ALGORITHM **2.2** *ILUT*

```
1. Do i=1, n
2. Do k=1,i-1
3. $A_{i,k} := A_{i,k}/A_{k,k}$
4. If $|A_{i,k}|$ is not too small then
     Do $j = k+1, n$
5.     $A_{i,j} := A_{i,j} - A_{i,k} * A_{k,j}$
6.     EndDo
7. EndDo
8. Drop small elements in row $A_{i,*}$
9. EndDo
```

2.2 Broyden update

The construction of the next preconditioner is inspired by the following fact: The construction of M and its factorization are time consuming when the dimension of the problem is large. Furthermore, for non-linear problems, the preconditioner should in principle be recomputed at each Newton iteration. A simple strategy to reduce this cost is to compute only one ILUT factorization of the approximate Jacobian matrix (denoted by M_*) and to keep it for a number of iterations and time-steps. For each subsequent Newton iterations, ${M_*}^{-1}$ is updated using a low order Broyden correction [4]:

$$M^{-1} = {M_*}^{-1} + \frac{(s_i - {M_*}^{-1} y_i) s_i^t {M_*}^{-1}}{s_i^t {M_*}^{-1} y_i}$$

where ${M_*}^{-1}$ is the computed ILUT preconditioner of M_*, $s_i = x_{i+1} - x_i$ is the correction of the solution x_i at iteration i and $y_i = F(x_{i+1}) - F(x_i)$.

ALGORITHM **2.3** *Broyden update*

```
1.  Set $M_* = A^{-1}(ilut)$.
2.  Compute $r_0 = b - Ax_0$, $\beta := ||r_0||_2$,
      $v_1 := r_0/\beta$
3.  Define the $(m+1) \times m$ matrix $\overline{H}_m = \{h_{i,j}\}_{1\le i\le m+1, 1\le j\le m}$. Set $\overline{H}_m = 0$.
4.  For $j = 1, 2, ..., m$ Do:
5.     Compute $w_j := AM^{-1}v_j$
6.        For $i = 1, ..., j$ Do:
7.        $h_{ij} := (w_j, v_i)$
8.        $w_j := w_j - h_{ij}v_i$
9.     EndDo
10.    $h_{j+1,j} = ||w_j||_2$. If $h_{j+1,j} = 0$ set
      $m := j$, goto 12
11.    $v_{j+1} = w_j/h_{j+1,j}$
12. EndDo
13. Compute $y_m$ the minimizer of $||\beta e_1 - \overline{H}_m y||_2$
      and $x_m = x_0 + V_m y_m$.
14. If satisfied Stop, else set $x_0 := x_m$
      set $s_0 = x_m - x_{m-1}$, $y_0 = y_m - y_{m-1}$,
      $M^{-1} = {M_*}^{-1} + \frac{(s_0 - {M_*}^{-1} y_0) s_0^t {M_*}^{-1}}{s_0^t {M_*}^{-1} y_0}$
      and goto 2.
```

2.3 Flexible GMRES

The second preconditioner presented here is the Flexible GMRES algorithm [5]. It is basically identical to the GMRES algorithm where step (4) is solved more accurately using an inner-iterative solver such as GMRES itself. In other words, applying M^{-1} to any direction z is equivalent to solving $My = z$. This system is solved approximately by GMRES using also a preconditioner which could be the ILUT(τ', p') factorization of M with p' generally smaller than p.

Recall from what was said above that the role of the preconditioner M is to solve the linear system $Ax = b$ *approximately and inexpensively.* At one extreme, we can find a preconditioner M that is very close to A, leading to a very fast convergence of GMRES, possibly in just one iteration. However, in this situation it is likely that M will require too much memory and be too expensive to compute. At the other extreme, we can compute a very inexpensive preconditioner such as one obtained with ILU(0) – but for realistic problems, convergence is unlikely to be achieved.

If the goal of the preconditioner is to solve the linear system approximately, then one may think of using a full-fledged iterative procedure, utilizing whatever preconditioner is available. The resulting overall method will be an inner-outer method, which includes two nested loops: an outer GMRES loop as defined earlier, and an inner GMRES loop in lieu of a preconditioning operation. In other words, we wish to replace the simple preconditioning operation in line 4 of algorithm (2.3) by an iterative solution procedure. The result of this is that each step the preconditioner M is actually defined as some complex iterative operation – and we can denote the result by $z_j = M_j^{-1} v_j$. Therefore, the effect of this on algorithm (2.3) is that the preconditioner M varies at every step j. However, algorithm (2.3) works only for constant preconditioners M. It would fail completely for the variable preconditioner case. To remedy this, a variant of GMRES called Flexible GMRES (FGMRES) has been developed, see [?]. For the sake of brevity, we will not sketch the method. The main difference between FGMRES and algorithm (2.3) is that the vectors z_j generated in line 4 must now be saved. These vectors are then used again in Line 13, in which they replace the vectors v_i of the basis V_m used to compute the approximation x_m. This gives the following algorithm:

Algorithm **2.4** *FGMRES*

```
1.  Compute $r_0 = b - Ax_0$, $\beta := ||r_0||_2$, $v_1 := r_0/\beta$.
2.  Define the $(m+1) \times m$ matrix $\overline{H}_m =$
        $\{h_{i,j}\}_{1 \leq i \leq m+1, 1 \leq j \leq m}$. Set $\overline{H}_m = 0$.
3.  For $j = 1, 2, ..., m$ Do:
4.      Compute $z_j := M_j^{-1} v_j$
5.      Compute $w_j := Az_j$
6.          For $i = 1, ..., j$ Do:
7.              $h_{ij} := (w_j, v_i)$
8.              $w_j := w_j - h_{ij} v_i$
9.          EndDo
10.     $h_{j+1,j} = ||w_j||_2$.
        If $h_{j+1,j} = 0$ set $m := j$, goto 13
11.     $v_{j+1} = w_j / h_{j+1,j}$
12. EndDo
13. Define $Z_m := [z_1, ..., z_m]$, and
        $H_m = \{h_{i,j}\}_{1 \leq i \leq m+1, 1 \leq j \leq m}$.
14. Compute $y_m$ the minimizer of
        $||\beta e_1 - \overline{H}_m y||_2$ and $x_m = x_0 + V_m y_m$.
15. If satisfied Stop, else set $x_0 := x_m$ goto 1.
```

A non-linear version of FGMRES is obtained, similar to the standard case of GMRES, by computing the matrix-by-vector product Az_j in line 5 via a finite difference formula.

2.4 Deflation

Deflation is another technique used to accelerate the convergence when the matrix A has some small eigenvalues. When used in the framework of GMRES algorithm, it consists of adding to the Krylov subspace a set of eigenvectors of A which are associated to the smallest eigenvalues. After constructing the m-dimensional Krylov subspace (step 13 of algorithm 2.1), p vectors are computed which are associated to the p smallest eigenvalues of A using a projection method (For more details, see [6,7]). Then, in the subsequent GMRES (or Newton) loop, a $m+p$-dimensional Krylov subspace is constructed using the available p eigenvectors..

Algorithm **2.5** *DEFLATED GMRES*

```
1.  Set $p = 0$.
2.  Compute $r_0 = b - Ax_0$, $\beta := ||r_0||_2$,
        $v_1 := r_0/\beta$
3.  Define the $(m+1) \times m$ matrix $\overline{H}_m =$
        $\{h_{i,j}\}_{1 \leq i \leq m+1, 1 \leq j \leq m}$. Set $\overline{H}_m = 0$.
4.  For $j = 1, 2, ..., m$ Do:

Set
    $w_j = \begin{cases} v_j & \text{if } j \leq m-p \\ u_{j-m+p}(\text{eigenvector}) & \text{otherwise} \end{cases}$

6.      Compute $z := AM^{-1} w_j$
7.          For $i = 1, ..., j$ Do:
8.              $h_{ij} := (z, v_i)$
9.              $z := z - h_{ij} v_i$
10.         EndDo
11.     $h_{j+1,j} = ||z||_2$. $v_{j+1} = z / h_{j+1,j}$
12. EndDo
13. Define $V_{m+1} := [v_1, ..., v_m]$ and
        $W_m := [w_1, ..., w_m]$
14. Compute $y_m$ the minimizer of $||\beta e_1 - \overline{H}_m y||_2$
        and $x_m = x_0 + V_m y_m$.
15. If satisfied Stop.
16. Set $x_0 := x_m$ and the number $p$ of
        eigenvalues to be used.
17. Compute $p$ eigenvectors $u_1, ..., u_p$ of $AM^{-1}$
        and goto 2.
```

3 Model problems and Stabilization techniques

Several three-dimensional CFD model problems are used for performing numerical experiments on the proposed preconditioners. The three-dimensional Navier-Stokes equations are solved for incompressible and compressible flows using stabilized finite element formulations. Also, we consider a two-field coupled problem where the incompressible Navier-Stokes equations are solved along with the Maxwell equations. All these problems are challenging, especially for relatively high Reynolds, Mach or Hartman numbers, for the iterative methods. These problems are good benchmark tests to assess the robustness and accuracy of the proposed preconditioners. In the following, the governing equations and the finite element formulations used are briefly presented.

Let Ω be a bounded domain of R^{nd} $(nd = 3)$ and $\Gamma = \partial\Omega$ be its boundary. The outward unit vector normal to Γ is denoted by $\boldsymbol{n}$. Throughout this paper, we consider a partition of the domain Ω into elements Ω^e where piecewise continuous approximations for the independent variables are employed. We will use the standard notation: a subscript h to a function denotes a finite element approximation.

3.1 Incompressible flows

The conservation equations for momentum and mass for an incompressible flow are written in differential form as:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} - \nu\nabla^2\mathbf{u} + \nabla p = \mathbf{f} \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0 \tag{2}$$

where p, ρ, ν and $\mathbf{f}$ are, respectively, pressure, density, kinematic viscosity and body force. The stabilized discrete formulation corresponding to these equations is established using either the Galerkin-Least-Squares method of Hughes et al. [8,9] or the regularized Galerkin method of Pitkaranta and Brezzi [ ]. In general, the GLS method is more stable than the regularized method for high Reynolds numbers. The discrete solution $(\mathbf{u_h}, p_h)$ is the solution of the variational problem: find $(\mathbf{u_h}, p_h)$ such that for all weighting functions $(\mathbf{v_h}, q_h)$ one has

$$\int_\Omega \mathbf{v_h}\cdot\left(\frac{\partial\mathbf{u_h}}{\partial t} + (\mathbf{u_h}\cdot\nabla)\mathbf{u_h} - \mathbf{f}\right)d\Omega$$
$$+\int_\Omega \nu\,\nabla\mathbf{u_h}\cdot\nabla\mathbf{v_h}\,d\Omega - \int_\Omega p_h\,\nabla\cdot\mathbf{v_h}\,d\Omega$$
$$+\int_\Gamma p_h\,\mathbf{v_h}\cdot\mathbf{n}\,d\Gamma \tag{3}$$
$$-\int_\Gamma(\nu\,\nabla\mathbf{u_h}\cdot\mathbf{n})\cdot\mathbf{v_h}\,d\Gamma + \int_\Omega p_h^*(\nabla\cdot\mathbf{u_h})\,d\Omega$$
$$+\sum_{\Omega_e\in\Omega_h}\tau_1\int_{\Omega_e}\Big((\mathbf{u_h}\cdot\nabla)\mathbf{v_h} - \nu\nabla^2\mathbf{v_h} + \nabla p_h^*\Big)$$
$$\Big((\mathbf{u_h}\cdot\nabla)\mathbf{u_h} - \nu\nabla^2\mathbf{u_h} + \nabla p_h - \mathbf{f}\Big)\,d\Omega_e$$
$$+\sum_{\Omega_e\in\Omega_h}\epsilon_1\int_{\Omega_e}(\nabla p_h^*\cdot\nabla p_h)\,d\Omega_e = 0$$

with thes parameters τ_1 and ϵ_1 functions of the mesh size h and the local Reynolds number $Re_h = ||\mathbf{u}||h/2$. For the GLS method $\epsilon_1 = 0$ and $\tau_1 = \left(\left(\frac{2||\mathbf{u}||}{h}\right)^2 + \left(\frac{4\nu}{h^2}\right)^2\right)^{-1/2} = \frac{h^2}{2\nu\sqrt{Re_h^2+4}}$. For the regularized stabilized technique $\epsilon_1 = \alpha\frac{h^2}{2\nu\sqrt{Re_h^2+4}}$ with $\alpha \le 1$ and $\tau_1 = 0$.

3.2 Magnetohydrodynamic flows

The conservative magnetohydrodynamic system of equations is obtained from the Maxwell equations [10], and is written as:

$$\frac{\partial\mathbf{B}}{\partial t} - \nabla\times(\mathbf{u}\times\mathbf{B}) - \eta\nabla^2\mathbf{B} + \eta\nabla(\nabla\cdot\mathbf{B}) + \nabla q = 0 \tag{4}$$

$$\nabla\cdot\mathbf{B} = 0 \tag{5}$$

where η, $\mathbf{B}$, $\mathbf{u}$ and q are respectively the magnetic diffusivity coefficient, magnetic induction field, velocity field and the scalar Lagrange multiplier for the magnetic free divergence constraint (5). For magnetohydrodynamics, this system is solved along with the incompressible Navier-Stokes equations where the body force is $\mathbf{f} = \frac{\mathbf{1}}{\mu}(\nabla\times\mathbf{B})\times\mathbf{B}$ which represents the Lorentz (Laplace) force due to the interaction between the current density $\mathbf{j} = \frac{\mathbf{1}}{\mu}(\nabla\times\mathbf{B})$ and the magnetic field where μ the magnetic permeability. Since in real applications the magnetic-Reynolds number $Rem = \frac{||\mathbf{u}||L}{\eta}$ is small (where L is a representative length), the discrete stabilized variational formulation used for the magnetic problem (4-5) is similar to the regularized method used in (3) and is stated as: find $(\mathbf{B_h},\ q_h)$ such that for all weighting functions $(\mathbf{B_h^*},\ q_h^*)$ one has

$$\int_\Omega \mathbf{B_h^*}\cdot\frac{\partial\mathbf{B_h}}{\partial t}\,d\Omega +$$

$$\int_\Omega \eta \nabla \mathbf{B}^*_\mathbf{h} \cdot \nabla \mathbf{B_h}\, d\Omega - \int_\Omega \mathbf{B}^*_\mathbf{h} \cdot \nabla \times (\mathbf{u} \times \mathbf{B_h})\, d\Omega$$
$$+ \int_\Omega \mathbf{B}^*_\mathbf{h} \cdot \nabla q_h\, d\Omega + (\tau_2 - \eta) \int_\Omega (\nabla \cdot \mathbf{B_h})\, (\nabla \cdot \mathbf{B}^*_\mathbf{h})\, d\Omega$$
$$- \int_\Gamma \eta\, ((\mathbf{n} \cdot \nabla \mathbf{B_h}) \cdot \mathbf{B}^*_\mathbf{h} - (\mathbf{n} \cdot \mathbf{B}^*_\mathbf{h})\, (\nabla \cdot \mathbf{B_h}))\, d\Gamma$$
$$+ \int_\Omega q^*_h\, (\nabla \cdot \mathbf{B_h})\, d\Omega + \sum_{\Omega_e \in \Gamma_h} \epsilon_2 \int_{\Omega_e} \nabla q_h \cdot \nabla q^*_h\, d\Omega = 0 \tag{6}$$

The parameter ϵ_2 is a function of the mesh size h and local magnetic Reynolds number ($Rem_h = \frac{||\mathbf{u}||h}{\eta}$) given by $\epsilon_2 = \frac{h^2}{2\eta\sqrt{Rem_h^2+4}}$. The parameter τ_2 is a penalization factor given by $\tau_2 = ||\mathbf{u}||h/2$. Note that $\tau_2 - \eta = \eta(\frac{Rem_h}{2} - 1)$. For $2 \leq Rem_h$, $(\tau_2 - \eta)$ tends to $\frac{||\mathbf{u}||h}{2}$ and ϵ_2 tends to $\frac{h}{2||\mathbf{u}||}$.

3.3 Coupling strategy

The fully-coupled problem is described in its weak form by the variational statements (3) and (6). As an alternative to the simultaneous solution approach which would require huge computational requirements in terms of memory and computational time, one can use a segregated scheme. Segregated methods consist of a sequence of solutions of the two-way coupled problems. The process is terminated when a certain convergence criterion is achieved during the iterative process. For time-dependent problems, coupling iterations are actually time steps. The generic form of the segregated algorithms could then be presented as:

<u>**Generic form of the Segregated Algorithm**</u>

```
1- Choose initial solutions $\{U^0\}$ and $\{B^0\}$ and convergence criterion $\epsilon$
2- Time Steps: For i=1,2,...,nsteps
    -2.1. Solve the Magnetic Problem:
        $[K(U^m)]\left\{\Delta B^i\right\} = -\left\{R'(B^{i-1}, U^{i-1})\right\}$
    -2.2. Update the Magnetic solution:
        $\left\{B^i\right\} = \left\{B^{i-1}\right\} + \left\{\Delta B^i\right\}$
    -2.3. Solve the Fluid Problem:
        $J(U^i)\left\{\Delta U^i\right\} = -\left\{R(U^{i-1}, B^n)\right\}$
    -2.4. Update the Fluid Solution:
        $\left\{U^i\right\} = \left\{U^{i-1}\right\} + \left\{\Delta U^i\right\}$
    -2.5. If $||\,\{R(U^i, B^n)\}\,||_2 \leq \epsilon$ Stop
3- End of Time Steps
4- End
```

In the above algorithm, $\{U^m\}$ and $\{B^n\}$ are respectively the fluid and the magnetic solutions at certain previous mth and nth iterations with $m \leq i$ and $n \leq i$ and $K(U^m)$ is the stiffness matrix for the magnetic problem and $J(U^i)$ is the approximate Jacobian matrix for the fluid problems. For instance, setting $m = i - 1$ and $n = i$ gives the simplest segregated algorithm referred to as segregated-coupling algorithm (1). In the segregated algorithm (1), the magnetic field lags one step behind the flow field. However, when the fluid and the magnetic problems are strongly coupled, this strategy may not converge. More tight coupling is possible by setting $n = i$ which means that magnetic field is computed at the same time as the for the flow field. In practice, this is difficult to implement. It is however possible to update the magnetic field whenever the flow field is perturbed, as in Newton-GMRES iterations, so that to keep the magnetic and flow fields tightly coupled . The resulting algorithm, referred to as segregated algorithm (2) involves magnetic-field solves in Newton-GMRES loops. The stiffness matrix $K(U)$ is infact computed and factorized only at every N time steps.

3.4 Compressible flows

The Euler and averaged Navier-Stokes equations are solved along with the Spalart-Allmaras (high Reynolds numbers) turbulence model [11]. The governing system of equations is written in a compact form and in terms of the conservation variables $\boldsymbol{V} = (\rho, \boldsymbol{U}, E, \nu_t)^t$ for the averaged N-S equations or $\boldsymbol{V} = (\rho, \boldsymbol{U}, E)^t$ for Euler equations as

$$\boldsymbol{V}_{,t} + \boldsymbol{F}^{adv}_{i,i}(\boldsymbol{V}) = \boldsymbol{F}^{diff}_{i,i}(\boldsymbol{V}) + \boldsymbol{\mathcal{F}} \tag{7}$$

where $\boldsymbol{U}$ is the vector of specific momentum, ρ the density, E the specific total energy and ν_t the turbulent kinematic viscosity (in the case of turbulent regime), $\boldsymbol{F}_i^{adv}$ and $\boldsymbol{F}_i^{diff}$ are respectively the convective and diffusive fluxes in the ith-space direction, and $\boldsymbol{\mathcal{F}}$ is the source vector. Lower commas denote partial differentiation and repeated indices indicate summation. The diffusive fluxes can be written in the form

$$\boldsymbol{F}_i^{diff} = \boldsymbol{K}_{ij}\boldsymbol{V}_{,j}$$

while the convective fluxes can be represented by diagonalizable Jacobian matrices $\boldsymbol{A}_i = \boldsymbol{F}^{adv}{}_{i,V}$. Recall that any linear combination of these matrices has real eigenvalues and a complete set of eigenvectors.

It is well known that the standard Galerkin finite element formulation often leads to numerical instabilities for convective dominated flows. In the Galerkin-Least-Squares method (or the generalized Streamline Upwind Petrove Galerkin method), the Galerkin variational formulation is modified to include an integral form depending on the local residual $\boldsymbol{\mathcal{R}}(\boldsymbol{V})$ of equation (7), i.e. $\boldsymbol{\mathcal{R}}(\boldsymbol{V}) = \boldsymbol{V}_{,t} + \boldsymbol{F}_{i,i}^{adv}(\boldsymbol{V}) - \boldsymbol{F}_{i,i}^{diff}(V) - \boldsymbol{\mathcal{F}}$, which is identical to zero for the exact solution. The SUPG formulation reads : find $\boldsymbol{V}$ such that for all weighting functions $\boldsymbol{W}$,

$$\sum_e \int_{\Omega^e} [\boldsymbol{W} \cdot (\boldsymbol{V}_{,t} + \boldsymbol{F}_{i,i}^{adv} - \boldsymbol{\mathcal{F}}) + \boldsymbol{W}_{,i}\boldsymbol{F}_i^{diff}]\, d\Omega$$
$$- \int_\Gamma \boldsymbol{W}.\boldsymbol{F}_i^{diff}\, n_i\, d\Gamma \qquad (8)$$
$$+ \sum_e \int_{\Omega^e} (\boldsymbol{A}_i^t \boldsymbol{W}_{,i}) \cdot \boldsymbol{\tau} \cdot \boldsymbol{\mathcal{R}}(\boldsymbol{V})\, d\Omega = 0.$$

In this formulation, the matrix $\boldsymbol{\tau}$ is referred to as *the matrix of time scales.* The SUPG formulation is built as a combination of the standard Galerkin integral form and a perturbation-like integral form depending on the local residual vector [12,13]. The objective is to reinforce the stability *inside the elements.* The matrix of time scales used here is given by $\boldsymbol{\tau} = (\sum |\boldsymbol{B}_i|)^{-1}$, for $i = 1, nd$ where $\boldsymbol{B}_i = \frac{\partial \eta_i}{\partial x_j}\boldsymbol{A}_j$ and $\frac{\partial \eta_i}{\partial x_j}$ are the components of the element Jacobian matrix [ref.]. Recently, a new method referred to as the Edge-Based-Stabilized finite element method (EBS) was introduced [14,15] aiming to stabilize the standard Galerkin method while considering the real characteristics of the flow as computed on the normal direction of element edges. This method has been proven to be stable and more accurate for solving viscous and inviscid compressible flows at all Mach numbers. However, it introduces very strong nonlinearities which could be difficult to solve with standard iterative methods. Let us now briefly describe the EBS formulation, for more details see [ ].

Consider the eigen-decomposition of $\boldsymbol{A}_n = \sum \boldsymbol{A}_i n_i$, $\boldsymbol{A}_n = S_n \Lambda_n S_n{}^{-1}$. Let $Pe_i = \lambda_i h/2\nu$ be the local Peclet number for the eigenvalue λ_i, h a measure of the element size on the element boundary, ν the physical viscosity and $\beta_i = min(Pe_i/3, 1.0)$. We define the matrix $\boldsymbol{B}_n$ by

$$\boldsymbol{B}_n = S_n L S_n{}^{-1} \qquad (9)$$

where L is a diagonal matrix whose entries are given by $L_i = (1+\beta_i)$ if $\lambda_i > 0$; $L_i = -(1-\beta_i)$ if $\lambda_i < 0$ and $L_i = 0$ if $\lambda_i = 0$.

The EBS formulation reads as follows: find $\boldsymbol{V}$ such that for all weighting function $\boldsymbol{W}$,

$$\sum_e \int_{\Omega^e} [\boldsymbol{W} \cdot (\boldsymbol{V}_{,t} + \boldsymbol{F}_{i,i}^{adv} - \boldsymbol{\mathcal{F}}) + \boldsymbol{W}_{,i}\boldsymbol{F}_i^{diff}]\, d\Omega$$
$$- \int_\Gamma \boldsymbol{W}.\boldsymbol{F}_i^{diff} n_i\, d\Gamma \qquad (10)$$
$$+ \sum_e \int_{\Gamma^e} \boldsymbol{W} \cdot \boldsymbol{\tau}_n^{ed} \cdot \boldsymbol{\mathcal{R}}(\boldsymbol{V})\, d\Gamma = 0$$

with $\boldsymbol{\tau}_n^{ed}$ the matrix of *intrinsic length scales* given by

$$\boldsymbol{\tau}_n^{ed} = \frac{h}{2} \cdot \boldsymbol{B}_n \qquad (11)$$

and computed on the element boundaries Γ^e. The function of the SUPG and EBS formulations is to add an amount of artificial viscosity in the characteristic directions. Since these formulations lead to a high-order scheme, high frequency oscillations in the vicinity of shocks or stagnation points can occur. A shock capturing viscosity depending on the discrete residual $\boldsymbol{\mathcal{R}}(\boldsymbol{V})$ as proposed in

[?] is employed. More dissipation is then added in high gradient zones to avoid any undesirable local oscillations.

4 Numerical experiments

Several numerical tests have been carried out to compare the performance of the proposed iterative methods related to the stabilization techniques used. The first problem presented here is the classical lid-driven cavity problem in case of a laminar incompressible flow at a Reynolds number $R_e = 1000$. The second test problem concerns an MHD flow in the lid-driven 3D cavity where a constant transversal magnetic field B_0 is imposed. At last, main results are drawn concerning numerical tests carried out on compressible inviscid and viscous flows using respectively GMRES and FGMRES. For all test problems, a time marching procedure is employed. The Jacobian matrix is computed and factorized using ILUT after each 10 time steps. At each time step, quasi-Newton iterations are performed. All the computations are done using a SUN-Ultra 167 MHz processor.

4.1 Incompressible flow problem

For the incompressible problem, results are obtained for the classical lid-driven cavity problem. The cavity is a cubic domain $(0 \leq x \leq 1, 0 \leq y \leq 1, 0 \leq z \leq 1)$ and the fluid movement in the cavity is induced by the imposed boundary condition $u = u_0$ of the plane $y = 1$. The Dirichlet boundary conditions are imposed for the velocity field, thus expressing the non-slip property at solid walls (Figure 1). The hydrodynamic Reynolds number is set to $Re = 10^3$. The numerical parameters of the solution algorithm are set to the following:

- Time step is $\Delta t = 5.0$.
- Number of time steps is $N = 100$.
- Number of Newton Iterations is set to 5.
- The dropping tolerance parameter in ILUT is set to 10^{-9} and the fill-in parameter is $lfil = 2$.
- The Preconditionning Matrix is computed each 10 time steps.
- For the GMRES algorithm, the number of Krylov basis vectors is $m = 12$ and and the convergence criteria is $||\mathbf{r}||/||\mathbf{r_0}|| \leq 10^{-3}$ where $||\mathbf{r}||$ is the euclidian norm of the residual vector and $\mathbf{r_0}$ is the initial residual vector.
- For the inner FGMRES loop, the number of directions is set to 2.
- For the Deflation algorithm, the number of deflation directions is 6.

Results have been obtained using all algorithms described in section (2). Convergence histories are shown in figures (2) to (4). It is seen that when Galerkin Least Squares formulation is used, all the preconditionning algorithms exhibit almost the same patern of convergence (Figure 2). When the stabilization of the finite element formulation is performed through the regularization term with $\alpha = 1.0$, again the performance of all the preconditionners are almost the same (Figure 3). When $\alpha = 0.1$, FGMRES and DEFLATION algorithms show the best results among the preconditionners. The BROYDEN and the ILUT algorithms lag behind with both algorithms showing stagnation plateau (Figure 4). In terms of CPU time, the FGMRES algorithm is the most efficient preconditionner far ahead of the three other algorithms and especially when associated with the Galerkin Least Squares stabilization (Table 1).

As a first conclusion, FGMRES seems to be robust and efficient compared to the other methods. The deflation method helps significantly in the case of incompressible flows tested so far. Broyden update of the Jacobian improves the preconditioner but in general it is less efficient than FGMRES.

4.2 Coupled MHD problem

For the coupled MHD problem, results are obtained for the lid-driven cavity problem under a constant transversal magnetic field B_0. When the external imposed magnetic field is zero, the problem becomes the classical hydrodynamic lid-driven cavity problem (Figure 1). The velocity field induces a perturbation of the magnetic field in the fluid domain. Moreover, the interaction between the velocity and the magnetic fields creates the electromagnetic Lorentz body forces which are responsible for the change of the structure of the flow. A stronger coupling between the flow field and the magnetic field results as the Hartmann number $H_a = B_0 L \frac{1}{|\mu| \eta \rho \nu}$ increases. The hydrodynamic and magnetic Reynolds numbers are respectively set to $Re = 10^3$ and $Re_m = 1$ and the Hartmann number to $Ha = 20$. The numerical parameters of the solution algorithm are the same as for the incompressible lid-driven cavity problem. The Dirichlet boundary conditions are imposed for both the velocity field and the magnetic field, thus expressing the non-slip property at solid walls and the non-perturbation of the external magnetic field by the induced one.

Again, results have been obtained respectively for ILUT, FGMRES, BROYDEN and DEFLATION algorithms with two versions of the segregated algorithm: algorithm (1) and algorithm (2).

Segregated coupling algorithm (1) Figures (5) to (7) concern the segregated algorithm (3). Figure (7) shows the convergence history when the finite element formulation is stabilized through a regularization term with $\alpha = 0.1$. It is seen that while the ILUT and the BROYDEN algorithms experience a stagnation plateau, the FGMRES and the DEFLATION algorithms reach a zero machine precision. The DEFLATION algorithm reachs this residual few time steps before the FGMRES algorithm. However, the FGMRES algorithm converges in much less CPU time (Table 2). When the regularization factor is set to $\alpha = 1.0$ and since the stabilization term is more important, all the preconditionning algorithms, all the preconditionning algorithms perform very well and reach a machine zero convergence (Figure 6). But again, the FGMRES algorithm performs better (Table 2). In figure 5, the convergence history is shown when the formulation is stabilized through a Galerkin-Least-Squares term. The convergence histories of all the algorithms are quite identical and again FGMRES is the moste efficient s (Table 2). As a conclusion, the FGMRES is the fittest algorithm as it does not seem to be sensitive to the stabilization technique and as it uses the less CPU resources.

Segregated coupling algorithm (2) Results for the segregated algorithm (2) show the same qualitative behavior with the four preconditionners experiencing stagnation plateau in almost all the cases (Figures (8) to (10)). However, when the BROYDEN algorithm is associated with the Galerkin Least Squares formulation, the convergence reaches a level of (10^{-12}). The FGMRES algorithm is again the fastest preconditionner in terms of CPU time.

4.3 Compressible flows

Several numerical tests have been conducted for GMRES as preconditioned by ILUT and for FGMRES on compressible inviscid and turbulent flows using SUPG and EBS formulations. For instance, some tests are carried out for a viscous flow around a flat plate at a Mach number $M_a = 1.9$. As a general trend, we can assert that for a reasonable fill-in parameter (around 15) and for inviscid or laminar flows GMRES performs as well as FGMRES in terms of convergence. At each time step FGMRES uses fewer Krylov directions (Figure 11) than GMRES. But, the preconditoning inner-loop of FGMRES calls for more matrix-vector products (or residual evaluations). In some cases, it is observed that FGMRES is much faster than GMRES, but this is not always the case. As the problem to solve becomes harder as in the case of turbulent flows, we observe that GMRES may diverge. For instance, Figure 12 shows the convergence for a turbulent flow around the Onera-M6 wing at $M_a = 0.8447$, $R_e = 11.710^6$, and an angle of attack of 5.06 degrees. GMRES diverges in this case.

Acknowledgments

This research has been funded by the Natural Sciences and Engineering Research Council of Canada (NSERC), PSIR research program of ETS and by Bombardier.

References

[1] Y. Saad. "Iterative Methods For Sparse Linear Systems", *Numerical Linear Algebra with Applications*, 1:387-402, (1994).

[2] Y. SAAD AND M.H. SCHULTZ. "GMRES: A Generalized Minimun Residual Algorithm For Solving Nonsymmetric Linear Systems", *SIAM J. Sci. Stat. Comput.*, **7**, 856-896, (1986).

[3] Y. Saad. "ILUT : A Dual Treshold Incomplete ILU Factorisation", *Numer. Linear Algebra Appl.*, **1**, 387-402, (1994).

[4] J.E. Dennis and R.B.Schnabel. "Numerical Methods for Unconstrained Optimization and Nonlinear Equations", Prentice-Hall, Inc., New-Jersey, (1983).

[5] R.B. Morgan. "A Restarted GMRES method augmented with eigenvectors", *SIAM J. Matrix Analysis and Applcations*, **16**, 4, 1154-1171, (1995).

[6] Y. Saad. "A Flexible inner-outer preconditionned GMRES Algorithm", *SIAM J. Sci. Comput.*, 14, 461-469, (1993).

[7] A. Chapman and Y. Saad. "Deflated and augmented Krylov subspace Techniques", *Numer. Linear Algebra Appl.*, to appear.

[8] T.J.R. Hughes, L.P. Franca and Hulbert. "A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-duffusive equations", *Computer Methods in Applied Mechanics and Engineering,* **73**, 173-189 (1989).

[9] A. Soulaïmani and Y. Saad. "An arbitrary Lagrangian-Eulerian finite element-method for solving three-dimensional free surface flows", *Computer Methods in Applied Mechanics and Engineering,* **162**, 79-106, (1998).

[10] N. Ben Salah, A. Soulaïmani, W. G. Habashi and M. Fortin. "A Conservative Stabilized Finite Element Method for the magneto-hydrodynamics equations ", *International Journal for Numerical Methods in Fluids*, **29**, 535-554, (1999).

[11] P.R. Spalart and S.R. Allmaras. "A one-equation turbulence model for aerodynamic flows", *La Recherche Aerospatiale*, **1**, 5-21 (1994).

[12] A. Soulaimani and M. Fortin. "Finite Element Solution of Compressible Viscous Flows Using Conservative Variables", *Computer Methods in Applied Mechanics and Engineering* , **118**, 319-350 (1994).

[13] T.J.R. Hughes and Mallet. " A new finite element formulation for computational fluid dynamics: III. The generalized streamline operator for multidimensional advective-diffusive systems", *Computer Methods in Applied Mechanics and Engineering*, **58**, 305-328 (1986).

[14] A. Soulaimani and C. Farhat. "On a finite element method for solving compressible flows", *Proceedings of the ICES-98 Conference: Modeling and Simulation Based Engineering.* Atluri and O'Donoghue editors, 923-928, October (1998).

[15] A. Soulaimani, A. Rebaine and Y. Saad. "An Edge Based Stabilized Finite Element Method For Solving Compressible Flows: Formulation and Parallel Computation", to appear in the Proceedings of the Seventh Aerodynamics Symposium, 46th CASI Annual Conference, May, (1999).

Table 1: Computational performance of different preconditioners for the incompressible problem. CPU time in seconde.

	ILUT	FGMRES	BROYDEN	DEFLATION
GLS	1.00	0.52	0.61	0.60
Regularized method, $\alpha = 1.0$	1.12	0.59	0.78	0.64
Regularized method, $\alpha = 0.1$	1.50	0.80	1.19	0.78

Table 2: Computational performance of different preconditioners for the coupled MHD problem. Segregated algorithm (1). CPU time in seconde.

	ILUT	FGMRES	BROYDEN	DEFLATION
GLS	1.00	0.33	0.66	0.57
Regularized method, $\alpha = 1.0$	1.15	0.40	0.80	0.64
Regularized method, $\alpha = 0.1$	1.53	0.54	1.22	0.79

Table 3: Computational performance of different preconditioners for the coupled MHD problem. Segregated algorithm (2).

	ILUT	FGMRES	BROYDEN	DEFLATION
GLS	1.00	0.36	0.75	0.55
Regularized method, $\alpha = 1.0$	2.76	1.87	2.16	2.84
Regularized method, $\alpha = 0.1$	2.64	0.84	2.22	2.93

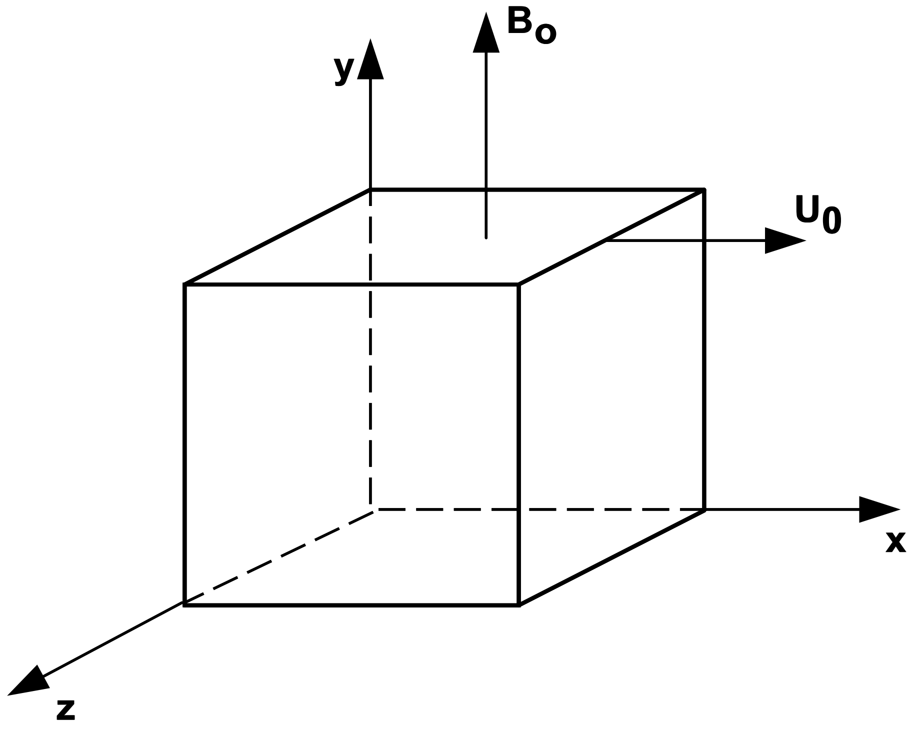

Figure 1: MHD flow. Domain and boundary conditions.

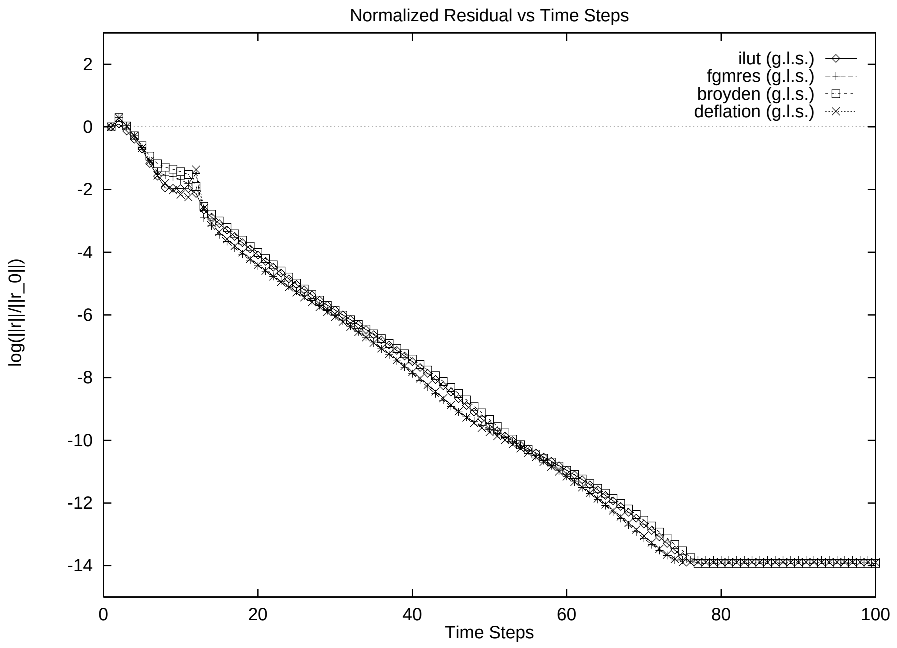

Figure 2: Incompressible flow, lid-driven cavity problem. Convergence history for the GLS method.

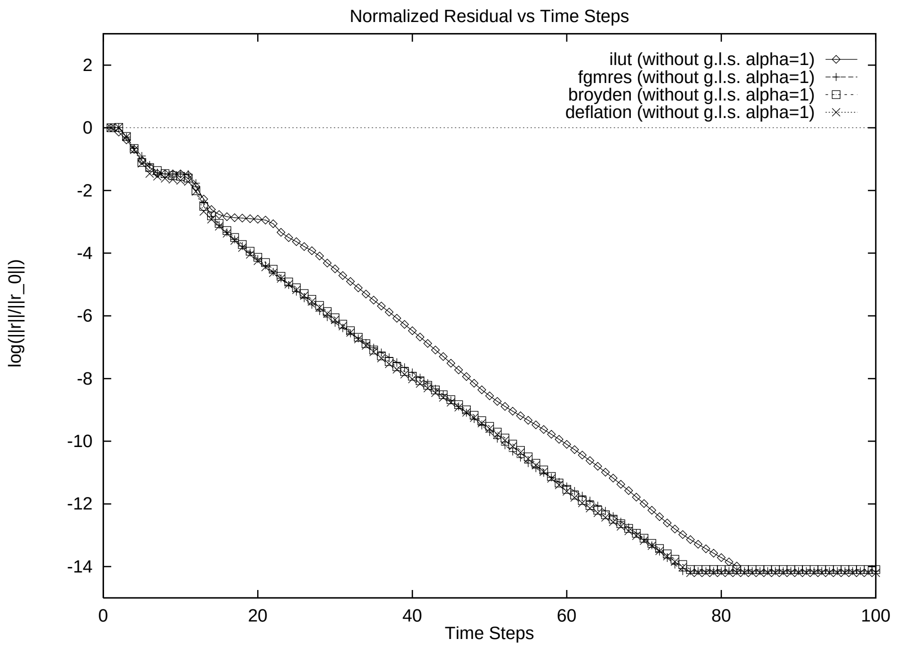

Figure 3: Incompressible flow, lid-driven cavity problem. Convergence history for the regularized method wilth $\alpha = 1.0$

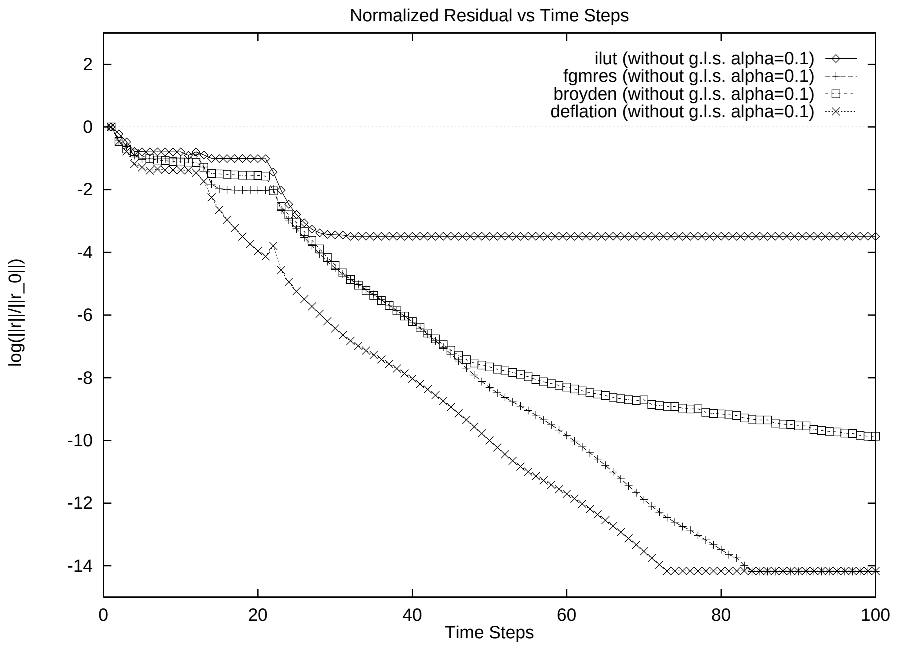

Figure 4: Incompressible flow, lid-driven cavity problem. Convergence history for the regularized method wilth $\alpha = 0.1$

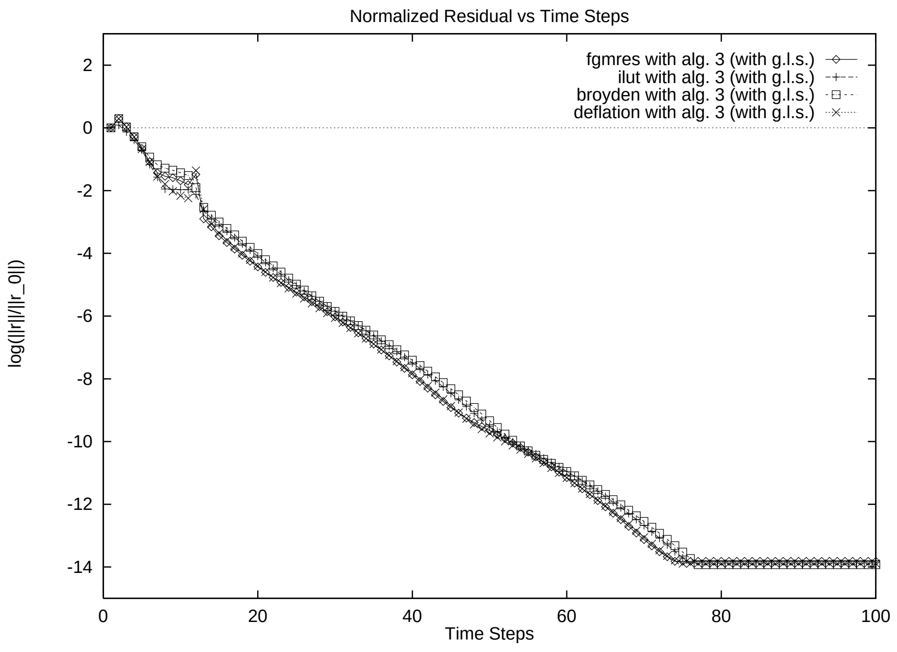

Figure 5: MHD flow. Convergence history for the GLS method and segregated algorithm (1).

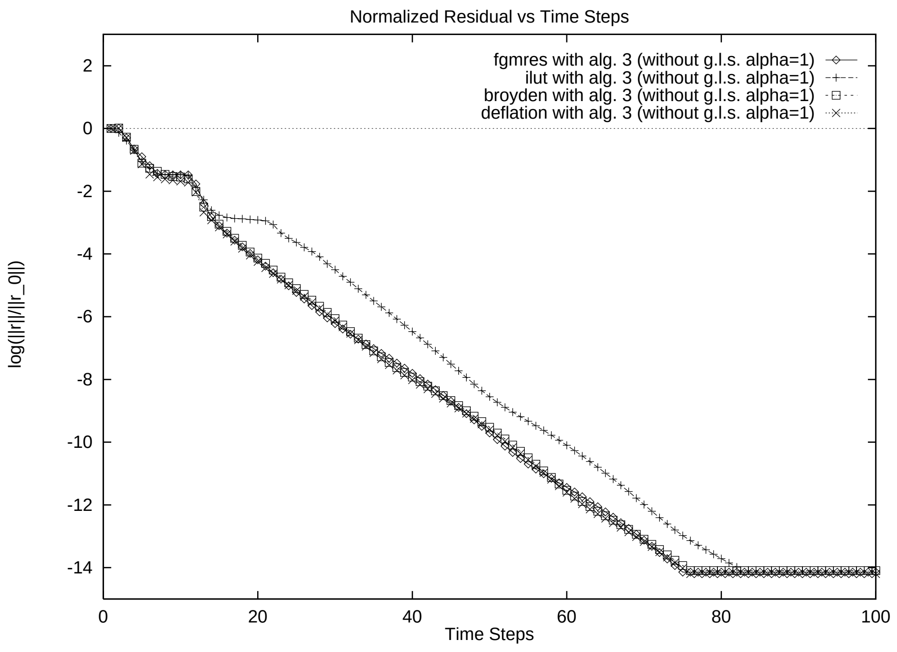

Figure 6: MHD flow. Convergence history for the regularized method wilth $\alpha = 1.0$ and segregated algorithm (1).

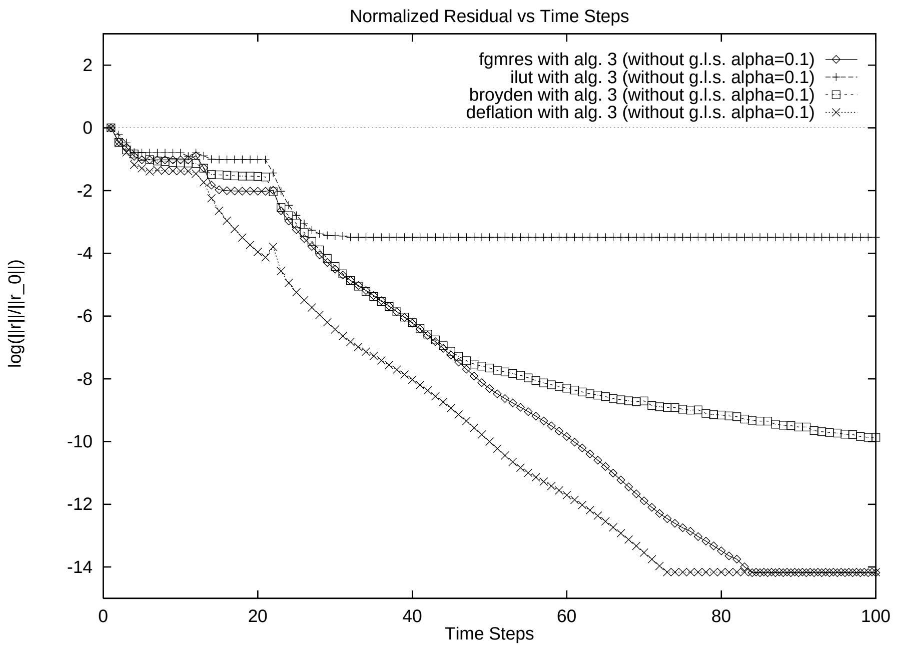

Figure 7: MHD flow. Convergence history for the regularized method wilth $\alpha = 0.1$ and segregated algorithm (1).

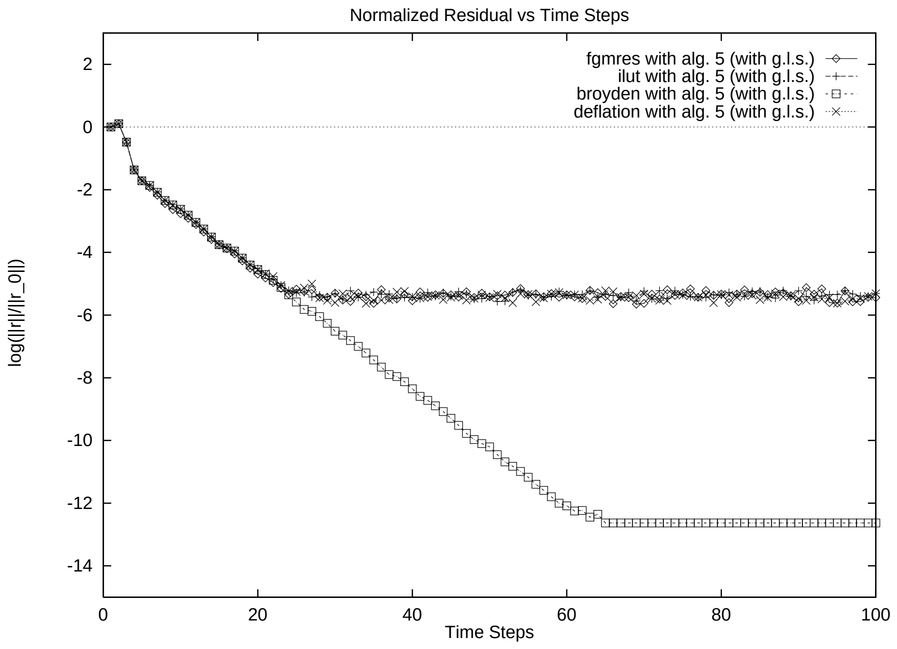

Figure 8: MHD flow. Convergence history for the GLS method and segregated algorithm (2).

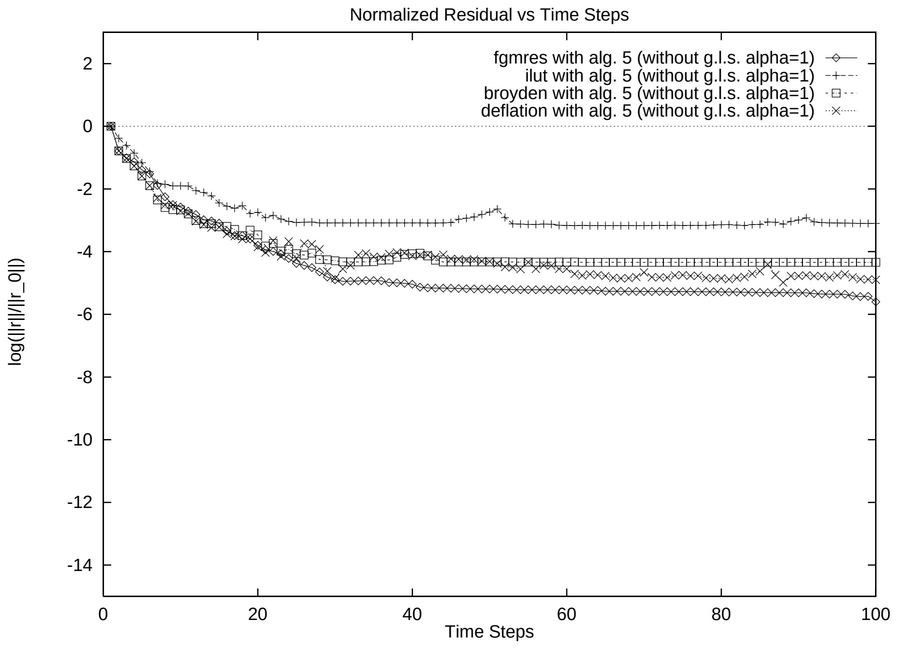

Figure 9: MHD flow. Convergence history for the regularized method wilth $\alpha = 1.0$ and segregated algorithm (2).

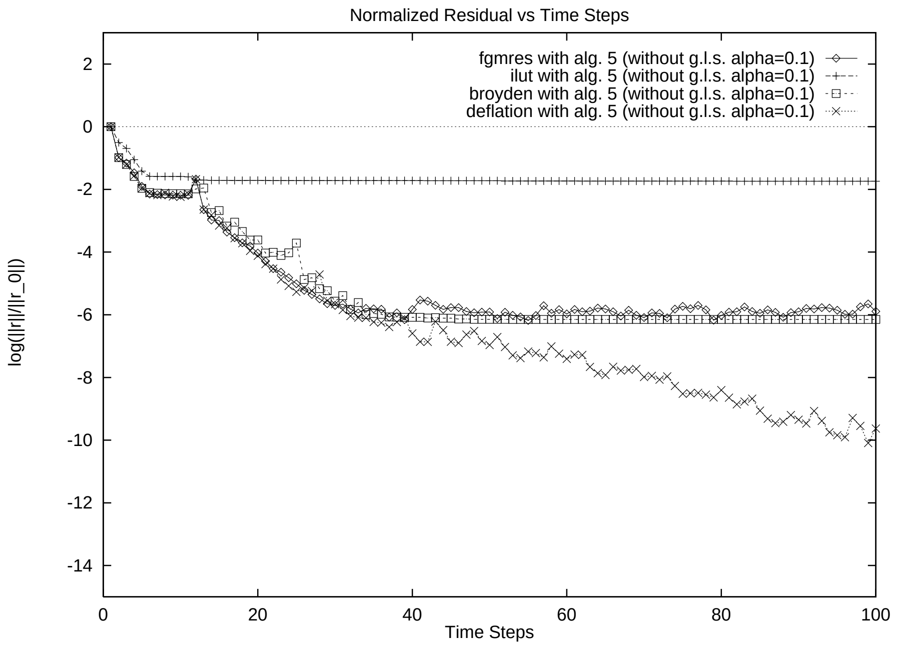

Figure 10: MHD flow. Convergence history for the regularized method wilth $\alpha = 0.1$ and segregated algorithm (2).

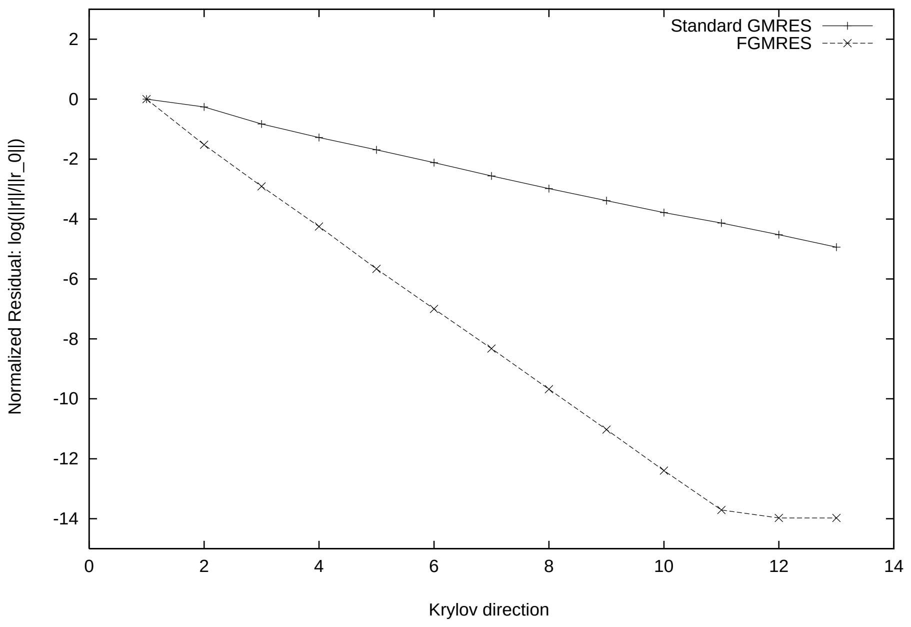

Figure 11: Compressible flow over a flat plate at Mach=0.1.9 and Re=100.

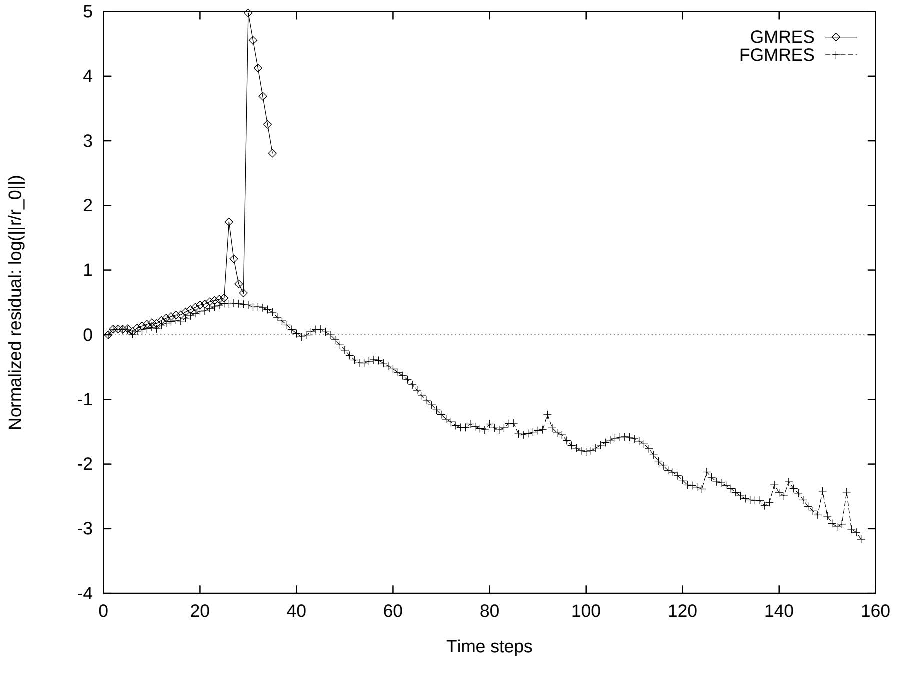

Figure 12: Compressible flow around Onera-M6 wing at Mach=0.85 and angle of attack=5