I.C.M.S.C.

INSTITUTO DE CIÊNCIAS MATEMÁTICAS DE SÃO CARLOS

SOLUÇÃO NUMÉRICA DE PROBLEMAS DE VALOR INICIAL POR MÉTODOS DE ADAMS

José Alberto Cuminato

UNIVERSIDADE DE SÃO PAULO

SÃO CARLOS - SÃO PAULO
BRASIL

INSTITUTO DE CIÊNCIAS MATEMÁTICAS
BIBLIOTECA
S. Carlos

SOLUÇÃO NUMÉRICA DE PROBLEMAS DE VALOR INICIAL POR MÉTODOS DE ADAMS

José Alberto Cuminato

Dissertação apresentada ao Instituto de Ciências Matemáticas de São Carlos, da Universidade de São Paulo, para obtenção do título de Mestre em Matemática.

Orientadora : Prof.ª Dr.ª Celia Maria Finazzi de Andrade

SÃO CARLOS
1981

Class. - T
Cutt. - C 969 s
v. 3
Tombo -

ABSTRACT

The main objective of this work is the study and implementation of Predictor-Corrector procedures based on the Adams methods for Initial Value Problems. These methods are used with variable order and variable step size.

Properties of convergence and stability concerning two different ways of changing the step size, namely: the variable step and interpolation techniques, are discussed.

The computer implementation of these techniques through STEP/DE and GEAR codes is analysed.

Numerical examples results with comments which illustrate the behavior of the mentioned codes are presented.

AGRADECIMENTOS

À professora Dr.ª Célia Maria Finazzi de Andrade pela orientação objetiva e constante e pela atenção a mim dedicadas.

Aos professores Laudecir Miguel Ferrari, João Carlos Vieira Sampaio e Messias Menegueti Júnior, pelo apoio humano e técnico a mim dispensados.

Ao amigo Edson Sardella pela compreensão e paciência.

Aos colegas do ICMSC-USP e professores da graduação pelo incentivo e pelos conhecimentos básicos necessários para a minha formação.

À FAPESP e CNPq pelo apoio financeiro sem o qual não seria possível a realização deste trabalho.

Enfim, a todas as pessoas que, direta ou indiretamente tiveram participação neste trabalho, em especial a Gisele Maria Saab que o datilografou.

I N D I C E

INTRODUÇÃO I

CAPÍTULO 0

PRELIMINARES 1

0.1. *Interpolação Polinomial* 1

0.2. *O Problema de Valor Inicial e a Discretização* 4

0.3. *O Problema Numérico* 7

CAPÍTULO I

APRESENTAÇÃO DOS MÉTODOS DE ADAMS 13

1.1. *Métodos de Adams* 13

1.2. *A Forma Matricial dos Métodos* PC 22

CAPÍTULO II

ESTABILIDADE E CONVERGÊNCIA 29

2.1. *Estabilidade e Convergência dos Métodos* PC *de Passo Variável* 30

2.2 *Estabilidade e Convergência dos Métodos* PC *com Passo e Ordem Variáveis* 53

CAPÍTULO III

IMPLEMENTAÇÃO DOS MÉTODOS DE ADAMS 63

3.1. *Implementação dos Métodos de Adams com a Técnica do Passo Variável* 64

3.2. *Implementação dos Métodos de Adams com a Técnica da Interpolação* 83

CAPÍTULO IV

ESTIMATIVAS E CONTROLE DO ERRO DE TRUNCAMENTO LOCAL 87

4.1. *Estimativas para o Erro Local de Ordem* k, k-1, k-2 ... 88

4.2. *Estimativa para o Erro Local de Ordem* k+1 97

CAPÍTULO V

ESTRATÉGIAS PARA SELEÇÃO DA ORDEM E DO PASSO. A FASE INICIAL DO PROCESSO. A INFLUÊNCIA DA REGIÃO DE ESTABILIDADE ABSOLUTA NESSAS ESTRATÉGIAS 107

5.1. *Argumentos em que se Baseiam as Seleções de Passo e Ordem nas Rotinas STEP/DE* 108

5.2. *Seleção da Ordem* 112

5.3. *Seleção do Passo* 117

5.4. *A Fase Inicial* 121

5.5. *Conseqüências das Estratégias de Seleção de Passo e Ordem* 123

5.6. *Argumentos em que se Baseiam as Seleções de Passo e Ordem na Rotina GEAR* 126

5.7. *Seleção do Passo* 127

5.8. *Seleção da Ordem e Fase Inicial* 130

5.9. *Influência da Região de Estratégias de Mudança do Passo e da Ordem nas Rotinas STEP e DE* 131

CAPÍTULO VI

RESULTADOS NUMÉRICOS 143

6.1. *Tempo de Processamento Quando Varia o Número de Equações no Sistema* 143

6.2. *Ilustração dos Teoremas de Estabilidade* 145

6.3. *Problemas com Singularidades em Derivadas de Ordem Baixa* 146

6.4. *Problemas "STIFF"* 150

BIBLIOGRAFIA 155

INTRODUÇÃO

Este trabalho estuda a implementação computacional dos métodos Previsor-Corretor com métodos de Adams de passo e ordem variáveis. Nos primeiros capítulos são estudadas algumas propriedades desses métodos com relação a duas técnicas de mudança de passo (Técnica do passo variável e técnica da interpolação).

Nos capítulos III, IV e V são analisadas as implementações dessas técnicas através das rotinas STEP/DE e GEAR.

Finalmente, no capítulo VI, são apresentados alguns exemplos numéricos e algumas conclusões que ilustram o comportamento das rotinas STEP/DE e GEAR e que podem servir de orientação para o usuário desses programas.

CAPÍTULO 0

PRELIMINARES

Neste capítulo daremos as definições e os resultados básicos usados no desenvolvimento deste trabalho.

Inicialmente trataremos da interpolação polinomial: existência e unicidade do polinômio interpolador, teorema do resto de interpolação, os polinômios interpoladores de Newton e de Lagrange. Trataremos também do problema de valor inicial (PVI): definição de PVI, teoremas de existência e unicidade de solução. Por fim, definiremos uma discretização do PVI, e introduziremos os métodos lineares de k-passos, os métodos previsor-corretor e o conceito de convergência dos métodos de passo múltiplo, dando os teoremas correspondentes.

0.1. *Interpolação Polinomial*

Sejam $f : [a,b] \to R$ uma função e $x_o, x_1, \ldots x_{k-1}$, k pontos distintos em $[a,b]$. Sob estas hipóteses temos o teorema.

<u>TEOREMA 0.1</u>

Existe um único polinômio de grau menor ou igual a $k-1$, denotado por $P_k(x)$ tal que, $P_k(x_i) = f_i$, $i = 0,1,\ldots k-1$, onde $f_i = f(x_i)$.

<u>Definição 0.1</u> Ao polinômio $P_k(x)$ do teorema 0.1 chamaremos polinômio interpolador da função f sobre os pontos $x_o, x_1, \ldots, \ldots, x_{k-1}$.

O polinômio interpolador pode ser considerado como uma aproximação para a função f sobre o intervalo $[a,b]$ e o teorema seguinte fornece a expressão para o erro de truncamento correspondente.

<u>TEOREMA 0.2</u>

Sejam $f : [a,b] \to R$, $f \in C^k[a,b]$, $x_o, x_1, \ldots, x_{k-1}$ pontos distintos de $[a,b]$ e $w(x) = (x-x_o)(x-x_1)\ldots(x-x_{k-1})$. Então para cada $x \in [a,b]$ vale,

$$f(x) - P_k(x) = E(x)w(x) \quad \text{onde}$$

$$E(x) = \begin{cases} \dfrac{f'(x_i)-p_k'(x_i)}{w'(x_i)} & \text{se } x = x_i \\[2ex] \dfrac{f^{(k)}(\xi(x))}{k!} & \text{se } x \neq x_i,\ \xi(x) \in [a,b]. \end{cases}$$

<u>Definição 0.2</u> Sejam $f : [a,b] \to R$, $x_o, x_1, \ldots, x_{k-1}$ pontos distintos de $[a,b]$. Definimos a diferença dividida de ordem $i \geq 1$ de f com respeito aos $i+1$ pontos $x_j, x_{j+1}, \ldots, x_{j+i}$, e anotamos $f[x_j, x_{j+1}, \ldots, x_{j+i}]$, por,

$$f[x_j,\ldots,x_{j+i}] = \frac{f[x_j,\ldots,x_{j+i-1}]-f[x_{j+1},\ldots,x_{j+i}]}{x_j - x_{j+i}}$$

e para $i = 0$ definimos a diferença dividida de ordem 0 em x_j por $f[x_j] = f_j$.

LEMA 0.1

A diferença dividida de ordem i é simétrica em relação aos argumentos, isto é, $f[x_j,x_{j+1},\ldots,x_{j+i}]=f[x_{j_o},x_{j_1},\ldots,x_{j_i}]$ onde $(j_o,j_1,\ldots,j_i)$ é uma permutação de $(j,j+1,\ldots,j+i)$.

Definição 0.3 Definimos a diferença descendente de ordem i de f no ponto x_j, $0 \le j \le k-1$ e indicamos por $\nabla^i f_j$, como

$$\nabla^i f_j = \nabla^{i-1} f_j - \nabla^{i-1} f_{j-1} \qquad \text{se } i \ge 1$$

$$\nabla^o f_j = f_j.$$

Caso os pontos $x_o,x_1,\ldots,x_{k-1}$, sejam igualmente espaçados, isto é, $x_i = x_o + ih$, para $i = 0,1,2,\ldots,k-1$ podemos relacionar diferenças divididas com diferenças descendentes como mostra o lema.

LEMA 0.2

Se os pontos $x_o,x_1,\ldots,x_{k-1}$ são igualmente espaçados então vale:

$$(0.1)\quad \nabla^i f_j = i!h^i f[x_j, x_{j-1}, \ldots, x_{j-i}]$$

O polinômio interpolador de f sobre os pontos $x_o, x_1, \ldots, x_{k-1}$ pode ser colocado sob a forma de Newton

$$(0.2)\quad P_k(x) = f[x_o] + (x-x_o)f[x_o,x_1] + \ldots$$

$$\ldots + (x-x_o)(x-x_1)\ldots(x-x_{k-2})f[x_o,x_1,\ldots,x_{k-1}]$$

ou sob a forma de Lagrange

$$(0.3)\quad P_k(x) = \sum_{i=0}^{k-1} \frac{w(x)}{w'(x_i)(x-x_i)} f_i .$$

Quando os pontos são igualmente espaçados, devido à fórmula (0.1), a fórmula (0.2) se transforma em:

$$(0.4)\quad P_k(x) = \nabla^o f_o + \frac{(x-x_o)\nabla^1 f_o}{h\ 1!} + \frac{(x-x_o)(x-x_1)\nabla^2 f_o}{h^2\ 2!} +$$

$$+ \ldots + \frac{(x-x_o)(x-x_1)\ldots(x-x_{k-2})\nabla^{k-1} f_o}{h^{k-1}(k-1)!}$$

que é a fórmula do polinômio interpolador com diferenças descendentes.

0.2. *O Problema de Valor Inicial e a Discretização*

<u>Definição 0.4</u> Sejam U um aberto do R^n, $f : [a,b] \times U \to R^n$ uma função. O problema

$$(0.5)\quad \begin{aligned} y' &= f(x,y) \\ y(a) &= A \end{aligned}$$

é dito problema de valor inicial para uma equação diferencial de primeira ordem (PVI).

<u>Definição 0.5</u> Uma função $y : [a,b] \to R^n$ é dita ser uma solução do PVI (0.5) se:

1) $(x,y(x)) \in [a,b] \times U$
2) $y \in C^1[a,b]$
3) $y' = f(x,y)$ e $y(a) = A$.

O problema (0.5) pode ser transformado por integração no seguinte problema equivalente:

$$(0.6) \quad y(x) = A + \int_a^x f(t,y(t))dt.$$

Com relação à solução do problema (0.5) ou (0.6) temos os resultados:

<u>TEOREMA 0.3</u> (EXISTÊNCIA)

Sejam U aberto do R^n, $f : [a,b] \times U \to R^n$ contínua. Então o PVI (0.5) tem pelo menos uma solução.

Prova: ver [7, p. 10]

<u>TEOREMA 0.4</u>

Sejam U aberto de R^n, $f : [a,b] \times U \to R^n$, $g : [a,b] \times U \to R^n$ funções contínuas. Suponhamos que exista

uma constante L, tal que:

$$|f(x,y) - f(x,y^*)| \leq L|y-y^*| \qquad \forall\, y,y^* \in U.$$

Suponhamos ainda que para um número fixo ε vale

$$|f(x,y) - g(x,y)| \leq \varepsilon, \quad \forall (x,y) \in [a,b] \times U.$$

Sejam $u,v : [a,b] \to R^n$ soluções dos problemas de valor inicial

$$u'(x) = f(x,u(x)) \;, \qquad v'(x) = g(x,v(x))$$

$$u(a) = A_o \qquad v(a) = A_1.$$

Então

$$(0.7) \quad |u(x) - v(x)| \leq [|A_o - A_1| + (b-a)\varepsilon]\exp(L(x-a)).$$

Prova: ver [16, p. 8]

<u>TEOREMA 0.5</u> (EXISTÊNCIA E UNICIDADE)

Sejam U aberto do R^n, $f : [a,b] \times U \to R^n$, f y-Lipschitziana, isto é, existe uma constante L, tal que:

$$|f(x,y)-f(x,y^*)| \leq L|y-y^*|, \quad \forall\, x \in [a,b], \forall\, y,y^* \in U.$$

Então o PVI (0.5) tem uma única solução.

Prova: ver [16, p. 10]

0.3. *O Problema Numérico*

Em muitos casos é difícil, quando não impossível, determinar a solução analítica explícita de (0.5). Nestes casos podemos estar interessados em conhecer os valores da função $y(x)$ solução de (0.5) em certos pontos $x_o, x_1, \ldots, x_N$ de $[a,b]$.

<u>Definição 0.6</u> Sejam $a = x_o < x_1 < \ldots < x_N = b$.

Uma discretização do problema (0.5) é definida como o problema de determinar os valores $y(x_1), y(x_2), \ldots, y(x_N)$.

Na maioria das vezes não podemos calcular os valores $y(x_1), \ldots, y(x_N)$ exatamente, o que fazemos é aproximá-los por certos valores, com um erro aceitável. Precisamos assim de uma maneira de calcular tais aproximações, à qual chamaremos método de discretização da equação diferencial, o qual denominamos simplesmente por método. Um método simples e talvez comum é o método da série de Taylor. Se a função y for suficientemente diferenciável temos, para $n = 1,2,\ldots,N$.

$$(0.8) \qquad y(x_n) = y(x_{n-1}) + (x_n - x_{n-1})y'(x_{n-1}) + \frac{(x-x_{n-1})^2 y''(x_{n-1})}{2!} + \ldots + \frac{(x-x_{n-1})^s y^{(s)}(x_{n-1})}{s!} + \frac{y^{(s+1)}(\xi)}{(s+1)!}, \quad \xi \in (x_{n-1}, x_n).$$

Denotando por $y_i^{(r)}$ a aproximação para $y^{(r)}(x_i)$, $r = 0,1,\ldots,$ se desprezamos o termo do resto em (0.8) obtemos:

$$y_n = y_{n-1} + (x_n - x_{n-1})y_{n-1}^{(1)} + \ldots + (x_n - x_{n-1})^s y_{n-1}^{(s)}$$

que constitui o método numérico.

É claro que podemos calcular as aproximações para $y(x_1),\ldots,y(x_N)$ por outros métodos que não os baseados na série de Taylor. Os métodos que usamos neste trabalho são os chamados métodos lineares de passo múltiplo (MLPM), definidos por:

<u>Definição 0.7</u> Um método linear de passo múltiplo de k-passos é definido pela fórmula;

$$(0.9)\quad y_n + \alpha_{k-1}y_{n-1} + \ldots + \alpha_o y_{n-k} = h[\beta_k f_n + \ldots + \beta_o f_{n-k}]$$

para $n = k, k+1,\ldots,N$, onde $f_i = f(x_i,y_i)$, e α_i,β_i são constantes tais que $|\alpha_o|+|\beta_o| \neq 0$.

Como vemos, para que possamos aplicar tais métodos é necessário conhecer as proximações iniciais $y_o,\ldots,y_{k-1}$, que são os valores iniciais para o método (0.9). Na fórmula (0.9), se $\beta_k \neq 0$ temos o que chamamos de método implícito, se $\beta_k = 0$ o método será dito explícito.

Se $\beta_k \neq 0$ e a função f for não linear, para determinar o valor de y_n teremos que resolver um sistema, em geral não linear. Podemos fazê-lo usando técnicas apropriadas entre as quais destacamos:

1) o processo de Newton-Raphson
2) O processo das aproximações sucessivas.

Enquadram-se no segundo caso os métodos previsor-corretor. Lembramos que, para que haja convergência, é preciso que o problema satisfaça condições especiais, ver [13, p. 86].

<u>Definição 0.8</u> Os métodos previsor-corretor (PC) são definidos pelo algorítmo:

Para $n = k, k+1, \ldots, N$

(0.10)

a) Obtemos por um método explícito uma primeira aproximação para y_n e f_n, isto é, calculamos:

$$y_n^{[o]} = - \left[\sum_{j=1}^{k} \alpha_{k-j} y_{n-j} - h\beta_{k-j} f_{n-j}\right] \qquad \text{(MÉTODO PREVISOR)}$$

b) Para $m = 0,1,2,\ldots,r-1$ avaliamos a função f

$f_n^{[m]} = f(x_n, y_n^{[m]})$ e então "corrigimos" o valor de y_n segundo

$$y_n^{[m+1]} = - \left[\sum_{j=1}^{k} \alpha^*_{k-j} y_{n-j} - h\beta^*_{k-j} f_{n-j}\right] + h\beta^*_k f_n^{[m]}$$

(MÉTODO CORRETOR)

c) Avaliamos $f_n = f(x_n, y_n^{[r]})$.

O método (0.10) é aplicado no modo $P(EC)^rE$, isto porque corrigimos o valor r vezes e aceitamos como aproximação para $y(x_n)$ o valor obtido no último estágio. Para maiores detalhes à respeito dos métodos previsor-corretor ver [13, p. 85].

Como já dissemos anteriormente, estamos interessados em determinar aproximações com um erro aceitável; com esse objetivo

daremos à seguir definições e faremos hipóteses sobre os coeficientes do método (0.9).

<u>Definição 0.9</u> O método (0.9) é dito ser convergente se para todo PVI com as hipóteses do teorema 0.5 vale

$$\lim_{\substack{n\to\infty \\ nh=x-a}} y_n = y(x) \quad \text{para todo} \quad x \in [a,b],$$

quando os valores iniciais que produzem a solução y_n são exatos.

<u>Definição 0.10</u> Sejam y uma função de classe C^1, $h \in R$; definimos o operador L associado com o método (0.9) por

$$(0.11) \qquad L[y(x),h] = \sum_{j=0}^{k} [\alpha_{k-j} y(x-jh) - h\beta_{k-j} y'(x-jh)].$$

Se y for suficientemente diferenciável, desenvolvendo o segundo membro de (0.11) em série de Taylor em torno do ponto x temos:

$$(0.12) \qquad L[y(x),h] = C_o y(x) + C_1 h y'(x) + \ldots + C_q h^q y^{(q)}(x).$$

<u>Definição 0.11</u> O operador L e o método (0.9) associado têm ordem p se $C_o = \ldots = C_p = 0$ e $C_{p+1} \neq 0$.

<u>Definição 0.12</u> O erro de truncamento local do método (0.9) em x_n é definido por:

$L[y(x_{n-k}),h]$, onde $y(x)$ é a solução exata do PVI.

Notemos que se o método (0.9) tem ordem p, o erro de truncamento local deste método será;

$$L[y(x_{n-k}),h] = C_{p+1}h^{p+1}y^{(p+1)}(x_{n-k}) + \ldots$$

Neste caso o erro de truncamento local é de ordem h^{p+1} e indicamos $0(h^{p+1})$.

<u>Definição 0.13</u> Diremos que o método (0.9) é consistente se tem ordem $p \geq 1$.

<u>Definição 0.14</u> Sejam

$$y_n = -\sum_{j=1}^{k}\alpha_{k-j}y_{n-j} + h\sum_{j=0}^{k}\beta_{k-j}f_{n-j} \quad \text{e}$$

$$\tilde{y}_n = -\sum_{j=1}^{k}\alpha_{k-j}\tilde{y}_{n-j} + h\sum_{j=0}^{k}\beta_{k-j}\tilde{f}_{n-j} + r_{n-1}$$

onde r_{n-1} é uma perturbação. O método é dito ser estável se, dado $\varepsilon > 0$, existe $\delta = \delta(\varepsilon)$ tal que

$$\text{se } \sum_{j=0}^{n-1}|r_j| < \delta(\varepsilon) \quad \text{então} \quad |y_n - \tilde{y}_n| < \varepsilon.$$

<u>TEOREMA 0.6</u>

O método (0.9) é convergente se e somente se ele for consistente e estável.

Prova: ver [8, p. 224 e 246]

O fato de um método ser convergente é uma condição básica para que as soluções produzidas por ele sejam aceitáveis. Lembramos porém que em certos casos a propagação dos erros acumulados pode tornar as soluções totalmente irreais, mesmo o método sendo convergente segundo a definição 0.9. Para que tenhamos um controle sobre a propagação do erro daremos a definição.

<u>Definição 0.15</u> Sejam

$$\rho(\xi) = \xi^k + \sum_{j=1}^{k} \alpha_{k-j}\xi^j \quad \text{e} \quad \sigma(\xi) = \sum_{j=0}^{k} \beta_{k-j}\xi^j.$$

O método (0.9) é dito ser absolutamente estável para um dado $\bar{h} \in C$, se as raízes do polinômio

$\pi(\xi,\bar{h}) = \rho(\xi) - \bar{h}\sigma(\xi)$ estão no interior do disco unitário.

O fato de um método ser absolutamente estável para um conjunto maior de $\bar{h} \in C$, significa que este método produzirá aproximações viáveis para um maior número de PVI.

CAPÍTULO I

APRESENTAÇÃO DOS MÉTODOS DE ADAMS

Apresentaremos os métodos de Adams, que são obtidos por interpolação polinomial, sobre um conjunto de pontos não necessariamente igualmente espaçados. Daremos as fórmulas para a construção dos coeficientes destes métodos bem como limitantes para o erro de truncamento local. Estes estudos foram realizados principalmente por Shampine [16].

Introduziremos também os métodos previsor-corretor numa notação matricial, a qual foi introduzida por Nordsieck em [14] e posteriormente utilizada por Gear em [3].

Definiremos métodos equivalentes e daremos a forma de Nordsieck para métodos previsor-corretor com métodos de Adams.

1.1. *Métodos de Adams*

Consideremos o PVI (0.5) e uma discretização de [a,b] pelos pontos $x_o, x_1, \ldots, x_N$ tais que

$$x_o = a \quad e \quad x_n = x_{n-1} + h_n, \quad n = 1,2,\ldots,N.$$

Desejamos continuar a solução do problema discretizado a partir de x_n para x_{n+1}, conhecendo as soluções em $x_o, x_1, \ldots, x_n$.

Pela fórmula (0.6) a solução do PVI em x_{n+1} pode ser escrita na forma

$$(1.1)\quad y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(t,y(t))dt.$$

Um método de Adams consiste em aproximar a função $f(t,y(t))$, pelo seu polinômio interpolador sobre pontos anteriores, e então integrar tal polinômio obtendo assim uma aproximação para $y(x_{n+1})$.

A fórmula de Adams-Bashforth de ordem k usa o polinômio interpolador $P_{k,n}(x)$ de f sobre k pontos $x_n, x_{n-1}, \ldots, x_{n+1-k}$, isto é,

$$(1.2)\quad P_{k,n}(x_{n+1-i}) = f_{n+1-i}, \qquad i = 1,2,\ldots,k$$

Assim uma aproximação para $y(x_{n+1})$ é dada por:

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} P_{k,n}(t)dt.$$

Usando a forma de Lagrange para $P_{k,n}(t)$, obtemos:

$$(1.3)\quad y_{n+1} = y_n + \sum_{i=1}^{k} f_{n+1-i} \int_{x_n}^{x_{n+1}} \frac{w(t)}{(t-x_{n+1-i})w'(x_{n+1-i})}dt =$$

$$= y_n + h_{n+1} \sum_{i=1}^{k} \beta_{i,k} f_{n+1-i}$$

onde $\beta_{i,k} = \dfrac{1}{h_{n+1}} \displaystyle\int_{x_n}^{x_{n+1}} \frac{w(t)}{(t-x_{n+1-i})w'(x_{n+1-i})}dt$

ou, se fizermos a mudança de variável $s = (t-x_n)/h_{n+1}$, temos:

$$\beta_{i,k} = \int_0^1 \frac{w(x_n+sh_{n+1})}{(x_n+sh_{n+1}-x_{n+1-i})w'(x_{n+1-i})}ds.$$

Quando os pontos são igualmente espaçados podemos usar a forma (0.4) para $P_{k,n}(x)$ e encontrar a fórmula de Adams--Bashforth com tamanho de passo constante h:

$$(1.4) \quad y_{n+1} = y_n + h \sum_{i=1}^{k} \gamma_{i-1} \nabla^{i-1} f_n$$

onde
$$\gamma_i = \begin{cases} 1, & i = 0 \\ \dfrac{1}{i!h} \displaystyle\int_0^s s(s+1)\ldots(s+i-1)\,ds, & i \geq 1. \end{cases}$$

Vemos então que os métodos acima construídos são do tipo (0.9), isto é, métodos lineares de passo múltiplo com k-passos.

A aproximação y_{n+1} em (1.3) tem dois tipos de erro, um devido a aproximação de $f(t,y(t))$ por um polinômio, definido no capítulo 0 como erro de truncamento local, e outro devido ao uso, no polinômio interpolador, de aproximações f_{n+1-j} para os valores de $f(x_{n+1-j}, y(x_{n+1-j}))$.

Seja $\tilde{P}_{k,n}(x)$ o polinômio tal que:

$$\tilde{P}_{k,n}(x_{n+1-j}) = f(x_{n+1-j}, y(x_{n+1-j})), \quad j = 1,2,\ldots,k$$

onde $y(x_{n+1-j})$ é o valor exato da solução $y(x)$ em x_{n+1-j}. Assim o erro de truncamento local para os métodos de Adams--Bashforth em x_{n+1}, que será indicado por σ^p_{n+1} é tal que:

$$(1.5) \quad y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} \tilde{P}_{k,n}(t)\,dt + \sigma^p_{n+1}.$$

Fazendo a diferença membro a membro entre as expressões (1.1) e (1.5) encontramos

$$(1.6)\quad \sigma^{p}_{n+1} = \int_{x_n}^{x_{n+1}} (f(t,y(t)) - \tilde{P}_{k,n}(t))dt =$$

$$= \int_{x_n}^{x_{n+1}} (y'(t) - \tilde{P}_{k,n}(t))dt .$$

Se a função $y(x) \in C^{k+1}[a,b]$, a aplicação do teorema 0.2 ao integrando nos permite escrever:

$$\sigma^{p}_{n+1} = \int_{x_n}^{x_{n+1}} E(t) \prod_{i=1}^{k} (t-x_{n+1-i})dt$$

que, pela aplicação do teorema do valor médio para integrais se transforma em:

$$\sigma^{p}_{n+1} = E(\eta) \int_{x_n}^{x_{n+1}} \prod_{i=1}^{k} (t-x_{n+1-i})dt$$

com $\eta \in (x_n, x_{n+1})$. Ainda pelo teorema 0.2 vale que

$E(\eta) = y^{(k+1)}(\xi)/k!$, $\xi \in (x_n, x_{n+1})$ e portanto:

$$(1.7)\quad \sigma^{p}_{n+1} = \frac{y^{(k+1)}(\xi)}{k!} \int_{x_n}^{x_{n+1}} \prod_{i=1}^{k} (t-x_{n+1-i})dt .$$

Seja $H = MAX\{h_{n+2-k},\ldots,h_n\}$ então

$$\sigma^p_{n+1} = \frac{y^{(k+1)}(\xi)}{k!} H^{k+1} \int_{x_n}^{x_{n+1}} \frac{(t-x_{n+1-k})}{H} \cdot \frac{(t-x_{n+2-k})}{H} \cdots$$

$$\cdots \frac{(t-x_n)}{H} \cdot \frac{dt}{H}$$

$$(1.8) \qquad |\sigma^p_{n+1}| \leq \left| \frac{y^{(k+1)}(\xi)}{k!} \right| H^{k+1} \int_{x_n}^{x_{n+1}} \left(\prod_{i=1}^{k} i\right) \frac{dt}{H} \leq$$

$$\leq |y^{(k+1)}(\xi)| H^{k+1}.$$

Vemos então que o erro de truncamento local dos métodos de Adams-Bashforth de k-passos é de ordem k+1, isto é

$$\sigma^p_{n+1} = 0(H^{k+1}).$$

Se em (1.7) os pontos forem igualmente espaçados, fazendo a mudança de variável $s = (t-x_n)/h$ temos:

$$(1.9) \qquad \sigma^p_{n+1} = \frac{y^{(k+1)}(\xi)}{k!} h^{k+1} \int_0^1 \prod_{i=1}^{k} (s+1-i)\, ds =$$

$$= \gamma_k h^{k+1} y^{(k+1)}(\xi).$$

Este fato mostra que estes métodos têm ordem k segundo a definição 0.11.

Notemos que na expressão (1.9) para avaliar σ^p_{n+1} precisamos do valor de $y^{(k+1)}(\xi)$ o qual não conhecemos; assim

é necessário desenvolver outra estratégia para a avaliação do erro de truncamento local, a qual se baseia na idéia de comparar os resultados obtidos usando o polinômio interpolador $\tilde{P}_{k,n}(x)$ com os obtidos usando o polinômio interpolador $\tilde{P}_{k+1,n}(x)$ que interpola f sobre os $k+1$ pontos $x_n, x_{n-1}, \ldots, x_{n-k}$.

Assim usando (1.6) temos

$$(1.10) \quad \sigma^p_{n+1} = \int_{x_n}^{x_{n+1}} (y'(t) - \tilde{P}_{k+1,n}(t))dt \; +$$

$$+ \int_{x_n}^{x_{n+1}} (\tilde{P}_{k+1,n}(t) - \tilde{P}_{k,n}(t))dt.$$

Agora $\int_{x_n}^{x_{n+1}} (y'(t) - \tilde{P}_{k+1,n}(t))dt = 0(H^{k+2})$

conforme (1.8), portanto (1.10) torna-se

$$(1.11) \quad \sigma^p_{n+1} = \int_{x_n}^{x_{n+1}} (\tilde{P}_{k+1,n}(t) - \tilde{P}_{k,n}(t))dt + 0(H^{k+2}).$$

Se admitimos que $y_{n+1-i} = y(x_{n+1-i})$, então $\tilde{P}_{k,n}(t) = P_{k,n}(t)$ e $\tilde{P}_{k+1,n}(t) = P_{k+1,n}(t)$, e com estas hipóteses temos:

$$\sigma^p_{n+1} = \int_{x_n}^{x_{n+1}} (P_{k+1,n}(t) - P_{k,n}(t))dt + 0(H^{k+2}).$$

Se os pontos são igualmente espaçados e se usamos para os polinômios $P_{k,n}(x)$ e $P_{k+1,n}(x)$ as fórmulas com diferenças descendentes (0.4) obtemos:

$$(1.12) \quad \sigma^{p}_{n+1} = h\gamma_k \nabla^k f_n + 0(h^{k+2}) \simeq h\gamma_k \nabla^k f_n .$$

Na expressão acima os valores de γ_k e $\nabla^k f_n$ são calculáveis desde que tenhamos armazenado valores f_i suficientes, no passo anterior. Temos assim uma estimativa prática para o erro de truncamento local.

Até aqui desenvolvemos uma aproximação para $y(x_{n+1})$ interpolando a função f sobre os pontos $x_n, x_{n+1}, \ldots, x_{n+1-k}$, dando portanto um método de passo múltiplo explícito. É claro que podemos desenvolver uma aproximação para $y(x_{n+1})$ interpolando f sobre os k pontos $x_{n+1}, x_n, \ldots, x_{n+2-k}$ por um polinômio que indicaremos por $P^*_{k,n}(x)$. Temos assim um método de passo múltiplo implícito. Para aplicá-lo precisamos obter uma aproximação inicial para $y(x_{n+1})$, que indicaremos por p_{n+1} e recorremos ao método explícito para calculá-la. Nestas condições o polinômio $P^*_{k,n}(x)$ é tal que:

$$(1.13) \quad P^*_{k,n}(x_{n+1-i}) = f_{n+1-i}, \quad i = 1,2,\ldots,k-1$$

$$P^*_{k,n}(x_{n+1}) = f(x_{n+1}, p_{n+1}).$$

A aproximação y_{n+1} será então dada por:

$$(1.14) \quad y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} P^*_{k,n}(t)\,dt$$

ou, usando para $P^*_{k,n}(t)$ a forma de Lagrange

$$(1.15) \quad y_{n+1} = y_n + h_{n+1} \sum_{i=1}^{k-1} \beta^*_{i,k} f_{n+1-i} + h_{n+1} \beta^*_{i_o,k} f(x_{n+1}, p_{n+1})$$

onde $\beta^*_{i,k} = \int_o^1 \frac{w(x_n + sh_{n+1})}{(x_n + sh_{n+1} - x_{n+1-i}) w'(x_{n+1-i})} ds, \quad i=0,1,\ldots k-1$

e

$$w(x) = (x - x_{n+1}) \ldots (x - x_{n+2-k}).$$

Se os pontos são igualmente espaçados y_{n+1} fica

$$y_{n+1} = y_n + h \sum_{i=1}^{k} \gamma^*_{i-1} \nabla^{i-1} f^p_{n+1}$$

onde

$$\gamma^*_i = \begin{cases} 1 & \text{se } i = 0 \\ \frac{1}{i!} \int_o^1 (s-1)s \ldots (s+i-2)\,ds & i \geq 1 \end{cases}$$

e

$$f^p_{n+1} = f(x_{n+1}, p_{n+1}).$$

Shampine prova em [16 p. 301] que $\gamma_i^* = \gamma_i - \gamma_{i-1}$, $i \geq 1$.

Os métodos de Adams implícitos são chamados métodos de Adams-Moulton. O erro de truncamento local desses métodos em x_n será indicado por σ_{n+1}. Exatamente como fizemos para os métodos explícitos podemos provar aqui que $\sigma_{n+1} = 0(H^{k+1})$ e que, para pontos igualmente espaçados

$$(1.16) \quad \sigma_{n+1} = \gamma_k^* h^{k+1} y^{(k+1)}(\tau) \ , \quad \tau \in (x_n, x_{n+1}).$$

Neste último caso uma estimativa prática para σ_{n+1} é dada por:

$$(1.17) \quad \sigma_{n+1} \simeq h\gamma_k^* \nabla^k f_{n+1}^p.$$

A expressão (1.16) nos diz que um método de Adams--Moulton de k-passos tem ordem k segundo a definição 0.11.

Como fizemos no capítulo 0, podemos definir os métodos previsor-corretor usando os métodos de Adams com passo variável. A referência [16] estuda somente o caso $r = 1$ ou seja o modo PECE, que consiste em

1) calcular o valor previsto, p_{n+1}

2) avaliar a função $f_{n+1}^p = f(x_{n+1}, p_{n+1})$

3) calcular o valor corrigido y_{n+1}

4) avaliar a função $f_{n+1} = f(x_{n+1}, y_{n+1})$.

Na aplicação do método PC podemos usar previsor e e corretor de ordens diferentes mas, como Lambert prova em [13, p. 90] se aplicamos no modo PECE, não existe vantagem em usar o corretor com ordem maior que $k+1$ se o previsor for de ordem k.

Se os pontos são igualmente espaçados e o previsor e o corretor são ambos de ordem k, [16, p. 301] mostra que:

$$(1.18) \quad y_{n+1} = p_{n+1} + h\gamma_{k-1}\nabla^k f^p_{n+1}$$

e se o previsor é de ordem k e o corretor de ordem $k+1$ a mesma referência mostra que:

$$(1.19) \quad y_{n+1} = p_{n+1} + h\gamma_k\nabla^k f^p_{n+1} .$$

As fórmulas acima mostram que tendo sido completada a previsão, para efetuar a correção basta calcular γ_{k-1} e $\nabla^k f^p_{n+1}$ no primeiro caso, γ_k e $\nabla^k f^p_{n+1}$ no segundo.

Esta propriedade é muito importante em termos computacionais e será usada na implementação computacional do método.

1.2. *A Forma Matricial dos Métodos* PC

Neste parágrafo desenvolvemos a notação matricial para os métodos PC com passo variável dados em [4].

Para efeito de simplicidade na notação, consideramos o PVI no caso real, isto é $f : [a,b] \times R \to R$. Seja o vetor

$$Y_{n-1} = [y_{n-1}, y_{n-2}, \ldots, y_{n-k}, h_{n-1}y'_{n-1}, \ldots, h_{n-k}y'_{n-k}]^t$$

<u>Definição 1.1</u> Um método de "valor múltiplo" é definido pelas relações

$$(1.20) \qquad \begin{cases} Y_{n(o)} = B_{n-1}Y_{n-1} \\ Y_{n(m+1)} = Y_{n(m)} + \ell_{n-1}G(Y_{n(m)}) \qquad m \geq 0 \end{cases}$$

onde

$$Y_{n(m)} = [y_{n(m)}, y_{n-1}, \ldots, y_{n-k+1}, h_n y'_{n(m)}, h_{n-1}y'_{n-1}, \ldots, h_{n-k+1}y'_{n-k+1}]^t$$

B_{n-1} são matrizes $2k \times 2k$

ℓ_{n-1} são vetores $2k$ dimensionais e

$G : R^{2k} \to R$ é definida por:

$$G(Y_{n(m)}) = h_n f(y_{n(m)}) - h_n y'_{n(m)}$$

Podemos provar por indução sobre m que um método PC com passo variável aplicado no modo $P(EC)^r E$, pode ser colocado na forma (1.20), onde $m = 0,1,\ldots,r$. Isto quer dizer que aplicar o método PC na forma (0.10) é equivalente a aplicar o método (1.20) onde:

$$\ell_n = (\beta^*_{kn}, 0, \ldots, 0, 1, 0, \ldots, 0)^t$$

com o 1 na $(k+1)$-ésima posição e para fazer a última avaliação usamos $\ell_n = e_{k+1}$, vetor da base canônica e

$$
(1.21)\quad B_n = \left[\begin{array}{cccc|cccc}
-\alpha_{k-1,n} & -\alpha_{k-2,n} & \cdots & -\alpha_{o,n} & \beta_{k-1,n} & \beta_{k-2,n} & \cdots & \beta_{o,n} \\
1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
0 & 1 & 0 \cdots & 0 & 0 & 0 & \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0 & 0 & \cdots 1 & 0 & 0 & \cdots & \cdots & 0 \\
\hline
\frac{\alpha^*_{k-1,n}-\alpha_{k-1,n}}{\beta^*_{k,n}} & \cdots & \frac{\alpha^*_{o,n}-\alpha_{o,n}}{\beta^*_{k,n}} & & \frac{\beta_{k-1,n}-\beta^*_{k-1,n}}{\beta^*_{k,n}} & \cdots & \frac{\beta_{o,n}-\beta^*_{o,n}}{\beta^*_{k,n}} & \\
0 & 0 & 0 & & 1 & 0 & \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0 & 0 & 0 & & & & 1 & 0
\end{array}\right]
$$

Os métodos de passo múltiplo que usamos no PC definido pelas equações (1.20) têm seus coeficientes dependentes do passo, por isso colocamos o segundo índice n nos coeficientes α e β, ou seja os métodos de passo múltiplo usados são da forma:

$$
(1.22)\quad y_n = -\sum_{j=1}^{k} \alpha_{k-j,n} y_{n-j} + \sum_{j=0}^{k} h_{n-j} \beta_{k-j,n} f_{n-j}.
$$

Em particular quando o método PC usar os métodos de Adams como previsor e corretor, (1.20) toma a forma:

$$(1.23)\quad \begin{cases} Y_{n(o)} = B_{n-1}Y_{n-1} \\ Y_{n(m+1)} = Y_{n(m)} + \ell_{n-1}G(Y_{n(m)}) \qquad m \geq 0 \end{cases}$$

onde:

$$Y_{n(m)} = [y_{n(m)}, h_n y'_{n(m)}, h_{n-1}y'_{n-1}, \ldots, h_{n-k+1}y'_{n-k+1}]^t$$

$$B_n = \begin{bmatrix} -1 & \beta_{k-1,n} & \beta_{k-2,n} & \cdots & \beta_{o,n} \\ 0 & r_{k-1,n} & r_{k-2,n} & \cdots & r_{o,n} \\ 0 & 1 & 0 & & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}$$

com $r_{i,n} = \dfrac{\beta_{i,n} - \beta^*_{i,n}}{\beta^*_{k,n}} \qquad i = 0,1,\ldots,k-1$

e $\ell_n = (\beta^*_{k,n}, 1, 0, \ldots, 0)^t$

<u>Definição 1.2</u> Seja Y_n a solução de um PVI por um método como (1.20) e a_n a solução do mesmo problema por um outro método como (1.20); dizemos que os dois métodos são equivalentes se existe uma matriz T_n inversível tal que:

$$a_n = T_n Y_n.$$

Os métodos de Adams na forma matricial de Nordsieck consistem em tomar a_n como o seguinte vetor

$$(1.24) \quad a_n = [y_n, h_n y_n', \frac{h_n^2 y_n''}{2!}, \ldots, \frac{h_n^k y_n^{(k)}}{k!}]^t$$

Podemos então definir um método como (1.20) com passo constante $h = h_n$ colocando

$$(1.25) \begin{cases} a_{n(o)} = Aa_{n-1} \\ a_{n(m+1)} = a_{n(m)} + bF(a_{n(m)}) \qquad m \geq 0 \end{cases}$$

Se desenvolvemos $y(x_{n-i})$ por Taylor em torno de x_n podemos ver que:

$Y_n = Ta_n$ e portanto $a_n = T^{-1}Y_n$ e o método definido acima é equivalente a um método como (1.20) com passo constante $h = h_n$ para o qual vale:

$$Y_{n(o)} = BY_{n-1}$$

$$Ta_{n(o)} = BTa_{n-1}$$

$$a_{n(o)} = T^{-1}BTa_{n-1}$$

Desta forma a matriz A de (1.25) pode ser tomada como $A = T^{-1}BT$

Por (1.20) temos ainda que

$$Y_{n(m+1)} = Y_{n(m)} + \ell G(Y_{n(m)})$$

$$Ta_{n(m+1)} = Ta_{n(m)} + \ell G(Ta_{n(m)})$$

$$a_{n(m+1)} = a_{n(m)} + T^{-1}\ell G(Ta_{n(m)})$$

e assim podemos ainda tomar em (1.25)

$$b = T^{-1}\ell \quad e \quad F(x) = G(Tx)$$

Gear em [3, capítulo 10] mostra que a matriz A acima é a matriz do triângulo de Pascal, ou seja, o elemento (i,j) é dado por $\binom{j}{i}$, $i,j = 0,1,\ldots,k$.

Suponhamos que completamos o passo de x_{n-1} para x_n com tamanho de passo h e queremos passar para x_{n+1} com tamanho de passo αh. Como estamos aplicando uma fórmula com passo constante devemos obter as aproximações nos pontos $\tilde{x}_{n-i}=x_n-i(\alpha h)$ $i = 0,1,\ldots,k-1$. É claro que $\tilde{x}_n = x_n$ e portanto as aproximações $y_n, y_n', \ldots, y_n^{(k)}$ em (1.24) não precisam ser modificadas. Assim, para obtermos as aproximações necessárias basta premultiplicar a_n por uma matriz $C(\alpha)$ onde

$$C(\alpha) = \mathrm{diag}[1,\alpha,\alpha^2,\ldots,\alpha^k].$$

Se colocamos $h = h_n$ e $\alpha h = h_{n+1}$ então $\alpha = h_{n+1}/h_n$ e $C(\alpha)$ torna-se:

$$(1.26) \qquad C_n = \mathrm{diag}\left(1, \frac{h_{n+1}}{h_n}, \left[\frac{h_{n+1}}{h_n}\right]^2, \ldots, \left[\frac{h_{n+1}}{h_n}\right]^k\right)$$

e assim temos

$$(1.27) \qquad a_{n+1(o)} = AC_n a_n$$

$$a_{n+1(m+1)} = a_{n+1(m)} + \ell F(a_{n+1(m)}), \qquad m \geq 0$$

Como vemos, a mudança de passo no método de Nordsieck é extremamente simples, e é esta a grande vantagem deste método.

CAPÍTULO II

ESTABILIDADE E CONVERGÊNCIA

Os métodos previsor-corretor produzem um valor y_{n+1}, que esperamos aproxime $y(x_{n+1})$. Não é totalmente claro que isso ocorra, desde que o valor de y_{n+1} depende dos valores de $y_n, y_{n-1}, \ldots,$ bem como dos valores $f_n, f_{n-1}, \ldots,$ os quais geralmente têm erro, pois já são aproximações. Podemos então encontrar um valor para y_{n+1} totalmente insatisfatório.

Neste capítulo, baseados em [4], estudaremos primeiramente a estabilidade e convergência dos métodos PC com métodos de Adams de passo variável usando as técnicas do passo variável e da interpolação. Estudaremos também a estabilidade e convergência desses métodos quando, além de variar o passo variamos também a ordem, usando a referência [5] como base.

Os resultados serão obtidos por argumentos limites e portanto só serão válidos para passo suficientemente pequeno.

Trabalharemos sempre com uma das normas vetoriais e correspondente norma matricial subordinada.

norma euclidiana (norma espectral)
norma do máximo (norma linha)
norma da soma (norma coluna)

2.1. *Estabilidade e Convergência dos Métodos* PC *de Passo Variável.*

<u>Definição 2.1</u> Um esquema de seleção de passo é uma função real θ, tal que

$$h_{n+1} = H\theta(x_n, H) \tag{2.1}$$

onde para todo $H > 0$ e $x \in [a,b]$ vale que:

$$0 < \Delta \leq \theta(x_n, H) \leq 1, \quad \Delta \text{ constante.}$$

<u>Definição 2.2</u> Sejam Y_n a solução obtida por (1.20) e o seu correspondente $Y(x_n) = (y(x_n), \ldots, y(x_{n-k+1}), h_n y'(x_n), \ldots$ $\ldots, h_{n-k+1} y'(x_{n-k+1}))^t$. Definimos o erro global em x_n por:

$$E_n = Y_n - Y(x_n). \tag{2.2}$$

Já definimos o erro de truncamento local quando não estamos usando a notação matricial; aqui usamos a mesma definição lembrando apenas que agora este erro será um vetor, o qual indicaremos por d_n, ou seja

$$d_n = \tilde{Y}_{n+1} - Y(x_{n+1})$$

onde:

$$\tilde{Y}_{n+1(o)} = B_n Y(x_n)$$

$$\tilde{Y}_{n+1(m+1)} = \tilde{Y}_{n+1(m)} + \ell_n G(\tilde{Y}_{n+1(m)})$$

$\tilde{Y}_{n+1} = \tilde{Y}_{n+1(M)}$ se $(M+1)$ correções são feitas.

Notemos que $\tilde{Y}_{n+1}$ foi obtido usando $Y(x_n)$, enquanto Y_{n+1} usou Y_n. Para E_{n+1} vale:

$$(2.3) \qquad E_{n+1} = Y_{n+1} - \tilde{Y}_{n+1} + d_n =$$

$$= Y_{n+1(M-1)} + \ell_n G(Y_{n+1(M-1)}) - \tilde{Y}_{n+1(M-1)} - \ell_n G(\tilde{Y}_{n+1(M-1)}) + d_n =$$

$$= Y_{n+1(M-1)} - \tilde{Y}_{n+1(M-1)} + \ell_n [G(Y_{n+1(M-1)}) - G(\tilde{Y}_{n+1(M-1)})] + d_n =$$

$$= [I + \ell_n \frac{\partial G}{\partial Y}(V_{M-1})](Y_{n+1(M-1)} - \tilde{Y}_{n+1(M-1)}) + d_n$$

onde V_{M-1} está no segmento $(Y_{n+1(M-1)}, \tilde{Y}_{n+1(M-1)})$.

Notemos agora que $Y_{n+1(M-1)}$ pode ser escrito como função de $Y_{n+1(M-2)}, Y_{n+1(M-2)}$ como função de $Y_{n+1(M-3)}$, e assim por diante. Logo, (2.3) fica:

$$(2.4) \qquad E_{n+1} = [I + \ell_n \frac{\partial G^{(M)}}{\partial Y}] \ldots [I + \ell_n \frac{\partial G^{(1)}}{\partial Y}][Y_{n+1(o)} - \tilde{Y}_{n+1(o)}] + d_n$$

onde

$$\frac{\partial G^{(i+1)}}{\partial Y} = \frac{\partial G}{\partial Y}(\xi_i), \quad \xi_i \in (Y_{n+1(i)}, \tilde{Y}_{n+1(i)})$$

Lembrando que $Y_{n+1(o)} - \tilde{Y}_{n+1(o)} = B_n E_n$ (2.4) se torna:

(2.5) $$E_{n+1} = S_n E_n + d_n$$

onde $$S_n = \prod_{m=1}^{M} [I + \ell_n \frac{\partial G^{(m)}}{\partial Y}] B_n .$$

Da definição da função G temos que:

(2.6) $$\frac{\partial G}{\partial Y} = (h_{n+1} f_y, 0, \ldots 0, -1, \ldots, 0) = h_{n+1} f_y e_1 - e_d$$

onde e_i é o i-ésimo vetor da base canônica e d é a posição correspondente a $h_n y'_n$ em Y_n.

Introduzimos a notação $f_y^{(m)}$ como:

(2.7) $$f_y^{(m)} = \frac{\partial f}{\partial y}(q_m), \quad q_m \in (y_{n+1(m-1)}, \tilde{y}_{n+1(m-1)}) .$$

Com essa notação temos que:

(2.8) $$S_n = \prod_{m=1}^{M} [(I - \ell_n e_d) + h_{n+1} f_y^{(m)} \ell_n e_1] B_n =$$

$$= [(I - \ell_n e_d)^M + \Sigma T_i] B_n$$

onde os T_i são produtos de M termos da forma:

$$h_{n+1} f_y^{(m)} \ell_n e_1 \quad \text{ou} \quad (I - \ell_n e_d) ,$$

com pelo menos um termo como o primeiro.

Se colocamos,

$$(2.9) \qquad \hat{S}_n = (I-\ell_n e_d)^M B_n$$

então (2.8) pode ser escrito como

$$(2.10) \qquad S_n = \hat{S}_n + h_{n+1}\tilde{S}_n$$

onde $\tilde{S}_n$ é uma matriz cujos elementos são polinômios em $h_{n+1}f_y^{(m)}$ e coeficientes em B_n e ℓ_n (esta afirmação será justificada na prova do teorema 2.1).

Por (2.5) vemos que a condição para que o erro não se propague está relacionada com $||S_n||$. Daremos a seguir resultados com tais objetivos.

<u>TEOREMA 2.1</u>

Se os vetores α, α^*, β e β^* são uniformemente limitados ou se B_n e ℓ_n são uniformemente limitados para $h_{n+1} < u$ então $||\tilde{S}_n||$ também o será.

PROVA:

Por (2.8) temos:

$$T_i B_n = [(I-\ell_n e_d)^{i_1} h_{n+1}^{i_2} (\ell_n e_1)^{i_2} \prod_{k=1}^{i_2} f_y^{(j_k)}] B_n$$

com $j_k \in \{1,2,\ldots,n\}$, $i_1 + i_2 = M$, $i_2 \neq 0$.

Notemos então que $T_i B_n$ é uma amtriz $P = (P_{rs})$ tal que P_{rs} ou é nulo ou então é um monômio da forma $C_{rs}(f_y^{(j_1)} \ldots f_y^{(j_{i_2})}) h_{n+1}^{i_2}$ com C_{rs} constante. É claro que $\tilde{S}_n = h_{n+1}^{-1} \Sigma T_i B_n$ será uma matriz cujos elementos são polinômios em h_{n+1}, pois $i_2 \neq 0$ Agora, $|f_y^{(j)}|$ é limitada pela constante de Lipschitz e os elementos de α, α^*, β e β^* são limitados por hipótese. Então B_n e ℓ_n serão limitados exceto possivelmente a linha d da matriz B_n onde os elementos são da forma $C/\beta^*_{k,n}$. Mas podemos verificar que esses elementos não influenciam na construção de $\tilde{S}_n$. Assim os coeficientes do polinômio em h_{n+1} são limitados, e podemos então concluir que $||\tilde{S}_n||$ é limitada se $h_{n+1} \leq u$.

Se B_n e ℓ_n são limitados o resultado segue analogamente.

<u>Definição 2.3</u> Sejam x_n e x_m pontos da discretização tais que $a \leq x_n \leq x_m \leq b$; então definimos a matriz:

$$S_n^m = \begin{cases} I & \text{se } m = n \\ S_{m-1} S_{m-2} \cdots S_n & \text{se } m > n \end{cases}$$

LEMA 2.1

Se $S_n = \hat{S}_n + h_{n+1}\tilde{S}_n$ e existem constantes k_o e k_1 tais que:

$$||\hat{S}_n^m|| \leq k_o \quad \text{para} \quad a \leq x_n \leq x_m \leq b \quad \text{e}$$

$$||\tilde{S}_n|| \leq k_1 \quad \text{para todo } n, \text{ então:}$$

$$||S_n^m|| \leq k_o e^{k_o k_1 (b-a)} = k_2, \quad \text{para} \quad m \geq n$$

PROVA:

Provaremos a tese por indução sobre m. Se $m = n$ temos que:

$$||S_n^n|| = ||\hat{S}_n^n|| \leq k_o \leq k_o e^{k_o k_1 (b-a)}.$$

Suponhamos que a tese seja válida para todo k tal que $n \leq k < m$ e provemos que vale para m. Pela definição 2.3 temos:

$$(2.11) \qquad S_n^m = S_{m-1} \cdots S_n = (\hat{S}_{m-1} + h_m \tilde{S}_{m-1}) \cdots (\hat{S}_n + h_{n+1}\tilde{S}_n) =$$

$$= \hat{S}_n^m + \sum_{k=n}^{m-1} \hat{S}_{k+1}^m h_{k+1} \tilde{S}_k S_n^k$$

e assim obtemos:

$$(2.12) \qquad ||S_n^m|| \leq ||\hat{S}_n^m|| + \sum_{k=n}^{m-1} ||\hat{S}_{k+1}^m|| \, |h_{k+1}| \, ||\tilde{S}_k|| \, ||S_n^k|| \leq$$

$$\leq k_o + \sum_{k=n}^{m-1} k_o k_1 h_{k+1} ||S_n^k||.$$

Usando agora a hipótese de indução em (2.12)

$$(2.13)\qquad ||S_n^m|| \le k_o + \sum_{k=n}^{m-1} k_o k_1 h_{k+1} (k_o e^{k_o k_1 (x_k - x_n)}) \le$$

$$\le k_o (1 + \sum_{k=n}^{m-1} (e^{k_o k_1 h_{k+1}} - 1) e^{k_o k_1 (x_k - x_n)}).$$

Como $h_{k+1} = x_{k+1} - x_k$, substituindo na expressão acima obtemos:

$$(2.14)\qquad ||S_n^m|| \le k_o (1 + \sum_{k=n}^{m-1} (e^{k_o k_1 (x_{k+1} - x_n)} - 1) e^{k_o k_1 (x_k - x_n)}) =$$

$$= k_o e^{k_o k_1 (x_m - x_n)} \le k_o e^{k_o k_1 (b-a)}$$

o que demonstra o lema.

<u>TEOREMA 2.2</u>

Se um método previsor-corretor é tal que:

$$(2.15)\qquad \begin{array}{l} ||\hat{S}_n^m|| \le k_o, \quad \text{para} \quad a \le x_n \le x_m \le b \quad \text{e} \\ ||\tilde{S}_n|| \le k_1, \quad \text{para} \quad x_n \in [a,b] \end{array}$$

então o método é estável (definição 0.14).

PROVA:

Sejam Y_n obtida pela iteração (1.20) e $\overline{Y}_n$ obtida pela iteração :

$$\overline{Y}_{n(o)} = B_{n-1}\overline{Y}_{n-1} + r_{n-1}$$

$$\overline{Y}_{n(m+1)} = \overline{Y}_{n(m)} + \ell_{n-1}G(\overline{Y}_{n(m)})$$

que é a versão perturbada da iteração (1.20). Não consideramos perturbação no valor inicial, isto é, $\overline{Y}_o = Y_o$. Definindo $Q_n = \overline{Y}_n - Y_n$, por um desenvolvimento análogo ao que fizemos para o erro global E_n em (2.3), (2.4) e (2.8) obtemos:

(2.16) $\quad Q_{n+1} = S_n Q_n + r_n \quad$ onde $\quad Q_o = 0$

Por recorrência podemos ver que:

(2.17) $\quad Q_{n+1} = \sum_{i=0}^{n} S_{i+1}^{n+1} r_i$ e daí,

$$||Q_{n+1}|| \leq \sum_{i=0}^{n} ||S_{i+1}^{n+1}|| \, ||r_i||.$$

Com as hipóteses, o lema 2.1 nos permite afirmar que existe k_2 tal que $||S_{i+1}^{n+1}|| \leq k_2$, $0 \leq i \leq n$. Logo

(2.18) $\quad ||Q_{n+1}|| \leq k_2 \sum_{i=0}^{n} ||r_i||$

que é a condição de estabilidade da definição 0.14.

Com os dois teoremas acima estabelecemos condições suficientes para que um método previsor-corretor seja estável.

Mostraremos agora que os métodos previsor-corretor u

sando métodos de Adams satisfazem tais condições. Para tal é necessário dizer como faremos as mudanças de passo e impor certas restrições quanto a essas mudanças. A seguir daremos duas técnicas de mudança de passo.

Definição 2.4 Diremos que aplicamos a técnica do passo variável quando começamos com k pontos não igualmente espaçados onde conhecemos aproximações da solução e então calculamos os coeficientes da fórmula (1.22), que será empregada no PC.

Definição 2.5 Diremos que aplicamos a técnica da interpolação, com tamanho de passo h_{n+1} quando:

a) interpolamos sobre as k aproximações conhecidas para obter aproximações da solução nos pontos igualmente espaçados $\tilde{x}_{n-i} = x_n - ih_{n+1}$, $i = 0,1,\ldots,k-1$.

b) Aplicamos um método PC de passo constante sobre os pontos $\tilde{x}_{n-i}$ para encontrar uma aproximação no ponto $x_{n+1} = x_n + h_{n+1}$.

Essas duas técnicas são totalmente diferentes e a estabilidade do método pode depender da técnica empregada.

TEOREMA 2.3

Os coeficientes α, α^*, β e β^* são uniformemente limitados para os métodos de Adams usando a técnica do passo variável se existem constantes μ e η tais que:

$$(2.19) \qquad 0 < \mu \leq \frac{h_{j-1}}{h_j} \leq \eta, \quad \text{para todo } j.$$

<u>Nota</u>: Podemos provar o teorema acima (e os demais teoremas que exigirem esta condição) exigindo ao invés da condição (2.19) a condição da definição 2.1.

PROVA:

Os coeficientes α e α^* são limitados pois são constantes. Provemos então que os coeficientes β^* são uniformemente limitados. Por (1.22) e (1.15) temos que:

$$y_{n+1} = y_n + \beta^*_{k,n} h_{n+1} y'_{n+1} + \ldots + \beta^*_{o,n} h_{n-k+1} y'_{n-k+1},$$

onde

$$\beta^*_{k-i,n} = \frac{h_{n+1}}{h_{n+1-i}} \int_o^1 \prod_{\substack{j=o \\ j\neq i}}^{k} \frac{(x_n + sh_{n+1} - x_{n+1-j})}{(x_{n+1-i} - x_{n+1-j})} ds.$$

Por (2.19) é claro que $\dfrac{h_{n+1-i}}{h_{n+1}} \geq \mu^i$, ou seja,

$$(2.20) \qquad \frac{h_{n+1}}{h_{n+1-i}} \leq \frac{1}{\mu^i}.$$

Portanto:

$$|\beta^*_{k-i,n}| \leq \frac{1}{\mu^i} \int_o^1 \prod_{\substack{j=o \\ j\neq i}}^{k} \frac{|x_n + sh_{n+1} - x_{n+1-j}|}{|x_{n+1-i} - x_{n+1-j}|} ds.$$

Podemos então deduzir:

$$(2.21)\qquad \prod_{\substack{j=o\\ j\neq i}}^{k} \frac{|x_n+sh_{n+1}-x_{n+1-j}|}{|x_{n+1-i}-x_{n+1-j}|} \leq \prod_{j=o}^{i-1} \frac{A_j}{B_j} \cdot \prod_{j>i}^{k} \frac{A_j}{C_j}$$

onde $A_j = \dfrac{\eta^j-1}{\eta-1}$, $B_j = \dfrac{\mu^i-\mu^j}{\mu-1}$, $C_j = -B_j$.

(Para detalhes à respeito de (2.21) ver [16, p. 65]).

Desta forma obtemos:

$$(2.22)\qquad |\beta^*_{k-i,n}| \leq \frac{1}{\mu^i}\left[\prod_{j=o}^{i-1} \frac{A_j}{B_j} \cdot \prod_{j>i}^{k} \frac{A_j}{C_j}\right] = D_i.$$

Notemos que D_i é independente dos espaçamentos. Se calculamos $D = \max\{D_i\}$ temos então

(2.23) $|\beta^*_{k-i,n}| \leq D$ para $i = 0,1,\ldots,k$, e, portanto β^* é uniformemente limitado.

Para β é claro que o desenvolvimento acima ainda é válido e o teorema fica provado.

<u>TEOREMA 2.4</u>

As matrizes B_n e os vetores ℓ_n são uniformemente limitados para os métodos de Adams usando a técnica da interpolação se os passos satisfazem a condição (2.19).

PROVA:

Usando a técnica da interpolação temos que $\ell_n = \ell$, portanto uniformemente limitado, e $B_n = BC_n$ onde

$$C_n = \mathrm{diag}[1, \frac{h_{n+1}}{h_n}, \ldots, (\frac{h_{n+1}}{h_n})^k].$$

Pela condição (2.19), C_n é uniformemente limitada e portanto B_n também o é, o que prova o teorema.

Estamos agora em condições de provar que os métodos PC com métodos de Adams usando qualquer das duas técnicas de mudança de passo são estáveis e convergentes.

TEOREMA 2.5

Seja um método PC com métodos de Adams usando a técnica do passo variável. Se os passos são escolhidos por um esquema de seleção como o da definição 2.1, então este método é estável e convergente.

PROVA:

Sejam B_n e ℓ_n como em (1.23) e $\hat{S}_n$ como em (2.9). Calculando $\hat{S}_n$ obtemos:

$$(2.24) \qquad \hat{S}_n = \begin{bmatrix} -1 & \beta^*_{k-1,n} & \beta^*_{k-2,n} & \cdots & \beta^*_{o,n} \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}$$

Assim as matrizes $\hat{S}_n$ serão uniformemente limitadas, pois pelo teorema (2.3), os $\beta^*_{i,n}$ o são.

Para $0 \leq i \leq k$ temos que:

$$\hat{S}_n^{n+i} = \hat{S}_{n+i-1} \hat{S}_{n+i-2} \cdots \hat{S}_n$$

e portanto

$$||\hat{S}_n^{n+i}|| \leq ||\hat{S}_{n+i-1}|| \cdots ||\hat{S}_n|| \leq p,$$

pois temos um produto com um número finito de fatores limitados.

Seja agora $i > k$.

A matriz $\hat{S}_n$ tem a propriedade

$$(2.25)\quad \hat{S}_n^{n+i} = \begin{bmatrix} (-1)^i & x_{k-1} & \cdots & x_o \\ 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$$

onde x_r, $r = 0,1,\ldots,k-1$ é uma soma de k elementos $\beta^*_{r,n+r}$ sendo portanto uniformemente limitado, assim as matrizes $\hat{S}_n^{n+i}$ serão uniformemente limitadas.

Usando novamente o teorema 2.3 temos que os coeficientes α,α^*,β e β^* são uniformemente limitados, logo pelo teorema 2.1, $||\tilde{S}_n|| \leq k_1$. Como provamos acima que $||\hat{S}_n^{n+i}|| < k_o$, o teorema 2.2 nos garante que o método PC com a técnica do passo variável é estável.

Provemos agora que este método é convergente. Por (2.5) temos que:

$$E_{n+1} = S_n E_n + d_n = S_o^{n+1} E_o + \sum_{i=o}^{n} S_{i+1}^{n+1} d_i$$

Obtemos então

$$(2.26)\qquad ||E_{n+1}|| \leq ||S_o^{n+1}|| \, ||E_o|| + \sum_{i=o}^{n} ||S_{i+1}^{n+1}|| \, ||d_i||.$$

Como $||\tilde{S}_n|| \leq k_1$ e $||\hat{S}_n^{n+i}|| \leq k_o$ para $i \geq 0$

temos pelo lema 2.1 que:

$||S_n^m|| \leq k_2$ para todo $m \geq n$. Segue que

(2.27) $$||E_{n+1}|| \leq k_2||E_o|| + \sum_{i=o}^{n} k_2||d_i|| =$$

$$= k_2||E_o|| + k_2 \sum_{i=o}^{n} ||d_i||.$$

Conforme mostra o lema 2.2 a seguir os d_i satisfazem:

(2.28) $$||d_i|| \leq h_{i+1}e(H) \quad \text{onde} \quad e(H) \to 0 \quad \text{para} \quad H \to 0 .$$

Nestas condições (2.27) se transforma em

(2.29) $$||E_{n+1}|| \leq k_2||E_o|| + k_2 \sum_{i=o}^{n} h_{i+1}e(H) \leq$$

$$\leq k_2||E_o|| + k_2 nHe(H) \leq$$

$$\leq k_2[||E_o|| + (b-a)e(H)] .$$

Se os valores iniciais são exatos $||E_o|| = 0$ e se $H \to 0$, então $E_n \to 0$ por (2.29) e temos a convergência garantida.

<u>LEMA 2.2</u>

O erro de truncamento local do método PC com métodos de Adams de ordem pelo menos r satisfaz:

$$||d_n|| \leq h_{n+1}(H^{r-1}Q \ \mathrm{Max}|y^{(r)}|) + 0(H^{r+1})$$

desde que os passos sejam como na definição 2.1.

Prova:

Por definição $d_n = \tilde{Y}_{n+1} - Y(x_{n+1})$.

Se olhamos como $\tilde{Y}_{n+1}$ foi construído vemos que:

$$\tilde{Y}_{n+1} = (\tilde{y}_{n+1}, h_{n+1}\tilde{y}'_{n+1}, h_n y'(x_n), \ldots, h_{n-k+1} y'(x_{n-k+1}))$$

e

$$Y(x_{n+1}) = (y(x_{n+1}), h_{n+1} y'(x_{n+1}), h_n y'(x_n), \ldots, h_{n-k+1} y'(x_{n-k+1})).$$

Assim

$$d_n = (\tilde{y}_{n+1} - y(x_{n+1}), h_{n+1}(\tilde{y}'_{n+1} - y'(x_{n+1})), 0, \ldots, 0). \tag{2.30}$$

Provemos inicialmente que os erros de truncamento local σ^p_{n+1} e σ_{n+1} dos métodos de Adams-Bashforth e Adams-Moulton de ordem pelo menos $r-1$ satisfazem:

$$|\sigma^p_{n+1}| \leq h_{n+1}(H^{r-1} C \max|y^{(r)}|) \tag{2.31}$$

$$|\sigma_{n+1}| \leq h_{n+1}(H^{r-1} C \max|y^{(r)}|) \tag{2.32}$$

Faremos a prova somente para o caso explícito pois o outro caso é inteiramente análogo.

De (1.8) obtemos:

$$|\sigma^p_{n+1}| \leq H(H^{r-1}\max|y^{(r)}|).$$

Por hipótese temos que:

$$h_{n+1} = H\phi(x_n,H) \geq H\Delta$$

e portanto

$$|\sigma^p_{n+1}| \leq h_{n+1}(\frac{1}{\Delta}H^{r-1}\max|y^{(r)}|).$$

Provemos agora a tese:

Conforme a definição de σ_{n+1} temos:

$$(2.33) \qquad y(x_{n+1}) = y(x_n) + h_{n+1}\sum_{j=1}^{r-1}\beta^*_{j,r}f(x_{n+1-j},y(x_{n+1-j})) +$$

$$+ h_{n+1}\beta^*_{o,r}f(x_{n+1},y(x_{n+1})) + \sigma_{n+1}.$$

Num estágio m+1 da aplicação do método PC, conforme (0.10), a aproximação é dada por:

$$y^{[m+1]}_{n+1} = y_n + h_{n+1}\sum_{j=1}^{r-1}\beta^*_{j,r}f_{n+1-j} + h_{n+1}\beta^*_{o,r}f(x_{n+1},y^{[m]}_{n+1}).$$

Fazendo a diferença membro a membro entre (2.33) e a expressão acima, e usando o teorema do valor médio encontramos:

$$(2.34) \qquad |y(x_{n+1}) - y_{n+1}^{[m+1]}| \le h_{n+1}\alpha_r |y(x_{n+1}) - y_{n+1}^{[m]}| + |\sigma_{n+1}|$$

onde $\alpha_r = |\beta^*_{o,r}|L$, L constante de Lipschitz de f.

Se $\tilde{y}_{n+1}$ é a aproximação produzida pela aplicação do corretor M vezes então de (2.34) obtemos por recorrência:

$$|y(x_{n+1})-\tilde{y}_{n+1}| \le h_{n+1}^{M-1}\alpha_r^{M-1}(h_{n+1}\alpha_r|y(x_{n+1})-y_{n+1}^{[o]}|+|\sigma_{n+1}|) +$$

$$+ h_{n+1}^{M-2}\alpha_r^{M-2}|\sigma_{n+1}|+ \dots + h_{n+1}\alpha_r|\sigma_{n+1}|+|\sigma_{n+1}|.$$

Se a ordem do previsor é tal que o primeiro termo do segundo membro da desigualdade acima é de $0(H^{r+1})$, então, pelo que provamos inicialmente obtemos:

$$|y(x_{n+1})-\tilde{y}_{n+1}| \le h_{n+1}(H^{r-1}\frac{1}{\Delta}\max|y^{(r)}|) + 0(H^{r+1}).$$

A expressão acima é um limitante para a primeira coordenada de d_n, dado em (2.30), pois os métodos PC na forma matricial são equivalentes aos na forma usual.

Para limitar a segunda coordenada fazemos:

$$|\tilde{y}'_{n+1} - y'(x_{n+1})| = |f(x_{n+1},y_{n+1}) - f(x_{n+1},y(x_{n+1}))| \le$$

$$\le L|\tilde{y}_{n+1}-y(x_{n+1})| \le h_{n+1}[H^{r-1}\frac{L}{\Delta}\max|y^{(r)}|]+0(H^{r+1})$$

e portanto

$$h_{n+1}|\tilde{y}'_{n+1} - y'(x_{n+1})| = 0(H^{r+1}).$$

Tomando a norma em (2.30) obtemos:

$$||d_n|| \leq h_{n+1}[H^{r-1}\Omega \max|y^{(r)}|] + 0(H^{r+1}) \tag{2.35}$$

o que demonstra o lema 2.2.

<u>TEOREMA 2.6</u>

Seja um método PC em que previsor e corretor são métodos de Adams de k passos. Se pelo menos k passos de tamanho constante são tomados entre cada mudança de passo e se os passos satisfazem (2.19), então o método PC usado com a técnica de interpolação (definição 2.5) é estável e convergente.

PROVA:

Como estamos usando a técnica da interpolação temos por (1.26) que $\ell_n = \ell$ e $B_n = BC_n$ com $C_n = I$ se $h_{n+1} = h_n$.

Pelo teorema 2.4, a matriz B_n e o vetor ℓ_n satisfazem as hipóteses do teorema 2.1, logo:

$$||\tilde{S}_n|| \leq k_1.$$

Assim para que tenhamos satisfeitas as hipóteses do teorema 2.2, o qual nos garantirá a estabilidade, precisamos provar que:

$$||\hat{S}_n^m|| \leq k_o, \qquad m \geq n.$$

Temos:

$$\hat{S}_n = (I-\ell_n e_2^t)^M B_n = (I-\ell e_2^t)^M BC_n = RC_n$$

onde

$$R = \begin{bmatrix} -1 & \beta^*_{k-1} & \beta^*_{k-2} & \cdots & \beta^*_o \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & 1 & 0 \end{bmatrix}$$

R tem k autovalores nulos e autovalor -1. A este autovalor corresponde o autovetor e_1. Além disso,

$$R^k = \begin{bmatrix} (-1)^k & z_{k-1} & \cdots & z_o \\ 0 & 0 & \cdots & 0 \\ & & 0 \quad \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$$

onde z_i é uma combinação dos β^*_j e assim,

(2.36) $R^k X = \mu_X e_1$, para todo vetor X.

Temos então

$$||R^k||\,||X|| \geq ||R^k X|| = |\mu_X|$$

o que leva a

(2.37) $$\frac{|\mu_X|}{||X||} \leq ||R^k|| \quad \text{e portanto} \quad \frac{|\mu_X|}{||X||}$$

é uniformemente limitada.

Observe-se que 1 é autovalor de C_n, sendo e_1 o autovetor associado, e que se $|i-j| < k$, pelo menos uma das duas matrizes C_i e C_j coincide com a identidade.

Analisemos o produto:

(2.38) $$\hat{S}_n^m = RC_{m-1}RC_{m-2} \ldots RC_n$$

a) se (2.38) contém $p < 2k+2$ matrizes do tipo RC_i, então

$$||\hat{S}_n^m|| \leq ||R||^p\,||C_{m-1}|| \ldots ||C_n||$$

e portanto $\hat{S}_n^m$ será limitada, pois cada C_i o é.

b) se (2.38) contém $p \geq 2k+2$ matrizes do tipo RC_i, então:

b.1) se não houve mudança de passo entre x_n e x_m temos:

$$\hat{S}_n^m = \begin{cases} R^{m-n} & \text{se não houve mudança em } x_{n-1} \\ C_n R^{m-n} & \text{se houve mudança.} \end{cases}$$

Mas como $m-n \geq 2(k+1) \geq k$ deduzimos:

$$\hat{S}_n^m = \begin{cases} R^{m-n}Z = R^{m-n-k}R^kZ = R^{m-n-k}\mu_Z e_1 = (-1)^{m-n-k}e_1\mu_Z \\ C_nR^{m-n}Z = C_nR^{m-n-k}R^kZ = C_nR^{m-n-k}\mu_Z e_1 = (-1)^{m-n-k}e_1\mu_Z \end{cases}$$

e portanto em ambos os casos temos:

(2.39) $\quad ||\hat{S}_n^m Z|| \leq k_3||Z||$

onde $\quad k_3 = ||R^k||.$

b.2) Se houve mudança entre x_n e x_m, seja x_p o primeiro ponto a partir de x_n onde ocorreu tal mudança. Então, pelo menos uma das relações abaixo é válida.

(2.40) $\quad p-n \geq k+1$

$m-p \geq k+1$

Valendo (2.40), seja $q = n+s$, onde

$$s = (p-n) - [\frac{p-n}{k}]*k + k$$

então:

$$\hat{S}_n^m Z = RC_{m-1}RC_{m-2} \ldots RC_q R^k R^{q-n-k} C_n Z$$

com $0 \le q-n-k < k$. Tomando $X = R^{q-n-k}C_n Z$, obtemos:

$$||X|| \le k_4 ||Z||$$

e portanto

$$\hat{S}_n^m Z = RC_{m-1} \ldots RC_q \mu_X e_1 = \mu_X e_1$$

Como conseqüência dos resultados acima temos:

(2.41) $$||\hat{S}_n^m Z|| \le \mu_X \le k_3 ||X|| \le k_3 k_4 ||Z||.$$

Se $p-n \le k+1$, então $m-p \ge k+1$ e neste caso:

$$\hat{S}_n^m = RC_{m-1} \ldots RC_{p+k} R^k C_p R^{p-n} C_n$$

$$\hat{S}_n^m Z = RC_{m-1} \ldots RC_{p+k} R^k X = \mu_X e_1$$

onde $X = C_p R^{p-n} C_n Z$ e portanto

$$||X|| \le k_5 ||Z|| \quad \text{pois} \quad p-n \le k.$$

Novamente temos:

(2.42) $$||\hat{S}_n^m Z|| \le \mu_X \le k_3 ||X|| \le k_3 k_5 ||Z||.$$

Logo por (2.39), (2.41) e (2.42) concluímos que $||\hat{S}_n^m||$ é uniformemente limitada e provamos assim a estabilidade.

O estudo da convergência é feito como no teorema 2.5 com a diferença que, para provar que d_n satisfaz (2.35) agora é mais simples, pois estamos trabalhando com passo constante h, ver [13, p. 90 e 91].

2.2. *Estabilidade e Convergência dos Métodos* PC *com Passo e Ordem Variáveis*

Passaremos agora à generalização dos dois últimos teoremas, provando serem os métodos PC com métodos de Adams estáveis e convergentes quando além de variar o passo variamos também a ordem. Para isso definimos:

<u>Definição 2.6</u> Seja $F = \{F_i\}$ $i = 1,2,\ldots,$ uma família de métodos. Um esquema I de seleção de fórmula é uma função inteira $I(H,x)$ tal que a fórmula usada no intervalo $[x_n, x_{n+1}]$ é $F_{I(H,x_n)}$, onde H é o parâmetro da definição 2.1.

<u>Observação 2.1</u> Consideremos os métodos de Adams na forma de Nordsieck com a técnica da interpolação.

Para métodos de i-passos $1 \le i \le k$, tomamos

$$Y_n = (y_n, h_n y'_n, \ldots, \frac{h_n^k y_n^{(k)}}{k!})^t$$

onde as componentes de Y_n de i+1 até k são tomadas nulas.

Como em (1.27), $B_n = AC_n$ onde:

$$(2.43)\quad A = \left[\begin{array}{cccccc|c} 1 & 1 & 1 & \cdots & \cdots & 1 & \\ 0 & 1 & 2 & 3 & \cdots & i & \\ 0 & 0 & 1 & 3 & 6 \cdots & \cdots & 0 \\ & & & & & & \\ & & & & & & \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & \\ \hline & & 0 & & & & 0 \end{array}\right]$$

$$C_n = \mathrm{diag}[1, \frac{h_{n+1}}{h_n}, (\frac{h_{n+1}}{h_n})^2, \ldots, (\frac{h_{n+1}}{h_n})^k]$$

e ℓ_n é o vetor para com i-passos estendido para k+1 componentes com componentes nulas a partir da (i+1)-ésima.

Com tais convenções vemos que diminuir a ordem é trivial, pois basta ignorar algumas componentes dos vetores em questão. Igualmente seria trivial aumentar a ordem considerando nulas as derivadas de ordem superior requeridas, entretanto esta não se mostra uma boa técnica.

Se permitimos que a ordem seja aumentada somente de 1 em cada vez, de i para i+1, podemos obter uma aproximação para $h^{i+1}y^{(i+1)}/(i+1)!$ calculando $\frac{1}{(i+1)} \frac{\nabla h^i y^{(i)}}{i!}$. Notemos que:

$$\nabla y_n^{(i)} = y_n^{(i)} - y_{n-1}^{(i)}$$

e para que possamos calcular esta diferença descendente precisamos no passo n do valor de $y_{n-1}^{(i)}$. Colocaremos este valor na coordenada $i+2$ de Y_n e isto é equivalente a modificar a matriz A para

$$
(2.44) \quad A = \left[\begin{array}{cccccc|c}
1 & 1 & 1 & & & \cdots & 1 & \\
0 & 1 & 2 & 3 & & \cdots & i & \\
0 & 0 & 1 & 3 & 6 & \cdots & & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & & \\
0 & \cdots & \cdots & \cdots & \cdots & 0 & 1 & \\
0 & \cdots & \cdots & \cdots & \cdots & 0 & 1 & \\
\hline
 & & 0 & & & & & 0
\end{array}\right] \begin{array}{l} \\ \\ \\ \\ \text{(linha } i+1) \\ \\ \\ \end{array}
$$

O vetor Y_n obtido usando (1.27) com A como acima é tal que contém na coordenada $i+2$ o valor $h_n^i y_{n-1}^{(i)}/i!$.

Finalmente o cálculo de $h_n^{i+1} y_n^{(i+1)}/(i+1)!$ será completado premultiplicando o vetor Y_n pela matriz:

$$(2.45)\quad O_{i+1,i} = \begin{bmatrix} 1 & 0 & 0 & \cdots & & & & 0 \\ 0 & 1 & 0 & \cdots & & & & 0 \\ & & \ddots & & & & & \\ & & & 1 & & & & \\ & & & \frac{1}{i+1} & \frac{-1}{i+1} & & & \\ & & & & & 1 & & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{bmatrix} \text{(linha } i+2)$$

a qual independe do passo.

Chamaremos a técnica de mudança de ordem recém descrita de técnica da diferença de mudança de ordem.

É interessante notar que, em qualquer técnica, para passar de uma fórmula F_i para outra fórmula F_j, devemos multiplicar Y_n por uma matriz O_{ji}, em geral dependente do tamanho do passo.

Por um desenvolvimento análogo a (2.3), (2.4) e (2.5) podemos mostrar que:

$$(2.46)\qquad E_{n+1} = S_n O_{i_n+1,i_n} E_n + d_n$$

onde F_{i_n} é a fórmula usada no passo de x_{n-1} para x_n.

A partir de agora omitiremos o índice i_n e usaremos simplesmente o índice n, além de indicar $T_n = S_n O_n$.

<u>Observação 2.2</u> Se tomamos $Y_n = (y_n, h_n y'_n, \ldots, h_{n-k+1} y'_{n-k+1})$ e aplicamos (1.20) com a técnica do passo variável temos $O_{ji} = I$, pois neste caso, para mudar a ordem é suficiente a avaliação de f no ponto adequado.

<u>Definição 2.7</u> Definimos a matriz $\hat{T}_n$ como sendo o valor de T_n para a equação $y' = 0$.

<u>TEOREMA 2.7</u>

Seja um método PC com métodos de Adams usando a técnica do passo variável. Então o método é estável e convergente para todo esquema de seleção de passo, mesmo que a ordem seja modificada em todo passo.

Prova:

Como estamos usando a técnica do passo variável temos que $O_n = I$, portanto $\hat{T}_n = \hat{S}_n$ e $\hat{T}_n^m = \hat{S}_{m-1} \ldots \hat{S}_n$, onde $\hat{S}_i$ é definida por (2.24), com as dimensões adequadamente aumentadas para $k+1$ à maneira de (2.43). O restante da prova segue como no teorema 2.5.

<u>TEOREMA 2.8</u>

Seja um método de discretização baseado numa família

finita de métodos PC com métodos de Adams, usando a técnica da interpolação para mudança de passo, e satisfazendo a propriedade que a ordem de cada corretor é igual ou supera de um a ordem de seu previsor.

Suponhamos que para os esquemas de seleção de passo e fórmula empregados, existe um Q finito tal que em quaisquer Q ou mais passos consecutivos, há pelo menos p passos consecutivos constantes com ordem r constante (onde r é a ordem do método PC atual), valendo a relação:

$$p = \begin{cases} r & \text{se todos os } O_n\text{'s são a identidade.} \\ r+1 & \text{se a técnica da diferença foi usada.} \end{cases}$$

Então o método de discretização é estável e convergente.

Prova:

Temos $\hat{T}_n = \hat{S}_n O_n$ e por (2.9) segue que:

$$\hat{T}_n = (I-\ell_n e_2^t)^M B_n O_n .$$

Como estamos usando a técnica da interpolação vale que $B_n = AC_n$ e portanto

$$\hat{T}_n = (I-\ell_n e_2^t)^M AC_n O_n = RC_n O_n$$

onde R é uma matriz como no teorema 2.6, só que neste caso R não será constante em todos os passos, pois dependerá

da ordem usada em cada passo. Então denotaremos R por R_n obtendo:

$$(2.47) \qquad \hat{T}_n = R_n C_n O_n .$$

Notemos que O_n é uniformemente limitada (pois O_n é a identidade ou então é como em 2.40), o mesmo ocorrendo com R_n (como vimos no teorema 2.6). Portanto se $0 \leq i \leq Q$, $\hat{T}_n^{n+i}$ será uniformemente limitada.

Se $i > Q$, por hipótese existe pelo menos p passos consecutivos com passo e ordem constantes, logo $C_q = O_q = I$ nos pontos correspondentes a esses passos. Além disso nestes mesmos pontos é claro que B_q e ℓ_q são constantes. Assim existe q, $n \leq q \leq n+Q$ tal que:

$$\hat{T}_n^{n+i} = \hat{T}_{p+q}^{n+i} (R_q)^p \, \hat{T}_n^q$$

onde:

$$(2.48) \quad R_q = \left[\begin{array}{c|c} R & 0 \\ \hline \tilde{R} & 0 \end{array}\right]$$

com R uma matriz $(r+1)\times(r+1)$ como no teorema 2.6 e $\tilde{R}$ uma matriz nula se todos os $O_i = I$ ou $\tilde{R}$ tem o elemento da primeira linha e última coluna igual a 1 se a técnica da diferença para o aumento da ordem foi usada. Das propriedades da matriz R dadas no teorema 2.6 temos que:

$$R_q^p X = \mu_X e_1$$

onde $\quad |\mu_X| \le K||X||.$

Pela estrutura de R_j, C_j e O_j, e_1 é um autovetor comum a elas, associado ao autovalor ± 1, assim também $\hat{T}_j e_1 = e_1$ e temos:

$$(2.49) \qquad ||\hat{T}_n^{n+i} Z|| = ||\hat{T}_{p+q}^{n+i} (R_q)^p \hat{T}_n^q Z|| = ||\hat{T}_{p+q}^{n+i} (R_q)^p X|| =$$

$$= ||\hat{T}_{p+q}^{n+i} \mu_X e_1|| = |\mu_X| \le K||X||$$

onde $X = \hat{T}_n^q Z = \hat{T}_{q-1} \ldots \hat{T}_n Z.$

Segue que:

$$||X|| \le ||\hat{T}_{q-1}|| \ldots ||\hat{T}_n|| ||Z|| \le$$

$$\le [\max_{n \le j \le q-1} ||\hat{T}_j||]^{q-n} ||Z||$$

que aplicado em (2.49) fornece

$$||\hat{T}_n^{n+i} Z|| \le K [\max_{n \le j \le q-1} ||\hat{T}_j||]^{q-n} ||Z|| = C||Z||.$$

Como evidentemente $\tilde{T}_n = \tilde{S}_n O_n$ é uniformemente limitada, o teorema 2.2 permite concluir a estabilidade.

A prova da convergência é feita como no teorema 2.6.

<u>Observação 2.3</u> Por (2.29) e pelo lema 2.2 podemos concluir que, se o erro de truncamento local d_n é de $O(H^r)$, então o erro global E_n é de $O(H^{r-1})$. No entanto Shampine pro

va em [16, p. 66] que, para o caso particular dos métodos PC com métodos de Adams aplicados no modo PECE usando a técnica do passo variável, podemos concluir que com erro de truncamento local de $0(H^r)$, o erro global também é de $0(H^r)$. Este fato será usado no capítulo 4.

Comparando a técnica do passo variável (definição 2.4) com a técnica da interpolação (definição 2.5) fica claro que, em virtude das hipóteses restritivas dos teoremas 2.6 e 2.8, a primeira técnica abrange um conjunto maior de esquemas de seleção de passo bem como de ordem.

CAPÍTULO III

IMPLEMENTAÇÃO DOS MÉTODOS DE ADAMS

Neste capítulo nosso objetivo será a implementação computacional de métodos PC com métodos de Adams usando a técnica do passo variável e da interpolação definidas anteriormente (definições 2.4 e 2.5).

Na técnica do passo variável usaremos os métodos de Adams na forma descrita no início do capítulo 1. Para tanto precisamos do cálculo do polinômio interpolador a cada passo, e isto é descrito na secção 3.1, onde é considerado também a armazenagem eficiente dos valores envolvidos. Estes elementos são muito importantes do ponto de vista do custo computacional seriamente onerado pelas mudanças de passo e ordem. Simplificações são obtidas quando um certo número n_s de passos permanecem constantes, o que evidentemente reduz o custo. O desenvolvimento apresentado é baseado nos trabalhos de Krogh [11] e de Shampine [16].

Na técnica da interpolação usaremos os métodos na forma de Nordsieck; na secção 3.2 daremos uma maneira eficiente de efetuar o produto $Y_{n+1(o)} = AY_n$, usando o fato que A é a matriz do triângulo de Pascal. Tomamos por base o trabalho de Gear [3].

3.1. *Implementação dos Métodos de Adams com a Técnica do Passo Variável*

O método de Adams-Bashforth de ordem k é dado pela expressão:

$$(3.1) \qquad p_{n+1} = y_n + \int_{x_n}^{x_{n+1}} P_{k,n}(t)\,dt,$$

onde $P_{k,n}(x)$ é o polinômio interpolador de f, para o qual usaremos a forma de Newtom:

$$P_{k,n}(x) = f[x_n] + (x-x_n)f[x_n,x_{n-1}] +$$

$$+ (x-x_n)(x-x_{n-1})f[x_n,x_{n-1},x_{n-2}]+$$

$$+ \ldots + (x-x_n)\ldots(x-x_{n-k+2})f[x_n,x_{n-1},\ldots,x_{n-k+1}].$$

<u>Definição 3.1</u> Sejam $x_o,\ldots,x_N$ os pontos da discretização. Definimos então:

$$s = (x-x_n)/h_{n+1}$$

$$\psi_i(n+1) = h_{n+1}+h_n+\ldots+h_{n+2-i} \qquad i \geq 1$$

$$a_i(n+1) = \frac{h_{n+1}}{\psi_i(n+1)} \qquad i \geq 1$$

$$b_1(n+1) = 1$$

$$b_i(n+1) = \frac{\psi_1(n+1)\psi_2(n+1)\ldots\psi_{i-1}(n+1)}{\psi_1(n)\psi_2(n)\ldots\psi_{i-1}(n)} \quad , \quad i \geq 2$$

$$\phi_1(n) = f[x_n] = f_n$$

$$\phi_i(n) = \psi_1(n)\ldots\psi_{i-1}(n) f[x_n,\ldots,x_{n-i+1}] \qquad i \geq 2.$$

Com as notações acima temos:

$$\psi_i(n+1) = \psi_{i-1}(n) + h_{n+1}, \quad i = 2,3,\ldots$$

(3.2)

$$\psi_1(n+1) = h_{n+1}$$

$$b_i(n+1) = b_{i-1}(n+1) . \frac{\psi_{i-1}(n+1)}{\psi_{i-1}(n)} \qquad i = 2,3,\ldots,$$

e quando trabalhamos com passo constante h temos:

$$\psi_i(n+1) = ih, \quad a_i(n+1) = \frac{1}{i}, \quad b_i(n+1) = 1 \qquad e$$

(3.3)

$$\phi_i(n) = \nabla^{i-1} f_n \quad \text{para} \quad i = 1,2,\ldots$$

Com estas notações o termo

$$(x-x_n)\ldots(x-x_{n-i+2}) f[x_n,\ldots,x_{n+2-i}]$$

de $P_{k,n}(x)$ pode ser escrito como:

(3.4) $$(sh_{n+1})(sh_{n+1}+h_n)+\ldots+(sh_{n+1}+h_n+\ldots+h_{n-i+3}) \; *$$

$$* \; \frac{\phi_i(n)}{\psi_1(n)\psi_2(n)\ldots\psi_{i-1}(n)}$$

$$= \frac{(sh_{n+1})}{\psi_1(n+1)} \; \frac{(sh_{n+1}+h_n)}{\psi_2(n+1)} \; \ldots \; \frac{(sh_{n+1}+\ldots+h_{n-i+3})}{\psi_{i-1}(n+1)} \; *$$

$$* \; \frac{\psi_1(n+1)\ldots\psi_{i-1}(n+1)}{\psi_1(n)\ldots\psi_{i-1}(n)}\phi_i(n) =$$

$$= \frac{(sh_{n+1})}{\psi_1(n+1)} \; \ldots \; \frac{(sh_{n+1}+\psi_{i-2}(n))}{\psi_{i-1}(n+1)} \; b_i(n+1)\phi_i(n).$$

Por simplicidade introduzimos a notação:

(3.5) $$C_{i,n}(s) = \begin{cases} 1 & , \; i = 1 \\ \dfrac{sh_{n+1}}{\psi_1(n+1)} = s & , \; i = 2 \\ \left(\dfrac{sh_{n+1}}{\psi_1(n+1)}\right)\left(\dfrac{sh_{n+1}+\psi_1(n)}{\psi_1(n+1)}\right)\ldots\left(\dfrac{sh_{n+1}+\psi_{i-2}(n)}{\psi_{i-1}(n+1)}\right), & i \geq 3. \end{cases}$$

Colocando $\phi_i^*(n) = b_i(n+1)\phi_i(n)$ temos:

(3.6) $$P_{k,n}(x) = P_{k,n}(sh_{n+1}+x_n) = \sum_{i=1}^{k} C_{i,n}(s)\phi_i^*(n)$$

Notemos que se $s = 1$, temos acima o valor de $P_{k,n}(x_{n+1}) = p'_{n+1}$, que é uma aproximação para a função f em x_{n+1}. Ainda, por (3.2), temos que $C_{i,n}(1) = 1$ e obtemos:

$$(3.7) \qquad p'_{n+1} = \sum_{i=1}^{k} \phi_i^*(n).$$

Para obter a aproximação em x_{n+1} substituímos (3.6) em (3.1) e assim

$$(3.8) \qquad p_{n+1} = y_n + h_{n+1} \sum_{i=1}^{k} \phi_i^*(n) \int_0^1 C_{i,n}(s)\,ds.$$

<u>LEMA 3.1</u>

$$(3.9) \qquad C_{i,n}(s) = \begin{cases} 1 & \text{se } i = 1 \\ s & \text{se } i = 2 \\ [a_{i-1}(n+1)s + \dfrac{\psi_{i-2}(n)}{\psi_{i-1}(n+1)}]C_{i-1,n}(s) & \text{se } i \geq 3. \end{cases}$$

A prova é imediata por indução sobre i.

<u>Definição 3.2</u>

$$C_{i,n}^{(-q)}(s) = \int_0^s \int_0^{s_{q-1}} \cdots \int_0^{s_1} C_{i,n}(s_0)\,d_{s_0}\,d_{s_1} \ldots d_{s_{q-1}}.$$

LEMA 3.2

As funções $C_{i,n}^{(-q)}(s)$ satisfazem:

$$(3.10) \qquad C_{i,n}^{(-q)}(s) = [a_{i-1}(n+1)*s + \frac{\psi_{i-2}(n)}{\psi_{i-1}(n+1)}]C_{i-1,n}^{(-q)}(s) - qa_{i-1}(n+1)C_{i-1,n}^{(-q-1)}(s).$$

A prova se faz facilmente por indução sobre q (ver [16, p. 80 e 81]).

Este lema será usado no cálculo de $C_{i,n}^{(-1)}(1)$ que aparece em (3.8).

Colocando $g_{i,q} = (q-1)!C_{i,n}^{(-q)}(1)$ temos:

$$(3.11) \qquad g_{i,q} = g_{i-1,q} - a_{i-1}(n+1)g_{i-1,q+1}, \qquad i \geq 3,$$

que é uma conseqüência imediata do lema 3.2. Da definição 3.2 e de (3.9) obtemos:

Observação 3.1

$$C_{1,n}^{(-1)}(s) = \int_0^s 1ds_o = s, \quad \text{portanto} \quad C_{1,n}^{(-q)}(s) = \frac{s^q}{q!}$$

$$C_{2,n}^{(-1)}(s) = \int_0^s s_o ds_o = \frac{s^2}{2}, \quad \text{portanto} \quad C_{2,n}^{(-q)}(s) = \frac{s^{(q+1)}}{(q+1)!}.$$

Podemos então escrever:

$$(3.12)\quad g_{i,q} = \begin{cases} \dfrac{1}{q} & \text{se } i = 1 \\[2ex] \dfrac{1}{q(q+1)} & \text{se } i = 2 \\[2ex] g_{i-1,q} - a_{i-1}(n+1)g_{i-1,q+1} & \text{se } i \geq 3 . \end{cases}$$

<u>Observação 3.2</u> Podemos provar que $g_{i,q} > 0$ e $g_{i,q} < g_{i-1,q}$ ([16, p. 306]). Além disso $g_{i,1} = C_{i,n}^{(-1)}(1)$ e portanto

$$p_{n+1} = y_n + h_{n+1} \sum_{i=1}^{k} g_{i,1}\phi_i^*(n) .$$

<u>Observação 3.3</u> Se o passo for constante então

$$g_{i,q} = g_{i-1,q} - \frac{g_{i-1,q+1}}{(i-1)} \quad \text{e} \quad \gamma_{i-1} = \int_0^1 C_{i,n}(s)\,ds,$$

onde γ_{i-1} é como em (1.14), e portanto

$$\gamma_{i-1} = g_{i,1}$$

o que constitui uma maneira simples de calcular os coeficientes dos métodos de Adams com passo constante.

Passamos agora à construção dos coeficientes para a correção. Usaremos o corretor com ordem um a mais que a ordem do previsor.

Seja o método:

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} P^*_{k+1,n}(t)\,dt\,.$$

$P^*_{k+1,n}(x)$ interpola a função f sobre os mesmos pontos que $P_{k,n}(x)$ dado em (3.1) e mais um ponto extra (x_{n+1}, p_{n+1}). Então,

$$(3.13)\qquad P^*_{k+1,n}(x) = P_{k,n}(x) + $$

$$+ (x-x_n)(x-x_{n-1})\ldots(x-x_{n-k+1})f^p[x_{n+1},\ldots,x_{n-k+1}] =$$

$$= P_{k,n}(x)+(sh_{n+1})(sh_{n+1}+h_n)\ldots(sh_{n+1}+\ldots+h_{n-k+2})f^p[x_{n+1},\ldots,x_{n-k+1}]=$$

$$= P_{k,n}(x) + C_{k+1,n}(s)\phi^p_{k+1}(n+1),$$

onde ϕ^p_{k+1} é como na definição 3.1. Integrando (3.13) vem:

$$(3.14)\qquad y_{n+1} = y_n + h_{n+1}\int_0^1 P_{k,n}(x_n+sh_{n+1})\,ds +$$

$$+ h_{n+1}\int_0^1 C_{k+1,n}(s)\,\phi^p_{k+1}(n+1)\,ds =$$

$$= p_{n+1} + h_{n+1}g_{k+1,1}\phi^p_{k+1}(n+1).$$

Notemos que $g_{k+1,1}$ pode ser calculado já no passo de previsão, juntamente com os demais coeficientes $g_{i,1}$.

Quando usamos previsor e corretor de mesma ordem temos uma fórmula semelhante a (3.14). De fato, seja $P^*_{k,n}(x)$ o polinômio que interpola f sobre os pontos $x_n, x_{n-1}, \ldots, x_{n+2-k}, x_{n+1}$ então podemos escrever:

$$P^*_{k,n}(x) = P_{k,n}(x) + (x-x_n)\ldots(x-x_{n+2-k}) * (f^p[x_{n+1},\ldots,x_{n+2-k}] +$$

$$- f[x_n,\ldots,x_{n-k+1}]) =$$

$$= P_{k,n}(x) + (x-x_n)\ldots(x-x_{n+2-k})(x_{n+1}-x_{n-k+1}) * f^p[x_{n+1},\ldots,x_{n-k+1}] =$$

$$= P_{k,n}(x) + (sh_{n+1})(sh_{n+1}+h_n)\ldots(sh_{n+1}+h_n+\ldots+h_{n-k+3}) *$$

$$* \ (h_{n+1}+h_n+\ldots+h_{n-k+2}) \frac{\phi^p_{k+1}(n+1)}{\psi_1(n+1)\ldots\psi_k(n+1)} =$$

$$= P_{k,n}(x) + \left(\frac{sh_{n+1}}{\psi_1(n+1)}\right)\left(\frac{sh_{n+1}+\psi_1(n)}{\psi_2(n+1)}\right)\ldots\left(\frac{sh_{n+1}+\psi_{k-2}(n)}{\psi_{k-1}(n+1)}\right) *$$

$$* \ \psi_k(n+1) \frac{\phi^p_{k+1}(n+1)}{\psi_1(n+1)\ldots\psi_k(n+1)} \psi_1(n+1)\ldots\psi_{k-1}(n+1) =$$

$$= P_{k,n}(x) + C_{k,n}(s)\phi^p_{k+1}(n+1).$$

Denotando a aproximação obtida usando o previsor e o corretor de ordem k por $y_{n+1}(k)$ temos, como anteriormente, que:

$$(3.15) \qquad y_{n+1}(k) = p_{n+1} + h_{n+1} g_{k,1} \phi^p_{k+1}(n+1) .$$

Construímos assim fórmulas para o cálculo dos coeficientes $g_{i,1}$ dos métodos PC com métodos de Adams, bem como fórmulas para a obtenção do valor corrigido a partir do valor previsto (fórmulas 3.14 e 3.15).

Resta-nos agora o cálculo e armazenamento de $\phi^p_{k+1}(n+1)$ e $\phi_i(n+1)$, para que tenhamos completado um passo.

$$\phi^p_{i+1}(n+1) = \psi_1(n+1)\ldots\psi_i(n+1)f^p[x_{n+1},\ldots,x_{n+1-i}], i\geq 1$$

$$= \psi_1(n+1)\ldots\psi_i(n+1)\left(\frac{f^p[x_{n+1},\ldots,x_{n+2-i}] - f[x_n,\ldots,x_{n+1-i}]}{x_{n+1} - x_{n+1-i}}\right).$$

Agora $x_{n+1} - x_{n+1-i} = \psi_i(n+1)$ e portanto temos:

$$\phi^p_{i+1}(n+1) = \psi_1(n+1)\ldots\psi_{i-1}(n+1)f^p[x_{n+1},\ldots,x_{n+2-i}] +$$

$$- \frac{\psi_1(n+1)\ldots\psi_{i-1}(n+1)}{\psi_1(n)\ \ldots\ \psi_{i-1}(n)}\psi_1(n)\ldots\psi_{i-1}(n)f[x_n,\ldots,x_{n+1-i}].$$

Usando a definição 3.1 vem:

$$\phi^p_{i+1}(n+1) = \phi^p_i(n+1)-b_i(n+1)\phi_i(n) = \phi^p_i(n+1)-\phi^*_i(n) \quad , \quad i \geq 1. \tag{3.16}$$

Para $i = 0$ $\phi^p_1(n+1) = f(x_{n+1},p_{n+1}) = f^p_{n+1}$ e temos uma maneira de calcular $\phi^p_i(n+1)$, $i = 1,2,\ldots,k+1$.

Com um raciocínio análogo podemos mostrar que:

$$\phi_1(n+1) = f_{n+1}$$

(3.17)

$$\phi_{i+1}(n+1) = \phi_i(n+1) - \phi_i^*(n) \ , \qquad i = 1,2,\ldots,k.$$

(3.16) e (3.17) são fórmulas simples para o cálculo de $\phi_i^p(n+1)$ e $\phi_i(n+1)$, mas podemos melhorá-las do ponto de vista de armazenamento na memória do computador. A idéia é formar $\phi_i^*(n)$ e armazená-lo sobre $\phi_i(n)$, então formar $\phi_i^p(n+1)$ e armazená-lo sobre $\phi_i^*(n)$ e por último formar $\phi_i(n+1)$ e armazená-lo sobre $\phi_i^p(n+1)$, o que não é possível usando (3.16) e (3.17) diretamente.

De (3.16) deduzimos:

$$\phi_1^p(n+1) = f_{n+1}^p$$

$$\phi_2^p(n+1) = \phi_1^p(n+1) - \phi_1^*(n) = f_{n+1}^p - \phi_1^*(n)$$

.....

$$\phi_{k+1}^p(n+1) = \phi_k^p(n+1) - \phi_k^*(n) = f_{n+1}^p - \sum_{i=1}^{k} \phi_i^*(n) = f_{n+1}^p - p'_{n+1}$$

de modo que $\phi_{k+1}^p(n+1)$ pode ser calculado independentemente. Em seguida este valor será utilizado no cálculo de $\phi_i^p(n+1)$, $i=k,\ldots,1$, a partir de:

(3.18) $$\phi_i^p(n+1) = \phi_{i+1}^p(n+1) + \phi_i^*(n) \ .$$

Por outro lado a diferença entre (3.17) e (3.16) resulta em:

$$\phi_{i+1}(n+1) - \phi^p_{i+1}(n+1) =$$

$$= \phi_i(n+1) - \phi^p_i(n+1) = \ldots = \phi_1(n+1) - \phi^p_1(n+1) = f_{n+1} - f^p_{n+1}$$

e então $\phi_i(n+1) - \phi^p_i(n+1) = f_{n+1} - f^p_{n+1}$ de onde obtemos:

$$\phi_i(n+1) = \phi^p_i(n+1) + (f_{n+1} - \phi^p_1(n+1)) \qquad i = 1,2,\ldots \tag{3.19}$$

Ficam assim resolvidos os problemas de cálculo e armazenamento dos ϕ^p_i e ϕ_i.

Observemos que apenas o valor de $\phi^p_{k+1}(n+1)$ é necessário para completar o passo que calcula y_{n+1}, mas a nossa preocupação em calcular mais valores de $\phi^p_i(n+1)$ e $\phi_i(n+1)$ é devida ao fato que os primeiros serão usados para avaliar o erro de truncamento local, como veremos no próximo capítulo, e os demais serão usados no previsor do próximo passo conforme (3.8). Como veremos no próximo capítulo somente alguns valores de $\phi^p_i(n+1)$ são necessários para avaliar o erro de truncamento local.

Se observamos a construção de $\phi^p_{k+1}(n+1)$ vemos que para o seu cálculo é necessária uma avaliação da função f. Existe porém uma maneira de evitar tal avaliação o que resultará num modo mais eficiente de gerar os valores ϕ^p_i e ϕ_i.

Como dissemos anteriormente $P_{k,n}(x)$ interpola f sobre os pontos x_{n+1-i}, $i = 1,2,\ldots,k$ e definimos $p'_{n+1} = P_{k,n}(x_{n+1})$. Podemos agora ver $P_{k,n}(x)$ como o polinômio interpolador de f sobre os pontos (x_{n+1}, p'_{n+1}), $(x_n, f_n), \ldots, (x_{n+1-k}, f_{n+1-k})$ e considerar como na definição 3.1, $\phi_i^e(n+1)$, para este polinômio. Análoga à prova de (3.16) é a prova de:

$$\phi_{i+1}^e(n+1) = \phi_i^e(n+1) - \phi_i^*(n) \qquad i = 1,2,\ldots,k$$

ou como anteriormente.

$$\phi_i^e(n+1) = \phi_{i+1}^e(n+1) + \phi_i^*(n) \qquad i = k,\ldots,1 \tag{3.20}$$

e precisamos agora calcular $\phi_{k+1}^e(n+1)$. Mas notemos que como $P_{k,n}(x)$ interpola sobre k pontos, este polinômio tem grau no máximo k-1 e portanto as diferenças divididas de ordem maior que k-1 são nulas, ou seja,

$$\phi_{k+1}^e(n+1) = \psi_1(n+1)\ldots\psi_k(n+1) f^e[x_{n+1},\ldots,x_{n+1-k}] = 0,$$

e com este resultado (3.20) gera os demais $\phi_i^e(n+1)$.

Com o mesmo raciocínio usado para encontrar (3.19) obtemos:

$$\phi_i^p(n+1) = \phi_i^e(n+1) + (f_{n+1}^p - \phi_1^e(n+1)) \qquad \text{e} \tag{3.21}$$

$$\phi_i(n+1) = \phi_i^e(n+1) + (f_{n+1} - \phi_1^e(n+1)). \tag{3.22}$$

O algorítmo abaixo representa a sintetização do que acabamos de descrever:

a) Cálculo de $g_{i,1}$ $\quad i = 1,2,\ldots,k+1$

b) Passo Previsor

$$\begin{cases} \phi_i^*(n) = b_i(n+1)\phi_i(n) & i = 1,2,\ldots,k \\ p_{n+1} = y_n + h_{n+1} \sum_{i=1}^{k} g_{i,1}\phi_i^*(n) & \\ \phi_{k+1}^e(n+1) = 0 & \\ \phi_{i+1}^e(n+1) = \phi_i^e(n+1)+\phi_i^*(n) & i=k,k-1,\ldots,1 \end{cases}$$

c) Avaliação $\quad f_{n+1}^p = f(x_{n+1},p_{n+1})$

d) Passo corretor $\quad y_{n+1} = p_{n+1} + h_{n+1}g_{k+1,1}(f_{n+1}^p-\phi_1^e(n+1))$

e) Avaliação $\quad f_{n+1} = f(x_{n+1},y_{n+1})$

f) preparação para o próximo passo

$$\begin{cases} \phi_{k+1}(n+1) = f_{n+1} - \phi_1^e(n+1) \\ \phi_i(n+1) = \phi_i^e(n+1)+\phi_{k+1}(n+1) & i = k,\ldots,1 \end{cases}.$$

Dissemos anteriormente que se n_s passos sucessivos (incluindo o passo atual para passar de x_n para x_{n+1}) são tomados com tamanho de passo constante h, o custo computa-

cional para o cálculo dos coeficientes $g_{i,1}$ é diminuído. Passamos agora à justificativa desse fato.

Por (3.3) temos:

$$\psi_i(n+1) = \psi_i(n) \qquad i = 1,2,\ldots,n_s-1$$

e por (3.2) $\psi_{n_s}(n+1) = \psi_{n_s-1}(n+1)+h = n_s h.$

Logo para o cálculo de $\psi_i(n+1)$ basta começar com ψ_{n_s} e calcular os demais por:

$$\psi_i(n+1) = \psi_{i-1}(n)+h \qquad i = n_s+1,\ldots,k.$$

Analogamente é fácil ver que:

$$a_i(n+1) = a_i(n), \quad b_i(n+1) = b_i(n) \qquad i = 1,\ldots,n_s-1 \tag{3.23}$$

e os demais podem ser obtidos por:

$$a_{n_s}(n+1) = \frac{1}{n_s}, \quad a_i(n+1) = \frac{h}{\psi_i(n+1)} \qquad i=n_s+1,\ldots,k$$

$$b_{n_s}(n+1) = 1, \quad b_i(n+1) = b_{i-1}(n+1)\frac{\psi_{i-1}(n+1)}{\psi_{i-1}(n)} \qquad i=n_s+1,\ldots,k.$$

Em virtude de (3.23) não precisamos recalcular os coeficientes $g_{i,q}$ de (3.12) para $i = 1,2,\ldots,n_s$.

Representando a matriz de $g_{i,q}$ como:

(3.24)

q \ i	2	3		n_s	n_s+1		$k+1$
1	x	x		x	x		x
2	x	⋮		x	⋮		
⋮	⋮			⋮	x		
$k-n_s+2$				x			
$k-1$		x					
k	x	⊗					
$k+1$	⊗						

precisamos para completá-la, somente gerar as colunas n_s+1 até $k+1$. Notemos que na coluna j temos $k-j+2$ elementos e que o número de elementos na figura entre as linhas cheias é sempre k, independente de n_s. Definimos então o vetor V colocando:

$$V(k) = g_{2,k} \qquad\qquad V(k+1-n_s) = g_{n_s,k+1-n_s}$$

$$V(k-1) = g_{3,k-1}$$

$$\vdots \qquad\qquad\qquad\qquad \vdots$$

$$V(k+3-n_s) = g_{n_s-1,k+3-n_s} \qquad\qquad V(2) = g_{n_s,2}$$

$$V(k+2-n_s) = g_{n_s,k+2-n_s} \qquad\qquad V(1) = g_{n_s,1}$$

Notemos que o vetor V contém, nas posições $1,2,\ldots,k-n_s+2$, as informações necessárias para continuar a geração dos $g_{i,q}$ e nas demais as informações necessárias para o caso de precisarmos aumentar a ordem de k para $k+1$.

Nós atualizamos $V(q)$ para este passo formando:

$$(3.25) \qquad g_{n_s+1,q} = V(q) - a_{n_s}(n+1)V(q+1), \quad q = 1,2,\ldots,k+1-n_s$$

e armazenando esses valores no próprio vetor $V(q)$, $q = 1,2,\ldots,k+1-n_s$, isto é ,$V(q) \leftarrow g_{n_s+1,q}$.

Dessa forma o vetor V contém a coluna n_s+1 da figura, nas posições $1,2,\ldots,k+1-n_s$, e está preparado para o caso do próximo passo usar o mesmo valor para h. Para completar a geração dos $g_{i,1}$ copiamos o vetor $V(q)$ num vetor de trabalho $W(q)$, $q = 1,2,\ldots,k+1-n_s$. Cada novo valor $g_{i,q}$ dado por:

$$(3.26) \qquad g_{i,q} = W(q) - a_{i-1}(n+1)W(q+1)$$

é armazenado em $W(q)$ para $q = 1,2,\ldots,k+2-i$ e assim $g_{i,1}=W(1)$ para cada $i = n_s+2,\ldots,k+1$.

Mudanças na ordem correspondem a adicionar ou subtrair um ponto de interpolação.

Se a ordem for diminuída nenhuma atitude especial deve ser tomada desde que a análise acima independe da ordem que foi usada no passo anterior. Se a ordem for aumentada , diga-

mos $k = k_{atual} = k_{anterior} + 1$, então devemos calcular mais os elementos marcados com $\otimes$ na figura anterior; para isso atualizamos o vetor V segundo:

$$V(k) = g_{2,s}$$

$$V(k-j) \leftarrow V(k-j) - a_{j+1}(n+1)V(k-j+1) \quad j = 1,2,\ldots,n_s-2$$

e voltamos em (3.25).

Caso n_s seja maior que a ordem do método a ser usado então os coeficientes $g_{i,1}$ não necessitam ser recalculados. Quando o passo for modificado ($n_s = 1$) todos os coeficientes $g_{i,1}$ devem ser recalculados. Podemos neste caso ainda aplicar (3.25) e (3.26) começando com $V(q)=g_{2,q}$, $q=1,2,\ldots,k$.

Em resumo quando n_s passos são tomados com tamanho constante h, reduzimos o custo computacional do cálculo dos coeficientes $g_{i,1}$ que representa grande parte do custo total. Ainda, como armazenamos $\phi_i^*(n)$ sobre $\phi_i(n)$ e $\phi_i^*(n) = b_i(n+1)\phi_i(n)$ com $b_i(n+1) = 1$ para $i = 1,2,\ldots,n_s$ os $\phi_i^*(n)$ devem ser calculados somente para $i = n_s+1,\ldots,k$, o que reduz também o custo computacional, principalmente quando trabalhamos com muitas equações no sistema.

As rotinas STEP e DE dadas em [16] são tais que, dados $(a,y(a))$ e um ponto z, elas devolvem uma aproximação para $y(z)$. Essas rotinas escolhem o tamanho do passo tão grande quanto possível desde que mantendo o erro de truncamento local menor que certa tolerância especificada pelo usuário.

Como conseqüência, certamente será ultrapassado o ponto z onde a solução é desejada, e esta será então aproximada usando interpolação polinomial. Esta estratégia se mostra mais eficiente que diminuir o tamanho do passo para atingir z, uma vez que normalmente a solução é desejada em outros pontos mais avançados. Daremos a seguir os detalhes do cálculo do polinômio interpolador referido.

Suponhamos que já conhecemos aproximações em x_{n+2-i} $i = 1,2,\ldots,k+1$ e que $x_n \leq z \leq x_{n+1}$. Seja então $P_{k+1,n+1}(x)$ tal que $P_{k+1,n+1}(x_{n+2-i}) = f_{n+2-i}$ e definamos:

$$h_I = z - x_{n+1}\ , \qquad s = (x - x_{n+1})/h_I .$$

Usando a definição 3.1, um termo genérico de $P_{k+1,n+1}(x)$ pode ser escrito como:

$$\left(\frac{sh_I}{\psi_1(n+1)}\right)\left(\frac{sh_I+\psi_1(n+1)}{\psi_2(n+1)}\right)\ldots\left(\frac{sh_I+\psi_{i-2}(n+1)}{\psi_{i-1}(n+1)}\right)\phi_i(n+1) .$$

Introduzimos então, como anteriormente, a notação:

$$C^I_{i,n+1}(s) = \begin{cases} 1 & \text{se } i = 1 \\ \dfrac{sh_I}{\psi_1(n+1)} & \text{se } i = 2 \\ \left(\dfrac{sh_I}{\psi_1(n+1)}\right)\left(\dfrac{sh_I+\psi_1(n+1)}{\psi_2(n+1)}\right)\ldots\left(\dfrac{sh_I+\psi_{i-2}(n+1)}{\psi_{i-1}(n+1)}\right) & i \geq 3 \end{cases}$$

e obtemos:

$$y(z) \simeq y_z = y_{n+1} + h_I \sum_{i=1}^{k+1} \phi_i(n+1) \int_0^1 C^I_{i,n+1}(s)ds.$$

Definindo:

$$\Gamma_i(s) = \begin{cases} \dfrac{sh_I}{\psi_1(n+1)} & \text{se } i = 1 \\[2ex] \dfrac{sh_I + \psi_{i-1}(n+1)}{\psi_i(n+1)} & \text{se } i \geq 2 \end{cases}$$

segue:

$$C^I_{i,n+1}(s) = \Gamma_i(s) C^I_{i-1,n+1}(s).$$

Considerando $C^{I(-q)}_{i,n+1}(s)$ como na definição 3.2, o uso de integração por partes permite deduzir:

$$C^{I(-q)}_{i,n+1}(s) = \Gamma_{i-1}(s) C^{I(-q)}_{i-1,n+1}(s) - \frac{qh_I}{\psi_{i-1}(n+1)} C^{I(-q-1)}_{i-1,n+1}(s).$$

Definindo $g^I_{i,q} = (q-1)! C^{I(-q)}_{i,n+1}(1)$ temos a recursão:

$$g^I_{i,q} = \begin{cases} \dfrac{1}{q} & \text{se } i = 1 \\[2ex] \Gamma_{i-1}(1) g^I_{i-1,q} - \eta_{i-1} g^I_{i-1,q+1} & i \geq 2 \end{cases}$$

onde $\eta_i = \dfrac{h_I}{\psi_i(n+1)}$.

Seguindo agora o raciocínio utilizado para encontrar as expressões de p_{n+1} e p'_{n+1}. Obtemos:

$$y_z = y_{n+1} + h_I \sum_{i=1}^{k+1} g^I_{i,1}\phi_i(n+1) \quad \text{e}$$

$$y'_z = \sum_{i=1}^{k+1} \rho_i \phi_i(n+1)$$

onde $$\rho_i = \begin{cases} 1 & \text{se } i = 1 \\ \rho_{i-1}\Gamma_{i-1}(1) & \text{se } i \geq 2. \end{cases}$$

Notemos que para calcular y_z precisamos obter os coeficientes $g^I_{i,q}$, mas não precisamos recalcular os $\phi_i(n+1)$, pois já são conhecidos do passo corretor, e assim o cálculo de y_z não envolve nenhuma nova avaliação da função f. Este é um fato muito importante, pois faz com que o programa seja pouco sensível ao número de pontos z onde perguntamos pela solução.

3.2. *Implementação dos Métodos de Adams com a Técnica da Interpolação*

A rotina Gear dada em [9] implementa a técnica da interpolação usando os métodos de Adams na forma matricial de Nordsieck.

Por (1.27) temos:

$$Y_{n+1(o)} = AC_nY_n$$

$$Y_{n+1(m+1)} = Y_{n+1(m)} + \ell G(Y_{n(m)}) \qquad m = 0,1,\ldots,M$$

onde $\quad Y_n = (y_n, h_n y'_n, \ldots, \dfrac{h_n^k y_n^{(k)}}{k!})^t$

e k é a ordem do método.

É claro que o vetor ℓ depende de k e em [3, p. 154] ele é dado para métodos até ordem 8.

A matriz A acima é a matriz do triângulo de Pascal e com base nesta particularidade é dada a seguir uma maneira econômica, do ponto de vista de custo computacional, de efetuar a multiplicação $Y_{n+1(o)} = AY_n$ (onde estamos identificando C_nY_n com Y_n).

Calculamos sucessivamente:

1) $y_{ni} \leftarrow y_{ni} + y_{n,i+1} \qquad i = k,\ldots,1$

2) $y_{ni} \leftarrow y_{ni} + y_{n,i+1} \qquad i = k,\ldots,2$

$\vdots$

k-1) $y_{ni} \leftarrow y_{ni} + y_{n,i+1} \qquad i = k,k-1$

k) $y_{nk} \leftarrow y_{nk} + y_{n,k+1}$

onde y_{ni} é a i-ésima coordenada do vetor Y_n. É claro que no algorítmo acima o vetor $Y_{n+1(o)}$ é gerado sobre o vetor Y_n. Vemos que efetuamos o produto desejado envolvendo apenas somas e armazenamentos, e são requeridas exatamente k*(k+1)/2 de cada uma dessas operações.

A implementação dos passos de correção do método acima é feita normalmente, sem considerações especiais.

Como pudemos observar, a implementação da técnica da interpolação é muito mais simples que a da técnica do passo variável, devido ao fato que na primeira os métodos têm seus coeficientes dados à priori, não sendo necessário calculá-los à todo passo como acontece no outro caso.
Entretanto essa vantagem é somente aparente, visto que a técnica da interpolação é mais instável, e além disso quase todas as operações envolvidas são operações com matrizes, o que acarreta um custo computacional maior do que parece à primeira vista.

CAPÍTULO IV

ESTIMATIVAS E CONTROLE DO ERRO DE TRUNCAMENTO LOCAL

Os teoremas do final do capítulo 2 nos garantem que o erro global E_n satisfaz: $||E_n|| \leq k_2||E_o|| + \sum_{i=o}^{k-1} k_2||d_i||$.

Assim, se controlamos os erros de truncamento local d_i mantendo-os menores que um certo valor ε a cada passo, poderemos a grosso modo controlar o erro global. Isto no entanto não é totalmente verdadeiro, pois a constante k_2, que depende do problema a ser resolvido, pode ser grande. Nos capítulos subsequentes suporemos sempre que tal não ocorre, isto é, estaremos resolvendo os chamados problemas não "STIFF".

Mudanças no passo e na ordem serão usadas para que o erro de truncamento local d_i se mantenha menor que ε.

Na técnica da interpolação (rotina GEAR) usamos métodos para passo constante e portanto uma estimativa para o erro de truncamento local é dada por $C_{k+1}h^{k+1}y^{(k+1)}(\theta)$, ver [13,p.90].

No entanto, para os métodos PC com a técnica do passo variável (rotinas STEP/DE), estimativas para o erro de truncamento local são inviáveis. Por esta razão definimos um outro tipo de erro mais natural e compatível com a prática, o qual tentamos estimar e controlar.

4.1. *Estimativas Para o Erro Local de Ordem* k,k-1,k-2

<u>Definição 4.1</u> Seja $u_n(x)$ solução de:

$$u_n'(x) = f(x,u_n(x)), \quad u_n(x_n) = y_n \tag{4.1}$$

Diremos que $u_n(x)$ é a solução local do problema (0.5).

<u>Definição 4.2</u> $LE_{n+1} = u_n(x_{n+1}) - y_{n+1}$ será dito erro local do método pelo qual calculamos y_{n+1}.

Notemos que um método numérico tenta na verdade aproximar a solução $u_n(x)$ ao invés de $y(x)$ e portanto o erro local é uma quantidade mais natural de ser controlada que o erro de truncamento local. Se $y_n = y(x_n)$ então $u_n(x) = y(x)$ e portanto $LE_{n+1} = E_{n+1}$ (erro global).

<u>Definição 4.3</u> Diremos que o erro local LE_{n+1} é controlado por:

a) Unidade de passo se $|LE_{n+1}| \leq \varepsilon h_{n+1}$

b) Unidade de passo no sentido generalizado se $|LE_{n+1}| \leq \sigma h_{n+1}\varepsilon$, onde σ é uma constante, em geral não conhecida

c) passo se $|LE_{n+1}| \leq \varepsilon$.

Provaremos a seguir um teorema que nos garante que, se controlamos o erro local por unidade de passo no sentido

generalizado, então o erro global E_{n+1} será proporcional a ε.

TEOREMA 4.1

Se $|LE_{k+1}| \leq \sigma\varepsilon h_{k+1}$ $k = 0,1,\ldots,n$, então

$|E_{n+1}| \leq \varepsilon\sigma(x_{n+1} - a)\exp(L(x_{n+1} - a))$.

Prova:

Aplicando o teorema 0.4 para os PVI's (0.5) e (4.1) tem-se:

$$|y(x) - u_n(x)| \leq |y(x_n) - y_n| * \exp(L(x-a)).$$

Para $x = x_{n+1}$ segue que:

$$|y(x_{n+1})-y_{n+1}|-|u_n(x_{n+1})-y_{n+1}| \leq |y(x_{n+1})-u_n(x_{n+1})|$$

$$|y(x_{n+1})-y_{n+1}|-|LE_{n+1}| \leq |y(x_n)-y_n|\exp(Lh_{n+1}).$$

Finalmente usando a hipótese temos:

$$|E_{n+1}| \leq |E_n|\exp(Lh_{n+1}) + \varepsilon\sigma h_{n+1}. \tag{4.2}$$

A prova acima foi feita para E_{n+1} por conveniência de notação mas é claro que (4.2) permanece válido para $E_n, E_{n-1}, \ldots, E_o = 0$. De posse desse resultado prova-se facilmente por indução sobre n que:

(4.3) $$|E_{n+1}| \leq \sum_{j=1}^{n} \sigma\varepsilon h_j \exp(L \sum_{i=j}^{n} h_{i+1}) + \sigma\varepsilon h_{n+1}.$$

Por outro lado vale:

$$\exp(L \sum_{i=j}^{n} h_{i+1}) \leq \exp(L(x_{n+1}-a))$$

e portanto (4.3) toma a forma:

$$|E_{n+1}| \leq \sum_{j=1}^{n+1} \sigma\varepsilon h_j \exp(L(x_{n+1}-a)) =$$

$$= \sigma\varepsilon \exp(L(x_{n+1}-a)) \sum_{j=1}^{n+1} h_j = \varepsilon\sigma(x_{n+1}-a)\exp(L(x_{n+1}-a))$$

o que demonstra o teorema.

Usaremos no que segue as notações:

$p_{n+1}(k-1)$ é o valor previsto em x_{n+1} por um previsor de ordem $k-1$.

p_{n+1} é o valor previsto por um previsor de ordem k.

$y_{n+1}(k)$ é a aproximação obtida usando previsor e corretor de ordem k.

y_{n+1} é a aproximação obtida usando um previsor de ordem k e corretor de ordem $k+1$.

<u>Definição 4.4</u> Definimos $\ell e_{n+1}(k)$ colocando:

(4.4) $$\ell e_{n+1}(k) = u_n(x_{n+1}) - y_{n+1}(k).$$

Esta nova quantidade será uma aproximação para LE_{n+1} e nos auxiliará na seleção do novo tamanho de passo.

É de se esperar que y_{n+1} aproxime melhor $u_n(x_{n+1})$ que $y_{n+1}(k)$. Baseados neste fato heurístico podemos construir uma estimativa para ℓe_{n+1}, comparando os resultados de fórmulas de ordens diferentes. Escrevemos:

$$(4.5) \qquad y_{n+1}-y_{n+1}(k) = [u_n(x_{n+1})-y_{n+1}(k)]-[u_n(x_{n+1})-y_{n+1}] =$$

$$= \ell e_{n+1}(k)-[u_n(x_{n+1})-y_{n+1}] \simeq \ell e_{n+1}(k).$$

Por (3.14) e (3.15) temos:

$$y_{n+1} = p_{n+1} + h_{n+1}g_{k+1,1}\phi^p_{k+1}(n+1)$$

$$y_{n+1}(k) = p_{n+1} + h_{n+1}g_{k,1}\phi^p_{k+1}(n+1)$$

e portanto (4.5) torna-se:

$$(4.6) \qquad \ell e_{n+1}(k) \simeq h_{n+1}(g_{k+1,1} - g_{k,1})\phi^p_{k+1}(n+1).$$

Temos assim uma estimativa para $\ell e_{n+1}(k)$. De posse dessa estimativa nada mais natural que somá-la à aproximação $y_{n+1}(k)$ para obter uma nova aproximação que acreditamos ser melhor (este processo é conhecido como extrapolação local):

$$(4.7) \qquad y_{n+1} = y_{n+1}(k) + (y_{n+1}-y_{n+1}(k)) \simeq y_{n+1}(k) + \ell e_{n+1}(k).$$

Daremos a seguir uma justificativa para a estimativa (4.5) de $\ell e_{n+1}(k)$. Mostraremos também que estimativas para $\ell e_{n+1}(k-1)$ e $\ell e_{n+1}(k-2)$ podem ser obtidas usando somente as informações disponíveis devido ao cálculo de y_{n+1}. A estimativa para $\ell e_{n+1}(k+1)$ só será possível se hipóteses adicionais forem feitas.

Consideremos inicialmente o erro no valor previsto p_{n+1}. Sejam $P_{k,n}(x)$ como em (1.1) e $\tilde{P}_{k,n}(x)$ tal que:

$$\tilde{P}_{k,n}(x_{n+1-j}) = f(x_{n+1-j}, u_n(x_{n+1-j})) \qquad j = 1,\ldots,k. \tag{4.8}$$

Então

$$u_n(x_{n+1}) - p_{n+1} = \int_{x_n}^{x_{n+1}} (u_n'(t) - P_{k,n}(t))dt = \tag{4.9}$$

$$= \int_{x_n}^{x_{n+1}} (u_n'(t) - \tilde{P}_{k,n}(t))dt + \int_{x_n}^{x_{n+1}} (\tilde{P}_{k,n}(t) - P_{k,n}(t))dt.$$

A primeira integral é o erro de truncamento local para a solução $u_n(x)$ e usando (1.8) temos que, se o método é de ordem k, então este erro é de $O(H^{k+1})$. A segunda integral é o erro devido aos valores anteriores não serem exatos e usando a fórmula de Lagrange para os polinômios acima podemos escrever:

$$(4.10)\qquad \int_{x_n}^{x_{n+1}} (\tilde{P}_{k,n}(t) - P_{k,n}(t))dt =$$

$$= h_{n+1} \sum_{j=1}^{k} \beta_{jk}[f(x_{n+1-j}, u_h(x_{n+1-j})) - f(x_{n+1-j}, y_{n+1-j})]$$

$$(4.11)\qquad \left|\int_{x_n}^{x_{n+1}} (\tilde{P}_{k,n}(t) - P_{k,n}(t))dt\right| \le |h_{n+1}| L \sum_{j=1}^{k} |\beta_{jk}| |u_n(x_{n+1-j}) - y_{n+1-j}|$$

onde L é a constante de Lipschtz de f, e se os quocientes de passos sucessivos são limitados então os $|\beta_{jk}|$ também o são (teorema 2.3).

Agora,

$$(4.12)\qquad |u_n(x_{n+1-j}) - y_{n+1-j}| \le |u_n(x_{n+1-j}) - y(x_{n+1-j})| + |y(x_{n+1-j}) - y_{n+1-j}|$$

onde, conforme a observação 2.3, se o erro de truncamento local é de $0(H^{k+1})$, então

$$(4.13)\qquad |y(x_{n+1-j}) - y_{n+1-j}| = 0(H^{k+1}), \quad j = 1,\ldots,k.$$

Consideremos novamente os PVI's(0.5) e (4.1); então pelo teorema 0.4 temos:

$$|u_n(x) - y(x)| \le |y_n - y(x_n)| \exp(L|x - x_n|)$$

e portanto podemos escrever:

$$|u_n(x_{n+1-j})-y(x_{n+1-j})| \le |y_n-y(x_n)|\exp(LkH).$$

Usando agora (4.13) com $j = 1$ vem:

$$(4.14) \qquad |u_n(x_{n+1-j}) - y(x_{n+1-j})| = 0(H^{k+1})$$

Levando (4.14) e (4.13) em (4.12) e esta a (4.11) obtemos:

$$(4.15) \qquad \left|\int_{x_n}^{x_{n+1}} (\tilde{P}_{k,n}(t)-P_{k,n}(t))dt\right| \le h_{n+1}0(H^{k+1}) = 0(H^{k+2})$$

concluindo a partir de (4.9):

$$(4.16) \qquad u_n(x_{n+1}) - p_{n+1} = 0(H^{k+1}).$$

(4.15) fornece o erro devido ao fato de os valores iniciais para o problema local não serem exatos. Observemos que tem ordem superior à do erro de truncamento local para $u_n(x)$.

Agora podemos encontrar a ordem do erro local do método PC. Sejam $P^*_{k,n}(x)$ e $\tilde{P}_{k,n+1}(x)$ polinômios tais que:

$$P^*_{k,n}(x_{n+1-j}) = f(x_{n+1-j},y_{n+1-j}),\ P^*_{k,n}(x_{n+1}) = f(x_{n+1},p_{n+1})$$

$$\tilde{P}_{k,n+1}(x_{n+1-j}) = u'_n(x_{n+1-j}),\ \tilde{P}_{k,n+1}(x_{n+1}) = u'(x_{n+1}),$$

para $j = 1,2,\ldots,k-1$. Então temos:

$$(4.17) \qquad \ell e_{n+1}(k) = \int_{x_n}^{x_{n+1}} (u'_n(t)-\tilde{P}_{k,n+1}(t))dt + \int_{x_n}^{x_{n+1}} (\tilde{P}_{k,n+1}(t)-P^*_{k,n}(t))dt.$$

Como anteriormente, a primeira integral é o erro de truncamento local para $u_n(x)$ e portanto, devido a (1.8), é de $0(H^{k+1})$. A segunda integral, podemos mostrar como fizemos de (4.10) a (4.15), que é de $0(H^{k+2})$.

Um raciocínio inteiramente análogo mostra que, quando usamos um previsor de ordem k e um corretor de ordem $k+1$, o erro local é $0(H^{k+2})$ e assim temos:

$$\ell e_{n+1}(k) = u_n(x_{n+1}) - y_{n+1}(k) = y_{n+1} - y_{n+1}(k) +$$

$$+ u_n(x_{n+1}) - y_{n+1} = y_{n+1} - y_{n+1}(k) + 0(H^{k+2}).$$

Justificando a estimativa (4.5) para o erro local.

Com procedimento análogo podemos estimar e analisar o erro local em x_{n+1} se o passo de x_n para x_{n+1} foi executado com previsor e corretor de ordens $k-1$.

Temos $\ell e_{n+1}(k-1)$ de $0(H^k)$ e

$$\ell e_{n+1}(k-1) = u_n(x_{n+1}) - y_{n+1}(k-1) = y_{n+1}(k) - y_{n+1}(k-1) + \tag{4.18}$$

$$+ u_n(x_{n+1}) - y_{n+1}(k) = y_{n+1}(k) - y_{n+1}(k-1) + 0(H^{k+1}).$$

O problema agora é estimar $\ell e_{n+1}(k-1)$ sem calcular $y_{n+1}(k-1)$, pois

$$y_{n+1}(k-1) = y_n + h_{n+1} \sum_{j=1}^{k-2} \beta^*_{j,k-1} f_{n+1-j} + h_{n+1} \beta^*_{o,k} f(x_{n+1}, p_{n+1}(k-1)) \tag{4.19}$$

e este cálculo envolveria mais uma avaliação da função $f(x_{n+1},p_{n+1}(k-1))$, o que devemos evitar. Para isso formamos:

$$(4.20)\qquad \bar{y}_{n+1}(k-1) = y_n + h_{n+1}\sum_{j=1}^{k-2}\beta^*_{j,k-1}f_{n+1-j} + h_{n+1}\beta^*_{o,k-1}f^p_{n+1}$$

e vamos provar que $|\bar{y}_{n+1}(k-1) - y_{n+1}(k-1)| = 0(H^{k+1})$.

De fato, devido a (4.19) e (4.20), tem-se:

$$|\bar{y}_{n+1}(k-1)-y_{n+1}(k-1)| = |h_{n+1}\beta^*_{o,k-1}(f(x_{n+1},p_{n+1})-f(x_{n+1},p_{n+1}(k-1)))| \leq$$

$$\leq H|\beta^*_{o,k-1}|L|p_{n+1}-p_{n+1}(k-1)| \leq H|\beta^*_{o,k-1}|L[\,|p_{n+1}-u_n(x_{n+1})|+$$

$$+\,|u_n(x_{n+1}) - p_{n+1}(k-1)|\,].$$

Agora por (4.16) temos:

$$|p_{n+1} - u_n(x_{n+1})| = 0(H^{k+1}) \quad \text{e} \quad |p_{n+1}(k-1)-u_n(x_{n+1})| = 0(H^k)$$

e portanto:

$$(4.21)\qquad |\bar{y}_{n+1}(k-1) - y_{n+1}(k-1)| = 0(H^{k+1}).$$

Por (4.18) temos:

$$(4.22) \qquad \ell e_{n+1}(k-1) = y_{n+1}(k) - y_{n+1}(k-1) + 0(H^{k+1}) =$$

$$= y_{n+1}(k) - \bar{y}_{n+1}(k-1) + 0(H^{k+1}) =$$

$$= h_{n+1}(g_{k,1}-g_{k-1,1})\phi_k^p(n+1) + 0(H^{k+1})$$

e obtivemos uma estimativa para $\ell e_{n+1}(k-1)$ sem a avaliação de $f(x_{n+1}, p_{n+1}(k-1))$.

Com um raciocínio totalmente análogo podemos mostrar que:

$$\ell e_{n+1}(k-2) = h_{n+1}(g_{k-1,1}-g_{k-2,1})\phi_{k-1}^p(n+1) + 0(H^k).$$

É interessante observar que as estimativas para ℓe_{n+1} dependem somente do valor da função f no valor previsto, não sendo portanto necessária a última avaliação do modo PECE para calcular tais estimativas. Isto é um fato muito importante para o caso de o passo não ter sucesso (quando $\ell e_{n+1}(k)$ não satisfaz $|\ell e_{n+1}(k)| \leq \varepsilon$).

4.2. *Estimativa para o Erro Local de Ordem* k+1

Para encontrar uma estimativa para $\ell e_{n+1}(k+1)$ precisamos das hipóteses adicionais que o tamanho do passo seja constante h assim como a ordem k. De maneira análoga à (4.15) e (4.16) valem aqui:

$$u_n(x_{n+1}) - p_{n+1}(k+1) = 0(H^{k+2}) \quad e$$

$$(4.23) \qquad \ell e_{n+1}(k+1) = \int_{x_n}^{x_{n+1}} (u_n'(t) - \tilde{P}_{k+1,n+1}(t))dt +$$

$$+ \int_{x_n}^{x_{n+1}} (\tilde{P}_{k+1,n+1}(t) - P^*_{k+1,n}(t))dt$$

Como anteriormente a primeira integral é o erro de truncamento local para $u_n(x)$ e portanto por (1.16) é de $0(h^{k+2})$.
O problema é que, com o mesmo raciocínio usado na estimativa de $\ell e_{n+1}(k)$, não conseguimos provar que a segunda integral é de $0(h^{k+3})$. Precisamos agora dos seguintes resultados.

<u>TEOREMA 4.2</u>

Sejam h o tamanho do passo, k e $k+1$, as ordens do previsor e do corretor respectivamente. Se h e k são fixos e se $y(x)$ é suficientemente diferenciável, então o erro global do método PC aplicado no modo PECE satisfaz:

$$(4.24) \qquad E_n = h^{k+1}\phi(x_n) + 0(h^{k+2})$$

onde $\phi(x)$ é solução de:

$$(4.25) \qquad \phi'(x) = G(x)\phi(x) + \beta^*_{o,k+1}G(x)\gamma_k y^{(k+1)}(x) + \gamma^*_{k+1}y^{(k+2)}(x)$$

$$\phi(a) = 0$$

com $\quad G(x) = f_y(x,y(x))$

Prova: ver [16, p. 72].

TEOREMA 4.3

Seja f de classe C^1. Então

$$u_n(x) = y(x) + \delta v(x) + 0(\delta^2) \tag{4.26}$$

onde v é solução de:

$$\begin{aligned} v'(x) &= f_y(x,y(x))v(x) \\ v(x_n) &= 1 \end{aligned} \tag{4.27}$$

e $\delta = y_n - y(x_n)$.

Prova:

Se definimos $y(x,A)$ como a solução do problema $y' = f(x,y)$ que no ponto x_n vale A, então pela hipótese sobre f podemos provar que $y(x,A)$ é diferenciável em relação a A.

Seja $v(x,A) = \dfrac{\partial y(x,A)}{\partial A}$.

Então $v(x_n,A) = \dfrac{\partial y(x_n,A)}{\partial A} = \dfrac{\partial A}{\partial A} = 1.$

(4.27) é satisfeito por $v(x,A)$. De fato:

$$\frac{\partial v(x,A)}{\partial A} = \frac{\partial}{\partial x}\ \frac{\partial y(x,A)}{\partial A} = \frac{\partial}{\partial A}\ \frac{\partial y(x,A)}{\partial x} =$$

$$= \frac{\partial}{\partial A}(f(x,y(x,A))) = f_y(x,y(x,A))\frac{\partial}{\partial A}y(x,A) =$$

$$= f_y(x,y(x,A))v(x,A).$$

Como $u_n(x)$ é a solução do problema que em x_n vale y_n temos $u_n(x) = y(x,y_n)$ e analogamente $y(x) = y(x,y(x_n))$. Por Taylor obtemos:

$$u_n(x) = y(x,y_n - y(x_n) + y(x_n)) =$$

$$= y(x,y(x_n))+(y_n-y(x_n))\frac{\partial y(x,y(x_n))}{\partial A} + 0((y_n - y(x_n))^2)$$

Tomando $v(x) = v(x,y(x_n))$ obtemos:

$$u_n(x) = y(x) + \delta v(x) + 0(\delta^2)$$

o que prova o teorema.

Voltando a analisar (4.23) escrevemos:

$$\left|\int_{x_n}^{x_{n+1}} (\tilde{P}_{k+1,n+1}(t)-P^*_{k+1,n}(t))dt\right| \le hL \sum_{j=1}^{k} |\beta^*_{j,k+1}||u_n(x_{n+1-j})-y_{n+1-j}|+$$

$$+hL|\beta^*_{o,k+1}||u_n(x_{n+1})-p_{n+1}(k+1)|$$

e usando o teorema 4.3 temos, para $j = 0,\ldots,k$,

$$u_n(x_{n+1-j})-y_{n+1-j} = y(x_{n+1-j})-y_{n+1-j}+\delta v(x_{n+1-j}) + 0(\delta^2)$$

onde $\delta = y_n - y(x_n)$.

Usando agora o teorema 4,2 temos que:

$$u_n(x_{n+1-j})-y_{n+1-j} = h^{k+1}\phi(x_{n+1-j}) + 0(h^{k+2}) +$$

$$- (h^{k+1}\phi(x_n)+0(h^{k+2}))v(x_{n+1-j}) + 0(\delta^2).$$

Desenvolvendo $v(x_{n+1-j})$ e $\phi(x_{n+1-j})$ por Taylor em torno do ponto x_n vem:

$$v(x_{n+1-j}) = v(x_n) + 0(h) = 1 + 0(h)$$

$$\phi(x_{n+1-j}) = \phi(x_n) + (1-j)h\phi'(x_n) + 0(h^2)$$

e substituindo na expressão anterior temos:

$$u_n(x_{n+1-j})-y_{n+1-j} = h^{k+1}\phi(x_n)+(1-j)h^{k+2}\phi'(x_n) +$$

$$+ 0(h^{k+2})-h^{k+1}\phi(x_n)+0(h^{k+2})+0(\delta^2) = 0(h^{k+2})$$

desde que suponhamos $\delta = 0(h^{k+1})$.

Como já dissemos $u_n(x_{n+1}) - p_{n+1}(k+1) = 0(h^{k+2})$ e então podemos escrever que:

$$\int_{x_n}^{x_{n+1}} (\tilde{P}_{k+1,n+1}(t) - P^*_{k+1,n}(t))dt = 0(h^{k+3})$$

e como conseqüência $\ell e_{n+1}(k+1)$ é de $0(h^{k+2})$.

As hipóteses de que o passo e a ordem sejam constantes não são tão irreais quanto parecem, pois a rotina STEP tem uma forte tendência em tomar grupos de passos com tamanho constante e ordem fixa. Dentro de cada um destes grupos as hipóteses acima são satisfeitas e portanto na prática as aproximações feitas são válidas.

Vamos agora encontrar uma estimativa para o erro $\ell e_{n+1}(k+1)$. Para isso suporemos que temos calculado uma aproximação para $u_n(x_{n+1})$, que chamaremos s_{n+1}, com um previsor de ordem $k+1$ e um corretor de ordem $k+2$. Então uma estimativa para $\ell e_{n+1}(k+1)$ pode ser construída exatamente como para $\ell e_{n+1}(k)$, isto é,

$$\ell e_{n+1}(k+1) = u_n(x_{n+1}) - y_{n+1}(k+1) = s_{n+1} - y_{n+1}(k+1) +$$

$$+ u_n(x_{n+1}) - s_{n+1} = s_{n+1} - y_{n+1}(k+1) + 0(h^{k+3}),$$

onde $y_{n+1}(k+1)$ denota a aproximação obtida por previsor e corretor ambos de ordem $k+1$.

Por (1.18) e (1.19) temos:

$$s_{n+1} = p_{n+1}(k+1) + h\gamma_{k+1}\nabla^{k+1}f(x_{n+1},p_{n+1}(k+1))$$

$$y_{n+1}(k+1) = p_{n+1}(k+1)+h\gamma_k\nabla^{k+1}f(x_{n+1},p_{n+1}(k+1))$$

e portanto obtemos a estimativa para $\ell e_{n+1}(k+1)$

$$\ell e_{n+1}(k+1) = h(\gamma_{k+1} - \gamma_k)\nabla^{k+1}f(x_{n+1},p_{n+1}(k+1)). \tag{4.28}$$

Notemos no entanto que é preciso calcular $f(x_{n+1},p_{n+1}(k+1))$ e não estamos interessados nisso, pois além do cálculo de $p_{n+1}(k+1)$, seria necessária uma avaliação da função f. Mostraremos a seguir que é possível aproximar $\nabla^{k+1}f^p_{n+1}$ usando valores já obtidos quando calculamos $y_{n+1}(k)$ com extrapolação local. De fato, se esta foi feita, já calculamos y_{n+1} e portanto podemos calcular agora $f_{n+1}=f(x_{n+1},y_{n+1})$.

Por outro lado temos:

$$\nabla^{k+1}f(x_{n+1},p_{n+1}(k+1)) = f(x_{n+1},p_{n+1}(k+1))+ \sum_{j=1}^{k+1}(-1)^j\binom{k+1}{j}f_{n+1-j} = \tag{4.29}$$

$$= f(x_{n+1},p_{n+1}(k+1)) + \sum_{j=0}^{k+1}(-1)^j\binom{k+1}{j}f_{n+1-j} - f_{n+1} =$$

$$= f(x_{n+1},p_{n+1}(k+1)) - f(x_{n+1},y_{n+1}) + \nabla^{k+1}f_{n+1} =$$

$$= \nabla^{k+1}f_{n+1} + O(h^{k+2}).$$

Substituindo (4.29) em (4.28) temos:

$$\ell e_{n+1}(k+1) = h(\gamma_{k+1} - \gamma_k)\nabla^{k+1} f_{n+1} + 0(h^{k+3}) =$$

$$= h\gamma^*_{k+1}(\nabla^k f_{n+1} - \nabla^k f_n) + 0(h^{k+3}).$$

Observemos que para calcular a estimativa acima é necessário que tenhamos feito a última avaliação do modo PECE, o que não ocorre para o cálculo das estimativas de $\ell e_{n+1}(k)$, $\ell e_{n+1}(k-1)$ e $\ell e_{n+1}(k-2)$. Este fato terá grande influência nos algorítmos de seleção de ordem tratados no próximo capítulo.

Para finalizar este capítulo provamos que o erro local do corretor é assintoticamente equivalente ao erro de truncamento local, se y é suficientemente diferenciável.
De fato, pelo teorema 4.3 temos:

$$u_n(x) = y(x) + 0(\delta)$$

e podemos concluir derivando sucessivamente que:

$$u_n^{(i)}(x_n) = y^{(i)}(x_n) + 0(\delta). \tag{4.30}$$

Usando (1.7) vem:

$$\sigma_{n+1} = \frac{y^{(k+1)}(\xi)}{k!}\int_{x_n}^{x_{n+1}} w(t)\,dt = \frac{y^{(k+1)}(x_n)}{k!}\int_{x_n}^{x_{n+1}} w(t)\,dt + 0(H^{k+2}) \tag{4.31}$$

e como fizemos no capítulo 1 podemos provar que o erro local é da forma:

$$(4.32) \qquad \ell e_{n+1}(k) = \frac{u_n^{(k+1)}(\eta)}{k!} \int_{x_n}^{x_{n+1}} w(t)\,dt = \frac{u_n^{k+1}(x_n)}{k!} \int_{x_n}^{x_{n+1}} w(t)\,dt + 0(H^{k+2}).$$

Então, usando agora (4.30) e que $\delta = 0(H^{k+1})$, temos que os erros em questão são assintoticamente equivalentes.

CAPÍTULO V

ESTRATÉGIAS PARA SELEÇÃO DA ORDEM E DO PASSO. A FASE INICIAL DO PROCESSO. A INFLUÊNCIA DA REGIÃO DE ESTABILIDADE ABSOLUTA NESSAS ESTRATÉGIAS

No Capítulo II nós desenvolvemos a base teórica que nos assegura a convergência e a estabilidade dos métodos de Adams. No Capítulo III demos maneiras de calcular os coeficientes desses métodos usando-os no Capítulo IV para estimar o erro local. Neste capítulo usaremos os resultados dos capítulos anteriores para desenvolver algorítmos para a seleção do passo e da ordem. Como já dissemos oportunamente, só conseguimos justificar a estimativa para o erro local $\ell e_{n+1}(k+1)$ quando o passo e a ordem são constantes, por essa razão nossos algorítmos terão uma forte tendência em tomá-los constantes por grupos.

Já que no início temos somente o valor inicial y_o, devemos começar aplicando um método de ordem um, para em seguida aumentar a ordem até um limite que o programa achar satisfatório para o controle eficiente do erro local. As rotinas STEP e DE tratam de modo especial este caso e chamam-no de fase inicial. O problema, como veremos, reside justamente no fato de compatibilizar aumento da ordem com tamanho de passo constante.

Desenvolvemos inicialmente as estratégias de seleção de passo e ordem para as técnicas do passo variável e da in-

terpolação.

Depois faremos algumas considerações a respeito das implicações das estratégias de seleção de passo e ordem quando resolvemos um problema "STIFF"; para isso será necessário analisar as regiões de estabilidade absoluta dos métodos de Adams (com passo constante).

Queremos ainda lembrar que as estratégias aqui estudadas são usadas pelas rotinas STEP e DE, as quais implementam os métodos PC no modo PECE com métodos previsor e corretor de Adams de ordem k, com extrapolação local e com a técnica do passo variável. Da mesma forma estudamos as estratégias usadas pela rotina GEAR que implementa os métodos PC no modo $P(EC)^M E$, $M \leq 3$ com métodos de Adams na forma matricial de Nordsieck e com a técnica da interpolação. A rotina GEAR implementa ainda uma opção para o tratamento dos problemas "STIFF" cuja análise foge ao contexto deste trabalho, podendo ser encontrada em [9,capítulo 11] e em [3].

De 5.1 até 5.5 estaremos tratando da técnica dopasso variável, (rotinas STEP/DE) de 5.6 até 5.8 trataremos da técnica da interpolação (rotina GEAR) e em 5.9 voltamos novamente às rotinas STEP/DE.

5.1. *Argumentos em que se Baseiam as Seleções de Passo e Ordem nas Rotinas STEP/DE*

Deduziremos nesta seção estimativas para o erro local em x_{n+2}, nas quais se baseiam as estratégias de seleção de passo e ordem usadas por STEP/DE [16].

Tendo completado o passo de x_n para x_{n+1} nós podemos avaliar o erro local $\ell e_{n+1}(k)$ a partir da fórmula (4.6):

$$|\ell e_{n+1}(k)| \simeq |h_{n+1}(g_{k+1,1} - g_{k,1})\phi^p_{k+1}(n+1)|$$

que denotaremos ERR. Aceitamos o passo se ERR $\leq \varepsilon$ e rejeitamos como contrário.

Conforme salientamos oportunamente no Capítulo IV, uma idéia básica para a construção das estimativas para o erro local é que o erro de truncamento local para $u_n(x)$ domina o erro devido à incorreção dos valores anteriores (y_{n+1-j}) usados no cálculo do polinômio interpolador. Uma outra idéia que usamos no capítulo anterior é que o erro de truncamento local para $u_n(x)$ resulta do erro de interpolação, e a sua estimativa é essencialmente a estimativa deste erro. Assim, conforme o teorema 0.2, aproximar o erro de truncamento local para $u_n(x)$ é aproximar $\dfrac{u_n^{(k+1)}(\xi)}{k!}$, e esta aproximação é feita pela diferença dividida $f[x_n,\ldots,x_{n-k}]$. Agora, quão boa é esta aproximação depende principalmente de quanto $u_n^{(k+1)}(\xi)$ varia sobre o intervalo que contém os pontos $x_n,\ldots,x_{n-k}$. Sobre este intervalo podemos escrever que:

$$u_n^{(k+1)}(\xi) = u_n^{(k+1)}(x_n) + O(H)$$

e portanto numa análise limite a aproximação é razoável. Um dos

objetivos do ajustamento da ordem é exatamente fazer com que o programa use métodos com ordem k, de tal maneira que $u_n^{(k+1)}(\xi)$ seja quase constante.

Suponhamos que o passo de x_n para x_{n+1} teve sucesso. A próxima etapa será adiantar uma estimativa para o erro local no passo de x_{n+1} para x_{n+2}, se este passo for feito usando métodos de ordem k. Esta estimativa, que estará fortemente ligada às duas idéias acima, é dada por:

$$(5.1) \qquad |\ell e_{n+2}(k)| \simeq |h_{n+2}(g_{k+1,1} - g_{k,1})\phi_{k+1}^{p}(n+2)|$$

onde $\phi_{k+1}^{p}(n+2) = \psi_1(n+2)\ldots\psi_k(n+2)f^{p}[x_{n+2},\ldots,x_{n+2-k}]$.

$\phi_{k+1}^{p}(n+2)$, cujo cálculo depende de y_{n+2}, será substituído por $\phi_{k+1}^{p}(n+1)$, o que é justificado pela hipótese feita de considerar $u_n^{(k+1)}(x)$ quase constante.

Ficamos então com:

$$(5.2) \qquad |\ell e_{n+2}(k)| \simeq |h_{n+2}(g_{k+1,1} - g_{k,1})\phi_{k+1}^{p}(n+1)|$$

que entretanto depende de h_{n+2} ainda não conhecido. Descrevemos a seguir uma maneira de resolver este problema mais satisfatoriamente que por simples tentativa e erro.

O objetivo final será determinar h_{n+2} que satisfaça $|\ell e_{n+2}(k)| \leq \varepsilon$.

Tomando $h = h_{n+1}$ e supondo que os $k-2$ passos anteriores bem como h_{n+2}, foram tomados com tamanho de passo constante rh obtemos:

$$\phi^p_{k+1}(n+2) = (rh)r(2h)\ldots r(kh)f^p[x_{n+2},\ldots,x_{n+2-k}] \simeq \tag{5.3}$$

$$\simeq r^k\sigma_{k+1}(n+1)\phi^p_{k+1}(n+1)$$

onde

$$\sigma_i(n+1) = \begin{cases} 1 & \text{se } i = 1 \\ \sigma_{i-1}(n+1)\dfrac{(i-1)h}{\psi_{i-1}(n+1)}, & i \geq 2. \end{cases} \tag{5.4}$$

Sabemos que quando o passo é constante vale $g_{k+1,1} - g_{k,1} = \gamma^*_k$, assim a estimativa (5.1) se torna:

$$|\ell e_{n+2}(k)| \simeq |r^{k+1}h\gamma^*_k\sigma_{k+1}(n+1)\phi^p_{k+1}(n+1)|.$$

Esta estimativa é muito conveniente porque ela é facilmente calculável a partir de valores já obtidos. Se n_s passos anteriores foram tomados com tamanho constante h, a formação dos $\sigma_i(n+1)$ torna-se ainda mais simples. De fato, como $\psi_i(n+1)=ih$, $i \leq n_s$ temos que:

$$(5.5) \qquad \sigma_i(n+1) = \begin{cases} 1 & \text{se } i \le n_s + 1 \\ (i-1)a_{i-1}(n+1)\sigma_{i-1}(n+1) & i \ge n_s+2 . \end{cases}$$

Formemos agora a quantidade

$$(5.6) \qquad ERK = |h\gamma_k^* \sigma_{k+1}(n+1)\phi_{n+1}^p(k+1)|,$$

que seria uma estimativa para o erro local em x_{n+1} se os passos anteriores tivessem sido tomados com tamanho constante h. Como mostramos acima, uma previsão para o erro local em x_{n+2} pode ser dada por $r^{k+1}ERK$.

Se o passo não teve sucesso devemos repetí-lo. Neste caso a "nova" previsão para o erro local em x_{n+1} será dada da mesma forma por $r^{k+1}ERK$, supondo um novo h_{n+1} de tamanho rh.

Devido a diversos fatores que já mencionamos e que mencionaremos mais adiante, não permitiremos que r varie muito estaremos interessados somente em valores de r próximos de 1, resultando daí que a nossa suposição de que os passos são igualmente espaçados não é tão irreal quanto parece.

5.2. *Seleção da Ordem*

Formemos as quantidades

$$(5.7) \qquad \begin{aligned} ERKM1 &= |h\gamma_{k-1}^* \sigma_k(n+1)\phi_k^p(n+1)| \\ ERKM2 &= |h\gamma_{k-2}^* \sigma_{k-1}(n+1)\phi_{k-1}^p(n+1)| \end{aligned}$$

que seriam as estimativas para o erro local em x_{n+1}, se o passo de x_n para x_{n+1} tivesse sido tomado com ordens $k-1$ e $k-2$ respectivamente e os passos anteriores tomados com tamanho constante h. Exatamente como ERK as quantidades acima representam uma previsão para o erro local em x_{n+2}. Como dissemos anteriormente, para que possamos calcular uma estimativa para $\ell e_{n+2}(k+1)$ é necessário que o passo de x_n para x_{n+1} não tenha falhado. Justamente como antes, a previsão deste erro é:

$$\text{(5.8)} \qquad \text{ERKP1} = |h\gamma^*_{k+1}\phi_{k+2}(n+1)| \, .$$

Desde que pretendemos que os passos sejam constantes h, a solução numérica em x_{n+2} é:

$$\text{(5.9)} \qquad y_{n+2} = y_{n+1} + hf^p_{n+2} + \gamma^*_1 h\nabla f^p_{n+2} + \ldots$$

que é um análogo à série de Taylor para a solução local, $u_n(x)$. A nossa estimativa para o erro local consiste em truncar tal série num dado termo e tomar o seguinte como esta estimativa. Para que haja convergência da série acima é necessário que seus termos decresçam em módulo. Assim, a idéia básica da seleção de ordem é considerar termos desta série até que eles comecem a crescer, pois neste caso eles não teriam mais significado. Estes termos dependem do tamanho de passo h_{n+2} que não conhecemos ainda, mas como no algorítmo de seleção de passo permitiremos que h_{n+2} varie somente no intervalo $[0.5h, 2h]$, escolhemos

para h_{n+2} o valor de h. Queremos enfatizar que esta escolha será somente no algorítmo de seleção de ordem, pois quando descrevermos o algorítmo de seleção de passo veremos que $h_{n+2} = rh$.

A filosofia do algorítmo de seleção de ordem é mudar a ordem somente se o erro com a nova ordem for diminuído e se existir uma tendência nos termos da série.

Em todo passo será considerada em primeiro lugar a possibilidade de diminuição da ordem de k para k-1, o que, segundo a nossa filosofia, será caracterizado por:

(5.10)
a) ERKM1 ≤ ERK (diminuição do erro local)
b) ERKM2 ≤ ERK (tendência nos termos da série em aumentar).

Em a) e b) usamos ≤ porque preferimos trabalhar com a menor ordem possível, isto por diversos fatores, inclusive a estabilidade absoluta.

Notemos que, se k = 2, não podemos calcular ERKM2, e neste caso exigiremos uma forte indicação de diminuição no erro local, para que diminuamos a ordem, através de:

(5.11) ERKM1 ≤ 0.5ERK.

No caso de (5.10) não ser satisfeito e a estimativa do erro local com ordem k+1, segundo (5.8) for possível, diminuímos a ordem se:

(5.12)
a) ERKM1 ≤ ERK (diminuição do erro)
b) ERKM1 ≤ ERKP1 (tendência nos termos da série em aumentar).

Em resumo, diminuímos a ordem de k para $k-1$ se:

Para $k > 2$, $MAX(ERKM1, ERKM2) \leq ERK$

Para $k = 2$, $ERKM1 \leq 0.5ERK$

ou então se:

$ERKM1 \leq MIN(ERK, ERKP1)$ (quando o cálculo de ERKP1 é possível).

Queremos chamar a atenção para o fato que ERKM1, ERKM2 e ERK são obteníveis mesmo se o passo de x_n para x_{n+1} não teve sucesso, por conseguinte a diminuição da ordem será considerada mesmo após passos falhos.

Conforme a nossa filosofia, o aumento da ordem não será possível depois de passos falhos, pois neste caso não dispomos de ERKP1.

Aumentamos a ordem se:

(5.13)
a) $ERKP1 < ERK$ (diminuição do erro local)
b) $ERKP1 < ERKM1$ (indicação de que os termos da série ainda estão diminuindo, e portanto a ordem deve ser aumentada).

Notemos que, como consideramos sempre em primeiro lugar a diminuição da ordem, se estamos tentando aumentá-la é porque não foi possível diminuí-la, e foi obtida a estimativa ERKP1. Isto nos garante que valem:

(5.14) $ERKM1 > ERK$ ou $ERKM1 > ERKP1$.

Portanto, se um aumento de ordem foi decidido é porque (5.13a) e (5.14) foram ambas satisfeitas, o que evidentemente implica em (5.13b) que será então desnecessária.

Quando $k = 1$ o cálculo da estimativa ERKM1 não é possível e neste caso aumentamos a ordem somente se for verificada uma forte indicação de diminuição do erro local, detectada por:

$$ERKP1 < 0.5ERK.$$

As rotinas STEP e DE usam métodos com ordem no máximo 12, por esta razão, quando $k = 12$, não consideramos o aumento da ordem.

Em resumo, aumentamos a ordem de k para $k+1$ se:

$$\begin{aligned} &ERKP1 < ERK \quad \text{para} \quad 1 < k < 12 \\ (5.15) \quad & \\ &ERKP1 < 0.5ERK \quad \text{para} \quad k = 1. \end{aligned}$$

O algorítmo de seleção da ordem deixará de ser usado quando estivermos na fase inicial ou quando tivermos muitos passos falhos num dado ponto. Os detalhes destas situações serão estudados mais adiante.

Em (5.10) e (5.12), uma condição para diminuir a ordem é que $ERKM1 \leq ERK$, o que parece contraditório, pois dissemos anteriormente que os termos da série tendem a diminuir. Na verdade o que ocorre é que os erros de aritmética do computador se propagam, nas diferenças descendentes, em função de sua ordem. Um outro argumento é que as estimativas estão baseadas na suposição de que as derivadas da função $u_n(x)$ sejam quase

constantes localmente, e se uma destas derivadas não satisfaz esta hipótese, ou talvez nem existe, isto fará com que a diferença dividida correspondente seja grande, ou então seja um valor sem muito significado. Nestes casos o algorítmo acima ajustará a ordem pela ordem da derivada de bom comportamento.

Observamos que a suposição de que os passos anteriores foram tomados com tamanho constante não afeta muito o algorítmo acima, pois como já dissemos, teremos uma forte tendência em trabalhar com tamanho de passo constante.

Um outro detalhe é que mudanças freqüentes na ordem só serão possíveis quando o passo for constante, em contrapartida o aumento de ordem contribui para que continuemos com tamanho de passo constante.

5.3. *Seleção do Passo*

Suponhamos que foi selecionada uma ordem k e calculada a estimativa correspondente ERK com tamanho de passo constante h. Então, o erro previsto em x_{n+2} com passo rh e a mesma ordem k será r^{k+1}ERK, como já mostramos no início deste capítulo. Impomos a condição:

$$r^{k+1}\text{ERK} \leq \varepsilon \,. \tag{5.16}$$

Procuraremos o maior h_{n+2} que satisfaça (5.16), e para isso exigimos:

$$r^{k+1}ERK = \varepsilon, \quad \text{ou seja}$$

$$r = \left(\frac{\varepsilon}{ERK}\right)^{1/(k+1)}.$$

Assim conseguimos selecionar o tamanho de passo a ser usado para alcançar x_{n+2}. É claro que não podemos aceitar este r sem reservas, pois o uso de passos muito grandes ou muito pequenos não é permitido dentro da teoria desenvolvida. Imporemos assim certas limitações sobre r, fazendo com que o algorítmo respeite a condição (2.19), a qual é essencial para o teorema de estabilidade e convergência (teorema 2.5). Além disso, a estimativa do erro em x_{n+2} foi feita com base nos valores da função nos pontos anteriores.

Como proteção exigimos que:

$$r^{k+1}ERK = 0.5\varepsilon, \quad \text{ou seja,}$$

$$r = \left(\frac{0.5\varepsilon}{ERK}\right)^{1/(k+1)} \tag{5.17}$$

e limitamos a variação de r ao intervalo $[0.5,2]$.

É claro que a proteção exigida em (5.17) levará a tamanhos de passo menores que os que poderiam ser usados quando as estimativas para o erro local são boas, mas essa deficiência será compensada por poucos passos falhos quando aquelas estimativas não são boas.

Como já mencionamos oportunamente, quando trabalhamos com passo constante os cálculos tornam-se mais simples, então achamos que, se podemos aumentar o tamanho do passo, devemos fa

zê-lo por um fator bem maior que um. Na prática tentamos, numa primeira atitude, dobrar o passo, o que será possível se:

(5.18) $2^{k+1}ERK \leq 0.5\varepsilon$, isto é, $2^{k+2}ERK \leq \varepsilon$.

Como consideramos o ajustamento do tamanho do passo em todo passo, um rápido crescimento deste é possível quando isso parecer razoável, segundo (5.18) ; entretanto esse crescimento pode levar a passos falhos quando a solução tiver mudanças bruscas.

O teste (5.18) é muito conveniente quando por exemplo $ERK \simeq 0$, pois neste caso, a estimativa de r por (5.17) daria valores muito grandes quando nada de especial ocorreu; $ERK \simeq 0$ significa apenas que alguma derivada de $u_n(x)$ é próxima de zero.

Se não for possível dobrar o tamanho do passo, então tentamos mantê-lo (aqui está o fato, várias vezes já mencionado, que as rotinas STEP e DE têm a forte tendência a tomar o tamanho de passo constante por grupo). Isto corresponde a tomar $r = 1$ exigindo que:

$$ERK \leq 0.5\varepsilon.$$

Se este teste também não for satisfeito, então devemos reduzir o tamanho do passo e neste caso usamos a estimativa para r dada por (5.17). Ainda aqui imporemos algumas restrições sobre r. Exigiremos que $r \leq 0.9$, isto para evitar que o passo seja diminuído por um fator insignificante, o que impli

caria em pequena variação das estimativas resultando entretanto um enorme trabalho computacional no passo atual e nos futuros.

Na prática calculamos o novo passo por:

$$h_{n+2} = h_{n+1}\max(0.5,\min(0.9,r)). \tag{5.19}$$

Notemos que a estratégia de aumento do tamanho do passo é muito diferente da usada na diminuição, visto que nesta nós praticamente aceitamos o valor de r dado por (5.17) e naquela nós não levamos este valor em consideração.

A intenção quando usamos (5.19) é chegar "suavemente" ao "melhor" tamanho de passo a partir do passo anterior. Quando somente dobramos ou dividimos ao meio o tamanho do passo isto não ocorre.

No caso de, após a mudança de passo, ocorrer um passo falho, a estimativa para o erro local torna-se menos segura, desde que ela está baseada em valores que foram inaceitáveis $(h_{n+1},\phi^{p}_{k+1}(n+1))$. Neste caso a nossa atitude será dividir o passo ao meio e tentar novamente. Em vista do ajustamento da ordem e do tamanho do passo feito nos passos anteriores, um passo falho é devido a uma mudança brusca na solução. Repetidas falhas num mesmo passo indicam que temos alguma derivada de ordem baixa descontínua, com grande salto. Se assim for, nós podemos considerar o nosso problema como a reunião de dois novos problemas:

$$\dot{y} = f(x,y) \qquad\qquad \dot{y} = f(x,y)$$

$$y(x_o) = y_o = A \qquad\qquad y(x_n) = y_n$$

onde x_n é o último ponto em que conseguimos obter a aproximação para a solução do primeiro problema. Então para o segundo problema podemos também aplicar o nosso método. Na prática tal situação é detectada se há três passos falhos consecutivos; neste caso colocamos $k = 1$ e voltamos à fase inicial.

Para valores de ε muito pequenos, há frequentemente necessidade de recomeçar um grande número de vezes até que seja detectado o ponto x_n de descontinuidade da derivada de ordem baixa. Só então será possível passar adiante e recomeçar a crescer o tamanho do passo e a ordem. Para facilitar este comportamento é que recomeçaremos o cálculo com $h_1 = \min(\frac{h}{2}, rh)$.

5.4. *A Fase Inicial*

No início dos cálculos das aproximações conhecemos apenas a solução y_o em x_o e portanto precisamos aplicar um método de ordem $k = 1$, para depois ir crescendo a ordem conforme tenhamos valores anteriores suficientes. O método de ordem 1 aplicado inicialmente precisa de um tamanho de passo inicial o qual estimamos supondo que calculamos uma aproximação y_1 com um método de ordem zero, isto é, $y_1 = y_o = A$. Observemos que o erro de um método de ordem um é h vezes o erro do de ordem zero, e portanto por Taylor temos:

$$y(x_1) = y(a) + hy'(a) + 0(h^2).$$

Mas, $u_o(x) = y(x)$ e assim temos:

$$u_o(x_1) - A = hf(a,A) + 0(h^2) \quad \text{ou}$$

$$u_o(x_1) - y_1 = hf(a,A) + 0(h^2) .$$

Com a hipótese feita:

$$\ell e_1 \simeq h^2 |f(a,A)|$$

que deve satisfazer $\ell e_1 \simeq 0.5\varepsilon$. Obtemos assim a estimativa para h:

$$h \simeq \left|\frac{0.5\varepsilon}{f(a,A)}\right|^{1/2} . \tag{5.20}$$

É claro que se $f(a,A) \simeq 0$, esta estimativa não tem sentido, e neste caso devemos escolher um outro valor para h, ou seja $h = h_{inicial}$, que em geral é escolhido em função da unidade de de arredondamento do computador. Assim, escolhemos h como sendo:

$$h = \min\{h_{inicial}, (1/4)\left|\frac{0.5\varepsilon}{f(a,A)}\right|^{1/2}\}$$

onde $1/4$ é um fator de segurança.

Esta estimativa para o tamanho de passo inicial é irrelevante porque não é vital para o bom funcionamento do programa: se h for muito grande o próprio algorítmo de seleção de passo se encarregará de diminuí-lo, se for muito pequeno, as estratégias que adotaremos para a fase inicial encarregar-se-ão de aumentá-lo rapidamente nos próximos passos. A atitude da rotina STEP durante a fase inicial é:

Dobrar o tamanho do passo e aumentar a ordem em todo passo, até que ocorra um passo falho, ou que o algorítmo de seleção da ordem permita que esta seja diminuída, ou que encontremos a ordem máxima permitida 12.

Sob qualquer das três situações acima a fase inicial é terminada.

A principal motivação para a atitude acima é aumentar a ordem o mais rápido possível. É claro que isto trará problemas quando aumentarmos a ordem além do necessário, mas este fato será contornado pois tentamos a diminuição da ordem em todo passo.

O crescimento da ordem é eficiente quando a tolerância ε for muito pequena. Notemos que pela estimativa (5.20), h diminue com ε. Em geral esta estimativa é menor do que realmente poderia ser, o que nos leva a crer que a fase inicial será terminada mais provavelmente por diminuição da ordem do que por passos falhos.

5.5. *Conseqüências das Estratégias de Seleção de Passo e Ordem*

Justificamos novamente, através da estratégia de seleção de passo, não ser tão irreal a atitude de considerar os passos constantes para a construção das estimativas para o erro local em x_{n+2}.

Por (1.7) o erro local $\ell e_{n+1}(k)$ é dado por:

$$(5.21) \qquad \frac{u_n^{(k+1)}(\xi)}{k!} \int_{x_n}^{x_{n+1}} (t-x_{n+1})(t-x_n)\ldots(t-x_{n+2-k})\,dt.$$

Agora, se $h = h_{n+1}$, devido ao algorítmo de seleção de passo temos que:

$$h_n \geq (1/2)h \qquad\qquad h_n \leq 2h$$

$$h_{n-1} \geq (1/2)h_n \geq \frac{1}{2^2}h \qquad\qquad h_{n-1} \leq 2h_n \leq 2^2 h$$

$$\vdots \qquad\qquad\qquad\qquad \vdots$$

$$h_{n+2-k} \geq \frac{h}{2^{k-1}} \qquad\qquad h_{n+2-k} \leq 2^{k-1}h.$$

Usando estas condições podemos provar, para $k \leq 12$, que:

$$2^{k+10}h^{k+1}\gamma_k^* \leq \frac{1}{k!}\int_{x_n}^{x_{n+1}} (t-x_{n+1})(t-x_n)\ldots(t-x_{n+2-k})\,dt \leq$$

$$\leq \frac{h^{k+1}\gamma_k^*}{2^{k+10}}.$$

Notemos que $h^{k+1}\gamma_k^* u_n^{(k+1)}(\xi)$ é a estimativa ERR que obtemos para o erro local $\ell e_{n+1}(k)$ supondo passo constante, então provamos acima que:

$$\frac{1}{2^{k+10}}\text{ERR} \leq |\ell e_{n+1}(k)| \leq 2^{k+10}\text{ERR}.$$

Uma outra conseqüência dos algorítmos anteriores, é que o controle sobre $\ell e_{n+1}(k)$ por passos implica no controle de LE_{n+1} por unidade de passos no sentido generalizado (def. 4.3). De fato, provamos no final do capítulo IV que:

$$\ell e_{n+1}(k) = \frac{y^{(k+1)}(x_n)}{k!} \int_{x_n}^{x_{n+1}} (t-x_{n+1})(t-x_n)\dots(t-x_{n+2-k})\,dt.$$

Quando trabalhamos com tamanho de passo constante $h = h_{n+1}$ temos também:

$$|\ell e_{n+1}(k)| \simeq |y^{(k+1)}(x_n)\gamma_k^* h_{n+1}^{k+1}.$$

Se o passo teve sucesso devemos ter:

$$|\ell e_{n+1}(k)| \simeq 0.5\varepsilon, \quad \text{ou seja}$$

$$|h_{n+1}^{k+1}\gamma_k^* y^{(k+1)}(x_n)| < \varepsilon.$$

Por um procedimento semelhante ao que fizemos para encontrar uma estimativa para $\ell e_{n+1}(k+1)$ no capítulo IV, podemos mostrar que:

$$|LE_{n+1}| \simeq |h_{n+1}^{k+2}\lambda_k(x_n)| = h_{n+1}|\lambda_k(x_n)|h_{n+1}^{k+1} \leq$$

$$\leq h_{n+1}\frac{\varepsilon|\lambda_k(x_n)|}{|\gamma_k^* y^{(k+1)}(x_n)|} = \sigma\varepsilon h_{n+1}.$$

5.6. *Argumentos em que se Baseiam as Seleções de Passo e Ordem na Rotina GEAR*

Como já dissemos anteriormente, esta rotina implementa os métodos PC (Previsor de ordem k e Corretor de ordem $k+1$), com métodos de Adams de passo constante na forma matricial de Nordsieck com a técnica do passo variável para mudança de passo. Lambert, em [13, p. 90] mostra que, se a ordem do método previsor mais o número de correções M é maior que a ordem do corretor, então o erro de truncamento local do método PC no modo $P(EC)^M E$ é da forma:

$$\gamma_k^* h^{k+1} y^{(k+1)}(x_n) \tag{5.22}$$

onde k é a ordem do corretor.

Notemos agora que, conforme (1.24), o vetor Y_n obtido usando um corretor de ordem k é dado por:

$$Y_n = [y_n, hy_n', \frac{h^2}{2} y_n'', \ldots, \frac{h^k y_n^{(k)}}{k!}] .$$

Temos ainda que:

$$\frac{h^{k+1} y^{(k+1)}(x_n)}{k!} \simeq \frac{h^k}{k!} \nabla y_n^{(k)} .$$

Se indicamos $(Y_n)_{k+1} = \dfrac{h^k y_n^{(k)}}{k!}$, então uma estima

tiva para o erro de truncamento local pode ser calculada por:

(5.23) $\quad \text{ERRO} = \gamma_k^* k! \nabla(Y_n)_{k+1}$

e então só aceitamos um passo se:

$$||\text{ERRO}|| \leq \varepsilon \ (\varepsilon \text{ pré-fixado}) .$$

5.7. *Seleção do Passo*

Novamente aqui suporemos que $y^{(k+1)}(x)$ é quase constante. Com esta suposição, podemos selecionar o tamanho do passo a ser usado para o próximo ponto, ou então para repetir este passo caso ele tenha sido rejeitado pelo teste (5.24).

Seja $h_{n+1} = rh$, então (5.23) fica:

$$\text{ERRO} = \gamma_k^* k! r^{k+1} \nabla(Y_n)_{k+1} .$$

e com as hipóteses feitas sobre $y^{(k+1)}(x)$, obtemos:

$$\text{ERRO} = \gamma_k^* k! r^{k+1} \nabla(Y_n)_{k+1}$$

Se exigimos agora que $||\text{ERRO}|| = \varepsilon$, podemos estimar r por:

$$r = \left[\frac{\varepsilon}{\gamma_k^* k!} \frac{1}{||\nabla(Y_n)_{k+1}||}\right]^{1/(k+1)} .$$

Para evitar passos falhos tomamos:

$$r = \frac{1}{1.2}\left[\frac{\varepsilon}{\gamma_k^* k!} \frac{1}{||\nabla(Y_n)_{k+1}||}\right]^{1/(k+1)} .$$

Podemos da mesma forma encontrar estimativas para os tamanhos de passo que poderiam ser usados para ordens $k+1$ e $k-1$.

Para ordem $k+1$, o erro de truncamento local é dado por:

$$\gamma_{k+1}^* h^{k+2} y^{(k+2)}(x_n).$$

Podemos estimar $h^{k+2} y^{(k+2)}(x_n) \approx k!\nabla^2(Y_n)_{k+1}$ e então obtemos:

$$r = \frac{1}{1.4}\left[\frac{\varepsilon}{\gamma_{k+1}^* k!} \frac{1}{||\nabla^2(Y_n)_{k+1}||}\right]^{1/(k+2)} \tag{5.26}$$

onde $\frac{1}{1.4}$ é um fator de segurança.

Da mesma forma, uma estimativa de r para ordem $k-1$ pode ser dada por:

$$r = \frac{1}{1.3}\left[\frac{\varepsilon}{\gamma_{k-1}^* k!} \frac{1}{||(Y_n)_{k+1}||}\right]^{1/(k)} . \tag{5.27}$$

Os fatores $\frac{1}{1.2}$, $\frac{1}{1.3}$ e $\frac{1}{1.4}$ são escolhidos nessa ordem a fim de considerar em primeiro lugar a conservação da ordem, desde que a mudança desta implica em um tempo adicional de processamento, e em segundo lugar a diminuição da ordem desde que este fato implica em menor trabalho por passo.

Para diminuir o tempo de processamento a rotina GEAR não calcula as estimativas acima em todo passo. Estas estimativas serão calculadas somente:

1) Se o passo foi rejeitado, e neste caso não calculamos a estimativa (5.26), pois isto evita o aumento da ordem quando o problema é "stiff".

2) k+1 passos depois da última mudança do tamanho do passo ou da ordem. Esta exigência permite que tenhamos as hipóteses dos teoremas de convergência e estabilidade (teoremas 2.6 e 2.8), satisfeitas.

3) Dez passos depois que r foi estimado pela última vez, desde que o tamanho do passo não tenha sido aumentado. Isto evita o uso de passos muito pequenos.

Para que a condição (2.19) (limitação do quociente de passos sucessivos) seja satisfeita, impomos a cada passo que:

$$h_{min} \leq h_{n+1} \leq h_{max}.$$

Depois de um passo que teve sucesso não consideramos a possibilidade de diminuir o tamanho do passo; além disso exigi-

mos que $r \geq 1.1$ para evitar pequenas modificações no tamanho do passo.

Notemos que em (5.25), (5.26) e (5.27), se algum dos valores no denominador são próximos de zero, obtemos valores exagerados de r, o que não seria razoável. Dessa forma limitamos r como:

$r \leq 10$ se o passo teve sucesso

$r \leq 2$ caso contrário.

Como já dissemos, a rotina GEAR implementa os métodos PC no modo $P(EC)^M E$, com $M \leq 3$. Assim calculamos uma aproximação com o previsor P e corrigimos esta aproximação no máximo três vezes. Como LAMBERT prova em [13, p. 86], uma condição suficiente para que o processo corretor seja convergente é que $|h\gamma_k^* \frac{\partial f}{\partial y}| < 1$. A rotina testa a cada iteração a convergência. Se após três iterações não houve convergência então h é reduzido por um fator $r = 0.25$.

Se ocorrerem três passos falhos consecutivos a rotina GEAR diminui o passo h por um fator $r = 0.1$.

Paramos de diminuir o tamanho do passo quando atingimos h_{min}.

5.8. *Seleção da Ordem e Fase Inicial*

Selecionamos a ordem correspondente ao maior valor de r entre os obtidos por (5.25),(5.26) e (5.27).

A rotina GEAR não faz distinção para o tratamento da fase inicial como no caso da rotina STEP. Iniciamos com o vetor $Y_o = [y_o, hy'_o]$ e daí em diante aplicamos o mesmo algorítmo para os demais passos.

A principal diferença entre as estratégias de seleção de passo e ordem existente entre as rotinas STEP e GEAR, é que a primeira trata o valor de r encontrado pelo algorítmo de seleção de passo com muitas reservas, chegando mesmo a ignorá-lo em certos casos, enquanto a segunda assume ser este valor "ótimo" para prever o próximo passo, chegando mesmo a escolher a ordem em função desse valor.

Como veremos no próximo capítulo, as reservas quanto à estimativa de r são convenientes, pois a rotina STEP, se comporta melhor frente a problemas que exigem grande variação no tamanho dos passos.

Este fato é observado principalmente para valores de ε muito pequenos.

Enquanto na rotina STEP a filosofia parece ser, prevenir contra os possíveis passos falhos, na rotina GEAR a filosofia é justamente deixá-los ocorrer para depois diminuir o tamanho dos passos.

5.9. *Influência da Região de Estabilidade Absoluta nas Estratégias de Mudança do Passo e da Ordem nas Rotinas STEP e DE*

No capítulo II mostramos a estabilidade dos métodos de Adams (PC) para tamanho de passo pequeno. Estávamos

então preocupados em estabelecer limitações sobre o tamanho do passo a fim de que as aproximações produzidas pelos métodos fossem razoáveis. Neste parágrafo estaremos preocupados com o mesmo problema, somente que agora consideramos uma única aplicação do método ao invés de analisar o que ocorre no limite de uma seqüência de aplicações.

A fim de compreender o significado da estabilidade e comparar vários métodos nós estudaremos um problema simples como modelo. As conclusões são rigorosamente válidas somente para este problema, mas esta análise pode nos sugerir o comportamento de tais métodos quando aplicados a um problema geral.

O problema a ser estudado é:

$$y' = Ay + g(x). \tag{5.28}$$

A estabilidade de um método é o modo como o erro se propaga quando ele é aplicado. Se em algum ponto x_r mudamos o valor da solução $y(x)$, de (5.28), de $y(x_r)$ para $y(x_r) + \delta(x_r)$, então a solução é modificada para:

$$u(x) = y(x) + \delta(x)$$

onde $\delta(x)$ é a solução de:

$$\delta' = A\delta, \quad \delta(x_r) \text{ dado.}$$

Notemos que, se em algum outro ponto $x_t > x_r$, modi-

ficamos o valor $u(x_t)$ para $u(x_t) + \eta(x_t)$, a solução novamente será modificada para $v(x) = y(x) + \delta(x) + \eta(x)$. Assim concluímos que o efeito dos erros na solução final é aditivo, e portanto podemos considerar o efeito de um único erro.

Consideremos agora o método PC com métodos de ordem k e passo constante h.

$$
(5.29)\qquad
\begin{aligned}
p_{n+1} &= y_n + h \sum_{j=1}^{k} \beta_{j,k} f_{n+1-j} \\
y_{n+1} &= y_n + h \sum_{j=1}^{k} \beta^*_{j,k+1} f_{n+1-j} + h\beta^*_{o,k+1} f^p_{n+1} .
\end{aligned}
$$

Suponhamos que aplicamos os métodos acima para resolver (5.28) e que os valores iniciais estejam afetados de erros $\delta_n, \delta_{n-1}, \ldots, \delta_{n+1-k}$, ou seja, ao invés de y_{n+1-j} temos os valores $y_{n+1-j} + \delta_{n+1-j}$. Queremos saber qual o efeito desses erros no cálculo de y_{n+1}. Então,

$$
p_{n+1}+\delta^p_{n+1} = (y_n + \delta_n) + h \sum_{j=1}^{k} \beta_{j,k} [A(y_{n+1-j}+\delta_{n+1-j}) + g_{n+1-j}]
$$

$$
\begin{aligned}
y_{n+1}+\delta_{n+1} = (y_n + \delta_n) + h \sum_{j=1}^{k} \beta^*_{j,k+1} [A(y_{n+1-j}+\delta_{n+1-j}) + g_{n+1-j}] + \\
+ h\beta^*_{o,k+1} [A(p_{n+1}+\delta^p_{n+1}) + g_{n+1}] .
\end{aligned}
$$

Usando agora (5.29) temos:

$$\delta^p_{n+1} = \delta_n + h \sum_{j=1}^{k} \beta_{j,k} A\delta_{n+1-j}$$

(5.30)

$$\delta_{n+1} = \delta_n + h \sum_{j=1}^{k} \beta^*_{j,k+1} A\delta_{n+1-j} + h\beta^*_{o,k+1} A\delta^p_{n+1}$$

que é a aplicação dos métodos (5.29) para resolver o problema $\delta' = A\delta$ com valores iniciais $\delta_n, \ldots, \delta_{n+1-j}$. Se em (5.30) substituímos o valor de δ^p_{n+1} da primeira equação na segunda, obtemos a seguinte equação de diferenças de ordem k:

(5.31) $$\delta_{n+1} = \delta_n + hA \sum_{j=1}^{k} \beta^*_{j,k+1} \delta_{n+1-j} + hA\beta^*_{o,k+1} [\delta_n + hA \sum_{j=1}^{k} \beta_{j,k} \delta_{n+1-j}]$$

cuja solução geral é dada por:

(5.32) $$\delta_n = d_1 r_1^n + d_2 r_2^n + \ldots + d_k r_k^n$$

onde os d_i são quaisquer números e $r_1, r_2, \ldots, r_k$ são as raízes, supostas distintas, do polinômio característico da equação (5.31), que é dado por:

(5.33) $$r^k = r^{k-1} + \lambda \sum_{j=1}^{k} \beta^*_{j,k+1} r^{k-j} + \lambda\beta^*_{o,k+1} [r^{k-1} + \lambda \sum_{j=1}^{k} \beta_{k,j} r^{k-j}]$$

com $\lambda = hA$.

Se impomos as condições iniciais em (5.32) encontramos a única solução da equação de diferenças.

Observando (5.32) concluímos que a condição para que δ_{n+1} seja limitado é que $|r_i| \leq 1$, $i = 1,2,\ldots,k$.

Notemos também que as raízes r_i dependem do parâmetro $\lambda = hA$. Definimos então:

<u>Definição 5.1</u> A região de estabilidade absoluta do par PC é definida como:

$$REA = \{\lambda \in C,\ Re(\lambda) < 0 \mid |r_i| \leq 1,\ \ i = 1,2,\ldots,k\}.$$

Shampine em [16, p. 135-140] mostra em gráficos, as regiões de estabilidade absoluta para ordens $k = 1,2,\ldots,12$ dos métodos:

Adams-Bashforth (ou explícito)

PECE com previsor e corretor de ordem k

PECE com previsor de ordem k e corretor de ordem $k+1$.

Daremos no que segue uma regra prática para que as rotinas STEP/DE possam detectar quando um problema é "stiff", justificando tal regra a partir das regiões de estabilidade absoluta e dos algorítmos de seleção de passo e ordem.

Normalmente, a exigência de que as estimativas para o erro local sejam, em cada passo, menores que um certo ε, man

têm o tamanho do passo dentro de um campo onde não ocorre instabilidade. Entretanto, existem problemas em que esta exigência impõe um limite superior muito pequeno sobre o tamanho do passo num dado intervalo. Estes problemas são chamados "stiff". Sem muito rigor podemos dizer que um problema é "stiff" num dado intervalo, se existem componentes da solução tais que, pequenas variações na variável independente x, neste intervalo, provocam grandes variações nessas componentes. Os métodos de Adams não são apropriados para resolver tais problemas, visto que a limitação no tamanho do passo faz com que o tempo de processamento seja grande. O que tentaremos é decidir quando um problema é "stiff", para então aplicar métodos apropriados. A rotina GEAR inclue, em sua programação, uma família de tais métodos.

Conforme mostramos em (4.5), a estimativa para o erro local dos métodos de ordem k, com tamanho de passo constante h é dada por $\gamma_k^* h \nabla^k f_n^p$. Primeiramente analisamos a estimativa $\gamma_k^* h \nabla^k f_n$ que é assintoticamente equivalente à anterior, mas é mais simples de ser analisada.

Se estamos resolvendo um problema como (5.28) temos:

$$\gamma_k^* h \nabla^k f_n = \gamma_k^* h \nabla^k [A(y_n + \delta_n) + g_n] =$$

$$= \gamma_k^* h \nabla^k [Ay_n + g_n] + \gamma_k^* h A \nabla^K \delta_n .$$

Olhemos agora o efeito de δ_n sobre o erro acima.

Por (5.32) podemos escrever:

$$\gamma_k^* hA\nabla^k \delta_n = \gamma_k^* hA \sum_{j=1}^{k} d_j \nabla^k r_j^n$$

Escrevendo r_j na forma polar temos:

$$\nabla r_j^n = \rho_j^n \exp(in\theta_j) - \rho_j^{n-1}\exp((n-1)i\theta_j) =$$

$$= r_j^n\left(1 - \frac{\exp(-i\theta_j)}{\rho_j}\right)$$

Desta forma podemos concluir que

$$\nabla^k r_j^n = r_j^n\left(1 - \frac{\exp(-i\theta_j)}{\rho_j}\right)^k$$

ou

$$\gamma_k^* hA\nabla^k \delta_n = \sum_{j=1}^{k} c_j r_j^n \tag{5.34}$$

onde $c_j = \gamma_k^* hAd_j\left(1 - \frac{\exp(-i\theta_j)}{\rho_j}\right)^k$ é

constante em relação a n quando h é fixado.

Voltemos agora à estimativa $\gamma_k^* h\nabla^k f_n^p$.

Para o problema (5.28) temos:

(5.35) $$\gamma_k^* h \nabla^k f_n^p = \gamma_k^* h \nabla^k [A(p_n + \delta_n^p) + g_n] =$$

$$= \gamma_k^* h \nabla^k [Ap_n + g_n] + \gamma_k^* hA \nabla^k \delta_n^p .$$

Pelas propriedades de diferenças descendentes sabemos que:

(5.36) $$\nabla^k \delta_n^p = \delta_n^p + \sum_{m=1}^{k} (-1)^m \delta_{n-m} \binom{k}{m} =$$

$$= \delta_n^p - \delta_n + \nabla^k \delta_n .$$

Por (5.32), $\delta_n = \sum_{j=1}^{k} d_j r_j^n$ e

por (5.30), $\delta_n^p = \delta_{n-1} + \lambda \sum_{i=1}^{k} \beta_{i,k} \delta_{n-i}$

Podemos então escrever:

$$\delta_n^p = \sum_{j=1}^{k} d_j r_j^{n-1} + \lambda \sum_{i=1}^{k} \beta_{i,k} d_j r_j^{n-i} =$$

$$= \sum_{j=1}^{k} \frac{r_j^n}{r_j} d_j + \lambda \sum_{i=1}^{k} \beta_{i,k} \sum_{j=1}^{k} \frac{r_j^n}{r_j^i} d_j =$$

$$= \sum_{j=1}^{k} \frac{r_j^n}{r_j} d_j + \lambda \sum_{j=1}^{k} r_j^n [\lambda d_j \sum_{i=1}^{k} \frac{\beta_{i,k}}{r_j^i}] =$$

$$= \sum_{j=1}^{k} e_j r_j^n .$$

Com essas expressões obtemos:

$$A\gamma_k^* h \nabla^k \delta_n^p = A\gamma_k^* h(\delta_n^p - \delta_n + \nabla^k \delta_n) =$$

$$= A\gamma_k^* h[\sum_{j=1}^{k} e_j r_j^n - \sum_{j=1}^{k} d_j r_j^n] + \sum_{j=1}^{k} c_j r_j^n =$$

$$= \sum_{j=1}^{k} c_j^* r_j^n$$

com e_j, c_j e c_j^* constantes.

Concluímos então:

$$\gamma_k^* hA \nabla^k \delta_n^p = \sum_{j=1}^{k} c_j^* r_j^n. \tag{5.37}$$

Observando (5.35) vemos que o primeiro membro (5.37) é uma parcela da estimativa para o erro local, e portanto se esta parcela é grande, a estimativa para o erro também será, havendo necessidade de uma redução no tamanho do passo.

Se $\lambda = hA$ está fora da região de estabilidade absoluta, pela definição 5.1 teremos que $|r_j| > 1$ para algum j, e portanto a estimativa para o erro deve crescer à medida que n cresce, devido à expressão (5.37). Este crescimento será detectado e h será reduzido até que λ esteja dentro da região de estabilidade absoluta novamente.

A seleção do tamanho do passo é feita de tal maneira que a estimativa para o erro local nesse passo é próxima da tolerância ε (pelo fato de tomar o maior tamanho de pas

so possível); por causa deste fato, somente uns poucos passos são necessários para que a rotina STEP detecte instabilidade.

Quando o problema é "stiff" a instabilidade limita a variação do tamanho do passo, e este tenderá a oscilar em torno do maior valor para o qual o método é estável. Este fato ocorre porque o erro é pequeno quando λ está dentro da região de estabilidade absoluta. Se o crescimento do tamanho do passo tira λ dessa região, teremos um crescimento do erro, o que leva a uma diminuição do tamanho do passo, e assim por diante.

Conforme o algorítmo de seleção de ordem, esta não poderá ser aumentada depois de um passo falho, mas poderá ser diminuída. Assim, quando ocorrem problemas no cálculo da aproximação, a rotina STEP possivelmente diminui a ordem, aplicando métodos com maior região de estabilidade absoluta. Portanto quando o problema é "stiff", as rotinas devem aplicar métodos de ordem baixa. Com base nesse fato, as rotinas STEP/DE dizem que um problema é "stiff" se uma seqüência de 50 passos consecutivos são tomados com ordem menor ou igual a 4.

Acontece que para um problema de fácil solução, uma seqüência de passos com ordem baixa pode ocorrer, então STEP/DE dizem que o problema é "stiff" se a regra acima é satisfeita e, além disso, o problema está exigindo um número excessivo de passos, ou seja, já foram considerados cerca de 500 passos.

Caso o problema seja "stiff", mas pode ser resolvido com um número razoável de passos, o teste acima não detec-

tará esta sua particularidade. Há também exemplos em que o problema não é "stiff", mas as rotinas, o concluem como tal.

No próximo capítulo daremos exemplos que ilustram o comportamento das rotinas STEP/DE frente a problemas "stiff".

CAPÍTULO VI

RESULTADOS NUMÉRICOS

Neste capítulo apresentaremos resultados numéricos que ilustram o comportamento das rotinas STEP/DE e GEAR. Ressaltaremos principalmente as diferenças entre estas rotinas com relação a:

- Tempo de processamento quando varia o número de equações do sistema.
- Ilustração dos teoremas de estabilidade.
- Solução de problemas com singularidade na primeira derivada.
- Solução de problemas "STIFF".

Os problemas teste que utilizamos foram apresentados por Hull et all em [10], Krogh em [12] e Shampine em [16] e [18]. Algumas observações aqui apresentadas são devidas a Enright [1]. Os resultados numéricos foram obtidos num computador PDP11/45 com o sistema operacional RSX11M, versão 3.1, cuja unidade de arredondamento em precisão simples é de $0(10^{-7})$.

6.1. *Tempo de Processamento Quando Varia o Número de Equações no Sistema*

Grande parte do custo computacional da rotina STEP é

devida ao cálculo dos $g_{i,1}$, que independem do número de equações do sistema, por conseguinte em STEP o custo computacional não depende tão fortemente deste número. Já na rotina GEAR o custo computacional depende muito do número de equações do sistema, pois estão envolvidas operações com matrizes cujas dimensões dependem deste número. Esta diferença pode ser constatada na prática quando resolvemos sistemas de NQ equações para NQ = 1,5,10,15,20. Para que possamos medir o tempo de processamento (TP) como função de NQ consideramos os sistemas com todas as equações iguais. No exemplo ilustrativo elas são:

$$y' = \frac{y(4x^3 - y)}{x^4-1}, \qquad y(2) = 15$$

cuja solução é $y(x) = 1+x+x^2+x^3$.

A tabela 6.1 mostra o TP, medido em segundos, para resolver tais sistemas no intervalo [2,10] com tolerância 10^{-5} para o erro local absoluto.

NQ	1	5	10	15	20
STEP/DE	0.13	0.23	0.37	0.49	0.62
GEAR	0.097	0.22	0.37	0.5	0.75

Tabela 6.1

Observamos que o sistema utilizado não dispõe de recursos refinados para medir o TP; os resultados apresentados na tabela 6.1 são a média aritmética dos TP's de várias execuções.

6.2. *Ilustração dos Teoremas de Estabilidade*

Para ilustrar a estabilidade resolvemos o problema:

$$y_1'' = 2y_2' + y_1 - \frac{\mu^*(y_1+\mu)}{r_1^3} - \frac{\mu(y_1-\mu^*)}{r_2^3}$$

$$y_2'' = 2y_1' + y_2 - \frac{\mu^* y_2}{r_1^3} - \frac{\mu y_2}{r_2^3}$$

$$y_1(0) = 1.2,\ y_1'(0) = 0,$$

$$y_2(0) = 0,\ y_2'(0) = -1.049357509,$$

onde $\quad r_1 = ((y_1+\mu)^2 + y_2^2)^{1/2}\ ,\quad r_2 = ((y_1-\mu^*)^2+y_2^2)^{1/2}$

$$\mu = 1/82.45 \quad \text{e} \quad \mu^* = 1-\mu.$$

Computadores de alta precisão mostraram que a solução, que não é conhecida exatamente, é periódica com período $T = 6.19216933131963$, (ver [16, p. 247] ou [12, p. 550]). O problema foi resolvido em $[0,T]$, e a solução deve coincidir nos extremos. O erro global em T, obtido com tolerância 10^{-5} para o erro local absoluto, é dado pela tabela 6.2

	$\|\|$ERRO GLOBAL$\|\|_\infty$
STEP/DE	1.867E-04
GEAR	1.227E-02

Como vemos a rotina GEAR produz resultados menos precisos. Achamos que isto ocorre porque a solução deste problema requer muitas mudanças no tamanho do passo, fazendo com que as hipóteses do teorema 2.8 deixem de ser satisfeitas, não sendo garantida a estabilidade. Relacionamos abaixo os intervalos onde as condições do teorema (2.8) foram violadas, a ordem usada pela rotina e o máximo de passos constantes.

[0.79,1.33] com ordem 3 e nº de passos 2

[4.24,4.70] com ordem 4 e nº de passos 2 .

Quando diminuímos a tolerância para o erro local para 10^{-6}, a rotina GEAR foi abortada em $x = 1.4719244$ com a mensagem que o tamanho do passo h tinha se tornado tal que $x+h = x$. Acreditamos que este fato é uma conseqüência clara da instabilidade que reduziu h para valores não significativos, relativamente a x, no computador usado.

6.3. *Problemas com Singularidade em Derivadas de Ordem Baixa*

Há problemas que apresentam dificuldades especiais quando se quer sua solução numérica, e para eles, em geral, é necessária a aplicação de métodos especiais. Os exemplos seguintes dão uma idéia do comportamento das rotinas STEP/DE e GEAR frente a algumas destas situações.

O problema

$$y' = \begin{cases} \frac{2}{3}x^{-1/3} & \text{se } x \neq 0 \text{ , } y(-1) = 1 \\ 0 & \text{se } x = 0 \end{cases}$$

cuja solução é $y(x) = x^{2/3}$ possui uma singularidade integrável em 0.

As rotinas podem resolver satisfatoriamente este problema em [-1,1] com tolerância 10^{-6} para o erro local absoluto, conforme mostra a tabela 6.4 que dá o erro global em $x = 1$.

	|ERRO GLOBAL|
STEP/DE	3.334E-05
GEAR	4.482E-05

Tabela 6.4

No entanto a solução de tais problemas, nas vizinhanças do ponto de singularidade se torna dispendiosa. A tabela 6.5 mostra o número de avaliações da função f necessárias para atingir o ponto x e o erro global nesse ponto.

	NÚMERO DE AVALIAÇÕES		|ERRO GLOBAL|	
x	STEP/DE	GEAR	STEP/DE	GEAR
-0.8	27	18	0.59E-07	0.95E-06
-0.6	35	23	0.59E-07	0.18E-05
-0.4	41	31	0.35E-06	0.42E-05
-0.2	56	49	0.50E-06	0.83E-05
0	144	193	0.62E-05	0.46E-04
0.2	369	314	0.33E-04	0.47E-04
0.4	391	323	0.33E-04	0.46E-04
0.6	401	329	0.33E-04	0.46E-04
0.8	409	333	0.33E-04	0.45E-04
1.0	415	336	0.33E-04	0.45E-04

Tabela 6.5

A solução de problemas como o acima exige que a rotina localize o ponto de singularidade, passe por este usando tamanho de passo pequeno e ordem baixa, e a seguir aumente rapidamente o tamanho do passo para que o custo computacional não se torne dispendioso. Observando a tabela 6.5 vemos que no ponto zero as rotinas STEP/DE fornecem resultados melhores com menos avaliações, isto se deve ao fato que esta rotina tem mais facilidade em localizar o ponto de descontinuidade. Nos pontos subseqüentes a rotina GEAR é superior, isto porque STEP é mais cautelosa em aumentar o passo.

Seja o problema

$$y' = e^{y}$$

$$y(-2) = -1.0986123$$

cuja solução exata é $y(x) = \ell n(\frac{1}{1-x})$ e portanto existe somente no intervalo $(-\infty,1)$. Quando resolvemos este problema pelas rotinas STEP/DE e GEAR no intervalo $[-2,1]$ com tolerância 10^{-7} para o erro local absoluto observamos o seguinte:

As rotinas STEP/DE conseguem dar a resposta $y(1) = 533.15$, dando porém antes várias mensagens de que ocorreu "overflow",(*) o que óbviamente é devido à não existência da solução no ponto 1.

A rotina GEAR porém não consegue alcançar o ponto 1, abortanto a execução no ponto 0.9999968, com erro global aproximadamente 24. nesse ponto. Vemos portanto que é muito perigoso tentar calcular a solução num ponto no qual ela não existe.

Sugerimos que, antes de se tentar resolver um problema por qualquer das rotinas aqui citadas, estejamos certos de que a solução existe no ponto desejado.

Para ilustrar como as nossas rotinas processam problemas que tenham descontinuidade (com grande salto) em derivadas de ordem baixa, resolvemos o problema (ver [16, p. 257]).

$$y' = \begin{cases} y & \text{se } 0 \le x \le 1 \\ -y & \text{se } 1 < x \le 2 \end{cases}$$

cuja solução exata é:

(*) No sistema utilizado é permitido até 15 ocorrências de "overflow", sendo que os valores que ultrapassam o limite de 10^{38} são substituídos por este para continuar o cálculo.

$$y(x) = \begin{cases} \exp(x) & \text{se } 0 \le x \le 1 \\ \exp(2-x) & \text{se } 1 < x \le 2, \end{cases}$$

no intervalo [0,2] com tolerância 10^{-7} para o erro local absoluto.

Ambas as rotinas resolvem satisfatoriamente este problema, obtendo erro global 10^{-6}. Queremos apenas salientar que o custo computacional é onerado quando passamos pelo ponto x=1. Dissemos, quando descrevíamos o algorítmo de seleção de passo, que as rotinas STEP/DE devem colocar a ordem igual a 1, e diminuir o tamanho do passo quando passam por um ponto de descontinuidade. Para ilustrar este fato mostramos na tabela 6.6: o intervalo em que o programa usou ordem 1, o maior tamanho de passo usado nesse intervalo e o tamanho de passo médio usado no intervalo todo ([0,2]).

intervalo onde tivemos ordem 1	maior h usado nesse intervalo	h médio usado em [0,2]
[0.92,1.001]	0.0004883	0.025

Tabela 6.6

6.4. *Problemas "STIFF"*

Como já havíamos chamado a atenção, as rotinas STEP/DE e a implementação dos métodos de Adams da rotina GEAR não são

apropriadas para resolver problemas "stiff". Para ilustrar tal fato resolvemos o problema abaixo, dado por Lambert, em [13, p. 229].

$$\begin{bmatrix} y_1' \\ y_2' \\ y_3' \end{bmatrix} = \begin{bmatrix} -21 & 19 & -20 \\ 19 & -21 & 20 \\ 40 & -40 & -40 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

$$[y_1(0),\ y_2(0),\ y_3(0)]^t = [1,0,-1]^t,$$

cuja solução é:

$$y_1(x) = \frac{1}{2}e^{-2x} + \frac{1}{2}e^{-40x}(\cos 40x + \operatorname{sen} 40x)$$

$$y_2(x) = \frac{1}{2}e^{-2x} - \frac{1}{2}e^{-40x}(\cos 40x + \operatorname{sen} 40x)$$

$$y_3(x) = -e^{-40x}(\cos 40x - \operatorname{sen} 40x).$$

Este problema foi resolvido em $[0,10]$ com tolerância 10^{-5} para o erro local absoluto. Na tabela 6.7 damos o erro global em $x = 10$, o número total de avaliações da função f e o TP utilizado. Os resultados foram obtidos com as rotinas STEP/DE, GEAR e GEAR 1 que é uma opção oferecida pela rotina GEAR para a solução de problemas "stiff" através da implementação de métodos específicos para tais problemas.

	$\|\|ERRO\|\|_\infty$	Nº AVALIAÇÕES	TP em segundos
STEP/DE	1.0E-7	1020	3.85
GEAR	1.0E-7	1373	4.03
GEAR 1	1.0E-8	396	1.27

Tabela 6.7

Como vemos, as rotinas STEP/DE e GEAR resolvem satisfatoriamente o problema, desde que não estejamos preocupados com o alto custo computacional delas. Já a rotina GEAR 1 resolve o problema com muito maior eficiência.

De acordo com a estratégia para detectar se um problema é "stiff", descrita no final do Capítulo V, as rotinas STEP/DE dizem que tal ocorre, em $x = 9.27$. Podemos observar também que na sua maioria os passos são executados com métodos de ordem menor que 5.

Em geral os métodos numéricos para equações diferenciais baseiam-se em valores da solução em pontos anteriores para obtê-la no próximo ponto. Como consequência, problemas cujas soluções oscilam muito, oferecem certa dificuldade ao serem resolvidos, principalmente se desejamos grande precisão nos resultados. O problema abaixo, dado por Lambert em [6,p. 37], é um exemplo em que as soluções são extremamente oscilantes nas vizinhanças da origem.

$$\begin{bmatrix} y_1' \\ y_2' \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{x} \\ -\frac{1}{x} & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$[y_1(x_o), y_2(x_o)]^t = [0,1]^t,$$

onde $x_o = e^{\frac{-5\pi}{2}} \simeq 0.388203204E-03$.

A solução é:

$$y_1(x) = \cos(\ell n(x))$$

$$y_2(x) = -\text{sen}(\ell n(x)) .$$

Resolvemos este problema em $[x_o,50]$ com tolerância 10^{-7} para o erro local absoluto. As rotinas STEP/DE, devido ao seu algorítmo de seleção de passo, resolvem este problema muito bem, atingindo o ponto $x = 50$, com erro global de $0(10^{-6})$ e usando apenas 600 avaliações da função. Já a rotina GEAR não consegue resolver o problema com tolerância 10^{-7}. Aumentamos então a tolerância para 10^{-4}, tendo obtido a solução em $x = 50$, com erro global de $0(10^{-3})$ e usando 170 avaliações de f.

Como pudemos observar nos testes aqui apresentados, as rotinas STEP/DE levam certa vantagem sobre a rotina GEAR nos problemas não "stiff", enquanto esta é superior nos problemas "stiff"(desde que usemos a implementação específica para tais problemas-GEAR 1).

Testes realizados por Shampine e apresentados em [18], nos chamam a atenção para a rotina RKF45, escrita por H.A. Watts e L.F. Shampine. Esta rotina implementa o método de Runge-Kutta-Felberg de ordem 4, com extrapolação local e passo variável.

RKF45 é uma rotina simples e fácil de se usar, além de ser muito eficiente quando a tolerância não é muito pequena

(maior que 10^{-4} no computador que usamos), ou quando a função f não é muito "difícil" de calcular (ver [18,p.405]).

Por essa razão montamos uma única rotina DRIVER que compatibiliza o uso conjunto das rotinas STEP/DE, GEAR e RKF45.

Esta rotina decide automaticamente (caso o usuário queira) qual rotina usar. Essa decisão é feita baseada principalmente em argumentos de Shampine [18].

Se estão sendo usadas as rotinas STEP/DE, e estas detectam que o problema é "stiff", DRIVER faz com que o problema passe a ser resolvido pela rotina apropriada para tais problemas, (GEAR), dando antes a mensagem apropriada.

Se está sendo usada a rotina RKF45, e o problema está exigindo muitos passos para atingir o ponto desejado, o problema passa a ser resolvido através das rotinas STEP/DE.

A rotina DRIVER permite ainda o uso independente das rotinas acima, bastando para isso que o usuário faça a opção adequada.

Observação: Este programa, e o seu manual de uso, podem ser obtidos no Instituto de Ciências Matemáticas de São Carlos, Departamento de Ciências de Computação e Estatística. As listagens das rotinas STEP/DE, GEAR e RKF45 podem também ser obtidas em [16], [9] e [2] respectivamente.

BIBLIOGRAFIA

[1] - ENRIGHT,W.H.;HULL,T.E. - Test results on initial value methods for non-stiff ordinary differential equations. *SIAM J. Num. Analysis*, 13(1976) p. 944-966.

[2] - FORSYTHE,G.E.;MALCOLM,M.A.;MOLER,C.B. - Computer Methods for Mathematical Computation. Prentice-Hall (1977).

[3] - GEAR,C.W. - Numerical Initial Value Problems in Ordinary Differential Equations. Prentice-Hall (1971).

[4] - GEAR,C.W.;TU,K.W. - The effect of variable mesh step size on the stability of multistep methods. *SIAM J. Num. Analysis*, 5(1974) p. 1025-1043.

[5] - GEAR,C.W.;WATANABE,D.S. - Stability and convergence of variable order multistep methods. *SIAM J. Num. Analysis*, 5(1974), p. 1044-1058.

[6] - GLADWELL,I.;SAYERS,D.K. - Computacional Tecniques for Ordinary Differential Equations, Chap 2 - Stiffness - - J.D.Lambert. Academic Press (1980).

[7] - HARTMAN,P. - Ordinary Differential Equations. John Wiley & Sons, Inc. (1964).

[8] - HENRICI,P. - Discrete Variable Methods in Ordinary Differential Equations. John Wiley & Sons, Inc. (1968).

[9] - HINDMARSH,A.C. - GEAR: Ordinary differential equation system solver. Lawrence Livermore Laboratory, Rept, UCID-30001,Rev. 3, (1974).

[10] - HULL,T.E.;ENRIGHT,W.H.; FELLEN,B.M.; SEDGWICK,A.E. - - Comparing numerical methods for ordinary differential equations.
SIAM J. Num. Analysis, 4(1972) p. 603-636.

[11] - KROGH,F.T. - Changing step size in the integration of differential equations using modified divided differences. Proceedings of the conference on the numerical solution of ordinary differential equations. Lecture Notes in Mathematics, 362 (1974), p. 22-71.

[12] - KROGH,F.T. - On testing a subroutine for the numerical integration of ordinary differential equations. *Journal of the Association for Computing Machinery*, 4(1973) p. 545-562.

[13] - LAMBERT,J.D. - Computational Methods in Ordinary Differential Equations. John Wiley & Sons, Inc. (1973).

[14] - NORDSIECK, A. - On the numerical integration of ordinary differential equations. *Math. Comp.*,16(1962) p. 22-49.

[15] - SHAMPINE,L.F. - Limiting precision in differential equation solvers. *Mathematics of Computation* ,28(1974),p.141-144.

[16] - SHAMPINE,L.F.;GORDON,M.K. - Computer Solution of Ordinary Differential Equation. W.H. Freeman (1975).

[17] - SHAMPINE,L.F.;WATTS,H.A. - Global error estimation for ordinary differential equations. *ACM Transactions on Mathematical Software*, 2(1976), p. 172-186.

[18] - SHAMPINE,L.F.;WATTS,H.A.;DAVENPORT,S.M. - Solving nonstiff ordinary differential equations - The state of the art.
SIAM Rev., 3(1976) p. 376-411.