PARÂMETROS ÓTIMOS DO MÉTODO *MULTIGRID* GEOMÉTRICO CS E FAS PARA PROBLEMAS 2D COM DUAS EQUAÇÕES

Cosmo D. Santiago
cosmo@unibrasil.com.br
Faculdades do Brasil – Unibrasil, Curitiba, PR, Brasil

Carlos Henrique Marchi
marchi@ufpr.br
Universidade Federal do Paraná (UFPR)
Departamento de Engenharia Mecânica
Curitiba, PR, Brasil

Resumo. *O objetivo deste trabalho é estudar a influência do número de equações sobre o desempenho do método multigrid geométrico. São considerados três problemas com condições de contorno de Dirichlet: (1) um problema linear envolvendo uma equação (equação de Laplace); (2) um problema linear envolvendo duas equações (equações de Navier); e (3) um problema não-linear envolvendo duas equações (equações de Burgers). Estas equações são discretizadas com o método de diferenças finitas, em malhas uniformes, com aproximações numéricas de primeira e segunda ordens de acurácia, respectivamente para termos advectivos e difusivos. Os sistemas de equações algébricas são resolvidos com o solver MSI associado ao método multigrid geométrico padrão, com ciclo V, restrição por injeção, prolongação por interpolação bilinear e razão de engrossamento dois. Investiga-se o efeito sobre o tempo de CPU causado por: número de pontos da malha (N); número de iterações internas (ITI) no solver; número de malhas (L); e esquemas CS e FAS. Verificou-se principalmente que o número de equações envolvidas em cada problema não afeta: (1) a taxa de convergência do método multigrid; (2) os valores ótimos de ITI e L; e (3) o fato do esquema FAS ser mais rápido que o CS.*

Palavras-chave: *Diferenças finitas, solvers, Burgers, Laplace, Navier, CFD.*

1. INTRODUÇÃO

Nas aplicações de Dinâmica dos Fluidos Computacionais (CFD) é comum a resolução de problemas complexos cujas soluções analíticas não são conhecidas. As complexidades destes problemas vão desde o número de equações diferenciais envolvidas no modelo, condições de contorno e geometria do domínio, até características do escoamento estabelecidas pelo número de Reynolds nas equações de Navier-Stokes. Aproximações numéricas para as derivadas parciais são usadas para transformar o modelo contínuo em um modelo discreto. Diversas metodologias podem ser aplicadas para este propósito e uma das mais usadas é o Método de Diferenças Finitas (MDF) (Tannehill et al., 1997). A discretização das equações com o MDF resulta em um sistema de equações algébricas do tipo

$$\mathbf{A}^h \vec{\phi}^h = \vec{f}^h \tag{1}$$

onde $\mathbf{A^h}$ é um operador discreto, $\vec{\phi}^h$ é uma aproximação da solução, $\vec{f}^h$ é o vetor de termos independentes e h é a distância entre dois nós consecutivos no domínio discretizado $\Omega^{\mathbf{h}}$. Na prática, a resolução do sistema dado pela Eq. (1) envolve milhões de variáveis, em que os métodos diretos baseados na Eliminação de Gauss (Burden e Faires, 2003) são naturalmente descartados, devido a limitações de memória e alto custo computacional (Zhang, 2003). Neste caso, os métodos iterativos, tais como Gauss-Seidel, Jacobi (Burden e Faires, 2003), MSI (*Modifield Strongly Implicit*) (Schneider e Zedan, 1981) e outros, são mais eficientes que os métodos diretos.

Segundo Ghia et al. (1982), o uso de técnicas iterativas simples para resolver as equações de Navier-Stokes, por exemplo, conduz a uma taxa de convergência muito lenta para obter a solução. Além disso, a taxa de convergência geralmente é fortemente dependente de parâmetros do problema, tal como o número de Reynolds e a quantidade de pontos da malha computacional. Muito esforço tem sido feito visando otimizar técnicas numéricas que podem ser aplicadas para resolver problemas em CFD em malhas cada vez mais refinadas. Nesse sentido, um método que tem chamado bastante atenção de pesquisadores da área de CFD é o método *multigrid* (Briggs et al., 2000).

O método *multigrid*, estudado originalmente por Fedorenko (1964), é uma técnica usada para melhorar a taxa de convergência de métodos iterativos convencionais. A partir do trabalho pioneiro de Brandt (1977), no qual foram feitas análises de vários componentes para problemas de advecção e advecção-difusão, tornou-se evidente na literatura o número crescente de aplicações em que o método *multigrid* é usado para acelerar a taxa de convergência de esquemas numéricos em CFD. Trabalhos publicados por Stüben (1999), Wesseling e Oosterlee (2001) apresentaram bons resultados do método *multigrid* para problemas de dinâmica dos fluidos. Sua vantagem está particularmente em problemas suficientemente grandes, porque a taxa de convergência teórica é independente do tamanho do problema (Hirsch, 1988; Ferziger e Peric, 1999).

A taxa de convergência dos métodos iterativos convencionais na resolução numérica de equações diferenciais é lenta depois de poucas iterações, tornando-se um agravante principalmente quando a malha é muito refinada. Entretanto, alguns destes métodos (Gauss-Seidel e Jacobi) tendem a serem bons suavizadores de erros de alta freqüência em malhas finas, isto é, nas primeiras iterações os modos oscilatórios de erros são rapidamente reduzidos, tornando-se oscilatórios novamente em uma malha mais grossa. Geralmente, para suavizar os erros de alta freqüência calcula-se o resíduo da Eq. (1) através da expressão

$$\vec{R}^h = \vec{f}^h - \mathbf{A}^h \vec{\phi}^h . \tag{2}$$

O método *multigrid* aproveita as propriedades de suavização do erro, cuja filosofia básica é suavizar o erro em malhas grossas e aproximar a solução em malhas finas. Para isso, usa-se uma seqüência de malhas auxiliares, cujas informações são transferidas através de operadores de restrição e de prolongação; mais detalhes podem ser encontrados em Briggs et al. (2000) ou em Trottenberg et al. (2001). Em cada nível de malha resolve-se um sistema de equações com um método iterativo (*solver*) que tem propriedades de suavização de erros oscilatórios (Trottenberg et al., 2001), tal como Gauss-Seidel e Jacobi. Vários algoritmos *multigrid* estão presentes na literatura, e em dois esquemas diferentes: CS (*Corretion Scheme*) e FAS (*Full Approximation Scheme*). Os dois esquemas podem ser associados às metodologias ciclo V, ciclo W, ciclo F, Full Multigrid (FMG) e outros (Briggs et al., 2000; Trottenberg et al., 2001).

Todos os algoritmos *multigrid* são dependentes de parâmetros que influenciam no tempo de CPU. Manipulações nos valores destes parâmetros podem melhorar o desempenho de um fator próximo de 2 entre a pior e a melhor combinação, segundo Ferziger e Peric (1999). Mas para Trottenberg et al. (2001), experiências com o método *multigrid* mostram que as escolhas de seus parâmetros, tais como suavizadores, número de passos de suavização, ciclos, malha grossa, operadores de restrição e prolongação, podem ter uma forte influência na taxa de convergência do algoritmo.

A maior vantagem do método *multigrid* é que o tempo de CPU é proporcional ao número de pontos da malha. Entretanto, segundo Ferziger e Peric (1999) isso não ocorre quando o *multigrid* é aplicado a problemas que envolvam as equações de Navier-Stokes com altos números de Reynolds. Em relação aos melhores algoritmos *multigrid* atuais, Brandt et al. (2002) mencionam que o tempo de CPU necessário para resolver um problema de CFD pode ser reduzido ainda de 10 a 100 vezes.

Wesseling e Oosterlee (2001) destacam alguns desafios para o *multigrid* geométrico, como por exemplo, a solução das equações de Navier-Stokes e problemas com perturbações singulares. Brandt (1998) resume as principais dificuldades computacionais com o método *multigrid*, onde destaca os escoamentos recirculantes, linhas de correntes não alinhadas com a malha, características da malha, pontos de estagnação, discretização e relaxação próxima a choques e contornos. Algumas possíveis soluções são discutidas.

O método *multigrid* pode ser encontrado em duas formulações, conhecido como *multigrid* geométrico e *multigrid* algébrico. O *multigrid* geométrico (Wesseling e Oosterlee, 2001) é mais adequado para problemas com geometrias regulares, enquanto que o algébrico para problemas com geometrias irregulares (Stüben, 1999).

Muitos dos parâmetros do método *multigrid* foram estudados e otimizados por Pinto et al. (2005) em problemas lineares unidimensionais de advecção, advecção-difusão e equação de Burgers. Em Pinto e Marchi (2006) podem ser encontradas várias análises com a equação de Laplace bidimensional, incluindo os esquemas CS e FAS e vários *solvers,* com a razão de engrossamento padrão (2). Tannehill et al. (1997) afirmam que o melhor desempenho do método *multigrid* é obtido com diversas malhas e sugerem o uso de 5 ou 6 malhas para o problema de Laplace 2D com malha de 129x129. Santiago e Marchi (2007) analisaram alguns parâmetros e o efeito do acoplamento entre as equações na performance do *multigrid* usando um modelo linear com duas equações. Mesquita e De-Lemos (2004) usaram uma variante do esquema CS na resolução das equações de Navier-Stokes.

Neste trabalho propõe-se otimizar e verificar se alguns parâmetros do método *multigrid* são afetados pelo número de equações diferenciais de dois modelos bidimensionais: equações de Navier e de Burgers. A equação de Laplace, modelo com uma equação, é usada como referência para as comparações entre os valores ótimos obtidos nas simulações numéricas. Sobre o tempo de CPU, estuda-se a influência causada pelo número de pontos da malha (N), número de iterações internas (ITI), e número de níveis de malhas (L). Uma análise sobre os esquemas CS e FAS é apresentada para as equações de Navier e de Laplace, que são modelos

lineares. As conclusões do presente trabalho poderão ser estendidas para importantes aplicações que envolvem duas ou mais equações, tais como as equações de Navier-Stokes nas formulações vorticidade-velocidade (Guj e Stella, 1993; Meitz e Fasel, 2000) e função corrente-vorticidade (Wesseling, 1984; Zhang, 2003).

Este artigo está organizado da seguinte forma: na seção 2 são apresentados os modelos matemáticos e numéricos, e o método *multigrid*. Na seção 3, são apresentados os resultados e a discussão dos mesmos. E, finalmente, na seção 5, apresenta-se a conclusão do trabalho.

2. MODELOS MATEMÁTICOS E NUMÉRICOS

Os modelos matemáticos são definidos em um domínio contínuo $\Omega = \{(x, y) \in R^2 : 0 \le x, y \le 1\}$. Em todos os modelos são consideradas as condições de contorno de Dirichlet.

2.1 Equação de Laplace

O problema linear de condução de calor bidimensional, no sistema de coordenadas cartesianas, com propriedades constantes, pode ser representado por (Ferziger e Peric, 1999)

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0\,, \tag{3}$$

com as seguintes condições de contorno:

$$T(x,0) = T(0,y) = 0, \quad T(x,1) = x, \quad T(1,y) = y\,, \tag{4}$$

onde T representa a temperatura. A solução analítica do problema definido pelas Eqs. (3) e (4) é dada por

$$T(x,y) = xy. \tag{5}$$

2.2 Equações de Navier

No sistema de coordenadas cartesianas, para um estado plano de tensões, as equações do problema de termoelasticidade linear bidimensional permanente, para corpos elásticos, cujos materiais são homogêneos e isotrópicos, a partir da lei de Hooke, podem ser reduzidas a duas equações diferenciais parciais escritas em termos dos deslocamentos. Para obter estas equações, as relações elásticas tensão-deformação são substituídas nas equações de equilíbrio. Assim, obtém-se as clássicas equações de Navier para a elasticidade em meio contínuo (Timoshenko e Goodier, 1970; Salençon, 2001), descritas por

$$C_\lambda \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) + \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 2C_\lambda \alpha \frac{\partial T}{\partial x} + S^u\,, \tag{6}$$

$$C_\lambda \frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) + \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} = 2C_\lambda \alpha \frac{\partial T}{\partial y} + S^v\,, \tag{7}$$

onde $C_\lambda = (1+\lambda)/(1-\lambda)$, sendo λ a razão de Poisson, α o coeficiente de expansão térmica e u e v são os deslocamentos nas direções coordenadas x e y, respectivamente. O coeficiente de expansão térmica e a razão de Poisson são constantes nesse modelo matemático. O campo de temperaturas é obtido da solução analítica de um problema de difusão bidimensional (Greenberg, 1998), dada por

$$T(x,y) = sen(\pi x)\frac{senh(\pi y)}{senh(\pi)}. \tag{8}$$

Propõe-se a seguinte solução analítica para o sistema dado pelas Eqs. (6) e (7),

$$u(x,y) = \beta sen(\pi x)\frac{\left(e^{2x}-1\right)\left(e^{y}-1\right)}{\left(e^{2}-1\right)(e-1)}, \tag{9}$$

$$v(x,y) = \gamma x y^2, \tag{10}$$

para a qual $\gamma = 1$ e $\beta = 0{,}01$ são parâmetros. As soluções analíticas dadas pelas Eqs. (5), (9) e (10) foram obtidas com o Método das Soluções Fabricadas (Roache, 2002; e Shih et al., 1989). Em geral, a utilização da solução fabricada é usada para verificar a existência de eventuais erros de programação e erros numéricos. A substituição das derivadas das funções u e v, Eqs. (9) e (10), nas Eqs. (6) e (7), resulta nas parcelas dos termos fontes S^u e S^v, respectivamente. Com base nas Eqs. (9) e (10), aplicam-se as condições de contorno para os deslocamentos *u* e *v*, que está ilustrado na Fig. 1.

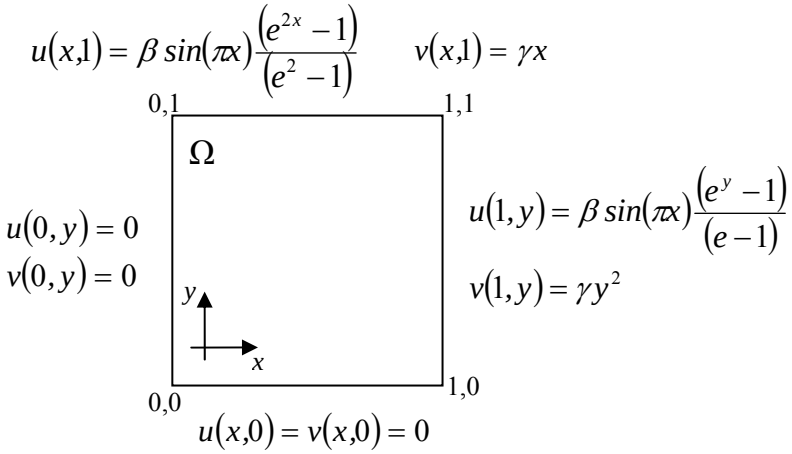


Figura 1 - Domínio de cálculo e condições de contorno para as equações de Navier.

2.3 Equações de Burgers

As equações de Navier-Stokes bidimensionais com algumas simplificações reduzem-se às equações de Burgers, ou simplesmente às equações da quantidade do movimento linear (QML). Estas equações, devido a suas fortes não-linearidades, constituem-se em um importante problema-teste. Uma revisão bibliográfica de pesquisas em que esse problema é usado como *benchmark* pode ser encontrada em Efe (2006). No modelo considerado nesse estudo, a pressão é dada analiticamente por Shih et al. (1989); por isso, as dificuldades relacionadas ao cálculo da pressão não estão presentes. As equações de Burgers com

propriedades constantes, em regime permanente, escritas no sistema de coordenadas cartesianas, são dadas por

$$\frac{\partial u^2}{\partial x}+\frac{\partial (vu)}{\partial y}=-\frac{\partial p}{\partial x}+\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}, \tag{11}$$

$$\frac{\partial (uv)}{\partial x}+\frac{\partial v^2}{\partial y}=-\frac{\partial p}{\partial y}+\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}+B, \tag{12}$$

onde p é a pressão (Shih et al., 1989), u e v são as componentes das velocidades nas direções coordenadas x e y, respectivamente, e B é um termo fonte.

A solução analítica para u e v é dada por (Shih et al., 1989)

$$u(x,y)=8\left(x^4-2x^3+x^2\right)\left(4y^3-2y\right), \tag{13}$$

$$v(x,y)=-8\left(4x^3-6x^2+2x\right)\left(y^4-y^2\right). \tag{14}$$

Trata-se do famoso problema da cavidade quadrada, em que a tampa movimenta-se na direção x com velocidade conhecida e um termo fonte $B(x,y,\mathrm{Re})$, com Reynolds unitário, atua sobre o escoamento, como mostrado na Fig. 2.

Com base nas Eqs. (13) e (14), aplicam-se as condições de contorno para as velocidades u e v, assumindo zero em todos os contornos, exceto no superior para a velocidade u. A Fig. 2 ilustra a aplicação das condições de contorno no domínio unitário.

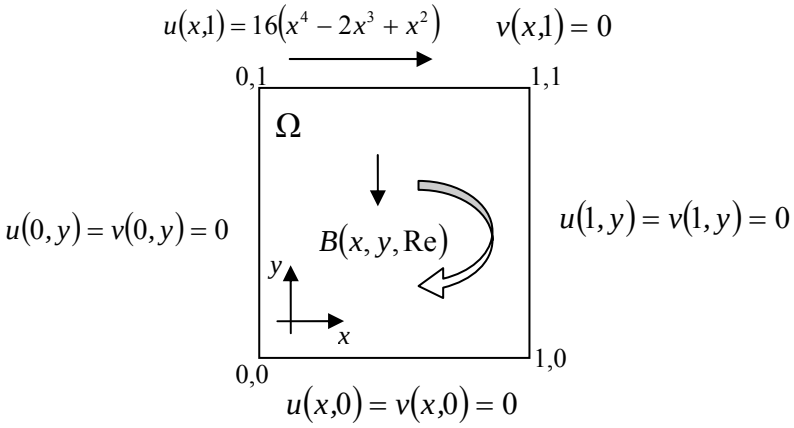


Figura 2 - Domínio de cálculo e condições de contorno para as equações de Burgers.

2.4 Método *multigrid*

Os problemas testes definidos nas seções anteriores são resolvidos no domínio $\Omega=\{0\le x\le 1;\ 0\le y\le 1\}$, que é particionado em um número de nós (ou pontos) dado por

$$N = N_x N_y \tag{15}$$

onde N_x e N_y são o número de nós nas direções x e y, respectivamente. Cada nó da malha é definido como

$$(x_i, y_j) = ((i-1)h_x, (j-1)h_y), \text{ com } h_x = \frac{1}{N_x - 1} \text{ e } h_y = \frac{1}{N_y - 1} \tag{16}$$

em que $i = 1, \cdots, N_x$, $j = 1, \cdots, N_y$, h_x e h_y são o tamanho dos elementos da malha nas direções *x* e *y*, isto é, a distância entre dois nós consecutivos. Em malhas uniformes, tem-se $h_x = h_y = h$.

As equações governantes são discretizadas com o Método de Diferenças Finitas (MDF), em um sistema de coordenadas cartesianas e malha uniforme nas duas direções. As derivadas de segunda ordem (termos difusivos) e derivadas cruzadas, Eqs (6) e (7), são aproximadas com o esquema de diferença central (Central Difference Scheme - CDS), que possui erro de truncamento de segunda ordem. Os termos advectivos que aparecem do lado esquerdo das equações (11) e (12) são aproximados com o esquema de diferença a montante (*Upwind Difference Scheme* - UDS), que possui erro de truncamento de primeira ordem. A discretização resulta na equação geral dada por

$$a_{P,P}\phi_{i,j} + a_{P,n}\phi_{i,j+1} + a_{P,s}\phi_{i,j-1} + a_{P,w}\phi_{i-1,j} + a_{P,e}\phi_{i+1,j} = b_P^{\phi}, \tag{17}$$

que é um sistema de equações algébricas com estrutura pentadiagonal. A variável ϕ representa a propriedade física envolvida na aproximação numérica. Os termos $a_{P,e}$, $a_{P,w}$, $a_{P,n}$ e $a_{P,s}$ são os coeficientes de conexão entre o nó *P* e seus vizinhos leste, oeste, norte e sul, respectivamente, e b_P^{ϕ} é o termo independente, ou termo fonte associado ao ponto *P*.

Neste trabalho, o método *multigrid* geométrico é adotado com ciclo V e esquemas CS e FAS. O algoritmo CS é implementado para os dois problemas lineares (equações de Laplace e de Navier), enquanto que o algoritmo FAS é implementado para os três problemas testes. Detalhes sobre os algoritmos CS e FAS podem ser encontrados em Trottenberg et al. (2001).

No esquema CS, a Eq. (1) é resolvida somente na malha mais fina, e o resíduo dado pela Eq. (2), é transferido para a malha grossa subseqüente onde é resolvida uma equação residual. Na malha mais grossa o erro é interpolado para corrigir a aproximação na malha fina subseqüente. No esquema FAS, além do resíduo, também a aproximação da solução é transferida para a malha grossa mais próxima. Na malha mais grossa, a correção é interpolada e usada para aproximar a solução na malha fina subseqüente. Os sistemas de equações são resolvidos em todos os níveis (Briggs et al., 2000). A principal diferença em relação ao esquema CS é que no FAS não se resolve a equação residual, mas sim equações com aproximações para a solução na malha grossa. De acordo com Trottenberg et al. (2001), para problemas lineares, os esquemas FAS e CS são equivalentes. Pinto e Marchi (2006) mostraram que para o problema de difusão bidimensional (equação de Laplace), com o método MSI, o esquema FAS é mais rápido que o esquema CS, mesmo em problemas lineares. No presente trabalho, as informações entre as malhas são transferidas por injeção na restrição, e por interpolação bilinear na prolongação (Trottenberg et al., 2001).

O método de relaxação ideal para ser usado com o método *multigrid* é aquele que possui boas propriedades de suavização de erros oscilatórios, por exemplo, o método de Gauss-Seidel (Briggs et al., 2000). O método MSI foi testado por Pinto e Marchi (2006) e apresentou resultados melhores que o Gauss-Seidel. Neste trabalho optou-se pelo método MSI como método de relaxação ou *solver*. Adotou-se também a razão de engrossamento 2 (valor padrão na literatura), isto é, a quantidade de elementos em uma malha qualquer é o dobro da

quantidade da malha imediatamente mais grossa. Outras razões de engrossamento foram analisadas por Pinto et al. (2005) para problemas unidimensionais de difusão, advecção-difusão e equação de Burgers.

2.5 Erro de iteração e tempo de CPU

Os esquemas CS e FAS foram implementados na linguagem FORTRAN 95, usando o compilador Visual Compaq FORTRAN 6.6. As simulações foram feitas em um microcomputador com processador Intel Core 2 Duo 2,66 GHz, 2 GB RAM, com precisão dupla.

O critério de convergência usado para interromper o processo iterativo é o seguinte: quando a média da norma l_1 (Ferziger e Peric, 1999) do erro de iteração for menor do que 10^{-12} interrompe-se os cálculos, onde

$$\bar{l}_1[E_k(\phi)]_k = \frac{\sum_{i=1}^{N}\left|(\phi_{k\to\infty} - \phi_k)_i\right|}{N}, \tag{18}$$

$\phi_{k\to\infty}$ é a solução exata do sistema de equações (para cada malha, ela é obtida ao se executar o programa até a eliminação do erro de iteração, isto é, até ser atingido o erro de arredondamento de máquina), ϕ_k é a solução na iteração k, N é o número total de nós na malha, E_k é o erro na iteração k, $\bar{l}_1$ denota a média da norma do na iteração k e i denota cada nó da malha.

Em todos os problemas adotou-se a estimativa inicial com valor nulo para todas as variáveis dependentes. O tempo de CPU foi medido com a função TIMEF da biblioteca PORTLIB do FORTRAN 95. A incerteza desta função é aproximadamente ± 0.05 s (Pinto et al., 2005).

2.6 Algoritmos *multigrid*

A seguir são descritos os algoritmos usados neste trabalho. O algoritmo MG-1Eq descreve o procedimento padrão do método *multigrid*, com ciclo V, para uma equação e qualquer número de malhas ($L > 1$). O parâmetro α define o tipo de esquema a ser usado: para $\alpha = 0$, o algoritmo MG-1Eq executa o esquema CS; e, para $\alpha = 1$, o esquema FAS.

Algoritmo MG-1Eq. Esquemas CS e FAS para uma equação:
MG-1Eq$\left(\alpha, ITI, u^h, b^h\right)$

Início

1. Suavizar $A^h u^h = b^h$ ITI vezes com estimativa inicial u_0^h;
2. Calcular $r^{2h} = I_h^{2h}\left(b^h - A^h u^h\right)$, $u_0^{2h} = \alpha\, I_h^{2h} u^h$;
3. Definir $b_u^{2h} = \alpha\, A^h u_0^{2h} + r^{2h}$;
4. Suavizar $A^{2h} u^{2h} = b^{2h}$ ITI vezes com estimativas inicial u_0^{2h};
5. Calcular $r^{4h} = I_{2h}^{4h}\left(b^{2h} - A^{2h} u^{2h}\right)$, $u_0^{4h} = \alpha\, I_{2h}^{4h} u^{2h}$;
6. Definir $b^{4h} = \alpha\, A^{4h} u_0^{4h} + r_u^{4h}$;
7. Suavizar $A^{4h} u^{4h} = b^{4h}$ ITI vezes com estimativas inicial u_0^{4h};

$\vdots$

8. Resolver $A^{Lh}u^{Lh} = b^{Lh}$;
9. Calcular o erro $e^{Lh} = u^{Lh} - \alpha u_0^{Lh}$;

$\vdots$

10. Corrigir a solução: $u_0^{4h} \leftarrow u^{4h} + I_{8h}^{4h} e^{8h}$;
11. Suavizar $A^{4h}u^{4h} = b^{4h}$ ITI vezes com estimativa inicial u_0^{4h};
12. Corrigir a solução: $u_0^{2h} \leftarrow u^{2h} + I_{4h}^{2h} e^{4h}$;
13. Suavizar $A^{2h}u^{2h} = b^{2h}$ ITI vezes com estimativa inicial u_0^{2h};
14. Corrigir a solução: $u_0^{h} \leftarrow u^{h} + I_{2h}^{h} e^{2h}$;
15. Suavizar $A^{h}u^{h} = b^{h}$ ITI vezes com estimativa inicial u_0^{h};

Fim

MG-1Eq$\left(\alpha, ITI, u^h, b^h\right)$

Algoritmo MG-2Eq. Esquemas CS e FAS para duas equações:

MG-2Eq$\left(\alpha, ITI, u^h, v^h, b_u^h, b_v^h\right)$

Início

1. Suavizar $A_u^h u^h = b_u^h$ e $A_v^h v^h = b_v^h$ ITI vezes com estimativa inicial u_0^h e v_0^h;
2. Calcular $r_u^{2h} = I_h^{2h}\left(b_u^h - A_u^h u^h\right)$, $r_v^{2h} = I_h^{2h}\left(b_v^h - A_v^h v^h\right)$, $u_0^{2h} = \alpha I_h^{2h} u^h$ e $v_0^{2h} = \alpha I_h^{2h} v^h$;
3. Definir $b_u^{2h} = \alpha A_u^{2h} u_0^{2h} + r_u^{2h}$ e $b_v^{2h} = \alpha A_v^{2h} v_0^{2h} + r_v^{2h}$;
4. Suavizar $A_u^{2h}u^{2h} = b_u^{2h}$ e $A_v^{2h}v^{2h} = b_v^{2h}$ ITI vezes com estimativas inicial u_0^{2h} e v_0^{2h};
5. Calcular $r_u^{4h} = I_{2h}^{4h}\left(b_u^{2h} - A_u^{2h} u^{2h}\right)$, $r_v^{4h} = I_{2h}^{4h}\left(b_v^{2h} - A_v^{2h} v^{2h}\right)$, $u_0^{4h} = \alpha I_{2h}^{4h} u^{2h}$ e $v_0^{4h} = \alpha I_{2h}^{4h} v^{2h}$;
6. Definir $b_u^{4h} = \alpha A_u^{4h} u_0^{4h} + r_u^{4h}$ e $b_v^{4h} = \alpha A_v^{4h} v_0^{4h} + r_v^{4h}$;
7. Suavizar $A_u^{4h}u^{4h} = b_u^{4h}$ e $A_v^{4h}v^{4h} = b_v^{4h}$ ITI vezes com estimativas inicial u_0^{4h} e v_0^{4h};

$\vdots$

8. Resolver $A_u^{Lh}u^{Lh} = b_u^{Lh}$ e $A_v^{Lh}v^{Lh} = b_v^{Lh}$;
9. Calcular o erro $e_u^{Lh} = u^{Lh} - \alpha u_0^{Lh}$ e $e_v^{Lh} = v^{Lh} - \alpha v_0^{Lh}$;

$\vdots$

10. Corrigir a solução: $u_0^{4h} \leftarrow u^{4h} + I_{8h}^{4h} e_u^{8h}$ e $v_0^{4h} \leftarrow v^{4h} + I_{8h}^{4h} e_v^{8h}$;
11. Suavizar $A_u^{4h}u^{4h} = b_u^{4h}$ e $A_v^{4h}v^{4h} = b_v^{4h}$ ITI vezes com estimativa inicial u_0^{4h} e v_0^{4h};
12. Corrigir a solução: $u_0^{2h} \leftarrow u^{2h} + I_{4h}^{2h} e_u^{4h}$ e $v_0^{2h} \leftarrow v^{2h} + I_{4h}^{2h} e_v^{4h}$;
13. Suavizar $A_u^{2h}u^{2h} = b_u^{2h}$ e $A_v^{2h}v^{2h} = b_v^{2h}$ ITI vezes com estimativa inicial u_0^{2h} e v_0^{2h};
14. Corrigir a solução: $u_0^{h} \leftarrow u^{h} + I_{2h}^{h} e_u^{2h}$ e $v_0^{h} \leftarrow v^{h} + I_{2h}^{h} e_v^{2h}$;
15. Suavizar $A_u^{h}u^{h} = b_u^{h}$ e $A_v^{h}v^{h} = b_v^{h}$ ITI vezes com estimativa inicial u_0^{h} e v_0^{h};

Fim

MG-2Eq$\left(\alpha, ITI, u^h, v^h, b_u^h, b_v^h\right)$

No algoritmo MG-2Eq, a restrição para cada uma das equações é feita separadamente e é uma generalização direta do caso de uma equação, tanto para o CS quanto para o FAS. As

equações são suavizadas nos passos 1, 4 e 7. Quando $\alpha = 0$ (esquema CS), o termo do lado direito (passos 3 e 6) é o próprio resíduo restrito e a estimativa inicial é nula (passos 2 e 5). No esquema FAS ($\alpha = 1$), além do resíduo, a solução aproximada também é restrita para a malha grossa (passos 2 e 5) e usada para corrigir o termo do lado direito (passos 3 e 6) dos sistemas de equações. Este procedimento é repetido até a malha mais grossa (do passo 7 ao 8).

Para a interpolação, o procedimento também é análogo ao caso de uma equação, ou seja, a interpolação e a adição da correção são feitas separadamente para cada equação, como pode ser observado nos passos 10, 12 e 14. Na malha mais grossa é feita uma aproximação do erro (passo 9) e depois interpolado. Nos passos 10, 12 e 14, a solução de cada equação é corrigida com o erro interpolado e usada como estimativa inicial para obter nova aproximação da solução (passos 11 e 13). Na malha mais fina (passo 15), os sistemas são resolvidos e é verificada a convergência.

Os algoritmos descritos acima percorrem apenas um ciclo V. Um procedimento recursivo (Trottenberg et al., 2001) pode ser usado para que seja percorrido o número de ciclos necessários até atingir a convergência.

3. RESULTADOS E DISCUSSÃO

Aproximadamente 250 simulações foram feitas para avaliar a influência, sobre o tempo de CPU, do número de iterações internas, número de malhas e número de nós da malha (desde a malha mais grossa, 5x5, até a malha mais fina, 2049x2049 nós).

Tem-se o valor ótimo de um parâmetro quando a solução do problema é obtida no menor tempo de CPU para valores fixos dos demais parâmetros. Assim, denota-se por $ITI_{ótimo}$ o número ótimo de iterações internas no *solver*, e por $L_{ótimo}$ o número ótimo de níveis de malhas.

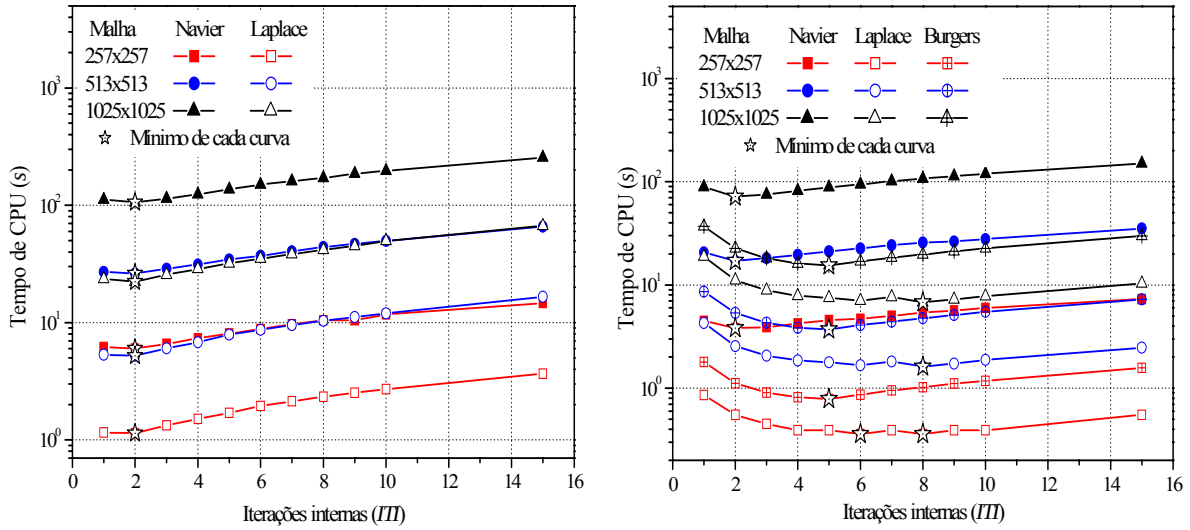
3.1 Iterações Internas (*ITI*)

A Fig. 3 ilustra a influência do número de iterações internas (*ITI*) sobre o tempo de CPU para os esquemas CS e FAS, e os três problemas. Em cada curva é indicado o valor do *ITI* que resulta no menor tempo de CPU, através do símbolo "estrela". A Fig. 3a mostra os resultados para o esquema CS; nota-se que o menor tempo de CPU, para os problemas de Laplace e de Navier, foi obtido com duas suavizações, ou seja, $ITI_{ótimo} = 2$ em qualquer malha.

Na Fig. 3b estão mostrados os resultados para o esquema FAS, onde inclui-se também os resultados para as equações de Burgers. Nota-se que o número de suavizações é mantido igual a 2 para as equações de Navier. Entretanto, para a equação de Laplace obteve-se, nas três malhas testadas, $ITI_{ótimo} = 8$; e, na malha mais grossa ($N = 257 \times 257$), obteve-se também $ITI_{ótimo} = 6$. Para as equações de Burgers, tem-se $ITI_{ótimo} = 5$ para os três tamanhos de malhas testadas. Em todos os casos analisados, deve-se notar que quando se diminui ou aumenta o valor de *ITI* em relação a $ITI_{ótimo}$, aumenta o tempo de CPU. Este aumento do tempo de CPU pode ser significativo dependendo do valor usado para *ITI*.

O comportamento do tempo de CPU, observado para o modelo com uma equação (Laplace), é igual ou muito semelhante ao observado nos modelos com duas equações (Navier e Burgers). Mesquita e De-Lemos (2004) encontraram *ITI* = 3 como o valor ótimo do número de iterações internas para as equações bidimensionais de Navier-Stokes e equação da energia, usando o esquena CS e até 4 malhas. Tannehill et al. (1997) resolveram a equação de Laplace bidimensional com o esquema CS, malha 129×129 nós com o suavizador Gauss-Seidel e

encontraram *ITI* = 3 ou 4, como parâmetro minimizador do tempo de CPU. Pinto e Marchi (2006) resolveram a equação de Laplace bidimensional e testaram três *solvers*; para o MSI, o mesmo usado neste trabalho, encontraram $ITI_{ótimo}$ = 1 ou 2 para o esquema CS, e 4 para o FAS.



(a) esquema CS (b) esquema FAS

Figura 3 - Número de iterações internas.

3.2 Número de níveis de malha (*L*)

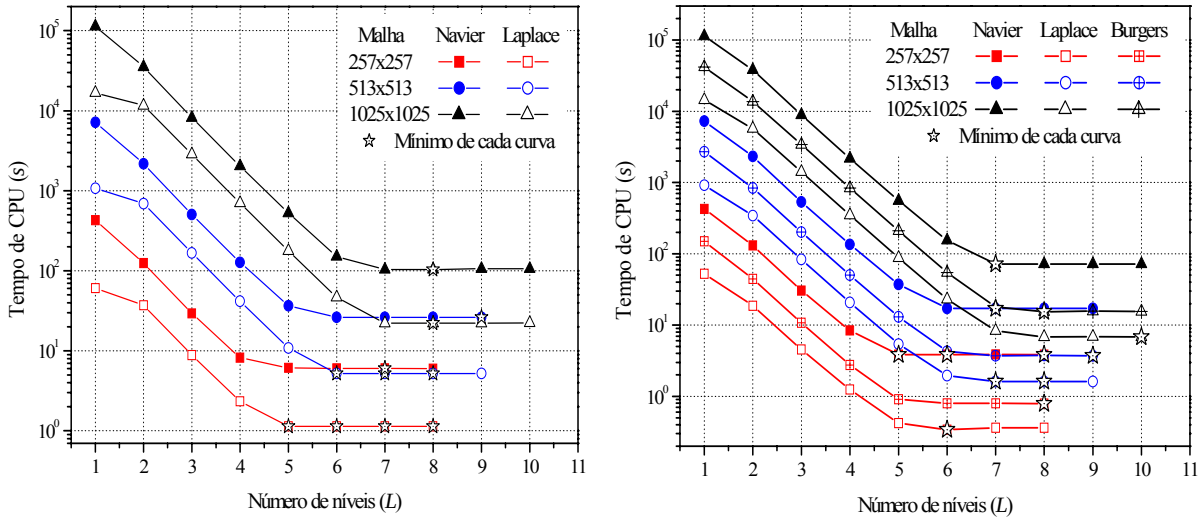
O estudo sobre a influência do número de níveis de malha (*L*) leva em consideração o número de iterações internas ótimo obtido na seção anterior, visto que a intenção é otimizar o tempo de CPU. A Fig. 4 mostra a influência do número de níveis sobre o tempo de CPU para os esquemas CS e FAS, e os três problemas. Para cada malha, o problema foi resolvido com um número de níveis de malha L tal que $1 \le L \le L_{máximo}$. O parâmetro $L_{máximo}$ representa o número máximo possível de malhas que se pode usar para uma dada malha, com a malha mais grossa tendo apenas um nó interno. Por exemplo, se N = 513 x 513 nós, as malhas são de 513x513, 257x257, 129x129, 65x65, 33x33, 17x17, 9x9, 5x5 e 3x3 nós; neste exemplo, portanto, $L_{máximo} = 9$.

A Fig. 4a mostra as curvas para as equações de Navier e de Laplace com o esquema CS. E, a Fig. 4b mostra as curvas para as equações de Navier, Laplace e de Burgers com o esquema FAS. Pode-se notar que o comportamento de cada curva e o $L_{ótimo}$ se assemelham tanto para as equações de Navier, com os esquemas CS e FAS, quanto para as equações de Burgers (FAS), em qualquer tamanho de malha. Para os dois esquemas (CS e FAS), as três malhas e os três problemas, o valor ótimo do número de níveis de malha é $L_{ótimo} = L_{máximo} - (3$ ou 4).

Para $L < L_{ótimo}$, o tempo de CPU aumenta significativamente. Mas para $L > L_{ótimo}$, o tempo de CPU é praticamente o mesmo. Para a malha mais fina, com o esquema FAS e equações de Navier, por exemplo, observa-se que a diferença do tempo de CPU entre $L_{máximo}$ e o $L_{ótimo}$ é de menos de 0,3%; e no caso das equações de Burgers essa diferença é de 0,1%.

Os resultados obtidos aqui para a equação de Laplace estão coerentes com resultados da literatura. Pinto e Marchi (2006) recomendam usar $L = L_{máximo}$ para a equação de Laplace

bidimensional. Tannehill et al. (1997) sugerem $L = 5$ ou 6 para o mesmo problema com $N = 129 \times 129$. E, Rabi e De-Lemos (2001) sugerem não menos que 4 níveis no problema bidimensional de advecção-difusão. Esta análise mostra que o parâmetro, número de níveis (L), no método *multigrid*, não se modifica nos problemas com duas equações para os esquemas CS e FAS.



(a) esquema CS

(b) esquema FAS

Figura 4 - Número de níveis.

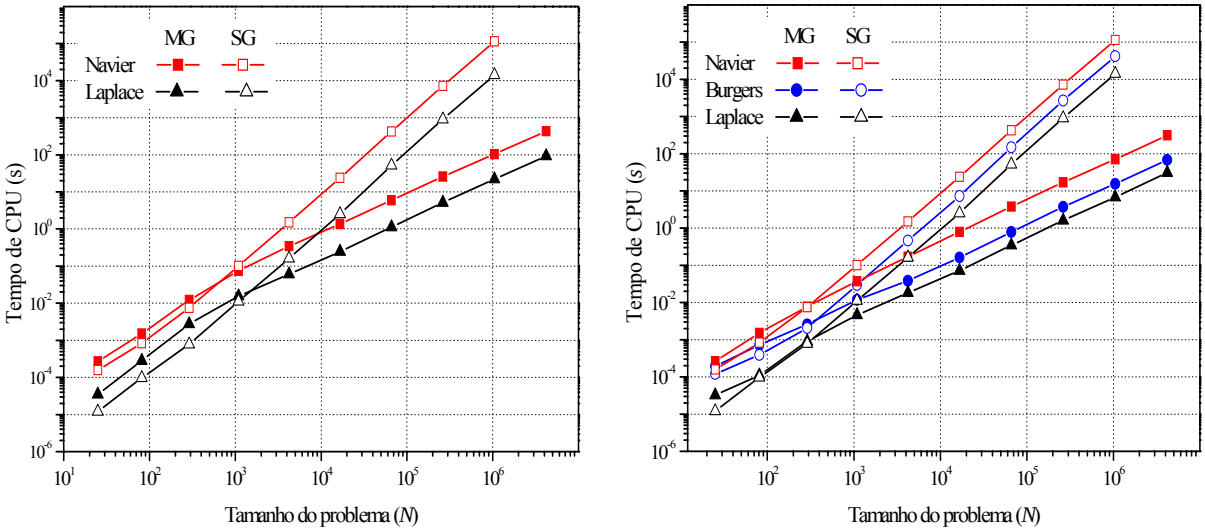
3.3 Tamanho do problema (*N*)

Na análise da influência do tamanho do problema, isto é, do número de incógnitas no sistema de equações, sobre o tempo de CPU são considerados o número ótimo de iterações internas ($ITI_{ótimo}$) e o número ótimo de níveis de malha ($L_{ótimo}$) obtidos nas seções anteriores. Nessa análise consideram-se problemas de tamanho N = 5x5 até o maior suportado pela memória física da máquina, N = 2049x2049. Para comparação, são mostrados também os resultados obtidos com o método *singlegrid* (malha única) e *solver* MSI para os três problemas. Os resultados são apresentados na Fig. 5.

Para malhas muito grossas, o tempo de CPU é muito próximo de zero, tanto no método *multigrid* quanto no *singlegrid*. Neste caso, adotou-se uma metodologia para obter o tempo de CPU que elimine o máximo possível do erro devido à incerteza de medição da função TIMEF. A idéia principal é obter uma incerteza absoluta aceitável para o tempo de CPU. Assim, para todas as malhas cujo tempo de uma simulação foi menor que 10 segundos, adotou-se um ciclo mais externo para que o programa fizesse um número necessário de simulações para atingir 10 segundos ou mais. O tempo de uma simulação é a média do tempo gasto em todas as simulações. Por exemplo, no problema de Navier, com esquema CS, para a malha de tamanho N = 5x5, simulada com o método *multigrid* e *solver* MSI, foram necessárias 38175 simulações para atingir 10,02 segundos, resultando no tempo médio de 0,000262 segundos para uma simulação.

Quanto mais fina a malha, isto é, quanto maior o número de nós N, maior é a vantagem do método *multigrid* em relação ao método *singlegrid* (Ferziger e Peric, 1999). Esta propriedade pode ser observada na Fig 5, em que o tempo de CPU do método *multigrid* é menor que o do método *singlegrid* a partir da malha N = 33x33 com o esquema CS; e com o

esquema FAS, a partir da malha N = 17x17. A partir destas malhas, fica mais evidente também, Figs. 5a e 5b, o crescimento linear do tempo de CPU com N. Na malha N = 513x513, com o esquema FAS, para as equações de Navier, o método *multigrid* é cerca de 1600 vezes mais rápido que o método *singlegrid;* e, se considerarmos as equações de Burgers essa vantagem passa para 2702 vezes.



(a) esquema CS

(b) esquema FAS

Figura 5 - Esforço computacional.

Para os dois problemas com duas equações (Navier e Burgers), pode-se ver nas Figs. 5a e 5b que o tempo de CPU do *multigrid* apresenta crescimento na mesma razão que o problema com apenas uma equação (Laplace). Portanto, o número de equações e sua complexidade (linear ou não-linear) não afetam o desempenho do método *multigrid*. Alguns resultados para os mesmos parâmetros estudados no presente trabalho, para as equações de Navier, são apresentados também em Santiago e Marchi (2007).

3.4 Análise dos *solvers*

Para determinar a ordem do *solver* associado aos métodos e o comportamento da curva em função do tempo de CPU, realizou-se um ajuste de curva geométrico de mínimos quadrados, cuja função considerada é dada por

$$t_{CPU}(N) = cN^{p} \tag{19}$$

onde p representa a ordem do *solver* associado ao método empregado, ou a inclinação de cada curva da Fig. 5, e c é um coeficiente que depende do método e do *solver*. Quanto mais próximo da unidade estiver o valor de p, melhor é o desempenho do algoritmo usado. Teoricamente, o método *multigrid* ideal tem $p = 1$, o que significa que o tempo de CPU cresce linearmente com a malha (N).

Na Tab. 1 estão os valores de p obtidos para os três problemas e esquemas CS e FAS para N > 33x33. Os resultados confirmam que o tempo de CPU do método *multigrid* e *solver* MSI cresce quase linearmente com o aumento de N. O valor de p é pouco afetado pelo número de

equações e sua complexidade (linear ou não-linear), e pelo tipo de esquema (CS ou FAS); o efeito relevante é o uso ou não de *multigrid.* Os valores de *p* próximos a dois para o método *singlegrid* concordam com os valores teóricos (Burden e Faires, 2003), bem como os valores próximos à unidade obtidos para o método *multigrid*.

Tabela 1. Valor de *p* da Eq. (19) com o *solver* MSI

Problema	*Singlegrid*	*Multigrid*	
		FAS	CS
Equação de Laplace	2,06	1,08	1,06
Equações de Navier	2,01	1,15	1,05
Equações de Burgers	1,92	1,06	---

3.5 Comparação entre os esquemas CS e FAS

A Fig. 6 mostra o tempo de CPU para os problemas lineares, equação de Laplace e equações de Navier, ambas com os esquemas CS e FAS, para N = 5x5 até 2049x2049 nós. O objetivo dessa análise é verificar qual dos dois esquemas resulta no menor tempo de CPU. Para um dado N, pode-se notar que o esquema FAS resulta no menor tempo de CPU em cada problema. Lembra-se que o esquema CS é indicado pela literatura para problemas lineares e o algoritmo FAS para problemas não-lineares. Portanto, embora as equações de Laplace e Navier sejam lineares, o esquema FAS é mais rápido do que o CS.

Para as equações de Navier com N = 513x513, por exemplo, o esquema FAS é cerca 1,5 vez mais rápido que o esquema CS. No caso da equação de Laplace, na mesma malha, o esquema FAS é aproximadamente 3,2 vezes mais rápido que o esquema CS. Essa vantagem é praticamente a mesma em qualquer N, visto que as curvas da Fig. 6 são praticamente paralelas para N > 33x33.

No caso de problemas lineares, Brandt (1977) mostra preferência pelo esquema CS em relação ao esquema FAS. De acordo com Brandt, cada ciclo iterativo do esquema FAS é mais caro computacionalmente se comparado ao esquema CS, devido à quantidade de cálculos necessários no esquema FAS. Pinto (2006) fez comparações entre os esquemas CS e FAS para a equação de Laplace e também concluiu que o esquema FAS é mais rápido que o esquema CS.

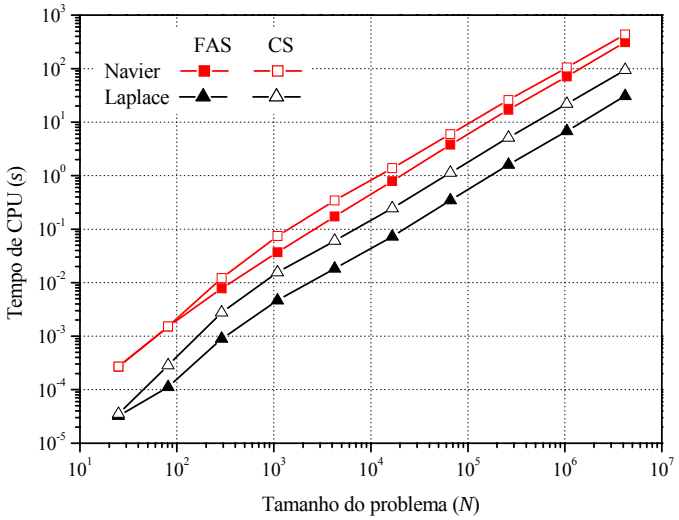


Figura 6 – Comparação entre os esquemas CS e FAS para dois problemas lineares.

4. CONCLUSÃO

Neste trabalho verificou-se a influência de alguns parâmetros do método *multigrid* geométrico sobre o tempo de CPU necessário para resolver três problemas com condições de contorno de Dirichlet: (1) um problema linear envolvendo uma equação (equação de Laplace); (2) um problema linear envolvendo duas equações (equações de Navier); e (3) um problema não-linear envolvendo duas equações (equações de Burgers). Os parâmetros considerados foram: número de iterações internas (*ITI*); número de níveis de malha (*L*); número de pontos da malha computacional (*N*); esquemas CS (*Corretion Scheme*) e FAS (*Full Approximation Scheme*); e a influência do número de equações diferenciais. As equações foram discretizadas com o método de diferenças finitas, em malhas uniformes, com aproximações numéricas de primeira e segunda ordens de acurácia, respectivamente para termos advectivos e difusivos. Os sistemas de equações algébricas foram resolvidos com o método MSI associado ao método *multigrid* geométrico padrão, com ciclo V, restrição por injeção, prolongação por interpolação bilinear e razão de engrossamento dois.

Com base nos resultados deste trabalho, verificou-se que:

1) Para o esquema CS, e as equações de Laplace e de Navier, obteve-se $ITI_{ótimo} = 2$. Para o esquema FAS, obteve-se $ITI_{ótimo} = 2$ para as equações de Navier; $ITI_{ótimo} = 5$ para as equações de Burgers e $ITI_{ótimo} = 6$ ou 8 para a equação de Laplace. Quando se diminui ou aumenta o valor de *ITI* em relação a $ITI_{ótimo}$, aumenta o tempo de CPU. Este aumento do tempo de CPU pode ser significativo dependendo do valor usado para *ITI*.
2) Para os dois esquemas (CS e FAS), as três malhas e os três problemas, o valor ótimo do número de níveis de malha é $L_{ótimo} = L_{máximo} - (3 \text{ ou } 4)$. Para $L < L_{ótimo}$, o tempo de CPU aumenta significativamente. Mas para $L > L_{ótimo}$, o tempo de CPU é praticamente o mesmo. Ao se utilizar $L_{máximo}$ em vez de $L_{ótimo}$, tem-se no máximo um aumento de 0,3% no tempo de CPU.
3) A ordem *p* da Eq. (19) varia entre 1,92 e 2,06 para o método *singlegrid*, dependendo do problema, e entre 1,05 e 1,15 para o método *multigrid*, dependendo do problema e do esquema.
4) Para problemas lineares, o esquema FAS resulta em menor tempo de CPU do que o esquema CS.
5) Para o método *multigrid*, o número de equações envolvidas em cada problema não afeta:
   5.1) A ordem *p*, da Eq. (19).
   5.2) Os valores ótimos de *ITI* e *L*.
   5.3) O fato do esquema FAS ser mais rápido que o CS.

Agradecimentos

O primeiro autor agradece ao Laboratório de Experimentação Numérica (LENA), do Departamento de Engenharia Mecânica da UFPR, pela disponibilidade de infra-estrutura necessária para o desenvolvimento do presente trabalho e também aos colegas do laboratório por sugestões e discussões. Os autores agradecem ao MCT/CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico, do Brasil) pelo apoio financeiro. O segundo autor é bolsista do CNPq.

REFERÊNCIAS

Brandt, A., 1977. Multi-Level adaptive solutions to boundary-value problems. *Mathematics of Computation*, v. 31, p. 333-390.

Brandt, A., 1998. Barriers to Achieving Textbook Multigrid Efficiency (TME) in CFD, *ICASE Interim Report,* n. 32, NASA/CR.

Brandt, A., & Thomas, J. L., 2002. Recent Advances in Efficiency for Computational Simulations. *ICASE Report,* n. 16, NASA/CR .

Briggs, W.L., & Henson, V.E., Mccormick, S.F., 2000. *A Multigrid Tutorial*. 2ª ed., SIAM.

Burden, R. L., & Faires, J. D., 2003. *Análise Numérica*. São Paulo, Pioneira Thomson Learning.

Efe, M., Ö., 2006. Observer-based boundary control for 2D Burgers equation. *Transactions of the Institute of Measurement and Control*, v. 28, p. 177-185.

Fedorenko, R. P., 1964. On the Speed of Convergence of an Iteration Process. *USSR Computational Mathematics and Mathematical Physics*, v. 4 (3), p. 559-564.

Ferziger, J. H., & Peric, M., 1999. *Computational Methods for Fluids Dynamics*. 2ª ed., Springer.

Ghia, U., Ghia, N., & Shin, C.T., 1982. High-Re solutions for incompressible flow using the Navier-Stokes equations and a Multigrid method. *Journal of Computational Physics*, v. 48, p. 387-411.

Greenberg, M. D., 1998. *Advanced Engineering Mathematics*. 2ª ed., Prentice Hall, Inc., New Jersey.

Guj, G., & Stella, F., 1993. A Vorticity-Velocity Method for the Numerical Solution of 3D Incompressible Flows. *Journal of Computational Physics,* v. 106, p. 286 – 298.

Hirsch, C., 1988. *Numerical Computational of Internal and External Flows*. v.1, Wiley.

Meitz, H. L., & Fasel, H. F., 2000. A Compact-Difference Scheme for the Navier–Stokes Equations in Vorticity–Velocity Formulation. *Journal of Computational Physics*, v. 157, p. 371 – 403.

Mesquita, M. S. & De-Lemos, M. J. S., 2004. Optimal Multigrid Solutions of Two-dimensional Convection-conduction Problems. *Applied Mathematics and Computation*, v. 152, p. 725-742.

Pinto, M. A. V., 2006. *Comportamento do Multigrid Geométrico em Problemas de Transferência de Calor*. Tese de Doutorado. Universidade Federal do Paraná (Método Numéricos em Engenharia), Curitiba, Brasil.

Pinto, M. A. V., & Marchi, C. H., 2006. Efeito dos Parâmetros do Método Multigrid CS e FAS sobre o tempo de CPU para a Equação de Laplace Bidimensional. *Proceedings of ENCIT*, Paper Cit 06-0348.

Pinto, M. A. V., Santiago, C. D., & Marchi, C. H., 2005. Effect of Parameters of a Multigrid Method on CPU Time for One-dimensional Problems. *Proceedings of COBEM.*

Rabi, J. A., & De-Lemos, M. J. S, 2001. Optimization of convergence acceleration in multigrid numerical solutions of conductive-convective problems. *Applied Mathematics and Computation*, v. 124, p. 215-226.

Roache, P. J., 2002. **Code verification by the method of manufactured solutions.** *Journal of Fluids Engineering*, v. 124, p. 4-10.

Salençon, J., 2001. *Handbook of Contnuum Mechanics: General Concepts Thermoelasticity*. Berlin, New York : Springer

Santiago, C. D., & Marchi, C. H., 2007. Optimum parameters of a geometric multigrid for a two-dimensional problem of two-equations. *Proceedings of COBEM*.

Schneider, G. E., & Zedan, M., 1981. A Modified Strongly Implicit Procedure for Numerical Solution of Field Problems. *Numerical Heat Transfer*, Vol. 4, pp. 1-19.

Shih, T. M., Tan, C. H., & Hwang, B. C., 1989. Effects of grid staggering on numerical scheme. *International Journal for Numerical Methods in Fluids*, v. 9, pp. 193-212.

Stüben, K., 1999. Algebraic Multigrid (AMG): An Introduction with Applications. em: *GMD-Report* 70.

Tannehill, J. C., Anderson, D. A., & Pletcher, R.H., 1997. *Computational fluid mechanics and heat transfer*. Washington: Taylor & Francis; 2 ed., p. 40, 46-48, 151, 166, 172.

Timoshenko, S. P., & Goodier, J. N., 1970. *Theory of elasticity*. Japan: McGraw-Hill.

Trottenberg, U., Oosterlee C. W., & Schüller A., 2001. *Multigrid*. Academics Press.

Wesseling, P., 1984. Multigrid solution of the Navier-Stokes equations in the vorticity-streamfunction. *Proceedings of a GAMM-Seminar*, 145-154.

Wesseling, P., & Oosterlee, C. W., 2001. Geometric Multigrid with Applications to Computational Fluid Dynamics. *Journal of Computation and Applied Mathematics*, v. 128, p. 311-334.

Zhang, J., 2003. Numerical Simulation of 2D Square Driven Cavity Using Fourth-Order Compact Finite Difference Schemes. *Computers and Mathematics with Applications*, v. 45, p. 43-52.