Proceedings of the 11th Brazilian Congress of Thermal Sciences and Engineering -- ENCIT 2006
Braz. Soc. of Mechanical Sciences and Engineering -- ABCM, Curitiba, Brazil, Dec. 5-8, 2006

Paper CIT06-0348

EFEITO DE PARÂMETROS DO MÉTODO MULTIGRID CS E FAS SOBRE O TEMPO DE CPU PARA A EQUAÇÃO DE LAPLACE BIDIMENSIONAL

Márcio Augusto Villela Pinto
Departamento de Matemática e Estatística, Universidade Estadual de Ponta Grossa, Ponta Grossa, PR, Brasil
marciovp@demec.ufpr.br

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

Resumo. *Sobre o tempo de CPU necessário para resolver numericamente a equação de Laplace bidimensional, verifica-se o efeito causado por: número de nós, de iterações internas e de malhas, três solvers (Gauss-Seidel, MSI e ADI), e esquemas de correção (CS) e de aproximação completa (FAS) com multigrid geométrico e ciclo V. O método de diferenças finitas é usado para discretizar a equação diferencial com um esquema de 2ª ordem de acurácia. Verificou-se que: (1) o esquema FAS é mais rápido do que o CS; (2) o solver MSI é mais rápido do que Gauss-Seidel e ADI; (3) o número ótimo de iterações internas é 1 para o CS e entre 3 e 5 para o FAS; e (4) recomenda-se usar o número máximo possível de malhas.*

Palavras-chave: solver, CFD, diferenças finitas, transferência de calor, métodos numéricos.

1. Introdução

Modelos matemáticos na dinâmica dos fluidos computacional surgem em fenômenos físicos que envolvem fluidos em movimento, com ou sem troca de calor (Fortuna, 2000; Maliska, 2004). Estes modelos matemáticos, em geral, não têm soluções analíticas conhecidas. Buscam-se então soluções numéricas transformando-se o modelo contínuo em um modelo discreto. Um método de discretização muito usado é o método de diferenças finitas (Golub e Ortega, 1992; Tannehill *et al.*,1997), onde, em problemas bidimensionais, o domínio $(x, y) \in \Re^2 : \ 0 \le x, y \le 1$ é particionado em um número de incógnitas (ou número de pontos), dado por:

$$N = N_x N_y , \tag{1}$$

onde N_x e N_y são o número de incógnitas nas direções coordenadas *x* e *y*, respectivamente.

Isto introduz uma malha com os pontos

$$(x_i, y_j) = ((i-1)h_x, (j-1)h_y) \text{ , com } h_x = \frac{1}{N_x - 1} \text{ e } h_y = \frac{1}{N_y - 1} , \tag{2}$$

onde $i = 1,..., N_x$, $j = 1,..., N_y$ e h_x e h_y os comprimentos de cada elemento nas direções coordenadas *x* e *y*, respectivamente. Neste caso, estabelece-se uma malha com elementos de tamanho h_x por h_y que se denota por Ω^h.

A discretização desses modelos matemáticos conduz a grandes sistemas de equações algébricas do tipo

$$\mathbf{A}\vec{v} = \vec{f} , \tag{3}$$

onde **A** é uma matriz quadrada, $\vec{f}$ é o vetor independente e $\vec{v}$ é o vetor de incógnitas. A estrutura da matriz **A** depende do método e das aproximações numéricas usados para discretizar o modelo matemático.

Várias técnicas numéricas têm sido estudadas para resolver o sistema dado pela Eq. (3) com o menor custo computacional e a solução a mais próxima possível da exata (sem erros de iteração, Ferziger e Peric, 1999). A resolução por métodos diretos não é recomendável, visto que na prática, a matriz dos coeficientes é muito grande e o custo da inversão da matriz é alto (Golub e Van Loan, 1989). Para problemas de grande porte os métodos iterativos são mais adequados (Burden e Faires, 1997). O método do gradiente conjugado (Burden e Faires, 1997; Golub e Ortega, 1992), introduzido por Hestenes e Stiefeld (1952), usa técnicas que são mais específicas para geometrias simples e é um método sensível ao condicionamento da matriz.

O método *multigrid*, proposto originalmente por Fedorenko (1964), é atualmente um método numérico muito usado para resolver iterativamente sistemas de equações do tipo da Eq. (3). A idéia básica é usar um conjunto de malhas e executar alternativamente iterações em cada nível de malha e soluções aproximadas desta equação em malhas mais grossas (Briggs *et al.*, 2000). São usados operadores para transferir informações da malha fina para a malha imediatamente mais grossa (processo chamado de restrição) e da malha grossa para a malha imediatamente mais fina (processo de prolongação). Briggs *et al.* (2000) trabalharam com a razão de engrossamento $r = 2$ afirmando ser uma prática universal e que $r \neq 2$ não traz vantagens. Em cada malha o sistema de equações é resolvido com um método iterativo com propriedades de reduzir rapidamente os erros oscilatórios (propriedades de suavização). Com este conceito, vários trabalhos (Brandt, 1977; Stüben, 1999; Wesseling e Oosterlee, 2001; Moro, 2004; Pinto *et al.*, 2005a e 2005b), apresentaram bons resultados numéricos para problemas de dinâmica dos fluidos. Os resultados de Fedorenko (1964) mostram que a taxa de velocidade de convergência com o uso da técnica *multigrid* é muito melhor que a dos métodos iterativos puros. O objetivo do método *multigrid* é acelerar a convergência a fim de reduzir o tempo de CPU necessário para resolver a Eq. (3). Os melhores desempenhos do método *multigrid* são obtidos em problemas totalmente elípticos (Wesseling, 1992), ou seja, problemas dominados pela difusão; e os menores em problemas dominados pela advecção (Ferziger e Peric, 1999).

O método *multigrid* pode ser aplicado a malhas estruturadas, conhecido como *multigrid* geométrico (Wesseling e Oosterlee, 2001), bem como a malhas não-estruturadas, conhecido como *multigrid* algébrico (Stüben, 2001). Em Wesseling e Oosterlee (2001) são apontados vários desafios para o *multigrid* geométrico, como: a solução das equações de Navier-Stokes, problemas com perturbações singulares, problemas de camada limite onde aparecem as malhas fortemente alongadas, ou mesmo a paralelização de algoritmos.

Neste artigo os seguintes parâmetros são estudados: número de incógnitas, número de iterações internas (número de iterações do método numérico a fim de suavizar as componentes de erro), o número de níveis (número de malhas percorridas) e os *solvers* Gauss-Seidel (GS), MSI e ADI (Tannehill *et al.*,1997). Os objetivos são: verificar o efeito desses parâmetros sobre o tempo de CPU para o *multigrid* geométrico e realizar comparações entre o esquema de correção (CS) e o esquema de aproximação completa (FAS) com ciclo V proposto em Wesseling (1992). Os resultados são comparados com os obtidos na bibliografia. São usados operadores de restrição por injeção e prolongação por interpolação bilinear (Briggs et al., 2000). O modelo matemático considerado neste trabalho envolve um problema bidimensional linear de condução de calor, ou seja, a equação de Laplace com condições de contorno de Dirichlet.

Este artigo está organizado da seguinte forma: na seção 2 é apresentada uma visão geral do método *multigrid*. Na seção 3, é apresentado o modelo matemático e numérico. Na seção 4 são descritos os experimentos numéricos e seus resultados. E na seção 5 é apresentado a conclusão do trabalho.

2. O método *multigrid*

Para reduzir o erro de discretização, malhas muito refinadas são necessárias a fim de se resolver problemas de mecânica dos fluidos e transferência de calor. Isso gera sistemas de equações muito grandes. A resolução destes problemas através de métodos numéricos requer um custo computacional demasiadamente alto e muitas vezes inviável devido ao grande número de equações a serem resolvidas em cada passo iterativo. Uma opção para melhorar a taxa de convergência destes problemas é o método *multigrid* (Briggs *et al.*, 2000), que acelera consideravelmente a resolução dos sistemas lineares envolvidos no problema. Métodos *multigrid* são métodos iterativos para a solução de sistemas lineares, sendo, portanto, fortemente dependentes da estimativa inicial atribuída às incógnitas do problema.

Uma técnica eficiente usada para aliviar as fortes oscilações do resíduo da Eq. (3) em cada malha, definido por:

$$\vec{R} = \vec{f} - \mathbf{A}\vec{v} , \tag{4}$$

é suavizar as oscilações por um método de relaxação (método iterativo). Neste trabalho pretende-se comparar o desempenho dos métodos: GS, MSI e ADI.

As primeiras iterações deste processo, geralmente, têm rápida convergência, caracterizando a presença de modos oscilatórios de erro. Porém, após algumas iterações o processo torna-se lento, sinalizando a predominância de modos suaves (Brandt, 1977). Este é exatamente o momento onde é recomendável transferir o problema de relaxação para uma malha mais grossa, pois os modos de erros suaves na malha fina tornam-se erros oscilatórios na malha grossa (Wesseling, 1992).

Podem ser usados dois tipos de esquemas com o método *multigrid* (Briggs et al., 2000): o esquema de correção (*Correction Scheme*, CS) e o esquema de aproximação completa (Full Approximation Scheme, FAS). NO esquema CS, a Eq. (3) é resolvida apenas na malha mais fina; nas malhas mais grossas, resolve-se a equação do resíduo. Já no caso do esquema FAS, a Eq. (3) é resolvida em todas as malhas. O esquema CS é geralmente utilizado em problemas lineares e o esquema FAS em problemas não-lineares.

A taxa de convergência ideal (teórica) do *multigrid* é independente do tamanho da malha, isto é, independe do número de pontos da malha (Hirsch, 1988; Ferziger e Peric, 1999). Não é muito efetivo usar somente dois níveis de malha (Roache, 1998); para obter um bom desempenho do *multigrid*, diversos níveis de malhas devem ser usados (Tannehill *et al.*, 1997). Pinto et al. (2005a e 2005b) recomendam usar todos os níveis.

Os operadores de transferência da malha fina para a malha grossa são chamados de operadores de restrição e são denotados genericamente por $I_h^H \vec{\phi}^h = \vec{\phi}^H$. Onde $\vec{\phi}$, no caso do esquema CS, assume o resíduo $\vec{R}$ dado pela Eq. (4) e no caso do esquema FAS assume a solução aproximada do problema além do resíduo $\vec{R}$. Neste trabalho faz-se uso do operador de injeção para problemas bidimensionais que pode ser visto em Briggs *et al.* (2000).

Os operadores de transferência da malha grossa para a malha fina são chamados de operadores de prolongação, ou interpolação, e são denotados genericamente por $I_H^h \vec{\phi}^H = \vec{\phi}^h$. Onde $\vec{\phi}$, no caso do esquema CS assume a aproximação do erro na equação residual, ou seja, a correção, e no caso do esquema FAS assume a solução aproximada do problema, além da correção. Neste trabalho faz-se uso do operador de interpolação bilinear que também pode ser visto em Briggs *et al.* (2000).

3. Modelos matemático e numérico

O problema linear de condução de calor bidimensional (equação de Laplace) com condições de contorno de Dirichlet, em coordenadas cartesianas, considerado neste trabalho é (Maliska, 2004):

$$\begin{cases} \dfrac{\partial^2 T}{\partial x^2} + \dfrac{\partial^2 T}{\partial y^2} = 0, \quad 0 < x, y < 1 \\ T(x,1) = sen(\pi.x), \quad T(x,0) = T(0,y) = T(1,y) = 0 \end{cases}, \tag{5}$$

onde *T* é a incógnita e representa a temperatura. A solução analítica do problema é

$$T(x,y) = sen(\pi.x)\frac{senh(\pi.y)}{senh(\pi)}. \tag{6}$$

A discretização do domínio é feita com malhas uniformes cujos pontos são dados pela Eq. (2). Para cada um dos $(N_x - 2)$ x $(N_y - 2)$ pontos interiores da malha, a Eq. (5) é discretizada com o método de diferenças finitas com diferença central (CDS) (Tannehill *et al.*,1997), resultando em

$$\begin{cases} \dfrac{v_{i-1,j} - 2v_{i,j} + v_{i+1,j}}{h_x^2} + \dfrac{v_{i,j-1} - 2v_{i,j} + v_{i,j+1}}{h_y^2} = 0, \quad 2 \le i \le N_x - 1, \quad 2 \le j \le N_y - 1 \\ v_{i,N_y} = sen(\pi.x_{i,N_y}), \quad v_{i,1} = v_{1,j} = v_{N_x,j} = 0 \end{cases}, \tag{7}$$

onde $v_{i,j}$ é uma aproximação (solução numérica) para a solução exata $T(x_i, y_j)$.

Rearranjando os termos da Eq. (7), obtém-se

$$a_p v_P = a_n v_N + a_s v_S + a_w v_W + a_e v_E + b_p, \tag{8}$$

onde os coeficientes são dados por $a_p = 2/h_x^2 + 2/h_y^2$, $a_n = 1/h_y^2$, $a_s = 1/h_y^2$, $a_w = 1/h_x^2$, $a_e = 1/h_x^2$ e $b_p = 0$.

Se $\vec{v}$ e $\vec{f}$ são denotados por $\vec{v} = (v_1,...,v_N)^t$ e $\vec{f} = (f_1,...,f_N)^t$, respectivamente, onde $\vec{f}$ é o vetor independente formado pelos termos b_p , então o sistema da Eq. (7), pode ser representado por um sistema de equações algébricas do tipo dado pela Eq. (3), onde **A** é uma matriz pentadiagonal *N* por *N*, simétrica e definida positiva (Briggs *et al.*, 2000; Burden e Faires, 1997).

A Eq. (3) é resolvida com o método *multigrid* fazendo-se uso dos dois esquemas (CS e FAS) citados na seção anterior. Neste caso, os sistemas de equações do tipo da Eq. (3), onde $\vec{f}$ agora representa o termo fonte (resíduo ou solução aproximada) a cada nível de malha, são resolvidos com os *solvers* GS, MSI e ADI. Resolve-se também o problema, apenas na malha mais fina, com os métodos *singlegrid*: GS, MSI, ADI e Eliminação de Gauss (Elim.Gauss). O número de iterações internas (*ITI*) do método *multigrid* é o número de iterações do *solver* (método numérico) a fim de suavizar os componentes do erro.

4. Resultados

Os algoritmos foram implementados na linguagem FORTRAN 95 com o uso do aplicativo Compaq Visual Fortran 6.6 usando-se precisão dupla. As simulações foram realizadas num microcomputador com processador Intel Pentium 4 com 2.66 GHz e 1 GB de RAM.

Cerca de 400 simulações foram realizadas com as seguintes variantes: número de incógnitas (*N*), número de iterações internas (*ITI*), número de níveis de malhas (*L*) e *solvers* (GS, MSI e ADI). São apresentados neste trabalho os resultados mais representativos.

O critério de convergência para as iterações externas (*ITE*) (número de ciclos necessários para suavizar as componentes de erro) é baseado na razão entre a norma L_1 do resíduo (Ferziger e Peric, 1999) numa determinada iteração e a norma do resíduo da estimativa inicial. O resíduo de cada nó é calculado através da Eq. (4). Neste trabalho adota-se $r = 2$. Adota-se também $\varepsilon = 10^{-7}$ e $\vec{v} = (0,0,...0)$ para a tolerância sobre o critério de convergência e a estimativa inicial, respectivamente.

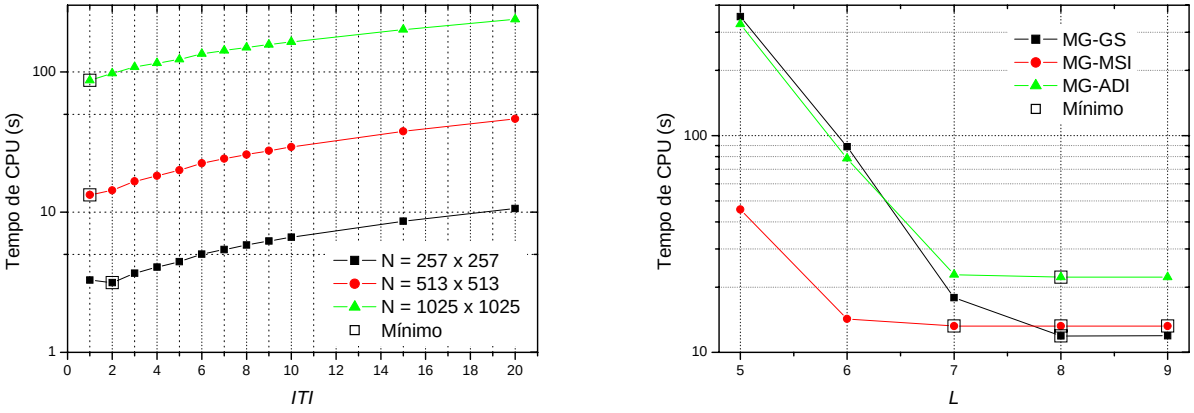
O foco deste trabalho é a minimização do tempo de CPU. Entende-se por tempo de CPU, o tempo gasto para gerar malhas, atribuir a estimativa inicial, calcular os coeficientes e resolver o sistema linear da Eq. (3). Este tempo é medido usando-se a função TIMEF da biblioteca PORTLIB do FORTRAN 95. Através de testes realizados verificou-se que a incerteza desta função é aproximadamente de ± 0.05 s.

4.1. Esquema de correção (CS)

4.1.1. Iterações internas (*ITI*)

A Fig. 1a mostra a influência do número de iterações internas (*ITI*) sobre o tempo de CPU para alguns valores de *N* e o *solver* MSI. Verificou-se que, para cada malha, em geral o menor tempo de CPU ocorre com o menor valor de *ITI*. Aumentando-se o valor de *ITI*, em geral aumenta-se significativamente o tempo de CPU. Este comportamento da Fig. 1a também ocorre com os outros *solvers* testados: GS e ADI.

Para os três *solvers* testados, a Tab. 1 mostra o número ótimo de iterações internas ($ITI_{ótimo}$), que é o *ITI* que resulta no menor tempo de CPU. Observa-se que *N* e o *solver* influenciam muito pouco o valor de $ITI_{ótimo}$. Para *N* muito grande, $ITI_{ótimo}$ = 1 com qualquer *solver*. Portanto, recomenda-se utilizar *ITI* = 1 com o esquema CS e os *solvers* GS, MSI e ADI. Este valor de $ITI_{ótimo}$ difere daquele encontrado (Pinto *et al.*, 2005) em problemas unidimensionais lineares (equações de Poisson e advecção-difusão), onde $ITI_{ótimo}$ = 3 para o *solver* GS e o mesmo método *multigrid*.



a) Número de iterações internas (*ITI*) com *solver* MSI. b) Número de níveis de malhas (*L*) para *N* = 513 x 513.

Figura 1. Método *multigrid* (MG) com esquema CS.

Tabela 1. Número ótimo de iterações internas ($ITI_{ótimo}$) para o esquema CS.

N	GS	MSI	ADI
257 x 257	2	2	1
513 x 513	1	1	1
1025 x 1025	1	1	1

4.1.2. Níveis de malhas (*L*)

A Fig. 1b mostra a influência do número de níveis de malhas (*L*) sobre o tempo de CPU, para *N* = 513 x 513 nós e os três *solvers* testados. Para esta malha, $L_{máximo}$ = 9, isto é, usando-se a razão de engrossamento *r* = 2, para resolver a malha mais fina de 513 x 513 nós, pode-se usar no máximo mais 8 malhas, que são: 257x257, 129x129, 65x65, 33x33, 17x17, 9x9, 5x5 e 3x3, que apresenta apenas um nó interno. Verificou-se que, para cada *solver*, o número ótimo de níveis de malha ($L_{ótimo}$), ocorre em geral para $L_{ótimo} = L_{máximo} - 1$, onde $L_{ótimo}$ é o *L* que resulta no menor tempo de CPU. Diminuindo-se o valor de *L*, em geral aumenta-se significativamente o tempo de CPU. Este resultado é praticamente o mesmo encontrado (Pinto *et al*., 2005) em problemas unidimensionais lineares (equações de Poisson e advecção-difusão), onde $L_{ótimo} \approx L_{máximo}$ para o *solver* GS e o mesmo método *multigrid*. Portanto, recomenda-se utilizar $L_{ótimo} = L_{máximo}$ com o esquema CS e os *solvers* GS, ADI e MSI.

4.1.3. Número de incógnitas (*N*)

A Fig. 2 mostra a influência do número de incógnitas (*N*) sobre o tempo de CPU para os três *solvers* (GS, MSI e ADI) usados com o método *multigrid* (MG). Também são mostrados resultados para os mesmos três *solvers* mas usando-se o método padrão de malha única, aqui chamado de *singlegrid* (SG). Além disso são apresentados resultados para um método direto, eliminação de Gauss (Elim.Gauss), também baseado em malha única. As malhas usadas são: 5x5, 9x9, ... até 65x65 (Elim.Gauss) ou 513x513 (SG) ou 2049x2049 (MG) nós.

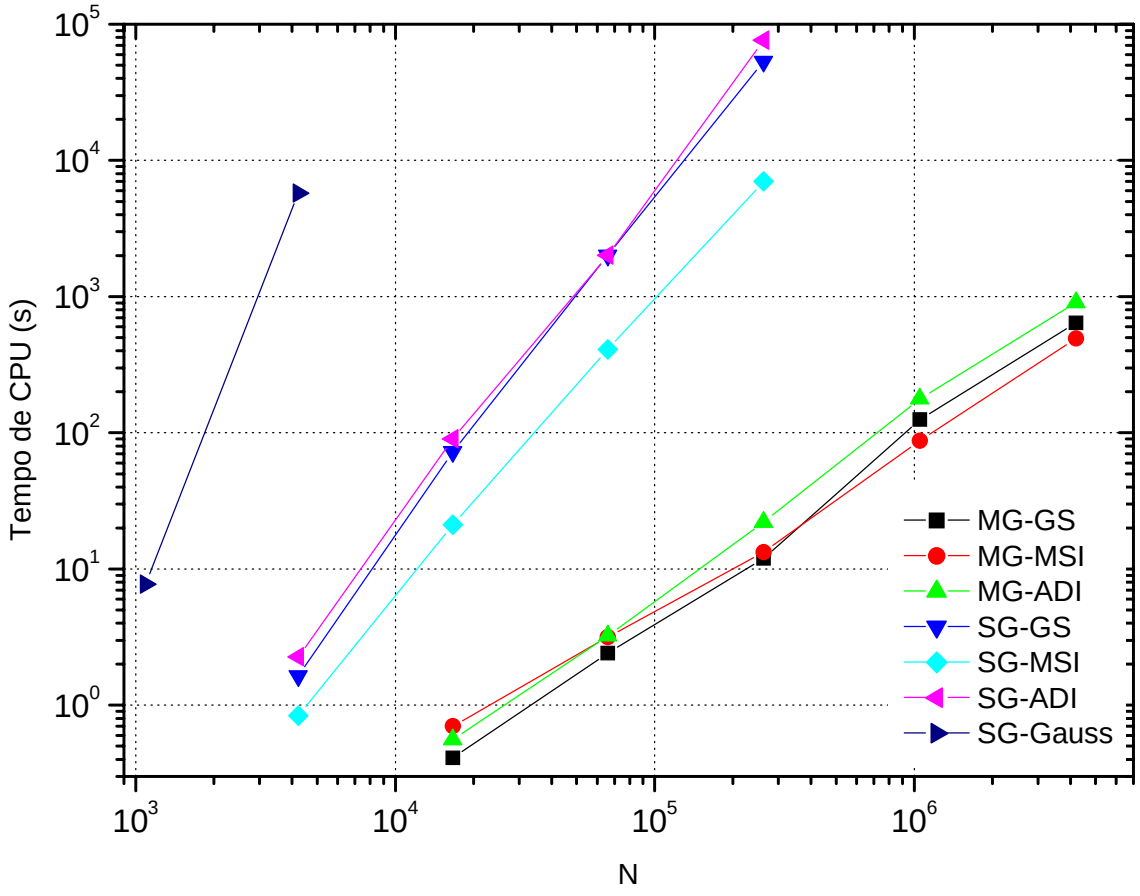


Figura 2. Tempo de CPU *versus* número de incógnitas (*N*) com esquema CS.

Na Fig. 2 são mostrados apenas os pontos cujo tempo de CPU não é muito influenciado pela incerteza de sua medição. Verificou-se que, para malhas com $N > 10^4$, o tempo de CPU do método *multigrid* com qualquer *solver* é significativamente menor do que o método *singlegrid* iterativo, que por sua vez é significativamente menor do que o método *singlegrid* direto (Elim.Gauss). Por exemplo, para a malha 65x65 nós, o tempo de CPU do SG-MSI e SG-Elim.Gauss é de 0,84 s e 5737,22 s, respectivamente, ou seja, o SG-MSI é cerca de 6830 vezes mais rápido do que o SG-Elim.Gauss. Outro exemplo, para a malha 513x513 nós, o tempo de CPU do MG-MSI e SG-MSI é de 13,3 s e 7006 s, respectivamente, ou seja, o MG-MSI é cerca de 527 vezes mais rápido do que o SG-MSI. Estas diferenças ficam cada vez maiores à medida que *N* aumenta, pois as inclinações das curvas dos *solvers* com MG são menores do que com SG. Entre os métodos iterativos, em geral, observa-se que:

$$t_{CPU}(MSI) \;<\; t_{CPU}(GS) \;<\; t_{CPU}(ADI)\,. \tag{9}$$

Para os pontos da Fig. 2, a Tab. 2 apresenta a inclinação (p) das curvas, obtida por ajuste geométrico de mínimos quadrados considerando a seguinte função:

$$t_{CPU} \;=\; c\,N^{\,p}\,, \tag{10}$$

onde p representa a ordem do *solver* associado ao método empregado, e c é um coeficiente que depende de cada método e cada *solver*. O método *multigrid* ideal é aquele cujo p = 1, isto é, aquele cujo tempo de CPU cresce linearmente com o tamanho do problema, ou com o número de incógnitas N. Portanto, para cada método e *solver*, quanto mais próximo da unidade estiver seu p, melhor é o seu desempenho. A ordem dos *solvers* com o *multigrid* é pouco afetada se o esquema adotado é o CS ou o FAS. Mas a ordem é muito afetada em função do método adotado, se direto ou iterativo, e se *singlegrid* ou *multigrid*. Entre os três *solvers* testados, o MSI é o mais eficiente com qualquer método empregado, isto é, requer o menor tempo de CPU para resolver um dado sistema com N incógnitas. E entre os métodos, o MSI é o mais eficiente com o método *multigrid* e o esquema FAS.

Pela Tab. 2, pode-se notar que, tanto para os métodos *singlegrid*, quanto para os métodos *multigrid* com o uso do esquema FAS, quanto mais fortemente implícito for o método, mais rápido ele é. Esta análise sofre uma pequena variação no caso dos métodos *multigrid* com o uso do esquema CS quando se trata dos métodos GS e ADI, mas mesmo assim, o método mais fortemente implícito (MSI) continua sendo o mais rápido.

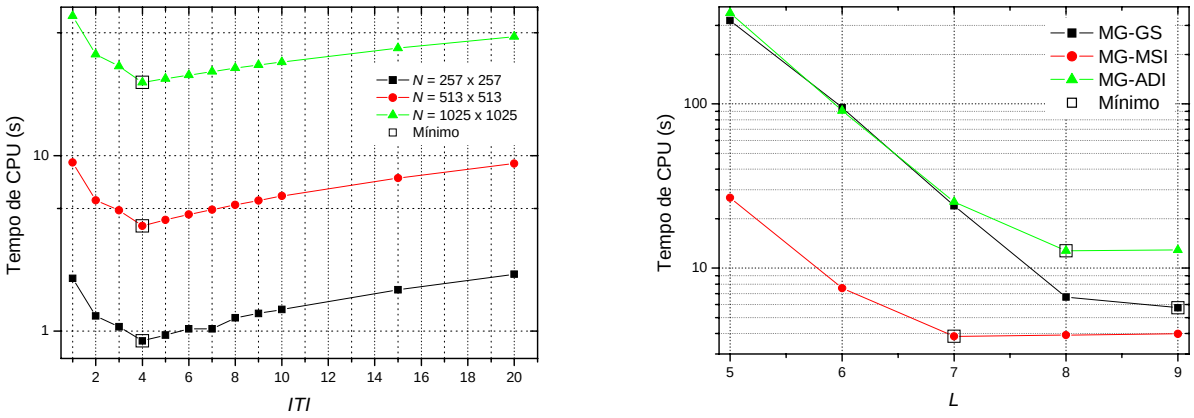
Tabela 2. Ordem (p) dos métodos e *solvers*.

Solver	Tipo de *solver*	*Singlegrid*	Esquema CS	Esquema FAS
Elim.Gauss	direto	4.88	---	---
GS	iterativo	2.50	1.35	1.40
ADI	iterativo	2.50	1.36	1.39
MSI	iterativo	2.18	1.19	1.16

4.2. Esquema de aproximação completa (FAS)

4.2.1. Iterações internas (*ITI*)

A Fig. 3a mostra a influência do número de iterações internas (*ITI*) sobre o tempo de CPU para alguns valores de N e o *solver* MSI. Verificou-se que, para cada malha, o menor tempo de CPU ocorre com $ITI_{ótimo}$ = 4. Diminuindo-se ou aumentando-se o valor de *ITI* em relação ao ótimo, pode-se aumentar significativamente o tempo de CPU. Este comportamento da Fig. 3a também ocorre com os outros *solvers* testados: GS e ADI. A Tab. 3 mostra o número ótimo de iterações internas ($ITI_{ótimo}$) para os três *solvers* testados. Observa-se que N e o *solver* influenciam muito pouco o valor de $ITI_{ótimo}$. Como na prática procura-se resolver problemas de grande porte, recomenda-se utilizar *ITI* = 3, 4 ou 5, respectivamente, para os *solvers* GS, MSI e ADI.



a) Número de iterações internas (*ITI*) com *solver* MSI.

b) Número de níveis de malhas (L) para N = 513 x 513.

Figura 3. Método *multigrid* (MG) com esquema FAS.

Tabela 3. Número ótimo de iterações internas ($ITI_{ótimo}$) para o esquema FAS.

N	GS	MSI	ADI
257 x 257	4	4	5
513 x 513	3	4	5
1025 x 1025	3	4	5

Na seção anterior, verificou-se que para o esquema CS, $ITI_{ótimo} = 1$ com qualquer *solver*. Já nesta seção, verificou-se que para o esquema FAS, $ITI_{ótimo} = 3$ a 5, dependendo do *solver*. Portanto, o esquema utilizado (CS ou FAS) influencia o número ótimo de iterações internas do método *multigrid*.

4.2.2. Níveis de malhas (*L*)

A Fig. 3b mostra a influência do número de níveis de malhas (*L*) sobre o tempo de CPU, para *N* = 513 x 513 nós e os três *solvers* testados. Verificou-se que, para cada *solver*, o número ótimo de níveis de malha ($L_{ótimo}$) ocorre na média para $L_{ótimo} = L_{máximo} - 1$. Diminuindo-se o valor de *L*, em geral aumenta-se significativamente o tempo de CPU. Portanto, recomenda-se utilizar $L_{ótimo} = L_{máximo}$ com o esquema FAS e os *solvers* GS, MSI e ADI. Na seção anterior, verificou-se que para o esquema CS, $L_{ótimo} = L_{máximo} - 1$, com qualquer *solver*. Já nesta seção, ocorre o mesmo para o esquema FAS. Portanto o esquema utilizado (CS ou FAS) não tem grande influência sobre o número ótimo de níveis para o método *multigrid*.

4.2.3. Número de incógnitas (*N*)

A Fig. 4 mostra a influência do número de incógnitas (*N)* sobre o tempo de CPU para os três *solvers* (GS, MSI e ADI) usados com o método *multigrid* (MG). Também são mostrados resultados para os mesmos três *solvers* mas usando-se o método *singlegrid* (SG) e eliminação de Gauss (Elim.Gauss) também baseado em malha única. As malhas usadas são: 5x5, 9x9, ... até 65x65 (Elim.Gauss) ou 513x513 (SG) ou 2049x2049 (MG) nós.

Na Fig. 4 são mostrados apenas os pontos cujo tempo de CPU não é muito influenciado pela incerteza de sua medição. Verificou-se que, para malhas com $N > 10^4$, o tempo de CPU do método *multigrid* com qualquer *solver* é significativamente menor do que o método *singlegrid* iterativo, que por sua vez é significativamente menor do que o método *singlegrid* direto (Elim.Gauss). Por exemplo, para a malha 513x513 nós, o tempo de CPU do MG-MSI e SG-MSI é de 3,8 s e 7006 s, respectivamente, ou seja, o MG-MSI é cerca de 1844 vezes mais rápido do que o SG-MSI. Estas diferenças ficam cada vez maiores à medida que *N* aumenta, pois as inclinações das curvas dos *solvers* com MG são menores do que com SG. Entre os métodos iterativos, em geral, para o esquema FAS também vale a Eq. (9). Para os pontos da Fig. 4, a Tab. 2 apresenta a inclinação (*p*) das curvas, obtida por ajuste de mínimos quadrados considerando a função dada na Eq. (10).

4.3. Esquema CS *versus* FAS

Na Fig. 5 se faz uma comparação entre os esquemas *multigrid* CS e FAS em função de diversos valores de *N*. Pode-se notar que o *solver* MSI que é o mais eficiente com qualquer esquema. Pode-se notar ainda que, o esquema FAS é mais rápido do que o CS em qualquer *N*, entre 3,2 e 4,0 vezes. Isso também ocorre para os *solvers* GS (entre 2,1 e 2,9 vezes) e ADI (entre 1,7 e 2,6 vezes). Por exemplo, para a malha 2049x2049 nós, o tempo de CPU do MG-MSI-FAS e MG-MSI-CS é de 124,7 s e 494,3 s, respectivamente, ou seja, o MG-MSI-FAS é cerca de 4,0 vezes mais rápido do que o MG-MSI-CS.

Portanto, recomenda-se o uso do esquema FAS com o *solver* MSI para a equação de Laplace bidimensional. Esta constatação é inédita e inesperada pois o esquema CS é indicado para resolver equações lineares, caso deste trabalho, e o esquema FAS, para equações não-lineares. Aqui vemos que o esquema CS perde a hegemonia perante o esquema FAS, mesmo que para um problema linear.

Os resultados do presente trabalho diferem daqueles comentados em Brandt (1977). Ele fez análises teóricas e experimentais (numéricas) entre as razões de engrossamento $r = 2$, 3 e 3/2 para diversos problemas, mas não se referiu à Eq. (5) exatamente. Brandt mostra preferência pelo esquema CS em relação ao esquema FAS no caso de problemas lineares. Segundo Brandt, cada ciclo iterativo do esquema FAS é mais caro computacionalmente se comparado ao esquema CS devido à transferência de informações a respeito do resíduo e da solução para as malhas mais grossas.

Seja o problema de minimização do tempo de CPU em função do uso dos diferentes esquemas CS e FAS. Para este problema, constatou-se que o esquema ótimo tem uma forte influência da qualidade das informações transferidas pela restrição e pela prolongação e, principalmente, pela transferência de informações a respeito do resíduo e da solução para as malhas mais grossas. Para o esquema FAS, por exemplo, a estimativa inicial para as malhas mais grossas é sempre a restrição da solução suavizada da malha imediatamente mais fina, enquanto que para o esquema CS, a estimativa inicial é sempre a estimativa inicial nula. Ou seja, o esquema FAS tem sempre estimativas iniciais melhores para os problemas

nas malhas grossas. Com isto, constatou-se no presente trabalho que o número de ciclos ou iterações do esquema FAS é significativamente menor do que o esquema CS. Por exemplo, para a malha 2049x2049 nós, o número de ciclos ou iterações do MG-MSI-FAS é de apenas 3 contra 22 do MG-MSI-CS.

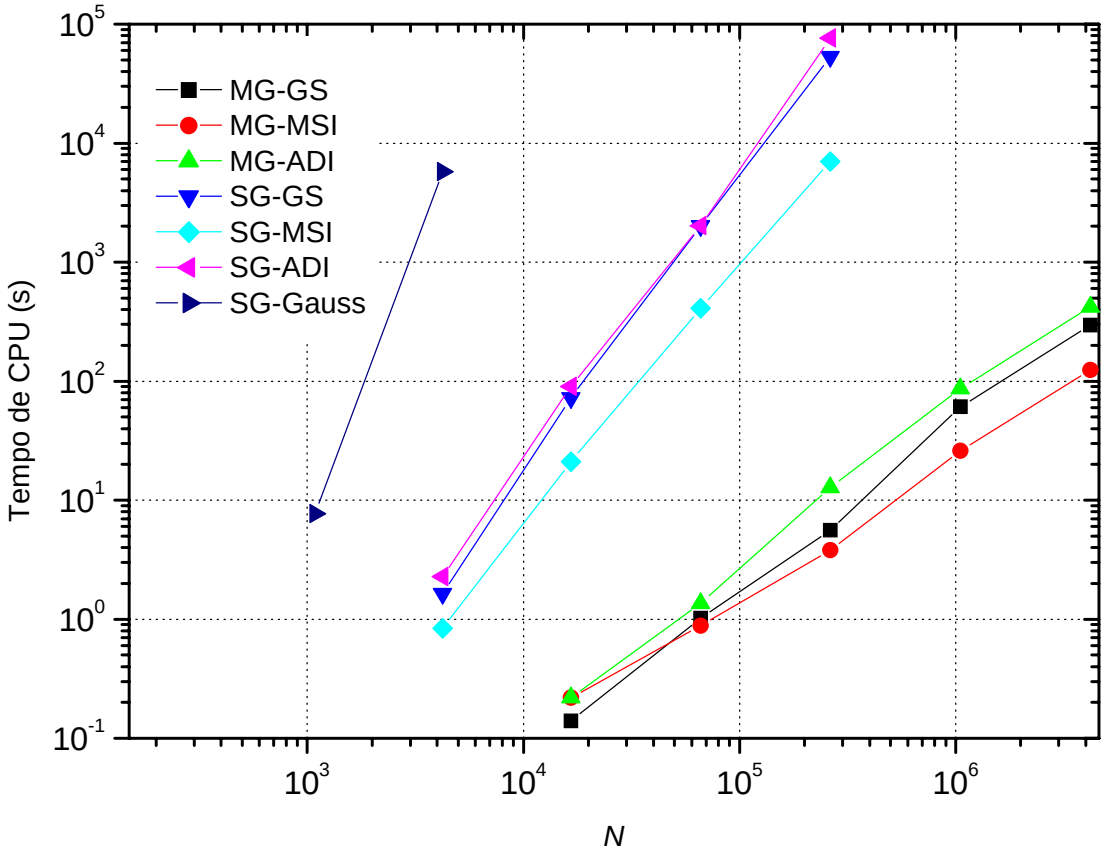


Figura 4. Tempo de CPU *versus* número de incógnitas (*N*) com esquema FAS.

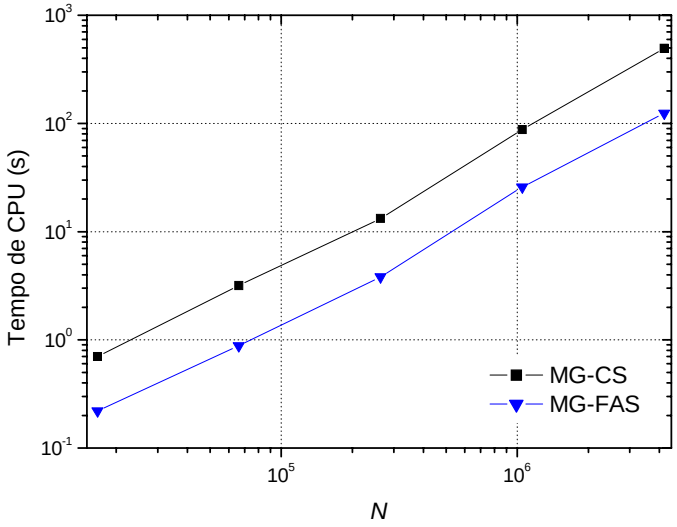


Figura 5. Tempo de CPU *versus* número de incógnitas (*N*) para o método *multigrid* (MG) com o *solver* MSI e os esquemas CS e FAS.

5. Conclusão

Neste trabalho verificou-se o efeito de diversos parâmetros sobre o tempo de CPU necessário para resolver um problema com o método *multigrid* (MG) geométrico. Os parâmetros considerados foram: número de nós (*N*), número de iterações internas (*ITI*), número de níveis de malhas (*L*), *solvers* (GS, MSI e ADI), e esquemas de correção (CS) e de aproximação completa (FAS). O modelo matemático considerado é um problema bidimensional linear, governado pela equação de Laplace com condições de contorno de Dirichlet. Esta equação foi discretizada com o método de diferenças finitas.

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

1) O esquema FAS é mais rápido do que o CS.
2) Para os esquemas CS e FAS, o *solver* MSI é mais rápido do que o GS e o ADI. Portanto, mesmo com o método *multigrid*, quanto mais fortemente implícito é o *solver*, mais rápido ele é.
3) O esquema utilizado influencia o número ótimo de iterações internas: para o esquema CS, $ITI_{ótimo} = 1$ com qualquer *solver*; e, para o esquema FAS, $ITI_{ótimo} = 3$ a 5, dependendo do *solver*.
4) O esquema utilizado não tem grande influência sobre o número ótimo de níveis de malha: $L_{ótimo} \approx L_{máximo}$ para os esquemas CS e FAS com os *solvers* GS, MSI e ADI.

Segundo o conhecimento dos autores, estas constatações são inéditas na literatura disponível.

6. Agradecimentos

O primeiro autor agradece o apoio do Laboratório de Experimentação Numérica (LENA), do Departamento de Engenharia Mecânica da UFPR, por disponibilizar sua infra-estrutura, da Universidade Estadual de Ponta Grossa e da CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pelo suporte financeiro e dos amigos do LENA. O segundo autor é bolsista do CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) .

7. Referências

Brandt, A., "Multi-Level adaptive solutions to boundary-value problems, Mathematics of Computation", vol. 31, pp. 333-390, 1977.

Briggs, W. L. and Henson, V.E., Mccormick, S.F., "A Multgrid Tutorial", 2ª ed., SIAM, 2000.

Burden, R. L. and Faires, J. D., "Numerical Analysis", 6ª ed., Brooks/Cole Publishing Company, 1997.

Dennis, J. E. and Schnabel, R., "Numerical Methods for Unconstrained Optimization and Nonlinear Equations". Pratice Hall, 1983.

Fedorenko, R. P., "On the Speed of Convergence of an Iteration Process", USSR Comput. Math. And Math. Phys., vol. 4 (3), 1964.

Ferziger, J. H. and Peric, M., "Computational Methods for Fluid Dynamicas", 2ª ed., Springer, 1999.

Fortuna, A. O., "Técnicas Computacionais para Dinâmica dos Fluidos", Edusp, 2000.

Golub, G. H. and Ortega, J. M., "Scientific Computing and Differential Equations: An Introduction to Numerical Methods", Academic Press, Inc., 1992.

Golub, G. H. and Van Loan, C., "Matrix Computations", 2ª ed., Johns Hopkins Press, Baltimore, 1989.

Hestenes, M. and Stiefeld, E., "Methods of Conjugate Gradients for Solving Linear Systems", Journal of Research of the National Bureau of Standards, vol. 49, pp. 409-436, 1952.

Hirsch, C., "Numerical Computational of Internal and External Flows", vol.1, Wiley, 1988.

Maliska, C. R., "Transferência de calor e mecânica dos fluidos computacional", LTC, 2ª ed., 2004.

Moro Filho, R. C., "Aplicação da Técnica Multigrid em Transferência de Calor Computacional", XXV Iberian Latin_American Congress on Computational Methods in Engineering, 2004.

Pinto, M. A. V.; Santiago, C. D.; Marchi, C. H., "Effect of Parameters of a Multigrid Method on the CPU Time for One-dimensional Problems", Proceedings of the International Congress of Mechanical Engineering (COBEM), Paper 0619, 2005a.

Pinto, M. A. V.; Santiago, C. D.; Marchi, C. H., "Efeito de Parâmetros do Método Multigrid sobre o Tempo de CPU para a Equação de Burgers Unidimensional", Proceedings of the Iberian Latin-American Congress on Computational Methods in Engineering (CILAMCE), Paper Cil 05-0098, 2005b.

Roache, P. J., "Fundamentals of Computational Fluid Dynamics", Hermosa Publishers, 1998.

Stüben, K., "Algebraic Multgrid (AMG): an introduction with applications", in: GMD-Report 70, November 1999.

Stüben, K., "A Review of Algebraic Multgrid", Journal of Computation and Applied Mathematics, vol. 128, pp. 281-309, 2001.

Tannehill, J. C., Anderson, D. A. and Pletcher, R. H., "Computational Fluid Mechanics and Heat Transfer", 2ª ed., Washington: Taylor & Francis, 1997.

Wesseling, P., "An Introduction to *Multgrid* Methods", John Wiley & Sons, 1992.

Wesseling, P. and Oosterlee, C. W., "Geometric Multgrid with Applications to Computational Fluid Dynamics", Journal of Computation and Applied Mathematics, vol. 128, pp. 311-334, 2001.

EFFECT OF CS AND FAS MULTIGRID PARAMETERS ON THE CPU TIME FOR TWO-DIMENSIONAL LAPLACE'S EQUATION

Márcio Augusto Villela Pinto
Department of Mathematics and Statistics, State University of Ponta Grossa, Ponta Grossa, PR, Brazil
marciovp@demec.ufpr.br

Carlos Henrique Marchi
Department of Mechanical Engineering, Federal University of Paraná, Curitiba, PR, Brazil
marchi@demec.ufpr.br

Abstract
On the necessary CPU time to solve a problem, one verifies the effect considered by: number of nodes, number of inner iterations, number of grid levels, solvers (GS, MSI and ADI) and Correction (CS) and Full Approximation Schemes (FAS). The considered problem involves a two-dimensional linear problem: Laplace´s equation with Dirichlet boundary conditions. The finite difference method is used to discretizate the differential equation. The algebraic linear systems are solved with the three solvers associated to geometric multigrid with V-cycle. Some literature results are confirmed and some new ones are presented. The main conclusions of this work are that FAS is faster than CS and MSI solver is faster than GS and ADI solvers for both CS and FAS schemes.

Keywords: solver, CFD, finite difference, heat transfer, numerical methods.