DEOVERSINO FRANÇA

FAST POISSON SOLVERS
UM ESTUDO COMPARATIVO

Dissertação apresentada ao Curso de Mestrado em Sistemas e Computação da Universidade Federal da Paraíba, em cumprimento a parte das exigências para obtenção do Grau de Mestre.

MÁRIO TOYOTARO HATTORI
- Orientador -

Campina Grande - Pb
Fevereiro - 1987

Às mulheres de minha vida

MIECO - minha esposa

KATIA MARA

e

GREICY MARA - minhas filhas

F814f Franca, Deoversino
Fast poisson solvers : um estudo comparativo / Deoversino Franca. - Campina Grande, 1987.
87 p.

Dissertacao (Mestrado em Sistemas e Computacao) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Algoritmos - Informatica 2. Analise de Foirier 3. Fast Poisson Solvers - Computacao 4. Dissertacao I. Hattori, Mario Toyotaro, Dr. II. Universidade Federal da Paraiba - Campina Grande (PB) III. Título

CDU 004.421.2(043)

FAST POISSON SOLVERS
UM ESTUDO COMPARATIVO

DEOVERSINO FRANÇA

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DO CURSO DE PÓS-GRADUAÇÃO EM SISTEMAS E COMPUTAÇÃO DA UNIVERSIDADE FEDERAL DA PARAÍBA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.).

Aprovada por:

MÁRIO TOYOTARO HATTORI - M.Sc.
- Presidente -

GREGÓRIO MARANGUAPE DA CUNHA - Dr.
- Examinador -

BRUNO CORREIA DA NÓBREGA QUEIROZ - M.Sc.
- Examinador -

CAMPINA GRANDE - Pb
FEVEREIRO - 1987

AGRADECIMENTOS

Ao professor Mário Toyotaro Hattori, que foi mais que "o orientador", pelas suas extremadas boa vontade, paciência e palavras de encorajamento.

A Universidade Federal de Mato Grosso do Sul, pelo apoio moral e financeiro recebido.

A Coordenação de Aperfeiçoamento de Pessoal de Nível Superior pela ajuda financeira.

Aos meus cunhados Firomi e Haruo pelo apoio e incentivo recebido.

Também muito mais que merecidos, a Mieco, minha esposa, e a Katia Mara e Greicy Mara, minhas filhas, que tanto me apoiaram e que por tantas vezes se viram privadas de minha presença por causa do desenvolvimento deste trabalho.

FRANÇA

RESUMO

Neste trabalho são estudadas quatro variantes do Fast Poisson Solvers. As quatro variantes, Método Básico FFT, Método da Redução Cíclica (Algoritmo de Buneman), FACR (Fourier Analysis/ Cyclic Redution)Método FACR(1,j) e Método FACR(1,i), são implementados e seus resultados comparados.

ABSTRACT

In this thesis are studied four variants of Fast Poisson Solvers. The four variants, Basic Method FFT, Cyclic Redution Method (Buneman's Algorithm), FACR (Fourier Analysis/Cyclic Reduction), Method FACR (1,j) and FACR Method (1,i), are implemented and theirs results compared.

SUMÁRIO

PÁGINA

CAPÍTULO III

CAPÍTULO IV

CAPÍTULO V

PÁGINA

CAPÍTULO I

INTRODUÇÃO

1.1 - Apresentação

A equação diferencial parcial de Poisson modela fenômeno de distribuição de potencial (elétrico, gravitacional) e sua solução tem interesse prático em ciências aplicadas e engenharia.

Uma solução analítica da equação de Poisson é raras vezes possível de ser obtida, porque a geometria do contorno e as condições impostas nesse contorno influenciam fortemente a solução. Para superar essa dificuldade, apela-se para a solução numérica. A solução numérica por sua vez não é trivial, porque aplicando a discretização temos sistemas lineares com muitas equações, principalmente se for em três dimensões. Por exemplo, em três dimensões, um cubo discretizado com 10 pontos em cada eixo coordenado conduzirá a 1000 equações lineares.

No início da década de 1970 os pesquisadores descobriram métodos de melhorar a eficiência da solução numérica da equação de Poisson, baseada no método das diferenças finitas. O método basicamente procura resolver o sistema de equações lineares proveniente da discretização do problema pelo método das diferenças finitas de modo eficiente, envolvendo uma transformada de Fourier (transformada rápida de Fourier devida a Cooley e Tuckey, conforme Hockney [12]). Esse método foi batizado de Fast Poisson Solver (FPS). Hoje existe uma família de Fast Poisson Solvers, constituída

a) pelo Método básico FFT
b) pelo Método da redução cíclica
c) pelos Métodos FACR (Fourier Analysis/Cyclic Redution)

Método FACR (1,j)
Método FACR (1,i)

O FPS foi originalmente desenvolvido para regiões retangulares, posteriormente foi estendido para regiões planas mais complicadas e finalmente para regiões tridimensionais.

1.2 - Importância do Trabalho

No Campus II da UFPb existem diversos computadores (com vários Sistemas Operacionais) que entre outras tarefas são bastante utilizados para a solução de problemas científicos.

Em um destes computadores existe a Biblioteca Transportável de Algoritmos Numérica (BITAN) com cerca de 200 rotinas que são implementações de algoritmos matemáticos.

É interesse da UFPb difundir a utilização da BITAN, não somente nela mas também proporcionar a outras instituições a sua utilização.

Tendo em vista a necessidade de uma rotina para resolver a equação de Poisson, em eletricidade e magnetismo, e para auxiliar a difusão da BITAN, direcionamos o estudo para quatro variantes do Fast Poisson Solver. Após o término deste trabalho espera-se que tenhamos o melhor método do Fast Poisson Solvers transformado em uma subrotina da BITAN para futuras utilizações.

1.3 - **Visão Global do Trabalho**

Este trabalho contém um estudo da equação linear de Poisson em regiões retangulares, utilizando a condição de contorno de Dirichlet.

Inicialmente, no segundo capítulo damos um resumo matemático: classificação das equações; existência e unicidade da solução; e princípio do máximo.

Uma revisão de diferenças finitas, convergência, erro de discretização da equação de Poisson é feita no terceiro capítulo.

No quarto capítulo apresentamos, a partir da discretização, quatro variantes do Fast Poisson Solvers:

a) Método Básico FFT
b) Método da Redução Cíclica (Algoritmo de Buneman)
c) Método FACR (1,j) de Hockney
d) Método FACR (1,i) de Hockney.

Os algoritmos das quatro variantes são apresentados com a condição de contorno de Dirichlet. Damos indicações para a condição periódica nos métodos Básico FFT e FACR. O algoritmo de Buneman baseia-se numa região quadrada

onde o valor do lado é 1.

No capítulo cinco, concluimos fazendo uma análise dos resultados obtidos nos testes, relacionando as dificuldades encontradas na implementação das quatro variantes do Fast Poisson Solvers, identificando pontos que podem ser estudados e apresentadas sugestões para estudos posteriores.

No apêndice I damos uma pequena visão do algoritmo FFT.

CAPÍTULO II

RESUMO DE MATEMÁTICA DA EQUAÇÃO DE POISSON

2.1 - Introdução

Para chegarmos a resolução numérica da equação de Poisson, necessitamos de alguns conhecimentos de sua matemática. Em virtude disto daremos a seguir um resumo da matemática usada no trabalho.

Inicialmente classificamos as equações diferenciais e mostramos algumas aplicações da Equação de Poisson.

Em seguida o teorema da existência e unicidade da solução é apresentado.

Por último apresentamos o princípio do máximo, baseado em Figueiredo [10].

2.2 - Classificação

A formulação matemática de muitos problemas na ciência envolvendo razão da mudança relativa de duas ou mais variáveis independentes, usualmente representando tempo, comprimento ou ângulo, conduz as equações diferenciais parciais ou a um sistema de equações.

Uma equação diferencial de segunda ordem a duas variáveis pode ser escrita na forma:

$$a\frac{\partial^2 u}{\partial x^2} + b\frac{\partial^2 u}{\partial x^2 y} + c\frac{\partial^2 u}{\partial y^2} + d\frac{\partial u}{\partial y} + e\frac{\partial u}{\partial y} + fu + g = 0$$

onde a, b, c, d, e, f e g podem ser constantes ou funções de x e y.

Se f for igual a zero e a, b, c, d, e, g não dependerem de u ou $\frac{\partial u}{\partial x}$ ou $\frac{\partial u}{\partial y}$ a equação diferencial é linear e se f for diferente de zero a equação diferencial é dita não linear.

Podemos classificá-la em:

Equação elíptica se $b^2 - 4ac < 0$

Equação parabólica se $b^2 - 4ac = 0$

Equação hiperbólica se $b^2 - 4ac > 0$

As equações elípticas mais conhecidas são:

a) $\frac{\partial u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + g = 0$ equação de Poisson

b) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$ equação de Laplace

c) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + fu = 0$ equação de Helmholtz

Para a equação de Helmholtz, se f for constante, tem-se um problema de valor de contorno e se f for um parâmetro a ser determinado tem-se um problema de auto valor.

A equação de Poisson tem aplicação no estudo da teoria de torsão de St Venant; movimento dos fluidos viscosos; eletricidade e magnetismo (Veja [16]).

2.3 - Existência e Unicidade da Solução

Teorema: (John F., [13])

Dada a equação de Poisson:

$$\nabla^2 u = F(x,y,u), \text{ em } \Omega \qquad (1)$$

$u = f$, no contorno $\partial\Omega$

A solução existe e é única, contanto que as condições abaixo sejam satisfeitas

I) $F(x,y,u) \in C^1$ para $(x,y) \in \Omega \cup \partial\Omega$ para todo u.

II) $|F(x,y,u)| \leq \dfrac{4(A - \max\limits_{\Omega} |f(S)|)}{R^2}$, para $(x,y) \in \Omega \cup$

$\cup \partial\Omega$ e $|u| \leq A$

onde R representa o diâmetro da $\Omega \cup \partial\Omega$ e A uma constante

III) $|\dfrac{\partial}{\partial u} F(x,y,u)| \leq \dfrac{4q}{R^2}$, para $(x,y) \in \Omega$ e $|u| \leq A$ onde q é a razão de uma série geométrica

Prova: A solução U_{n+1} para o problema do valor de contorno da equação de Poisson, $\nabla^2 u_{n+1} = F(x,y,u_n)$, $u_{n+1} = f$ no contorno $\partial\Omega$, pode ser escrita na forma $u_{n+1} = u'_{n+1} + u''_{n+1}$

onde

$$u'_{n+1} = \int_{\Omega} F(\zeta,\eta,u_n(\zeta,\eta)) \log r \, d\zeta d\eta \qquad (2)$$

e onde u''_{n+1} é uma função potencial satisfazendo

$\nabla^2 u''_{n+1} = 0$, em Ω

$u''_{n+1} = f - u_{n+1}$, no contorno $\partial\Omega$

Agora já que $u_n \to u$ uniformemente em $\Omega \cup \partial\Omega$ segue por (2) que

$$u'_{n+1} = \int_\Omega F(\zeta,\eta,u_n) \log r d\zeta d\eta \qquad (3)$$

$$\to \int_\Omega F(\zeta,\eta,u) \log r d\zeta d\eta = u'$$

uniformemente em $\Omega \cup \partial\Omega$. Pelo teorema 7 de [13] segue-se que a função potencial $u''_{n+1} \to u''$ onde

$\nabla^2 u'' = 0$, em Ω

$u'' = f-u'$, no contorno $\partial\Omega$ logo, como $n \to \infty$

$u_{n+1} = u'_{n+1} + u''_{n+1} \to u' + u'' \equiv 0$

Mas já foi estabelecido que $u_{n+1} \to u$.

Por isso $u = \phi = u' + u''$. Então isto estabelece que a função limite $u = u' + u''$, onde

$$u' = \int_\Omega F(\zeta,\eta,u) \log r d\zeta d\eta \qquad (4)$$

e u'' satisfazem o problema do valor de contorno

$\nabla^2 u'' = 0$, em Ω

$u'' = f - u'$, no contorno $\partial\Omega$

Mas agora é claro que se $F(x,y,u)$ satisfaz a hipótese do teorema 1 de [13] então u' satisfará $\nabla^2 u' = F(x,y,u)$ e u satisfará $\nabla^2 u = F(x,y,u)$ com o contorno dado $u = f$. Assim só resta mostrar que

$F(x,y,u) \in C^0$ em $\Omega \cup \partial\Omega$ e $F(x,y,u) \in C^1$ em Ω.

Mas isto é facilmente verificado pela condição I, lema 3, de [13] e o fato de que (4) pode ser diferenciado sob a integral.

Para provar a unicidade nós supomos duas soluções u e v. Isto é

$$\nabla^2 u = F(x,y,u), \text{ em } \Omega$$

$$\nabla^2 v = F(x,y,u), \text{ em } \Omega$$

$$u = v = f, \text{ no contorno } \partial\Omega.$$

Então a função $W = u-v$ satisfaz

$$\nabla^2 W = F(x,y,u) - F(x,y,v) = \frac{\partial}{\partial u} F(x,y,u)W \text{ em } \Omega$$

$$W = 0, \text{ no contorno } \partial\Omega$$

Por $|u| \leq \max_{\Omega} |f(S)| + \frac{1}{4} R^2 \max_{\Omega \cup \partial\Omega} |W|$

e a condição III segue que

$$\max|W| \leq \frac{1}{4R^2} \max\left|\frac{\partial}{\partial u} F\right| \max(W) \leq q \max(W),$$

onde $q < 1$. Mas isto é impossível, a não ser que

$W \equiv 0$ ou $u = v$.

Isso prova a unicidade.

A solução da equação de Poisson depende de F, que pode ser visto como uma excitação, da geometria do contorno e das condições de contorno (na nossa formulação, de f). Quando a solução for única e sua dependência das condições de contorno for contínua, teremos um problema bem posto. Somente problemas bem postos podem ser resolvidos com sucesso por métodos numéricos [15].

2.4 - **Princípio do Máximo** (Figueiredo [16])

Sejam Ω uma região limitada do plano e $u : \bar{\Omega} \to \mathbf{R}$ uma função contínua em $\bar{\Omega}$ ($\bar{\Omega}$ a aderência de Ω) e harmônica em Ω. Então o máximo de u é atingido em $\partial\Omega$.

Demonstração (Privalov) sendo Ω limitada, segue-se que $\bar{\Omega}$ e $\partial\Omega$ são conjuntos compactos (isto é, fechados e limitados) do plano. Utilizaremos um teorema da Análise segundo o qual a toda função real contínua $g : K \to \mathbf{R}$, em um conjunto compacto K, tem um valor máximo u, isto é, existe $(\bar{x},\bar{y}) \in K$, tal que $f(x,y) \leq f(\bar{x},\bar{y}) = u$, para todo (x,y) em K.

Sejam

$$M = \max_{\bar{\Omega}} u(x,y) \quad \text{e} \quad m = \max_{\partial\Omega} u(x,y)$$

e suponhamos , por contradição, que $m < M$. Então u assume

seu máximo em um ponto (x_0,y_0) que deve estar em Ω e não em $\partial\Omega$. Definamos a função

$$v(x,y) = u(x,y) + \frac{M-m}{2d^2}\left[(x-x_0)^2 + (y-y_0)^2\right] \tag{5}$$

onde d é o diâmetro da região Ω, isto é, d é o supremo das distâncias entre pares de pontos de Ω. Agora, observando que

$$v(x_0,y_0) = u(x_0,y_0) = M$$

e

$$v(x,y) \leq m + \frac{M-m}{2d^2}\, d^2 = \frac{M+m}{2} < M, \quad (x,y) \in \partial\Omega$$

concluimos que a função v assume seu máximo em um ponto (x_1,y_1) de Ω, e não em $\partial\Omega$. Portanto neste ponto $v_{xx}(x_1,y_1) \leq 0$ e $v_{yy}(x_1,y_1) \leq 0$, o que nos dá

$$\nabla^2 v(x_1,y_1) \leq 0 \tag{6}$$

Por outro lado, temos a partir de (1), que

$$\nabla^2 v = \frac{M-m}{d^2} \cdot 2 > 0 \tag{7}$$

para todos os pontos de Ω. As desigualdades (6) e (7) são contraditórias, o que prova o princípio. Para ver que o mí-

nimo também é atingido, basta trabalhar com -u.

Os máximos ou mínimos são sempre assumidos no contorno. Então toda solução não constante u da equação diferencial assume seu mínimo, se ela for negativa (ou seu máximo, se ela for positiva) no contorno $\partial\Omega$ e não no interior da região Ω.

Os máximos ou mínimos são assumidos no interior se e somente se u for constante.

Baseado nesse fato podem-se construir métodos numéricos que só utilizam as informações do contorno, por exemplo, um método que minimize o erro entre a aproximação e a solução no contorno.

CAPÍTULO III

DIFERENÇAS FINITAS

3.1 - Introdução

Neste capítulo apresentamos a idéia das diferenças finitas; o tratamento descritivo da convergência; erro de discretização; e a discretização da equação de Poisson.

3.2 - Diferenças Finitas

Quando uma função u e suas derivadas são funções contínuas, finitas de x, então pelo Teorema de Taylor

$$u(x+h) = u(x)+hu'(x)+\frac{1}{2}h^2u''(x)+\frac{1}{6}h^3u'''(x)+\ldots \qquad (1)$$

e

$$u(x-h) = u(x)-hu'(x)+\frac{1}{2}h^2u''(x)-\frac{1}{6}h^3u'''(x)+\ldots \qquad (2)$$

Adicionando estas duas expressões temos:

$$u(x+h) + u(x-h) = 2u(x) + h^2u''(x) + 0(h^4) \qquad (3)$$

onde $0(h^4)$ denota o termo contendo a quarta ou maior potência de h.

Assumindo que $0(h^4)$ seja desprezível em comparação com h^3, seque que

$$u''(x) = \frac{d^2u}{dx^2} \simeq \frac{1}{h^2}\{u(x+h) - 2u(x) + u(x-h)\}, \qquad (4)$$

com um erro cometido no 2º membro da ordem h^2.

Subtraindo da equação (1) a equação (2) e abandonando o termo de ordem h^3,

$$u'(x) = \frac{du}{dx} \simeq \frac{1}{2h}\{u(x+h) - u(x-h)\} \qquad (5)$$

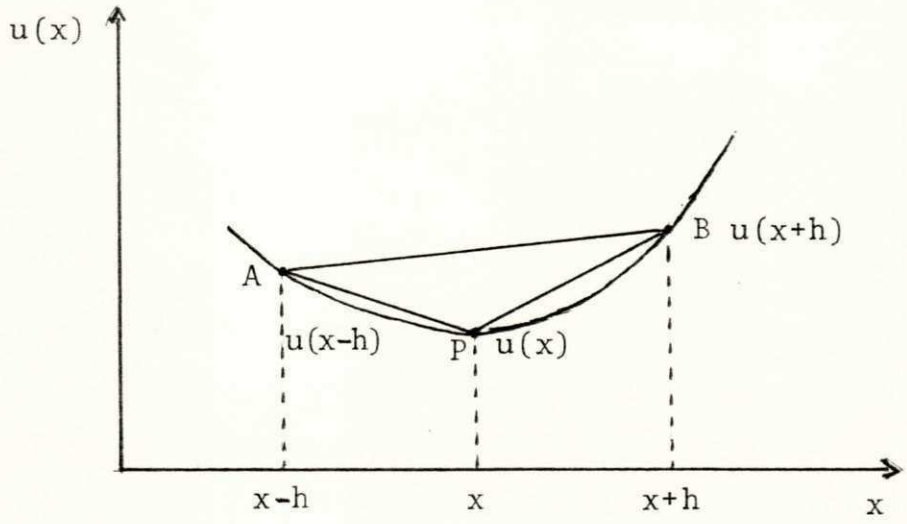
A equação (5) aproxima evidentemente a inclinação da tangente em P pela inclinação da corda AB, e é chamada uma aproximação de diferença central. Podemos também aproximar a inclinação da tangente em P por ambas as inclinações; da corda PB, resultando a fórmula da diferença progressiva

$$u'(x) \simeq \frac{1}{h}\{u(x+h) - u(x)\} \qquad (6)$$

ou a inclinação da corda AP resultando a fórmula da dife-

rença regressiva

$$u'(x) \simeq \frac{1}{h}\{u(x) - u(x-h)\}. \qquad (7)$$



Ambas (6) e (7) podem ser escritas imediatamente das equações (1) e (2), respectivamente, assumindo que as potências dois ou maiores de h são desprezíveis. Isto mostra que os erros cometidos nas fórmulas de diferença progressiva e regressiva são ambos 0(h).

As diferenças finitas são amplamente usadas para a representação dos polinômios de interpolação e na aplicação destes polinômios na solução de equações diferenciais e na quadratura numérica.

Uso importante de tais diferenças se faz para

obter aproximações das derivadas de certas funções.

A seguir apresentamos como as derivadas aparecem ao serem aproximadas pelas diferenças.

a) $u'_i \cong \frac{du_i}{dx} = \frac{u_{i+1} - u_i}{h}$

b) $u''_i \cong \frac{d^2u_i}{dx^2} = \frac{u_{i+2} - 2u_{i+1} + u_i}{h^2}$

c) $u'''_i \cong \frac{d^3u_i}{dx^3} = \frac{u_{i+3} - 3u_{i+2} + 3u_{i+1} - u_i}{h^3}$

d) $\frac{\partial u}{\partial x} \simeq \frac{u(x+h,y) - u(x,y)}{h}$

e) $\frac{\partial^2 u}{\partial x^2} \simeq \frac{u(x+2h,y) - 2u(x+h,y) + u(x,y)}{h^2}$

f) $\frac{\partial^3 u}{\partial u^3} \simeq \frac{u(x+3h,y) - 3u(x+2h) + 3u(x+h) - u(x,y)}{h^3}$

A aproximação da diferença central é relevante para a equação de Poisson.

Na malha de discretização tomamos sempre três

pontos ao longo de um eixo, sendo que dois deles são simétricos em relação ao terceiro.

3.3 - Consistência

Seja o problema

$$\nabla^2 u = F(x,y) \qquad F \in \Omega \tag{8}$$
$$u = f \qquad f \in \partial\Omega$$

A definição seguinte contém as condições gerais do método das diferenças para resolver o problema dado acima.

Definição: (segundo [14]) A sequência

$$D = \{(M_j, F_j, R_j) \quad j = 1,\ldots,\infty\}$$

é chamada método da diferença para resolver (8) se as três condições seguintes forem satisfeitas.

a) Os M_j são malhas em Ω com o tamanho das malhas $h_j = |M_j|$ tendendo para zero.

b) Os F_j são mapeamentos contínuos $C^0(\partial\Omega,\mathbf{R})xC^0(M_j,\mathbb{R}) \rightarrow$

$\rightarrow C^0(M_j,\mathbf{R})$.

Para cada $f \in C^0(\partial\Omega,\mathbb{R})$ fixo, todos os $F_j(f_i)$ são mapea-

mentos continuamente diferenciáveis de $C^0(M_j,\mathbb{R})$ para $C^0(M_j,\mathbb{R})$.

c) Os R_j são mapeamentos lineares

$$C^0(M_j,\mathbb{R}) \to C^0(M_j,\mathbb{R})$$

O método D é chamado consistente se a condição seguinte for satisfeita.

d) Exite um $m \geq 2$ tal que para todo $u \in C^m(\bar{\Omega},\mathbb{R})$,

$$\lim_{j\to\infty} \| F_j(f,r_j(u)) - R_j(r_j(q)) \|_\infty = 0$$

Aqui $r_j = r_{Mj}$, $f = r_{\partial\Omega}(u)$, e $q(x,y) = \nabla^2 u$

para todo $(x,y) \in \Omega$

3.4 - Estimativa de Erro

Tendo resolvido as equações a diferenças finita com um intervalo particular, gostaríamos de saber quão boa é a nossa resposta, isto é, quanto ela difere em cada ponto da malha da solução verdadeira da equação diferencial e como o erro deveria decrescer quando tomamos intervalos menores.

Uma discussão de alguns resultados relevantes nesta área é dada por Hubbard, descrita em Fox [11] e que procuraremos resumir.

Considere a equação de Poisson

$$\nabla^2 u = F(x,y) \qquad (a)$$

$u = f(x,y)$ no contorno $\partial\Omega$ e assuma que a solução $u(x,y)$ tenha a quarta derivada contínua no interior da região. Isto será verificado por uma análise a priori. Se $u(x,y)$ tiver um comportamento suave no contorno, então a fórmula de cinco pontos nos pontos internos regulares (nos quais todos os vizinhos estão dentro da região) tem erro de truncamento local $0(h^2)$.

Em situações menos suaves Hubbard discute o caso onde $F(x,y)$ na equação (9) é tal que, próximo a um ponto singular no interior da região, a solução $u(x,y)$ é finita mas tem derivadas que se comportam de acordo com

$$|\partial' u(x,y)| = 0(r^{\alpha-j+1}), \quad j = 1,2,3,4 \qquad (10)$$

onde r é a distância do ponto singular.

Ele então descobriu uma estimativa efetiva

$$e(x,y) = 0(h^{\alpha}+h^2) \qquad (11)$$

Se o ponto singular pertence ao contorno então

(10) é válido.

A estimativa de erro descoberta é

$$e(x,y) = 0(h^{\alpha}+h^{i+2}) \qquad (12)$$

Para isto ele deriva os resultados para o caso acima, no qual o contorno é composto de fragmentos de arcos analíticos, obtendo a fórmula

$$|\partial^{j}u(x,y)| = 0(r^{\alpha-j}), \qquad j = 0,1,2,\ldots \qquad (13)$$

O erro em ambas as formas é igual a (11).

Resultados posteriores deste tipo, e para a equação de Poisson em mais de duas dimensões são dadas por Bramble e outros conforme Fox [11]. Uma característica importante do artigo é que as estimativas são expressas em termo dos comportamentos de F e f e não em relação a incógnita u, onde os coeficientes de ∇^2 e o contorno são infinitamente diferenciáveis. O resultado principal é que se o grau de precisão da aproximação das equações a diferenças finitas for $0(h^k)$, e a matriz resultante for do tipo positivo, e se F e f, têm derivadas contínuas de ordem maior que $m < k$, então a razão da convergência é $0(h^m)$; caso $m > k$ a razão da convergência será $0(h^k)$.

Estes resultados para o caso singular indicam

que deveremos usar malhas muito pequenas em métodos não refinados, para conseguir uma precisão razoável para a aproximação U.

3.5 - Tratamento Descritivo da Convergência

Seja u uma representação da solução exata da equação diferencial parcial e U uma solução aproximada.

Então a solução é dita convergente quando U tende para u quando h tende a zero.

Embora a condição acima segundo a qual U converge para u esteja bem estabelecida para equações diferenciais parciais elípticas lineares, parabólicas e hiperbólicas de segunda ordem com soluções satisfazendo justamente as condições gerais de contorno e inicial, elas não são conhecidas para equações não lineares, exceto em poucos casos particulares. Como o nosso estudo refere-se a equações diferenciais parciais lineares elípticas de segunda ordem, com soluções satisfazendo as condições de Dirichlet, não nos importamos com as equações não lineares.

3.6 - Discretização da Equação de Poisson

Seja a equação $\nabla^2 u = F$ onde u e F são funções de x e y, isto é, funções de duas variáveis, com a condição de contorno de Dirichlet.

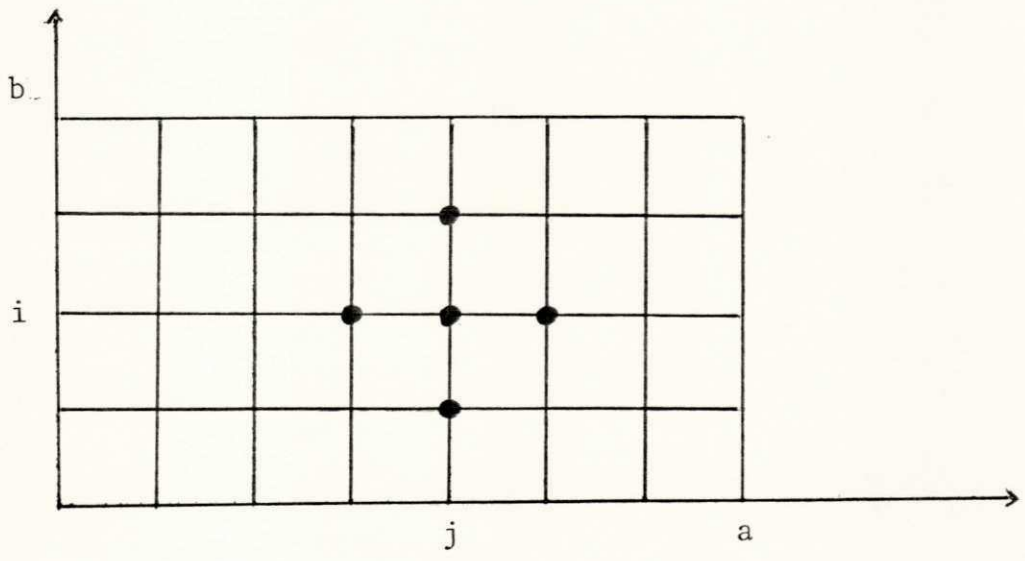
$u(x,y) = 0$ para $(x,y) \in \partial\Omega$

para a região retangular

$\Omega = \{x,y \mid 0<x<a,\ 0<y<b\} \subseteq \mathbb{R}^2$

com o contorno $\partial\Omega$

Utilizando a fórmula de cinco pontos na malha de discretização



Segue-se que

$$\frac{\partial^2 u}{\partial x^2} = \frac{U_{i-1,j} + U_{i+1,j} - 2U_{i,j}}{h^2}$$

$$\frac{\partial^2 u}{\partial y^2} = \frac{U_{i,j-1} + U_{i,j+1} - 2U_{i,j}}{k^2}$$

como $\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$, com $h = k$.

temos $U_{i-1,j} + U_{i+1,j} + U_{i,j-1} + U_{i,j+1} - 4U_{i,j} = f(i,j)$

onde $f(i,j) = h^2 F(x_i, y_j)$.

Variando i e j de modo que todos os pontos da região sejam considerados, resulta um sistema linear, cuja matriz dos coeficientes tem a estrutura e tamanho N x M.

$$
\begin{bmatrix}
-4 & 1 & & 0 & 1 & & & 0 & & & & \\
1 & -4 & 1 & & & \ddots & & & & & & \\
 & \ddots & \ddots & \ddots & & & \ddots & & & & & \\
 & & \ddots & \ddots & \ddots & & & \ddots & & & & \\
 & & & 1 & -4 & 1 & & & \ddots & & & \\
0 & & & 1 & -4 & 0 & & & & 1 & & \\
\hline
1 & & & 0 & -4 & 1 & & 0 & & & & \\
 & \ddots & & & 1 & -4 & 1 & & \ddots & & & \\
 & & \ddots & & & \ddots & \ddots & \ddots & & \ddots & & \\
 & & & \ddots & & \ddots & \ddots & \ddots & & & \ddots & \\
 & & & & 1 & -4 & 1 & & \ddots & & & 0 \\
 & & & & & 1 & -4 & 1 & & \ddots & & \\
0 & & & 1 & 0 & & 1 & -4 & & & & 1 \\
\hline
 & & & & \ddots & & & & 1 & & 0 & \\
 & & & & & \ddots & & & & \ddots & & \\
 & & & & & & \ddots & & & & \ddots & \\
 & & & & & & & \ddots & & & & 0 \\
 & & & & & & & & 0 & & & 1 \\
\hline
 & & & & 1 & & & 0 & -4 & 1 & & 0 \\
 & & & & & \ddots & & & 1 & -4 & 1 & \\
 & & & & & & \ddots & & & \ddots & \ddots & \ddots \\
 & & & & & & & \ddots & & \ddots & \ddots & \ddots \\
 & & & & & & & & & 1 & -4 & 1 \\
 & & & & 0 & & & 1 & 0 & & 1 & -4
\end{bmatrix}
\begin{bmatrix}
U_{1,1} \\ U_{1,2} \\ \vdots \\ \vdots \\ \vdots \\ U_{1,M} \\
\hline
U_{2,1} \\ U_{2,2} \\ \vdots \\ \vdots \\ \vdots \\ U_{2,M} \\
\hline
\vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\
\hline
U_{N,1} \\ U_{N,2} \\ \vdots \\ \vdots \\ \vdots \\ U_{N,M}
\end{bmatrix}
=
\begin{bmatrix}
f_{1,1} \\ f_{1,2} \\ \vdots \\ \vdots \\ \vdots \\ f_{1,M} \\
\hline
f_{2,1} \\ f_{2,2} \\ \vdots \\ \vdots \\ \vdots \\ f_{2,M} \\
\hline
\vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\
\hline
f_{N,1} \\ f_{N,2} \\ \vdots \\ \vdots \\ \vdots \\ f_{N,M}
\end{bmatrix}
$$

Os N blocos ao longo da diagonal

$$
A = \begin{bmatrix}
-4 & 1 & & 0 \\
1 & -4 & 1 & \\
 & & & \\
 & & & \\
 & & & \\
0 & 1 & -4 & 1 \\
 & & 1 & -4
\end{bmatrix}
$$

são MxM, tendo 3M-2 elementos não nulos. Os 2N-2 blocos da sub- e super-diagonais são matrizes identidades cada um com M elementos não nulos. A matriz pentadiagonal tem o total de 5MN-2M-2N elementos.

CAPITULO IV

FAST POISSON SOLVERS

4.1 - Introdução

Durante os últimos 10 anos tornou-se altamente popular resolver a equação discreta de Poisson (e problemas correlatos) através de métodos diretos ao invés de iterativos.

Primeiramente tais métodos foram aplicados apenas as equações de Poisson numa grade retangular NxM, onde N era restrito à forma $N = 2^k$.

Mais recentemente, os métodos diretos foram aplicados a outras regiões retangulares com N arbitrário, a regiões irregulares e a equações elípticas gerais separáveis.

Muitos dos algoritmos desenvolvidos para a equa-

ção de Poisson caem em duas categorias aparentemente distintas: as baseadas na decomposição de Fourier numa dimensão, usando as técnicas da transformada rápida de Fourier (FFT) e as baseadas na redução bloco cíclica (algoritmo de Buneman).

É natural perguntarmos qual das duas abordagens leva ao algoritmo mais rápido; pelo que dispomos a respeito, a resposta não é clara. Por exemplo, Sweet (1973) (segundo [17]) acha que para um problema em particular, o algoritmo de Buneman é pelo menos duas vezes mais rápido que o método baseado em FFT enquanto Fischer et alli (1974) (segundo [17]) acha que os métodos FFT são mais rápidos.

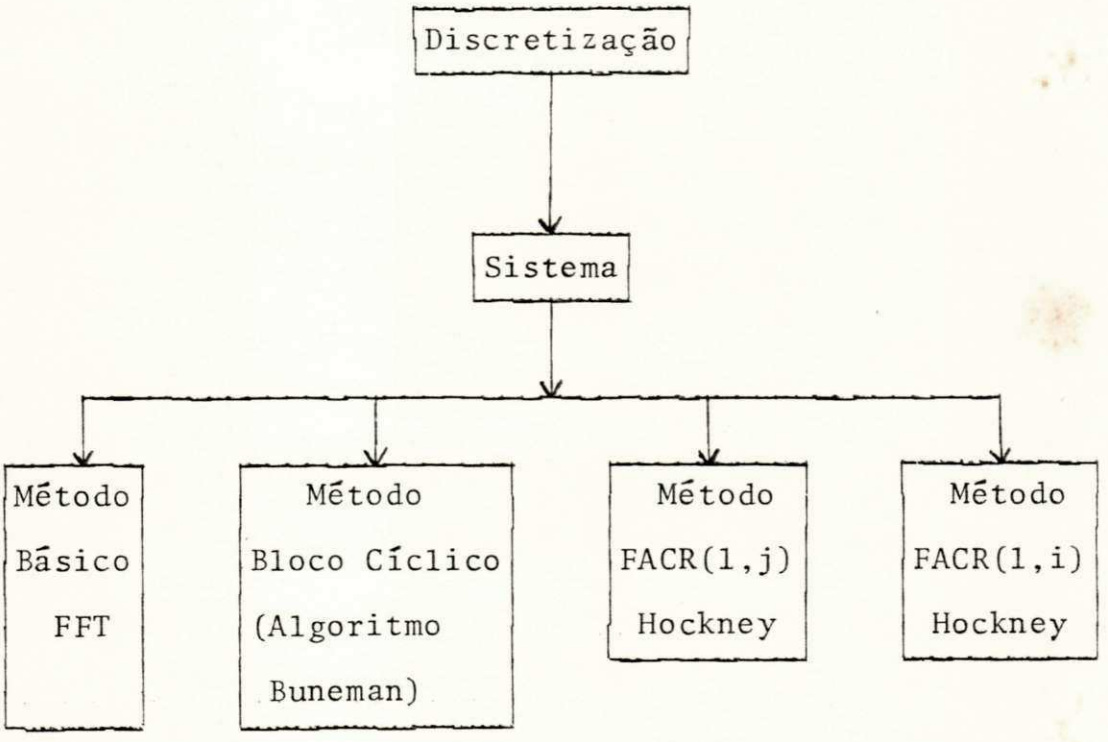
As razões para esta confusão são devidas a variedade de métodos disponíveis que poderão ser usados na aplicação dos algoritmos (principalmente as transformadas rápidas de Fourier e a solução de sistemas tridiagonais) e nas suposições feitas por diferentes autores. Hockney (1965, 1966) (segundo [17]) apresentou uma variante do algoritmo FACR(1) no qual a análise de Fourier é combinada com um passo preliminar da redução cíclica.

Além dos métodos FFT, redução bloco-cíclica (algoritmo de Buneman) FACR(1), FAC(1,1) existem outros métodos directos tais como redução total (Schroder (1973), segundo [17]) e marching (fronteira) (Schumann (1978),

veja [17]).

Descrevemos somente quatro variantes: Método Básico FFT, Método bloco cíclico (algoritmo de Buneman), FACR (1,j) (Hockney) e FACR (1,i) (Hockney).

Todas as quatro variantes têm um ponto de partida. Inicialmente discretizamos a equação de Poisson obtendo assim sistemas de equações lineares. Para a resolução dos sistemas de equações é que utilizamos as quatro variantes conforme a figura.



4.2 - O Método Básico FFT

Nós aqui trataremos o método básico FFT para a equação de Poisson com as condições de contorno de Dirichlet em i=0,N. Em cada linha expressamos os valores da solução (incluindo os valores de contorno) como soma dos coeficientes de seno de Fourier:

$$U_{i,j} = \sum_{K=1}^{N-1} \hat{U}_{k,j} \operatorname{sen}(ik\pi/N), \quad 0 \leq i \leq N, \quad 0 \leq j \leq M \tag{1}$$

substituindo na equação

$$U_{i-1,j} + U_{i+1,j} + U_{i,j-1} + U_{i,j+1} - 4U_{i,j} = b_{i,j}$$

obtemos (após algumas manipulações) (N-1) sistemas tridiagonais com M-1 incógnitas, cada um da forma

$$\hat{U}_{k,j-1} + \lambda_k \hat{U}_{k,j} + \hat{U}_{k,j+1} = \hat{b}_{k,j} \quad 1 \leq k \leq N-1, \quad 1 \leq j \leq M-1 \tag{2}$$

onde $\lambda_k = 2\cos(K\pi/N) - 4$ e

$$\hat{b}_{k,j} = (2/N) \sum_{i=1}^{N-1} b_{i,j} \operatorname{sen}(ik\pi/N) \tag{3}$$

O processo de solução tem assim três estágios.

No primeiro estágio é aplicada uma transformada

do seno em cada linha do 2º membro para implementar a equação (3) e obtém-se os coeficientes $\hat{b}_{k,j}$.

Então a equação (2) é resolvida para cada valor de k no segundo estágio. No terceiro estágio é aplicada uma transformada inversa de seno em cada linha para implementar a equação (1).

Podemos verificar que se não usarmos o FFT teremos $(N-1)\ (M-1)^2$ multiplicações, ao passo que utilizando FFT ao todo temos que executar 2(M-1) transformadas de seno de comprimento N e resolver (M-1) sistemas tridiagonais de ordem (M-1).

Para condições de contorno periódicas onde i=0,N substituimos a transformada de seno pela transformada periódica total

$$U_{i,j} = \frac{1}{2}\bar{U}_{0,j} + \frac{1}{2}(-1)^i \bar{U}_{N/2,j} + \sum_{k=1}^{N/2-1} \{\bar{U}_{k,j}\cos(2ik\pi/N + \bar{U}_{k,j}\operatorname{sen}(2ik\pi/N))\} \qquad (4)$$

e obtemos (N/2+1) sistemas da forma

$$\bar{U}_{k,j-1} + \lambda_k \bar{U}_{k,j} + \bar{U}_{k,j+1} = \hat{b}_{k,j}, \quad 0 \leq k \leq N/2, \quad 1 \leq j \leq M-1 \qquad (5)$$

onde $\lambda_k = 2\cos(2k\pi/N) - 4$

$$\hat{b}_{k,j} = (2/N) \sum_{i=0}^{N-1} b_{i,j} \cos(2ik\pi/N)$$

juntamente com (N/2-1) sistemas da forma

$$\bar{U}_{k,j-1} + \lambda_k \hat{U}_{k,j+1} = \hat{b}_{k,j} \quad , \quad 1 \leq k \leq N/2-1 \quad , \quad 1 \leq j \leq M-1 \ ,$$

onde λ_k é novamente dado pela equação (5) e

$$\hat{b}_{k,j} = (2/N) \sum_{i=0}^{N-1} b_{i,j} \operatorname{sen}(2ik\pi/N)$$

Neste caso temos que executar 2(M-1) transformadas periódicas de comprimento N e resolver N sistemas tridiagonais de ordem (M-1).

O número de coeficientes usados na solução de sistemas tridiagonais é (N/2+1)(M-1).

A substituição de $U_{i,j}$ por $\sum_{k=1}^{N-1} U_{k,j} \operatorname{sen}(ik\pi/N)$

$0 \leq i \leq N \ , \ 0 \leq j \leq M$

permite a obtenção de sistemas de equações tridiagonais, cujas soluções são mais rápidas que os sistemas pontadiagonais obtidos na discretização.

A utilização do FFT no cálculo das somatórias dos produtos de $\hat{U}_{k,j}$ e $b_{i,j}$ pelo sen(ikπ/N), traz uma gran-

de economia de tempo comparado com a computação direta.

4.3 - Redução Bloco Cíclica

O método é detalhado em Buzbee et alli [6].

Daremos a seguir o algoritmo de Buneman para a solução da equação discretizada de Poisson. Generalizando o problema modelo, isto é, após a discretização da equação de Poisson no modo usual, nós obtemos para os valores aproximados $U_{i,j}$ de $u(x_i,y_j)$, $x_i=ih$, $y_j=jk$, $h=a/(N+1)$, $k=b/(M+1)$, as equações

$$\frac{-U_{i-1,j}+2U_{ij}-U_{i+1,j}}{h^2} + \frac{U_{i,j-1}+2U_{ij}-U_{i,j+1}}{k^2} + \tau U_{ij}=f_{ij}=f(x_i,y_j)$$

Para $i = 1,2,\ldots,N$, $j = 1,2,\ldots,M$

juntamente com os valores de contorno

$U_{0,j} = U_{N+1,j} = 0$ para $j = 0,1,\ldots,M+1$,

$U_{i,0} = U_{i,M+1} = 0$ para $i = 0,1,\ldots,N+1$,

obtemos assim para as incógnitas

$$\underline{U} = \begin{bmatrix} \underline{U}_1 \\ \underline{U}_2 \\ \vdots \\ \underline{U}_M \end{bmatrix} \quad , \quad \underline{U}_j = \left[\underline{U}_{j1}, \underline{U}_{j2}, \ldots, \underline{U}_{jN} \right]^T \quad , \quad 1 \leq j \leq M$$

um sistema de equações lineares, que pode ser escrito na forma

$$B\underline{U} = \underline{b} \tag{6}$$

com

$$B = \begin{bmatrix} A & I & & 0 \\ I & A & I & \\ & \ddots & \ddots & \ddots \\ & & \ddots & I \\ 0 & & I & A \end{bmatrix} \quad , \quad \underline{b} = \begin{bmatrix} \underline{b}_1 \\ \underline{b}_2 \\ \vdots \\ \underline{b}_M \end{bmatrix} \tag{7}$$

onde $I = I_p$ é a matriz identidade NxN, A é a matriz Hermitiana NxM e B consiste de M blocos, cuja ordem de cada bloco é igual ao número de pontos da linha.

A observação seguinte é essencial para o método da redução de Buneman. Podemos escrever o sistema de equações como

$$\begin{aligned} A\underline{U}_1 + \underline{U}_2 &= \underline{b}_1 \\ \underline{U}_{j-1} + A\underline{U}_j + \underline{U}_{j+1} &= \underline{b}_j \quad j = 2,3,\ldots,M-1 \end{aligned} \tag{8}$$

$$\underline{U}_{M-1} + A\underline{U}_M = \underline{b}_M$$

Das três equações consecutivas

$$\underline{U}_{j-2} + A\underline{U}_{j-1} + \underline{U}_j = \underline{b}_{j-1}$$

$$\underline{U}_{j-1} + A\underline{U}_j + \underline{U}_{j+1} = \underline{b}_j$$

$$\underline{U}_j + A\underline{U}_{j+1} + \underline{U}_{j+2} = \underline{b}_{j+1}$$

podemos para todos j pares $j = 2,4,\ldots$, eliminar os vetores $\underline{U}_{j+1}$ da seguinte maneira: multiplicando a segunda equação por-A e somando com as outras duas equações teremos

$$\underline{U}_{j-2} + (2I-A^2)\underline{U}_j + \underline{U}_{j+2} = \underline{b}_{j-1} - A\underline{b}_j + \underline{b}_{j+1}$$

Para M ímpar, desta forma obtemos o sistema reduzido

$$\begin{bmatrix} 2I-A^2 & I & & & \\ I & 2I-A^2 & I & & \\ & \ddots & \ddots & \ddots & \\ & & & I & \\ & & I & 2I-A^2 \end{bmatrix} \begin{bmatrix} \underline{U}_2 \\ \underline{U}_4 \\ \vdots \\ \underline{U}_{M-1} \end{bmatrix} = \begin{bmatrix} \underline{b}_1 + \underline{b}_3 - A\underline{b}_2 \\ \underline{b}_3 + \underline{b}_5 - A\underline{b}_4 \\ \vdots \\ \underline{b}_{M-2} + \underline{b}_M - A\underline{b}_{M-1} \end{bmatrix} \quad (9)$$

para $\underline{U}_2$, $\underline{U}_4,\ldots$ Imediatamente uma solução de (9) (isto é , os subvetores $\underline{U}_{2j}$ com índice pares) é conhecido, os vetores $\underline{U}_1$, $\underline{U}_3,\ldots$ com índices ímpares podem ser determinados das

equações seguintes as quais seguem imediatamente de (8) para j = 1, 3...

$$\begin{bmatrix} A & & & 0 \\ & A & & \\ & & A & \\ & & & \ddots \\ 0 & & & A \end{bmatrix} \begin{bmatrix} \underline{U}_1 \\ \underline{U}_3 \\ \underline{U}_5 \\ \vdots \\ \underline{U}_M \end{bmatrix} = \begin{bmatrix} \underline{b}_1 - \underline{U}_2 \\ \underline{b}_3 - \underline{U}_2 - \underline{U}_4 \\ \underline{b}_5 - \underline{U}_4 - \underline{U}_6 \\ \vdots \\ \underline{b}_M - \underline{U}_{M-1} \end{bmatrix} \qquad (10)$$

Deste modo, a solução de (6) é reduzida para a solução do sistema (9) com metade do número de incógnitas e subsequente solução de (10). Agora (9) novamente tem a mesma estrutura de (7).

$$B^{(1)} \; \underline{U}^{(1)} = \underline{b}^{(1)}$$

com

$$B^{(1)} = \begin{bmatrix} A^{(1)} & I & & 0 \\ I & A^{(1)} & I & \\ & & & \\ & & & I \\ 0 & & I & A^{(1)} \end{bmatrix} \qquad A^{(1)} = 2I - A^2$$

$$\underline{U}^{(1)} = \begin{bmatrix} \underline{U}_1^{(1)} & \underline{U}_2 \\ \underline{U}_2^{(1)} & \underline{U}_4 \\ \vdots & \vdots \\ \underline{U}_{M1}^{(1)} & \underline{U}_{M-1} \end{bmatrix} \quad b^{(1)} = \begin{bmatrix} \underline{b}_1^{(1)} \\ \underline{b}_2^{(1)} \\ \vdots \\ \underline{b}_{M1}^{(1)} \end{bmatrix} = \begin{bmatrix} \underline{b}_1 & \underline{b}_3 - A\underline{b}_2 \\ \underline{b}_3 & \underline{b}_5 - Ab_4 \\ \vdots & \\ \underline{b}_{M-2} + \underline{b}_M & - A\underline{b}_{M-1} \end{bmatrix}$$

de modo que a mesma redução descrita acima pode ser novamente aplicada para $B^{(1)}$ etc. Em geral para $M = M_0 = 2^{k+1}-1$, obtemos deste modo a sequência de matrizes $A^{(r)}$ e vetores $\underline{b}^{(r)}$, conforme procedimento seguinte. (11)

Inicialização: Defina $A^{(0)}=A$; $\underline{b}_j^{(0)}=\underline{b}_j$ $j=1,2,\ldots,M_0$

$M_0=M=2^{k+1}-1$ Para $r = 0,1,2,\ldots,k-1$; Por

a) $A^{(r+1)} = 2I - (A^{(r)})^2$

b) $\underline{b}_j^{(r+1)} = \underline{b}_{2j-1}^{(r)} - A^{(r)}\underline{b}_{2j}^{(r)}$ $j = 1,2,\ldots,2^{k+1}-1 = M_{r+1}$

Para cada estágio r+1, r=0,1,...,k-1, assim obtemos o sistema de equação linear

$$B^{(r+1)}\ \underline{U}^{(r+1)} = \underline{b}^{(r+1)}$$

ou, escrevendo na forma completa

$$\begin{bmatrix} A^{(r+1)} & I & & 0 \\ I & A^{(r+1)} & & \\ & & & \\ & & I & \\ 0 & I & & A^{(r+1)} \end{bmatrix} \begin{bmatrix} \underline{U}_1^{(r+1)} \\ \underline{U}_2^{(r+1)} \\ \vdots \\ \underline{U}_{M_{r+1}} \end{bmatrix} = \begin{bmatrix} \underline{b}_1^{(r+1)} \\ \underline{b}_2^{(r+1)} \\ \vdots \\ \underline{b}_{M_{r+1}}^{(r+1)} \end{bmatrix}$$

Esta solução $\underline{U}^{(r+1)}$ fornece imediatamente os sub-vetores com índices pares da solução $\underline{U}^{(r)}$ do sistema de equações correspondentes $B^{(r)}\ \underline{U}^{(r)} = \underline{b}^{(r)}$ no estágio r.

$$\begin{bmatrix} \underline{U}_2^{(r)} \\ \underline{U}_4^{(r)} \\ \vdots \\ \underline{U}_{M_{r-1}}^{(r)} \end{bmatrix} = \begin{bmatrix} \underline{U}_1^{(r+1)} \\ \underline{U}_2^{(r+1)} \\ \vdots \\ \underline{U}_{M_{r+1}}^{(r+1)} \end{bmatrix}$$

enquanto os subvetores com índices ímpares de $U^{(r)}$ podem ser obtidos pela solução da equação

$$\begin{bmatrix} A^{(r)} & & 0 \\ & A^{(r)} & \\ & & \\ 0 & & A^{(r)} \end{bmatrix} \begin{bmatrix} \underline{U}_1^{(r)} \\ \underline{U}_3^{(r)} \\ \vdots \\ \underline{U}_{M_r}^{(r)} \end{bmatrix} = \begin{bmatrix} \underline{b}_1^{(r)} - \underline{U}_2^{(r)} \\ \underline{b}_3^{(r)} \quad \underline{U}_2^{(r)} - \underline{U}_4^{(r)} \\ \vdots \quad \vdots \\ \underline{b}_{M_r}^{(r)} - \underline{U}_{M_{r-1}}^{(r)} \end{bmatrix}$$

Dos dados $A^{(r)}$, $\underline{b}^{(r)}$ produzidos por (11), a solução $\underline{U}=\underline{U}^{(0)}$ de (7) é assim finalmente obtida pelo seguinte procedimento:

a) Inicialização: Determine $\underline{U}^{(k)} = \underline{U}_1^{(k)}$ pela solução do sistema de equações

$$A^{(k)}\underline{U}^{(k)} = \underline{b}^{(k)} = \underline{b}_1^{(k)}$$

b) Para r = k-1, k-2,...,0

1) Fazer $\underline{U}_{2j}^{(r)} = \underline{U}_j^{(r+1)}$; $j = 1,2,\ldots,M_{r+1} = 2^{k+1}-1$

2) Para $j = 1,3,5,\ldots,M_r$ compute o vetor $\underline{U}_j^{(r)}$ pela solução de

$$A^{(r)}\underline{U}_j^{(r)} = \underline{b}_j^{(r)} - \underline{U}_{j-1}^{(r)} - \underline{U}_{j+1}^{(r)} \quad (\underline{U}_0^{(r)} = \underline{U}_{M_{r+1}}^{(r)} = 0)$$

c) Fazer $\underline{U} = \underline{U}^{(0)}$

Com o algoritmo na forma (11) e (12) tem-se sérias desvantagens numéricas. Em primeiro lugar, a computação explícita de $A^{(r+1)} = 2I-(A^{(r)})^2$ em (11.a) é dispendiosa. A matriz tridiagonal $A^{(0)} = A$ ao incrementar r, a matriz $A^{(r)}$ torna-se rapidamente densa que é uma matriz banda com a largura de banda (2^r+1), de modo que, primeiramente faz-se a computação de $(A^{(r)})^2$, e em seguida a solução do sistema de equações lineares em (12.b.2). A medida que r cresce a computação torna-se mais e mais dis-

pendiosa.

Para o problema modelo (6), isto é, com

$$\begin{bmatrix} -4 & 1 & 0 \\ 1 & -4 & \\ & & 1 \\ 0 & 1 & -4 \end{bmatrix}$$

$$\| A^{(0)} \| \geq 4 \qquad \| A^{(r)} \| = \| A^{(r-1)} \| \geq 4^{2r}$$

de maneira que na computação de $b_j^{(r+1)}$ em (11.b) incorremos em substancial negligência da precisão para grandes valores de r, assim em geral,

$$\| A^{(r)} \underline{b}_{2j}^{(r)} \| > \| \underline{b}_{2j-1}^{(r)} \|, \| \underline{b}_{2j+1}^{(r)} \| ,$$

e portanto da informação contida em $\underline{b}_{2j-1}^{(r)}$, $\underline{b}_{2j+1}^{(r)}$, ao efetuar a adição em (11) obtém-se uma resposta não precisa.

Ambos obstáculos podem ser evitados por uma reformulação conveniente do algoritmo.

A computação explícita de $A^{(r)}$ é evitada se explorarmos o fato que $A^{(r)}$ pode ser representada como um produto de matrizes tridiagonais.

Teorema (Bulirsch & Stoer [5]) (13)

Temos, para todo $r \geq 0$,

$$A^{(r)} = -\prod_{j=1}^{2^r} \left[-(A+2\cos\theta_j^{(r)} I) \right],$$

onde $\theta_j^{(r)} = (2j-1)\pi/2^{r+1}$ para $j = 1,2,\ldots,2^r$

Prova: Por (11.a), temos com $A^{(0)} = A$,

$$A^{(r+1)} = 2I - (A^{(r)})^2$$

desta maneira existe um polinômio $P_r(t)$ de grau 2^r tal que

$$A^{(r)} = P_r(A) \qquad (14)$$

Evidentemente, o polinômio P_r satisfaz

$$P_0(t) = t$$

$$P_{r+1}(t) = 2-(P_r(t))^2,$$

deste modo P_r tem a forma

$$P_r(t) = -(-t)^{2r} + \ldots \qquad (15)$$

Agora mostraremos por indução matemática, usando a substituição $t = -2\cos\theta$, que

$$P_r(-2\cos\theta) = 2\cos(2^r\theta) \qquad (16)$$

A fórmula é trivial para r=0. Se ela for válida para algum $r \geq 0$, então, para r+1

$$P_{r+1}(-2\cos\theta) = 2-(P_r(-2\cos\theta)^2$$

$$= 2-4\cos^2(2^r\theta)$$

$$= -2\cos(2.2^r\theta)$$

$$= -2\cos(2^{r+1}\theta)$$

Na visão de (16) o polinômio $P_r(t)$ tem 2^r distintos zeros reais.

$$t_j = 2\cos(\frac{2j-1}{2^{r+1}}\pi), \quad j = 1,2,\ldots,2^r,$$

e portanto, por (15) a apresentação do produto

$$P_r(t) = -\prod_{j=1}^{2r} [-(t-t_j)]$$

Disto, em virtude de (11), a afirmação do teorema segue de imediato.

O teorema precedente pode ser agora empregado, na prática para reduzir a solução dos vários sistemas de equações

$$A^{(r)}\underline{x} = \underline{c}$$

em (12.b.2), com as matrizes $A^{(r)}$, recursivamente para a solução dos 2^r sistemas de equações com matrizes tridiagonais

$$A_j^{(r)} = -A-2\cos\theta_j^{(r)}, \quad j = 1,2,\ldots,2^r$$

como segue

$$A_1^{(r)} \underline{v}_1 = \underline{c}$$

$$A_2^{(r)} \underline{v}_2 = \underline{v}_1 \qquad (17)$$

$$A_{2^r}^{(r)} \underline{v}_{2^r} = \underline{u}_{2^r-1} \Longrightarrow \underline{v}_{2^r} \Longrightarrow \underline{x} = -\underline{v}_{2^r}$$

Assim, como é fácil verificar, as matrizes tridiagonais $A_j^{(r)}$ para nossa discretização do problema são positivas definidas. Podemos resolver estes sistemas muito economicamente por meio da fatoração triangular de $A_j^{(r)}$.

A instabilidade numérica que ocorre em (11.b) por causa do crescimento exponencial da norma de $A^{(r)}$ pode ser evitada, seguindo a sugestão de Buneman, pela introdução no lugar de $\underline{b}_j^{(r)}$ outros vetores $\underline{P}_j^{(r)}$, $q_j^{(r)}$, $j = 1,2,\ldots,M_r$, os quais estão relacionados com $\underline{b}_j^{(r)}$ do seguinte modo

$$\underline{b}_j^{(r)} = A^{(r)} \underline{P}_j^{(r)} + \underline{q}_j^{(r)} \quad j = 1,2,\ldots,M_r \qquad (18)$$

e o qual pode ser computado numericamente de uma maneira mais estável que $\underline{b}_j^{(r)}$. Os vetores $\underline{P}_j^{(r)}$, $\underline{q}_j^{(r)}$ com estas propriedades são geradas recursivamente como se segue

Inicialização: Fazer $\underline{P}_j^{(0)} = 0$, $\underline{q}_j^{(0)} = \underline{b}_j = \underline{b}_j^{(0)}$, $j=1,2,\ldots,M_0$ (19)

Para $r = 0,1,\ldots,k-1$

Compute para $j = 1,2,\ldots,M_{r+1}$

a) $\underline{P}_j^{(r+1)} = \underline{P}_{2j}^{(r)} - (A^{(r)})^{-1} \left[\underline{P}_{2j+1}^{(r)} - \underline{q}_{2j}^{(r)} \right]$,

b) $\underline{q}_j^{(r+1)} = \underline{q}_{2j}^{(r)} + \underline{q}_{2j+1}^{(r)} - 2\underline{P}_j^{(r+1)}$

Naturalmente, a computação de $\underline{P}_j^{(r+1)}$ no subpasso (a) atinge primeiro determinando, como exata descrição a solução u do sistema de equações

$$A^{(r)}\underline{u} = \underline{P}_{2j-1}^{(r)} + \underline{P}_{2j+1}^{(r)} - \underline{q}_{2j}^{(r)}$$

Com o auxílio da fatoração de $A^{(r)}$ no teorema (13), e então computando $\underline{P}_j^{(r+1)}$ de u por meio de

$$\underline{P}_j^{(r+1)} = \underline{P}_{2j}^{(r)} - \underline{u} ,$$

permite-nos provar por indução em r que os vetores $\underline{P}_j^{(r)}$, $\underline{q}_j^{(r)}$, definidos por (19), satisfazem a relação (18).

Para $r = 0$ (18) é trivial. Assumindo que (18) é verdadeira para algum $r \geq 0$, vamos mostrar que é verdadeira para r+1. Por causa de (11.b) e $A^{(r+1)} = 2I-(A^{(r)})^2$, temos então

$$\underline{b}_j^{(r+1)} = \underline{b}_{2j+1}^{(r)} + \underline{b}_{2j-1}^{(r)} - A^{(r)}\underline{b}_{2j}^{(r)}$$

$$= A^{(r)}\underline{P}_{2j+1}^{(r)}+\underline{q}_{2j+1}^{(r)}+A^{(r)}\underline{P}_{2j-1}+\underline{q}_{2j-1}^{(r)}-A^{(r)}\left[\underline{A}^{(r)}P_{2j}^{(r)}+q_{2j}^{(r)}\right]$$

$$= A^{(r)}\left[\underline{P}_{2j+1}^{(r)}+\underline{P}_{2j-1}^{(r)}-\underline{q}_{2j}^{(r)}\right] + A^{(r+1)}\underline{P}_{2j}^{(r)}+\underline{q}_{2j-1}^{(r)}+$$

$$+\underline{q}_{2j+1}^{(r)}-2\underline{P}_{2j}^{(r)}$$

$$= A^{(r+1)}\underline{P}_{2j}^{(r)}+(A^{(r)})^{-1}\{\left[2I-A^{(r+1)}\right]\left[\underline{P}_{2j+1}^{(r)}+\right.$$

$$\left.+\underline{P}_{2j-1}^{(r)}-\underline{q}_{2j}^{(r)}\right]\}+\underline{q}_{2j-1}^{(r)}+\underline{q}_{2j+1}^{(r)}-2\underline{P}_{2j}^{(r)}$$

$$= A^{(r+1)}\{\underline{P}_{2j}^{(r)}-(A^{(r)})^{-1}\left[\underline{P}_{2j-1}^{(r)}+\underline{P}_{2j+1}^{(r)}-\underline{q}_{2j}^{(r)}\right]\}+$$

$$+\underline{q}_{2j-1}^{(r)}+\underline{q}_{2j+1}^{(r)}-2\underline{P}_{j}^{(r+1)}$$

$$= A^{(r+1)}\underline{P}_{j}^{(r+1)}+\underline{q}_{j}^{(r+1)}$$

Por (18) podemos agora expressar os vetores $b_j^{(r)}$ (12) nos termos de $\underline{P}_j^{(r)}$, $q_j^{(r)}$ e obter, por exemplo de (12.b.2) de $\underline{U}_j^{(r)}$ o sistema de equações

$$A^{(r)}\underline{U}_j^{(r)} = A^{(r)}\underline{P}_j^{(r)}+\underline{q}_j^{(r)}-\underline{U}_{j-1}^{(r)}-\underline{U}_{j+1}^{(r)} ,$$

o qual pode ser resolvido como segue:

Determine a solução $\underline{u}$ de

$$\underline{A}^{(r)}\underline{u} = \underline{q}_j^{(r)}-\underline{U}_{j-1}^{(r)}-\underline{U}_{j+1}^{(r)}$$

[usamos novamente a fatoração do teorema (13)] e fazemos $\underline{U}_j^{(r)} = \underline{u}+\underline{P}_j^{(r)}$. Substituindo deste modo $\underline{b}_j^{(r)}$ em (11), (12)

sistematicamente por $\underline{P}_j^{(r)}$ e $\underline{q}_j^{(r)}$, obtemos o Algoritmo de Buneman

Suposição: Considere o sistema de equações

$B\underline{U} = \underline{b}$, com $M = 2^{k+1}-1$

a) Inicialização: Fazer $\underline{P}_j^{(0)} = 0$, $\underline{q}_j^{(r)} = \underline{b}_j$, $j=1,2$ $M_0=M$

b) Para $r = 0,1,\ldots,k-1$

para $j = 1,2,\ldots,M_{r+1} = 2^{k-r}-1$

Compute a solução u do sistema de equações

$$A^{(r)}\underline{u} = \underline{P}_{2j-1}^{(r)} + \underline{P}_{2j+1}^{(r)} - \underline{q}_{2j}^{(r)}$$

por meio da fatoração do teorema (13) e faça

$$\underline{P}_j^{(r+1)} = \underline{P}_{2j}^{(r)} - \underline{U}, \underline{q}_j^{(r+1)} = \underline{q}_{2j-1}^{(r)} + \underline{q}_{2j+1}^{(r)} - 2\underline{P}_j^{(r+1)}$$

c) Determine a solução $\underline{u}$ do sistema de equações

$$A^{(k)}\underline{u} = \underline{q}_1^{(k)} ,$$

$$\underline{U}^{(k)} = \underline{U}_1^{(k)} = \underline{P}_1^{(k)} + \underline{u}$$

e faça

d) Para $r = k-1, k-2,\ldots,0$

1) Faça $\underline{U}_{2j}^{(r)} = \underline{U}_j^{(r+1)}$ para $j = 1,2,\ldots,M_{r+1}$

2) Para $j = 1,3,5,\ldots,M_r$, determine a solução u do sistema de equações

$$A^{(r)}\underline{u} = \underline{q}_j^{(r)} - \underline{U}_{j-1}^{(r)} - \underline{U}_{j+1}^{(r)}$$

e faça $\underline{U}_j^{(r)} = \underline{P}_j^{(r)} + \underline{u}$

e) Faça $\underline{U} = \underline{U}^{(0)}$

Este método é muito eficiente. Uma operação importante mostra que para a solução do problema modelo (6) ($c=b=1$, $p=q=N=2^{(k+1)}-1$), com N^2 incógnitas, necessitamos de aproximadamente $3KN^2 \simeq 3N^2\log_2 N$ multiplicações e cerca do mesmo número de adições. Uma análise da estabilidade numérica do método foi feita por Buzbee, Golub e Nielson [6].

Na forma descrita, o método serve para resolver o sistema de equações lineares proveniente da discretização da equação de Poisson com valor de contorno de Direchlet com domínios retangulares.

4.4 - **Métodos FACR(1)**

4.4.1 - Algoritmo FACR(1,j) de Hockney

Hockney [12] mostrou que metade das transfor-

madas no método básico FFT pode ser eliminada primeiro executando um passo da redução cíclica na direção j, eliminando todo $U_{i,j}$, com j ímpar.

Para isso tomamos as equações de 5 pontos, onde (i,j) são as coordenadas do ponto central.

$$U_{i-1,j}+U_{i,j-1}+U_{i,j-2}+U_{i,j}-4U_{i,j-1} = b_{i,j-1}$$

$$U_{i-2,j}+U_{i,j}+U_{i-1,j+1}+U_{i-1,j+1}-4U_{i-1,j} = b_{i-1,j}$$

$$U_{i-1,j}+U_{i+1,j}+U_{i,j-1}+U_{i,j+1}-4U_{i,j} = b_{i,j}$$

$$U_{i,j}+U_{i+1,j}+U_{i+1,j-1}+U_{i+1,j+1}-4_{i+1,j} = b_{i+1,j}$$

$$U_{i-1,j+1}+U_{i+1,j+1}+U_{i,j}+U_{i,j+2}-4_{i,j+1} = b_{i,j+1}$$

Multiplicando a segunda e a quarta equação por -1, a terceira por 4 e em seguida adicionando as cinco equações, as equações resultantes ficam na forma:

$$U_{i,j-2}-U_{i-2,j}+8U_{i-1,j}-16U_{i,j}+8U_{i+1,j}-U_{i+2,j}+U_{i,j+2} =$$

$$= b^{(1)}_{i,j} = b_{i,j-1}-b_{i-1,j}+4b_{i,j}-b_{i+1,j}+b_{i,j+1} \qquad (20)$$

$$1 \leq i \leq N \quad e \quad 1 \leq j \leq M$$

Com as seguintes condições:

Para $i-1<1$, $i-2<1$, $j-1<1$, $j-2<1$

$i+1>N$, $i+2>N$, $j+1>M$ e $j+2>M$

temos $U_{i,j} = 0$

Novamente podemos escrever $U_{i,j}$ como na equação (1) (desta vez só para j par) e obtemos (condições de contorno de Dirichlet em i=0,N) sistemas tridiagonais da forma

$$\hat{U}_{k,j-2}+\lambda_k\hat{U}_{k,j}+\hat{U}_{k,j+2} = \hat{b}^{(1)}_{k,j} \quad , \quad 1\leq k\leq N-1, \quad j = 2,4,\ldots,M-2$$

$$\text{onde } \lambda_k = 2-4(\cos(k\pi/N)-2)^2 \qquad (21)$$

$$\text{e } \hat{b}^{(1)}_{k,j} = (2/N)\sum_{i=1}^{N-1} b^{(1)}_{i,j}\ \text{sen}(ik\pi/N)$$

Consequentemente primeiro obtemos $b^{(1)}_{i,j}$ para j par usando a equação (20); então executando a transformada do seno para j par usando a equação (21). Seguindo a solução de (N-1) de sistemas tridiagonais de ordem (M/2-1) executamos as transformadas inversas do seno para j par e finalmente a solução $U_{i,j}$ para j ímpar é obtida resolvendo sistemas tridiagonais ao longo das linhas.

Para o método de Hockney o número de coeficientes do sistema requerido para soluções tridiagonais é (M/2-1)(N-1) nas condições de contorno de Dirichlet com i=0,N.

4.4.2 - FACR(1,i)

Em comparação com o método básico FFT, o algoritmo de Hockney divide o número de transformadas de

Fourier requeridas formando um conjunto de equações envolvendo $U_{i,j}$ apenas para j par. Uma estratégia alternativa é obter a metade do número das transformadas de Fourier formando um conjunto análogo envolvendo $U_{i,j}$ apenas para i par.

Sejam as equações de cinco pontos consecutivos.

$$U_{i-2,j}+U_{i,j}+U_{i-1,j-1}+U_{i-1,j+1}-4U_{i-1,j} = b_{i-1,j}$$

$$U_{i-1,j-1}+U_{i+1,j-1}+U_{i,j-2}+U_{i,j}-4U_{i,j-1} = b_{i,j-1}$$

$$U_{i-1,j}+U_{i+1,j}+U_{i,j-1}+U_{i,j+1}-4U_{i,j} = b_{i,j}$$

$$U_{i-1,j+1}+U_{i+1,j+1}+U_{i,j}+U_{i,j+2}-4U_{i,j+1} = b_{i,j+1}$$

$$U_{i,j}+U_{i+2,j}+U_{i+1,j-1}+U_{i+1,j+1}-4U_{i+1,j} = b_{i+1,j}$$

Multiplicando a segunda e quarta equação por -1, a terceira por 4 e em seguida adicionando as cinco equações, as equações resultantes são da forma

$$U_{i-2,j}-U_{i,j-2}+8U_{i,j-1}-16U_{i,j}+8U_{i,j+1}-U_{i,j+2}+U_{i+2,j} =$$

$$= b^{(1)}_{i,j} = b_{i-1,j} - b_{i,j-1} +4b_{i,j} - b_{i,j+1} + b_{i+1,j} \qquad (22)$$

$$1 \leq i \leq N \quad e \quad 1 \leq j \leq M$$

Com as seguintes condições:

Para $i-1<1$, $i-2<1$, $j-1<1$, $j-2<1$

$i+1>N$, $i+2>N$, $j+1>M$ e $j+2>M$

temos $U_{i,j} = 0$

Em cada linha agora temos $(N/2+1)$ valores de $U_{i,j}$ (incluindo os pontos de contorno) e estes podem ser expressos como somas de $(N/2-1)$ coeficientes de seno:

$$U_{2i,j} = \sum_{k=1}^{N/2-1} \hat{U}_{k,j} \operatorname{sen}(2ik\pi/N) \quad 0 \leq i \leq N/2, \quad 0 \leq j \leq M$$

Substituindo na equação (22) obtemos $N/2-1$ sistemas pentadiagonais cada um dos quais podendo ser fatorado na forma

$$\hat{W}_{k,j-1} + \lambda_{N-k} \hat{W}_{k,j} + \hat{W}_{k,j+1} = \hat{b}^{(1)}_{k,j} \tag{23}$$

$$\hat{U}_{k,j-1} + \lambda_k \hat{U}_{k,j} + \hat{U}_{k,j+1} = \hat{W}_{k,j}$$

$$0 \leq k \leq N/2+1 \quad \text{e} \quad 0 \leq j \leq M$$

onde $\lambda_k = 2\cos(k\pi/N) - 4$ e

$$\hat{b}_{k,j} = (4/N) \sum_{i=1}^{N/2-1} b^{(1)}_{2i,j} \operatorname{sen}(2ik\pi/N) \tag{24}$$

$$0 \leq j \leq M$$

Donde o algoritmo prossegue: o lado direito da equação (22) é calculado em todos os pontos da grade com i par. Então a equação (24) é implementada, envolvendo (M-1) transformadas

de seno cada uma de comprimento N/2. A seguir os (N/2-1) sistemas de equações pentadiagonais (23), são resolvidos. Então são executadas (M-1) transformadas de seno inversas cada um de comprimento N/2 para obter $U_{2i,j}$ e finalmente $U_{i,j}$ para i ímpar é obtido resolvendo simples sistemas tridiagonais na direção j.

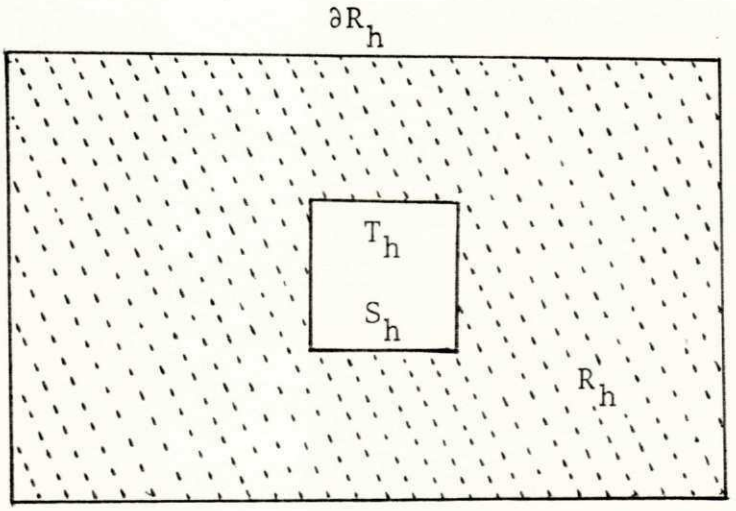
Para condições de contorno periódica em i=0,N pode ser derivado um algoritmo similar.

O número de coeficientes requerido para sistemas tridiagonais é o mesmo que para o método básico FFT. Novamente a solução pode ser reescrita e o único espaço de trabalho requerido é para FFT.

4.5 - Equação de Poisson em Regiões Irregulares

Quando as regiões não são retangulares elas são embutidas numa região retangular. Damos a seguir alguns exemplos de como resolver a equação.

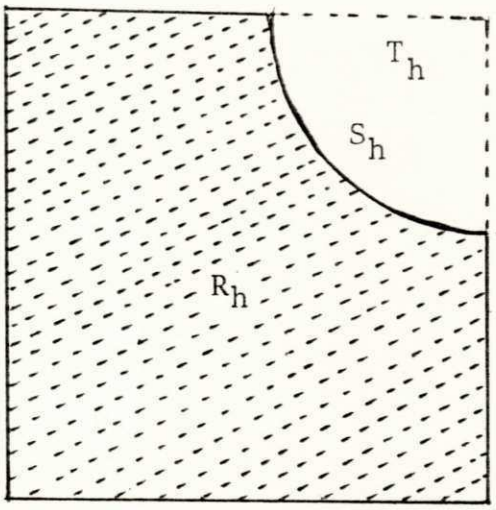
1 - Seja R uma região retangular com um retângulo interior removido, conforme figura



Para simplicidade assumimos que o contorno discreto ∂R_h é um subconjunto de ∂R. O retângulo embutido é $R'_h = R_h \cup S_h \cup T_h$. A extensão da função requer neste exemplo, que f seja definida (arbitrariamente) em $S_h \cup T_h$. Para definir esta extensão, podemos fixar $f \equiv 0$ em $S_h \cup T_h$ ou podemos definir f tal que seja contínua em todos os pontos de R_h.

Portanto, no exemplo consideramos a função f tal que $f \equiv 0$ em $S_h \cup T_h$, e a solução é encontrada através de um método qualquer.

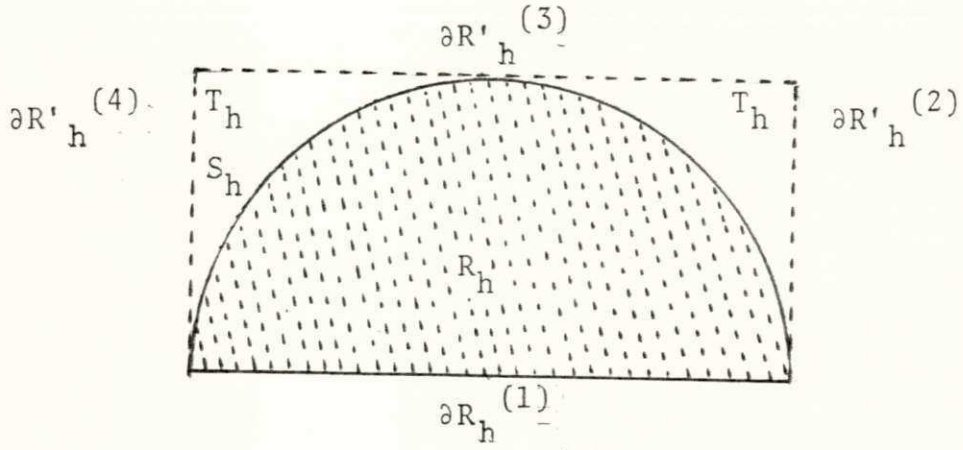
2 - Consideremos a região abaixo



Este problema aparece no estudo da dependência do tempo da rotação do fluido, e a superfície do fluido é lentamente movimentado.

A condição de contorno de Dirichlet é dada em S_h, e a condição de contorno de Neumann em $\partial R_h - S_h$. O retângulo embutido é $R'_h = R_h \cup S_h \cup T_h$, e usamos as condições de contorno de Neumann em $\partial R'_h$. Podemos usar o método direto para resolver o problema retangular de Neumann [9].

3 - Um outro exemplo é dado no estudo de dependência do tempo do plasma e, uma equação de Poisson pode ser resolvida a cada passo no tempo.



As condições de contorno são de Dirichlet em S_h e Neumann em $\partial R_h{}^{(1)}$. Nós usamos as condições de Neumann em $\partial R_h{}^{(1)}$ e $\partial R'_h{}^{(3)}$ e as condições de contorno de Dirichlet em $\partial R'_h{}^{(2)}$ e $R'_h{}^{(4)}$ na região embutida $R'_h = R_h \cup S_h \cup T_h$. A equação diferencial elíptica é então resolvida no retângulo R'_h.

CAPÍTULO V

TESTES E CONCLUSÕES

Iniciamos este último capítulo fazendo uma análise dos resultados que foram obtidos com a utilização das quatro variantes de Fast Poisson Solvers, bem como damos detalhes da programação de cada uma. Nas secções seguintes relacionamos dificuldades encontradas durante a implementação e apresentamos sugestões de tópicos para trabalhos posteriores.

Os programas que apresentamos foram escritos tomando-se por base uma região quadrada de lado 1, com a condição de contorno de Dirichlet. O caso da região retangular também foi testado mas não apresentamos o programa.

5.1 - Procedimento Adotado na Programação

Daremos a seguir como procedemos na elaboração dos programas de cada uma das variantes.

5.1.1 - Método Básico FFT

Programa Principal

Entrada: K - expoente de dois para calcular os valores de N e M.

VIX - limite inferior de X.

VFX - limite superior de X.

VIY - limite inferior de Y.

VFY - limite superior de Y.

Chama a subrotina DISFFT

Saída - solução da equação (W) em cada ponto da discretização.

Subrotina DISFFT

Esta subrotina resolve a equação de Poisson pelo Método Básico FFT.

Sequência de Chamada

CALL DISFFT (QUELL,CONT1,CONT2,K,VIX,VFX,VIX,VFY,W)

Parâmetros na chamada

K,VIX,VFX,VIY,VFY.

QUELL - função que avalia o 2º membro da equação de Poisson.

CONT1, CONT2 - avalia as condições de contorno, onde

CONT1: (0,y), (x,0),(x,b) e

CONT2: (a,y).

Retorno

W - um conjunto de tamanho NxM conterá uma aproximação da solução.

Subrotinas chamadas FFT, COSVEC, GAUBDS e FFTI.

Subrotina FFT

Esta subrotina calcula a transformada do seno.

Sequência de chamada

CALL FFT (K2,TR,TI)

Parâmetros de chamada

K2 - números de pontos.

TR - parte real do valor da função nos pontos da malha.

TI - parte imaginária do valor da função nos pontos da malha.

Retorno

TR, TI - contendo parte real e imaginária dos coeficientes.

Subrotina COSVEC

Esta subrotina calcula λ da equação

$$\hat{U}_{k,j-1} + \lambda_{k,j}\hat{U}_{k,j} + \hat{U}_{k,j+1} = b_{k,j}$$

Sequência de chamada

CALL COSVEC (R,COS2,J3)

Parâmetros na chamada

R - variável usada como expoente de 2 nos cálculos.

COS2 - vetor que conterá os λ.

J3 - número de linhas.

Retorno

COS2 - vetor contendo λ.

Subrotina RESOL

Esta subrotina resolve os sistemas de equações tridiagonais, utilizando o algoritmo modificado de Gauss (LU).

Sequência de chamada

CALL RESOL (COS2,B,J3)

Parâmetros na chamada

COS2 - vetor contendo λ.

B - matriz do sistema tridiagonal de ordem J3.

J3 - ordem da matriz.

Retorno

B - solução das equações tridiagonais.

Subrotina FFTI

Esta subrotina calcula a transformada inversa do seno.

Sequência de chamada

CALL FFTI (J3,TR,TI)

Parâmetros na chamada

J3 - número de elementos.

TR - parte real da solução dos sistemas de equações tridiagonais.

TI - parte imaginária da solução dos sistemas de equações tridiagonais.

Retorno

TR e TI - contendo parte real e imaginária da solução da equação em cada ponto da discretização.

5.1.2 - Algoritmo de Buneman [6]

Programa principal

Entrada: K - expoente de dois para calcular os valores de N e M

VIX - limite inferior de X.

VFX - limite superior de X.

VIY - limite inferior de Y.

VFY - limite superior de Y.

Chama a subrotina BUNEMA

Saída - solução da equação (W) em cada ponto da discretização.

Subrotina BUNEMA

Esta subrotina resolve a equação de Poisson utilizando o algoritmo de Buneman.

Sequência de chamada

CALL BUNEMA (QUELL,CONT1,CONT2,K,VIX,VFX,VIY,VFY,W)

Parâmetros na chamada

K,VIX,VFX,VIY,VFY

QUELL - (descrita em 5.1.1).

CONT1,CONT2 - (descrita em 5.1.1).

Retorno

W - um conjunto de tamanho NxM conterá uma aproximação da solução.

Subrotina chamada GLSAR

Subrotina GLSAR

Esta subrotina resolve o sistema $A^{(r)}X=B$ pela fatoração de $A^{(r)}$.

Sequência de chamada

CALL GLSAR (R,B,N)

Parâmetros na chamada

R - variável usada como expoente de 2 nos cálculos.

B - matriz do sistema tridiagonal.

N - ordem da matriz.

Retorno

B - solução da matriz tridiagonal

Subrotinas chamadas COSVEC e GAUBDS

Subrotina COSVEC (descrita em 5.1.1).

Subrotina GAUBDS (veja subrotina RESOL, 5.1.1).

5.1.3 - FACR(1,j)

Programa Principal

Entrada: K - expoente de dois (para calcular os valores de N e M)

XRI - limite inferior de X.

XRF - limite superior de X.

YRI - limite inferior de Y.

YRF - limite superior de Y.

Chama a subrotina FACR1

Saída: solução da equação (W) em cada ponto da discretização.

Subrotina FARC1

Esta subrotina resolve a equação de Poisson pelo algoritmo FACR (1,j) de Hockney.

Sequência de chamada

CALL FARC1 (QUELL,CONT1,CONT2,K,XRI,XRF,YRI,YRF,W)

Parâmetros na entrada

K,XRI,XRF,YRI,YRF

QUELL - (descrita em 5.1.1)

CONT1, CONT2 - (descrita em 5.1.1)

Retorno

W - um conjunto de tamanho NxM

Conterá uma aproximação da solução.

Subrotinas chamadas FFT, COSVEC, RESOL , FFTI

Subrotina FFT - (descrita em 5.1.1).

Subrotina COSVEC - (descrita em 5.1.1).

Subrotina RESOL - (descrita em 5.1.1).

Subrotina FFTI - (descrita em 5.1.1).

5.1.4 - FACR (1,i)

Programa Principal

Entrada: K - expoente de dois (para calcular os valores de N e M)

XRI - limite inferior de X .

XRF - limite superior de X .

YRI - limite inferior de Y.

YRF - limite superior de Y.

Chama a subrotina DIFARC

Saída: solução da equação (W) em cada ponto da discretização.

Subrotina DIFARC

Esta subrotina resolve a equação de Poisson pelo algoritmo FACR(1,i) de Hockney.

Sequência de chamada

CALL DIFARC (QUELL,CONT1,CONT2,K,XRI,XRF,YRI,YRF,W)

Parâmetros de entrada

K, XRI, YRF, YRI, YRF

QUELL - (descrito em 5.1.1).

CONT1,CONT2 - (descrito em 5.1.1).

Retorno

W - um conjunto de tamanho NxM. Conterá uma aproximação da solução subrotinas.

Chamadas FFT, COSVEC, RESOL, FFTI.

Subrotina FFT - (descrita em 5.1.1).

Subrotina COSVEC - (descrita em 5.1.1).

Subrotina RESOL - (descrita em 5.1.1).

Subrotina FFTI - (descrita em 5.1.1).

5.2 - Procedimentos Adotados Durante os Testes e Análise de Resultados Obtidos

As quatro variantes foram testadas no computador IBM 4341, cujo sistema operacional é o VM 370 controlador do OSV/SI sob o qual roda o compilador WATFIV (precisão simples).

Inicialmente testamos para regiões quadradas, $\Omega = (0,1)x(0,1)$ e $\Omega = (-1,1)x(-1,1)$, com as condições de contorno de Dirichlet $u(x,y) = 0$, $(x,y) \in \partial\Omega$. Os métodos FFT, FACR(1) e FACR(1,i) foram testados utilizando-se duas subrotinas FFT, uma fornecida pelo professor Hattori e a outra o algoritmo de Brigham [4]; a subrotina FFT mais eficiente foi a do professor Hattori.

Além das regiões quadradas testamos também para as regiões retangulares $\Omega = (0,1)x(0,2)$, fazendo pequenas

modificações nas subrotinas, e testando também o comportamento do erro.

Erro máximo

$$EM = \max_{i,j} |u-U_{i,j}|$$

Raiz quadrada da média dos quadrados (RMS)

$$RMS = \sqrt{\frac{1}{NxM} \Sigma (u-U_{i,j})^2}$$

NxM = número de pontos da malha

Equações usadas:

a) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -2\left[x(1-x)+y(1-y)\right]$ $\quad 0<x,y<1$

$u(x,y)=0$ no contorno

Solução exata: $u(x,y)=x(1-x)y(1-y)$

b) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = xe^y$ $\quad 0<x<2$, $0<y<1$

Condições de contorno

$u(0,y) = 0$, $u(2,y) = 2e^y$ $\quad 0\leq y\leq 1$

$u(x,y) = x$, $u(x,1) = ex$ $\quad 0\leq x\leq 2$

Solução exata: $u(x,y) = xe^y$

c) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 y}{\partial y^2} = -2 \qquad 0<x,\ y<1$

Condições de contorno

$u(0,y)=0\ ,\quad u(1,y) = \sinh(\pi x)\sin\pi y$

$u(x,0) = u(x,1) = x(1-x)$

Solução exata: $u(x,y)=\sinh(\pi x)\sin(\pi y)+x(1-x)$

d) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -\left[\cos(x+y)+\cos(x-y)\right]$

$0<x<\pi\ ,\ 0<y<\frac{\pi}{2}$

Condições de contorno

$u(0,y) = \cos y\ ,\ u(\pi,y) = -\cos y$

$u(x,0) = \cos x\ ,\ u(x,\frac{\pi}{2}) = 0$

Solução exata: $u(x,y) = \cos x \cos y$

e) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = (x^2+y^2)e^{xy} \qquad 0<x<2\ ,\ 0<y<1$

Condições de contorno

$u(0,y) = 1\ ,\quad u(2,y) = e^{2y}$

$u(x,0) = 1\ ,\quad u(x,1) = e^{x}$

Solução exata: $u(x,y) = e^{xy}$

f) $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -2\pi^2 \, sim(\pi x)\sin(\pi y)$ $\quad -1<x, \quad y<1$

$u(x,y) = 0$ no contorno

Solução exata: $u(x,y) = \sin(\pi x)\sin(\pi y)$

Damos a seguir os resultados obtidos nos testes da equação f.

Foram testadas também as equações a, b e c, cujo comportamento foi idêntico ao da equação f.

O tempo apresentado na tabela corresponde ao de execução. O cálculo dos erros implica em 20% de tempo adicional.

NxM		BÁSICO FFT	BUNEMAN	FACR (1,j)	FACR (1,i)
64	Tempo (seg)	0,80	0,32	0,54	0,47
	Erro max	0.996612	0.375467	0.088214	0.121004
	RMS	0.125000	0.119572	0.125000	0.125000
256	Tempo (seg)	2,36	0,83	1,41	1,02
	Erro max	0.999133	0.075138	0.069518	0.077881
	RMS	0.062500	0.018270	0.062500	0.062500
1024	Tempo (seg)	9,42	3,29	5,57	3,85
	Erro max	0.999782	0.011744	0.104261	0.066391
	RMS	0.031250	0.002554	0.031250	0.031250
4096	Tempo (seg)	38,50	14,33	22,66	15,75
	Erro max	0.999945	0.001646	0.310622	0.429586
	RMS	0.015625	0.000379	0.015625	0.015625
16384	Tempo (seg)	177,53	65,97	98,62	68,86
	Erro max	0.999734	0.000332	1.189671	0.996560
	RMS	0.007813	0.000175	0.007813	0.007813

Podemos observar que para equações lineares de Poisson, com as condições de contorno de Dirichlet, a variante mais eficiente é o algoritmo de Buneman e o menos eficiente é o método Básico FFT.

5.3 - **Dificuldades Encontradas**

As maiores dificuldades encontradas foram:

a) A quase não existência de algoritmo sobre as

variantes, o que nos levou a partir da teoria elaborar os algoritmos e testá-los.

b) A maioria da literatura traz a solução em regiões quadradas de comprimento 1, quando os comprimentos na grade na direção i e j, eram diferentes os algoritmos sofreram uma pequena modificação.

c) Não testamos para o número de pontos da malha maior que 16384, tendo em vista ser a memória do sistema insuficiente.

5.4 - Sugestões para Trabalhos Posteriores

Como utilizamos somente as equações diferenciais parciais lineares (equação de Poisson), na região retangular de duas dimensões, e com a condição de contorno de Dirichlet, utilizando 5 pontos, sugerimos os seguintes trabalhos.

a) O mesmo trabalho com a condição de contorno periódico ou de Neumann.

b) Fast Poisson Solvers para regiões não retangulares, utilizando 5 pontos [7].

c) Fast Poisson Solvers para equações a três dimensões, utilizando 9 pontos [1].

d) Fast Poisson Solvers para regiões não retangulares utilizando 9 pontos.

e) O marching Algorithm de Bank [2,3,9] merece ser estudada.

f) Esparsidade [1].

5.5 - Comentários Finais

Como o objetivo do trabalho era obter o melhor algoritmo Fast Poisson Solvers e, como o algoritmo de Buneman é o mais eficiente, optamos por incluí-lo na Biblioteca de Software Matemático, atualmente encontrado no IBM 4341 do Núcleo de Processamento de Dados do Campus II da Universidade Federal da Paraíba.

A utilização deste algoritmo como também da Biblioteca é de fácil aprendizagem, sendo que qualquer dúvida poderá ser solucionada pelo setor de usuário do Núcleo de Processamento de Dados.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Banegas, A., Fast Poisson for Problems With Sparsity Mathematics of Computation , Vol. 32, 1978, p. 441-446.

[2] Bank, E. Randolph and Donald J. Rose , Marching Algorithms for Elliptic Boundary Value Problems I: The Constant Coefficient Case. SIAM J. Numer. Anal. Vol. 14, 1977, p.792-829.

[3] Bank, E. Randolph., Marching Algorithmy for Elliptic Boundary Value Problems II: The Variable Coefficient Case , SIAM J. Numer.Anal. Vol. 14, 1977, p.950-969.

[4] Brigham, E. Oran., The Fast Fourier Transform., Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1974.

[5] Bulirsch, R. & J. Stoer , Introduction to Numerical Analysis., Springer-Verlag, Inc., New York, 1980.

[6] Buzbee, B.L., G.N. Golub and C.W. Nelson, On Direct Method for Solving Poisson's Equations, SIAM J. Numer. Anal. Vol. 7, 1970, p. 627-656.

[7] Buzbee, B.L., Dorr, F.W.,George, J.A. and Golub, G.H. The Direction Solution of The Discrete Poisson Equation on Irregular Regions., SIAM J. Numer. Anal., Vol. 28, 1971, p. 722-736.

[8] Buzbee, B.L. and Fred W. Dorr., The Direct Solution of The Biharmonic Equation on Retangular Regions. SIAM J. Numer. Anal., Vol. 11, 1974, p. 753-763.

[9] Ehrlich, Louis W., A Marching Thechnique for Nonseparable Equations, Mathematics of Computation, Vol. 33, 1979, p. 881-890.

[10] Figueiredo Djauro Guèdes de., Análise de Fourier e Equações Diferenciais Parciais , Editora Edgard Blücher, Ltda, S.Paulo, 1977.

[11] Fox, L., Finite Difference Methods for Elliptic Boundary - Value Problems - in D.A.H. Jacobs (ed.) The Stats of The Art in Numerical Analysis, Academic Press, London, 1977, p. 799-881.

[12] Hockney, R.W., The Potencial Calculation and Some Applications, Methods in Comp. Phys., Vol. 9, Academic Press, New York, 1970, p. 136-210.

[13] John F., Partial Differential Equations., Springer-Verlag, New York - Heidelberg - Berlin, Second Edition, 1975.

[14] Meis, Theodor and Marcowitz, Ulrich., Numerical Solution of Partial Differential Equations , Springer-Verlag , New York - Heidelberg - Berlin, 1978.

[15] Richtmyer , Robert D. and Morton, K.W., Difference Methods for Initial - Value Problems , Interscience Publishers, John Wiley & Sons, New York, 1967.

[16] Stephenson, G., Uma Introdução às Equações Diferenciais Parciais para Estudantes de Ciências , Edgard Blucher Ltda - Editora da Universidade de São Paulo, São Paulo, 1975.

[17] Temperton, Clive , Direct Methods for The Solution of the Discrete Poisson Equation , Journal of Computational Physics, 31, 1979, p. 1-20.

APÊNDICE I

1. Introdução

A transformada rápida de Fourier (FFT) está tendo, de uma maneira crescente, papel importante nas aplicações práticas da engenharia.

Não somente é aplicada na análise spectral, sonar, radar e detectação de vibrações, mas também na redução de faixas em transmissão de vídeo e filtragem de sinais. A FFT é usada diretamente para sinais filtrados e o domínio de frequência e indiretamente para projeto de filtros digitais.

É usado também para convoluções, evoluções e decomposição de sinais.

Como sabemos, o FFT é um método particular para resolver problemas que envolvem as séries de Fourier na computação.

Podemos dizer também que o FFT é simplesmente

um algoritmo que pode computar a transformada discreta de Fourier mais rapidamente do que diretamente a partir da formulação matemática.

Por esta razão o nosso estudo da FFT será unicamente no aspecto computacional do algoritmo.

1.1 - **Formulação Matricial**

Consideremos a transformada discreta de Fourier

$$X(n) = \sum_{k=0}^{N-1} x_0(k)\, e^{-j2\pi nk/N} \qquad n = 0,1,2,\ldots,N-1$$

Notamos que a transformada considerada descreve a computação de N equações.

Se tomarmos N = 4 e fazendo

$$W = e^{-j2\pi/N}$$

temos então

$$X(0) = x_0(0)W^0+x_0(1)W^0+x_0(2)W^0+x_0(3)W^0$$

$$X(1) = x_0(0)W^0+x_0(1)W^1+x_0(2)W^2+x_0(3)W^3$$

$$X(2) = x_0(0)W^0+x_0(1)W^2+x_0(2)W^4+x_0(3)W^6$$

$$X(3) = x_0(0)W^0+x_0(1)W^3+x_0(2)W^6+x_0(3)W^9$$

A equação acima pode ser escrita na forma matricial

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W^0 & W^0 & W^0 & W^0 \\ W0 & W^1 & W^2 & W^3 \\ W^0 & W^2 & W4 & W^6 \\ W^0 & W^3 & W^6 & W^9 \end{bmatrix} \begin{bmatrix} x_0(0) \\ x_0(1) \\ x_0(2) \\ x_0(3) \end{bmatrix} \qquad (1)$$

ou

$$\underline{X}(n) = \underline{W}^{nk}\ \underline{x}_0(k)$$

onde $\underline{W}$ e $\underline{x}_0(k)$ podem ser complexos e então serão necessárias N^2 multiplicações e (N)(N-1) adições para a computação de $\underline{X}(n)$.

O FFT deve seu sucesso ao fato de que o algoritmo reduz substancialmente o número de multiplicações e adições exigidas na computação, como a seguir veremos.

1.2 - Cálculo de X(n) Utilizando a Matriz Fatorada

Por conveniência escolhemos o número de pontos $x_o(k)$ tal que $N = 2^\gamma$ e $\gamma \in N$

Reescrevemos $X(n) = W^{nk}\ x_0(k)$ como

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W^1 & W^2 & W^3 \\ 1 & W^2 & W^0 & W^2 \\ 1 & W^3 & W^2 & W^1 \end{bmatrix} \begin{bmatrix} x_0(0) \\ x_0(1) \\ x_0(2) \\ x_0(3) \end{bmatrix} \qquad (2)$$

usando a relação

$$W^{nk} = W^{nk \bmod (N)}$$

Visto que nkmod(N) é o resto da divisão de nk por N.

Fatoração da Matriz W^{nk}

O método da fatoração é baseado na teoria do algoritmo FFT de Cooley & Tuckey.

Consideremos a transformada discreta de Fourier:

$$X(n) = \sum_{k=0}^{N-1} x_0(k) W^{nk} \qquad n = 0,\ldots,N-1 \qquad (3)$$

onde $W = e^{-j2\pi/N}$

Desejamos representar os inteiros n e k na forma binária, isto é, se tivermos N = 4, então $\gamma = 2$ e nós representamos k e n com 2 bits na forma binária

$k = 0,1,2,3$ ou $k = (k_1,k_0) = 00, 01, 10, 11$

$n = 0,1,2,3$ ou $n = (n_1,n_0) = 00, 01, 10, 11$

Escrevendo k e n na base 10 temos

$k = 2k_1 + k_0 \qquad n = 2n_1 + n_0$

onde k_1, k_0, n_1 e n_0 só podem ter os valores 0 e 1.

Utilizando a representação acima podemos escrever a transformada discreta de Fourier, para o caso de N=4, como:

$$X(n_1,n_0) = \sum_{k_0=0}^{1} \sum_{k_1=0}^{1} x_0(k_1,k_0) W^{(2n_1+n_0)(2k_1+k_0)} \qquad (4)$$

Consideremos agora termos W^P. Sendo

$W^{a+b} = W^a W^b$ então

$$W^{(2n_1+n_0)(2k_1+k_0)} = W^{(2n_1+n_0)2k_1} \; W^{(2n_1+n_0)k_0} =$$

$$= \left[W^{4n_1k_1} \right] W^{2n_0k_1} W^{(2n_1+n_0)k_0} = W^{2n_0k_1} W^{(2n_1+n_0)k_0}$$

Tendo em vista que $W^{4n_1k_1} = 1$, isto é,

$$W^{4n_1k_1} = \left[W^4 \right]^{n_1k_1} = \left[\exp\left[\left(\frac{-2j\pi 4}{4} \right) \right] \right]^{n_1k_1} = \left[1 \right]^{n_1k_1} = 1$$

Retomando a equação (4) temos:

$$X(n_1,n_0) = \sum_{k_0=0}^{1} \left[\sum_{k_1=0}^{1} x_0(k_1,1_0) W^{2n_0k_1} \right] W^{(2n_1+n_0)k_0} \qquad (5)$$

Consideremos agora cada uma das somatórias de (5) separadamente.

Primeiro a somatória

$$x_1(n_0,k_0) = \sum_{k_1=0}^{1} x_0(k_1,k_0) W^{2n_0k_1} \qquad (6)$$

As equações representadas são:

$$x_1(0,0) = x_0(0,0) + x_0(1,0)W^0$$

$$x_1(0,1) = x_0(0,1) + x_0(1,1)W^0$$

$$x_1(1,0) = x_0(0,0) + x_0(1,0)W^2$$

$$x_1(1,1) = x_0(0,1) + x_0(1,1)W^2$$

Escrevendo em forma de matriz temos:

$$\begin{bmatrix} x_1\ (0,0) \\ x_1\ (0,1) \\ x_1\ (1,0) \\ x_1\ (1,1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & W^0 & 0 \\ 0 & 1 & 0 & W^0 \\ 1 & 0 & W^2 & 0 \\ 0 & 1 & 0 & W^2 \end{bmatrix} \begin{bmatrix} x_0\ (0,0) \\ x_0\ (0,1) \\ x_0\ (1,0) \\ x_0\ (1,1) \end{bmatrix} \qquad (7)$$

Onde W^0 não é reduzido a unidade para melhor desenvolvimento generalizado do resultado.

Por último a somatória

$$x_2(n_0,n_1) = \sum_{k_0=0}^{1} x_1(n_0,k_0)W^{(2n_1+n_0)k_0} \qquad (8)$$

As equações representadas são:

$$x_2(0,0) = x_1(0,0) + x_1(0,1)W^0$$

$$x_2(0,1) = x_1(0,0) + x_1(0,1)W^2$$

$$x_2(1,0) = x_1(1,0) + x_1(1,1)W^1$$

$$x_2(1,1) = x_1(1,0) + x_1(1,1)W^3$$

Escrevendo na forma matricial temos:

$$\begin{bmatrix} x_2(0,0) \\ x_2(0,1) \\ x_2(1,0) \\ x_2(1,1) \end{bmatrix} = \begin{bmatrix} 1 & W^0 & 0 & 0 \\ 1 & W^2 & 0 & 0 \\ 0 & 0 & 1 & W^1 \\ 0 & 0 & 1 & W^3 \end{bmatrix} \begin{bmatrix} x_1(0,0) \\ x_1(0,1) \\ x_1(1,0) \\ x_1(1,1) \end{bmatrix} \qquad (9)$$

Notamos que (9) é exatamente a matriz fatorada da equação (1), com o índice k escrito na notação binária.

Das equações (5) e (8) temos:

$$X(n_1,n_0) = x_2(n_0,n_1) \qquad (10)$$

O resultado final $x_2(n_0,n_1)$ obtido da somatória

exterior são os bits em ordem reversa para os valores desejados

$X(n_1, n_0)$

Se combinarmos (6) (7) e (10)

$$x_1(n_0,k_0) = \sum_{k_1=0}^{1} x_0(k_1,k_0) W^{2n_0k_1}$$

$$x_2(n_0,n_1) = \sum_{k_0=0}^{1} x_1(n_0,k_0) W^{(2n_1+n_0)k_0} \qquad (11)$$

$$X(n_1,n_0) = x_2(n_0,n_1)$$

então o conjunto (11) representa a formulação original de Cooley-Tukey do algoritmo FFT para N=4. Então esta equação é recursiva, onde o segundo termo é novamente fatorado.

Após a fatoração temos:

$$\bar{X}(n) = \begin{bmatrix} X(0) \\ X(2) \\ X(1) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & W^0 & 0 & 0 \\ 1 & W^2 & 0 & 0 \\ 0 & 0 & 1 & W^1 \\ 0 & 0 & 1 & W^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & W^0 & 0 \\ 0 & 1 & 0 & W^0 \\ 1 & 0 & W^2 & 0 \\ 0 & 1 & 0 & W^2 \end{bmatrix} \begin{bmatrix} x_0(0) \\ x_0(1) \\ x_0(2) \\ x_0(3) \end{bmatrix}$$

Maiores detalhes em [4].

1.3 - Comparação Dois Dois Métodos

Podemos verificar que a computação de $\bar{X}(n)$ requer um total de 4 multiplicações e 8 adições, isto é, em (7)

$$x_1(0,0) = x_0(0,0) + W^0 x(1,0)$$

$$x_1(0,1) = x_0(0,1) + W^0 x(1,1)$$

$$x_1(1,0) = x_0(0,0) - W^0 x(1,0)$$

$$x_1(1,1) = x_0(0,1) - W^0 x(1,1)$$

e em (9)

$$x_2(0,0) = x_1(0,0) + W^0 x_1(0,1)$$

$$x_2(0,1) = x_1(0,0) - W^0 x(0,1)$$

$$x_2(1,0) = x_1(1,0) + W_1 x(0,1)$$

$$x_2(1,1) = x_1(1,0) - W_1 x(0,1)$$

tendo com vista que $W^2 = -W^0$ e $W^3 = -W^1$

A computação de X(n) requer 16 multiplicações e 12 adições. Notamos que o processo da fatoração reduz o número de multiplicações e adições, eis então a razão da eficiência do algoritmo FFT.

Se verificarmos o resultado do exemplo dado, notamos que o FFT requer $N\gamma/2$ multiplicações e $N\gamma$ adições, ao passo que o método direto requer N^2 multiplicações e $(N)(N-1)$ adições.

Se assumirmos que o tempo de computação é proporcional ao número de multiplicações, logo a relação aproximada entre o tempo de computação direta e pelo FFT é dada [4] por

$$\frac{N^2}{N\gamma/2} = \frac{2N}{\gamma}$$

Para $N = 1024 = 2^{10}$ obtém-se uma redução computacional de mais de 200 para um.