

Projeto de Ensino

MULTIGRID

Bruno Martins da Costa Fonseca

e-mail: brunomacf@gmail.com

Projeto Financiado pela PROGRAD/UFMG como Projeto de Ensino de Graduação (PEG): "O Ensino de Multigrid em Álgebra Linear Numérica".

Orientador: Rodney Josué Biezuner

Coorientador: Denise Burgarelli Duczmal

15 de Agosto de 2009

Sumário

1	O Caso Unidimensional	2
1.1	Discretização do Problema	2
1.2	Autovalores e Autofunções do Laplaciano	4
1.3	Autovalores e Autovetores da Matriz de Discretização	4
1.4	Métodos Iterativos	5
1.4.1	Método de Jacobi	5
1.4.2	Método de Gauss-Seidel	8
1.4.3	Método de Jacobi Amortecido (ω -Jac)	10
1.4.4	Método de Gauss-Seidel Sobrerelaxado (ω -GS ou SOR)	11
1.5	Bigrid	13
1.6	Multigrid	17
1.6.1	Multigrid Completo (Full Multigrid - FMG)	18
2	O Caso Bidimensional	20
2.1	Discretização do Problema	20
2.2	Autovalores e Autofunções do Laplaciano	23
2.3	Autovetores e Autovalores da Matriz de Discretização	24
2.4	Métodos Iterativos	27
2.4.1	Método de Jacobi	27
2.4.2	Método de Gauss-Seidel	29
2.4.3	Método de Jacobi Amortecido (ω -Jac)	30
2.4.4	Método de Gauss-Seidel Amortecido (ω -GS ou SOR)	32
2.5	Bigrid	33
2.6	Multigrid	36
2.6.1	Multigrid Completo (Full Multigrid - FMG)	37

Capítulo 1

O Caso Unidimensional

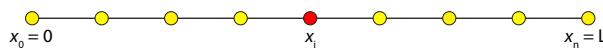
"Os métodos Multigrid são geralmente considerados os métodos numéricos mais rápidos para resolução de equações diferenciais parciais elípticas. Além disso, situam-se dentre os métodos mais rápidos para resolução de muitos outros problemas, como outros tipos de equações diferenciais parciais, equações integrais, etc." [TOS01] Trata-se de uma família de métodos destinados a resolver os sistemas lineares $A\mathbf{x} = \mathbf{f}$ que surgem no processo de resolver computacionalmente os problemas em questão.

Estamos interessados em resolver o problema de Dirichlet para a equação de Poisson. Começando pelo caso unidimensional mais simples, isso significa que buscamos uma função $u(x)$ que satisfaça o seguinte sistema:

$$\begin{cases} -u_{xx} = f & x \in (0, L) \\ u(0) = u(L) = 0 \end{cases} \quad (1.1)$$

1.1 Discretização do Problema

Para resolvermos o problema anterior computacionalmente, transformaremos o sistema (1.1) num sistema



de equações lineares cuja solução será uma aproximação da solução $u(x)$ e que poderemos obter via métodos numéricos conhecidos (p. ex., métodos iterativos como os de Jacobi ou Gauss-Seidel). Para tanto, ao invés de trabalhar com o domínio contínuo $[0, L]$, tome apenas um conjunto finito $\{x_0 = 0 < x_1 < \dots < x_{i-1} < x_i < x_{i+1} < \dots < x_{n-1} < x_n = L\}$ com $n+1$ pontos igualmente espaçados (chame de h a distância entre dois pontos x_i 's consecutivos e observe que $x_i = x_{i-1} + h$ para todo i de 1 até n) no domínio $[0, L]$. De posse desse conjunto, o qual é denominado um **grid** (ou **malha**) **uniforme** de espaçamento h em $[0, L]$ ou simplesmente de um grid h , utilizamos a série de Taylor para obter uma aproximação do valor que u_{xx} assume num ponto x_i desse grid em termos dos valores que u assume nos pontos consecutivos x_{i-1} , x_i e x_{i+1} . Isso porque pela série de Taylor sabemos que

$$u(x_i + h) = u(x_i) + u_x(x_i)h + \frac{1}{2}u_{xx}(x_i)h^2 + O(h^3) \quad (1.2)$$

e

$$u(x_i - h) = u(x_i) - u_x(x_i)h + \frac{1}{2}u_{xx}(x_i)h^2 + O(h^3) \quad (1.3)$$

de modo que, somando membro a membro essas duas desigualdades e colocando $u_{xx}(x_i)$ em evidência, obtemos

$$u_{xx}(x_i) = \frac{u(x_i - h) - 2u(x_i) + u(x_i + h)}{h^2} + O(h^3). \quad (1.4)$$

Agora, do nosso problema original (1.1) queremos que $-u_{xx}(x_i) = f(x_i)$ para todo x_i de x_1 até x_{n-1} , de modo que, denotando $u(x_i) = u_i$ e $f(x_i) = f_i$ para condensar a notação, então obtemos o seguinte sistema de equações lineares:

$$\begin{cases} \frac{1}{h^2}(-u_0 + 2u_1 - u_2) = f_1 \\ \frac{1}{h^2}(-u_1 + 2u_2 - u_3) = f_2 \\ \vdots \\ \frac{1}{h^2}(-u_{n-2} + 2u_{n-1} - u_n) = f_{n-1} \end{cases} \quad (1.5)$$

Se observarmos que as condições de fronteira de (1.1) implicam $u_0 = u_n = 0$, então o sistema anterior em forma matricial se torna (verifique!!):

$$\frac{1}{h^2} \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-2} \\ u_{n-1} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n-2} \\ f_{n-1} \end{bmatrix} \quad (1.6)$$

Dessa forma, o problema de encontrar uma função $u(x)$ que satisfaz (1.1) é reduzido a encontrar um vetor $\mathbf{u}_h = (u_1, \dots, u_{n-1})$ que satisfaz a equação matricial (1.6) para o grid h . Observe que esse vetor $\mathbf{u}_h$ aproxima os valores que a função $u(x)$ assume nos pontos x_i do grid h , de modo que podemos dizer que $\mathbf{u}_h$ é uma aproximação da solução contínua $\mathbf{u} = (u(x_1), \dots, u(x_{n-1}))$ e que essa aproximação é tanto melhor quanto maior é o número de pontos que tomamos na construção do grid (i.e., menor é a distância h entre pontos consecutivos do grid), já que a aproximação de Taylor é melhorada. Por fim, em (1.6) a matriz $(n-1) \times (n-1)$

$$A_h = \frac{1}{h^2} \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \quad (1.7)$$

é denominada a **matriz de discretização** do problema (1.1).

Dessa forma, se garantirmos que a equação matricial (1.6) de fato possui solução única, então poderemos aplicar métodos numéricos tais como os métodos iterativos de Jacobi ou Gauss-Seidel para obtermos nossa solução aproximada de (1.1). Contudo, esses métodos se mostram muito lentos quando a matriz de discretização é grande, isto é, quando tomamos muitos pontos em $[0, L]$ para compor um grid mais fino (= com muitos pontos) e, portanto, obter uma solução discreta $\mathbf{u}_h$ mais próxima da solução contínua $u(x)$! Nosso objetivo é utilizar as propriedades das EDPs para desenvolver um método numérico que seja mais eficiente para esses problemas específicos. Para tanto, a primeira coisa que faremos será investigar os autovalores e autofunções do operador $D_x^2(u) = u_{xx}$ e como eles se relacionam com os autovalores e autovetores da matriz de discretização A_h .

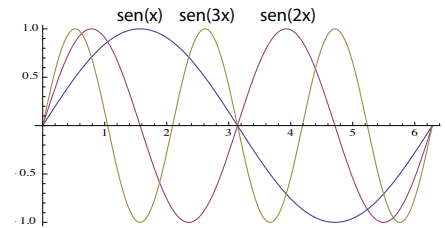
1.2 Autovalores e Autofunções do Laplaciano

Procuramos as funções $u(x)$ que satisfazem o sistema:

$$\begin{cases} -u_{xx} = \lambda u & x \in (0, L) \\ u(0) = u(L) = 0 \end{cases} \quad (1.8)$$

Esse é o problema de Dirichlet para a equação de Helmholtz unidimensional cujas soluções são as autofunções do operador $D_x^2(u) = u_{xx}$ (i.e. do operador derivada segunda com relação a x) com condição de Dirichlet homogênea. Não é muito difícil ver que a função $u(x) = \sin\sqrt{\lambda}x$ satisfaz a primeira equação de (1.8), uma vez que $u_x = \sqrt{\lambda}\cos\sqrt{\lambda}x$ e, portanto, $u_{xx} = -\lambda\sin\sqrt{\lambda}x = -\lambda u$. Observe que qualquer múltiplo de $\sin\sqrt{\lambda}x$ continua satisfazendo a primeira equação de (1.8), de modo que mais geralmente tenhamos $u(x) = c \cdot \sin\sqrt{\lambda}x$. Por fim, das condições de fronteira em (1.8), concluímos que $c \cdot \sin\sqrt{\lambda}L = 0$ e, portanto, como não queremos $c = 0$, que $\sqrt{\lambda}L = i\pi$ com $i \in \mathbb{N}$. Dessa forma, temos que $\lambda_i = i^2\pi^2/L^2$ são os autovalores e $v_i(x) = \sin(i\pi/L)$ são as autofunções de D_x^2 .

Observe que as autofunções v_i são tais que sua frequência aumenta à medida que i aumenta (veja a figura ao lado)!! Isso será importante quando viermos a discutir mais adiante as propriedades suavizadoras (i.e., propriedades de remover as autofunções de frequência elevada) de alguns métodos iterativos, o que será a idéia central do método Multigrid.



1.3 Autovalores e Autovetores da Matriz de Discretização

Podemos agora nos perguntar se existiria alguma relação entre os autovalores e autovetores da matriz de discretização A_h e os autovalores e autofunções do operador $D_x^2(u) = u_{xx}$. A Proposição abaixo revela que os autovetores de A_h são justamente as autofunções de D_x^2 discretizadas, ou seja, que as coordenadas do i -ésimo autovetor de A_h são justamente os valores que a i -ésima autofunção de D_x^2 assume nos pontos do grid!

Proposition 1.1. *A matriz de discretização A_h do Problema de Dirichlet para a Equação de Poisson Unidimensional (i.e., o problema (1.1)) possui autovalores $\lambda_i = \frac{2}{h^2}(1 - \cos(\frac{i\pi h}{L}))$ e autovetores $\mathbf{v}_i = (\sin(\frac{i\pi x_1}{L}), \dots, \sin(\frac{i\pi x_{n-1}}{L}))$ correspondentes com $i = 1, 2, \dots, n - 1$.*

Demonstração. Basta verificar que os vetores $\mathbf{v}_i = (\sin(\frac{i\pi x_1}{L}), \dots, \sin(\frac{i\pi x_{n-1}}{L}))$ de fato cumprem o requisito para serem autovetores de A_h , ou seja, verificar que $A_h \mathbf{v}_i = \lambda_i \mathbf{v}_i$. Faça as contas utilizando relações trigonométricas conhecidas e depois verifique em [Bie07] ou [Bie09]. $\square$

Remark 1.2. Poderíamos denotar $\mathbf{v}_i = (\sin(\frac{i\pi x}{L}))_h = (v_i)_h$ para enfatizar o fato de que as coordenadas do i -ésimo autovetor de A_h são exatamente os valores que a i -ésima autofunção v_i do laplaciano assume nos pontos do grid h .

Agora, observe que por Taylor temos $\sin(x) \approx 1 - \frac{1}{2}x^2$ e, portanto, os autovalores de A_h são aproximadamente dados por $\lambda_i = \frac{2}{h^2}(1 - \sin(\frac{i\pi h}{L})) \approx \frac{2}{h^2}[1 - (1 - \frac{1}{2}\frac{i^2\pi^2 h^2}{L^2})] = \frac{i^2\pi^2}{L^2}$. Como a aproximação de Taylor é tanto melhor quanto menor é o argumento x de $\sin(x)$, segue-se que os autovalores de A_h com menor índice são uma boa aproximação dos autovalores correspondentes do operador D_x^2 . Dessa forma, podemos pensar na matriz de discretização A_h como sendo uma discretização (ou aproximação) do operador derivada segunda D_x^2 .

1.4 Métodos Iterativos

A idéia básica dos métodos iterativos consiste em construir uma sequência $(\mathbf{u}^{(k)})$ de vetores a partir de um chute inicial $\mathbf{u}^{(0)}$ que converge para a solução exata $\mathbf{u}$ de uma equação matricial $A\mathbf{u} = \mathbf{f}$. Agora, para que possamos falar nessa sequência convergente devemos garantir que a equação matricial possui solução única, ou seja, que a matriz A é invertível. Felizmente, como o Teorema 1 garante que nossa matriz de discretização A_h $(n - 1) \times (n - 1)$ possui $n - 1$ autovalores não-nulos e, portanto, é diagonalizável, então A_h pode ser expressa como um produto PDP^{-1} de matrizes invertíveis, de modo que A_h seja invertível. Agora, claro que não é sempre tão fácil determinar os autovalores de matrizes de discretização quaisquer, o que nos leva ao problema de investigar as propriedades que a matriz A deve satisfazer para ser invertível. Consulte [Bie09, Cap. 2] para maiores informações.

1.4.1 Método de Jacobi

1.4.1.1 Matriz de Iteração

No método de Jacobi, resolvemos a j -ésima equação do sistema (1.5) para u_j , obtendo

$$u_j = \frac{u_{j-1} + u_{j+1}}{2} + \frac{h^2}{2} f_j. \quad (1.9)$$

Observe que, no geral, essa igualdade só é verdadeira para as componentes $j - 1, j$ e $j + 1$ da solução exata $\mathbf{u}$. Por exemplo, se utilizarmos as componentes do chute inicial $\mathbf{u}_h^{(0)}$, então não teremos, em geral, que sua j -ésima componente será aquilo expresso no lado direito de (1.9). Sendo assim, podemos definir a j -ésima componente

de um vetor $\mathbf{u}_h^{(1)}$ para ser justamente a j -ésima componente que $\mathbf{u}_h^{(0)}$ deveria ter para que a igualdade em (1.9) seja verdadeira, ou seja, podemos definir:

$$[\mathbf{u}_h^{(1)}]_j = \frac{[\mathbf{u}_h^{(0)}]_{j-1} + [\mathbf{u}_h^{(0)}]_{j+1}}{2} + \frac{h_2}{2} [\mathbf{f}_h]_j. \quad (1.10)$$

onde, em geral, teremos $[\mathbf{u}_h^{(1)}]_j \neq [\mathbf{u}_h^{(0)}]_j$ e, portanto, $[\mathbf{u}_h^{(1)}]_j \neq u_j$ (já que, em geral, o chute inicial certamente não será a solução exata!!!). Observe que a equação (1.10) é justamente a equação (1.9) corrigida para o caso de $\mathbf{u}_h^{(0)}$, no sentido de que ela troca a relação (falsa) de igualdade entre as componentes $j, j-1$ e $j+1$ de $\mathbf{u}_h^{(0)}$ pela relação (verdadeira) de igualdade entre as componentes $j-1, j+1$ de $\mathbf{u}_h^{(0)}$ e a componente j de um vetor $\mathbf{u}_h^{(1)}$. Da mesma forma, poderíamos corrigir (1.9) para $\mathbf{u}_j^{(1)}$ e obter um vetor $\mathbf{u}_j^{(2)}$, onde ainda teríamos $[\mathbf{u}_h^{(2)}]_j \neq u_j$ [pois $[\mathbf{u}_h^{(2)}]_j \neq [\mathbf{u}_h^{(1)}]_j$ para alguns j 's e, portanto, $[\mathbf{u}_h^{(2)}]_j \neq u_j$], embora esperamos que essa diferença seja menor que no caso de $\mathbf{u}_h^{(1)}$. Em geral, o método de Jacobi corrige (1.9) para $\mathbf{u}^{(k-1)}$ e obtém um vetor $\mathbf{u}^{(k)}$ tal que

$$[\mathbf{u}_h^{(k)}]_j = \frac{[\mathbf{u}_h^{(k-1)}]_{j-1} + [\mathbf{u}_h^{(k-1)}]_{j+1}}{2} + \frac{h_2}{2} [\mathbf{f}_h]_j. \quad (1.11)$$

Em forma matricial, o anterior pode ser expresso por

$$\mathbf{u}_h^{(k)} = R_J \mathbf{u}_h^{(k-1)} + \frac{h^2}{2} \mathbf{f}_h \quad (1.12)$$

onde $R_J = I - \frac{h^2}{2} A_h$ (verifique!).

1.4.1.2 Autovalores e Autovetores da Matriz de Iteração

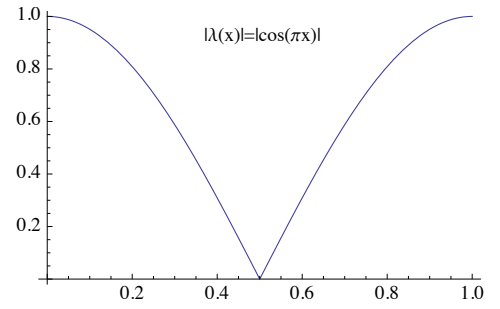
O teorema a seguir revela que os autovetores da matriz de iteração R_J do método de Jacobi para nosso problema (1.6) são exatamente os autovetores da matriz de discretização A_h .

Proposition 1.3. *A matriz de iteração R_J do método de Jacobi para o problema (1.6) possui autovalores $\lambda_i = \cos(\frac{i\pi h}{L})$ e autovetores correspondentes $\mathbf{v}_i = (\sin(\frac{i\pi x_1}{L}), \dots, \sin(\frac{i\pi x_{n-1}}{L}))$ com $i = 1, 2, \dots, n-1$, ou seja, os mesmo autovetores da matriz de discretização A_h .*

Demonstração. Os autovetores de R_J são aqueles vetores $\mathbf{v}$ tais que $R_J \mathbf{v} = \lambda \mathbf{v}$ para algum $\lambda \neq 0$. Agora, como $R_J = I - \frac{h^2}{2} A_h$, então $R_J \mathbf{v} = \lambda \mathbf{v} \iff A_h \mathbf{v} = \frac{2}{h^2} (1 - \lambda) \mathbf{v}$ e, portanto, que os autovetores de R_J são os mesmos de A_h . Além disso, a expressão $\frac{2}{h^2} (1 - \lambda)$ fornece os autovalores de A_h , de modo que $\frac{2}{h^2} (1 - \lambda) = \frac{2}{h^2} (1 - \cos(\frac{i\pi h}{L}))$ e, portanto, que $\lambda_i = \cos(\frac{i\pi h}{L})$. $\square$

Dessa forma, como um método iterativo converge se, e somente se, $\rho(R) < 1$, onde $\rho(R)$ é o raio espectral da matriz R , i.e., o maior autovalor em módulo de R (consulte [Bie09, Cap. 3] para maiores informações), então o teorema anterior garante que a sequência de Jacobi de fato converge para a solução exata de $A_h \mathbf{u}_h = \mathbf{f}_h$, uma vez que $|\lambda_i| < 1$ para todo i (verifique que o argumento da função cosseno em λ_i está estritamente entre 0 e π).

Agora, se escrevermos os autovalores de R_J como $\lambda_i = \lambda(x_i) = \cos(\pi x_i)$ onde $x_i = \frac{ih}{L} = \frac{i}{n}$ e $i = 1, 2, \dots, n-1$ e observarmos que $0 < x_i < 1$, então o gráfico de $|\lambda(x)|$ ao lado nos revela que os menores autovalores em módulo de R_J são exatamente aqueles que possuem índice i intermediário, i.e., próximo de $\frac{n}{2}$ (observe que isso acontece, porque os menores autovalores em módulo ocorrem nos x_i próximos de 0,5; mas como $x_i = \frac{i}{n}$, o anterior significa que i está próximo de $\frac{n}{2}$).



Mas como a matriz R_J é diagonalizável por se tratar de uma matriz $(n-1) \times (n-1)$ com $n-1$ autovalores não-nulos distintos e também constitui a matriz de propagação do erro para o método de Jacobi (ou seja, $\mathbf{e}^{(k)} = R_J \mathbf{e}^{(0)}$), então escrevendo o erro inicial em termos da base $\{\mathbf{v}_i\}$ de autovetores de R_J como $\mathbf{e}^{(0)} = \sum_{i=1}^{n-1} c_i \mathbf{v}_i$ segue-se que $\mathbf{e}^{(k)} = R_J^k \mathbf{e}^{(0)} = R_J^k \sum c_i \mathbf{v}_i = \sum c_i (R_J^k \mathbf{v}_i) = \sum c_i (\lambda_i^k \mathbf{v}_i)$. Assim, como os menores λ_i são aqueles com índice i intermediários, então dizemos que o método de Jacobi elimina mais eficientemente as componentes intermediárias do erro à medida que as iterações avançam (i.e., k aumenta). Agora, como os autovetores $\mathbf{v}_i$ de A_h são exatamente as autofunções $v_i(x) = \sin(\frac{i\pi x}{L})$ discretizadas (lembre-se da seção 1.2) e como à medida em que i aumenta as frequências dessas autofunções também aumentam, então costuma-se dizer que o método de Jacobi elimina mais rapidamente as componentes de frequência intermediária do erro.

Mais adiante veremos que um dos pilares centrais do método multigrid é utilizar métodos iterativos suavizadores de erro que eliminam rapidamente componentes de alta frequência do erro, ou seja, cujos menores autovalores são aqueles com índice elevado (considerando-se elevado $i > \frac{n}{2}$). Dessa forma, vemos que o método de Jacobi não é um método muito bom para suavizar o erro, já que as componentes de alta frequência não são eliminadas com eficiência. Antes de prosseguirmos, contudo, observe que, se denominarmos a região $0,5 < x < 1$ de *região de alta frequência*, uma vez que $\lambda(x_i)$ está associada a uma autofunção de frequência elevada quando x_i está nessa região, então o melhor dos casos seria garantir que o maior dos autovalores nessa região de alta frequência seja o menor possível, pois nesse caso asseguramos que todas as componentes de alta frequência são eliminadas o mais rápido possível. No caso do método de Jacobi, vemos que o maior dos autovalores na região de alta frequência encontra-se próximo de 1 (para qualquer grid temos $x_i < 1$ (veja o início do parágrafo anterior) e, portanto, $\lambda_i < 1$, já que, na região de alta frequência, $\lambda(x) = 1$ se, e somente se, $x = 1$) e que, quanto mais pontos tomamos no grid, mais próximo esse valor se torna de 1 (já que o último ponto do grid $x_n = \frac{n-1}{n}$ torna-se cada vez mais próximo de 1), de modo que, no limite, seja igual a 1. Esse limite é denominado *fator de suavização* do método iterativo e constitui uma boa medida do poder de suavização do método iterativo, no sentido que quanto menor o fator de suavização, maior é o poder de suavização do método iterativo. Observe que se minimizarmos o fator de suavização, então garantimos que todas as componentes de alta frequência são eliminadas rapidamente qualquer que seja o grid que estamos utilizando (dizemos que o fator de suavização é independente do grid). Veremos mais adiante que no método de Jacobi Amortecido somos capazes de aumentar o poder de suavização do método de Jacobi, i.e., minimizar o fator de suavização do método de Jacobi, às custas da velocidade de convergência.

1.4.2 Método de Gauss-Seidel

1.4.2.1 Matriz de Iteração

Na definição (1.11) de uma iterada de Jacobi utilizamos a componente $j - 1$ do vetor $\mathbf{u}_h^{(k-1)}$ na definição da componente j do vetor $\mathbf{u}_h^{(k)}$. Contudo, como definimos essas componentes de $\mathbf{u}_h^{(k)}$ uma a uma de $j = 1$ a $j = n - 1$, então na j -ésima componente de $\mathbf{u}_h^{(k)}$ já teremos definido sua $(j - 1)$ -ésima componente, de forma que talvez seja melhor utilizar essa componente na definição (1.11) ao invés da $(j - 1)$ -ésima componente de $\mathbf{u}_h^{(k-1)}$, pois sabemos que as componentes de $\mathbf{u}_h^{(k)}$ estão mais próximas das componentes do vetor solução $\mathbf{u}_h$ do que as componentes de $\mathbf{u}_h^{(k-1)}$. No método de Gauss-Seidel fazemos justamente isso, definindo:

$$[\mathbf{u}_h^{(k)}]_j = \frac{[\mathbf{u}_h^{(k)}]_{j-1} + [\mathbf{u}_h^{(k-1)}]_{j+1}}{2} + \frac{h_2}{2} [\mathbf{f}_h]_j. \quad (1.13)$$

Para encontrarmos a forma matricial do método de Gauss-Seidel, escrevemos o anterior como:

$$[\mathbf{u}_h^{(k)}]_j - \frac{1}{2} [\mathbf{u}_h^{(k)}]_{j-1} = \frac{1}{2} [\mathbf{u}_h^{(k-1)}]_{j+1} + \frac{h_2}{2} [\mathbf{f}_h]_j \quad (1.14)$$

e observamos que isso fornece:

$$\begin{bmatrix} 1 & & \cdots \\ -\frac{1}{2} & 1 & \cdots \\ & -\frac{1}{2} & 1 & \cdots \\ \vdots & \vdots & -\frac{1}{2} & \ddots & \vdots \\ & & \cdots & 1 \end{bmatrix} \begin{bmatrix} [\mathbf{u}_h^{(k)}]_1 \\ [\mathbf{u}_h^{(k)}]_2 \\ [\mathbf{u}_h^{(k)}]_3 \\ \vdots \\ [\mathbf{u}_h^{(k)}]_{n-1} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{2} & \cdots \\ & 0 & \frac{1}{2} & \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ & & \cdots & \frac{1}{2} \\ & & \cdots & 0 \end{bmatrix} \begin{bmatrix} [\mathbf{u}_h^{(k-1)}]_1 \\ [\mathbf{u}_h^{(k-1)}]_2 \\ [\mathbf{u}_h^{(k-1)}]_3 \\ \vdots \\ [\mathbf{u}_h^{(k-1)}]_{n-1} \end{bmatrix} + \frac{h_2}{2} \mathbf{f}_h$$

$$(I - L) \quad \mathbf{u}_h^{(k)} \quad = \quad U \quad \mathbf{u}_h^{(k-1)} \quad (1.15)$$

onde I é a identidade, L é uma matriz triangular inferior e U é uma matriz triangular superior. Como a matriz $(I - L)$ é uma matriz triangular inferior cujo determinante [que neste caso é o produto dos elementos na diagonal principal] é diferente de 0, então $(I - L)$ é invertível e, portanto, chegamos em:

$$\mathbf{u}_h^{(k)} = R_{GS} \mathbf{u}_h^{(k-1)} + \frac{h_2}{2} (I - L)^{-1} \mathbf{f}_h \quad (1.16)$$

onde $R_{GS} = (I - L)^{-1} U$.

Remark 1.4. Antes de prosseguirmos, utilizando a forma das matrizes L, U e a equação (1.13), verifique que podemos escrever:

$$\mathbf{u}_h^{(k)} = L \mathbf{u}_h^{(k)} + U \mathbf{u}_h^{(k-1)} + \frac{h_2}{2} \mathbf{f}_h \quad (1.17)$$

Essa expressão será útil no método SOR que veremos mais adiante.

1.4.2.2 Autovalores e Autovetores da Matriz de Iteração

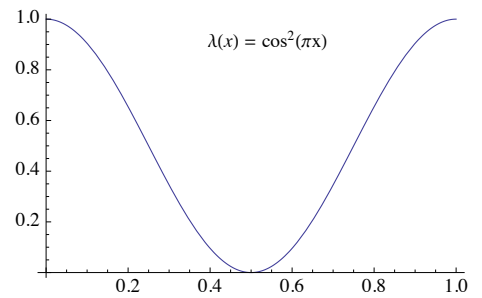
Assim como no caso de Jacobi, o teorema a seguir revela que os autovetores da matriz de iteração R_{GS} do método de Gauss-Seidel são exatamente os autovetores da matriz de discretização A_h .

Proposition 1.5. *A matriz de iteração R_{GS} do método de Gauss-Seidel para o problema (1.6) possui autovalores $\lambda_i = \cos^2(\frac{i\pi h}{L})$ e autovetores correspondentes $\mathbf{v}_i = (\cos(\frac{i\pi h}{L}) \cdot \sin(\frac{i\pi x_1}{L}), \dots, [\cos(\frac{i\pi h}{L})]^j \cdot \sin(\frac{i\pi x_j}{L}), \dots, [\cos(\frac{i\pi h}{L})]^{n-1} \cdot \sin(\frac{i\pi x_{n-1}}{L}))$ com $i = 1, 2, \dots, n-1$, ou seja, NÃO coincidindo com os autovetores da matriz de discretização A_h .*

Demonstração. Os autovetores de R_{GS} são aqueles vetores $\mathbf{v}$ tais que $R_{GS}\mathbf{v} = \lambda\mathbf{v}$ para algum $\lambda \neq 0$. Agora, como $R_{GS} = (I - L)^{-1}U$, então $R_{GS}\mathbf{v} = \lambda\mathbf{v} \implies (I - L)^{-1}U\mathbf{v} = \lambda\mathbf{v}$ e, portanto, que $U\mathbf{v} = \lambda(I - L)\mathbf{v}$. Agora, se observarmos a forma das matrizes L e U em (1.15), então veremos que o anterior significa $\frac{1}{2}[\mathbf{v}]_{j+1} = \lambda([\mathbf{v}]_j - \frac{1}{2}[\mathbf{v}]_{j-1})$ e, portanto, que $\lambda\frac{1}{2}[\mathbf{v}]_{j-1} + \frac{1}{2}[\mathbf{v}]_{j+1} = \lambda[\mathbf{v}]_j$. Das expressões (1.11) e (1.12) vemos que o lado esquerdo da igualdade anterior é muito parecido com o resultado de aplicarmos R_J [a matriz de Jacobi] ao vetor $\mathbf{v}$. De fato, se fizermos a substituição $[\mathbf{v}]_j = \lambda^{\frac{j}{2}}[\mathbf{w}]_j$, então obteremos $\frac{1}{2}\lambda^{\frac{j+1}{2}}[\mathbf{w}]_{j-1} + \frac{1}{2}\lambda^{\frac{j+1}{2}}[\mathbf{w}]_{j+1} = \lambda^{\frac{j+2}{2}}[\mathbf{w}]_j$, de modo que, dividindo tudo por $\lambda^{\frac{j+1}{2}}$, tenhamos $\frac{1}{2}[\mathbf{w}]_{j-1} + \frac{1}{2}[\mathbf{w}]_{j+1} = \lambda^{1/2}[\mathbf{w}]_j$ e, portanto, que $R_J\mathbf{w} = \lambda^{1/2}\mathbf{w}$. Assim, concluímos que $\mathbf{w}$ é um autovetor de R_J [e, portanto, de A_h] e que $\lambda^{1/2}$ é o autovalor de R_J correspondente. Disso segue-se que $\lambda_i = \cos^2(\frac{i\pi h}{L})$ e que as componentes do autovetor $\mathbf{v}_i$ de R_{GS} satisfazem $[\mathbf{v}_i]_j = \cos(\frac{i\pi h}{L})^j \cdot \sin(\frac{i\pi x_j}{L})$ [já que fizemos a substituição $[\mathbf{v}]_j = \lambda^{\frac{j}{2}}[\mathbf{w}]_j$ e $\mathbf{w}$ é autovetor de A_h]. $\square$

Observe que a Proposição anterior garante que, para o nosso problema (1.6), o método de Gauss-Seidel é convergente, uma vez que os autovalores λ_i de R_{GS} estão todos no intervalo aberto $(-1, 1)$ (basta observar que o argumento da função cosseno está sempre entre 0 e π e que nunca assume esses valores; para tanto, escreva $\frac{i\pi h}{L}$ como $\frac{i\pi}{n}$ onde $i = 1, 2, \dots, n-1$) e, portanto, o maior autovalor em módulo de R_{GS} é estritamente menor do que 1.

Agora, se escrevermos os autovalores de R_{GS} como $\lambda(x_i) = \cos^2(\pi x_i)$ onde $x_i = \frac{ih}{L} = \frac{i}{n}$ e $i = 1, 2, \dots, n-1$, então o gráfico da função $\lambda(x)$ ao lado mostra que os menores autovalores em módulo de R_{GS} ocorrem nos valores intermediário de i assim como aconteceu com o método de Jacobi. Isso significa que para nosso problema particular (1.6) o método de Gauss-Seidel também não é um método muito bom para suavizar o erro.



Por fim, observe que o método de Gauss-Seidel será mais rápido do que o método de Jacobi, uma vez que seus autovalores são todos menores que os autovalores de Jacobi e, portanto, eliminará mais rápido todas as componentes de um mesmo erro inicial $\mathbf{e}^{(0)}$ [lembre-se que podemos escrever $\mathbf{e}^{(k)} = R\mathbf{e}^{(0)} = \sum c_i \lambda_i^k \mathbf{v}_i$, onde λ_i é autovalor de R e $\mathbf{v}_i$ é seu autovetor associado; para maiores informações consulte [Bie09, Cap. 3]].

1.4.3 Método de Jacobi Amortecido (ω -Jac)

Nos métodos de Jacobi e Gauss-Seidel fornecemos correções do sistema de equações (1.9) para um vetor $\mathbf{u}_h^{(k-1)}$ [lembre-se da discussão na seção sobre o Método de Jacobi]. Antes da correção, vimos que (1.9) não é verdadeira, em geral, para $\mathbf{u}_h^{(k-1)}$. Tudo acontece como se as equações de (1.9) não se ajustassem muito bem para fornecer o resultado desejado sobre $\mathbf{u}_h^{(k-1)}$. Essa falta de ajustamento cria uma espécie de tensão no sistema que é relaxada quando realizamos a correção devida. Por esse motivo, os métodos de Jacobi e Gauss-Seidel são denominados métodos de correção ou métodos de relaxamento.

Em muitos casos, a convergência pode ser significativamente acelerada através de um sobrerelaxamento. Nesse aspecto, para o método de Jacobi, a idéia seria começar denotando por $[\hat{\mathbf{u}}_h^{(k)}]_j$ [ao invés de $[\mathbf{u}_h^{(k)}]_j$] a j -ésima componente da k -ésima iteração do método de Jacobi como definida em (1.11) para tomar, em seguida, $\delta = [\hat{\mathbf{u}}_h^{(k)}]_j - [\mathbf{u}_h^{(k-1)}]_j$ [i.e., a “distância” entre $[\hat{\mathbf{u}}_h^{(k)}]_j$ e $[\mathbf{u}_h^{(k-1)}]_j$]. Por fim, realizamos o sobrerelaxamento adotando o fator ω (denominado fator de sobrerelaxamento) para definir $[\mathbf{u}_h^{(k)}]_j = [\mathbf{u}_h^{(k-1)}]_j + \omega\delta$. Em suma, ficamos com:

$$[\mathbf{u}_h^{(k)}]_j = [\mathbf{u}_h^{(k-1)}]_j + \omega \left([\hat{\mathbf{u}}_h^{(k)}]_j - [\mathbf{u}_h^{(k-1)}]_j \right) \quad (1.18)$$

1.4.3.1 Matriz de Iteração

Utilizando (1.12) e o fato que $R_J = I - \frac{h^2}{2}A_h$, podemos deduzir da definição (1.18) que:

$$\begin{aligned} \mathbf{u}_h^{(k)} &= \mathbf{u}_h^{(k-1)} + \omega(\hat{\mathbf{u}}_h^{(k)} - \mathbf{u}_h^{(k-1)}) \\ &= \mathbf{u}_h^{(k-1)} + \omega(R_J \mathbf{u}_h^{(k-1)} + \frac{h^2}{2}\mathbf{f}_h - \mathbf{u}_h^{(k-1)}) \\ &= \mathbf{u}_h^{(k-1)} + \omega[(I - \frac{h^2}{2}A_h)\mathbf{u}_h^{(k-1)} + \frac{h^2}{2}\mathbf{f}_h - \mathbf{u}_h^{(k-1)}] \end{aligned}$$

e, portanto, após as devidas simplificações, que:

$$\mathbf{u}_h^{(k)} = R(\omega) \cdot \mathbf{u}_h^{(k-1)} + \frac{h^2}{2}\mathbf{f}_h \quad (1.19)$$

onde $R(\omega) = I - \omega\frac{h^2}{2}A_h$. Observe que o método de Jacobi ocorre como um caso especial do Jacobi Amortecido quando $\omega = 1$.

1.4.3.2 Autovalores e Autovetores da Matriz de Iteração

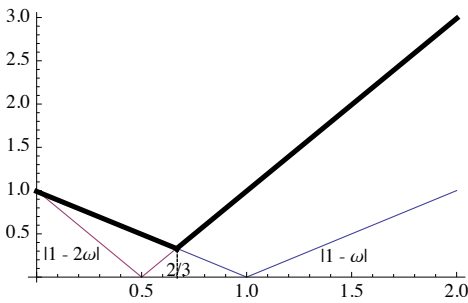
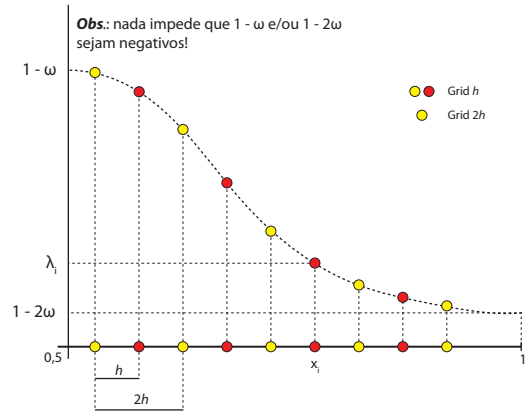
Assim como no caso de Jacobi, o teorema a seguir revela que os autovetores da matriz de iteração $R(\omega)$ do método de Jacobi Amortecido são exatamente os autovetores da matriz de discretização A_h .

Proposition 1.6. *A matriz de iteração $R(\omega)$ do método de Jacobi Amortecido para o problema (1.6) possui autovalores $\lambda_i = 1 - \omega(1 - \cos(\frac{i\pi h}{L})) = 1 - 2\omega \sin^2(\frac{i\pi h}{2L})$ e autovetores correspondentes $\mathbf{v}_i = (\sin(\frac{i\pi x_1}{L}), \dots, \sin(\frac{i\pi x_{n-1}}{L}))$ com $i = 1, 2, \dots, n-1$, ou seja, os mesmo autovetores da matriz de discretização A_h .*

Demonstração. Os autovetores de $R(\omega)$ são aqueles vetores $\mathbf{v}$ tais que $R(\omega)\mathbf{v} = \lambda\mathbf{v}$ para algum $\lambda \neq 0$. Agora, como $R(\omega) = I - \omega \frac{h^2}{2} A_h$, então $R(\omega)\mathbf{v} = \lambda\mathbf{v} \implies A_h\mathbf{v} = \frac{2}{\omega h^2}(1 - \lambda)\mathbf{v}$ e, portanto, que os autovetores de $R(\omega)$ são os mesmos de A_h . Além disso, a expressão $\frac{2}{\omega h^2}(1 - \lambda)$ fornece exatamente os autovalores de A_h , de modo que $\frac{2}{\omega h^2}(1 - \lambda) = \frac{2}{h^2}(1 - \cos(\frac{i\pi h}{L}))$ e, portanto, que $\lambda_i = 1 - \omega(1 - \cos(\frac{i\pi h}{L}))$. Por fim, lembre-se que pelas identidades trigonométricas sabemos que $\cos(2a) = \cos^2 a - \sin^2 a = 1 - 2\sin^2 a$ e, portanto, que $2\sin^2 a = 1 - \cos(2a)$. Dessa forma, se tomarmos $2a = \frac{i\pi h}{L}$ na expressão de λ_i , então podemos escrever $\lambda_i = 1 - 2\omega \sin^2(\frac{i\pi h}{L})$. $\square$

1.4.3.3 Valor Ótimo de Suavização

Como fizemos antes para o método de Jacobi, permita-nos escrever os autovalores de $R(\omega)$ como $\lambda(\omega, x_i) = 1 - 2\omega \sin^2(\frac{\pi}{2}x_i)$ com $x_i = \frac{ih}{L} = \frac{i}{n}$ e $i = 1, \dots, n-1$. Agora, se fixarmos ω e restringirmos x ao intervalo $[\frac{1}{2}, 1]$ (pois é neste intervalo que temos $i > \frac{n}{2}$ e, portanto, onde os autovalores estão associados a autofunções de frequência elevada), então teremos que a função de uma variável $\lambda(x) = \lambda(\omega, x)$ é estritamente decrescente, uma vez que a função $\sin(\frac{\pi}{2}x)$ é estritamente crescente em $[\frac{1}{2}, 1]$. Isso significa que a função $\lambda(x)$ atinge seu máximo em $x = \frac{1}{2}$ (onde obtemos $\lambda(\omega) = 1 - \omega$) e seu mínimo em $x = 1$ (onde obtemos $\lambda(\omega) = 1 - 2\omega$). Observe que esses extremos são cotas inferior e superior para todos os autovalores λ_i de $R(\omega)$ que viermos a obter variando a quantidade de pontos no grid e, portanto, $1 - 2\omega \leq \lambda_i \leq 1 - \omega$ será válido em absolutamente qualquer grid (veja a figura ao lado). Dessa forma, não é muito difícil de ver que em qualquer grid o maior autovalor em módulo dentre aqueles com $i > \frac{n}{2}$ (i.e., na região de alta frequência) será menor que $\max\{|1 - 2\omega|, |1 - \omega|\}$, isto é, menor que o fator de suavização.



Amortecido.

Assim, se escolhermos ω de tal forma que o fator de suavização $\max\{|1 - 2\omega|, |1 - \omega|\}$ seja o menor possível, então garantimos que, em qualquer grid, estremos eliminando rapidamente todas as componentes de alta frequência do erro. Plotando o gráfico das funções $|1 - 2\omega|$ e $|1 - \omega|$ (ao lado), podemos ver que $\max\{|1 - 2\omega|, |1 - \omega|\}$ é justamente o traço mais grosso na figura e que atingimos o seu mínimo quando $\omega = \frac{2}{3}$. Portanto, dizemos que esse é o valor ótimo de suavização para o método de Jacobi

1.4.4 Método de Gauss-Seidel Sobrerelaxado (ω -GS ou SOR)

Ao invés de utilizarmos o método de Jacobi no sobrerelaxamento dado em (1.18) poderíamos utilizar o método de Gauss-Seidel, obtendo aquilo que é denominado método SOR (successive overrelaxation).

1.4.4.1 Matriz de Iteração

Da equação (1.18) com $\hat{\mathbf{u}}_h^{(k)}$ sendo o k -ésimo vetor obtido na iteração de Gauss-Seidel e utilizando a expressão em (1.17) obtemos que:

$$\begin{aligned}
 \mathbf{u}_h^{(k)} &= \mathbf{u}_h^{(k-1)} + \omega(\hat{\mathbf{u}}_h^{(k)} - \mathbf{u}_h^{(k-1)}) \\
 &= \mathbf{u}_h^{(k-1)} + \omega(L\mathbf{u}_h^{(k)} + U\mathbf{u}_h^{(k-1)} + \frac{h^2}{2}\mathbf{f}_h - \mathbf{u}_h^{(k-1)}) \\
 &= \mathbf{u}_h^{(k-1)} + \omega[L\mathbf{u}_h^{(k)} + (U - I)\mathbf{u}_h^{(k-1)} + \frac{h^2}{2}\mathbf{f}_h] \\
 &= [I + \omega(U - I)]\mathbf{u}_h^{(k-1)} + \omega L\mathbf{u}_h^{(k)} + \omega\frac{h^2}{2}\mathbf{f}_h \\
 (I - \omega L)\mathbf{u}_h^{(k)} &= [I + \omega(U - I)]\mathbf{u}_h^{(k-1)} + \frac{\omega h^2}{2}\mathbf{f}_h
 \end{aligned}$$

e, portanto, que

$$\mathbf{u}_h^{(k)} = R_{SOR}\mathbf{u}_h^{(k-1)} + \frac{\omega h^2}{2}(I - \omega L)^{-1}\mathbf{f}_h \quad (1.20)$$

onde $R_{SOR} = (I - \omega L)^{-1}[(1 - \omega)I + \omega U]$.

1.4.4.2 Autovalores e Autovetores da Matriz de Iteração

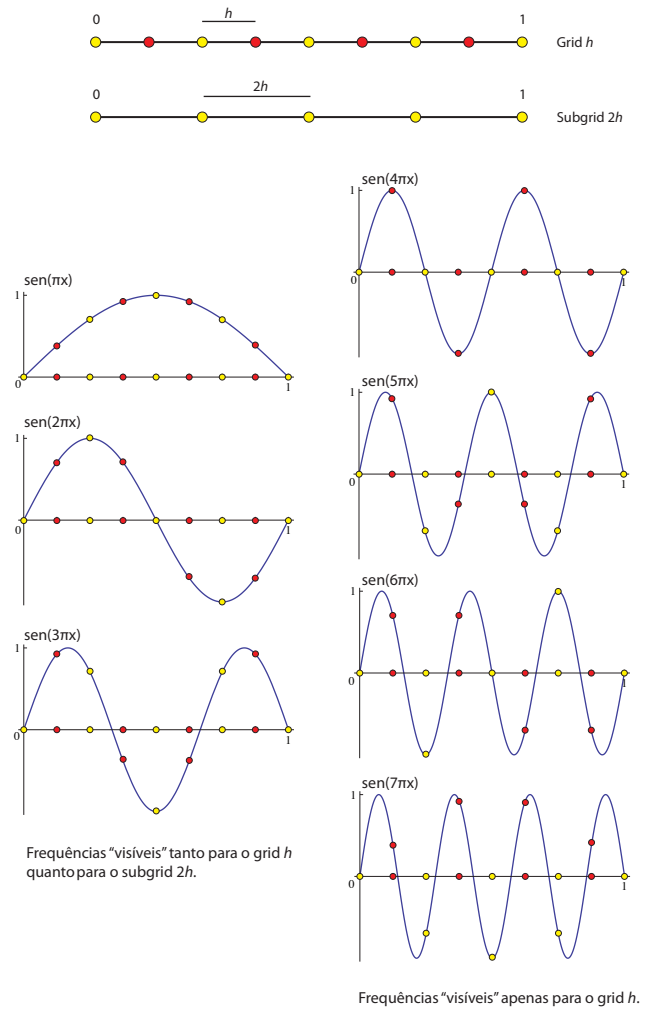
Proposition 1.7. *A matriz de iteração R_{SOR} do método SOR para o problema (1.6) possui autovalores λ_i que satisfazem $\lambda_i^{(J)} = (\frac{\lambda_i + \omega - 1}{\lambda_i^{1/2}\omega})$ onde $\lambda_i^{(J)}$ é o i -ésimo autovalor da matriz de iteração R_J do método de Jacobi e autovetores $\mathbf{v}_i$ cujas componentes satisfazem $[\mathbf{v}_i]_j = \lambda_i^{j/2} \sin(\frac{i\pi x_j}{L})$ com $i = 1, 2, \dots, n - 1$, ou seja, NÃO coincidindo com os autovetores da matriz de discretização A_h .*

Demonstração. Os autovetores de R_{SOR} são aqueles vetores $\mathbf{v}$ tais que $R_{SOR}\mathbf{v} = \lambda\mathbf{v}$ para algum $\lambda \neq 0$. Agora, como $R_{SOR} = (I - \omega L)^{-1}[(1 - \omega)I + \omega U]$, então $R_{SOR}\mathbf{v} = \lambda\mathbf{v} \implies (I - \omega L)^{-1}[(1 - \omega)I + \omega U]\mathbf{v} = \lambda\mathbf{v}$ e, portanto, que $[(1 - \omega)I + \omega U]\mathbf{v} = \lambda(I - \omega L)\mathbf{v}$. Agora, se observarmos a forma das matrizes L e U em (1.15), então veremos que o anterior significa $(1 - \omega)[\mathbf{v}]_j + \frac{\omega}{2}[\mathbf{v}]_{j+1} = \lambda([\mathbf{v}]_j - \frac{\omega}{2}[\mathbf{v}]_{j-1})$ e, portanto, que $\lambda\frac{1}{2}[\mathbf{v}]_{j-1} + \frac{1}{2}[\mathbf{v}]_{j+1} = (\frac{\lambda + \omega - 1}{\omega})[\mathbf{v}]_j$. Das expressões (1.11) e (1.12) vemos que o lado esquerdo da igualdade anterior é muito parecido com o resultado de aplicarmos R_J (a matriz de Jacobi) ao vetor $\mathbf{v}$. De fato, se fizermos a substituição $[\mathbf{v}]_j = \lambda^{\frac{j}{2}}[\mathbf{w}]_j$, então obteremos $\frac{1}{2}\lambda^{\frac{j+1}{2}}[\mathbf{w}]_{j-1} + \frac{1}{2}\lambda^{\frac{j+1}{2}}[\mathbf{w}]_{j+1} = \lambda^{\frac{j}{2}}(\frac{\lambda + \omega - 1}{\omega})[\mathbf{w}]_j$, de modo que, dividindo tudo por $\lambda^{\frac{j+1}{2}}$, tenhamos $\frac{1}{2}[\mathbf{w}]_{j-1} + \frac{1}{2}[\mathbf{w}]_{j+1} = (\frac{\lambda + \omega - 1}{\lambda^{1/2}\omega})[\mathbf{w}]_j$ e, portanto, que $R_J\mathbf{w} = (\frac{\lambda + \omega - 1}{\lambda^{1/2}\omega})\mathbf{w}$. Assim, concluímos que $\frac{\lambda + \omega - 1}{\lambda^{1/2}\omega}$ é um autovalor de R_J cujo autovetor correspondente é $\mathbf{w}$ [e, portanto, $\mathbf{w}$ é autovetor de A_h]. Disso segue-se que $\lambda_i^{(J)} = (\frac{\lambda_i + \omega - 1}{\lambda_i^{1/2}\omega})$ e que as componentes do autovetor $\mathbf{v}_i$ de R_{SOR} satisfazem $[\mathbf{v}_i]_j = \cos(\frac{i\pi h}{L}) \cdot \sin(\frac{i\pi x_j}{L})$ (já que fizemos a substituição $[\mathbf{v}]_j = \lambda^{\frac{j}{2}}[\mathbf{w}]_j$ e $\mathbf{w}$ é autovetor de A_h). $\square$

1.5 Bigrid

Sabemos que é preciso um grid com muitos pontos ou, equivalentemente, com espaçamento h muito pequeno entre os pontos, para que a solução discreta $\mathbf{u}_h$ do sistema $A_h \mathbf{u}_h = \mathbf{f}_h$ seja uma boa aproximação da solução contínua $u(x)$ de nosso problema (1.6). Ora, mas nas seções anteriores observamos que a convergência dos métodos iterativos geralmente torna-se muito lenta quando a matriz A_h é muito grande, embora algum desses métodos (os suavizadores) possuam a propriedade de eliminar rapidamente as componentes de alta frequência do erro inicial. Essa propriedade peculiar dos métodos suavizadores é a primeira peça fundamental do quebra-cabeça "Multigrid".

A segunda peça fundamental do jogo é a observação de que componentes de baixa frequência num grid h são componentes de frequência elevada num (sub)grid $2h$ com o dobro do espaçamento do grid original (veja a figura ao lado)!!! Suponha, por exemplo, um grid com 9 pontos no intervalo $[0, 1]$ e, portanto, com $n = 8$ (lembre-se da seção 1.1; para esse grid temos $h = L/n = 1/8$). Da nossa discussão sobre os autovetores da matriz de discretização A_h (lembre-se da Proposição 1.1 e da Observação que lhe segue), sabemos que A_h possui 7 autovetores $\mathbf{v}_i$ com $i = 1, \dots, 7$ e que esses autovetores são as 7 primeiras autofunções do Laplaciano discretizadas (i.e., as componentes do i -ésimo autovetor são os valores que a i -ésima autofunção assume nos pontos da grid). Assim, podemos dizer que existem 7 frequências nesse grid, cada uma das quais associada a uma das autofunções $\text{sen}(i\pi x)$ com $i = 1, \dots, 7$ (observe a figura ao lado). Dessas frequências, apenas as 3 primeiras são "visíveis" para o grid $2h$ (que possui 5 pontos, de modo que $n = 4$ e, portanto, estão presentes apenas as 3 primeiras autofunções). Agora, como convencionamos que frequência alta está associada a autofunções $\text{sen}(i\pi x)$ tais que $i > \frac{n}{2}$, então temos que $\text{sen}(3\pi x)$ é de baixa frequência no grid h (pois $3 < 8/2$), mas é de alta frequência no grid $2h$ (pois $4/2 < 3$). Esse fato sugere uma estratégia interessante! Como um método iterativo suavizador elimina rapidamente as componentes de alta frequência do erro inicial $\mathbf{e}_h^{(0)}$ e deixa as componentes de baixa frequência praticamente inalteradas, então seria interessante que após algumas poucas iterações para eliminar as componentes de alta frequência transportássemos o problema para um subgrid com menos



pontos, onde o método é capaz de eliminar rapidamente componentes que eram de baixa frequência no grid original!

Para tanto, entra em cena a segunda peça importante do jogo: a equação do resíduo. Observe que o erro algébrico $\mathbf{e}_h = \mathbf{u}_h - \mathbf{u}_h^{(k)}$ de um iterado $\mathbf{u}_h^{(k)}$ possui importância apenas teórica, uma vez não o conhecemos de fato (para tanto, deveríamos conhecer a própria solução exata $\mathbf{u}_h$!). Dessa forma, para se medir a “distância” entre o iterado $\mathbf{u}_h^{(k)}$ e a solução exata $\mathbf{u}_h$ utilizamos aquilo que é denominado seu *resíduo* $\mathbf{r}_h = \mathbf{f}_h - A_h \mathbf{u}_h^{(k)}$. Não é muito difícil de ver que o erro algébrico e o resíduo de $\mathbf{u}_h^{(k)}$ estão relacionados pela *equação do resíduo*: $A_h \mathbf{e}_h = \mathbf{r}_h$ (basta substituir $\mathbf{u}_h^{(k)} = \mathbf{u}_h - \mathbf{e}_h$ obtido da definição do erro algébrico na definição do resíduo). Agora, observe que resolver $A_h \mathbf{u}_h = \mathbf{f}_h$ é equivalente a resolver $A_h \mathbf{e}_h = \mathbf{r}_h$ relacionada a um iterado $\mathbf{u}_h^{(k)}$, uma vez que em ambos os casos somos levados a conhecer a solução exata $\mathbf{u}_h$ (no segundo caso teremos $\mathbf{u}_h = \mathbf{u}_h^{(k)} + \mathbf{e}_h$). O processo de resolver o problema $A_h \mathbf{u}_h = \mathbf{f}_h$ através da equação do resíduo pode ser esquematicamente representado como (seguindo [TOS01]):

$$\mathbf{u}_h^{(k)} \longrightarrow \mathbf{r}_h = \mathbf{f}_h - A_h \mathbf{u}_h^{(k)} \longrightarrow A_h \mathbf{e}_h = \mathbf{r}_h \longrightarrow \mathbf{u}_h = \mathbf{u}_h^{(k)} + \mathbf{e}_h \quad (1.21)$$

Esse processo em si não é significativo numericamente falando. Contudo, se pudermos aproximar A_h por um $\hat{A}_h$ mais “simples”, então a solução de $\hat{A}_h \hat{\mathbf{e}}_h = \mathbf{r}_h$ será uma aproximação de $\mathbf{e}_h$, de modo que possamos utilizar o esquema (1.21) para obter um novo iterado $\mathbf{u}_h^{(k+1)}$ e, portanto, um método iterativo, da seguinte forma:

$$\mathbf{u}_h^{(k)} \longrightarrow \mathbf{r}_h = \mathbf{f}_h - A_h \mathbf{u}_h^{(k)} \longrightarrow \hat{A}_h \hat{\mathbf{e}}_h = \mathbf{r}_h \longrightarrow \mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)} + \hat{\mathbf{e}}_h \quad (1.22)$$

Uma das opções mais evidentes seria tentar aproximar $A_h \mathbf{e}_h = \mathbf{r}_h$ através de $A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h}$. Isso nos leva a considerar a terceira peça fundamental do quebra-cabeça: os operadores de transferência intergrids. Para converter um vetor $\mathbf{x}_h$ do grid h a um vetor $\mathbf{x}_{2h}$ do subgrid $2h$ definimos um operador $I_{h \rightarrow 2h}$ (denominado o operador de *restrição*) tal que $\mathbf{x}_{2h} = I_{h \rightarrow 2h} \mathbf{x}_h$. Uma das opções mais simples seria tomar $I_{h \rightarrow 2h}$ para ser o operador de injeção que faz apenas

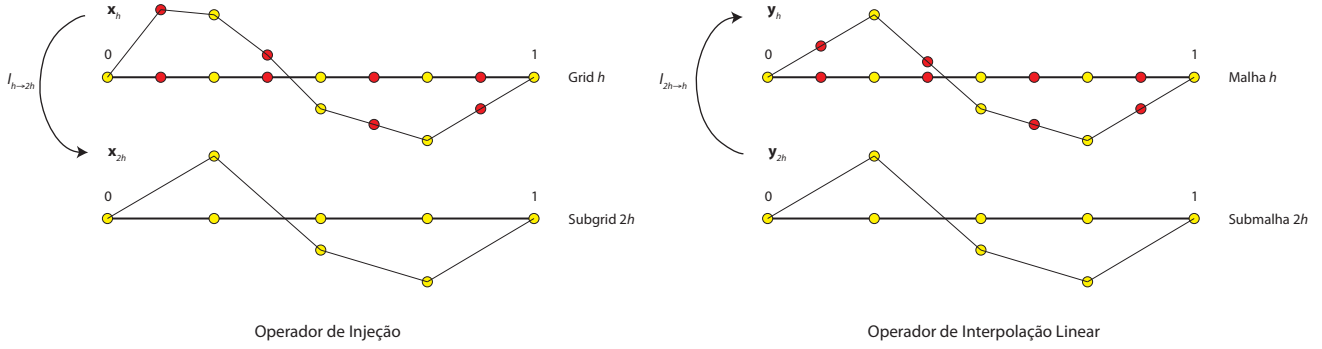
$$[\mathbf{x}_{2h}]_j = [\mathbf{x}_h]_{2j} \quad (1.23)$$

ou seja, que “constrói” $\mathbf{x}_{2h}$ simplesmente tomando as componentes de $\mathbf{x}_h$ associadas aos pontos do grid $2h$ (veja a figura abaixo). Da mesma forma, para converter um vetor $\mathbf{y}_{2h}$ do subgrid $2h$ a um vetor $\mathbf{y}_h$ do grid h definimos um operador $I_{2h \rightarrow h}$ (denominado operador de *extensão* ou *prolongamento*) tal que $\mathbf{y}_h = I_{2h \rightarrow h} \mathbf{y}_{2h}$. Uma das opções mais utilizadas consiste em tomar $I_{2h \rightarrow h}$ para ser o operador de interpolação linear que faz

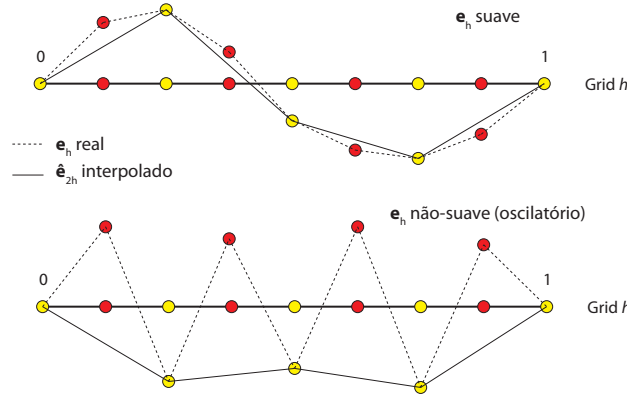
$$\begin{cases} [\mathbf{y}_h]_{2j} = [\mathbf{y}_{2h}]_j & 1 \leq j \leq \frac{n}{2} - 1 \\ [\mathbf{y}_h]_1 = \frac{1}{2} [\mathbf{y}_{2h}]_1 \\ [\mathbf{y}_h]_{2j+1} = \frac{1}{2} ([\mathbf{y}_{2h}]_j + [\mathbf{y}_{2h}]_{j+1}) & 1 \leq j \leq \frac{n}{2} - 1 \end{cases} \quad (1.24)$$

ou seja, que “constrói” $\mathbf{y}_h$ tomando as componentes de $\mathbf{y}_{2h}$ para serem as componentes pares de $\mathbf{y}_h$ (as quais são

associadas aos pontos do grid h que também estão no grid $2h$) e as médias de componentes adjacentes de $\mathbf{y}_{2h}$ para serem as componentes ímpares de $\mathbf{y}_h$ (as quais são associadas aos pontos do grid h que não estão presentes no grid $2h$). Para fixar as idéias, veja as figuras abaixo que representam esquematicamente a transferência de vetores entre os grids:



Dessa forma, poderíamos tentar aproximar $A_h \mathbf{e}_h = \mathbf{r}_h$ da seguinte forma: [1] começamos por transferir o resíduo $\mathbf{r}_h$ para o grid $2h$ fazendo $\mathbf{r}_{2h} = I_{h \rightarrow 2h} \mathbf{r}_h$; [2] em seguida, resolvemos $A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h}$ exatamente para obter $\hat{\mathbf{e}}_{2h} = A_{2h}^{-1} \mathbf{r}_{2h}$; [3] por fim, transferimos esse $\hat{\mathbf{e}}_{2h}$ de volta para o grid h fazendo $\hat{\mathbf{e}}_h = I_{2h \rightarrow h} \hat{\mathbf{e}}_{2h}$. Se esse "interpolado" $\hat{\mathbf{e}}_h$ for uma boa aproximação da solução de $A_h \mathbf{e}_h = \mathbf{r}_h$, então diremos que essa equação é bem aproximada pela equação mais simples $A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h}$. Agora, o interessante é que, se o erro $\mathbf{e}_h$ associado a um iterado $\mathbf{u}_h^{(k)}$ é suave (i.e., não possui componentes de alta frequência), então a interpolação de $\hat{\mathbf{e}}_{2h}$ realmente fornece uma boa aproximação de $\mathbf{e}_h$! Para tanto, observe a figura abaixo.



Com essa abordagem, o processo iterativo expresso em (1.22) ficaria da seguinte forma:

$$\begin{array}{rclcl}
 \text{Grid } h & \mathbf{u}_h^{(k)} & \longrightarrow & \mathbf{r}_h = \mathbf{f}_h - A_h \mathbf{u}_h^{(k)} & \hat{\mathbf{e}}_h = I_{2h \rightarrow h} \hat{\mathbf{e}}_{2h} \longrightarrow \mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)} + \hat{\mathbf{e}}_h \\
 & & & \downarrow & \uparrow \\
 \text{Subgrid } 2h & & & \mathbf{r}_{2h} = I_{h \rightarrow 2h} \mathbf{r}_h & \hat{\mathbf{e}}_{2h} = A_{2h}^{-1} \mathbf{r}_{2h}
 \end{array} \quad (1.25)$$

o qual chamaremos de uma **Iteração Bigrid** ou um **Esquema de Correção**. Com isso em mãos, obtemos aquilo que é denominado um **Ciclo Bigrid**, unidade básica de qualquer método Multigrid, o qual consiste das seguintes três etapas:

1. [Pre-suavização] Aplique um método iterativo suavizador m_1 vezes a $A_h \mathbf{u}_h = \mathbf{f}_h$ com chute inicial $\mathbf{u}_h^{(0)}$ para obter $\mathbf{u}_h^{(k_1)}$.
2. [Iteração Bigrid] Aplique a Iteração Bigrid (1.25) a $\mathbf{u}_h^{(k_1)}$ para obter $\mathbf{u}_h^{(k_1+1)}$.
3. [Pós-suavização] Aplique o método iterativo suavizador m_2 vezes a $A_h \mathbf{u}_h = \mathbf{f}_h$ com chute inicial $\mathbf{u}_h^{(k_1+1)}$ para obter $\mathbf{u}_h^{(k_2)}$.

Se observarmos cada passo da Iteração Bigrid (1.25), poderemos deduzir que [verifique!]:

$$\mathbf{u}_h^{(k+1)} = R_{BG} \mathbf{u}_h^{(k)} + I_{2h \rightarrow h} A_{2h}^{-1} I_{h \rightarrow 2h} \mathbf{f}_h \quad (1.26)$$

onde $R_{BG} = I - I_{2h \rightarrow h} A_{2h}^{-1} I_{h \rightarrow 2h} A_h$. Denotando por R a matriz de iteração do método suavizador escolhido, então não é muito difícil ver que podemos escrever $\mathbf{u}_h^{(k)} = R^k \mathbf{u}_h^{(0)} + C(\mathbf{f}_h)$ onde $C(\mathbf{f}_h)$ representa um conjunto de operações sobre $\mathbf{f}_h$ (para ver isso, faça $\mathbf{u}_h^{(1)} = R \mathbf{u}_h^{(0)} + B \mathbf{f}_h$, $\mathbf{u}_h^{(2)} = R \mathbf{u}_h^{(1)} + B \mathbf{f}_h = R^2 \mathbf{u}_h^{(0)} + (R + B) \mathbf{f}_h$ e perceba o resultado geral). Utilizando essa observação e a equação (1.26), obtemos a matriz BG de um Ciclo Bigrid (verifique!):

$$BG = R^{m_2} R_{BG} R^{m_1} \quad (1.27)$$

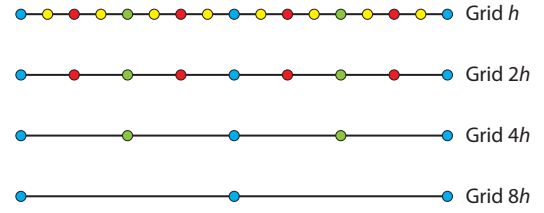
A convergência do processo Bigrid depende fortemente dos elementos de BG , ou seja, daquilo que denominamos as *componentes* de um ciclo bigrid:

- O método iterativo suavizador, isto é, a matriz R ;
- Os números m_1 e m_2 de iterações suavizadoras;
- O subgrid e, portanto, a matriz de discretização desse subgrid;
- Os operadores de transferência intergrids.

Além disso, um pouco de estudo adicional sobre os efeitos de um ciclo bigrid no erro inicial revelaria que a iteração bigrid em si (i.e., um ciclo bigrid sem as etapas de pré-suavização e pós-suavização) é responsável pela eliminação das componentes de baixa frequência do erro, enquanto que as etapas de pré e pós-suavização são responsáveis pela eliminação das componentes de alta frequência do erro. O leitor interessado deve ler o Capítulo 5 de [BHM00].

1.6 Multigrid

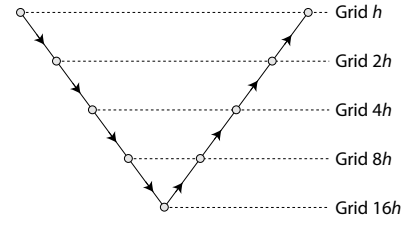
Observe que na discussão anterior sobre o ciclo Bigrid resolvemos a equação do resíduo $A_{2h}\mathbf{e}_{2h} = \mathbf{r}_{2h}$ exatamente no subgrid $2h$. Contudo, como tomamos um grid h com muitos pontos, então certamente o subgrid $2h$ ainda terá uma quantidade elevada de pontos, de modo que resolver $A_{2h}\mathbf{e}_{2h} = \mathbf{r}_{2h}$ ainda seja tão difícil quanto resolver o problema original $A_h\mathbf{u}_h = \mathbf{f}_h$! Mas observe



que esses problemas são essencialmente iguais! Poderíamos até mesmo utilizar a mesma notação do problema original e escrever $A_{2h}\mathbf{u}_{2h} = \mathbf{f}_{2h}$, onde $\mathbf{u}_{2h}$ é o erro $\mathbf{e}_{2h}$ e $\mathbf{f}_{2h}$ é o resíduo $\mathbf{r}_{2h}$. Isso sugere que apliquemos o método Bigrid novamente, mas agora a $A_{2h}\mathbf{e}_{2h} = \mathbf{r}_{2h}$ e rumo a um subgrid $4h$. Isso nos levará ao problema de resolver $A_{4h}\mathbf{e}_{4h} = \mathbf{r}_{4h}$, para o qual podemos aplicar novamente o ciclo Bigrid. O esquema geral já deve estar claro. Prosseguimos com esse processo rumo ao subgrid com apenas 1 ponto interior, onde o problema se reduz a encontrar e em $ae = r$ e, portanto, que pode ser resolvido exatamente. Dessa forma, o procedimento consiste nas seguintes etapas [para simplificar as coisas, utilize a mesma notação que o problema original como feito anteriormente]:

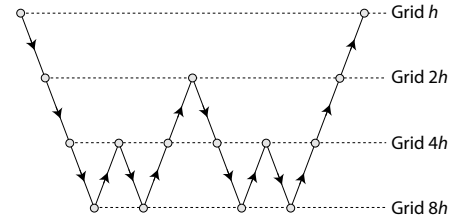
1. Aplique m_1 vezes um método iterativo suavizador a $A_h\mathbf{u}_h = \mathbf{f}_h$ no grid h com chute inicial $\mathbf{u}_h^{(0)}$ para obter $\mathbf{u}_h^{(k)}$.
2. Obtenha o resíduo $\mathbf{r}_h = \mathbf{f}_h - A_h\mathbf{u}_h^{(k)}$ e o transfira para o subgrid $2h$ fazendo $\mathbf{f}_{2h} = I_{h \rightarrow 2h}\mathbf{r}_h$.
3. Aplique m_1 vezes o método iterativo suavizador a $A_{2h}\mathbf{u}_{2h} = \mathbf{f}_{2h}$ no grid $2h$ com chute inicial $\mathbf{u}_{2h}^{(0)} = \mathbf{0}$ para obter $\mathbf{u}_{2h}^{(k)}$.
4. Obtenha o resíduo $\mathbf{r}_{2h} = \mathbf{f}_{2h} - A_{2h}\mathbf{u}_{2h}^{(k)}$ e o transfira para o subgrid $4h$ fazendo $\mathbf{f}_{4h} = I_{2h \rightarrow 4h}\mathbf{r}_{2h}$.
5. $\vdots$
6. Resolva $A_{th}\mathbf{u}_{th} = \mathbf{f}_{th}$ exatamente no grid th com apenas 1 ponto interior (i.e., o grid em si possui 3 pontos).
7. $\vdots$
8. Transfira $\hat{\mathbf{u}}_{4h}$ para o grid $2h$ fazendo $\hat{\mathbf{u}}_{2h} = I_{4h \rightarrow 2h}\hat{\mathbf{u}}_{4h}$ e obtenha $\mathbf{u}_{2h}^{(k+1)} = \mathbf{u}_{2h}^{(k)} + \hat{\mathbf{u}}_{2h}$.
9. Aplique m_2 vezes o método iterativo suavizador a $A_{2h}\mathbf{u}_{2h} = \mathbf{f}_{2h}$ no grid $2h$ com chute inicial $\mathbf{u}_{2h}^{(0)} = \mathbf{u}_{2h}^{(k+1)}$ para obter $\hat{\mathbf{u}}_{2h}$.
10. Transfira $\hat{\mathbf{u}}_{2h}$ para o grid h fazendo $\hat{\mathbf{u}}_h = I_{2h \rightarrow h}\hat{\mathbf{u}}_{2h}$ e obtenha $\mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)} + \hat{\mathbf{u}}_h$.
11. Aplique m_2 vezes o método iterativo suavizador a $A_h\mathbf{u}_h = \mathbf{f}_h$ no grid h com chute inicial $\mathbf{u}_h^{(0)} = \mathbf{u}_h^{(k+1)}$ para obter $\hat{\mathbf{u}}_h$.

Essas etapas definem aquilo que denominamos um V-ciclo Multigrid (obs.: de posse das matrizes BG dos ciclos bigrid, poderíamos até mesmo deduzir a matriz MG que representa o ciclo multigrid), uma vez que sua representação esquemática (figura ao lado) possui a forma de um V. Esse ciclo Multigrid elimina todas as componentes do erro inicial de forma bem eficiente, pois deixa as contas "mais pesadas" (i.e., aquelas envolvidas com a resolução de $A_{th}\mathbf{u}_{th} = \mathbf{f}_{th}$) para grids com um menor número de pontos (onde as operações são mais baratas). Seguindo [BHM00], uma forma mais compacta e algorítmica de se escrever as etapas de um v-ciclo Multigrid seria:



1. Aplique m_1 vezes um método iterativo suavizador a $A_h\mathbf{u}_h = \mathbf{f}_h$ no grid h com chute inicial $\mathbf{u}_h^{(0)}$ para obter $\mathbf{u}_h^{(k)}$.
2. Se o grid h só possui 1 ponto interior, então vá para a etapa número 5 tomando $\mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)}$; caso contrário continue.
3. Obtenha o resíduo $\mathbf{r}_h = \mathbf{f}_h - A_h\mathbf{u}_h^{(k)}$ e o transfira para o subgrid $2h$ fazendo $\mathbf{f}_{2h} = I_{h \rightarrow 2h}\mathbf{r}_h$.
4. Aplique o v-ciclo agora ao grid $2h$ [ou seja, comece novamente em 1 só que agora pra o grid $2h$] $\mu = 1$ vezes.
5. Transfira $\hat{\mathbf{u}}_{2h}$ para o grid h fazendo $\hat{\mathbf{u}}_h = I_{2h \rightarrow h}\hat{\mathbf{u}}_{2h}$ e obtenha $\mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)} + \hat{\mathbf{u}}_h$.
6. Aplique m_2 vezes o método iterativo suavizador a $A_h\mathbf{u}_h = \mathbf{f}_h$ no grid h com chute inicial $\mathbf{u}_h^{(0)} = \mathbf{u}_h^{(k+1)}$ para obter $\hat{\mathbf{u}}_h$.

O V-ciclo Multigrid é apenas um de uma família de ciclos Multigrid que são denominados μ -ciclos Multigrid. Um μ -ciclo arbitrário é obtido quando utilizamos um μ inteiro arbitrário ao invés de 1 na etapa 4 do V-ciclo como expresso anteriormente. Na prática, utiliza-se apenas $\mu = 1$ (v-ciclo) e $\mu = 2$ (denominado W-ciclo e representado na figura ao lado; observando o algoritmo acima para o v-ciclo, substitua $\mu = 2$ e tente visualizar o porquê da representação ao lado ter a forma que tem), os quais nos levam a obter uma solução $\hat{\mathbf{u}}_h$ muito próxima da solução verdadeira quando os aplicamos algumas poucas vezes ao problema original $A_h\mathbf{u}_h = \mathbf{f}_h$.



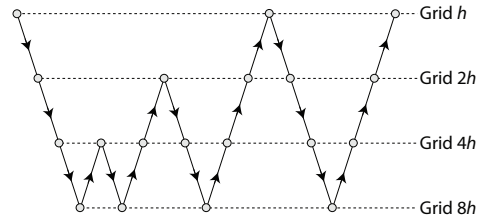
1.6.1 Multigrid Completo (Full Multigrid - FMG)

No que foi dito anteriormente sobre o μ -ciclo Multigrid, observe que podemos aprimorar significativamente esse método se utilizarmos um chute inicial $\mathbf{u}_h^{(0)}$ melhorado (i.e., um pouco mais próximo da solução $\mathbf{u}_h$) na iteração sobre $A_h\mathbf{u}_h = \mathbf{f}_h$ do grid h original (etapa (1)). Para tanto, uma das alternativas mais plausíveis seria: [1] transferir $\mathbf{f}_h$ para o subgrid $2h$ por meio de $\mathbf{f}_{2h} = I_{h \rightarrow 2h}\mathbf{f}_h$; [2] resolver $A_{2h}\mathbf{u}_{2h} = \mathbf{f}_{2h}$ aproximadamente para

obter $\hat{\mathbf{u}}_{2h}$; [3] transferir $\hat{\mathbf{u}}_{2h}$ para o grid h por meio de $\hat{\mathbf{u}}_h = I_{2h \rightarrow h} \hat{\mathbf{u}}_{2h}$ e utilizar esse $\hat{\mathbf{u}}_h$ como chute inicial para $A_h \mathbf{u}_h = \mathbf{f}_h$. Agora, claro que em [2] utilizaremos novamente o μ -ciclo para resolver $A_{2h} \mathbf{u}_{2h} = \mathbf{f}_{2h}$ no grid $2h$, o que nos levará novamente ao problema de utilizar um chute inicial $\mathbf{u}_{2h}^{(0)}$ melhorado para essa equação. O esquema geral já deve estar claro e pode ser resumido nas seguintes etapas:

1. Transfira $\mathbf{f}_h$ para o subgrid $2h$ por meio de $\mathbf{f}_{2h} = I_{h \rightarrow 2h} \mathbf{f}_h$.
2. Transfira $\mathbf{f}_{2h}$ para o subgrid $4h$ por meio de $\mathbf{f}_{4h} = I_{2h \rightarrow 4h} \mathbf{f}_{2h}$.
3. $\vdots$
4. Resolva $A_{th} \mathbf{u}_{th} = \mathbf{f}_{th}$ no grid th que possui apenas 1 ponto interior para obter $\hat{\mathbf{u}}_{th}$.
5. $\vdots$
6. Transfira $\hat{\mathbf{u}}_{4h}$ para o grid $2h$ por meio de $\mathbf{u}_{2h}^{(0)} = I_{4h \rightarrow 2h} \hat{\mathbf{u}}_{4h}$.
7. Aplique o μ -ciclo um certo número v_0 de vezes a $A_{2h} \mathbf{u}_{2h} = \mathbf{f}_{2h}$ utilizando $\mathbf{u}_{2h}^{(0)}$ como chute inicial para obter $\hat{\mathbf{u}}_{2h}$.
8. Transfira $\hat{\mathbf{u}}_{2h}$ para o grid h por meio de $\mathbf{u}_h^{(0)} = I_{2h \rightarrow h} \hat{\mathbf{u}}_{2h}$.
9. Aplique o μ -ciclo um certo número v_0 de vezes a $A_h \mathbf{u}_h = \mathbf{f}_h$ utilizando $\mathbf{u}_h^{(0)}$ como chute inicial para obter $\hat{\mathbf{u}}_h$.

Esse processo de obter chutes iniciais aprimorados para os grids mais finos (= com mais pontos) utilizando os grids mais grosseiros (= com menos pontos) é denominado de *Iteração Aninhada* (*Nested Iteration*) e sua fusão com o μ -ciclo Multigrid (expresso nas etapas acima) fornece aquilo que denominamos um μ -ciclo Multigrid Completo ou FMG (Full Multigrid) (representado na figura ao lado quando utilizamos V-ciclos). Esse método constitui o nó que junta as muitas idéias apresentadas nas seções anteriores. Trata-se de "uma síntese impressionante de idéias e técnicas conhecidas e utilizadas individualmente por muito tempo. Sozinhas, muitas dessas idéias possuem sérias deficiências. O Multigrid Completo é a técnica que as integra de uma tal forma que possam trabalhar em conjunto e superar esses obstáculos. O resultado é um algoritmo extremamente poderoso." [BHM00, Cap. 3]



Capítulo 2

O Caso Bidimensional

Embora tenhamos apresentado o Multigrid aplicado ao problema unidimensional (1.1), o poder desse método só fica evidenciado quando aplicado ao caso bidimensional desse mesmo problema. Ou seja, a partir de agora estaremos interessados em resolver o Problema de Dirichlet para a Equação de Poisson Bidimensional, o que significa que estamos buscando uma função $u(x)$ que satisfaça o seguinte sistema:

$$\begin{cases} -u_{xx} - u_{yy} = f & (x, y) \in \Omega \\ u(x, y) = 0 & (x, y) \in \partial\Omega \end{cases} \quad (2.1)$$

onde Ω representa um conjunto aberto do plano $\mathbb{R}^2$ e $\partial\Omega$ representa a fronteira desse conjunto. Por conveniência, tomaremos Ω como sendo o quadrado unitário $(0, 1) \times (0, 1)$.

A expressão anterior poderia ser representada de forma mais geral (e elegante) como

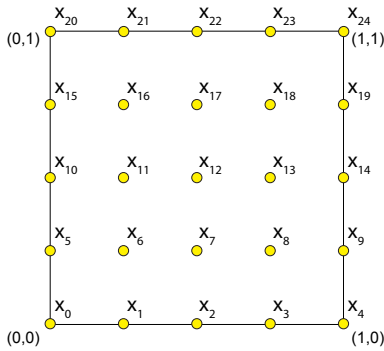
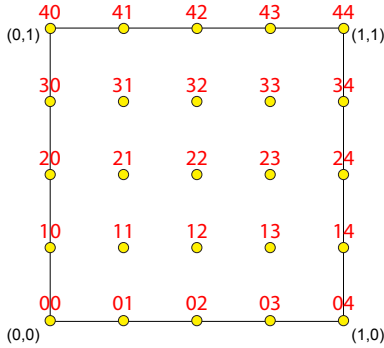
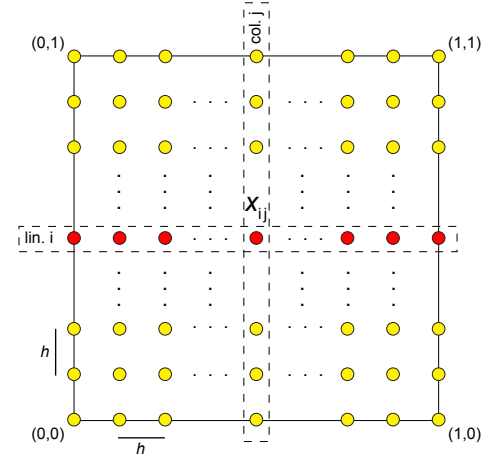
$$\begin{cases} -\Delta u(\mathbf{x}) = f(\mathbf{x}) & \mathbf{x} \in \Omega \\ u(\mathbf{x}) = 0 & \mathbf{x} \in \partial\Omega \end{cases} \quad (2.2)$$

onde $u : \bar{\Omega} \rightarrow \mathbb{R}$ é uma função real definida num conjunto fechado $\bar{\Omega} \subset \mathbb{R}^n$ que leva um $\mathbf{x} = (x_1, \dots, x_n) \in \bar{\Omega}$ no seu $u(\mathbf{x}) \in \mathbb{R}$ e Δ é denominado o operador Laplaciano, o qual leva a função u no seu $\Delta u = \sum_{i=1}^n u_{ii}$, onde u_{ii} representa a derivada segunda de u em relação a sua i -ésima variável. Observe que, no caso particular em que u está definida em $\mathbb{R}^2$, o anterior se resume a $\Delta u = u_{11} + u_{22} = u_{xx} + u_{yy}$ e, portanto, (2.2) se resume a (2.1).

2.1 Discretização do Problema

Assim como no caso unidimensional, para que possamos resolver o problema anterior computacionalmente devemos transformá-lo num sistema de equações lineares cuja solução será uma aproximação da solução $u(x)$ procurada e que poderemos obter via métodos numéricos.

Para tanto, ao invés de trabalhar com o domínio contínuo Ω consideraremos apenas um conjunto finito de pontos de Ω . Já que por comodidade estamos tomando Ω como o quadrado unitário, então podemos seguir o padrão adotado no caso unidimensional e considerar a i -ésima linha de pontos $\{\mathbf{x}_{i0} = (0, ih), \mathbf{x}_{i1} = (h, ih), \dots, \mathbf{x}_{ij} = (jh, ih), \dots, \mathbf{x}_{i(n-1)} = ((n-1)h, ih), \mathbf{x}_{in} = (1, ih)\}$ (pontos em vermelho na figura ao lado; observe que nessa i -ésima linha as ordenadas de todos os pontos são iguais a ih , onde h é o espaçamento vertical que por comodidade coincide com o espaçamento horizontal) e, portanto, considerar nosso grid de espaçamento h (denotado simplesmente por Ω_h) como sendo o conjunto de todas essas linhas de pontos com $i = 0, 1, \dots, n$ (observe que para cada i temos $j = 0, 1, \dots, n$)¹.



Adotando um grid com 5×5 pontos como exemplo, não é muito difícil de ver que os índices ij de seus pontos $\mathbf{x}_{ij}$ estão dispostos tal como a figura ao lado representa. Para que possamos montar o sistema de equações lineares assim como fizemos para o caso unidimensional, a primeira coisa que precisamos decidir é como iremos ordenar o nosso grid (i.e., quais pontos diremos vir antes e quais pontos diremos vir depois). Isso acontece porque, diferentemente da reta [cujos pontos são ordenados de maneira intuitiva da esquerda para a direita], não existe uma forma natural de se organizar os pontos no plano. Uma das ordenações possíveis é conhecida como *ordenação lexicográfica* e ocorre exatamente como nos dicionários convencionais (onde a palavra "ab" vem antes de "ac", a qual, por sua vez, vem antes da palavra "ba"). Nessa ordenação diremos que o ponto $\mathbf{x}_{ij}$ vem antes do ponto $\mathbf{x}_{i'j'}$ quando $i < i'$; no caso em que $i = i'$, então $\mathbf{x}_{ij}$ vem antes de $\mathbf{x}_{i'j'}$ se $j < j'$ (em suma, olhamos o primeiro índice, i.e., o índice i , para tentar decidir quem vem antes de quem; não conseguindo, i.e., caso $i = i'$, passamos a olhar o segundo índice, ou seja, o índice j). Assim, para o nosso grid de 5×5 pontos, teríamos a seguinte ordenação: $\{\mathbf{x}_{00}, \dots, \mathbf{x}_{04}, \mathbf{x}_{10}, \dots, \mathbf{x}_{14}, \mathbf{x}_{20}, \dots, \mathbf{x}_{24}, \mathbf{x}_{30}, \dots, \mathbf{x}_{34}, \mathbf{x}_{40}, \dots, \mathbf{x}_{44}\}$ (para simplificar as coisas, poderíamos denotar $\mathbf{x}_0 = \mathbf{x}_{00}, \dots, \mathbf{x}_4 = \mathbf{x}_{04}, \mathbf{x}_5 = \mathbf{x}_{10}, \dots$ e expressar o anterior naturalmente como $\{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_{24}\}$ - veja a figura ao lado; não é muito difícil de ver que no caso geral de um grid com $N \times N$ pontos obteríamos $\{\mathbf{x}_0, \dots, \mathbf{x}_{N^2-1}\}$). O leitor interessado em conhecer outras ordenações como a ordenação *red-black* pode procurar [WJ05, Cap. 3].²

Agora, utilizando a série de Taylor, podemos obter (assim como fizemos para o caso unidimensional) aproximações dos valores que u_{xx} e u_{yy} assumem num ponto $\mathbf{x}_{ij} = (x_i, y_j)$ (lembre-se que $x_i = ih$ e $y_j = jh$) do grid em

¹Assim como no caso unidimensional, observe que para cada i há $n + 1$ pontos no conjunto horizontal. Para evitar confusão pense em n como sendo o índice dos pontos x e $N = n + 1$ como sendo o número de pontos num conjunto horizontal.

²O modo de se ordenar os pontos no grid é importante uma vez que altera a forma da matriz de discretização A_h . Isso se refletirá fortemente nas propriedades suavizadoras dos métodos iterativos (em particular do método de Gauss-Seidel).

termos dos valores que u assume numa vizinhança desse $\mathbf{x}_{ij}$. Isso porque, pela série de Taylor sabemos que:

$$u(x_i + h, y_j) = u(x_i, y_j) + u_x(x_i, y_j)h + \frac{1}{2}u_{xx}(x_i, y_j)h^2 + O(h^3) \quad (2.3)$$

$$u(x_i - h, y_j) = u(x_i, y_j) - u_x(x_i, y_j)h + \frac{1}{2}u_{xx}(x_i, y_j)h^2 + O(h^3) \quad (2.4)$$

e

$$u(x_i, y_j + h) = u(x_i, y_j) + u_y(x_i, y_j)h + \frac{1}{2}u_{yy}(x_i, y_j)h^2 + O(h^3) \quad (2.5)$$

$$u(x_i, y_j - h) = u(x_i, y_j) - u_y(x_i, y_j)h + \frac{1}{2}u_{yy}(x_i, y_j)h^2 + O(h^3) \quad (2.6)$$

Manipulando cada um desses pares de equações exatamente como fizemos no caso unidimensional obtemos que:

$$u_{xx}(x_i, y_j) = \frac{u(x_i - h, y_j) - 2u(x_i, y_j) + u(x_i + h, y_j)}{h^2} + O(h^3) \quad (2.7)$$

$$u_{yy}(x_i, y_j) = \frac{u(x_i, y_j - h) - 2u(x_i, y_j) + u(x_i, y_j + h)}{h^2} + O(h^3) \quad (2.8)$$

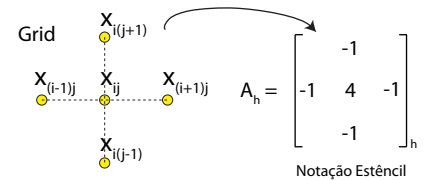
Denotando (por comodidade) $u(x_i, y_j) = u_{ij}$, então as expressões anteriores nos leva à **fórmula dos cinco pontos**, pois o membro esquerdo é aproximado utilizando-se cinco pontos do grid:

$$-u_{xx}(x_i, y_j) - u_{yy}(x_i, y_j) \approx \frac{1}{h^2}(-u_{(i-1)j} - u_{(i+1)j} + 4u_{ij} - u_{(i-1)j-1} - u_{(i+1)j+1}) \quad (2.9)$$

Combinando essa expressão com as exigências impostas pelo problema (2.1) original e utilizando a ordenação lexicográfica que estabelecemos, então devemos ter que (novamente, denote $f(x_i, y_j) = f_{ij}$):

$$\left\{ \begin{array}{l} \frac{1}{h^2}(-u_{01} - u_{10} + 4u_{11} - u_{21} - u_{12}) = f_{11} \\ \frac{1}{h^2}(-u_{02} - u_{11} + 4u_{12} - u_{22} - u_{13}) = f_{12} \\ \vdots \\ \frac{1}{h^2}(-u_{(i-1)j} - u_{(i+1)j} + 4u_{ij} - u_{(i-1)j-1} - u_{(i+1)j+1}) = f_{ij} \\ \vdots \end{array} \right. \quad (2.10)$$

Escrever esse sistema de equações lineares em sua forma matricial não será tão fácil como quando fizemos para o caso unidimensional. A matriz de discretização A_h que obteríamos seria uma matriz pentadiagonal esparsa, i.e., com cinco diagonais (ao redor da diagonal principal) possuindo algumas entradas não-nulas e sendo todas as demais entradas nulas (veja [Bie07, Cap. 1]). Dessa forma, costuma-se representar A_h através da *notação estêncil*, a qual consiste em dispor os coeficientes



de $u_{(i-1)j}, u_{i(j-1)}, u_{ij}, u_{(i+1)j}, u_{i(j+1)}$ da fórmula dos cinco pontos (2.9) com a mesma posição em que os pontos $\mathbf{x}_{(i-1)j}, \mathbf{x}_{i(j-1)}, \mathbf{x}_{ij}, \mathbf{x}_{(i+1)j}, \mathbf{x}_{i(j+1)}$ se encontram no grid (veja figura anterior). De posse dessa notação, podemos representar o sistema de equações (2.10) simplesmente como:

$$\frac{1}{h^2} \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}_h \mathbf{u}_h = \mathbf{f}_h \quad (2.11)$$

A_h

Assim como aconteceu no caso unidimensional, quanto mais pontos tomarmos no grid inicial melhor será a aproximação dada por Taylor. Mas novamente com uma grande matriz de discretização A_h vem a ineficiência dos métodos iterativos clássicos. Para estudarmos as propriedades suavizadoras desses métodos aplicados agora ao caso bidimensional, devemos percorrer os mesmos caminhos trilhados no caso unidimensional.

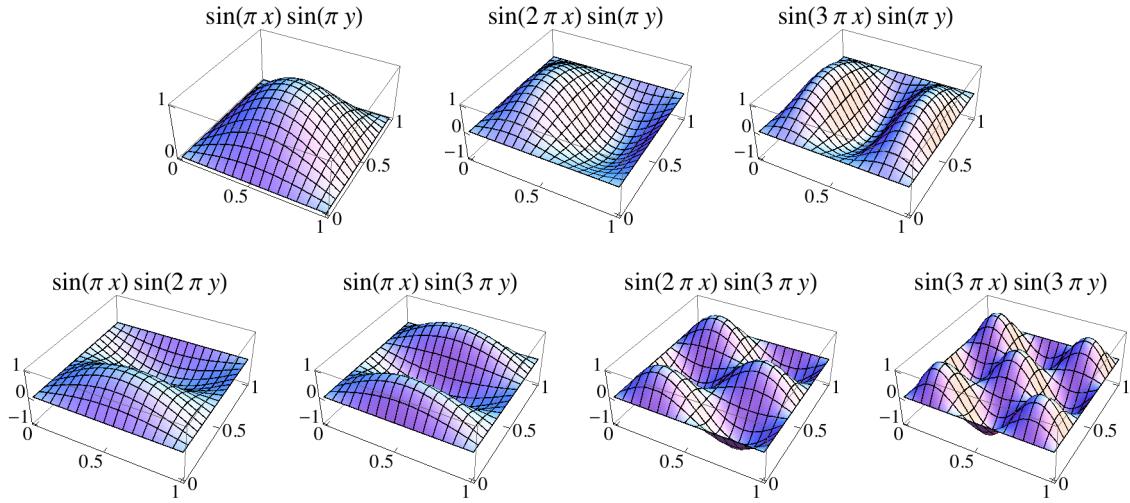
2.2 Autovalores e Autofunções do Laplaciano

Antes de qualquer coisa, procuremos pelas funções $u(x)$ que satisfazem o sistema

$$\begin{cases} -\Delta u = -u_{xx} - u_{yy} = \lambda u & x \in \Omega \\ u = 0 & x \in \partial\Omega \end{cases} \quad (2.12)$$

Esse é o problema de Dirichlet para a equação de Helmholtz bidimensional cujas soluções são as autofunções do operador laplaciano Δ . Motivados pelas autofunções do caso unidimensional, talvez não seja muito difícil de ver que a função $u(x, y) = \sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}y)$ satisfaz a primeira condição de (2.12). Isso porque $u_{xx} = -\lambda_1 \sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}y)$ e $u_{yy} = -\lambda_2 \sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}y)$, de modo que $u_{xx} + u_{yy} = -(\lambda_1 + \lambda_2) (\sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}y))$ e, portanto, $-u_{xx} - u_{yy} = \lambda u$ onde $\lambda = \lambda_1 + \lambda_2$. Observe que qualquer múltiplo de $\sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}y)$ continuaria satisfazendo a primeira condição, de modo que tenhamos em geral $u(x, y) = c \cdot \sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}y)$. Por fim, da condição de fronteira estabelecida em 2.12, concluimos que $c \cdot \sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}y) = 0$ e $c \cdot \sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}y) = 0$ [lembre-se de que estamos lidando com o quadrado unitário; no caso geral, em que lidamos com o quadrado $(0, L) \times (0, L)$, a condição de fronteira nos daria $\sin(\sqrt{\lambda_1}x) \cdot \sin(\sqrt{\lambda_2}L) = 0$ e $\sin(\sqrt{\lambda_1}L) \cdot \sin(\sqrt{\lambda_2}y) = 0$], de modo que $c \cdot \sin(\sqrt{\lambda_1}x) = 0$ e $c \cdot \sin(\sqrt{\lambda_2}y) = 0$. Como não queremos $c = 0$ (o que forneceria uma solução trivial), então devemos ter $\sqrt{\lambda_1} = r\pi$ e $\sqrt{\lambda_2} = s\pi$, o que nos fornece os autovalores $\lambda_{rs} = (r^2 + s^2)\pi^2$ cujas respectivas autofunções são $v_{rs} = \sin(r\pi x) \cdot \sin(s\pi y)$.

Remark 2.1. Como no caso unidimensional, observe que a frequência das autofunções v_{rs} aumenta à medida que r e s aumentam (veja figura abaixo). Para nossos atuais objetivos, diremos que a autofunção é de alta frequência quando $r > \frac{n}{2}$ ou $s > \frac{n}{2}$, ou seja, quando $\max(r, s) > \frac{n}{2}$.



2.3 Autovetores e Autovalores da Matriz de Discretização

Assim como no caso unidimensional, a Proposição abaixo revela que os autovetores da matriz de discretização A_h são justamente as autofunções do laplaciano Δ discretizadas, i.e., restritas aos pontos do grid.

Proposition 2.2. *A matriz de discretização A_h do Problema de Dirichlet para a Equação de Poisson Bidimensional no quadrado unitário (i.e., o problema (2.1)) possui autovalores $\lambda_{rs} = \frac{2}{h^2}(2 - \cos(r\pi h) - \cos(s\pi h))$ e autovetores $\mathbf{v}_{rs} = (\sin(r\pi x_1) \cdot \sin(s\pi y_1), \dots, \sin(r\pi x_{n-1}) \cdot \sin(s\pi y_{n-1}))$ correspondentes com $r, s = 1, 2, \dots, n-1$.*

Demonstração. Ao invés de utilizar a estratégia de verificação adotada no caso unidimensional, utilizaremos aqui um método semelhante ao método de separação de variáveis empregado na resolução de equações diferenciais ordinárias [embora o método de verificação também seja possível]. A vantagem de utilizarmos esse método é que ele independe da matriz A_h e, portanto, independe de como o grid está ordenado e do tamanho de A_h [esse método também poderia ter sido utilizado no caso unidimensional]. Estamos interessados nos vetores $\mathbf{u}_h$ tais que $A_h \mathbf{u}_h = \lambda \mathbf{u}_h$ e, portanto, cujas componentes satisfazem

$$\frac{1}{h^2}(-u_{(i-1)j} - u_{i(j-1)} + 4u_{ij} - u_{(i+1)j} - u_{i(j+1)}) = \lambda u_{ij} \quad (2.13)$$

Como acontece para equações diferenciais, começamos por algo similar à hipótese de D'Alembert, supondo que somos capazes de escrever as componentes u_{ij} de $\mathbf{u}_h$ como $u_{ij} = F(i)G(j)$, onde F, G são funções de uma variáveis inteira. Determinando $F(i)$ e $G(j)$ para i e j quaisquer, determinamos as componentes da solução $\mathbf{u}_h$ procurada e, portanto, a própria $\mathbf{u}_h$. Para tanto, observe que a nossa suposição permite reescrever 2.13 como

$$\frac{1}{h^2}(-F(i-1)G(j) - F(i)G(j-1) + 4F(i)G(j) - F(i+1)G(j) - F(i)G(j+1)) = \lambda F(i)G(j). \quad (2.14)$$

Dividindo a equação anterior por $F(i)G(j)$ e separando os termos que contêm F dos termos que contêm G , então obtemos (verifique!):

$$\frac{F(i-1) - 2F(i) + F(i+1)}{h^2 F(i)} + \frac{G(j-1) - 2G(j) + G(j+1)}{h^2 G(j)} = -\lambda. \quad (2.15)$$

Como a soma do membro à esquerda da expressão anterior é igual à constante $-\lambda$, então cada uma das parcelas dessa soma deve ser constante. Em particular, como h já é uma constante, então devemos ter

$$\frac{F(i-1) - 2F(i) + F(i+1)}{F(i)} = A \quad (2.16)$$

e

$$\frac{G(j-1) - 2G(j) + G(j+1)}{G(j)} = B. \quad (2.17)$$

onde A e B são constantes. Claro que se reescrevermos (2.15) em termos de A e B , então não é muito difícil de ver que essas constantes satisfazem:

$$\frac{1}{h^2}(A + B) = -\lambda \quad (2.18)$$

Agora, assim como no método de separação de variáveis para equações diferenciais ordinárias, observe que (2.16) e (2.17) podem ser escritas como

$$\begin{cases} F(i-1) - (2+A)F(i) + F(i+1) = 0 \\ G(j-1) - (2+B)G(j) + G(j+1) = 0 \end{cases} \quad (2.19)$$

Para simplificar as coisas no que está por vir é conveniente definirmos constantes α e β tais que $2\alpha = 2 + A$ e $2\beta = 2 + B$ (o porquê disso ficará claro mais adiante). Com isso, o sistema anterior se torna

$$\begin{cases} F(i-1) - 2\alpha F(i) + F(i+1) = 0 \\ G(j-1) - 2\beta G(j) + G(j+1) = 0 \end{cases} \quad (2.20)$$

Para resolver a primeira equação desse sistema e determinar $F(i)$, tentamos uma solução da forma $F(i) = z^i$ para obter:

$$z^{i-1} - 2\alpha z^i + z^{i+1} = 0 \quad (2.21)$$

e, portanto, dividindo a expressão anterior por z^{i-1} , para obter a equação quadrática

$$z^2 - 2\alpha z + 1 = 0 \quad (2.22)$$

que pode ser facilmente resolvida para z [determinando, assim, a função F]. Como se pode verificar, as duas raízes da equação anterior são $z_1 = \alpha - \sqrt{\alpha^2 - 1}$ e $z_2 = \alpha + \sqrt{\alpha^2 - 1}$ [definimos α da maneira anterior justamente

para simplificar a notação dessas raízes] e são tais que $z_1 + z_2 = -2\alpha$ e $z_1 z_2 = 1$. Dessa forma, a solução geral da primeira equação de (2.20) é da forma

$$F(i) = c_1 z_1^i + c_2 z_2^i. \quad (2.23)$$

Para determinar as constantes c_1, c_2 utilizamos as condições de fronteira do nosso problema, pois sabemos que $u_{ij} = 0$ para $i = 0, i = n$ ou $j = 0, j = n$ [já que assim, $\mathbf{x}_{ij} = (x_i, y_j)$ estará na fronteira do quadrado unitário]. Dessa forma, sabemos que $F(i) = 0$ quando $i = 0$ ou $i = n$. No primeiro caso, substituindo $i = 0$ em (2.23) obtemos que $c_1 = -c_2$ e, portanto, que $F(i) = c_1(z_1 - z_2)$. No segundo caso, substituindo $i = n$ em (2.23) obtemos que $z_1^n = z_2^n$ e, portanto, como $z_1 z_2 = 1$, que $z_1^{2n} = 1$. Assim, z_1 é uma $2n$ -ésima raiz complexa de 1, ou seja, $z_1 = e^{ir\pi/n}$ com $0 \leq r \leq 2n - 1$ e $\mathbf{i} = \sqrt{-1}$. Como $z_1 = 1/z_2$, então podemos restringir $0 \leq r \leq n - 1$ e, portanto, $F(i) = c_1(z_1 - z_2)$ produz todas as soluções não-triviais F da primeira equação em (2.20).

Assim,

$$\alpha = \frac{z_1 + z_2}{2} = \frac{e^{ir\pi/n} + e^{-ir\pi/n}}{2} = \cos(r\pi/n) \quad 0 \leq r \leq n - 1 \quad (2.24)$$

e, escolhendo $c = 1/2$,

$$F_r(i) = e^{ir\pi i/n} - e^{-ir\pi i/n} = \text{sen}(ir\pi/n). \quad (2.25)$$

Da mesma forma, temos que

$$\beta = \cos(s\pi/n) \quad (2.26)$$

e

$$G_s(j) = \text{sen}(js\pi/n). \quad (2.27)$$

Por fim, os autovalores são dados por

$$\lambda_{rs} = 2 \left[\frac{1}{h^2} (1 - \cos(r\pi/n)) + \frac{1}{h^2} (1 - \cos(s\pi/n)) \right] \quad (2.28)$$

e as coordenadas dos autovetores são dadas por

$$(\mathbf{v}_{rs})_{ij} = \text{sen}(r\pi i h) \cdot \text{sen}(s\pi j h) = \text{sen}(r\pi x_i) \cdot \text{sen}(s\pi y_j). \quad (2.29)$$

□

Remark 2.3. Poderíamos denotar $\mathbf{v}_{rs} = (\text{sen}(r\pi h) \cdot \text{sen}(s\pi h))_h = (v_{rs})_h$ para enfatizar o fato de que as coordenadas do rs -ésimo autovetor de A_h são exatamente os valores que a rs -ésima autofunção v_{rs} do laplaciano assume nos pontos do grid h .

2.4 Métodos Iterativos

Assim como aconteceu no caso unidimensional, observe que nossa matriz de discretização A_h $(n-1)^2 \times (n-1)^2$ possui $(n-1)^2$ autovalores não-nulos e, portanto, é diagonalizável. Dessa forma, A_h é invertível, já que pode ser escrita como um produto de matrizes invertíveis $A_h = PDP^{-1}$, o que implica a unicidade da solução de $A_h \mathbf{u}_h = \mathbf{f}_h$ e, portanto, a possibilidade de aplicarmos métodos iterativos para encontrá-la. Novamente é importante lembrar que nem sempre é tão fácil determinar os autovalores da matriz de discretização, o que nos leva ao problema de investigar as propriedades que A_h deve possuir para garantir sua invertibilidade. Consulte [Bie09, Cap. 2] para maiores informações.

2.4.1 Método de Jacobi

2.4.1.1 Matriz de Iteração

Exatamente como no caso unidimensional, começamos resolvendo a j -ésima equação do nosso sistema $A_h \mathbf{u}_h = \mathbf{f}_h$ como expresso em (2.10) para u_{ij} , obtendo:

$$u_{ij} = \frac{u_{(i-1)j} + u_{(i+1)j} + u_{i(j-1)} + u_{i(j+1)}}{4} + \frac{h^2}{4} f_{ij} \quad (2.30)$$

Em geral, o método de Jacobi corrige a equação anterior para um $\mathbf{u}_h^{(k-1)}$ e obtém um vetor $\mathbf{u}^{(k)}$ tal que (lembre-se da discussão feita para o Método de Jacobi no caso unidimensional):

$$[\mathbf{u}_h^{(k)}]_{ij} = \frac{[\mathbf{u}_h^{(k-1)}]_{(i-1)j} + [\mathbf{u}_h^{(k-1)}]_{(i+1)j} + [\mathbf{u}_h^{(k-1)}]_{i(j-1)} + [\mathbf{u}_h^{(k-1)}]_{i(j+1)}}{4} + \frac{h^2}{4} [\mathbf{f}_h]_{ij} \quad (2.31)$$

Em forma matricial o anterior pode ser expresso como

$$\mathbf{u}_h^{(k)} = R_J \mathbf{u}_h^{(k-1)} + \frac{h^2}{4} \mathbf{f}_h \quad (2.32)$$

onde $R_J = I - \frac{h^2}{4} A_h$ (verifique!).

2.4.1.2 Autovalores e Autovetores da Matriz de Iteração

A proposição a seguir revela que os autovetores da matriz de iteração R_J do método de Jacobi para nosso problema (2.10) são exatamente os autovetores da matriz de discretização A_h .

Proposition 2.4. *A matriz de iteração R_J do método de Jacobi para o problema (2.10) possui autovalores $\lambda_{rs} = \frac{1}{2}[\cos(r\pi h) + \cos(s\pi h)]$ e autovetores correspondentes $\mathbf{v}_{rs} = (\sin(r\pi x_1) \cdot \sin(s\pi y_1), \dots, \sin(r\pi x_{n-1}) \cdot \sin(s\pi y_{n-1}))$ com $r, s = 1, 2, \dots, n-1$, ou seja, os mesmo autovetores da matriz de discretização A_h .*

Demonstração. A idéia da demonstração é exatamente a mesma daquela apresentada no caso unidimensional (Proposição 1.3). Demonstre você mesmo! $\square$

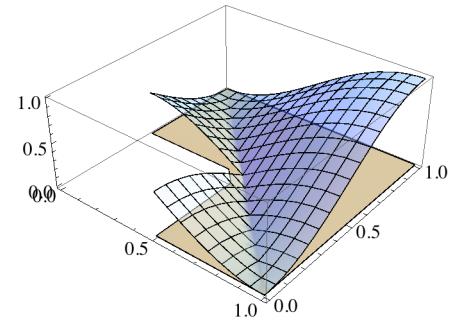
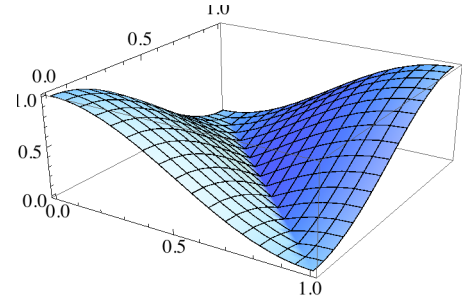
Agora, assim como fizemos no caso unidimensional, se escrevermos os autovalores de R_J como $\lambda(x_r, y_s) = \frac{1}{2}[\cos(\pi x_r) + \cos(\pi y_r)]$, onde $x_r = rh$ e $y_s = sh$ com $r, s = 1, \dots, n-1$, então o gráfico de $|\lambda(x, y)|$ (ao lado) nos revela que os menores autovalores em módulo de R_J são tais que:

(1) $x_r < 0,5$ e $y_s > 0,5$; ou (2) $x_r > 0,5$ e $y_s < 0,5$ (basta observar que no gráfico ao lado os menores autovalores ocorrem ao longo da reta $y = 1 - x$). Mas como $x_r = rh = \frac{r}{n}$ e $y_s = sh = \frac{s}{n}$, então o anterior significa que os menores autovalores em módulo acontecem

quando os índices r, s são tais que: (1) $r < \frac{n}{2}$ e $s > \frac{n}{2}$; ou (2) $r > \frac{n}{2}$ e $s < \frac{n}{2}$. Agora, uma vez que a matriz R_J é diagonalizável por se tratar de uma matriz $(n-1)^2 \times (n-1)^2$ com $(n-1)^2$ autovalores não-nulos distintos e é também a matriz de propagação do erro para o método de Jacobi (ou seja, $\mathbf{e}^{(k)} = R_J \mathbf{e}^{(0)}$), então escrevendo o erro inicial em termos da base $\{\mathbf{v}_{rs}\}$ de autovetores de R_J como $\mathbf{e}^{(0)} = \sum_{r,s=1}^{n-1} c_{rs} \mathbf{v}_{rs}$ segue-se que $\mathbf{e}^{(k)} = R_J^k \mathbf{e}^{(0)} = R_J^k \sum c_{rs} \mathbf{v}_{rs} = \sum c_{rs} (R_J^k \mathbf{v}_{rs}) = \sum c_{rs} (\lambda_{rs}^k \mathbf{v}_{rs})$. Assim, como os autovetores $\mathbf{v}_{rs}$ de R_J (e, portanto, de A_h) são exatamente as autofunções $v_{rs}(x) = \sin(r\pi x) \cdot \sin(s\pi y)$ discretizadas (lembre-se da Observação 2.3) e já que convencionamos dizer que uma autofunção é de alta frequência quando pelo menos um dos índices r, s é maior do que $\frac{n}{2}$ (lembre-se da Observação 2.1), então podemos dizer que o método de Jacobi é de fato um método suavizador, pois elimina algumas componentes de alta frequência rapidamente.

Se na discussão anterior denominarmos a região do plano cujos pontos $\mathbf{x} = (x, y)$ são tais que $0,5 < x < 1$ ou $0,5 < y < 1$ de *região de alta frequência* (já que $\lambda_{rs} = \lambda(x_r, y_s)$ está associada a uma autofunção de alta frequência quando $\mathbf{x}_{rs} = (x_r, y_s)$ está nessa região; observe que na figura ao lado essa região é representada em laranja e que omitimos a parte do gráfico que corresponde à região de baixa frequência, já que estamos interessados apenas em alta frequência), então podemos ver que o método de Jacobi é suavizador justamente porque seus menores autovalores em módulo recaem dentro dessa região. Contudo, observe

que o melhor dos casos seria garantirmos que o maior dos autovalores λ_{rs} dentro da região de alta frequência (que podemos denotar por $\lambda_{rs}^{(max)}$) fosse o menor possível, pois nesse caso garantiríamos que todas as componentes de alta frequência do erro inicial sejam removidas rapidamente (já que para todos os autovalores λ_{rs} dentro da região de alta frequência temos $0 \leq |\lambda_{rs}| \leq \lambda_{rs}^{(max)}$). No caso do método de Jacobi, observe que $\lambda_{rs}^{(max)}$ é próximo de 1, uma vez que o máximo da função $|\lambda(x, y)|$ restrita à região de alta frequência é exatamente 1. À medida que tomamos mais pontos no grid, mais próximo de 1 se torna $\lambda_{rs}^{(max)}$, de forma que no limite tenhamos $\lambda_{rs}^{(max)} = 1$. Esse valor limite para $\lambda_{rs}^{(max)}$ é o *fator de suavização* e fornece uma boa medida do poder de suavização do método. Veremos mais adiante que no método de Jacobi amortecido somos capazes de minimizar o fator de suavização do método de Jacobi às custas da sua velocidade de convergência (i.e., o poder de suavização aumenta, mas a velocidade de convergência diminui), obtendo um método suavizador otimizado.



2.4.2 Método de Gauss-Seidel

2.4.2.1 Matriz de Iteração

Observe que na definição (2.31) de uma iterada de Jacobi utilizamos as componentes $(i-1)j, (i+1)j, i(j-1), i(j+1)$ do vetor $\mathbf{u}_h^{(k-1)}$ na definição da componente ij do vetor $\mathbf{u}_h^{(k)}$. Contudo, como definimos as componentes de $\mathbf{u}_h^{(k)}$ uma a uma seguindo a ordem dos índices ij imposta pela ordenação lexicográfica, então talvez não seja muito difícil de ver que quando estivermos definindo a ij -ésima componente de $\mathbf{u}_h^{(k)}$ já teremos definido suas $(i-1)j$ -ésima e $i(j-1)$ -ésima componentes, uma vez que na ordenação lexicográfica temos que o índice $(i-1)j$ vem antes do índice $i(j-1)$ que, por sua vez, vem antes do índice ij (lembre-se da discussão feita na seção 2.1). Assim, talvez seja melhor utilizarmos essas componentes de $\mathbf{u}_h^{(k)}$ na definição (2.31) ao invés daquelas de $\mathbf{u}_h^{(k-1)}$, pois sabemos que as componentes de $\mathbf{u}_h^{(k)}$ estão mais próximas das componentes do vetor solução $\mathbf{u}_h$ do que as componentes de $\mathbf{u}_h^{(k-1)}$. No método de Gauss-Seidel fazemos justamente isso, definindo:

$$[\mathbf{u}_h^{(k)}]_{ij} = \frac{[\mathbf{u}_h^{(k)}]_{(i-1)j} + [\mathbf{u}_h^{(k-1)}]_{(i+1)j} + [\mathbf{u}_h^{(k)}]_{i(j-1)} + [\mathbf{u}_h^{(k-1)}]_{i(j+1)} + \frac{h^2}{4}[\mathbf{f}_h]_{ij}}{4} \quad (2.33)$$

Observe que diferentemente do caso unidimensional, o modo como ordenamos o grid afeta decisivamente a definição da iteração de Gauss-Seidel, o que terá importantes consequências em suas propriedades. Para encontrarmos a forma matricial do método de Gauss-Seidel, reescrevemos a expressão anterior como

$$-\frac{1}{4}[\mathbf{u}_h^{(k)}]_{(i-1)j} + [\mathbf{u}_h^{(k)}]_{ij} - \frac{1}{4}[\mathbf{u}_h^{(k)}]_{i(j-1)} = \frac{[\mathbf{u}_h^{(k-1)}]_{(i+1)j} + [\mathbf{u}_h^{(k-1)}]_{i(j+1)}}{4} + \frac{h^2}{4}[\mathbf{f}_h]_{ij} \quad (2.34)$$

e observamos que podemos reescrevê-la numa forma matricial como

$$[I - L]\mathbf{u}_h^{(k)} = U\mathbf{u}_h^{(k-1)} + \frac{h^2}{4}I\mathbf{f}_h \quad (2.35)$$

onde I é a identidade, L é uma matriz triangular inferior e U é uma matriz triangular superior. Para ver isso, escreva $\mathbf{u}_h^{(k)}$ como um vetor coluna cujas entradas são suas componentes na ordem imposta pela ordenação lexicográfica e observe que uma matriz B tal que $B\mathbf{u}_h^{(k)}$ fornece aquilo no membro esquerdo de (2.34) deve possuir entradas na diagonal iguais a 1 e algumas entradas abaixo da diagonal iguais a $-\frac{1}{4}$. Em seguida, basta decompor $B = I - L$, onde I é a identidade e L é uma matriz triangular inferior cujas entradas não nulas são exatamente $\frac{1}{4}$. Por fim, repita o raciocínio para o membro direito (se necessário, recorde o que foi feito para o caso unidimensional). Da expressão anterior chegamos em

$$\mathbf{u}_h^{(k)} = R_{GS}\mathbf{u}_h^{(k-1)} + \frac{h^2}{4}(I - L)^{-1}I\mathbf{f}_h \quad (2.36)$$

onde $R_{GS} = (I - L)^{-1}U$.

2.4.2.2 Autovalores e Autovetores da Matriz de Iteração

Assim como no caso de Jacobi (e no caso unidimensional), a proposição a seguir revela que os autovetores da matriz de iteração R_{GS} do método de Gauss-Seidel são exatamente os autovetores da matriz de discretização A_h .

Proposition 2.5. *A matriz de iteração R_{GS} do método de Gauss-Seidel para o problema (2.10) possui autovalores $\lambda_{rs} = \frac{1}{4}[\cos^2(r\pi h) + \cos^2(s\pi h)]$ e autovetores cujas componentes satisfazem $[\mathbf{v}_{rs}]_{ij} = \frac{1}{2}[\cos^2(r\pi h) + \cos^2(s\pi h)]^{(i+j)/2} \cdot \sin(r\pi x_i) \cdot \sin(s\pi y_j)$ com $r, s = 1, 2, \dots, n-1$, ou seja, NÃO coincidindo com os autovetores da matriz de discretização A_h .*

Demonstração. A demonstração segue o mesmo raciocínio do caso unidimensional (Proposição 1.5) e, portanto, é deixada como exercício. $\square$

Observe que a Proposição anterior garante que, para o nosso problema (2.10), o método de Gauss-Seidel é convergente, uma vez que os autovalores λ_{rs} de R_{GS} estão todos no intervalo aberto $(-1, 1)$ (basta observar que os argumentos das funções cosseno estão sempre entre 0 e π e que nunca assume esses valores; para tanto, escreva $r\pi h, s\pi h$ como $\frac{r\pi}{n}, \frac{s\pi}{n}$ onde $r, s = 1, 2, \dots, n-1$) e, portanto, o maior autovalor em módulo de R_{GS} é estritamente menor do que 1.

A análise do poder de suavização do método de Gauss-Seidel é um tanto mais complicada que aquela feita para o método de Jacobi, uma vez que os autovetores de R_{GS} não coincidem com os autovetores da matriz de discretização A_h . Dessa forma, quando escrevemos o erro inicial $\mathbf{e}^{(0)}$ em termos da base de autovetores $\{\mathbf{v}_{rs}\}$ como $\mathbf{e}^{(0)} = \sum_{r,s=1}^{n-1} c_{rs} \mathbf{v}_{rs}$, de modo que tenhamos $\mathbf{e}^{(k)} = R_{GS}^k \mathbf{e}^{(0)} = R_{GS}^k \sum c_{rs} \mathbf{v}_{rs} = \sum c_{rs} (R_{GS}^k \mathbf{v}_{rs}) = \sum c_{rs} (\lambda_{rs}^k \mathbf{v}_{rs})$, não somos capazes de dizer imediatamente quais são os efeitos do método sobre as componentes de alta frequência [as quais se relacionam aos autovetores de A_h] do erro inicial. Uma análise mais profunda, envolvendo a Teoria Local de Fourier, revela que o fator de suavização para o método de Gauss-Seidel é 0,5 e, portanto, que o método de Gauss-Seidel é um método muito mais suavizador do que o método de Jacobi.

2.4.3 Método de Jacobi Amortecido (ω -Jac)

Como no caso unidimensional, o método de Jacobi pode ser sobrerelaxado utilizando-se um fator de sobrerelaxamento $\omega < 1$ em

$$[\mathbf{u}_h^{(k)}]_j = [\mathbf{u}_h^{(k-1)}]_j + \omega \left([\widehat{\mathbf{u}}_h^{(k)}]_j - [\mathbf{u}_h^{(k-1)}]_j \right) \quad (2.37)$$

onde $\widehat{\mathbf{u}}_h^{(k)}$ é a k -ésima iteração do método de Jacobi como obtida em (2.31).

2.4.3.1 Matriz de Iteração

Utilizando (2.37) e o fato que $R_J = I - \frac{h^2}{4} A_h$, podemos deduzir da definição (2.32) que:

$$\begin{aligned}
\mathbf{u}_h^{(k)} &= \mathbf{u}_h^{(k-1)} + \omega(\widehat{\mathbf{u}}_h^{(k)} - \mathbf{u}_h^{(k-1)}) \\
&= \mathbf{u}_h^{(k-1)} + \omega(R_J \mathbf{u}_h^{(k-1)} + \frac{h^2}{4} \mathbf{f}_h - \mathbf{u}_h^{(k-1)}) \\
&= \mathbf{u}_h^{(k-1)} + \omega[(I - \frac{h^2}{4} A_h) \mathbf{u}_h^{(k-1)} + \frac{h^2}{4} \mathbf{f}_h - \mathbf{u}_h^{(k-1)}]
\end{aligned}$$

e, portanto, após as devidas simplificações, que:

$$\mathbf{u}_h^{(k)} = R_J(\omega) \cdot \mathbf{u}_h^{(k-1)} + \frac{h^2}{4} \mathbf{f}_h \quad (2.38)$$

onde $R_J(\omega) = I - \omega \frac{h^2}{4} A_h$. Observe que o método de Jacobi ocorre como um caso especial do Jacobi Amortecido quando $\omega = 1$.

2.4.3.2 Autovalores e Autovetores da Matriz de Iteração

Assim como no caso de Jacobi, a proposição a seguir revela que os autovetores da matriz de iteração $R_J(\omega)$ do método de Jacobi Amortecido são exatamente os autovetores da matriz de discretização A_h .

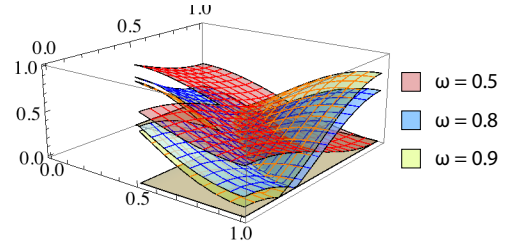
Proposition 2.6. *A matriz de iteração $R_J(\omega)$ do método de Jacobi Amortecido para o problema (2.10) possui autovalores $\lambda_{rs} = 1 - \frac{\omega}{2} [2 - \cos(r\pi h) - \cos(s\pi h)] = 1 - \omega [\sin^2(\frac{r\pi h}{2}) + \sin^2(\frac{s\pi h}{2})]$ e autovetores correspondentes $\mathbf{v}_{rs} = (\sin(r\pi x_1) \cdot \sin(s\pi y_1), \dots, \sin(r\pi x_{n-1}) \cdot \sin(s\pi y_{n-1}))$ com $r, s = 1, 2, \dots, n-1$, ou seja, os mesmo autovetores da matriz de discretização A_h .*

Demonstração. A demonstração é exatamente análoga ao do caso unidimensional (Proposição 1.6) e, portanto, é deixada como exercício. $\square$

2.4.3.3 Valor Ótimo de Suavização

Como de costume, permita-nos escrever os autovalores de $R_J(\omega)$ como $\lambda(\omega, x_r, y_s) = 1 - \omega [\sin^2(\frac{\pi}{2} x_r) + \sin^2(\frac{\pi}{2} y_s)]$ com $x_r = rh = \frac{r}{n}, y_s = sh = \frac{s}{n}$ e $r, s = 1, \dots, n-1$. Observe que na região de alta frequência (i.e., $0,5 < x < 1$ ou $0,5 < y < 1$) a função $f(x, y) = \sin^2(\frac{\pi}{2} x) + \sin^2(\frac{\pi}{2} y)$ é crescente, de modo que a função $\lambda(\omega, x, y) = 1 - \omega f(x, y)$ seja decrescente para um ω fixo (lembre-se que $0 < \omega < 1$). Isso significa que, na região de alta frequência e para um ω fixo, a função $\lambda(\omega, x, y)$ atinge seus máximos nos pontos $(\frac{1}{2}, 0)$ e $(0, \frac{1}{2})$ (onde temos $\lambda(\omega) = 1 - \frac{\omega}{2}$), enquanto que seu mínimo é atingido em $(1, 1)$ (onde temos $\lambda(\omega) = 1 - 2\omega$). Dessa forma, sabemos que o fator de suavização (i.e., o máximo da função $|\lambda(\omega, x, y)|$ na região de alta frequência e para um ω fixo) é igual ao $\max\{|1 - \frac{\omega}{2}|, |1 - 2\omega|\}$ e depende da nossa escolha de ω . Plotando o gráfico de $\max\{|1 - \frac{\omega}{2}|, |1 - 2\omega|\}$ em função de ω , vemos que para $\omega = \frac{4}{5} = 0.8$ (basta encontrar ω tal que $|1 - \frac{\omega}{2}| = |1 - 2\omega|$) atingimos o menor fator de suavização possível, de modo que o método de Jacobi Amortecido com esse fator de sobre-relaxamento constitui um suavizador ótimo.

Para exemplificar o anterior, observe no gráfico ao lado como o máximo da função $\lambda(0.8, x, y)$ é menor do que os máximos das funções $\lambda(0.9, x, y)$ e $\lambda(0.5, x, y)$ restritas à região de alta frequência. No caso da função $\lambda(0.5, x, y)$ observe que seu máximo ocorre nos pontos $(\frac{1}{2}, 0)$ e $(0, \frac{1}{2})$.



2.4.4 Método de Gauss-Seidel Amortecido (ω -GS ou SOR)

Ao invés de utilizarmos o método de Jacobi no sobrerelaxamento dado em (2.37) poderíamos utilizar o método de Gauss-Seidel, obtendo aquilo que é denominado método SOR (successive overrelaxation).

2.4.4.1 Matriz de Iteração

Da equação (2.37) com $\hat{\mathbf{u}}_h^{(k)}$ sendo o k -ésimo vetor obtido na iteração de Gauss-Seidel e utilizando a expressão em (2.36) obtemos que:

$$\begin{aligned}
 \mathbf{u}_h^{(k)} &= \mathbf{u}_h^{(k-1)} + \omega(\hat{\mathbf{u}}_h^{(k)} - \mathbf{u}_h^{(k-1)}) \\
 &= \mathbf{u}_h^{(k-1)} + \omega(L\mathbf{u}_h^{(k)} + U\mathbf{u}_h^{(k-1)} + \frac{h^2}{4}\mathbf{f}_h - \mathbf{u}_h^{(k-1)}) \\
 &= \mathbf{u}_h^{(k-1)} + \omega[L\mathbf{u}_h^{(k)} + (U - I)\mathbf{u}_h^{(k-1)} + \frac{h^2}{4}\mathbf{f}_h] \\
 &= [I + \omega(U - I)]\mathbf{u}_h^{(k-1)} + \omega L\mathbf{u}_h^{(k)} + \omega\frac{h^2}{2}\mathbf{f}_h \\
 (I - \omega L)\mathbf{u}_h^{(k)} &= [I + \omega(U - I)]\mathbf{u}_h^{(k-1)} + \frac{\omega h^2}{2}\mathbf{f}_h
 \end{aligned}$$

e, portanto, que

$$\mathbf{u}_h^{(k)} = R_{SOR}\mathbf{u}_h^{(k-1)} + \frac{\omega h^2}{2}(I - \omega L)^{-1}\mathbf{f}_h \quad (2.39)$$

onde $R_{SOR} = (I - \omega L)^{-1}[(1 - \omega)I + \omega U]$.

2.4.4.2 Autovalores e Autovetores da Matriz de Iteração

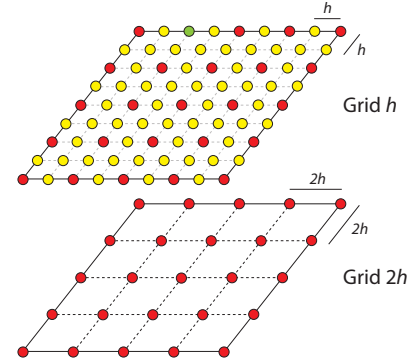
Proposição 2.7. A matriz de iteração R_{SOR} do método SOR para o problema (2.10) possui autovalores λ_i que satisfazem $\lambda_{rs}^{(J)} = (\frac{\lambda_{rs} + \omega - 1}{\lambda_{rs}^{1/2} \omega})$ onde $\lambda_{rs}^{(J)}$ é o rs -ésimo autovalor da matriz de iteração R_J do método de Jacobi e autovetores $\mathbf{v}_{rs}$ cujas componentes satisfazem $[\mathbf{v}_{rs}]_{ij} = \lambda_{rs}^{(i+j)/2} \sin(r\pi x_i) \cdot \sin(s\pi y_j)$ com $rs = 1, 2, \dots, n-1$, ou seja, NÃO coincidindo com os autovetores da matriz de discretização A_h .

Demonstração. A demonstração é exatamente análoga ao do caso unidimensional (Proposição 1.7) e, portanto, é deixada como exercício. $\square$

2.5 Bigrid

Assim como no caso unidimensional, precisamos utilizar um grid com muitos pontos para que a solução $\mathbf{u}_h$ do problema discretizado $A_h \mathbf{u}_h = \mathbf{f}_h$ apresentado em (2.10) seja uma boa aproximação da solução $u(x)$ do problema contínuo (2.1). Contudo, isso significa que teremos uma matriz de discretização A_h grande e, portanto, nossos métodos iterativos para encontrar $\mathbf{u}_h$ serão bem lentos, embora tenhamos visto que muitos deles possuam a propriedade de eliminar rapidamente as componentes de alta frequência do erro inicial. Essa propriedade suavizadora dos métodos iterativos mais uma vez será a primeira grande peça do Multigrid.

Como no caso unidimensional, a segunda peça importante do jogo é que componentes de baixa frequência num grid h são componentes de alta frequência num grid $2h$. Num grid h com 9×9 pontos (e, portanto, com $n = 8$), por exemplo, sabemos da Proposição 2.2 que A_h possui 64 autovetores $\mathbf{v}_{rs}$ com $r, s = 1, \dots, 8$ e que esses autovetores são as 64 primeiras (na ordem lexicográfica) autofunções do Laplaciano discretizadas, i.e., as componentes do rs -ésimo autovetor são os valores que a rs -ésima autofunção assume nos pontos da grid. Assim, podemos dizer que existem 64 modos de frequências no grid h , cada um dos quais associado a uma das autofunções $\sin(r\pi x) \cdot \sin(s\pi y)$ com $r, s = 1, \dots, 8$. Observe que desses 64 modos de frequências, apenas os 9 primeiros são visíveis no subgrid $2h$ (que possui 5×5 pontos, de modo que $n = 4$ e, portanto, estão presentes apenas as 9 primeiras autofunções). Agora, como convencionamos que alta frequência está associada a autofunções $\sin(r\pi x) \cdot \sin(s\pi y)$ tais que $r > \frac{n}{2}$ ou $s > \frac{n}{2}$, então temos que $\sin(\pi x) \cdot \sin(3\pi y), \sin(2\pi x) \cdot \sin(3\pi y), \sin(3\pi x) \cdot \sin(3\pi y)$ são todas de baixa frequência no grid h (pois ambos $r, s < 8/2$), mas são de alta frequência no grid $2h$ (pois ou $r > 4/2$ ou $s > 4/2$). Assim como no caso unidimensional, isso sugere uma estratégia interessante. Como um método iterativo suavizador elimina rapidamente as componentes de alta frequência do erro inicial $\mathbf{e}_h^{(0)}$ e deixa as componentes de baixa frequência praticamente inalteradas, então seria interessante que após algumas poucas iterações para eliminar as componentes de alta frequência transportássemos o problema para um subgrid com menos pontos, onde o método é capaz de eliminar rapidamente componentes que eram de baixa frequência no grid original!



Para tanto, novamente utilizamos a equação do resíduo $A_h \mathbf{e}_h = \mathbf{r}_h$ (lembre-se do caso unidimensional) para resolver $A_h \mathbf{u}_h = \mathbf{f}_h$ da seguinte forma:

$$\mathbf{u}_h^{(k)} \longrightarrow \mathbf{r}_h = \mathbf{f}_h - A_h \mathbf{u}_h^{(k)} \longrightarrow A_h \mathbf{e}_h = \mathbf{r}_h \longrightarrow \mathbf{u}_h = \mathbf{u}_h^{(k)} + \mathbf{e}_h \quad (2.40)$$

Esse processo em si não é significativo numericamente falando. Contudo, se pudermos aproximar A_h por um $\hat{A}_h$ mais "simples", então a solução de $\hat{A}_h \hat{\mathbf{e}}_h = \mathbf{r}_h$ será uma aproximação de $\mathbf{e}_h$, de modo que possamos utilizar o esquema (2.40) para obter um novo iterado $\mathbf{u}_h^{(k+1)}$ (e, portanto, um método iterativo) da seguinte forma:

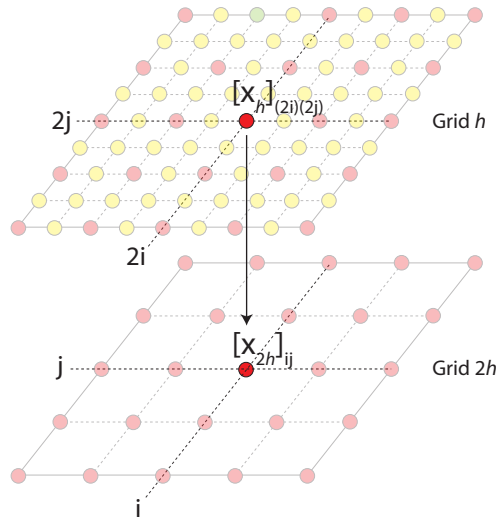
$$\mathbf{u}_h^{(k)} \longrightarrow \mathbf{r}_h = \mathbf{f}_h - A_h \mathbf{u}_h^{(k)} \longrightarrow \hat{A}_h \hat{\mathbf{e}}_h = \mathbf{r}_h \longrightarrow \mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)} + \hat{\mathbf{e}}_h \quad (2.41)$$

Assim como no caso unidimensional, uma das opções mais evidentes seria tentar aproximar $A_h \mathbf{e}_h = \mathbf{r}_h$ através

de $A_{2h}\mathbf{e}_{2h} = \mathbf{r}_{2h}$, o que nos leva a considerar os operadores de transferência intergrids. Para converter um vetor $\mathbf{x}_h$ do grid h a um vetor $\mathbf{x}_{2h}$ do subgrid $2h$ definimos um operador $I_{h \rightarrow 2h}$ (denominado o operador de *restrição*) tal que $\mathbf{x}_{2h} = I_{h \rightarrow 2h}\mathbf{x}_h$. Uma das opções mais simples seria tomar $I_{h \rightarrow 2h}$ para ser o operador de injeção que faz apenas

$$[\mathbf{x}_{2h}]_{ij} = [\mathbf{x}_h]_{(2i)(2j)} \quad (2.42)$$

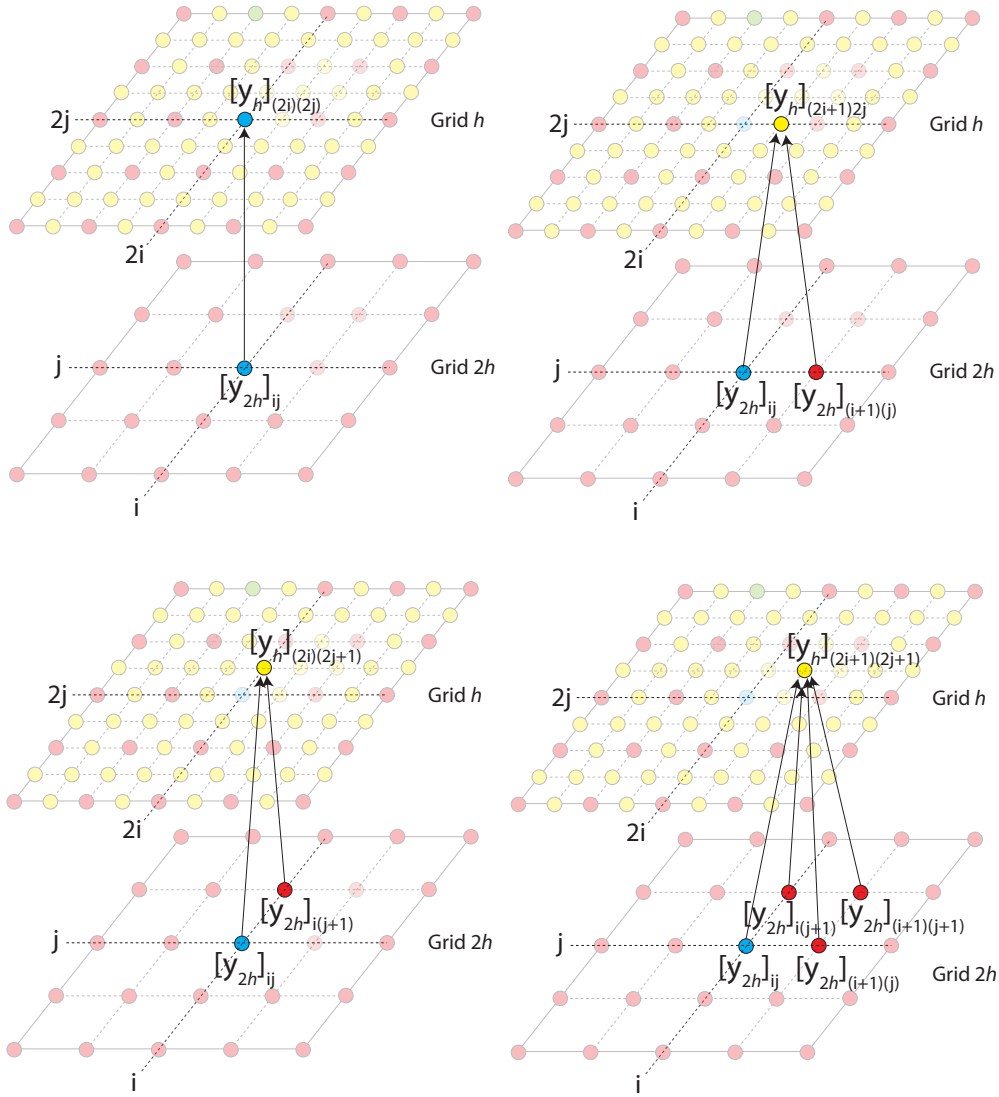
ou seja, que “constrói” $\mathbf{x}_{2h}$ simplesmente tomando as componentes de $\mathbf{x}_h$ associadas aos pontos do grid $2h$ (veja a figura abaixo).



Da mesma forma, para converter um vetor $\mathbf{y}_{2h}$ do subgrid $2h$ a um vetor $\mathbf{y}_h$ do grid h definimos um operador $I_{2h \rightarrow h}$ (denominado operador de *extensão* ou *prolongamento*) tal que $\mathbf{y}_h = I_{2h \rightarrow h}\mathbf{y}_{2h}$. Uma das opções mais utilizadas consiste em tomar $I_{2h \rightarrow h}$ para ser o operador de interpolação linear que faz

$$\begin{cases} [\mathbf{y}_h]_{(2i)(2j)} = [\mathbf{y}_{2h}]_{ij} & 1 \leq i, j \leq \frac{n}{2} - 1 \\ [\mathbf{y}_h]_{(2i+1)(2j)} = \frac{1}{2} ([\mathbf{y}_{2h}]_{ij} + [\mathbf{y}_{2h}]_{(i+1)j}) & 0 \leq i, j \leq \frac{n}{2} - 1 \\ [\mathbf{y}_h]_{(2i)(2j+1)} = \frac{1}{2} ([\mathbf{y}_{2h}]_{ij} + [\mathbf{y}_{2h}]_{i(j+1)}) & 0 \leq i, j \leq \frac{n}{2} - 1 \\ [\mathbf{y}_h]_{(2i+1)(2j+1)} = \frac{1}{4} ([\mathbf{y}_{2h}]_{ij} + [\mathbf{y}_{2h}]_{(i+1)j} + [\mathbf{y}_{2h}]_{i(j+1)} + [\mathbf{y}_{2h}]_{(i+1)(j+1)}) & 0 \leq i, j \leq \frac{n}{2} - 1 \end{cases} \quad (2.43)$$

(nas expressões anteriores desconsidere aquelas componentes que possuem índice 0, já que elas não existem nos vetores; só consideramos esses índices para simplificar as expressões gerais) ou seja, que “constrói” $\mathbf{y}_h$ tomando as componentes de $\mathbf{y}_{2h}$ para serem as componentes com i, j pares de $\mathbf{y}_h$ (as quais são associadas aos pontos do grid h que também estão no grid $2h$) e as médias de componentes adjacentes de $\mathbf{y}_{2h}$ para serem as componentes ímpares de $\mathbf{y}_h$ (as quais são associadas aos pontos do grid h que não estão presentes no grid $2h$). Para fixar as idéias, observe a figura abaixo, onde duas ou mais setas saindo do grid $2h$ representam a média aritmética das respectivas componentes de $\mathbf{y}_{2h}$.



Dessa forma, novamente como no caso unidimensional poderíamos tentar aproximar $A_h \mathbf{e}_h = \mathbf{r}_h$ da seguinte forma: [1] começamos por transferir o resíduo $\mathbf{r}_h$ para o grid $2h$ fazendo $\mathbf{r}_{2h} = I_{h \rightarrow 2h} \mathbf{r}_h$; [2] em seguida, resolvemos $A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h}$ exatamente para obter $\hat{\mathbf{e}}_{2h} = A_{2h}^{-1} \mathbf{r}_{2h}$; [3] por fim, transferimos esse $\hat{\mathbf{e}}_{2h}$ de volta para o grid h fazendo $\hat{\mathbf{e}}_h = I_{2h \rightarrow h} \hat{\mathbf{e}}_{2h}$. Se esse "interpolado" $\hat{\mathbf{e}}_h$ for uma boa aproximação da solução de $A_h \mathbf{e}_h = \mathbf{r}_h$, então diremos que essa equação é bem aproximada pela equação mais simples $A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h}$. O interessante é que, se o erro $\mathbf{e}_h$ associado a um iterado $\mathbf{u}_h^{(k)}$ é suave, i.e., não possui componentes de alta frequência, então o mesmo raciocínio do caso unidimensional mostra que a interpolação de $\hat{\mathbf{e}}_{2h}$ realmente fornece uma boa aproximação de $\mathbf{e}_h$!

Com essa abordagem, o processo iterativo expresso em (2.41) ficaria da seguinte forma:

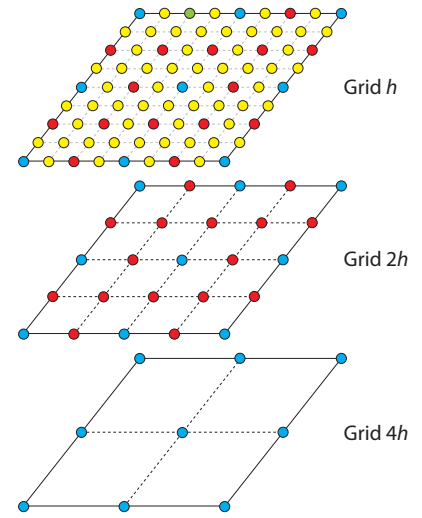
$$\begin{array}{ccccccc}
\text{Grid } h & \mathbf{u}_h^{(k)} & \longrightarrow & \mathbf{r}_h = \mathbf{f}_h - A_h \mathbf{u}_h^{(k)} & & \hat{\mathbf{e}}_h = I_{2h \rightarrow h} \hat{\mathbf{e}}_{2h} & \longrightarrow & \mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)} + \hat{\mathbf{e}}_h \\
& & & \downarrow & & \uparrow & & \\
\text{Subgrid } 2h & & & \mathbf{r}_{2h} = I_{h \rightarrow 2h} \mathbf{r}_h & \longrightarrow & A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h} & \longrightarrow & \hat{\mathbf{e}}_{2h} = A_{2h}^{-1} \mathbf{r}_{2h}
\end{array} \tag{2.44}$$

o qual mais uma vez chamaremos de uma **Iteração Bigrid** ou um **Esquema de Correção**. Com isso em mãos, obtemos aquilo que é denominado um **Ciclo Bigrid**, unidade básica de qualquer método Multigrid, o qual consiste das seguintes três etapas:

1. [Pre-suavização] Aplique um método iterativo suavizador m_1 vezes a $A_h \mathbf{u}_h = \mathbf{f}_h$ com chute inicial $\mathbf{u}_h^{(0)}$ para obter $\mathbf{u}_h^{(k_1)}$.
2. [Iteração Bigrid] Aplique a Iteração Bigrid (1.25) a $\mathbf{u}_h^{(k_1)}$ para obter $\mathbf{u}_h^{(k_1+1)}$.
3. [Pós-suavização] Aplique o método iterativo suavizador m_2 vezes a $A_h \mathbf{u}_h = \mathbf{f}_h$ com chute inicial $\mathbf{u}_h^{(k_1+1)}$ para obter $\mathbf{u}_h^{(k_2)}$.

2.6 Multigrid

Observe que na discussão anterior sobre o ciclo Bigrid resolvemos a equação do resíduo $A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h}$ exatamente no subgrid $2h$. Contudo, como tomamos um grid h com muitos pontos, então certamente o subgrid $2h$ ainda terá uma quantidade elevada de pontos, de modo que resolver $A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h}$ ainda seja tão difícil quanto resolver o problema original $A_h \mathbf{u}_h = \mathbf{f}_h$! Mas assim como no caso unidimensional observe que esses problemas são essencialmente iguais entre si! Poderíamos até mesmo utilizar a mesma notação do problema original e escrever $A_{2h} \mathbf{u}_{2h} = \mathbf{f}_{2h}$, onde $\mathbf{u}_{2h}$ é o erro $\mathbf{e}_{2h}$ e $\mathbf{f}_{2h}$ é o resíduo $\mathbf{r}_{2h}$. Isso sugere que apliquemos o método Bigrid novamente, mas agora a $A_{2h} \mathbf{e}_{2h} = \mathbf{r}_{2h}$ e rumo a um subgrid $4h$. Isso nos levará ao problema de resolver $A_{4h} \mathbf{e}_{4h} = \mathbf{r}_{4h}$, para o qual podemos aplicar novamente o ciclo Bigrid. O esquema geral já deve estar claro. Prossequimos com esse processo rumo ao subgrid com apenas 1 ponto interior, onde o problema se reduz a encontrar e em $ae = r$ e, portanto, que pode ser resolvido exatamente. Esse processo constitui aquilo que denominamos um v-ciclo Multigrid, o qual pode ser compactamente expresso nas seguintes etapas (relembre do caso unidimensional para maiores detalhes):



1. Aplique m_1 vezes um método iterativo suavizador a $A_h \mathbf{u}_h = \mathbf{f}_h$ no grid h com chute inicial $\mathbf{u}_h^{(0)}$ para obter $\mathbf{u}_h^{(k)}$.
2. Se o grid h só possui 1 ponto interior, então vá para a etapa número 5 tomando $\mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)}$; caso contrário continue.

3. Obtenha o resíduo $\mathbf{r}_h = \mathbf{f}_h - A_h \mathbf{u}_h^{(k)}$ e o transfira para o subgrid $2h$ fazendo $\mathbf{f}_{2h} = I_{h \rightarrow 2h} \mathbf{r}_h$.
4. Aplique o v-ciclo agora ao grid $2h$ (ou seja, comece novamente em 1 só que agora pra o grid $2h$) $\mu = 1$ vezes.
5. Transfira $\hat{\mathbf{u}}_{2h}$ para o grid h fazendo $\hat{\mathbf{u}}_h = I_{2h \rightarrow h} \hat{\mathbf{u}}_{2h}$ e obtenha $\mathbf{u}_h^{(k+1)} = \mathbf{u}_h^{(k)} + \hat{\mathbf{u}}_h$.
6. Aplique m_2 vezes o método iterativo suavizador a $A_h \mathbf{u}_h = \mathbf{f}_h$ no grid h com chute inicial $\mathbf{u}_h^{(0)} = \mathbf{u}_h^{(k+1)}$ para obter $\hat{\mathbf{u}}_h$.

Como no caso unidimensional, se utilizarmos um μ arbitrário diferente de 1 na etapa 4 do processo anterior obtemos uma família de ciclos Multigrid denominados μ -ciclos, dos quais os mais utilizados são o 1-ciclo (V-ciclo) e o 2-ciclo (W-ciclo).

2.6.1 Multigrid Completo (Full Multigrid - FMG)

Assim como no caso unidimensional, podemos aprimorar os μ -ciclos Multigrid utilizando um chute inicial $\mathbf{u}_h^{(0)}$ melhorado na iteração sobre $A_h \mathbf{u}_h = \mathbf{f}_h$ (etapa 1). Para tanto, repetimos o raciocínio empregado no caso unidimensional:

1. Transfira $\mathbf{f}_h$ para o subgrid $2h$ por meio de $\mathbf{f}_{2h} = I_{h \rightarrow 2h} \mathbf{f}_h$.
2. Transfira $\mathbf{f}_{2h}$ para o subgrid $4h$ por meio de $\mathbf{f}_{4h} = I_{2h \rightarrow 4h} \mathbf{f}_{2h}$.
3. $\vdots$
4. Resolva $A_{th} \mathbf{u}_{th} = \mathbf{f}_{th}$ no grid th que possui apenas 1 ponto interior para obter $\hat{\mathbf{u}}_{th}$.
5. $\vdots$
6. Transfira $\hat{\mathbf{u}}_{4h}$ para o grid $2h$ por meio de $\mathbf{u}_{2h}^{(0)} = I_{4h \rightarrow 2h} \hat{\mathbf{u}}_{4h}$.
7. Aplique o μ -ciclo um certo número v_0 de vezes a $A_{2h} \mathbf{u}_{2h} = \mathbf{f}_{2h}$ utilizando $\mathbf{u}_{2h}^{(0)}$ como chute inicial para obter $\hat{\mathbf{u}}_{2h}$.
8. Transfira $\hat{\mathbf{u}}_{2h}$ para o grid h por meio de $\mathbf{u}_h^{(0)} = I_{2h \rightarrow h} \hat{\mathbf{u}}_{2h}$.
9. Aplique o μ -ciclo um certo número v_0 de vezes a $A_{2h} \mathbf{u}_{2h} = \mathbf{f}_{2h}$ utilizando $\mathbf{u}_{2h}^{(0)}$ como chute inicial para obter $\hat{\mathbf{u}}_h$.

Esse processo de obter chutes iniciais aprimorados para os grids mais finos (= com mais pontos) utilizando os grids mais grosseiros (= com menos pontos) é denominado de *Iteração Aninhada* (*Nested Iteration*) e sua fusão com o μ -ciclo Multigrid (expresso nas etapas acima) fornece aquilo que denominamos um μ -ciclo Multigrid Completo ou FMG (Full Multigrid).

Referências Bibliográficas

- [BHM00] BRIGGS, Wiliam L. ; HENSON, Van E. ; MCCORMICK, Steve F.: *A Multigrid Tutorial*. 2a Ed. Siam, 2000
- [Bie07] BIEZUNER, Rodney J.: *Métodos Numéricos Para EDPs Elípticas*. 2007
- [Bie09] BIEZUNER, Rodney J.: *Álgebra Linear Numérica*. www.mat.ufmg.br/~rodney, 2009
- [TOS01] TROTTEBERG, Ulrich ; OOSTERLEE, Cornelis ; SCHÜLLER, Anton: *Multigrid*. Elsevier Academic Press, 2001
- [WJ05] WIENANDS, Roman ; JOPPICH, Wolfgang: *Practical Fourier Analysis for Multigrid Methods*. CRC Press, 2005