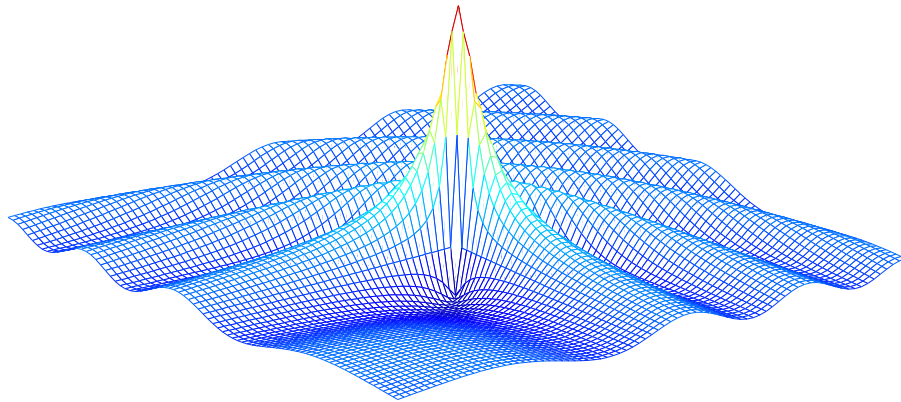


UNIVERSIDADE FEDERAL DE UBERLÂNDIA  
FACULDADE DE MATEMÁTICA

# CÁLCULO NUMÉRICO



PROF. JOSÉ EDUARDO CASTILHO

MARÇO DE 2001

# Conteúdo

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	O MatLab . . . . .	3
1.1.1	Cálculo na Janela de Comandos . . . . .	3
1.1.2	M-arquivos . . . . .	7
1.2	Exercícios . . . . .	10
<b>2</b>	<b>Zeros de Funções</b>	<b>11</b>
2.1	Isolamento das Raízes . . . . .	11
2.2	Refinamento . . . . .	14
2.3	Método da Bissecção . . . . .	14
2.3.1	Estudo da Convergência . . . . .	15
2.3.2	Estimativa do Número de Iterações . . . . .	16
2.4	Método Iterativo Linear (M.I.L.) . . . . .	17
2.4.1	Critério de Parada . . . . .	19
2.5	Método de Newton-Raphson (M.N.R) . . . . .	20
2.6	Ordem de Convergência . . . . .	22
2.7	Observações Finais . . . . .	24
2.8	Exercícios . . . . .	25
<b>3</b>	<b>Sistemas Lineares</b>	<b>27</b>
3.1	Métodos Diretos . . . . .	27
3.1.1	Sistema Triangular Superior . . . . .	28
3.1.2	Método de Eliminação de Gauss . . . . .	29
3.1.3	Pivotamento Parcial . . . . .	32
3.1.4	Cálculo da Matriz Inversa . . . . .	34
3.2	Métodos Iterativos . . . . .	38
3.2.1	Critério de Convergência . . . . .	40
3.2.2	Método Iterativo de Gauss-Jacobi . . . . .	40
3.2.3	Critério das Linhas . . . . .	42
3.2.4	Método Iterativo de Gauss-Seidel . . . . .	44
3.2.5	Critério de Sassenfeld . . . . .	45
3.3	Observações Finais . . . . .	46

3.4	Exercícios . . . . .	47
<b>4</b>	<b>Ajuste de Curvas: Método dos Mínimos Quadrados</b>	<b>49</b>
4.1	Método dos Mínimos Quadrados - Caso Discreto . . . . .	49
4.2	Método dos Mínimos Quadrados - Caso Contínuo . . . . .	53
4.3	Ajuste Não Linear . . . . .	55
4.4	Observações Finais . . . . .	56
4.5	Exercícios . . . . .	58
<b>5</b>	<b>Interpolação Polinomial</b>	<b>60</b>
5.1	Forma de Lagrange . . . . .	62
5.2	Forma de Newton . . . . .	63
5.2.1	Construção do Polinômio . . . . .	64
5.3	Estudo do Erro . . . . .	65
5.4	Escolha dos Pontos . . . . .	67
5.5	Interpolação Inversa . . . . .	67
5.6	Observações Finais . . . . .	68
5.7	Exercícios . . . . .	70
<b>6</b>	<b>Integração Numérica - Fórmulas de Newton Côtes</b>	<b>72</b>
6.1	Regra do Trapézio . . . . .	72
6.2	Cálculo do Erro . . . . .	73
6.3	Regra do Trapézio Repetida . . . . .	75
6.4	Regra de Simpson 1/3 . . . . .	76
6.5	Regra de Simpson Repetida . . . . .	78
6.6	Observações Finais . . . . .	79
6.7	Exercícios . . . . .	79
<b>7</b>	<b>Equações Diferenciais Ordinárias (P.V.I.)</b>	<b>81</b>
7.1	Método Euler . . . . .	81
7.2	Métodos da Série de Taylor . . . . .	83
7.3	Métodos de Runge-Kutta . . . . .	85
7.4	Métodos de Adams-Bashforth . . . . .	88
7.4.1	Métodos Explícitos . . . . .	89
7.4.2	Métodos Implícitos . . . . .	90
7.5	Equações de Ordem Superior . . . . .	92
7.6	Exercícios . . . . .	93

# Capítulo 1

## Introdução

O Cálculo Numérico tem por objetivo estudar esquemas numéricos (algoritmos numéricos) para resolução de problemas que podem ser representados por um modelo matemático. Um esquema é eficiente quando este apresenta soluções dentro de uma precisão desejada com custo computacional (tempo de execução + memória) baixo. Os esquemas numéricos nos fornecem aproximações para o que seria a solução exata do problema. Os erros cometidos nesta aproximação são decorrentes da discretização do problema, ou seja passar do modelo matemático para o esquema numérico, e da forma como as máquinas representam os dados numéricos.

Como exemplo de discretização consideremos que desejamos calcular uma aproximação para a derivada de uma função  $f(x)$  num ponto  $\bar{x}$ . O modelo matemático é dado por

$$f'(\bar{x}) = \lim_{h \rightarrow 0} \frac{f(\bar{x} + h) - f(\bar{x})}{h}$$

Um esquema numérico para aproximar a derivada é dado por tomar  $h$  “pequeno” e calcular

$$f'(\bar{x}) \approx \frac{f(\bar{x} + h) - f(\bar{x})}{h}$$

Neste caso quanto menor for o valor de  $h$  mais preciso será o resultado, mas em geral, este esquema não fornecerá a solução exata.

A representação de números em máquinas digitais (calculadoras, computadores, etc) é feita na forma de ponto flutuante com um número finito de dígito. Logo os números que tem representação infinita (Ex.  $1/3, \pi, \sqrt{2}$ ) são representados de forma truncada. Com isto algumas das propriedades da aritmética real não valem na aritmética computacional. Como exemplo, na aritmética computacional temos

$$\sum_{k=0}^n \frac{a_k}{N} \neq \frac{1}{N} \sum_{k=0}^n a_k,$$

onde estamos considerando que no primeiro somatório para cada  $k$  fazemos  $a_k/N$  e depois somamos e no segundo somatório somamos todos os  $a_k$  e o resultado da soma dividimos por

$N$ . Do ponto de vista analítico, as duas expressões são equivalentes, mas a segunda forma apresenta melhor resultado do ponto de vista computacional, pois realiza menos operações e comete menos erro de truncamento. Outro exemplo interessante é que em aritmética computacional é possível que para um dado  $A$  exista um  $\varepsilon \neq 0$  tal que

$$A + \varepsilon = A.$$

Analiticamente a expressão acima é verdadeira se e somente se  $\varepsilon = 0$ .

Outro fator que pode influenciar no resultado é o tipo de máquina em que estamos trabalhando. Numa calculadora simples que represente os números com 7 dígitos teríamos

$$1/3 + 1/3 + 1/3 = 0.9999999$$

Enquanto que calculadoras mais avançadas teríamos como resposta um **falso 1**, pois na realidade, internamente estas calculadoras trabalham com mais dígitos do que é apresentado no visor e antes do resultado ser apresentado este é arredondado.

Os esquemas numéricos são classificados como esquemas diretos e esquemas iterativos. Os esquemas diretos são aqueles que fornecem a solução após um número finito de passos. Por exemplo o esquema que apresentamos para o cálculo da derivada. Os esquemas iterativos são aqueles que repetem um número de passos até que um critério de parada seja satisfeito. Como exemplo considere o algoritmo que é usado para determinar a precisão de uma máquina digital

**Algoritmo:** Epsilon da Máquina

$Ep \leftarrow 1$

Enquanto  $(1 + Ep) > 1$ , faça:

$Ep \leftarrow Ep/2$

fim enquanto

**OutPut:**  $2Ep$

O critério de parada é  $(1 + Ep) \leq 1$  e cada execução do laço **Enquanto** é chamado de iteração. Máquinas diferentes apresentarão resultados diferentes.

Um outro fator que pode influenciar nos resultados é a linguagem de programação usada na implementação dos algoritmos (Pascal, Fortran, C++, MatLab, etc). Diferentes linguagens podem apresentar diferentes resultados. E mesmo quando usamos uma mesma linguagem, mas compiladores diferentes (Ex. C++ da Borland e C++ da Microsoft), os resultados podem apresentar diferenças. Existem várias bibliotecas de rotinas numéricas em diversas linguagens e algumas disponíveis na Internet. Um exemplo é a LIMPACK: uma coleção de rotinas em Fortran para solução de sistemas lineares.

Para exemplificar os esquemas numéricos, que estudaremos nos próximos capítulos, usaremos o MatLab pela sua facilidade de programação. Todo o material desta apostila é baseado nas referências: [?] e [?].

## 1.1 O MatLab

O MatLab surgiu nos anos 1970 como um *Laboratório de Matrizes* para auxiliar os cursos de Teoria Matricial, Álgebra Linear e Análise Numérica. Hoje, a capacidade do MatLab se estende muito além da manipulação de matrizes. Ele é tanto um ambiente quanto uma linguagem de programação, e um de seus aspectos mais poderosos é que os problemas e as soluções são expressos numa linguagem matemática bem familiar. Devido a sua capacidade de fazer cálculos, visualização gráfica e programação, num ambiente de fácil uso, o MatLab torna-se uma ferramenta eficiente para a compreensão tanto de tópicos fundamentais quanto avançados a uma gama de disciplinas. Nosso objetivo é dar uma rápida visão dos comandos e funções básicas do MatLab para exemplificar os tópicos do curso de Cálculo Numérico. Maiores detalhes podem ser obtidos em ??.

Apesar das últimas versões do MatLab ter expandido sua capacidade, o elemento básico dos dados ainda é um vetor, o qual não requer declaração de dimensão ou tipo de variável. O MatLab é um sistema interativo, onde os comandos podem ser executados na janela de comandos ou por programas. Estes programas são conhecidos como m-arquivos (ou arquivos com extensão .m) e serão discutidos posteriormente. Em primeiro lugar vamos discutir alguns comandos básicos que serão úteis para a manipulação de dados na janela de comandos e nos m-arquivos.

### 1.1.1 Cálculo na Janela de Comandos

Um cálculo simples pode ser executado na janela de comandos digitando as instruções no prompt como você faria numa calculadora. Por exemplo

```
EDU>> 3*4 +5
```

```
ans =
```

```
17
```

o resultado é mostrado na tela como ans ( abreviatura de “answer”). Os símbolos dos operadores aritméticos são dados na Tabela 1.1. As expressões são calculadas da esquerda para a direita, com a potenciação tendo a maior precedência, seguido da multiplicação e divisão (mesma precedência) e pela adição e subtração (também com mesma precedência).

#### As Variáveis

A forma de armazenar o resultado para uso posterior é pelo uso de variáveis.

```
EDU>> s=3+4+7+12
```

```
s =
```

```
26
```

Tabela 1.1: Operadores Aritméticos

Operação	Símbolo
Adição	$a + b$
Multiplicação	$a * b$
Subtração	$a - b$
Divisão	$a/b$ ou $b \setminus a$
Potenciação	$a^b$

O nome da variável pode consistir de no máximo 31 caracteres, iniciando sempre por um caracter alfa seguido de qualquer combinação de caracteres do tipo alfa, numérico e underscores. Ex. `resultado_da_soma_2`. Ao contrário de outras linguagens, o **MatLab** diferencia as variáveis que usam letras minúsculas e maiúsculas. Isto é as variáveis **Contas**, **contas**, **conTas** e **CoNtAs**, são consideradas como quatro variáveis diferentes. Todas as variáveis são armazenadas internamente e podem ser usadas a qualquer momento. Para saber quais as variáveis que estão ativas utilizamos o comando **who**. Para eliminar a variável **conta**, usamos o comando **clear conta**. As variáveis são tratadas como matrizes, apesar dos escalares não serem apresentados na notação matricial. Um vetor linha pode ser definido como

```
EDU>> x=[1 2 3 4]
```

```
x =
     1     2     3     4
```

Também podemos separar os elementos por vírgula. Já um vetor coluna pode ser definido da seguinte forma

```
EDU>> y=[5; 6; 7; 8]
```

```
y =
     5
     6
     7
     8
```

Um exemplo de uma matriz  $3 \times 4$ .

```
EDU>> a=[1 2 3 4; 5 6 7 8; 9 10 11 12]
```

```
a =
     1     2     3     4
     5     6     7     8
     9    10    11    12
```

Os elementos na  $i$ -ésima linha e na  $j$ -ésima coluna, denotados por  $a_{ij}$  podem ser obtidos pelo comando `a(i,j)`, por exemplo `a(2,3)=7`. Note que os elementos no `MatLab` são indexados iniciando em 1. Em algumas situações necessitamos de vetores com alguma estrutura particular. Por exemplo, um vetor cujo o primeiro termo vale  $-2$  e o ultimo vale  $3$  e os termos intermediários variam um passo de  $0.5$ . Este vetor pode ser definido pela linha de comando

```
EDU>> v=-2:0.5:3
```

```
v =
```

```
Columns 1 through 7
```

```
-2.0000   -1.5000   -1.0000   -0.5000           0    0.5000    1.0000
```

```
Columns 8 through 11
```

```
1.5000    2.0000    2.5000    3.0000
```

De uma forma geral este comando tem a sintaxe `v=a:passo:b`. Quando o passo é igual a um podemos escrever o comando na forma reduzida `v=a:b`. Algumas matrizes elementares podem ser geradas através de comandos simples, por exemplo:

A matriz identidade:

```
EDU>> I=eye(3)
```

```
I =
```

```
1     0     0
0     1     0
0     0     1
```

Matriz  $n \times m$  formada por 1's:

```
EDU>> A=ones(2,3)
```

```
A =
```

```
1     1     1
1     1     1
```

Matriz Nula de ordem  $n \times m$ :

```
EDU>> B=zeros(3,4)
```

```
B =
```

```
0     0     0     0
0     0     0     0
0     0     0     0
```



## Operações com Matrizes e Vetores

As operações de subtração, adição e multiplicação entre matrizes são definidas como o usual. O símbolo da divisão representa o cálculo da multiplicação pela inversa da matriz, por exemplo

```
EDU>> A=[1 2 1;3 2 4;5 3 2];
```

```
EDU>> A/A
```

```
ans =
     1     0     0
     0     1     0
     0     0     1
```

Note que neste exemplo terminamos o comando que define a matriz  $A$  com um ponto e vírgula. Isto faz com que o resultado do comando (ou de uma expressão em geral) não seja apresentado no monitor. Isto é útil para programas de grande porte, pois este processo de apresentar os resultados no monitor consome muito tempo de execução. Entre vetores e matrizes de mesma dimensão é possível operar elemento com elemento. Isto é possível através de uma notação especial, que consiste de usar um ponto antes do símbolo da operação. Na Tabela 1.2 damos um resumo destas operações aplicada a dois vetores  $a$  e  $b$  de dimensão  $n$ .

Tabela 1.2: Operações Elementares entre Vetores

Operação	Símbolo	Resultado
Adição	$a+b$	$[a_1 + b_1, a_2 + b_2, a_3 + b_3, \dots, a_n + b_n]$
Subtração	$a-b$	$[a_1 - b_1, a_2 - b_2, a_3 - b_3, \dots, a_n - b_n]$
Multiplicação	$a.*b$	$[a_1 * b_1, a_2 * b_2, a_3 * b_3, \dots, a_n * b_n]$
Divisão	$a./b$	$[a_1/b_1, a_2/b_2, a_3/b_3, \dots, a_n/b_n]$
Potenciação	$a.^b$	$[a_1^{b_1}, a_2^{b_2}, a_3^{b_3}, \dots, a_n^{b_n}]$

Como na maioria das linguagens de programação, o **MatLab** oferece diversas funções elementares que são importantes em matemática. A Tabela 1.3 apresenta uma lista destas funções e sua sintaxe.

## Gráficos

Para plotar um gráfico no **MatLab**, devemos criar dois vetores de mesma dimensão  $x$  e  $f$ , onde  $x$  corresponde aos valores do eixo  $x$  e  $f$  os valores da função nestes pontos. O

Tabela 1.3: Funções Elementares

Função	Sintaxe
Valor Absoluto	abs(x)
Arco Co-seno	acos(x)
Arco Seno	asin(x)
Co-seno	cos(x)
Exponencial $e^x$	exp(x)
Logaritmo Natural	log(x)
Logaritmo base 10	log10(x)
Seno	sin(x)
Raiz Quadrada	sqrt(x)
Tangente	tan(x)

gráfico é gerado pelo comando **plot(x,f)**. Se desejamos gerar o gráfico da função  $\text{sen}(x)$  no intervalo  $[-\pi, \pi]$  devemos proceder da seguinte forma:

```
EDU>> x=-pi:0.01:pi;
EDU>> f=sin(x);
EDU>> plot(x,f)
```

Note, que na definição do vetor  $x$ , usamos o passo igual a 0.01. Isto determina a quantidade de pontos que o comando **plot** usa par gerar o gráfico. Quanto mais pontos mais perfeito será o gráfico (em contra partida maior o tempo de execução). se tivéssemos usado o passo 0.5 não teríamos um gráfico de boa qualidade.

### 1.1.2 M-arquivos

Existem dois tipos de programas em **MatLab** : scripts e funtions. Ambos devem ser salvos com extensão **.m** no diretório corrente. Uma diferença básica entre os dois é que os scripts trata as variáveis, nele definidas, como variáveis globais, enquanto as functions trata as variáveis como variáveis locais. Desta forma a functions tem que ter um valor de retorno.

#### Scripts

Os scripts permite que um conjunto de comandos e definições sejam executados através de um único comando na janela de comandos. Como exemplo, o script seguinte calcula a aproximação da derivada de  $\text{sen}(x)$  usando diferenças finitas.

```
% Aproximacao da derivada do seno
% Usando o operador de diferen\c{c}a finita progressiva.
```

```
clear;
h=0.0001;
x=input('Entre com o valor de, x=');      % Atribui Valores a x
disp('O valor da aproximacao eh...')      % Mostra mensagem no monitor
dsen=(sin(x+h)-sin(x))/h
```

As primeiras duas linha são comentários que descrevem o script. Na quinta linha temos o comando que permite atribuir valores a uma variável. E na sexta linha o comando que permite mostrar uma mensagem no monitor. Vamos supor que este arquivo seja salvo com o nome de `devira_seno.m`. Para executar o script digitamos seu nome no prompt do **MatLab**

## Functions

Numa função em **MatLab** a primeira linha é da forma **function y=nome(argumentos)**. A função se troca informações com o **MatLab** workspace por intermédio da variável **y** e dos argumentos. Para ilustrar o uso de funções em **MatLab** considere o seguinte código

```
function dsen=deriva_seno(x,h)
% Aproximacao da derivada do seno
% Usando o operador de diferen\c{c}a finita progressiva.
dsen=(sin(x+h)-sin(x))/h;
```

Apesar deste arquivo poder ser salvo com um nome qualquer, é usual usar o mesmo nome da função, ou seja, `deriva_seno.m`. Para executá-lo devemos digitar seu nome e informar os valores dos argumentos, por exemplo,

```
EDU>>y= deriva_seno(3.14,0.001)
```

o que forneceria em **y** uma aproximação da derivada da função seno em 3.14 Uma diferença importante entre esta versão, usando **function**, com a anterior é que o valor calculado pode ser atribuído a uma variável. Além disso, agora podemos escolher o valor de **h**, que na versão anterior estava fixo em **h=0.0001**. Vale notar que no primeiro caso todas as variáveis do scrips estão ativas, isto é são variáveis globais. Enquanto que no segundo ca so as variáveis são locais, isto é, a variável **h** só é ativa na execução da função

## Controle de Fluxo

O controle de fluxo é um recurso que permite que resultados anteriores influenciem operações futuras. Como em outras linguagens, o **MatLab** possui recursos que permitem o controle de fluxo de execução de comandos, com base em estruturas de tomada de decisões. Apresentamos as estrutura de loops **for**, loops **while** e **if-else-end**. A forma geral do loop **for** é

```
for x = vetor
    comandos...
end
```

Os comandos entre **for** e **end** são executados uma vez para cada coluna de **vetor**. A cada iteração atribui-se a **x** a próxima coluna de **vetor**. Por exemplo

```
EDU>> for n=1:5
        x(n) = cos(n*pi/2);
    end
EDU>> x
```

```
x =
    0.0000   -1.0000   -0.0000    1.0000    0.0000
```

Traduzindo, isto diz que para **n** igual a 1 até 10 calcule os comandos até **end**.

Ao contrário do loop **for**, que executa um grupo de comandos um número fixo de vezes, o loop **while** executa um grupo um de comandos quantas vezes forem necessárias para que uma condição seja negada. Sua forma geral é

```
while expressao
    comandos...
end
```

O grupo de comandos entre **while** e **end** são executados até que a **expressão** assuma um valor falso. Por exemplo,

```
EDU>> while abs(x(n)-x(n-1)) > 10^(-6)
        x(n) = 2*x(n-1) + 1/4;
        n=n+1;
    end
```

Neste caso o grupo de comandos são executados até que o valor absoluto da diferença entre dois valores consecutivos seja menor ou igual a  $10^{-6}$ .

A estrutura **if-else-end** permite que grupos de comandos sejam executados por um teste relacional. A forma geral é dada por

```
if expressao
    comandos 1...
else
    comandos 2...
end
```

Se a **expressão** for verdadeira é executado o grupo de comandos 1, caso contrário é executado o grupo de comandos 2. Esta estrutura permite o uso da forma mais simples que envolve só um condicional

```

if expressao
  comandos ...
end

```

Como exemplo considere o seguinte fragmento de código que calcula o valor absoluto de um número

```

if x < 0
  x=-x;
end

```

Isto é, se  $x$  for menor que zero então troca de sinal, caso contrário nada é feito.

## 1.2 Exercícios

**Exercício 1.1** Usando o esquema numérico para a aproximação da derivada dado abaixo ache uma aproximação para  $f'(\pi)$ , onde  $f(x) = \text{sen}(x)$  e tome  $h = 0.1, 0.01, 0.001, \dots, 10^{-10}$ . Repita os cálculos para  $f'(0)$ . Comente os resultados.

$$f'(\bar{x}) \approx \frac{f(\bar{x} + h) - f(\bar{x})}{h}$$

**Exercício 1.2** Faça um programa que calcule

$$A + \sum_{k=1}^{10^7} 10^{-7}$$

com  $A = 10, 10^2, 10^3, \dots, 10^{15}$ . Comente os resultados.

**Exercício 1.3** Calcule a precisão de sua máquina usando o algoritmo

**Algoritmo:** *Epsilon da Máquina*

**Input:**  $A$  : número que represente a grandeza

$Ep \leftarrow 1$

Enquanto  $(A + Ep) > 1$ , faça:

$Ep \leftarrow Ep/2$

fim enquanto

**Output:** Imprimir  $2Ep$

tomando  $A = 1, 10, 100, 1000$ . Comente os resultados.

# Capítulo 2

## Zeros de Funções

Neste capítulo estudaremos esquemas numéricos para resolver equações da forma  $f(x) = 0$ . Na maioria dos casos estas equações não tem solução algébrica como há para as equações de 2º grau. No entanto esquemas numéricos podem fornecer uma solução satisfatória. O processo para encontrar uma solução envolve duas fases:

Fase I Isolamento das raízes - Consiste em achar um intervalo fechado  $[a, b]$  que contém a raiz.

Fase II Refinamento - Partindo de uma aproximação inicial refinamos a solução até que certos critérios sejam satisfeitos.

### 2.1 Isolamento das Raízes

Um número  $x$  que satisfaz a equação  $f(x) = 0$  é chamado de raiz ou zero de  $f$ . O objetivo é encontrar um intervalo  $[a, b]$ , de pequena amplitude ( $b - a \ll 1$ ), que contenha a raiz que desejamos encontrar. Para isto usaremos duas estratégias: Análise Gráfica e Tabelamento da função.

A análise gráfica é baseada na idéia de que, a partir da equação  $f(x) = 0$ , podemos obter uma equação equivalente  $g(x) - h(x) = 0$ , onde  $g$  e  $h$  sejam funções mais simples e de fácil análise gráfica. Esboçando o gráfico de  $g$  e  $h$  podemos determinar os pontos  $x$ , onde as curvas se interceptam, pois estes pontos serão as raízes de  $f(x)$  ( $g(\xi) = h(\xi) \Rightarrow f(\xi) = 0$ ).

**Exemplo 2.1.1** Sendo  $f(x) = e^{-x} - x$  temos  $f(x) = g(x) - h(x)$ , onde  $g(x) = e^{-x}$  e  $h(x) = x$ . Na Figura 2.1 temos que as curvas se interceptam no intervalo  $[0, 1]$ . Também podemos observar que pelo comportamento das funções  $g(x)$  e  $h(x)$  estas funções não vão se interceptar em nenhum outro ponto. Logo  $f(x)$  admite uma única raiz.

Na prática usamos algum software matemático para esboçar os gráficos. Quanto menor for a amplitude do intervalo que contém a raiz, mais eficiente será a Fase de Refinamento. Para obtermos um intervalo de menor amplitude usaremos a estratégia do tabelamento que é baseada no seguinte Teorema.

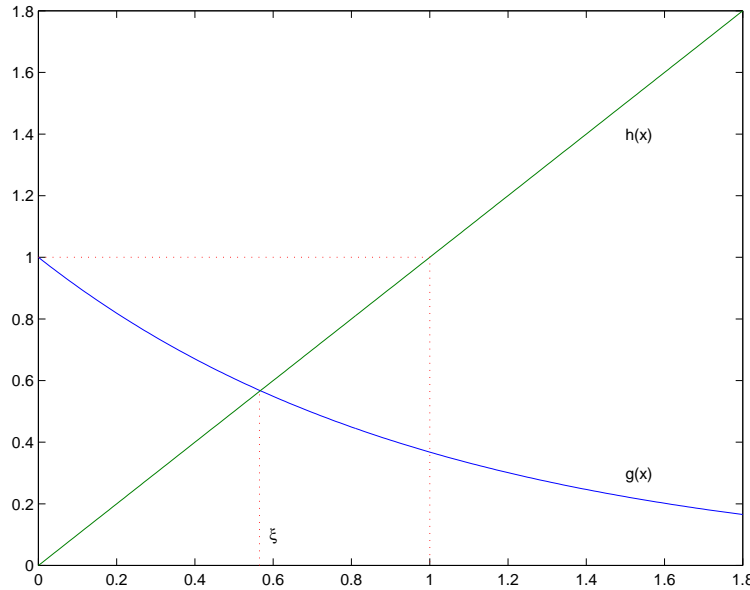


Figura 2.1: Gráficos de  $g(x)$  e  $h(x)$

**Teorema 2.1.1** *Seja  $f(x)$  uma função contínua num intervalo  $[a, b]$ . Se  $f(a)f(b) < 0$  então existe pelo menos uma raiz  $\xi \in [a, b]$ .*

O Teorema garante a existência de pelo menos uma raiz, mas pode ser que o intervalo contenha mais de uma raiz como mostra os exemplos na Figura 2.2.

Pelo exemplos podemos notar que se  $f'(x)$  preserva o sinal em  $[a, b]$  e  $f(a)f(b) < 0$ , então o intervalo contém uma única raiz. Se  $f(a)f(b) > 0$  não podemos afirmar nada sobre a existência ou não de raízes.

**Exemplo 2.1.2** *Da análise gráfica vimos que a função  $f(x) = e^{-x} - x$  tem uma raiz em  $[0, 1]$ . Tabelando a função para valores a partir de zero e espaçados de 0.25 temos*

$x$	0	0.25	0.5	0.75	1
$f(x)$	1	0.528	0.106	-0.277	-0.632

*Temos que  $f(0.5)f(0.75) < 0$ . Logo a raiz pertence ao intervalo  $[0.5, 0.75]$ . Note que  $f'(x) = -e^{-x} - 1 < 0 \forall x \in \mathbb{R}$ , isto é  $f'$  preserva o sinal em  $[a, b]$  e com isto podemos concluir que esta raiz é única.*

Devemos observar que o tabelamento é uma estratégia que completa a análise gráfica. Somente com o tabelamento não conseguimos determinar se existe outras raízes no intervalo ou ainda em que intervalo devemos tabelar a função.

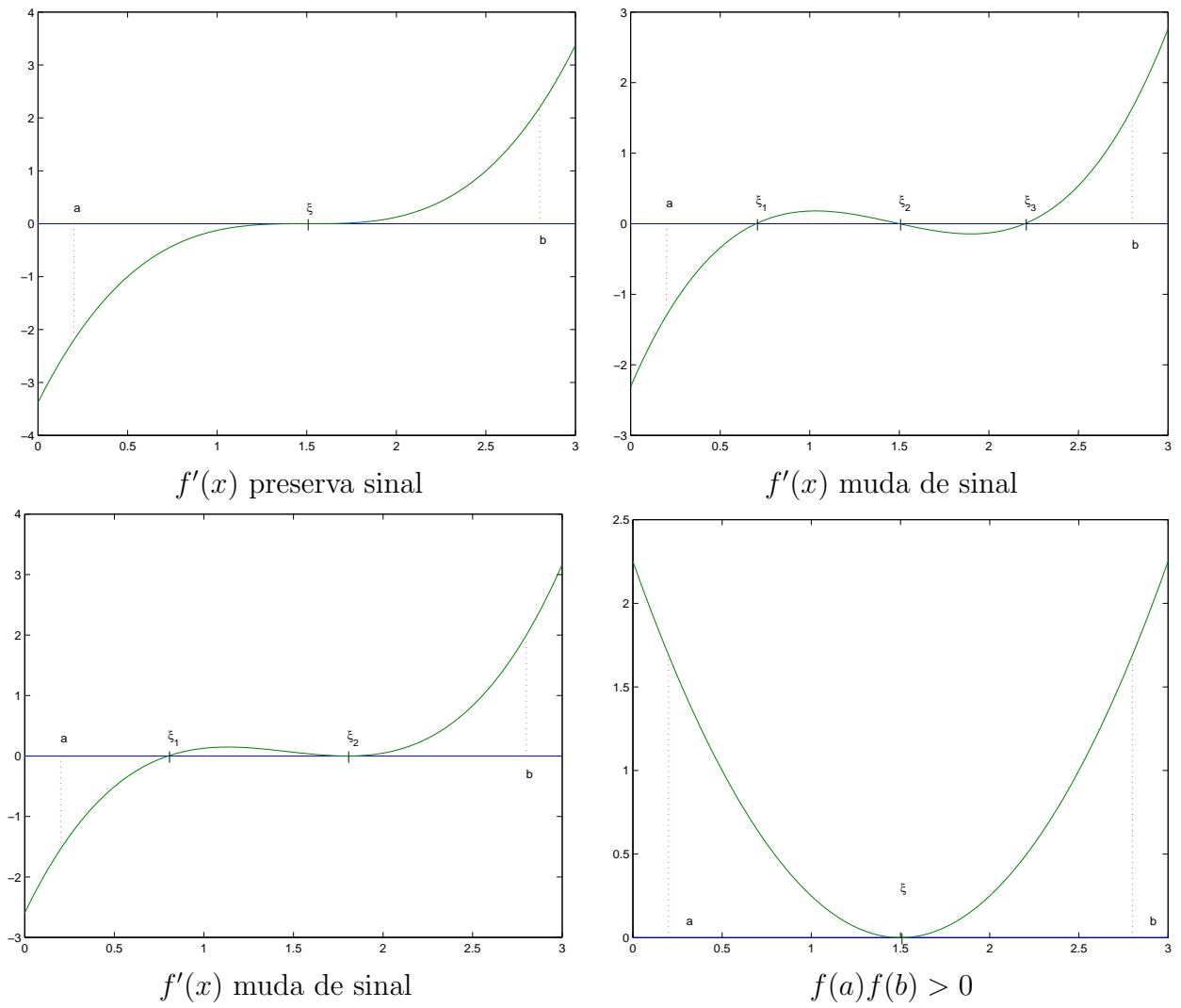


Figura 2.2: Exemplos do comportamento de  $f(x)$



## 2.2 Refinamento

Nas próximas seções estudaremos os esquemas numéricos que partindo de uma aproximação inicial  $x_0$ , vão gerar uma seqüência  $\{x_k\}$  que converge para a raiz procurada, isto é  $x_k \rightarrow \xi$  quando  $k \rightarrow \infty$ . A aproximação inicial parte do intervalo encontrado na Fase I, Isolamento das Raízes, e os termos da seqüência são calculados até que a aproximação tenha atingido uma precisão desejada (critério de parada).

## 2.3 Método da Bissecção

Este método é baseado no Teorema 2.1.1. Seja  $f(x)$  uma função contínua no intervalo  $[a, b]$  tal que  $f(a)f(b) < 0$  e seja  $\varepsilon > 0$  um número dado. A idéia é reduzir a amplitude do intervalo até atingir a precisão requerida:  $b - a < \varepsilon$ , usando divisão sucessivas do intervalo.

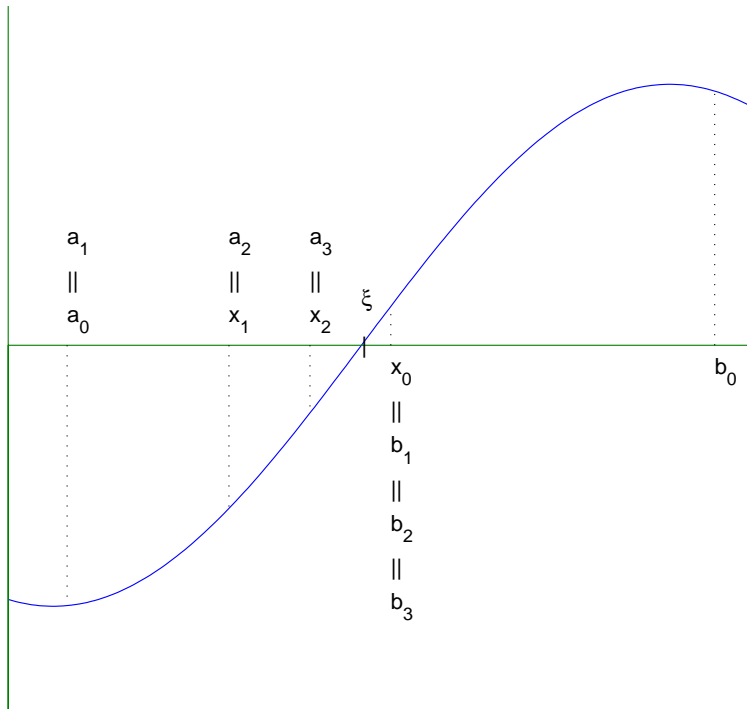


Figura 2.3: Método da Bissecção

O método procede da seguinte forma: faça  $[a_0, b_0] = [a, b]$ ,

$$x_0 = \frac{a_0 + b_0}{2} \Rightarrow \begin{cases} f(a_0) < 0 \\ f(b_0) > 0 \\ f(x_0) > 0 \end{cases} \Rightarrow \begin{cases} \xi \in (a_0, x_0) \\ a_1 = a_0 \\ b_1 = x_0 \end{cases}$$

$$x_1 = \frac{a_1 + b_1}{2} \Rightarrow \begin{cases} f(a_1) < 0 \\ f(b_1) > 0 \\ f(x_1) < 0 \end{cases} \Rightarrow \begin{cases} \xi \in (x_1, b_1) \\ a_2 = x_1 \\ b_2 = b_1 \end{cases}$$

$$x_2 = \frac{a_2 + b_2}{2} \Rightarrow \begin{cases} f(a_2) < 0 \\ f(b_2) > 0 \\ f(x_2) < 0 \end{cases} \Rightarrow \begin{cases} \xi \in (x_2, b_2) \\ a_3 = x_2 \\ b_3 = b_2 \end{cases}$$

E assim vamos calculando a seqüência  $x_k$  até que seja satisfeito o critério de parada

$$b_k - a_k < \varepsilon.$$

Este critério garante se tomarmos  $\bar{x} \in [a_k, b_k]$  teremos a garantia que o erro é menor que  $\varepsilon$ , isto é

$$|\bar{x} - \xi| \leq b_k - a_k < \varepsilon$$

Abaixo apresentamos a listagem do método implementado como função do MatLab.

```
% Disciplina de Cálculo Numérico - Prof. J. E. Castilho
% Método da Bisseccao
% Calcula uma aproximação para uma raiz de função f(x)
% definida no arquivo f.m, onde esta raiz pertence ao
% intervalo [ao,bo] e a predição dado por Ep.
```

```
function y=bissec(ao,bo,Ep)
while (bo-ao) > Ep,
    x=(ao+bo)/2;
    if f(x)*f(ao) > 0,
        ao=x;
    else
        bo=x;
    end;
end;
y=(ao+bo)/2;
```

### 2.3.1 Estudo da Convergência

A convergência é bastante intuitiva, como podemos ver na Figura 2.3. Vamos dar uma demonstração analítica através do seguinte teorema:

**Teorema 2.3.1** *Seja  $f$  uma função contínua em  $[a, b]$ , onde  $f(a)f(b) < 0$ . Então o método da Bisseccão gera uma seqüência  $\{x_k\}$  que converge para a raiz  $\xi$  quando  $k \rightarrow \infty$ .*

Prova: O método gera três seqüências:

$\{a_k\}$ : Seqüência não decrescente e limitada superiormente por  $b_0$ . Logo

$$a_0 \leq a_1 \leq \dots < b_0 \Rightarrow \exists M \in \mathbb{R} \text{ tal que } \lim_{k \rightarrow \infty} a_k = M$$

$\{b_k\}$ : Seqüência não crescente e limitada inferiormente por  $a_0$ . Logo

$$b_0 \geq b_1 \geq \dots > a_0 \Rightarrow \exists m \in \mathbb{R} \text{ tal que } \lim_{k \rightarrow \infty} b_k = m$$

$\{x_k\}$ : Por construção temos que

$$x_k = \frac{a_k + b_k}{2} \Rightarrow a_k < x_k < b_k \quad \forall k \in \mathbb{N} \quad (2.1)$$

A amplitude de cada intervalo gerado é metade da amplitude do intervalo anterior, assim temos,

$$b_k - a_k = \frac{b_0 - a_0}{2^k}.$$

Calculando o limite quando  $k \rightarrow \infty$  temos

$$\lim_{k \rightarrow \infty} (b_k - a_k) = \lim_{k \rightarrow \infty} \frac{b_0 - a_0}{2^k} = 0$$

Isto segue que

$$\lim_{k \rightarrow \infty} b_k - \lim_{k \rightarrow \infty} a_k = 0 \Rightarrow M - m = 0 \Rightarrow M = m.$$

Usando este fato e calculando o limite em (2.1) temos

$$m = \lim_{k \rightarrow \infty} a_k \leq \lim_{k \rightarrow \infty} x_k \leq \lim_{k \rightarrow \infty} b_k = m \Rightarrow \lim_{k \rightarrow \infty} x_k = m.$$

Falta mostrar que  $m$  é raiz de  $f$ , isto é  $f(m) = 0$ . Em cada iteração  $f(a_k)f(b_k) < 0$ . Como  $f$  é contínua temos então que

$$0 \geq \lim_{k \rightarrow \infty} f(a_k)f(b_k) = \lim_{k \rightarrow \infty} f(a_k) \lim_{k \rightarrow \infty} f(b_k) = f\left(\lim_{k \rightarrow \infty} a_k\right) f\left(\lim_{k \rightarrow \infty} b_k\right) = f^2(m) \geq 0$$

Portanto  $f(m) = 0$  ■

### 2.3.2 Estimativa do Número de Iterações

Pelo critério de parada podemos observar que o número de iterações depende do intervalo inicial  $[a_0, b_0]$  e da precisão requerida  $\varepsilon$ . Dada uma precisão  $\varepsilon$  temos,

$$b_k - a_k < \varepsilon \Rightarrow \frac{b_0 - a_0}{2^k} < \varepsilon \Rightarrow 2^k > \frac{b_0 - a_0}{\varepsilon}$$

Como estes valores são sempre positivos, podemos aplicar a função logaritmo, obtendo,

$$k > \frac{\log(b_0 - a_0) - \log(\varepsilon)}{\log(2)}$$

**Exemplo 2.3.1** No exemplo 2.1.2 isolamos uma raiz de  $f(x) = e^{-x} - x$  no intervalo  $[0.5, 0.75]$ . Usando a precisão  $\varepsilon = 10^{-8}$ , temos

$$k > \frac{\log(0.75 - 0.5) - \log(10^{-8})}{\log(2)} = 24.575.$$

Logo será necessário no mínimo 25 iterações para que o método da Bisseccção possa atingir a precisão desejada.

## 2.4 Método Iterativo Linear (M.I.L.)

Seja  $f(x)$  contínua em  $[a, b]$ , onde existe uma raiz da equação  $f(x) = 0$ . A estratégia deste método é escrever a função  $f$  de tal forma que  $f(x) = x - \phi(x)$ . Se  $f(x) = 0$ , então

$$x - \phi(x) = 0 \Rightarrow x = \phi(x)$$

Isto é, encontrar as raízes de  $f$  é equivalente a achar os pontos fixo da função  $\phi$ . Através da equação acima montamos um processo iterativo, onde, dado  $x_0$

$$x_{n+1} = \phi(x_n), \quad n = 1, 2, \dots$$

A função  $\phi$  é chamada de função de iteração e esta não é determinada de forma única. As condições de convergência são dadas no teorema abaixo.

**Teorema 2.4.1** *Seja  $\xi$  uma raiz da função  $f$  isolada no intervalo  $[a, b]$ . Seja  $\phi$  uma função de iteração da função  $f$  que satisfaz:*

- 1)  $\phi$  e  $\phi'$  são contínuas em  $[a, b]$ ,
- 2)  $|\phi'(x)| \leq M < 1 \quad \forall x \in [a, b]$ ,
- 3)  $x_0 \in [a, b]$ .

Então a seqüência  $\{x_k\}$  gerada pelo processo iterativo  $x_{n+1} = \phi(x_n)$  converge para  $\xi$ .

Prova: Sendo  $\xi$  uma raiz então  $f(\xi) = 0 \Rightarrow \xi = \phi(\xi)$ , logo

$$x_{n+1} = \phi(x_n) \Rightarrow x_{n+1} - \xi = \phi(x_n) - \phi(\xi).$$

Como  $\phi$  é contínua e diferenciável, pelo Teorema do Valor Médio temos que existe  $c_n$  pertencente ao intervalo entre  $c_n$  e  $\xi$  tal que

$$\phi(x_n) - \phi(\xi) = \phi'(c_n)(x_n - \xi)$$

Logo

$$|x_{n+1} - \xi| = |\phi'(c_n)| |x_n - \xi| \leq M|x_n - \xi|$$

Aplicando esta relação para  $n - 1, n - 2, \dots, 0$  e usando o fato que  $x_0 \in [a, b]$  temos

$$|x_{n+1} - \xi| \leq M^{n+1}|x_0 - \xi|$$

Como  $M < 1$ , aplicando o limite para  $n \rightarrow \infty$  segue que

$$0 \leq \lim_{n \rightarrow \infty} |x_{n+1} - \xi| \leq \lim_{n \rightarrow \infty} M^{n+1}|x_0 - \xi| = 0$$

Logo

$$\lim_{n \rightarrow \infty} x_{n+1} = \xi$$

Observamos que quanto menor for  $|\phi'(x)|$  mais rápida será a convergência. ■

**Exemplo 2.4.1** Consideremos a função  $f(x) = e^{-x} - x$ , onde existe uma raiz  $\xi \in [0.5, 0.75]$ . Uma forma de escrever  $f(x) = x - \phi(x)$  é considerar  $\phi(x) = e^{-x}$ . Verificando as condições de convergência temos:

- 1) As funções  $\phi(x) = e^{-x}$  e  $\phi'(x) = -e^{-x}$  são contínuas em  $[0.5, 0.75]$ .
- 2) A função  $\phi'$  satisfaz

$$\max_{x \in [0.5, 0.75]} |\phi'(x)| = 0.6065... < 1 \quad (\text{Por que? Ver Nota 1})$$

- 3) Tomando  $x_0 \in [0.5, 0.75]$  teremos garantia de convergência, por exemplo podemos tomar  $x_0$  como o ponto médio do intervalo

$$x_0 = \frac{0.5 + 0.75}{2} = 0.625$$

Assim temos que

$$\begin{aligned} x_1 &= \phi(x_0) = \phi(0.625) = 0.53526... \\ x_2 &= \phi(x_1) = \phi(0.53526) = 0.58551... \\ x_3 &= \phi(x_2) = \phi(0.58551) = 0.55681... \\ x_4 &= \phi(x_3) = \phi(0.55681) = 0.57302... \\ x_5 &= \phi(x_4) = \phi(0.57302) = 0.56381... \\ x_6 &= \phi(x_5) = \phi(0.56381) = 0.56903... \\ &\vdots \quad \vdots \quad \vdots \end{aligned}$$

Na Figura 2.4 podemos ver que o comportamento do processo iterativo converge para a raiz.

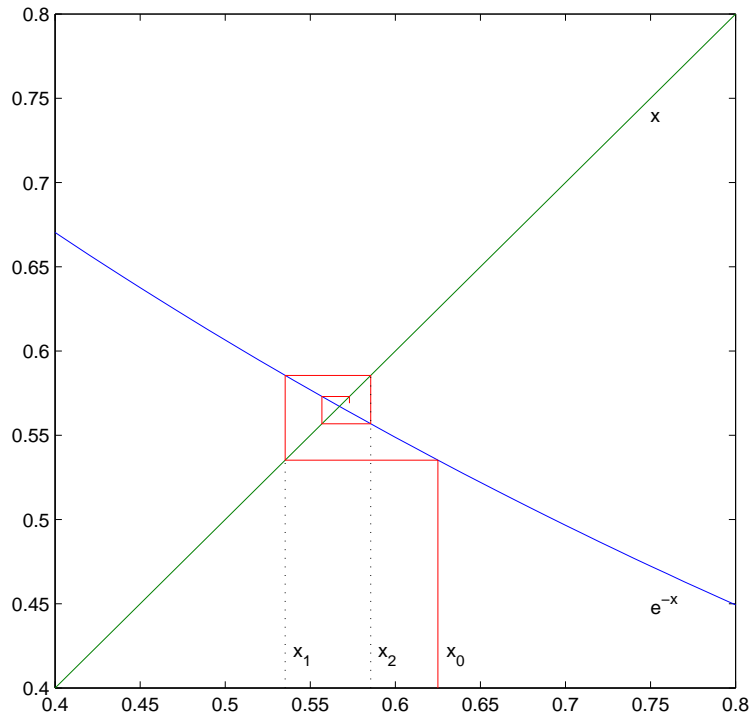


Figura 2.4: Método Iterativo Linear

### 2.4.1 Critério de Parada

Uma questão ainda está em aberto. Qual o  $x_n$  que fornece uma aproximação para a raiz, com uma certa precisão  $\varepsilon$  dada. Neste caso podemos usar como critério de parada as seguintes condições

$$|x_{n+1} - x_n| \leq \varepsilon \quad (\text{Erro Absoluto})$$

$$\frac{|x_{n+1} - x_n|}{|x_{n+1}|} \leq \varepsilon \quad (\text{Erro Relativo})$$

e vamos tomar  $x_{n+1}$  como aproximação para a raiz. Se no exemplo anterior tivéssemos escolhido  $\varepsilon = 0.006$  e o Erro Absoluto teríamos

$$\begin{aligned} |x_1 - x_0| &= |0.53526 - 0.625| = 0.08974 > \varepsilon \\ |x_2 - x_1| &= |0.58551 - 0.53526| = 0.05025 > \varepsilon \\ |x_3 - x_2| &= |0.55681 - 0.58551| = 0.02870 > \varepsilon \\ |x_4 - x_3| &= |0.57302 - 0.55681| = 0.01621 > \varepsilon \\ |x_5 - x_4| &= |0.56381 - 0.57302| = 0.00921 > \varepsilon \\ |x_6 - x_5| &= |0.56903 - 0.56381| = 0.00522 < \varepsilon \end{aligned}$$

Logo a aproximação para a raiz seria  $x_6 = 0.56903$ .

## 2.5 Método de Newton-Raphson (M.N.R)

No método anterior, vimos que quanto menor for  $|\phi'(x)|$  mais rápida será a convergência. O método de Newton-Raphson é determinado de tal forma que teremos uma função de iteração tal que  $\phi'(\xi) = 0$ , onde  $\xi$  é uma raiz de  $f$ . Com isto temos a garantia que existe um intervalo  $[\bar{a}, \bar{b}]$  que contém a raiz e que  $|\phi'(x)| \ll 1$  e conseqüentemente a convergência será mais rápida.

Para determinar a forma de  $\phi$  consideremos uma função  $A(x)$  contínua diferenciável e que  $A(x) \neq 0, \forall x$ . Assim temos

$$f(x) = 0 \Rightarrow A(x)f(x) = 0 \Rightarrow x = x + A(x)f(x) = \phi(x)$$

Calculando a derivada de  $\phi$  na raiz  $\xi$  temos que

$$\phi'(\xi) = 1 + A'(\xi)f(\xi) + A(\xi)f'(\xi) = 0.$$

Como  $f(\xi) = 0$  e considerando que  $f'(\xi) \neq 0$ , segue que

$$A(\xi) = -\frac{1}{f'(\xi)}.$$

Assim tomamos a função  $A(x) = -1/f'(x)$ , e portanto teremos

$$\phi(x) = x - \frac{f(x)}{f'(x)}$$

Com esta função de iteração montamos o processo iterativo conhecido como método de Newton-Raphson, onde dado  $x_0$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 0, 1, 2, \dots$$

Graficamente este método tem a interpretação mostrada na Figura 2.5. A derivada de uma função no ponto  $x_n$  é igual a tangente do ângulo que a reta tangente a curva no ponto  $x_n$  forma com o eixo  $x$ . Usando a relação sobre o triângulo retângulo temos

$$f'(x_n) = \tan(\alpha) = \frac{f(x_n)}{x_n - x_{n+1}} \Rightarrow x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

**Teorema 2.5.1** *Sejam  $f, f'$  e  $f''$ , funções contínuas num intervalo  $[a, b]$ , onde existe uma raiz  $\xi$ . Supor que  $f'(\xi) \neq 0$ . Então existe um intervalo  $[\bar{a}, \bar{b}] \subset [a, b]$ , contendo a raiz  $\xi$ , tal que se  $x_0 \in [\bar{a}, \bar{b}]$ , a seqüência  $\{x_n\}$  gerada pelo processo iterativo*

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

*converge para a raiz.*

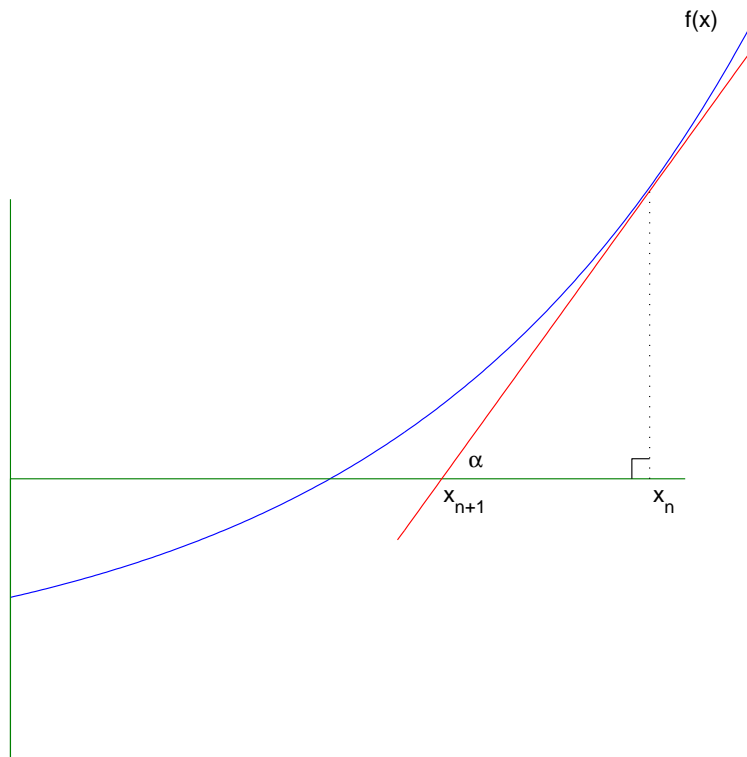


Figura 2.5: Método Newton-Raphson

Prova:(Exercício 2.7)

Uma observação deve ser feita. A condição de que  $x_0 \in [\bar{a}, \bar{b}]$  não é uma condição de fácil verificação, visto que o Teorema garante a existência do intervalo, mas não como determiná-lo. Observamos na Figura 2.6 casos em que o método de Newton-Raphson é falho.

**Exemplo 2.5.1** Considerando  $f(x) = e^{-x} - x$  que possui uma raiz no intervalo  $[0.5, 0.75]$ , vamos achar uma aproximação usando  $x_0 = 0.625$  e  $\varepsilon = 0.006$ . Sendo

$$f'(x) = -e^{-x} - 1$$

teremos o processo iterativo

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n + \frac{e^{-x} - x}{e^{-x} + 1}$$

Assim temos que

$$\begin{aligned} x_1 &= x_0 + \frac{e^{-x_0} - x_0}{e^{-x_0} + 1} = 0.56654 & |x_1 - x_0| &= 0.0584 > \varepsilon \\ x_2 &= x_1 + \frac{e^{-x_1} - x_1}{e^{-x_1} + 1} = 0.56714 & |x_2 - x_1| &= 0.0006 < \varepsilon \end{aligned}$$



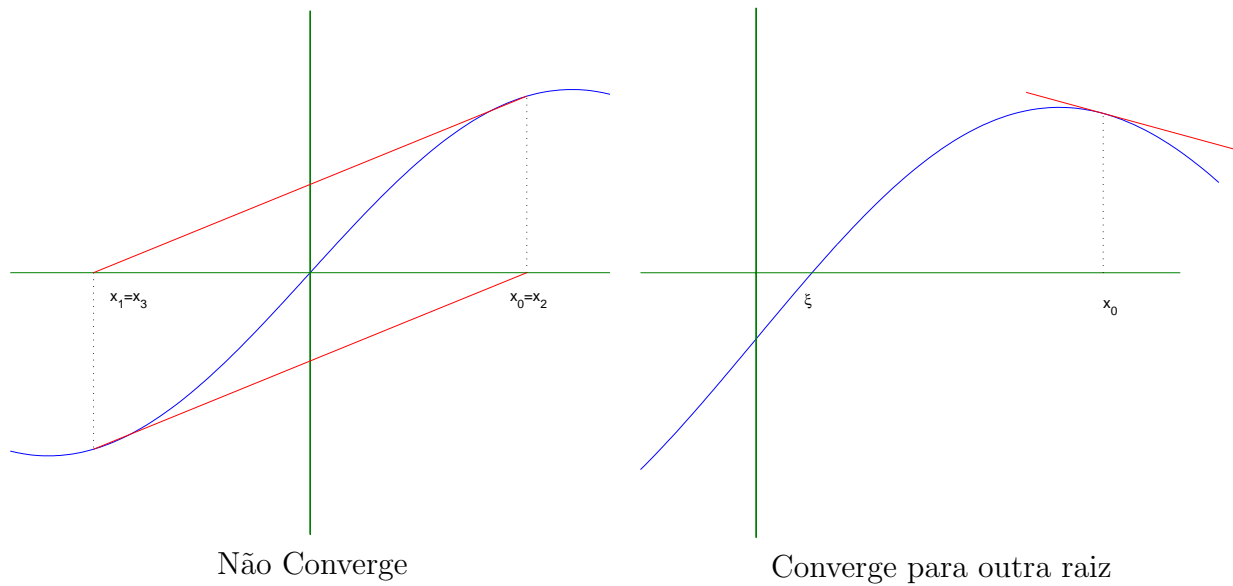


Figura 2.6: Casos em que Método Newton-Raphson é falho

Logo a aproximação é dada por  $x_2 = 0.56714$ .

Abaixo segue a implementação do método como função do MatLab:

```
% Disciplina de Cálculo Numérico - Prof. J. E. Castilho
% Método de Newton-Raphson
% Calcula uma aproximação para uma raiz de função f(x)
% definida no arquivo f.m. A derivada da função f(x) esta
% definida no arquivo df.m, tomamos x0 como condição inicial e
% a predição dada por Ep.
```

```
function x1=newton(x0,Ep)
x1=x0-f(x0)/df(x0)
while abs(x1-x0) > Ep,
    x0=x1;
    x1=x0-f(x0)/df(x0)
end;
```

## 2.6 Ordem de Convergência

Na seção anterior determinamos o Método de Newton-Raphson que pode ser interpretado como um caso particular do Método Iterativo Linear, onde a convergência é mais rápida. A “medida” que permite comparar a convergência entre os métodos é o que chamamos de ordem de convergência, definida por:

**Definição 2.6.1** Seja  $\{x_n\}$  uma seqüência que converge para um número  $\xi$  e seja  $e_k = x_k - \xi$  o erro na iteração  $k$ . Se

$$\lim_{k \rightarrow \infty} \frac{|e_{k+1}|}{|e_k|^p} = C \quad p > 1 \quad C > 0$$

dizemos que a seqüência converge com ordem  $p$  e com constante assintótica  $C$ .

Como a seqüência converge, para valores de  $k$  suficientemente grande temos

$$|e_{k+1}| \approx C|e_k|^p, \quad \text{com } |e_k| < 1$$

Assim quanto maior for o valor de  $p$ , menor será o erro  $|e_{k+1}|$ . Quando  $p = 1$  dizemos que o método têm convergência linear. Se  $p = 2$  dizemos que a convergência é quadrática.

Primeiramente vamos determinar a ordem de convergência do M.I.L. Sendo a seqüência  $\{x_n\}$  gerada por  $x_{k+1} = \phi(x_k)$ ,  $k = 0, 1, 2, \dots$  e que  $\xi = \phi(\xi)$  temos

$$x_{k+1} - \xi = \phi(x_k) - \phi(\xi) = \phi'(c_k)(x_k - \xi),$$

onde a última igualdade é conseqüência do Teorema do Valor Médio e  $c_k$  é um número entre  $x_k$  e  $\xi$ . Logo segue

$$\frac{x_{k+1} - \xi}{x_k - \xi} = \phi'(c_k) \Rightarrow \frac{e_{k+1}}{e_k} = \phi'(c_k)$$

Aplicando o módulo e calculando o limite quando  $k$  tende ao infinito temos

$$\lim_{k \rightarrow \infty} \frac{|e_{k+1}|}{|e_k|} = \lim_{k \rightarrow \infty} |\phi'(c_k)| = |\phi'(\xi)| = C$$

Portanto temos que o M.I.L. têm ordem de convergência  $p = 1$ .

No caso do Método de Newton-Raphson temos que a seqüência é gerada pelo processo iterativo

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Subtraindo  $\xi$  de cada lado temos

$$x_{n+1} - \xi = x_n - \xi - \frac{f(x_n)}{f'(x_n)} \Rightarrow e_{n+1} = e_n - \frac{f(x_n)}{f'(x_n)} \quad (2.2)$$

Através da fórmula de Taylor da função  $f$  no ponto  $x_n$  temos

$$f(x) = f(x_n) + f'(x_n)(x - x_n) + \frac{f''(c_n)}{2}(x - x_n)^2 \quad c_n \in [x, x_n]$$

Que calculada em  $x = \xi$  fornece

$$0 = f(\xi) = f(x_n) + f'(x_n)(\xi - x_n) + \frac{f''(c_n)}{2}(\xi - x_n)^2$$

Dividindo por  $f'(x_n)$  e fazendo  $e_n = x_n - \xi$  segue que

$$\frac{f(x_n)}{f'(x_n)} = e_n - \frac{f''(c_n)}{2f'(x_n)} e_n^2$$

Substituindo em (2.2) obtemos

$$\frac{e_{n+1}}{e_n^2} = \frac{f''(c_n)}{2f'(x_n)}$$

Finalmente aplicamos o módulo e calculamos o limite quando  $k$  tende ao infinito obtendo

$$\lim_{k \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^2} = \lim_{k \rightarrow \infty} \left| \frac{f''(c_n)}{2f'(x_n)} \right| = \left| \frac{f''(\xi)}{2f'(\xi)} \right| = \frac{1}{2} |\phi''(\xi)| = C$$

Portanto temos que o Método de Newton-Raphson têm ordem de convergência  $p = 2$ .

## 2.7 Observações Finais

Neste capítulo vimos três métodos diferentes para resolver equações da forma  $f(x) = 0$ . Faremos um breve comentário das vantagens e desvantagens de cada método.

No Método da bissecção vimos que o número de iterações depende apenas do intervalo inicial  $[a_0, b_0]$ . Logo este pode ser aplicado a qualquer função  $f(x)$  que satisfaz  $f(a)f(b) < 0$ . Não importa o quanto  $f(x)$  seja complicada. A desvantagem é que tem uma convergência lenta. Na prática ele é usado para refinar o intervalo que contém a raiz. Aplicamos o método em um número fixo de iterações.

Em geral o M.I.L. é mais rápido que o Método da Bissecção. Usa menos operações por cada iteração. Pode encontrar raízes em intervalos onde  $f(a)f(b) > 0$ . A dificuldade é encontrar a função de iteração  $\phi$  que seja convergente.

O Método de Newton-Raphson têm convergência quadrática. Porém este necessita da avaliação da função e sua derivada em cada ponto  $x_n$ . Pode ocorrer de termos uma raiz isolada num intervalo  $[a, b]$  e o método acabe convergindo para uma outra raiz que não pertence a  $[a, b]$ . Isto ocorre porque temos que tomar  $x_0 \in [\bar{a}, \bar{b}] \subset [a, b]$ . Na prática tomamos  $x_0$  como ponto médio do intervalo, isto é

$$x_0 = \frac{a + b}{2}$$

**Nota 1** Em muitas situações vamos necessitar de calcular o máximo do módulo de uma função restrita a um intervalo, isto é

$$\max_{x \in [a, b]} |f(x)|.$$

Uma forma prática para este cálculo é seguir os passos:

1: Calcula-se os valores da função nos extremos do intervalo,  $|f(a)|$  e  $|f(b)|$ .

2: Verifique se a função não possui ponto crítico no intervalo, ou seja, achamos os valores de  $x_k$  tal que  $f'(x_k) = 0$  e  $x_k \in [a, b]$

3: Tomamos como o valor máximo o  $\max\{|f(a)|, |f(b)|, |f(x_k)|\}$

## 2.8 Exercícios

**Exercício 2.1** Localize graficamente e dê intervalos de amplitude 0.5 que contenha as raízes das equações

a)  $\ln(x) + 2x = 0$     b)  $e^x - \sin(x) = 0$     c)  $\ln(x) - 2^x = -2$   
 d)  $2\cos(x) - \frac{e^x}{2} = 0$     e)  $3\ln(x) - \frac{x^2}{2}$     f)  $(5-x)e^x = 1$

**Exercício 2.2** Utilize o Método da Bissecção e aproxime a menor raiz em módulo com erro relativo menor que  $10^{-1}$  para as equações a) e b) do exercício anterior.

**Exercício 2.3** Utilize o Método Iterativo Linear e aproxime a menor raiz em módulo com erro relativo menor que  $10^{-2}$  para as equações c) e d) do exercício anterior.

**Exercício 2.4** Utilize o Método de Newton-Raphson e aproxime a menor raiz em módulo com erro relativo menor que  $10^{-3}$  para as equações d) e f) do exercício anterior.

**Exercício 2.5** Achar a raiz  $p$ -ésima de um número positivo  $a$  é equivalente a achar a raiz positiva da equação  $\sqrt[p]{a} = x$ .

- Encontre um intervalo que depende do valor de  $a$  e que contenha a raiz.
- Verifique se a função de iteração  $\phi(x) = a/x^{p-1}$  satisfaz os critérios de convergência do Método Iterativo Linear.
- Verifique que o processo iterativo gerado pelo M.N.R. é dado por

$$x_{n+1} = \frac{1}{p} \left[ (p-1)x_n + \frac{a}{x_n^{p-1}} \right]$$

- Implemente um programa em Matlab que execute o processo iterativo dado em c).

**Exercício 2.6** Dada a função  $f(x) = e^x - 4x^2$ .

- Isole as raízes da função  $f(x)$ .
- Verifique que as funções abaixo são função de iteração de  $f$  e verifique se satisfazem o critério de convergência do M.I.L. para a raiz positiva.

$$\phi_1(x) = \frac{1}{2}e^{x/2} \quad \phi_2(x) = \ln(4x^2)$$

c) Tomando  $x_0 = 0.6$  e  $\varepsilon = 0.01$ , aplique o M.I.L. para encontrar uma aproximação para a raiz positiva, usando uma função de iteração que satisfaça os critérios de convergência

**Exercício 2.7** Prove o Teorema 2.5.1.

**Exercício 2.8** A função  $f(x) = \text{sen}(\cos(\sqrt{3x}))$  tem uma raiz no intervalo  $[0.7, 0.9]$ . Encontre uma aproximação com  $\varepsilon = 0.07$ , escolhendo entre os métodos numéricos estudados o mais adequado. Justifique sua resposta.

# Capítulo 3

## Sistemas Lineares

A resolução de sistemas lineares surge em diversas áreas do conhecimento. O caso geral em que o sistema linear envolve  $m$  equações com  $n$  incógnitas, o sistema pode apresentar uma única solução, infinitas soluções ou não admitir solução. Este tipo de problema é tratado na Álgebra Linear usando o processo de escalonamento. Neste capítulo vamos analisar esquemas numéricos para soluções de sistemas lineares de  $n$  equações com  $n$  incógnitas, isto é

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + \cdots + a_{1,n}x_n = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + a_{2,3}x_3 + \cdots + a_{2,n}x_n = b_2 \\ a_{3,1}x_1 + a_{3,2}x_2 + a_{3,3}x_3 + \cdots + a_{3,n}x_n = b_3 \\ \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + a_{n,3}x_3 + \cdots + a_{n,n}x_n = b_n \end{cases}$$

onde  $a_{ij}$  são os coeficientes,  $x_j$  são as incógnitas e os  $b_j$  são as constantes. Este sistema pode ser escrito na forma matricial  $\mathbf{Ax} = \mathbf{b}$  com  $\mathbf{A} \in \mathbb{R}^{n \times n}$  e  $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ . Analisaremos duas classes de esquemas numéricos: Métodos Diretos e Métodos Iterativos.

### 3.1 Métodos Diretos

Os Métodos Diretos são aqueles que após um número finito de operações fornecem a solução exata do sistema, a menos dos erros de arredondamentos. Estes métodos são baseados no processo de escalonamento estudado em Álgebra Linear. São eficientes para sistemas de pequeno porte (não mais que 50 equações) e para sistemas de bandas, como por exemplo sistemas tridiagonais (ver Ex. 3.3). Primeiramente vamos considerar os sistemas lineares triangulares.



### 3.1.2 Método de Eliminação de Gauss

Dois sistemas lineares são ditos ser equivalentes se estes tem a mesma solução. A estratégia do Método de Eliminação de Gauss é transformar um sistema linear  $\mathbf{Ax} = \mathbf{b}$  em um sistema triangular superior equivalente  $\mathbf{Sx} = \tilde{\mathbf{b}}$ , cuja a solução é facilmente obtida pela Retro-Solução. Esta transformação é realizada através das operações elementares

- (I) Trocar duas equações.
- (II) Multiplicar uma equação por uma constante não nula.
- (III) Adicionar a uma equação uma outra multiplicada por uma constante não nula.

Aplicando qualquer seqüência dessas operações elementares num sistema  $\mathbf{Ax} = \mathbf{b}$  obtemos um novo sistema  $\tilde{\mathbf{Ax}} = \tilde{\mathbf{b}}$  de tal forma que estes serão equivalentes. Para descrever o Método de Eliminação de Gauss vamos considerar o sistema linear

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + \cdots + a_{1,n}x_n = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + a_{2,3}x_3 + \cdots + a_{2,n}x_n = b_2 \\ a_{3,1}x_1 + a_{3,2}x_2 + a_{3,3}x_3 + \cdots + a_{3,n}x_n = b_3 \\ \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + a_{n,3}x_3 + \cdots + a_{n,n}x_n = b_n \end{cases},$$

onde  $\det(A) \neq 0$ , isto é, o sistema admite uma única solução. Um sistema linear pode ser representado na forma de matriz estendida  $(\mathbf{A}^0 | \mathbf{b}^0)$ , ou seja

$$\left( \begin{array}{cccc|c} a_{1,1}^{(0)} & a_{1,2}^{(0)} & a_{1,3}^{(0)} & \cdots & a_{1,n}^{(0)} & b_1^{(0)} \\ a_{2,1}^{(0)} & a_{2,2}^{(0)} & a_{2,3}^{(0)} & \cdots & a_{2,n}^{(0)} & b_2^{(0)} \\ a_{3,1}^{(0)} & a_{3,2}^{(0)} & a_{3,3}^{(0)} & \cdots & a_{3,n}^{(0)} & b_3^{(0)} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ a_{n,1}^{(0)} & a_{n,2}^{(0)} & a_{n,3}^{(0)} & \cdots & a_{n,n}^{(0)} & b_n^{(0)} \end{array} \right)$$

onde o índice superior indica a etapa do processo.

**Etapa 1:** Eliminar a incógnita  $x_1$  das equações  $k = 2, 3, \dots, n$ . Sendo  $a_{1,1}^{(0)} \neq 0$ , usaremos a operação elementar (III) e subtraímos da linha  $k$  a primeira linha multiplicada por

$$m_{k,1} = \frac{a_{k,1}^{(0)}}{a_{1,1}^{(0)}}.$$

Os elementos  $m_{k,1}$  são chamados de multiplicadores e o elemento  $a_{1,1}^{(0)}$  é chamado de pivô da Etapa 1. Indicando a linha  $k$  da matriz entendida por  $L_k^{(0)}$  esta etapa se resume em

$$\begin{aligned} L_1^{(1)} &= L_1^{(0)} \\ L_k^{(1)} &= L_k^{(0)} - m_{k,1}L_1^{(0)}, \quad k = 2, 3, \dots, n \end{aligned}$$



Ao final desta etapa teremos a matriz

$$\left( \begin{array}{cccc|c} a_{1,1}^{(1)} & a_{1,2}^{(1)} & a_{1,3}^{(1)} & \cdots & a_{1,n}^{(1)} & b_1^{(1)} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} & b_2^{(1)} \\ 0 & a_{3,2}^{(1)} & a_{3,3}^{(1)} & \cdots & a_{3,n}^{(1)} & b_3^{(1)} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & a_{n,2}^{(1)} & a_{n,3}^{(1)} & \cdots & a_{n,n}^{(1)} & b_n^{(1)} \end{array} \right)$$

que representa um sistema linear equivalente ao sistema original, onde a incógnita  $x_1$  foi eliminada das equações  $k = 2, 3, \dots, n$ .

**Etapa 2:** Eliminar a incógnita  $x_2$  das equações  $k = 3, 4, \dots, n$ . Supondo que  $a_{2,2}^{(1)} \neq 0$ , vamos tomar este elemento como pivô desta etapa e desta forma os multiplicadores são dados por

$$m_{k,2} = \frac{a_{k,2}^{(1)}}{a_{2,2}^{(1)}}$$

A eliminação se procede da seguinte forma:

$$\begin{aligned} L_1^{(2)} &= L_1^{(1)} \\ L_2^{(2)} &= L_2^{(1)} \\ L_k^{(2)} &= L_k^{(1)} - m_{k,2}L_2^{(1)}, \quad k = 3, 4, \dots, n \end{aligned}$$

obtendo ao final da etapa a matriz

$$\left( \begin{array}{cccc|c} a_{1,1}^{(2)} & a_{1,2}^{(2)} & a_{1,3}^{(2)} & \cdots & a_{1,n}^{(2)} & b_1^{(2)} \\ 0 & a_{2,2}^{(2)} & a_{2,3}^{(2)} & \cdots & a_{2,n}^{(2)} & b_2^{(2)} \\ 0 & 0 & a_{3,3}^{(2)} & \cdots & a_{3,n}^{(2)} & b_3^{(2)} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & a_{n,3}^{(2)} & \cdots & a_{n,n}^{(2)} & b_n^{(2)} \end{array} \right)$$

Com procedimentos análogos ao das etapas 1 e 2 podemos eliminar as incógnitas  $x_k$  das equações  $k + 1, k + 2, \dots, n$  e ao final de  $n - 1$  etapas teremos a matriz

$$\left( \begin{array}{cccc|c} a_{1,1}^{(n-1)} & a_{1,2}^{(n-1)} & a_{1,3}^{(n-1)} & \cdots & a_{1,n}^{(n-1)} & b_1^{(n-1)} \\ 0 & a_{2,2}^{(n-1)} & a_{2,3}^{(n-1)} & \cdots & a_{2,n}^{(n-1)} & b_2^{(n-1)} \\ 0 & 0 & a_{3,3}^{(n-1)} & \cdots & a_{3,n}^{(n-1)} & b_3^{(n-1)} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{n,n}^{(n-1)} & b_n^{(n-1)} \end{array} \right)$$

Esta matriz representa um sistema triangular superior equivalente ao sistema original. Logo a solução deste sistema, obtido pela Retro-Solução, é solução do sistema original.

**Algoritmo:** Método de Eliminação de Gauss

**Input:** Matriz  $\mathbf{A}$  e vetor  $\mathbf{b} \in \mathbb{R}^n$

**Eliminação:**

Para  $k = 1, 2, \dots, n - 1$ , faça:

Para  $i = k + 1, \dots, n$ , faça:

$$m \leftarrow \frac{a_{ij}}{a_{k,k}}$$

Para  $j = k + 1, \dots, n$ , faça:

$$a_{ij} \leftarrow a_{ij} - m * a_{kj}$$

fim para

$$b_i \leftarrow b_i - m * b_k$$

fim para

fim para

**Retro-Solução:**

$$x_n \leftarrow b_n / a_{n,n}$$

Para  $k = n - 1, n - 2, \dots, 1$ , faça:

$$x_k \leftarrow \frac{1}{a_{k,k}} \left( b_k - \sum_{j=k+1}^n a_{k,j} x_j \right)$$

fim para

**Output:**  $\mathbf{x} \in \mathbb{R}^n$  : solução do sistema

**Exemplo 3.1.1** Vamos considerar o sistema linear abaixo

$$\begin{cases} 3x_1 + 2x_2 - x_3 = 1 \\ 7x_1 - x_2 - x_3 = -2 \\ x_1 + x_3 = 1 \end{cases}$$

Escrevendo na forma de matriz estendida teremos

$$\left( \begin{array}{ccc|c} 3 & 2 & -1 & 1 \\ 7 & -1 & -1 & -2 \\ 1 & 0 & 1 & 1 \end{array} \right)$$

**Etapa 1:** Eliminar  $x_1$  das linhas 2 e 3.

$$L_1^{(1)} = L_1^{(0)}$$

$$L_2^{(1)} = L_2^{(0)} - m_{2,1} L_1^{(0)}, \quad \text{onde } m_{2,1} = \frac{a_{21}^{(0)}}{a_{1,1}^{(0)}} = \frac{7}{3}$$

$$L_3^{(1)} = L_3^{(0)} - m_{3,1} L_1^{(0)}, \quad \text{onde } m_{3,1} = \frac{a_{31}^{(0)}}{a_{1,1}^{(0)}} = \frac{1}{3}$$

e com isto obtemos a matriz

$$\left( \begin{array}{ccc|c} 3 & 2 & -1 & 1 \\ 0 & -17/3 & 4/3 & -13/3 \\ 0 & -2/3 & 4/3 & 12/3 \end{array} \right)$$

**Etapa 2:** Eliminar  $x_2$  da linha 3.

$$L_1^{(2)} = L_1^{(1)}$$

$$L_2^{(2)} = L_2^{(1)}$$

$$L_3^{(2)} = L_3^{(1)} - m_{3,2}L_2^{(1)}, \quad \text{onde } m_{3,2} = \frac{a_{32}^{(01)}}{a_{2,2}^{(1)}} = \frac{2}{17}$$

obtendo assim a matriz

$$\left( \begin{array}{ccc|c} 3 & 2 & -1 & 1 \\ 0 & -17/3 & 4/3 & -13/3 \\ 0 & 0 & 20/17 & 20/17 \end{array} \right)$$

**Retro-Solução:** Encontrar a solução do sistema triangular superior.

$$\begin{aligned} x_3 &= \frac{b_3}{a_{3,3}} = 1 \\ x_2 &= \frac{1}{a_{2,2}}(b_2 - a_{2,3}x_3) = 1 \\ x_1 &= \frac{1}{a_{1,1}}(b_1 - a_{1,2}x_2 - a_{1,3}x_3) = 0 \end{aligned}$$

Logo a solução do sistema é dada por  $x = (0, 1, 1)^T$ .

A solução encontrada é a solução exata, pois mantivemos os números resultantes na forma de fração. Porém máquinas digitais representam estes números na forma de ponto flutuante finita e erros de arredondamento podem ocorrer. Em sistemas lineares de grande porte estes erros vão se acumulando e prejudicando a solução do sistema.

### 3.1.3 Pivotamento Parcial

Em cada etapa  $k$  da eliminação temos o cálculo do multiplicador

$$m_{k,j} = \frac{a_{k,j}^{(k-1)}}{a_{k,k}^{(k-1)}}.$$

Se o pivô  $|a_{k,k}^{(k-1)}| \ll 1$ , ou seja este é próximo de zero teremos problemas com os erros de arredondamento, pois operar números de grandezas muito diferentes aumenta os erros ( ver Ex. 3.4). A estratégia de pivotamento parcial é baseada na operação elementar (I). No início de cada etapa  $k$  escolhemos como pivô o elemento de maior módulo entre os coeficientes  $a_{ik}^{k-1}$  para  $i = k, k+1, \dots, n$ .

**Exemplo 3.1.2** Vamos considerar o sistema linear, representado pela matriz estendida

$$\left( \begin{array}{ccc|c} 1 & 3 & 2 & 3 \\ 2 & 1 & 2 & 4 \\ -3 & 2 & 1 & -2 \end{array} \right)$$

**Etapa 1:** Escolha do pivô

$$\max_{1 \leq i \leq 3} |a_{i,1}| = |a_{3,1}|$$

Trocamos a linha  $L_1$  com a linha  $L_3$ , obtendo,

$$\left( \begin{array}{ccc|c} -3 & 2 & 1 & -2 \\ 2 & 1 & 2 & 4 \\ 1 & 3 & 2 & 3 \end{array} \right)$$

Eliminar  $x_1$  das linhas 2 e 3.

$$L_1^{(1)} = L_1^{(0)}$$

$$L_2^{(1)} = L_2^{(0)} - m_{2,1}L_1^{(0)}, \quad \text{onde } m_{2,1} = \frac{2}{3}$$

$$L_3^{(1)} = L_3^{(0)} - m_{3,1}L_1^{(0)}, \quad \text{onde } m_{3,1} = \frac{1}{3}$$

e com isto obtemos a matriz

$$\left( \begin{array}{ccc|c} -3 & 2 & 1 & -2 \\ 0 & 7/3 & 8/3 & 8/3 \\ 0 & 11/3 & 7/3 & 7/3 \end{array} \right)$$

**Etapa 2:** Escolha do pivô

$$\max_{2 \leq i \leq 3} |a_{i,1}| = |a_{3,2}|$$

Trocamos a linha  $L_2$  com a linha  $L_3$ , obtendo,

$$\left( \begin{array}{ccc|c} -3 & 2 & 1 & -2 \\ 0 & 11/3 & 7/3 & 7/3 \\ 0 & 7/3 & 8/3 & 8/3 \end{array} \right)$$

Eliminar  $x_2$  da linha 3.

$$L_1^{(2)} = L_1^{(1)}$$

$$L_2^{(2)} = L_2^{(1)}$$

$$L_3^{(2)} = L_3^{(1)} - m_{3,2}L_2^{(1)}, \quad \text{onde } m_{3,2} = \frac{-7}{11}$$

obtendo assim a matriz

$$\left( \begin{array}{ccc|c} -3 & 2 & 1 & -2 \\ 0 & 11/3 & 8/3 & 8/3 \\ 0 & 0 & 13/11 & 13/11 \end{array} \right)$$

**Retro-Solução:** *Encontrar a solução do sistema triangular superior.*

$$\begin{aligned}x_3 &= \frac{b_3}{a_{3,3}} = 1 \\x_2 &= \frac{1}{a_{2,2}}(b_2 - a_{2,3}x_3) = 0 \\x_1 &= \frac{1}{a_{1,1}}(b_1 - a_{1,2}x_2 - a_{1,3}x_3) = 1\end{aligned}$$

Logo a solução do sistema é dada por  $x = (1, 0, 1)^T$ .

Na prática a troca de linhas não é realizada. O controle é feito por um vetor de inteiros  $n$ -dimensional, onde inicialmente na posição  $k$  está armazenado  $k$ , ou seja  $trc = [1, 2, \dots, s, \dots, n]$ . Se, por exemplo, trocamos a linha 1 pela linha  $s$  o vetor passa a ser  $trc = [s, 2, \dots, 1, \dots, n]$ .

### 3.1.4 Cálculo da Matriz Inversa

Vamos supor que desejamos resolver os sistemas lineares  $\mathbf{Ax} = \mathbf{b}^1, \mathbf{Ax} = \mathbf{b}^2, \dots, \mathbf{Ax} = \mathbf{b}^k$ , onde a matriz  $\mathbf{A}$  é a mesma para todos os sistemas. A matriz triangular superior, resultante do processo de eliminação, não depende do vetor  $\mathbf{b}$  e portanto será a mesma em qualquer um dos sistemas. Assim podemos resolver estes sistemas num único processo de eliminação usando a matriz estendida  $(\mathbf{A}|\mathbf{b}^1|\mathbf{b}^2|\dots|\mathbf{b}^k)$  e aplicando a Retro-Solução para cada vetor  $\mathbf{b}^k$ .

O Cálculo da inversa de uma matriz é um caso particular do esquema acima. A inversa de uma matriz  $\mathbf{A} \in R^{n \times n}$ , denotada por  $\mathbf{A}^{-1}$ , é uma matriz  $n \times n$  tal que

$$\mathbf{A} \mathbf{A}^{-1} = \mathbf{A}^{-1} \mathbf{A} = \mathbf{I}$$

Como exemplo vamos considerar uma matriz  $\mathbf{A}$  de dimensão  $3 \times 3$

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}$$

cuja a inversa  $\mathbf{A}^{-1}$  é dada por

$$\begin{pmatrix} x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,1} & x_{3,2} & x_{3,3} \end{pmatrix},$$

logo temos que

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,1} & x_{3,2} & x_{3,3} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

Portanto cada coluna  $k$  da inversa da matriz  $\mathbf{A}$  é solução de um sistema linear, onde o vetor dos termos independentes é a  $k$ -ésima coluna da matriz identidade, isto é

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} x_{1,1} \\ x_{2,1} \\ x_{3,1} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix},$$

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} x_{1,2} \\ x_{2,2} \\ x_{3,2} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix},$$

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} x_{1,3} \\ x_{2,3} \\ x_{3,3} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

Em resumo, se temos uma matriz  $n \times n$ , podemos achar a inversa resolvendo  $n$  sistemas lineares, representados pela matriz estendida  $(\mathbf{A}|\mathbf{b}^1|\mathbf{b}^2|\dots|\mathbf{b}^n)$ , onde os vetores  $\mathbf{b}^k$  são os vetores unitários (1 na posição  $k$  e zeros nas demais posições).

**Exemplo 3.1.3** Vamos achar a inversa da matriz abaixo, usando o método de Eliminação de Gauss.

$$\begin{pmatrix} 4 & 1 & -6 \\ 3 & 2 & -6 \\ 3 & 1 & -5 \end{pmatrix},$$

Para o processo de eliminação consideremos a matriz estendida

$$\left( \begin{array}{ccc|ccc} 4 & 1 & -6 & 1 & 0 & 0 \\ 3 & 2 & -6 & 0 & 1 & 0 \\ 3 & 1 & -5 & 0 & 0 & 1 \end{array} \right),$$

**Etapa 1:** Eliminar  $x_1$  das linhas 2 e 3.

$$L_1^{(1)} = L_1^{(0)}$$

$$L_2^{(1)} = L_2^{(0)} - m_{2,1}L_1^{(0)}, \quad \text{onde } m_{2,1} = \frac{3}{4}$$

$$L_3^{(1)} = L_3^{(0)} - m_{3,1}L_1^{(0)}, \quad \text{onde } m_{3,1} = \frac{3}{4}$$

e com isto obtemos a matriz

$$\left( \begin{array}{ccc|ccc} 4 & 1 & -6 & 1 & 0 & 0 \\ 0 & 5/4 & -3/2 & -3/4 & 1 & 0 \\ 0 & 1/4 & -1/2 & -3/4 & 0 & 1 \end{array} \right),$$

**Etapa 2:** Eliminar  $x_2$  da linha 3.

$$\begin{aligned} L_1^{(2)} &= L_1^{(1)} \\ L_2^{(2)} &= L_2^{(1)} \\ L_3^{(2)} &= L_3^{(1)} - m_{3,2}L_2^{(1)}, \quad \text{onde } m_{3,2} = \frac{1}{5} \end{aligned}$$

obtendo assim a matriz

$$\left( \begin{array}{ccc|ccc} 4 & 1 & -6 & 1 & 0 & 0 \\ 0 & 5/4 & -3/2 & -3/4 & 1 & 0 \\ 0 & 0 & -1/5 & -3/5 & -1/5 & 1 \end{array} \right),$$

**Retro-Solução:** Encontrar a solução dos sistemas triangulares superior.

Primeira coluna da inversa

$$\begin{aligned} x_{3,1} &= \frac{b_3^1}{a_{3,3}} = 3 \\ x_{2,1} &= \frac{1}{a_{2,2}}(b_2^1 - a_{2,3}x_3) = 3 \\ x_{1,1} &= \frac{1}{a_{1,1}}(b_1^1 - a_{1,2}x_2 - a_{1,3}x_3) = 4 \end{aligned}$$

Segunda coluna da inversa

$$\begin{aligned} x_{3,2} &= \frac{b_3^2}{a_{3,3}} = 1 \\ x_{2,2} &= \frac{1}{a_{2,2}}(b_2^2 - a_{2,3}x_3) = 2 \\ x_{1,2} &= \frac{1}{a_{1,1}}(b_1^2 - a_{1,2}x_2 - a_{1,3}x_3) = 1 \end{aligned}$$

Terceira coluna da inversa

$$\begin{aligned} x_{3,3} &= \frac{b_3^3}{a_{3,3}} = -5 \\ x_{2,3} &= \frac{1}{a_{2,2}}(b_2^3 - a_{2,3}x_3) = -6 \\ x_{1,3} &= \frac{1}{a_{1,1}}(b_1^3 - a_{1,2}x_2 - a_{1,3}x_3) = -6 \end{aligned}$$

Logo a matriz inversa  $s$  é dada por

$$\begin{pmatrix} 4 & 1 & -6 \\ 3 & 2 & -6 \\ 3 & 1 & -5 \end{pmatrix},$$

No exemplo acima temos que a inversa da matriz  $\mathbf{A}$  é a própria  $\mathbf{A}$ . Este tipo de matriz é usado como matriz teste para verificar a eficiência dos métodos numéricos.

Abaixo apresentamos uma implementação do Método de Eliminação de Gauss em MatLab que resolve  $k$  sistemas, onde a matriz  $\mathbf{A}$  é comum a todos.

```
% Disciplina de Cálculo Numérico - Prof. J. E. Castilho
% Método de eliminação de Gauss -
% Este procedimento resolve k-sistemas lineares, onde a
% matriz A de dimensão n x n é comum a todos.
% Os parâmetros devem ser passados da forma
%   x=EGauss(A,b1,b2,b3,...,bk)
% o resultado é uma matriz x de dimensão n x k onde a
% coluna j armazena a solução do sistema Ax=bj
% Dados A: matriz do sistema
%   varargin: lista dos vetores dos termos independentes
```

```
function x=EGauss(A,varargin)
b=[varargin{:}];
db=size(b);
% Esta parte verifica se o sistema é quadrado
da=size(A);
n=da(1);
if n ~= da(2),
disp('??? A matriz deve ser quadrada');
break;
end;

% Esta parte verifica se a dimensão do vetor b
% está de acordo com a dimensão do sistema
if n ~= db(1), disp('??? Erro na dimensão de b'); break; end;
% Cria matriz estendida
Ax=[A,b];
% Fase da eliminação
for k=1:(n-1)
    for i=k+1:n
        if abs(Ax(k,k)) < 10^(-16),
            disp('??? Pivô Numericamente Nulo');
            break;
        end;
        m=Ax(i,k)/Ax(k,k);
        for j=k+1:da(2) + db(2)
            Ax(i,j) = Ax(i,j)-m*Ax(k,j);
        end;
    end;
end;
```



```

    end;
end;

% Fase da Retro solu\c{c}\~{a}o

if abs(Ax(n,n)) < 10^(-16),
    disp('??? Piv\^{o} Numericamente Nulo');
    break;
end;
for m=1:db(2)
    x(n,m) = Ax(n,n+m)/Ax(n,n);
end;
for k=(n-1):-1:1
    for m=1:db(2)
        som=0;
        for j=k+1:n
            som=som+Ax(k,j)*x(j,m);
        end;
        x(k,m) = (Ax(k,n+m)-som)/Ax(k,k);
    end;
end;
end;

```

## 3.2 Métodos Iterativos

Vamos considerar um sistema linear  $\mathbf{Ax} = \mathbf{b}$ , onde  $\mathbf{A} \in \mathbb{R}^{n \times n}$  e  $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ . Os métodos iterativos seguem um esquema semelhante aos métodos para o cálculo de zeros de funções. O sistema linear é escrito na forma

$$\mathbf{x} = \mathbf{Cx} + \mathbf{g},$$

onde  $\mathbf{g} \in \mathbb{R}^n$  e  $\mathbf{C} \in \mathbb{R}^{n \times n}$  é chamada de matriz de iteração. Assim montamos o processo iterativo:

Dado  $\mathbf{x}^0$

$$\mathbf{x}^{k+1} = \mathbf{Cx}^k + \mathbf{g}.$$

Sendo um processo iterativo, necessitamos de um critério de parada. E para isto temos que ter uma medida entre as aproximações  $\mathbf{x}^{k+1}$  e  $\mathbf{x}^k$ . Para isto vamos usar o conceito de norma de matrizes, definida abaixo

**Definição 3.2.1** *Uma norma em  $\mathbb{R}^{n \times m}$  é uma aplicação  $\|\cdot\| : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}$  que satisfaz as seguintes propriedades:*

$$(M1) \quad \|\mathbf{A}\| \geq 0 \text{ e } \|\mathbf{A}\| = 0 \Leftrightarrow \mathbf{A} = \mathbf{0}, \forall \mathbf{A} \in \mathbb{R}^{n \times m}.$$

$$(M2) \quad \|\alpha \mathbf{A}\| = |\alpha| \|\mathbf{A}\|, \quad \forall \alpha \in \mathbb{R} \text{ e } \forall \mathbf{A} \in \mathbb{R}^{n \times m}.$$

$$(M3) \quad \|\mathbf{A} + \mathbf{B}\| \leq \|\mathbf{A}\| + \|\mathbf{B}\|, \quad \forall \mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times m}.$$

As normas matriciais mais usadas são

$$\|\mathbf{A}\|_1 = \max_{1 \leq j \leq m} \left\{ \sum_{i=1}^n |a_{ij}| \right\} \quad \text{Norma do Máximo das Coluna}$$

$$\|\mathbf{A}\|_\infty = \max_{1 \leq i \leq n} \left\{ \sum_{j=1}^m |a_{ij}| \right\} \quad \text{Norma do Máximo das Linha}$$

$$\|\mathbf{A}\|_2 = \left( \sum_{i=1}^n \sum_{j=1}^m |a_{ij}|^2 \right)^{1/2} \quad \text{Norma Euclidiana}$$

A norma vetorial pode ser vista como um caso particular da norma matricial, onde um vetor  $x \in \mathbb{R}^n$  é equivalente a uma matriz de ordem  $n \times 1$ . Com isto temos as normas de vetores dadas por

$$\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i| \quad \text{Norma da Soma}$$

$$\|\mathbf{x}\|_\infty = \max_{1 \leq i \leq n} |x_i| \quad \text{Norma do Máximo}$$

$$\|\mathbf{x}\|_2 = \left( \sum_{i=1}^n |x_i|^2 \right)^{1/2} \quad \text{Norma Euclidiana}$$

O conceito de norma nos permite definir convergência de uma seqüência de vetores  $\{\mathbf{x}^k\}$ . Dizemos que  $\mathbf{x}^k \rightarrow \bar{\mathbf{x}}$  se

$$\lim_{k \rightarrow \infty} \|\mathbf{x}^k - \bar{\mathbf{x}}\| = 0$$

Com isto podemos definir os critérios de parada: Dado um  $\varepsilon > 0$

$$\|\mathbf{x}^{k+1} - \mathbf{x}^k\| \leq \varepsilon \quad \text{Erro Absoluto}$$

$$\frac{\|\mathbf{x}^{k+1} - \mathbf{x}^k\|}{\|\mathbf{x}^k\|} \leq \varepsilon \quad \text{Erro Relativo}$$

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}^k\| \leq \varepsilon \quad \text{Teste do Resíduo}$$

Além disso, as normas  $\|\cdot\|_1$ ,  $\|\cdot\|_2$  e  $\|\cdot\|_\infty$  satisfazem as seguintes propriedades:

$$(M4) \quad \|\mathbf{A}\mathbf{x}\| \leq \|\mathbf{A}\| \|\mathbf{x}\|$$

$$(M5) \quad \|\mathbf{A}\mathbf{B}\| \leq \|\mathbf{A}\| \|\mathbf{B}\|$$

### 3.2.1 Critério de Convergência

Dependendo da forma da matriz  $\mathbf{C}$  a seqüência gerada pelo processo iterativo pode ou não convergir para a solução do sistema. Seja  $\bar{\mathbf{x}}$  a solução do sistema  $\mathbf{Ax} = \mathbf{b}$ , logo  $\bar{\mathbf{x}}$  satisfaz

$$\bar{\mathbf{x}} = \mathbf{C}\bar{\mathbf{x}} + \mathbf{g}.$$

Com isto temos que

$$\mathbf{x}^{k+1} - \bar{\mathbf{x}} = \mathbf{C}(\mathbf{x}^{k+1} - \bar{\mathbf{x}})$$

Sendo o erro em cada iteração dado por  $\mathbf{e}^k = \mathbf{x}^k - \bar{\mathbf{x}}$  e usando as propriedades de norma (M4) e (M5) segue que

$$\begin{aligned} \|\mathbf{e}^k\| &\leq \|\mathbf{C}\| \|\mathbf{e}^{k-1}\| \\ &\leq \|\mathbf{C}\|^2 \|\mathbf{e}^{k-2}\| \\ &\vdots \\ &\leq \|\mathbf{C}\|^k \|\mathbf{e}^0\| \end{aligned}$$

Logo a seqüência  $\{\mathbf{x}^k\}$  converge para a solução do sistema  $\bar{\mathbf{x}}$  se

$$\lim_{k \rightarrow \infty} \|\mathbf{e}^k\| = \lim_{k \rightarrow \infty} \|\mathbf{C}\|^k \|\mathbf{e}^0\| = 0$$

e isto ocorre se e somente se a matriz  $\mathbf{C}$  satisfaz a condição

$$\|\mathbf{C}\| < 1$$

Note que o critério de convergência não depende do vetor inicial  $\mathbf{x}^0$ . A escolha de  $\mathbf{x}^0$  influencia no número de iterações necessárias para atingir a precisão desejada. Quanto menor for  $\|\mathbf{x}^0 - \bar{\mathbf{x}}\|$  menos iterações serão necessárias.

### 3.2.2 Método Iterativo de Gauss-Jacobi

Vamos considerar o sistema linear  $\mathbf{Ax} = \mathbf{b}$  dado por

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + \cdots + a_{1,n}x_n = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + a_{2,3}x_3 + \cdots + a_{2,n}x_n = b_2 \\ a_{3,1}x_1 + a_{3,2}x_2 + a_{3,3}x_3 + \cdots + a_{3,n}x_n = b_3 \\ \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + a_{n,3}x_3 + \cdots + a_{n,n}x_n = b_n \end{cases},$$

onde os  $a_{ii} \neq 0$  para  $i = 1, 2, \dots, n$ . Em cada equação  $i$  podemos isolar a incógnita  $x_i$  obtendo as seguintes relações

$$\begin{aligned} x_1 &= \frac{1}{a_{1,1}} (b_1 - a_{1,2}x_2 - a_{1,3}x_3 - \dots - a_{1,n}x_n) \\ x_2 &= \frac{1}{a_{2,2}} (b_2 - a_{2,1}x_1 - a_{2,3}x_3 - \dots - a_{2,n}x_n) \\ x_3 &= \frac{1}{a_{3,3}} (b_3 - a_{3,1}x_1 - a_{3,2}x_2 - \dots - a_{3,n}x_n) \\ &\vdots \\ x_n &= \frac{1}{a_{n,n}} (b_n - a_{n,1}x_1 - a_{n,2}x_2 - \dots - a_{n,n-1}x_{n-1}) \end{aligned}$$

Na forma matricial estas equações são equivalentes à

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 0 & -\frac{a_{1,2}}{a_{1,1}} & -\frac{a_{1,3}}{a_{1,1}} & \dots & -\frac{a_{1,n}}{a_{1,1}} \\ -\frac{a_{2,1}}{a_{2,2}} & 0 & -\frac{a_{2,3}}{a_{2,2}} & \dots & -\frac{a_{2,n}}{a_{2,2}} \\ -\frac{a_{3,1}}{a_{3,3}} & -\frac{a_{3,2}}{a_{3,3}} & 0 & \dots & -\frac{a_{3,n}}{a_{3,3}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\frac{a_{n,1}}{a_{n,n}} & -\frac{a_{n,2}}{a_{n,n}} & -\frac{a_{n,3}}{a_{n,n}} & \dots & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} \frac{b_1}{a_{1,1}} \\ \frac{b_2}{a_{2,2}} \\ \frac{b_3}{a_{3,3}} \\ \vdots \\ \vdots \\ \frac{b_n}{a_{n,n}} \end{pmatrix} \quad (3.1)$$

Desta forma temos o sistema linear na forma  $\mathbf{x} = \mathbf{C}\mathbf{x} + \mathbf{g}$  e assim montamos o processo iterativo conhecido como Método Iterativo de Gauss Jacobi:

Dado  $\mathbf{x}^0$

$$\begin{aligned} x_1^{(k+1)} &= \frac{1}{a_{1,1}} (b_1 - a_{1,2}x_2^{(k)} - a_{1,3}x_3^{(k)} - \dots - a_{1,n}x_n^{(k)}) \\ x_2^{(k+1)} &= \frac{1}{a_{2,2}} (b_2 - a_{2,1}x_1^{(k)} - a_{2,3}x_3^{(k)} - \dots - a_{2,n}x_n^{(k)}) \\ x_3^{(k+1)} &= \frac{1}{a_{3,3}} (b_3 - a_{3,1}x_1^{(k)} - a_{3,2}x_2^{(k)} - \dots - a_{3,n}x_n^{(k)}) \\ &\vdots \\ x_n^{(k+1)} &= \frac{1}{a_{n,n}} (b_n - a_{n,1}x_1^{(k)} - a_{n,2}x_2^{(k)} - \dots - a_{n,n-1}x_{n-1}^{(k)}) \end{aligned} \quad (3.2)$$

**Algoritmo:** Método Iterativo de Gauss-Jacobi

**Input:** Matriz  $\mathbf{A} \in \mathbb{R}^{n \times n}$   $\mathbf{b}, \mathbf{x}^0 \in \mathbb{R}^n$  e  $\varepsilon > 0$

Enquanto  $\|\mathbf{x}^{k+1} - \mathbf{x}^k\| > \varepsilon$  faça:

Para  $s = 1, 2, \dots, n$ , faça:

$$x_s^{(k+1)} \leftarrow \frac{1}{a_{s,s}} \left( b_s - \sum_{j=1, j \neq s}^n a_{s,j} x_j^{(k)} \right)$$

fim para

fim enquanto

**Output:**  $\mathbf{x} \in \mathbb{R}^n$  : solução do sistema

### 3.2.3 Critério das Linhas

Como critério de convergência, vimos que a matriz de iteração  $\mathbf{C}$  deve satisfazer a condição  $\|\mathbf{C}\| < 1$ . Usando a Norma do Máximo das Linhas sobre a matriz  $\mathbf{C}$  em (3.2) temos o seguinte critério de convergência para o Método de Gauss-Jacobi

**Teorema 3.2.1 (Critério das Linhas)** *Dado o sistema linear  $\mathbf{Ax} = \mathbf{b}$ . Seja os  $\alpha_k$  de tal forma que:*

$$\alpha_k = \frac{1}{|a_{k,k}|} \sum_{j=1, j \neq k}^n |a_{k,j}| < 1 \quad \text{para } k = 1, 2, \dots, n$$

*Então o Método de Gauss-Jacobi gera uma seqüência  $\{\mathbf{x}^k\}$  que converge para a solução do sistema.*

Este critério fornece uma condição necessária, mas não suficiente. Isto é, o critério pode não ser satisfeito e o método ainda pode convergir. Outros critérios podem ser obtidos usando outras normas. Por exemplo, podemos obter o chamado critério das colunas aplicando a Norma do Máximo das Colunas na matriz em (3.2).

**Exemplo 3.2.1** *Dado o sistema linear*

$$\begin{cases} -7x_1 + 3x_2 + 2x_3 = -2 \\ x_1 + 3x_2 - x_3 = 3 \\ x_1 + x_2 - 3x_3 = -1 \end{cases}$$

*vamos procurar uma aproximação da solução, com precisão  $\varepsilon = 0.1$  na Norma do Máximo, usando o Método de Gauss-Jacobi. Vamos tomar como condição inicial o vetor  $\mathbf{x}^0 = (0.5, 0.5, 0.5)^T$ .*

O primeiro passo é verificar se o critério de convergência é satisfeito. Calculando os  $\alpha_k$  temos

$$\begin{aligned}\alpha_1 &= \frac{1}{|a_{1,1}|} (|a_{1,2}| + |a_{1,3}|) = \frac{5}{7} < 1 \\ \alpha_2 &= \frac{1}{|a_{2,2}|} (|a_{2,1}| + |a_{2,3}|) = \frac{2}{3} < 1 \\ \alpha_3 &= \frac{1}{|a_{3,3}|} (|a_{3,1}| + |a_{3,2}|) = \frac{2}{3} < 1\end{aligned}$$

Logo o critério das linhas é satisfeito e com isto temos garantia que o Método de Gauss-Jacobi converge. Montando o processo iterativo temos

$$\begin{aligned}x_1^{(k+1)} &= -\frac{1}{7} (-2 - 3x_2^{(k)} - 2x_3^{(k)}) \\ x_2^{(k+1)} &= \frac{1}{3} (3 - x_1^{(k)} + x_3^{(k)}) \\ x_3^{(k+1)} &= -\frac{1}{3} (-1 - x_1^{(k)} - x_2^{(k)})\end{aligned}\tag{3.3}$$

Assim para  $k = 0$  segue que

$$\begin{aligned}x_1^{(1)} &= -\frac{1}{7} (-2 - 3 * 0.5 - 2 * 0.5) = 0.642 \\ x_2^{(1)} &= \frac{1}{3} (3 - 0.5 + 0.5) = 1.000 \\ x_3^{(1)} &= -\frac{1}{3} (-1 - 0.5 - 0.5) = 0.666\end{aligned}\tag{3.4}$$

Verificando o critério de parada temos

$$\mathbf{x}^1 - \mathbf{x}^0 = \begin{pmatrix} 0.642 - 0.500 \\ 1.000 - 0.500 \\ 0.666 - 0.500 \end{pmatrix} = \begin{pmatrix} 0.142 \\ 0.500 \\ 0.166 \end{pmatrix} \Rightarrow \|\mathbf{x}^1 - \mathbf{x}^0\|_\infty = 0.500 > \varepsilon$$

Para  $k = 1$  temos

$$\begin{aligned}x_1^{(2)} &= -\frac{1}{7} (-2 - 3 * 1.000 - 2 * 0.666) = 0.904 \\ x_2^{(2)} &= \frac{1}{3} (3 - 0.642 + 0.666) = 1.008 \\ x_3^{(2)} &= -\frac{1}{3} (-1 - 0.642 - 1) = 0.880\end{aligned}\tag{3.5}$$

Verificando o critério de parada temos

$$\mathbf{x}^2 - \mathbf{x}^1 = \begin{pmatrix} 0.904 - 0.642 \\ 1.008 - 1.000 \\ 0.880 - 0.666 \end{pmatrix} = \begin{pmatrix} 0.262 \\ 0.008 \\ 0.214 \end{pmatrix} \Rightarrow \|\mathbf{x}^2 - \mathbf{x}^1\|_\infty = 0.262 > \varepsilon$$

Devemos continuar as iterações até que o critério de parada seja satisfeito (*Exercício*).

### 3.2.4 Método Iterativo de Gauss-Seidel

A cada iteração  $\mathbf{x}^k$  se aproxima da solução do sistema. Baseado nesta observação vamos modificar o Método de Gauss-Jacobi com o objetivo de acelerar a convergência. Numa iteração  $k + 1$ , o Método de Gauss-Jacobi calcula o elemento  $s$  pela equação

$$x_s^{(k+1)} = \frac{1}{a_{s,s}} \left( b_s - \sum_{j=1}^{s-1} a_{s,j} x_j^{(k)} - \sum_{j=s+1}^n a_{s,j} x_j^{(k)} \right) \quad (3.6)$$

Neste ponto os elementos  $x_1^{k+1}, x_2^{k+1}, \dots, x_{s-1}^{k+1}$ , já foram calculados e espera-se que estes estejam mais próximos da solução que os elementos  $x_1^k, x_2^k, \dots, x_{s-1}^k$ . Assim vamos substituir os elementos da iteração  $k$ , que aparecem no primeiro somatório de (3.6), pelos correspondentes elementos da iteração  $k + 1$ , isto é

$$x_s^{(k+1)} = \frac{1}{a_{s,s}} \left( b_s - \sum_{j=1}^{s-1} a_{s,j} x_j^{(k+1)} - \sum_{j=s+1}^n a_{s,j} x_j^{(k)} \right).$$

Como estamos usando valores mais próximos da solução, o cálculo de  $x_s^{k+1}$  será mais preciso. Este procedimento é conhecido como Método Iterativo de Gauss-Seidel, cujo o algoritmo é dado abaixo.

**Algoritmo:** Método Iterativo de Gauss-Seidel

**Input:** Matriz  $\mathbf{A} \in \mathbb{R}^{n \times n}$   $\mathbf{b}, \mathbf{x}^0 \in \mathbb{R}^n$  e  $\varepsilon > 0$

Enquanto  $\|\mathbf{x}^{k+1} - \mathbf{x}^k\| > \varepsilon$  faça:

Para  $s = 1, 2, \dots, n$ , faça:

$$x_s^{(k+1)} \leftarrow \frac{1}{a_{s,s}} \left( b_s - \sum_{j=1}^{s-1} a_{s,j} x_j^{(k+1)} - \sum_{j=s+1}^n a_{s,j} x_j^{(k)} \right)$$

fim para

fim enquanto

**Output:**  $\mathbf{x} \in \mathbb{R}^n$  : solução do sistema

### 3.2.5 Critério de Sassenfeld

O Método de Gauss-Seidel também pode ser representado na forma matricial  $\mathbf{x}^{k+1} = \mathbf{C}\mathbf{x}^k + \mathbf{g}$  e critérios de convergência podem ser obtidos impondo a condição  $\|\mathbf{C}\| < 1$ . Aplicando a Norma do Máximo das Linhas obtemos o seguinte critério de convergência:

**Teorema 3.2.2 (Critério de Sassenfeld)** *Dado o sistema linear  $\mathbf{Ax} = \mathbf{b}$ . Seja os  $\beta_k$  de tal forma que:*

$$\beta_k = \frac{1}{|a_{k,k}|} \left( \sum_{j=1}^{k-1} |a_{k,j}| \beta_j + \sum_{j=k+1}^n |a_{k,j}| \right) < 1 \quad \text{para } k = 1, 2, \dots, n$$

*Então o Método de Gauss-Seidel gera uma seqüência  $\{\mathbf{x}^k\}$  que converge para a solução do sistema.*

**Exemplo 3.2.2** *Dado o sistema linear*

$$\begin{cases} -7x_1 + 3x_2 + 2x_3 = -2 \\ x_1 + 2x_2 - x_3 = 2 \\ x_1 + x_2 - 2x_3 = 0 \end{cases}$$

*vamos procurar uma aproximação da solução, com precisão  $\varepsilon = 0.1$  na norma do máximo, usando o Método de Gauss-Seidel. Vamos tomar como condição inicial o vetor  $\mathbf{x}^0 = (0.5, 0.5, 0.5)^T$ .*

*O primeiro passo é verificar se o critério de convergência é satisfeito. Calculando os  $\beta_k$  temos*

$$\begin{aligned} \beta_1 &= \frac{1}{|a_{1,1}|} (|a_{1,2}| + |a_{1,3}|) = \frac{5}{7} < 1 \\ \beta_2 &= \frac{1}{|a_{2,2}|} (|a_{2,1}| \beta_1 + |a_{2,3}|) = \frac{6}{7} < 1 \\ \beta_3 &= \frac{1}{|a_{3,3}|} (|a_{3,1}| \beta_1 + |a_{3,2}| \beta_2) = \frac{11}{14} < 1 \end{aligned}$$

*Logo o critério de Sassenfeld é satisfeito e com isto temos garantia que o Método de Gauss-Seidel converge. Note que se aplicarmos o critério das linhas obtemos que  $\alpha_2 = \alpha_3 = 1$ , ou seja, o critério das linhas não é satisfeito. Este é um exemplo em que não temos a garantia de convergência do Método de Gauss-Jacobi.*

*Montando o processo iterativo temos*

$$\begin{aligned} x_1^{(k+1)} &= -\frac{1}{7} (-2 - 3x_2^{(k)} - 2x_3^{(k)}) \\ x_2^{(k+1)} &= \frac{1}{2} (2 - x_1^{(k+1)} + x_3^{(k)}) \\ x_3^{(k+1)} &= -\frac{1}{2} (0 - x_1^{(k+1)} - x_2^{(k+1)}) \end{aligned} \tag{3.7}$$



Assim para  $k = 0$  segue que

$$\begin{aligned}x_1^{(1)} &= -\frac{1}{7}(-2 - 3 * 0.5 - 2 * 0.5) = 0.642 \\x_2^{(1)} &= \frac{1}{2}(2 - 0.642 + 0.5) = 0.929 \\x_3^{(1)} &= -\frac{1}{2}(0 - 0.642 - 0.929) = 0.785\end{aligned}\tag{3.8}$$

Verificando o critério de parada temos

$$\mathbf{x}^1 - \mathbf{x}^0 = \begin{pmatrix} 0.642 - 0.500 \\ 0.929 - 0.500 \\ 0.785 - 0.500 \end{pmatrix} = \begin{pmatrix} 0.142 \\ 0.429 \\ 0.285 \end{pmatrix} \Rightarrow \|\mathbf{x}^1 - \mathbf{x}^0\|_\infty = 0.429 > \varepsilon$$

Para  $k = 1$  temos

$$\begin{aligned}x_1^{(2)} &= -\frac{1}{7}(-2 - 3 * 0.929 - 2 * 0.785) = 0.908 \\x_2^{(2)} &= \frac{1}{2}(2 - 0.908 + 0.785) = 0.938 \\x_3^{(2)} &= -\frac{1}{2}(0 - 0.908 - 0.938) = 0.923\end{aligned}\tag{3.9}$$

Verificando o critério de parada temos

$$\mathbf{x}^2 - \mathbf{x}^1 = \begin{pmatrix} 0.908 - 0.642 \\ 0.938 - 0.929 \\ 0.923 - 0.785 \end{pmatrix} = \begin{pmatrix} 0.266 \\ 0.009 \\ 0.138 \end{pmatrix} \Rightarrow \|\mathbf{x}^2 - \mathbf{x}^1\|_\infty = 0.266 > \varepsilon$$

Devemos continuar as iterações até que o critério de parada seja satisfeito (*Exercício*).

### 3.3 Observações Finais

A escolha do método, que deve ser aplicado a um determinado problema, deve ser orientada nas características de cada método que apresentamos nesta seção.

Os métodos diretos apresentam a solução de qualquer sistema linear não singular, porém não temos um controle sobre a precisão da solução. Aplicados em sistemas de grande porte e matriz cheia ( dimensão acima  $50 \times 50$  e poucos elementos  $a_{i,j} = 0$  ) apresentam grandes erros de arredondamentos. Os métodos iterativos permitem um controle sobre a precisão da solução, porém estes não se aplicam a qualquer sistema. O sistema deve satisfazer certas condições de convergência para que determinado método seja aplicado.

O Método de Gauss-Jacobi é indicado para processamento paralelo ou vetorial, pois os elementos no passo  $k + 1$  dependem somente dos elementos no passo  $k$ . Se  $T$  for o tempo que uma máquina seqüencial toma para executar uma iteração. Numa máquina paralela este tempo cai para  $\lceil T/Np \rceil$ , onde  $Np$  é o número de processadores.

O Método de Gauss-Seidel não é indicado para processamento paralelo, pois o cálculo de  $x_s^{k+1}$  depende de  $x_1^{k+1}, x_2^{k+1}, \dots, x_{s-1}^{k+1}$ . Porém este converge mais rapidamente que o Método de Gauss-Jacobi, quando ambos são executado em processamento seqüencial. Além disso, todo sistema que satisfaz o Critério das Linhas também satisfaz o Critério de Sassenfeld. Ou seja, todo sistema em que podemos aplicar o Método de Gauss-Jacobi, automaticamente podemos aplicar o Método de Gauss-Seidel.

### 3.4 Exercícios

**Exercício 3.1** *Resolva o sistema linear abaixo, usando o Método de Eliminação de Gauss.*

$$\begin{cases} 1.7x_1 + 2.3x_2 - 0.5x_3 = 4.55 \\ 1.1x_1 + 0.6x_2 - 1.6x_3 = -3.40 \\ 2.7x_1 - 0.8x_2 + 1.5x_3 = 5.50 \end{cases}$$

**Exercício 3.2** *Ache a inversa da matriz abaixo.*

$$\begin{pmatrix} 1 & -2 & 2 \\ 2 & -3 & 2 \\ 2 & -2 & 1 \end{pmatrix}$$

**Exercício 3.3** *Um sistema é dito ser tridiagonal se este é formado pela diagonal principal, a primeira diagonal secundária inferior e a primeira diagonal secundária superior. Os outros elementos são nulos. Isto é, a matriz associada é da forma:*

$$\begin{pmatrix} a_{1,1} & a_{1,2} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & a_{3,2} & a_{3,3} & a_{3,4} & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & a_{4,3} & a_{4,4} & a_{4,5} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & a_{n-1,n-2} & a_{n-1,n-1} & a_{n-1,n} \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

*Faça uma modificação no Método de Eliminação de Gauss explorando a forma do sistema. Implemente o algoritmo em MatLab.*

**Exercício 3.4** *Considere o sistema linear*

$$\begin{cases} 0.0002x_1 + 2x_2 = 2 \\ 2x_1 + 2x_2 = 4 \end{cases}$$

*Calcule a solução do sistema por Eliminação de Gauss e Pivotamento Parcial, usando 4 casas decimais, sem arredondamento. Calcule o resíduo  $\mathbf{r} = \mathbf{b} - \mathbf{A}\bar{\mathbf{x}}$  e comente seus resultados.*

**Exercício 3.5** Dado o sistema linear

$$\begin{cases} 0.780x + 0.563y = 0.217 \\ 0.913x + 0.659y = 0.254 \end{cases}$$

- a) Calcule a solução do sistema por (i)-Eliminação de Gauss e (ii)-Pivotamento Parcial, usando no mínimo 7 casas decimais, sem arredondamento.
- b) Calcule o resíduo  $\mathbf{r} = \mathbf{b} - \mathbf{A}\bar{\mathbf{x}}$  para os casos (i) e (ii).
- c) Se no item a) tivéssemos usado 3 casas decimais, o que ocorreria com a solução do sistema? Comente seus resultados.

**Exercício 3.6** Mostre que, se um sistema linear satisfaz o Critério das Linhas, então este também satisfaz o Critério de Sassenfeld.

**Exercício 3.7** Seja  $k$  um número inteiro, positivo, considere:

$$\begin{cases} \mathbf{k}x_1 + x_2 = 2 \\ \mathbf{k}x_1 + 2x_2 + \frac{\mathbf{k}}{5}x_3 = 3 \\ \mathbf{k}x_1 + x_2 + 2x_3 = 2 \end{cases}$$

- a) Verifique para que valores de  $\mathbf{k}$ , a convergência do Método de Gauss-Jacobi pode ser garantida.
- b) Verifique para que valores de  $\mathbf{k}$ , a convergência do Método de Gauss-Seidel pode ser garantida.
- c) Utilize um método iterativo adequado para calcular a aproximação da solução deste sistema de equações considerando:
- (i)  $\mathbf{x}^{(0)} = (1.0, 1.0, 1.0)^T$
- (ii) Escolha  $\mathbf{k}$  como o menor inteiro que satisfaça as condições de convergência.
- (iii) Faça duas iterações e calcule o erro absoluto cometido, usando a norma do máximo.

**Exercício 3.8** Dado o sistema  $\mathbf{Ax} = \mathbf{b}$  podemos montar um processo iterativo da seguinte forma

$$\mathbf{x}^{k+1} = (\mathbf{I} + \mathbf{A})\mathbf{x}^k - \mathbf{b}$$

- a) Enuncie uma condição suficiente de convergência baseada na Norma do Máximo das Linhas.
- b) Faça três iterações do processo acima para o sistema linear

$$\begin{cases} -1.3x_1 + 0.3x_2 = 1 \\ 0.5x_1 - 0.5x_2 = 0 \end{cases} \quad \text{tomando } \mathbf{x}^{(0)} = \begin{pmatrix} 0.8 \\ 0.8 \end{pmatrix}$$

# Capítulo 4

## Ajuste de Curvas: Método dos Mínimos Quadrados

Em geral, experimentos em laboratório gera uma gama de dados que devem ser analisados para a criação de um modelo. Obter uma função matemática que represente (ou que ajuste) os dados permite fazer simulações do processo de forma confiável, reduzindo assim repetições de experimentos que podem ter um custo alto. Neste capítulo vamos analisar o esquema dos Mínimos Quadrados, que fornece uma função que melhor represente os dados.

### 4.1 Método dos Mínimos Quadrados - Caso Discreto

Dado um conjunto de pontos  $(x_k, f(x_k))$ ,  $k = 0, 1, 2, \dots, m$ . O problema de ajuste de curvas consiste em encontrar uma função  $\varphi(x)$  tal que o desvio em cada ponto  $k$ , definido por

$$d_k = f(x_k) - \varphi(x_k),$$

seja mínimo, onde  $\varphi(x)$  é uma combinação linear de funções contínuas  $g_i(x)$ ,  $i = 1, 2, \dots, n$ , escolhidas de acordo com os dados do problema. Isto é

$$\varphi(x) = \alpha_1 g_1(x) + \alpha_2 g_2(x) + \dots + \alpha_n g_n(x)$$

Neste caso dizemos que o ajuste é linear. A escolha das funções  $g_i$  depende do gráfico dos pontos, chamado de *diagrama de dispersão*, através do qual podemos visualizar que tipo de curva que melhor se ajusta aos dados.

**Exemplo 4.1.1** *Vamos considerar a tabela de pontos dada abaixo.*

$x$	0.10	0.20	0.50	0.65	0.70	0.80	0.90	1.10	1.23	1.35	1.57	1.70	1.75	1.80	1.94
$f(x)$	0.19	0.36	0.75	0.87	0.91	0.96	0.99	0.99	0.94	0.87	0.67	0.51	0.43	0.36	0.11

A análise do gráfico de dispersão (Fig. 4.1) mostra que a função que procuramos se comporta como uma parábola. Logo poderíamos escolher as funções  $g_1(x) = 1$ ,  $g_2(x) = x$  e

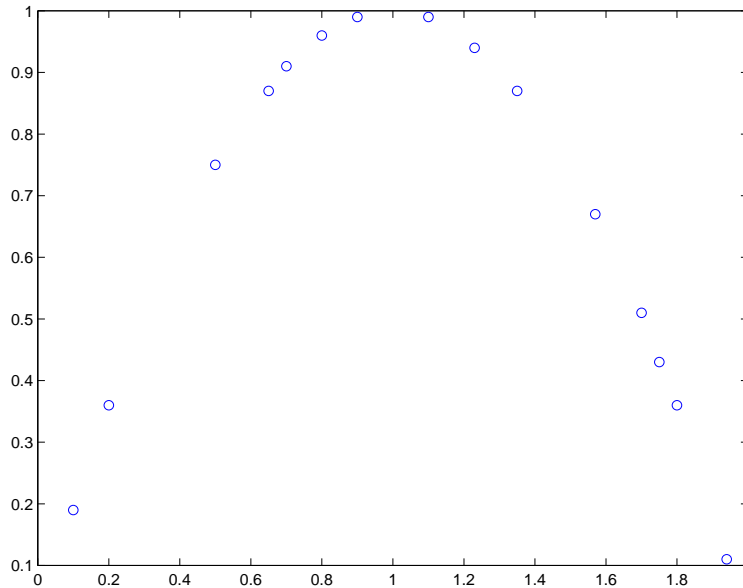


Figura 4.1: Diagrama de Dispersão

$g_3(x) = x^2$ , pois  $\varphi(x) = \alpha_1 g_1(x) + \alpha_2 g_2(x) + \alpha_3 g_3(x)$  representa “todas” as parábolas e com a escolha adequada dos  $\alpha_i$  teremos aquela que melhor se ajusta aos pontos.

O Método dos Mínimos Quadrados consiste em determinar os  $\alpha_i$  de tal forma que a soma dos quadrados dos desvios em seja mínimo, Isto é: Achar os  $\alpha_i$  que minimizam a função

$$F(\alpha_1, \alpha_2, \dots, \alpha_n) = \sum_{k=1}^m [f(x_k) - \overbrace{(\alpha_1 g_1(x_k) + \dots + \alpha_n g_n(x_k))}^{\varphi(x_k)}]^2.$$

A função  $F$  é uma função que satisfaz  $F(\alpha) \geq 0 \quad \forall \alpha \in \mathbb{R}^m$ . Isto é, uma função limitada inferiormente e portanto esta tem um ponto de mínimo. Este ponto pode ser determinado pelo teste da primeira derivada, sendo

$$\left. \frac{\partial F}{\partial \alpha_i} \right|_{(\alpha_1, \dots, \alpha_n)} = 0 \quad i = 1, \dots, n.$$

Desta forma temos

$$-2 \sum_{k=1}^m [f(x_k) - \alpha_1 g_1(x_k) - \alpha_2 g_2(x_k) - \dots - \alpha_n g_n(x_k)] g_j(x_k) = 0 \Rightarrow$$



$$\begin{aligned}
 a_{2,2} &= \sum_{k=1}^{15} g_2(x_k) * g_2(x_k) = 22.62 \\
 a_{2,3} &= \sum_{k=1}^{15} g_2(x_k) * g_3(x_k) = 34.92 = a_{3,2} \\
 a_{3,3} &= \sum_{k=1}^{15} g_3(x_k) * g_3(x_k) = 57.09 \\
 b_1 &= \sum_{k=1}^{15} f(x_k) * g_1(x_k) = 9.91 \\
 b_2 &= \sum_{k=1}^{15} f(x_k) * g_2(x_k) = 10.28 \\
 b_3 &= \sum_{k=1}^{15} f(x_k) * g_3(x_k) = 12.66
 \end{aligned}$$

Obtendo assim um sistema linear que pode ser resolvido por um esquema numérico estudado no capítulo 3. A solução do sistema é dado por

$$\alpha_1 = 0.00, \quad \alpha_2 = 1.99, \quad \alpha_3 = -0.99$$

Portanto a função  $\varphi$  é dada por

$$\varphi(x) = 1.99x - 0.99x^2$$

A figura 4.2 compara a função  $\varphi(x)$  com o gráficos dos pontos.

Através da função  $\varphi$  podemos aproximar valores de  $f$  em pontos que não pertencem a tabela. Por exemplo:

$$\begin{aligned}
 f(0) &\approx \varphi(0) = 0 \\
 f(1) &\approx \varphi(1) = 1 \\
 f(2) &\approx \varphi(2) = 0.02
 \end{aligned}$$

No exemplo ajustamos os dados a uma parábola, mas outras funções bases poderiam ser usadas. Como exemplo, poderíamos pensar que os dados representam o primeiro meio ciclo de uma função senoidal. E neste caso poderíamos tomar  $g_1(x) = 1$  e  $g_2(x) = \text{sen}(2\pi x)$  Mas afinal qual seria a melhor escolha? (Veja exercício 4.1) O desvio fornece uma medida que pode ser usada como parâmetro de comparação entre ajustes diferentes. No caso do ajuste pela parábola temos que o desvio é dado por

$$D = \sum_{k=1}^{15} (f(x_k) - \varphi(x_k))^2 = 0.0019$$

Se o ajuste feito por uma função senoidal tiver um desvio menor, então este ajuste representaria melhor os dados. Outro ponto a ser observado é que a dimensão do sistema linear depende do número de funções bases que estamos usando. No caso da parábola usamos três funções bases e temos um sistema  $3 \times 3$ . No caso de uma função senoidal teremos um sistema  $2 \times 2$ .

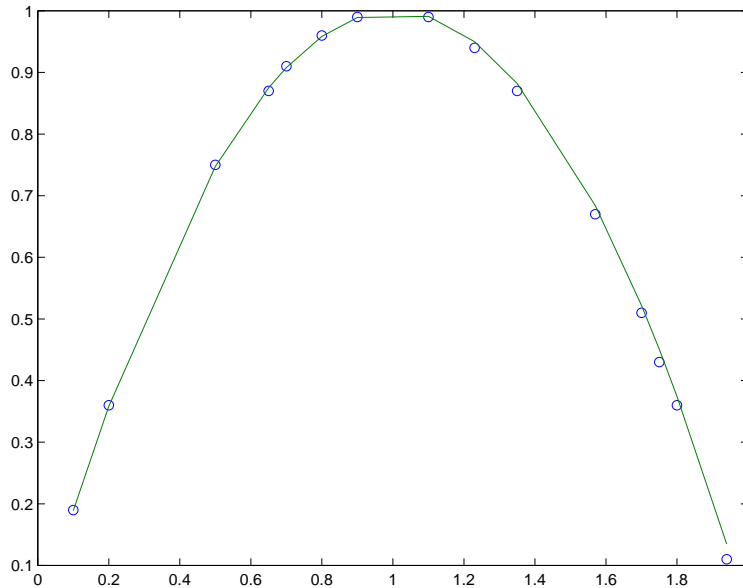


Figura 4.2: Diagrama de Dispersão com o gráfico da  $\varphi(x)$ .

## 4.2 Método dos Mínimos Quadrados - Caso Contínuo

No caso contínuo temos uma função  $f(x)$  dada num intervalo  $[a, b]$  e não mais uma tabela de pontos. O procedimento é análogo ao caso discreto. Escolhidas as funções bases  $g_i$  devemos determinar a função  $\varphi(x) = \alpha_1 g_1(x) + \alpha_2 g_2(x) + \dots + \alpha_n g_n(x)$  de modo que o desvio seja mínimo, onde

$$D = \int_a^b (f(x) - \varphi(x))^2 dx$$

Neste caso os  $\alpha_i$  também são determinados pela resolução de um sistema, onde o elemento  $a_{i,j}$  é obtido através do produto interno entre as funções  $g_i(x)$  e  $g_j(x)$  e o elementos  $b_i$  pelo produto interno entre  $f(x)$  e  $g_i(x)$ , sendo

$$a_{i,j} = \int_a^b g_i(x)g_j(x)dx \quad \text{e} \quad b_i = \int_a^b f(x)g_i(x)dx$$

**Exemplo 4.2.1** *Vamos determinar a melhor parábola que se ajuste a função  $f(x) = \sin(\pi x)$  no intervalo  $[0, 1]$ . Para isto devemos tomar, como funções bases, as funções  $g_1(x) = 1$ ,  $g_2(x) = x$  e  $g_3(x) = x^2$ . Calculando os termos do sistema linear temos*

$$\begin{aligned} a_{1,1} &= \int_0^1 g_1(x)g_1(x)dx = 1 \\ a_{1,2} &= \int_0^1 g_1(x)g_2(x)dx = \frac{1}{2} = a_{2,1} \end{aligned}$$



$$\begin{aligned}
 a_{1,3} &= \int_0^1 g_1(x)g_3(x)dx = \frac{1}{3} = a_{3,1} \\
 a_{2,2} &= \int_0^1 g_2(x)g_2(x)dx = \frac{1}{3} \\
 a_{2,3} &= \int_0^1 g_2(x)g_3(x)dx = \frac{1}{4} = a_{3,2} \\
 a_{3,3} &= \int_0^1 g_3(x)g_3(x)dx = \frac{1}{25} \\
 b_1 &= \int_0^1 f(x)g_1(x)dx = 0.636 \\
 b_2 &= \int_0^1 f(x)g_2(x)dx = 0.318 \\
 b_3 &= \int_0^1 f(x)g_3(x)dx = 0.189
 \end{aligned}$$

cuja a solução é dada por  $\alpha_1 = -0.027$ ,  $\alpha_2 = 4.032$  e  $\alpha_3 = -4.050$ . Assim temos que  $\varphi(x) = -0.027 + 4.032x - 4.050x^2$ . A figura (4.3) mostra o gráfico comparativo entre a função  $f(x)$  (linha: —) e o ajuste  $\varphi(x)$  (linha:  $\cdots$ ).

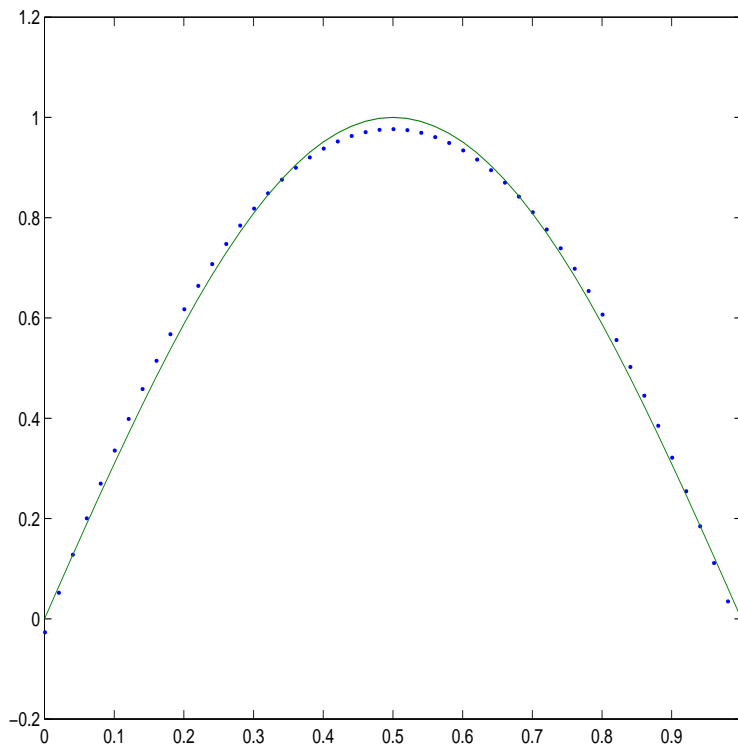


Figura 4.3: — :  $f(x) = \text{sen}(\pi x)$ ;  $\cdots$  :  $\varphi(x)$

### 4.3 Ajuste Não Linear

Existem casos, onde o diagrama de dispersão de uma função indica que os dados devem ser ajustado por uma função que não é linear com relação aos  $\alpha_i$ . Como exemplo, vamos considerar os dados

$x$	-1.0	-0.5	0	0.5	1.0	1.5	2.0	2.5	3.0
$f(x)$	0.157	0.234	0.350	0.522	0.778	1.162	1.733	2.586	3.858

Montando o diagrama de dispersão (Veja figura 4.4) podemos considerar que  $f(x)$  tem um comportamento exponencial. Isto é,  $f(x) \approx \varphi(x) = \alpha_1 e^{\alpha_2 x}$ . Desta forma, um processo

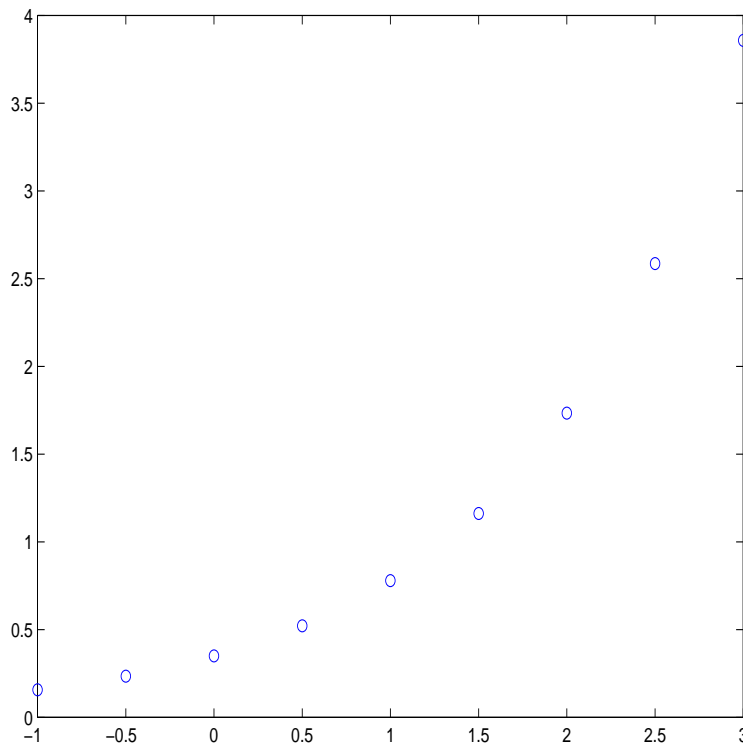


Figura 4.4: Diagrama de dispersão

de linearização deve ser empregado, para que seja possível aplicar o Método dos Mínimos Quadrados. Neste caso podemos proceder da seguinte forma.

$$f(x) = \alpha_1 e^{\alpha_2 x} \Rightarrow z = \ln(f(x)) = \ln(\alpha_1) + \alpha_2 x.$$

Fazendo  $\beta_1 = \ln(\alpha_1)$  e  $\beta_2 = \alpha_2$  o problema consiste em ajustar os dados de  $z$  por uma reta. Para isto tomamos  $g_1(x) = 1$  e  $g_2(x) = x$ . Calculando as funções em cada um dos pontos

temos

$x$	-1.0	-0.5	0	0.5	1.0	1.5	2.0	2.5	3.0
$f(x)$	0.157	0.234	0.350	0.522	0.778	1.162	1.733	2.586	3.858
$z = \ln(f(x))$	-1.851	-1.452	-1.049	-0.650	-0.251	0.150	0.549	0.950	1.350
$g_1(x)$	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
$g_2(x)$	-1.0	-0.5	0	0.5	1.0	1.5	2.0	2.5	3.0

Calculando os elementos da matriz e vetor dos termos independente temos que

$$\begin{aligned}
 a_{1,1} &= \sum_{k=1}^9 g_1(x_k) * g_1(x_k) = 9 \\
 a_{1,2} &= \sum_{k=1}^9 g_1(x_k) * g_2(x_k) = 9 = a_{2,1} \\
 a_{2,2} &= \sum_{k=1}^9 g_2(x_k) * g_2(x_k) = 24 \\
 b_1 &= \sum_{k=1}^{15} z(x_k) * g_1(x_k) = -2.254 \\
 b_2 &= \sum_{k=1}^{15} z(x_k) * g_2(x_k) = 9.749
 \end{aligned}$$

Cuja a solução é dada por

$$\beta_1 = -1.050 \quad \text{e} \quad \beta_2 = 0.800$$

Desta forma os valores de  $\alpha_i$  são dados por:

$$\alpha_1 = e^{\beta_1} = 0.349 \quad \text{e} \quad \alpha_2 = \beta_2 = 0.800$$

Portanto temos

$$\varphi(x) = \alpha_1 e^{\alpha_2 x} = 0.349 e^{0.800x}$$

A figura 4.5 mostra a comparação dos dados com a função obtida. Para verificar se função escolhida para a aproximação foi bem feita, usamos o teste de alinhamento. Este consiste em tomarmos os dados “linearizados”, isto é, os pontos  $z$  da tabela, e fazer o diagrama de dispersão. Se os pontos estiverem alinhados, então a escolha da função foi boa. A figura 4.6 mostra o diagrama de dispersão dos dados em  $z$ , obtidos no nosso exemplo. Podemos concluir que a nossa escolha pela exponencial foi uma escolha acertada.

## 4.4 Observações Finais

O ajuste de curvas permite extrapolar os valores tabelados. Isto é, se os dados estão tabelados num intervalo  $[x_0, x_m]$  podemos aproximar um  $\bar{x} \notin [x_0, x_m]$  com uma certa segurança. Como

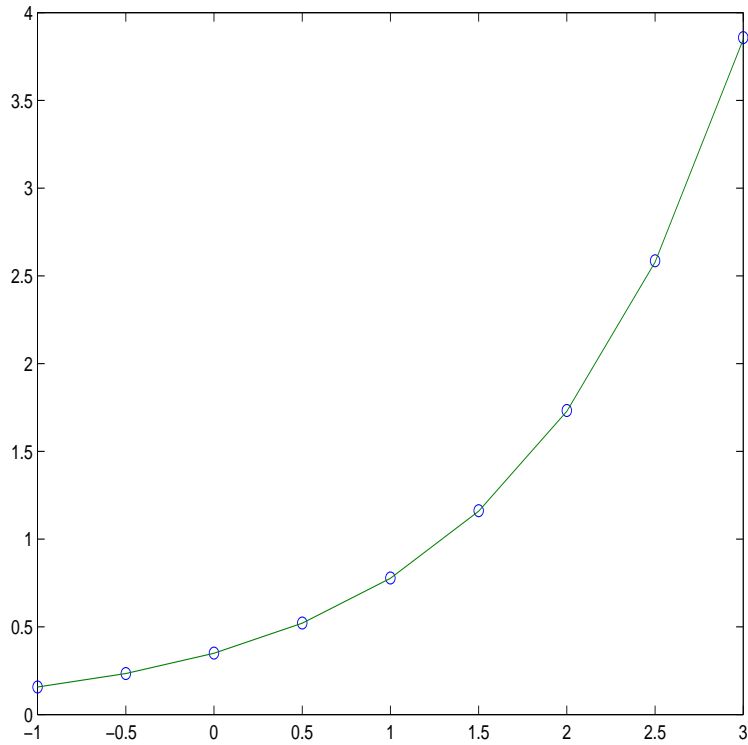


Figura 4.5: Diagrama de Dispersão e o Gráfico da  $\varphi(x)$

os dados provêm de experimentos que estão sujeitos a erros de medições, podemos ter mais de um valor para um determinado ponto. (Veja exercício 4.4) A função obtida considera os dois valores faz “ uma média ” entre estes valores.

Os elementos  $a_{i,j}$  são obtidos pelo produto interno entre as funções  $g_i$  e  $g_j$  definidos por

$$\text{Caso Discreto: } \langle g_i, g_j \rangle = \sum_{k=1}^m g_i(x_k)g_j(x_k)$$

$$\text{Caso Contínuo: } \langle g_i, g_j \rangle = \int_a^b g_i(x)g_j(x)dx$$

Se as funções  $g_i$  forem ortogonais, isto é

$$\langle g_i, g_j \rangle = \begin{cases} 0, & \text{para } i \neq j \\ k_i, & \text{para } i = j \end{cases}$$

a matriz obtida será diagonal e conseqüentemente a solução do sistema é imediata. Como exemplo, veja exercício 4.5.

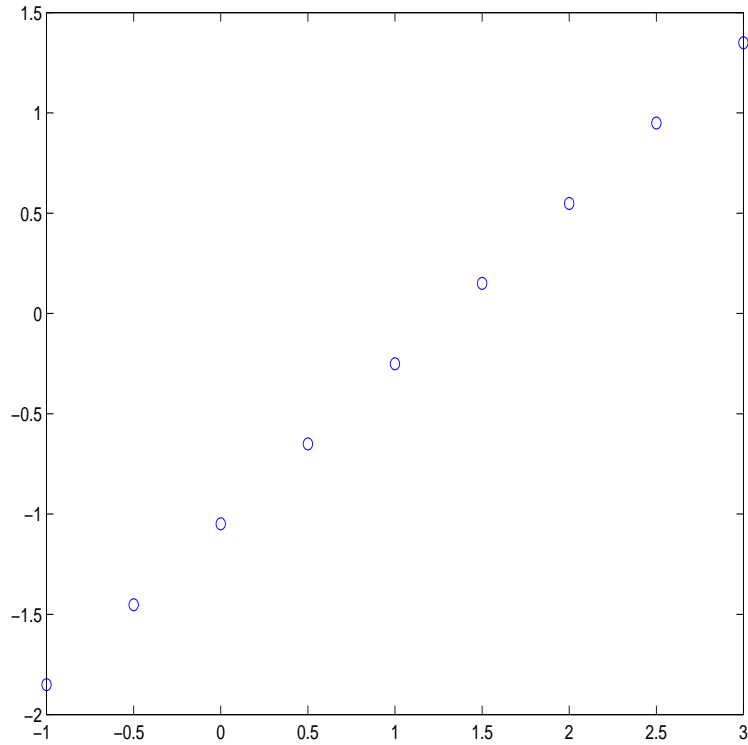


Figura 4.6: Diagrama dos dados linearizados

## 4.5 Exercícios

**Exercício 4.1** Usando os dados abaixo, faça um ajuste de curva com  $g_1(x) = 1$  e  $g_2(x) = \sin(2\pi x)$ . Calcule o desvio e compare com os resultados obtidos no Exemplo (4.1.1).

$x$	0.10	0.20	0.50	0.65	0.70	0.80	0.90	1.10	1.23	1.35	1.57	1.70	1.75	1.80	1.94
$f(x)$	0.19	0.36	0.75	0.87	0.91	0.96	0.99	0.99	0.94	0.87	0.67	0.51	0.43	0.36	0.11

**Exercício 4.2** Dada a tabela abaixo

$x$	0.50	0.75	1.00	1.50	2.00	2.50	3.00
$f(x)$	0.479	0.681	0.841	0.997	0.909	0.598	0.141

Entre os grupos de funções bases abaixo, escolha aquele que representará melhor resultado num ajuste de curvas. Justifique sua escolha. Faça um ajuste considerando sua escolha.

Grupo I:  $g_1(x) = 1$  e  $g_2(x) = x$ .

Grupo II:  $g_1(x) = 1$  e  $g_2(x) = e^x$ .

Grupo III:  $g_1(x) = 1$  e  $g_2(x) = x^2$ .

**Exercício 4.3** A tabela abaixo representa o calor específico da água em função da temperatura.

$t(^{\circ}C)$	0	5	10	25	30	35
$C(t)$	1.00762	1.00392	1.00153	0.99852	0.99826	0.99818

Fac ca um ajuste linear, um quadrático e um cúbico. Faça um ajuste não linear da forma  $\varphi(x) = \alpha_1 e^{-\alpha_2}$ , com  $\alpha_1, \alpha_2 > 0$ . Calcule o desvio e ache uma aproximação para  $t = 15$  em cada um dos casos. Sabendo que o valor exato da função  $C(15) = 1.00000$ , qual dos casos acima apresentou melhor aproximação?

**Exercício 4.4** Ajustar os dados abaixo à função  $z = \frac{1}{1 + e^{(\alpha_1 x + \alpha_2)}}$

$x$	0.1	0.3	0.5	0.5	0.7	0.8	0.8	1.10	1.30	1.80
$f(x)$	0.833	0.625	0.500	0.510	0.416	0.384	0.395	0.312	0.277	0.217

Verifique, pelo teste do alinhamento, qual é a melhor escolha para ajustar os dados entre as funções  $z = \alpha_1 \alpha_2^x$  e  $z = \alpha_1 x^{\alpha_2}$ . (**Obs:** Note que neste caso a tabela apresenta dois valores diferentes para os pontos  $x = 0.5$  e  $x = 0.8$ . Isto pode ocorrer se estamos considerando que os dados provém de experimentos que são realizados mais de uma vez. )

**Exercício 4.5** Usando os polinômios de Legendre  $g_1(x) = x$ ,  $g_2(x) = x$  e  $g_2(x) = \frac{1}{2}(3x^2 - 1)$ , que são ortogonais em  $[-1, 1]$ , ache a melhor parábola que aproxima a função  $f(x) = \cos(x)$  no intervalo  $[-3, 3]$ . (**Obs:** A ortogonalidade dos polinômios nos forneceria uma matriz diagonal, se o ajuste fosse feito no intervalo  $[-1, 1]$ . Logo devemos fazer uma mudança de variável de tal forma que obteremos novas  $g_i$  que serão ortogonais em  $[-3, 3]$  )



A matriz  $\mathbf{A}$ , associada ao sistema, é uma matriz de Vandermonde, cujo o determinante é dado por

$$\text{Det}(\mathbf{A}) = \prod_{l=1}^n \prod_{j=0}^{l-1} (x_l - x_j)$$

Como  $x_l \neq x_j$  para  $l \neq j$ , segue que o determinante da matriz  $\mathbf{A}$  é diferente de zero e portanto o sistema admite uma única solução ■

**Exemplo 5.0.1** *Vamos achar uma aproximação para  $f(0.3)$  usando o polinômio interpolador dos dados abaixo.*

$$\begin{array}{c|ccc} x_k & 0.0 & 0.2 & 0.4 \\ \hline f_k & 4.00 & 3.84 & 3.76 \end{array}$$

Como temos, três pontos ( $n + 1 = 3$ ), o grau do polinômio será menor ou igual a dois. Logo

$$p(x) = a_0 + a_1x + a_2x^2$$

Impondo a condição  $f_k = p(x_k)$  obtemos:

$$\begin{aligned} f_0 = 4.00 &= p(0) = a_0 + a_1 \cdot 0 + a_2 \cdot 0^2 \\ f_1 = 3.84 &= p(0.2) = a_0 + a_1 \cdot 0.2 + a_2 \cdot 0.2^2 \\ f_2 = 3.76 &= p(0.4) = a_0 + a_1 \cdot 0.4 + a_2 \cdot 0.4^2 \end{aligned}$$

Que equivale ao sistema linear na forma matricial

$$\left( \begin{array}{ccc|c} 1 & 0 & 0 & 4.00 \\ 1 & 0.2 & 0.04 & 3.84 \\ 1 & 0.4 & 0.16 & 3.76 \end{array} \right)$$

A solução deste sistema é  $a_0 = 4$ ,  $a_1 = -1$  e  $a_2 = 1$ , obtendo assim

$$p(x) = x^2 - x + 4$$

Desta forma

$$f(0.3) \approx p(0.3) = 3.79$$

Existem outras formas de encontrar o polinômio interpolador que a resolução de sistemas. Teoricamente estes procedimentos resultam no mesmo polinômio  $p_n(x)$ . A escolha de uma ou outra forma depende dos dados que devemos interpolar.



## 5.1 Forma de Lagrange

Vamos considerar o conjunto de  $n+1$  pontos  $(x_k, f_k)$ ,  $k = 0, 1, \dots, n$  distintos e vamos considerar o polinômio representado por

$$p_n(x) = f_0L_0(x) + f_1L_1(x) + \dots + f_nL_n(x) = \sum_{k=0}^n f_kL_k(x)$$

onde  $L_k(x)$  é um polinômio de grau  $\leq n$  que satisfaz a relação

$$L_k(x_j) = \begin{cases} 0 & \text{se } k \neq j \\ 1 & \text{se } k = j \end{cases}$$

Com isto temos que

$$p_n(x_j) = f_0L_0(x_j) + f_1L_1(x_j) + \dots + f_jL_j(x_j) + \dots + f_nL_n(x_j) = f_j$$

Logo  $p_n(x)$  é o polinômio interpolador de  $f(x)$  nos pontos  $x_0, x_1, \dots, x_n$ . Os polinômios  $L_k(x)$  são chamados de polinômios de Lagrange e estes são obtidos da forma:

$$L_k(x) = \frac{(x-x_0)(x-x_1)\dots(x-x_{k-1})(x-x_{k+1})\dots(x-x_n)}{(x_k-x_0)(x_k-x_1)\dots(x_k-x_{k-1})(x_k-x_{k+1})\dots(x_k-x_n)}$$

**Exemplo 5.1.1** *Vamos considerar a tabela de pontos do exemplo anterior*

$x$	0.0	0.2	0.4
$f(x)$	4.00	3.84	3.76

Calculando os  $L_k(x)$  temos

$$\begin{aligned} L_0(x) &= \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} = \frac{(x-0.2)(x-0.4)}{(0-0.2)(0-0.4)} = \frac{1}{0.08}(x^2 - 0.6x + 0.08) \\ L_1(x) &= \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} = \frac{(x-0)(x-0.4)}{(0.2-0)(0.2-0.4)} = \frac{-1}{0.04}(x^2 - 0.4x) \\ L_2(x) &= \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} = \frac{(x-0)(x-0.2)}{(0.4-0)(0.4-0.2)} = \frac{1}{0.08}(x^2 - 2.6x) \end{aligned}$$

Assim temos que

$$p(x) = x^2 - x + 4$$

Observe que o polinômio é o mesmo obtido pela resolução de sistema. Isto já era esperado, pois o polinômio interpolador **é único**.

## 5.2 Forma de Newton

A forma de Newton do polinômio interpolador é baseada nos operadores de diferenças divididas. Seja  $f(x)$  uma função tabelada em  $n + 1$  pontos distintos  $x_0, x_1, \dots, x_n$ . Definimos o operador de diferença dividida de ordem zero em  $x_k$  por:

$$f[x_k] = f(x_k).$$

O operador de diferença dividida de ordem um, nos pontos  $x_k, x_{k+1}$ , é definido da seguinte forma:

$$f[x_k, x_{k+1}] = \frac{f[x_k] - f[x_{k+1}]}{x_k - x_{k+1}}$$

Este valor pode ser interpretado como uma aproximação para a primeira derivada de  $f(x)$ , em  $x_k$ . O operador de diferença dividida de ordem dois, nos pontos  $x_k, x_{k+1}, x_{k+2}$ , é definido da seguinte forma:

$$f[x_k, x_{k+1}, x_{k+2}] = \frac{f[x_k, x_{k+1}] - f[x_{k+1}, x_{k+2}]}{x_k - x_{k+2}}.$$

De forma análoga, definimos o operador diferença dividida de ordem  $n$ , nos pontos  $x_k, x_{k+1}, \dots, x_{k+n}$ , da seguinte forma:

$$f[x_k, x_{k+1}, \dots, x_{k+n}] = \frac{f[x_k, x_{k+n-1}] - f[x_{k+1}, x_{k+n}]}{x_k - x_{k+n}}.$$

Note que a forma de cálculo desses operadores é construtiva, no sentido de que para obter a diferença dividida de ordem  $n$  necessitamos das diferenças divididas de ordem  $n - 1, n - 2, \dots, 1, 0$ . Um esquema prático para o cálculo desses operadores é dado pela tabela abaixo

$x$	$f[x_k]$	$f[x_k, x_{k+1}]$	$f[x_k, x_{k+1}, x_{k+2}]$	$\dots$	$f[x_k, x_{k+1}, \dots, x_{k+n}]$
$x_0$	$f_0$				
$x_1$	$f_1$	$> \frac{f_0 - f_1}{x_0 - x_1}$	$> \frac{f[x_0, x_1] - f[x_1, x_2]}{x_0 - x_2}$		
$x_2$	$f_2$	$> \frac{f_1 - f_2}{x_1 - x_2}$	$> \frac{f[x_1, x_2] - f[x_2, x_3]}{x_1 - x_3}$		
$x_3$	$f_3$	$> \frac{f_2 - f_3}{x_2 - x_3}$			$> \frac{f[x_0, \dots, x_{n-1}] - f[x_1, \dots, x_n]}{x_0 - x_n}$
$\vdots$	$\vdots$		$\vdots$		
$x_{n-1}$	$f_{n-1}$	$> \frac{f_{n-1} - f_n}{x_{n-1} - x_n}$			
$x_n$	$f_n$				

### 5.2.1 Construção do Polinômio

Vamos considerar o conjunto de pontos  $x_0, x_1, \dots, x_n$ , onde conhecemos os valores da função  $f(x)$ , dados por  $f_0, f_1, \dots, f_n$ . Calculando a diferença dividida de ordem dois entre os pontos  $x, x_0, x_1$  temos

$$f[x, x_0, x_1] = \frac{f[x, x_0] - f[x_0, x_1]}{x - x_1}$$

Isolando a diferença de ordem um que depende de  $x$  segue que

$$f[x, x_0] = f[x_0, x_1] + (x - x_1)f[x, x_0, x_1]$$

Aplicamos a definição de diferença de ordem um no primeiro termo, assim segue que

$$\frac{f(x) - f(x_0)}{x - x_0} = f[x_0, x_1] + (x - x_1)f[x, x_0, x_1],$$

e isto implica que

$$f(x) = f(x_0) + x - x_0 f[x_0, x_1] + (x - x_0)(x - x_1)f[x, x_0, x_1] = p_1(x) + E_1(x).$$

Ou seja a função  $f(x)$  é igual a um polinômio de grau um,  $p_1(x)$ , mais uma função  $E_1(x)$  que depende da diferença dividida de ordem dois. Desta forma podemos dizer que a função  $f(x)$  é aproximada por  $p_1(x)$  com erro de  $E_1(x)$ . O polinômio  $p_1(x)$  é o polinômio interpolador de  $f(x)$ , nos pontos  $x_0, x_1$ , pois,

$$\begin{aligned} p_1(x_0) &= f(x_0) + (x_0 - \cancel{x_0})f[x_0, x_1] + (x_0 - \cancel{x_0})(x - x_1)f[x, x_0, x_1] \\ &= f(x_0) \end{aligned}$$

$$\begin{aligned} p_1(x_1) &= f(x_0) + (x_1 - x_0)f[x_0, x_1] + (x_1 - x_0)(x - \cancel{x_1})f[x, x_0, x_1] \\ &= f(x_0) + (x_1 - x_0)\frac{f(x_0) - f(x_1)}{x_0 - x_1} \\ &= f(x_1) \end{aligned}$$

De forma análoga, podemos calcular a diferença dividida de ordem  $n$ , sobre os pontos  $x, x_0, x_1, \dots, x_n$ , obtendo

$$f(x) = p_n(x) + E_n(x),$$

onde

$$p_n(x) = f(x_0) + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2] + \dots + (x - x_0)(x - x_1) \cdots (x - x_{n-1})f[x_0, x_1, \dots, x_n] \quad (5.1)$$

$$E_n(x) = (x - x_0)(x - x_1) \cdots (x - x_n)f[x, x_0, x_1, \dots, x_n] \quad (5.2)$$

Assim podemos aproximar  $f(x)$  por  $p_n(x)$ , sendo que o erro é dado por  $E_n(x)$ . O polinômio  $p_n(x)$  é o polinômio interpolador de  $f(x)$  sobre os pontos  $x_0, x_1, \dots, x_n$ , pois  $p(x_j) = f(x_j)$ , para  $j = 0, 1, \dots, n$ .

**Exemplo 5.2.1** *Vamos considerar a função  $f(x)$  tabelada abaixo.*

$x$	0	0.5	1	1.5
$f(x)$	0.0000	1.1487	2.7183	4.9811

Montando a tabela das diferenças divididas temos

$x_k$	$f[x_k]$	$f[x_k, x_{k+1}]$	$f[x_k, \dots, x_{k+2}]$	$f[x_k, \dots, x_{k+3}]$
0.0	0.0000			
		<u>2.2974</u>		
0.5	1.1487		<u>0.8418</u>	
		3.1392		<u>0.36306</u>
1.0	2.7183		1.3864	
		4.5256		
1.5	4.9811			

Através da equação (5.1) podemos notar que as diferenças divididas que necessitamos são as primeiras de cada coluna. Logo polinômio interpolador é dado por

$$\begin{aligned}
 p(x) &= f(x_0) + x - x_0 f[x_0, x_1] + (x - x_0)(x - x_1) f[x_0, x_1, x_2] + (x - x_0)(x - x_1)(x - x_2) f[x_0, x_1, x_2, x_3] \\
 &= 0.00 + (x - 0.0)2.2974 + (x - 0.0)(x - 0.5)0.8418 + (x - 0.0)(x - 0.5)(x - 1.0)0.36306 \\
 &= 2.05803x + 0.29721x^2 + 0.36306x^3
 \end{aligned}$$

### 5.3 Estudo do Erro

A equação (5.2) representa o erro cometido na interpolação sobre os pontos  $x_0, \dots, x_n$ . Se aproximamos  $f(\bar{x}) \approx p_n(\bar{x})$  o erro será dado por  $E_n(\bar{x})$ . Porém este depende da diferença dividida  $f[\bar{x}, x_0, x_1, \dots, x_n]$ , que por sua vez, depende do valor de  $f(\bar{x})$ . Como a função  $f(x)$  é tabelada, não temos como calcular este valor. Estimativas para o erro podem ser obtidas se conhecemos algumas propriedades da função.

**Teorema 5.3.1** *Sejam  $x_0, x_1, \dots, x_n$ ,  $n + 1$  pontos distintos. Seja  $p_n(x)$  o polinômio interpolador sobre  $x_0, x_1, \dots, x_n$ . Se  $f(x)$  é  $n + 1$  vezes diferenciável em  $[x_0, x_n]$  então para  $\bar{x} \in [x_0, x_n]$  o erro é dado por:*

$$E_n(\bar{x}) = f(\bar{x}) - p_n(\bar{x}) = (\bar{x} - x_0)(\bar{x} - x_1) \cdots (\bar{x} - x_n) \frac{f^{(n+1)}(\xi)}{(n + 1)!} \quad \text{com } \xi \in [x_0, x_n]$$

Prova: Seja  $G(x) = (x - x_0)(x - x_1) \cdots (x - x_n)$ , logo  $G(x_i) = 0$  para  $i = 0, 1, \dots, n$ .  
 Seja  $H(t) = E_n(x)G(t) - E_n(t)G(x)$ , logo  $H$  satisfaz

**1:**  $H(t)$  possui derivadas até ordem  $n + 1$ , pois  $G$  e  $E_n$  possuem derivadas até esta ordem.

**2:**  $H(t)$  possui pelo menos  $(n + 2)$  zeros em  $[x_0, x_n]$ , pois para  $t = x_i$  temos

$$H(x_i) = E_n(x)G(x_i) - E_n(x_i)G(x) = 0$$

e para  $t = x$  temos  $H(x) = E_n(x)G(x) - E_n(x)G(x) = 0$ .

**3:** Aplicando o Teorema de Rolle a  $H(t)$  e suas derivadas até ordem  $n + 1$ , temos

$$\begin{array}{ll} H(t) & \text{tem } n + 2 \text{ zeros em } [x_0, x_n] \\ H'(t) & \text{tem } n + 1 \text{ zeros em } [x_0, x_n] \\ H''(t) & \text{tem } n \text{ zeros em } [x_0, x_n] \\ \vdots & \vdots \\ H^{(n+1)} & \text{tem } 1 \text{ zeros em } [x_0, x_n] \end{array}$$

Por outro lado temos que

$$H^{(n+1)}(t) = E_n(x)G^{(n+1)}(t) - E_n^{(n+1)}(t)G(x)$$

onde,

$$\begin{aligned} E_n^{(n+1)}(t) &= f^{(n+1)}(t) - p_n^{(n+1)}(t) \\ G^{(n+1)}(t) &= (n + 1)! \end{aligned}$$

Como o polinômio  $p_n$  é de grau  $n$  temos que  $p_n^{(n+1)}(t) = 0$  e segue que

$$H^{(n+1)}(t) = E_n(x)(n + 1)! - f^{(n+1)}(t)G(x)$$

A função  $H^{(n+1)}(t)$  possui um zero em  $[x_0, x_n]$  que vamos chamar de  $\xi$ . Substituindo na equação acima temos que

$$E_n(x) = (x - x_0)(x - x_1) \cdots (x - x_n) \frac{f^{(n+1)}(\xi)}{(n + 1)!}$$

■

Na prática usamos um limitante para o erro, sendo

$$|E_n(x)| \leq |(x - x_0)(x - x_1) \cdots (x - x_n)| \max_{x \in [x_0, x_n]} \frac{|f^{(n+1)}(x)|}{(n + 1)!},$$

onde temos que ter alguma informação sobre a função que permita limitar sua derivada de ordem  $n + 1$ .

## 5.4 Escolha dos Pontos

Uma das características da interpolação é que esta pode fornecer uma aproximação local, sem a necessidade de usar todos os dados disponíveis. Como exemplo, vamos considerar a tabela abaixo

$x$	0.2	0.34	0.4	0.52	0.6	0.72
$f(x)$	0.16	0.22	0.27	0.29	0.32	0.37

Vamos achar uma aproximação para  $f(0.44)$ , usando um polinômio de grau 2. Neste caso, necessitamos de 3 pontos e o ideal é escolher aqueles que estão mais próximos do valor que desejamos aproximar. Logo a melhor escolha será  $x_0 = 0.34, x_1 = 0.4$  e  $x_2 = 0.52$ . Isto se justifica pela fórmula do erro, pois

$$|E_n(0.44)| \leq |(0.44-0.34)(0.44-0.4)(0.44-0.52)| \max_{x \in [x_0, x_2]} \frac{|f^{(n+1)}(x)|}{(n+1)!} = 0.00032 \max_{x \in [x_0, x_2]} \frac{|f^{(n+1)}(x)|}{(n+1)!}$$

Se tivéssemos escolhido  $x_0 = 0.2$  e  $x_2 = 0.72$ , o erro estaria limitado por

$$|E_n(0.44)| \leq 0.00268 \max_{x \in [x_0, x_2]} \frac{|f^{(n+1)}(x)|}{(n+1)!}.$$

## 5.5 Interpolação Inversa

Considere o seguinte problema:

*Dada uma tabela de pontos  $(x_k, f_k)$  e um número  $y \in [f_0, f_n]$ . Desejamos achar o valor de  $x$  de tal forma que  $f(x) = y$ .*

Temos duas formas de resolver o problema: Obter o polinômio interpolador de  $f(x)$  e resolver a equação  $p_n(x) = y$ . Em geral a equação  $p_n(x) = y$  tem mais de uma solução e se o grau do polinômio for maior que 2, não temos um procedimento analítico que determine as soluções. A outra forma de se achar  $x$  é fazer uma interpolação inversa. Se  $f(x)$  é inversível num intervalo contendo  $y$ , então interpolamos a função inversa, isto é consideramos o conjunto de dados  $x = f^{-1}(y)$  e achamos o polinômio interpolador de  $f^{-1}(y)$ .

A condição para que a função seja inversível é que esta seja monótona crescente ou decrescente. Em termos dos pontos tabelados isto significa que os pontos devem satisfazer  $f_0 < f_1 < \dots < f_n$  ou  $f_0 > f_1 > \dots > f_n$

**Exemplo 5.5.1** *Dada a tabela de pontos*

$x$	0.2	0.3	0.4	0.5	0.6	0.7	0.8
$f(x)$	0.587	0.809	0.951	1.000	0.951	0.809	0.587

Vamos procurar uma aproximação de  $x$  de tal forma que  $f(x) = 0.9$ , usando uma interpolação quadrática na forma de Newton.

Em primeiro lugar devemos determinar em que intervalo pode ocorrer  $f(x) = 0.9$ . Neste exemplo temos duas possibilidades, para  $x \in [0.3, 0.4]$  ou  $x \in [0.6, 0.7]$ . Em segundo lugar devemos verificar se a função  $f(x)$  admite inversa. Para o primeiro caso temos que a função  $f(x)$  é crescente no intervalo  $[0.2, 0.5]$ . Logo esta admite inversa neste intervalo. No segundo caso a função admite inversa no intervalo  $[0.5, 0.8]$ , pois esta é decrescente neste intervalo. Como desejamos uma interpolação quadrática temos que ter no mínimo três pontos e nos dois casos temos quatro pontos. Portanto é possível achar as duas aproximações. Vamos nos concentrar no primeiro caso. Montando a tabela da função inversa temos

$y$	0.587	0.809	0.951	1.000
$f^{-1}(y)$	0.2	0.3	0.4	0.5

Como desejamos um polinômio de grau 2, devemos escolher três pontos e a melhor escolha são os pontos que estão mais próximos do valor a ser aproximado,  $y = 0.9$  ( $x_0 = 0.809, x_1 = 0.951$  e  $x_2 = 1.000$ ). Calculando as diferenças divididas temos

$y$	$f^{-1}$	ordem 1	ordem 2
0.809	0.3		
		0.704	
0.951	0.4		6.999
		2.040	
1.000	0.5		

e conseqüentemente o polinômio é dado por

$$\begin{aligned} p(y) &= 0.3 + (y - 0.809)0.704 + (y - 0.951)(y - 0.809)6.999 \\ &= 5.115 - 11.605y + 6.994y^2 \end{aligned}$$

Portanto o valor de  $x$  tal que  $f(x) = 0.9$  é aproximado por  $x = f^{-1}(0.9) \approx p(0.9) = 0.3315$ .

## 5.6 Observações Finais

Como no caso do ajuste de curvas, a interpolação aproxima uma função, sobre um conjunto de dados. Porém a interpolação exige que os pontos sejam distintos. Fato que não precisa ocorrer com o ajuste de curvas. Além disso, o grau do polinômio interpolador depende da quantidade de pontos. Logo para um conjunto de 100 pontos teremos um polinômio de grau  $\leq 99$ , o que não é muito prático se desejamos montar um modelo matemático. A interpolação é mais indicada para aproximações quantitativas, enquanto que o ajuste de curvas é indicado para uma aproximações qualitativas. Se desejamos saber a taxa de variação de uma determinada função (ou seja a derivada), o mais indicado é o ajuste de curvas, pois na interpolação não temos garantia de que  $f'(x) \approx p'_n(x)$  (Veja figura 5.1).

Se a função que estamos interpolando, sobre  $n + 1$  pontos é um polinômio de grau  $\leq n$ , então o polinômio interpolador é a própria  $f(x)$ . Isto pode ser verificado pela fórmula do erro, onde o termo  $f^{(n+1)}(\xi) = 0 \forall \xi \in [x_0, x_n]$ .

A interpolação é mais indicada para aproximações quantitativas, enquanto que o ajuste de curvas é indicado para uma aproximação qualitativa.

A medida que aumentamos a quantidade de pontos num intervalo  $[a, b]$ , ocorre o fenômeno de Runge, que é caracterizado em aumentar o erro nos pontos extremos do intervalo e melhorar a aproximação nos pontos centrais. Isto é justificado pela fórmula do erro, em que os pontos extremos do intervalo faz com que o fator  $(x - x_0) \cdots (x - x_n)$  seja grande. Desta forma, o polinômio interpolador não é indicado para extrapolar valores, isto é aproximar valores que não pertencem ao intervalo  $[x_0, x_n]$ . Abaixo apresentamos um exemplo implementado no MatLab, onde a figura 5.1 mostra o fenômeno de Runge.

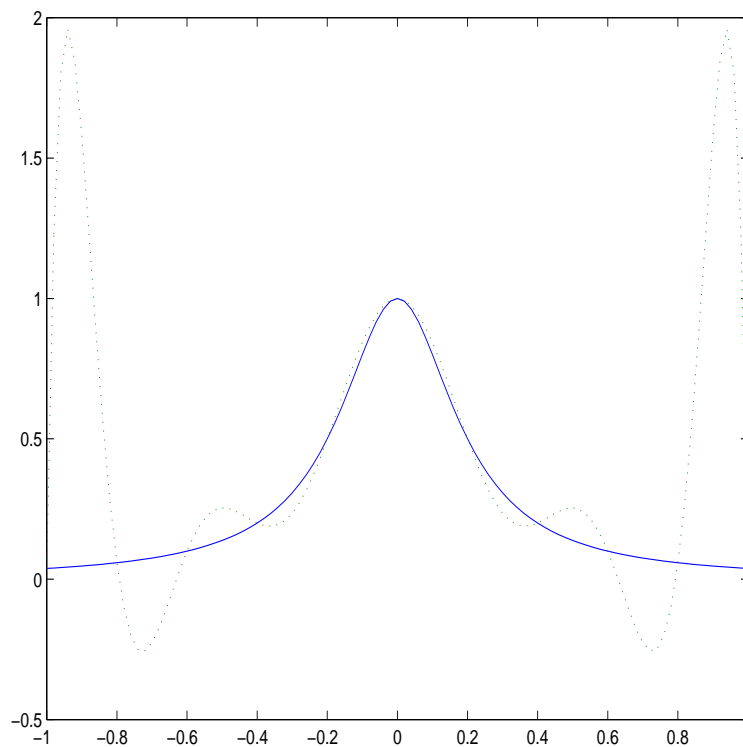


Figura 5.1: Fenômeno de Runge. - - -  $p_n(x)$ , —  $f(x)$

```
% Disciplina de Calculo Numerico - Prof. J. E. Castilho
% Forma de Lagrange do Pol. Interpolador
% Interpola a funcao f(x)=1/(1+25x^2) nos pontos
% x=[-1.0,-0.8,-0.6,...,0.6,0.8,1.0]
% Calcula o polinomio e a funcao nos pontos
```



```

% xf=[-1.0,-0.98,-0.96,...,0.96,0.98,1.0]
% Compara os graficos e mostra o fenomeno de Runge
%
clear;
xf=-1:0.02:1;
f=1./(1+25 *xf.^2);
x=-1:0.2:1;
fk=1./(1+25 *x.^2);

% Forma de Lagrange
n=size(xf); % pontos onde vamos calcular o polinomio
m=size(x); % pontos de interpolacao

for s=1:n(2)
    p(s)=0;
    for l=1:m(2)
        L=1;
        for k=1:l-1
            L=L*(xf(s) -x(k))/(x(l)-x(k));
        end;
        for k=l+1:m(2)
            L=L*(xf(s) -x(k))/(x(l)-x(k));
        end;
        p(s)=p(s)+fk(l)*L;
    end;
end;
plot(xf,f,xf,p,':');
print -depsc fig51.eps

```

## 5.7 Exercícios

**Exercício 5.1** *A tabela abaixo fornece o número de habitantes do Brasil (em milhões) de 1900 a 1970.*

<i>ano</i>	<i>1900</i>	<i>1920</i>	<i>1940</i>	<i>1950</i>	<i>1960</i>	<i>1970</i>
<i>Hab.</i>	<i>17.4</i>	<i>30.6</i>	<i>41.2</i>	<i>51.9</i>	<i>70.2</i>	<i>93.1</i>

- Usando o polinômio interpolador de grau 2, na forma de Lagrange, ache uma aproximação para a população no ano de 1959.*
- Usando interpolação quadrática na forma de Newton, estime, com o menor erro possível, em que ano a população ultrapassou os 50 milhões.*

- c) Com os resultados obtidos no ítem a) podemos estimar a taxa de crescimento da população neste período? Justifique sua resposta.

**Exercício 5.2** Considere a função  $f(x) = \sqrt{x}$  e os pontos  $x_0 = 16$ ,  $x_1 = 25$  e  $x_2 = 36$ . Com que precisão podemos calcular  $\sqrt{20}$ , usando interpolação sobre estes pontos?

**Exercício 5.3** Considere o problema de interpolação linear para  $f(x) = \sin(x) + x$ , usando os pontos  $x_0$  e  $x_1 = x_0 + h$ . Mostre que  $|E_1(x)| \leq h^2/8$ .

**Exercício 5.4** Dada a tabela abaixo.

$x$	-0.2	-0.1	0.1	0.15	0.35
$f(x)$	0.980	0.995	0.996	0.988	0.955

- a) Quando possível, ache uma aproximação para  $f(-0.25)$  e  $f(0)$ , usando o polinômio interpolador na forma de Newton, com o menor erro possível.
- b) Se tivéssemos usado o polinômio interpolador na forma de Lagrange sobre os mesmos pontos obteríamos melhor resultado? Justifique.

**Exercício 5.5** Num experimento de laboratório, uma reação química liberou calor de acordo com as medições mostradas na tabela abaixo

hora	8:00 hr	9:00 hr	9:30 hr	10:00 hr
$C^\circ$	0	130	210	360

- a) Determine uma função que dê a temperatura em função do tempo, de modo que os pontos tabelados sejam representados sem erro.
- b) Calcule a provável temperatura ocorrida às 9:45 hr.

# Capítulo 6

## Integração Numérica - Fórmulas de Newton Côtes

O objetivo deste capítulo é estudar esquemas numéricos que aproxime a integral definida de uma função  $f(x)$  num intervalo  $[a, b]$ . A integração numérica é aplicada quando a primitiva da função não é conhecida ou quando só conhecemos a função  $f(x)$  num conjunto discreto de pontos.

As fórmulas de Newton-Côtes são baseadas na estratégia de aproximar a função  $f(x)$  por um polinômio interpolador e aproximamos a integral pela integral do polinômio. As aproximações são do tipo

$$\int_a^b f(x)dx \approx A_0f(x_0) + A_1f(x_1) + \cdots + A_nf(x_n) = \sum_{i=0}^n A_i f(x_i)$$

onde os pontos são igualmente espaçados, isto é  $x_k = x_0 + kh$ , com  $h = (b - a)/n$ , e os coeficientes  $A_i$  são determinado pelo polinômio escolhido para aproximar a função  $f(x)$ .

### 6.1 Regra do Trapézio

A Regra do Trapézio é a fórmula de Newton-Côtes que aproxima a função  $f(x)$  pelo polinômio interpolador de grau 1 sobre os pontos  $x_0 = a$  e  $x_1 = b$ . O polinômio interpolador de grau um, na forma de Newton é dado por

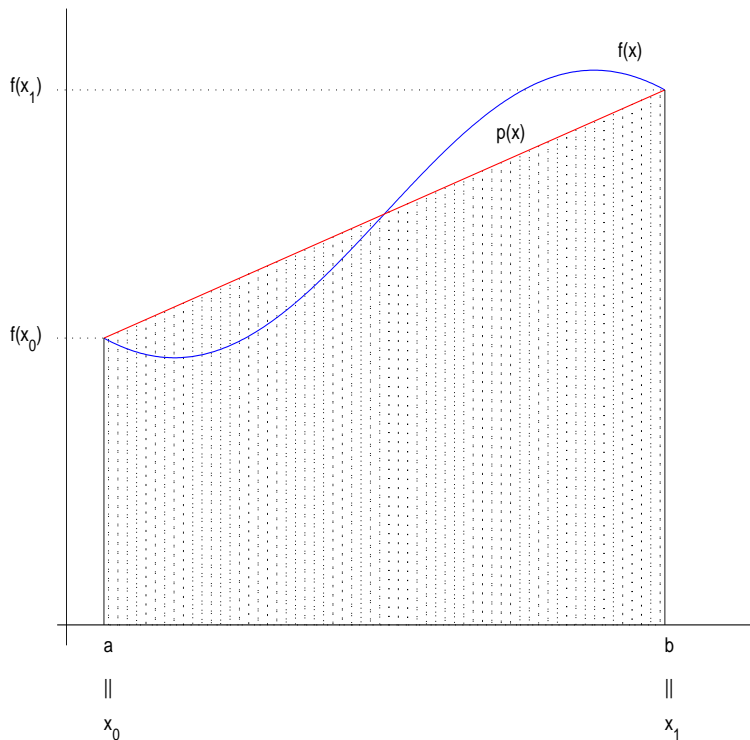
$$p_1(x) = f(x_0) + f[x_0, x_1](x - x_0).$$

Desta forma vamos aproximar a integral da função  $f(x)$  pela integral do polinômio, obtendo

$$\begin{aligned} \int_a^b f(x)dx &\approx \int_{x_0}^{x_1} p_1(x)dx \\ &= \int_{x_0}^{x_1} f(x_0) + f[x_0, x_1](x - x_0)dx \end{aligned}$$

$$\begin{aligned}
 &= f(x_0)(x_1 - x_0) + f[x_0, x_1] \left( \frac{x_1^2}{2} - \frac{x_0^2}{2} - x_0(x_1 - x_0) \right) \\
 &= f(x_0)(x_1 - x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0} \frac{(x_1 - x_0)^2}{2} \\
 &= \frac{(x_1 - x_0)}{2} (f(x_0) + f(x_1)) \\
 &= \frac{h}{2} (f(x_0) + f(x_1)),
 \end{aligned}$$

onde  $h = x_1 - x_0$ . A fórmula acima representa a área do trapézio que tem  $f(x_1)$  e  $f(x_0)$  como os valores das bases e  $h$  como o valor da altura. Na figura 6.1 temos uma representação desta aproximação.



## 6.2 Cálculo do Erro

No capítulo de interpolação vimos que uma função  $f(x)$  pode ser representada por

$$f(x) = p_n(x) + E_n(x),$$

onde  $p_n(x)$  é o polinômio interpolador e  $E_n(x)$  o erro na interpolação definido no Teorema 5.3.1. Calculando a integral da função  $f(x)$  no intervalo  $[a, b]$ , segue que

$$\int_a^b f(x)dx = \int_a^b p_n(x)dx + \int_a^b E_n(x)dx,$$

ou seja o erro na integração é dado pela integração do erro cometido na interpolação. No caso da Regra do Trapézio segue que o erro é dado por

$$E_T = \int_a^b E_1(x)dx = \int_a^b (x - x_0)(x - x_1)\frac{f''(\xi)}{2}dx = -\frac{h^3}{12}f''(\xi), \quad \xi \in [a, b]$$

Como o erro depende do ponto  $\xi$ , que é desconhecido, na prática usamos a estimativa

$$|E_T| \leq \frac{h^3}{12} \max_{\xi \in [a, b]} |f''(\xi)|$$

**Exemplo 6.2.1** Como exemplo, vamos considerar a função  $f(x) = e^{-x^2}$ , cuja a primitiva não tem uma forma analítica conhecida. Vamos aproximar a integral no intervalo  $[0, 1]$  usando a Regra do Trapézio. Desta forma temos que  $h = 1 - 0 = 1$  e segue que

$$\int_0^1 e^{-x^2} dx \approx \frac{1}{2}(e^{-0^2} + e^{-1^2}) = 0.6839397$$

Para calcular o erro cometido temos que limitar a segunda derivada da função no intervalo  $[0, 1]$ . Sendo

$$f''(x) = (4x^2 - 2)e^{-x^2}$$

temos que nos extremos do intervalo vale

$$|f''(0)| = 2 \quad e \quad |f''(1)| = 0.735759.$$

Para calcular os pontos críticos da  $f''(x)$ , devemos derivar  $f''(x)$  e igualar a zero, obtendo,

$$f'''(x) = (12x - 8x^3)e^{-x^2} = 0 \Leftrightarrow x = 0 \text{ ou } x = \pm\sqrt{\frac{3}{2}}$$

Como o único ponto crítico pertencente ao intervalo é  $x = 0$  segue que

$$\max_{\xi \in [a, b]} |f''(\xi)| = 2$$

Com isto temos que

$$E_T \leq \frac{h^3}{12} \max_{\xi \in [a, b]} |f''(\xi)| = \frac{1}{6} = 0.166667$$

Note que esta estimativa do erro informa que a aproximação obtida não tem garantida nem da primeira casa decimal como exata. Logo a solução exata da integral está entre os valores  $0.6839397 \pm 0.166667$ .

### 6.3 Regra do Trapézio Repetida

A Regra do Trapézio aproxima bem funções suaves ( $|f'(x)| \ll 1$ ) e/ou em intervalos de integração de amplitude pequena. Para intervalos de amplitude grande podemos fazer uma subdivisão do intervalo de integração  $[a, b]$  em  $n$  subintervalos de mesma amplitude e aplicamos a Regra do Trapézio em cada subintervalo (Ver Figura 6.1). Neste caso temos que

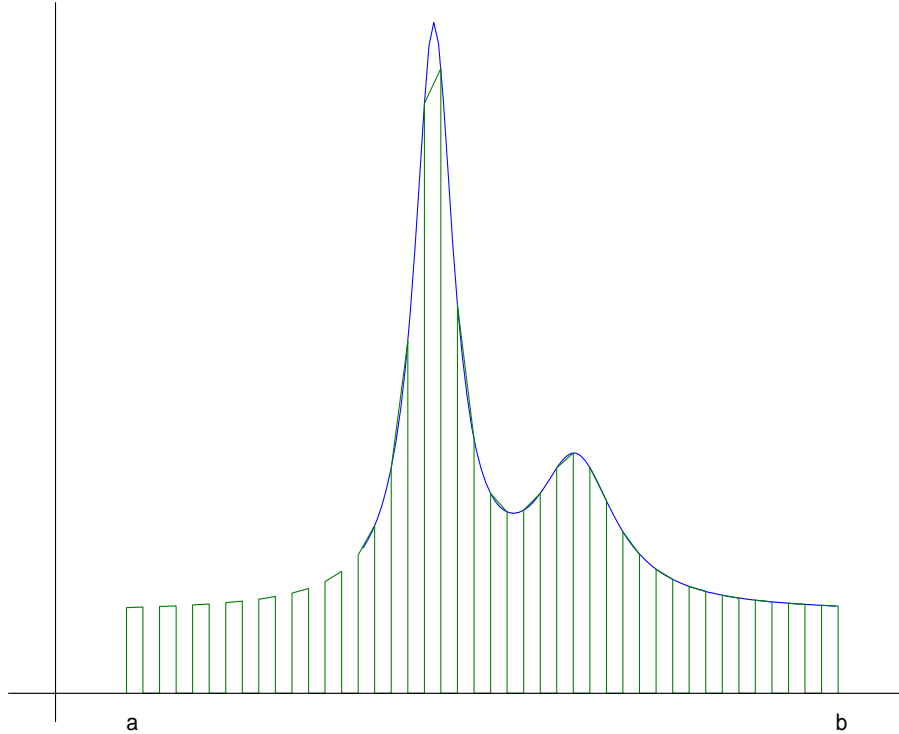


Figura 6.1: Regra do Trapézio Repetida

$$h = \frac{b-a}{n} \text{ e } x_k = x_0 + kh \text{ com } k = 0, 1, \dots, n$$

Aplicando a Regra do Trapézio em cada subintervalo  $[x_k, x_{k+1}]$  segue que

$$\begin{aligned} \int_a^b f(x) dx &\approx \frac{h}{2}(f_0 + f_1) + \frac{h}{2}(f_1 + f_2) + \frac{h}{2}(f_2 + f_3) + \dots + \frac{h}{2}(f_{n-2} + f_{n-1}) + \frac{h}{2}(f_{n-1} + f_n) \\ &= \frac{h}{2} [f_0 + 2(f_1 + f_2 + \dots + f_{n-1}) + f_n] \end{aligned}$$

O erro cometido na aproximação é igual a soma dos erros cometidos em cada subintervalo, logo temos que o erro é da forma

$$|E_{TG}| \leq n \frac{h^3}{12} \max_{\xi \in [a,b]} |f''(\xi)|$$

**Exemplo 6.3.1** Considerando a função do exemplo anterior,  $f(x) = e^{-x^2}$  em  $[0, 1]$ , vamos determinar o número de subintervalos necessários para que a Regra do Trapézio Repetida forneça uma aproximação com pelo menos 3 casas decimais exatas. Para isto devemos ter que  $|E_{TG}| \leq 10^{-4}$ , logo

$$n \frac{h^3}{12} \max_{\xi \in [0,1]} |f''(\xi)| \leq 10^{-4}$$

Sendo  $h = (b - a)/n$  e que o máximo da segunda derivada da função em  $[0, 1]$  é 2 (Ver ex. anterior) segue que

$$\begin{aligned} n \frac{h^3}{12} \max_{\xi \in [0,1]} |f''(\xi)| \leq 10^{-4} &\Leftrightarrow n \frac{1}{n^3} \frac{1}{12} 2 \leq 10^{-4} \\ &\Leftrightarrow \frac{1}{n^2} \leq 6 \times 10^{-4} \\ &\Leftrightarrow n^2 \geq 1666.666 \\ &\Leftrightarrow n \geq 40.824 \end{aligned}$$

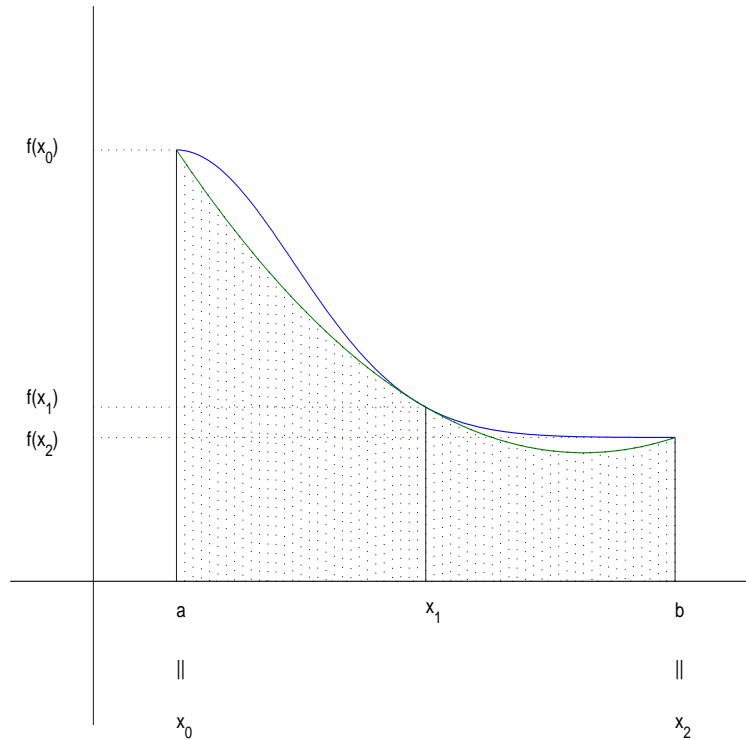
Devemos tomar no mínimo  $n = 41$  subintervalos para atingir a precisão desejada. Abaixo apresentamos o uso da função `trapz.m` do MatLab que fornece a aproximação da integral pela Regra do Trapézio. Usando 41 subintervalos temos a aproximação  $It = 0.746787657$

```
% Disciplina de Calculo Numerico - Prof. J. E. Castilho
% Exemplo do uso da funcao trapz(x,y)
% Calcula a integral usando a regra do trapezio
% para os pontos
% x=[x0,x1,...,xn]
% f=[fo,f1,...,fn]
%
h=1/41;
x=0:h:1;
f=exp(-x.^2);
It=trapz(x,f)
```

## 6.4 Regra de Simpson 1/3

Neste caso, usamos o polinômio de grau 2 que interpola a função  $f(x)$ . Para isto necessitamos de três pontos  $x_0, x_1$  e  $x_2$ . Como os pontos devem ser igualmente espaçados, tomamos  $h = (b - a)/2$ . Na figura 6.4 temos uma representação desta aproximação. Para obter a aproximação da integral vamos considerar o polinômio interpolador na forma de Newton,

$$p_2(x) = f(x_0) + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2].$$



E com isto segue que a integral da função  $f(x)$  é aproximada por

$$\begin{aligned} \int_a^b f(x)dx &\approx \int_{x_0}^{x_2} p_2(x)dx \\ &= \frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2)), \end{aligned}$$

De forma análoga a Regra do Trapézio, obtemos a fórmula do erro, integrando o erro cometido na interpolação. Desta forma obtemos

$$E_S = \int_a^b E_2(x)dx = \int_a^b (x - x_0)(x - x_1)(x - x_2) \frac{f'''(\xi)}{3!} dx = -\frac{h^5}{90} f^{(iv)}(\xi), \quad \xi \in [a, b]$$

Na prática usamos a estimativa para o erro

$$|E_S| \leq \frac{h^5}{90} \max_{\xi \in [a, b]} |f^{(iv)}(\xi)|$$

**Exemplo 6.4.1** Vamos considerar a função do exemplo anterior:  $f(x) = e^{-x^2}$  no intervalo  $[0, 1]$ . Para usar a Regra de Simpson temos que ter três pontos. Desta forma tomamos

$$h = \frac{b - a}{2} = \frac{1 - 0}{2} = \frac{1}{2}$$



E com isto segue a aproximação é dada por:

$$\int_a^b f(x)dx \approx \frac{1}{6}(f(0) + 4f(1/2) + f(1)) = 0.74718$$

A limitação do erro depende da limitação da quarta derivada da função no intervalo  $[0, 1]$ , sendo:

$$f^{(iv)}(x) = (12 - 48x^2 + 16x^4)e^{-x^2}$$

Calculando nos extremos do intervalo temos

$$|f^{(iv)}(0)| = 12 \quad e \quad |f^{(iv)}(1)| = 0.735759$$

Calculando os pontos críticos temos

$$f^{(v)}(x) = (-32x^5 + 160x^3 - 120x)e^{-x^2} = 0 \Leftrightarrow x = 0 \text{ ou } x = \pm 0.958572 \text{ ou } x = \pm 2.02018$$

Como o ponto  $x = 0.958572$  pertence ao intervalo  $[0, 1]$  temos

$$|f^{(iv)}(0.958572)| = 7.41948$$

Assim temos que

$$\max_{\xi \in [a, b]} |f^{(iv)}(\xi)| = 12$$

Obtemos desta forma que

$$E_S \leq \frac{h^5}{90} \max_{\xi \in [a, b]} |f^{(iv)}(\xi)| = 0.00416667$$

Desta forma podemos garantir que as duas primeiras casas decimais estão corretas.

## 6.5 Regra de Simpson Repetida

Podemos melhorar a aproximação da mesma forma que fizemos com a Regra do Trapézio. Vamos dividir o intervalo de integração em  $n$  subintervalos de mesma amplitude. Porém devemos observar que a Regra de Simpson necessita de três pontos, logo o subintervalo que aplicaremos a fórmula será da forma  $[x_k, x_{k+2}]$  o que implica que  $n$  deve ser um número par. Neste caso temos que

$$h = \frac{b-a}{n} \quad e \quad x_k = x_0 + kh \quad k = 0, 1, \dots, n$$

Aplicando a Regra de Simpson em cada subintervalo  $[x_k, x_{k+2}]$  segue que

$$\begin{aligned} \int_a^b f(x)dx &\approx \frac{h}{3}(f_0 + 4f_1 + f_2) + \frac{h}{3}(f_2 + 4f_3 + f_4) + \dots + \frac{h}{3}(f_{n-4} + 4f_{n-3} + f_{n-2}) + \frac{h}{3}(f_{n-2} + 4f_{n-1} + f_n) \\ &= \frac{h}{3}[f_0 + 4(f_1 + f_3 + \dots + f_{n-1}) + 2(f_2 + f_4 + \dots + f_{n-2}) + f_n] \end{aligned}$$

O erro cometido nesta aproximação é a soma dos erros em cada subintervalo  $[x_k, x_{k+2}]$  e como temos  $n/2$  subintervalos segue que:

$$|E_{SR}| \leq n \frac{h^5}{180} \max_{\xi \in [a,b]} |f^{(iv)}(\xi)|$$

**Exemplo 6.5.1** Considerando a integral da função  $f(x) = e^{-x^2}$ , no intervalo  $[0, 1]$ , vamos determinar o número de subintervalos necessários para obtermos uma aproximação com três casas decimais exatas. Para isto limitamos o erro por  $|E_{SR}| \leq 10^{-4}$ . Sendo  $h = (b - a)/n$  e  $\max_{\xi \in [0,1]} |f^{(iv)}(\xi)| = 12$  segue que

$$\begin{aligned} n \frac{h^5}{180} \max_{\xi \in [a,b]} |f^{(iv)}(\xi)| \leq 10^{-4} &\Leftrightarrow n \frac{1}{n^5} \frac{12}{180} \leq 10^{-4} \\ &\Leftrightarrow \frac{1}{n^4} \leq 15 \times 10^{-4} \\ &\Leftrightarrow n^4 \geq 666.66666 \\ &\Leftrightarrow n \geq 5.0813274 \end{aligned}$$

O menor valor de  $n$  que garante a precisão é  $n = 6$ . Note que a Regra de Simpson necessita de bem menos subintervalos que a Regra do Trapézio ( $n = 41$ ).

## 6.6 Observações Finais

As fórmulas de Newton-Côtes são obtidas aproximando a função por um polinômio interpolador. Quanto maior for o grau do polinômio, mais preciso será o resultado obtido. Porém a estratégia das fórmulas repetidas permitem obter uma aproximação com uma certa precisão desejada, usando fórmulas que são obtidas por polinômios de baixo grau.

Pelas fórmulas de erro podemos observar que a Regra do Trapézio é exata para polinômios de grau um, enquanto que a Regra de Simpson é exata para polinômios de grau três.

## 6.7 Exercícios

**Exercício 6.1** Calcule as integrais pela Regra do Trapézio e pela Regra de Simpson usando seis subintervalos.

$$\int_1^4 \sqrt{x} dx \quad e \quad \int_0^{0.6} \frac{dx}{1+x}$$

**Exercício 6.2** Calcule o valor de  $\pi$  com três casas decimais exatas usando a relação

$$\frac{\pi}{4} = \int_0^1 \frac{dx}{1+x^2}$$

**Exercício 6.3** Mostre que se  $f'(x)$  é contínua em  $[a, b]$  e que  $f''(x) > 0, \forall x \in [a, b]$ , então a aproximação obtida pela Regra do Trapézio é maior que o valor exato da integral.

**Exercício 6.4** Dada a tabela abaixo, calcule a integral  $\int_0^{.15^{0.30}} f(x) dx$  com o menor erro possível.

$x$	0.15	0.22	0.26	0.30
$f(x)$	1.897	1.514	1.347	1.204

**Exercício 6.5** Baseado na Regra de Simpson, determine uma regra de integração para a integral dupla

$$\int_a^b \int_c^d f(x, y) dx dy$$

Aplique a regra para calcular uma aproximação para

$$\int_0^1 \int_0^1 x^2 + y^2 dx dy$$

# Capítulo 7

## Equações Diferenciais Ordinárias (P.V.I.)

Muitos dos modelos matemáticos nas áreas de mecânica dos fluidos, fluxo de calor, vibrações, reações químicas são representados por equações diferenciais. Em muitos casos, a teoria garante a existência e unicidade da solução, porém nem sempre podemos obter a forma analítica desta solução.

Neste capítulo vamos nos concentrar em analisar esquemas numéricos para solução de Problemas de Valor Inicial (P.V.I.), para equações diferenciais de primeira ordem. Isto é, achar a função  $y(x)$  tal que

$$\begin{cases} y' = f(x, y) \\ y(x_0) = y_0 \end{cases}$$

A aproximação de  $y(x)$  é calculada nos pontos  $x_1, x_2, x_3, \dots$ , onde  $x_k = x_0 + kh$ . O valor da função no ponto  $x_k$  é aproximado por  $y_k$ , que é obtido em função dos valores anteriores  $y_{k-1}, y_{k-2}, \dots, y_0$ . Com isto podemos classificar os métodos em duas classes:

**Métodos de Passo Simples:** São aqueles em que o cálculo de  $y_k$  depende apenas de  $y_{k-1}$ .

**Métodos de Passo Múltiplo:** São aqueles em que o cálculo de  $y_k$  depende  $m$ -valores anteriores,  $y_{k-1}, y_{k-2}, \dots, y_{k-m}$ . Neste caso dizemos que o método é de  $m$ -passos.

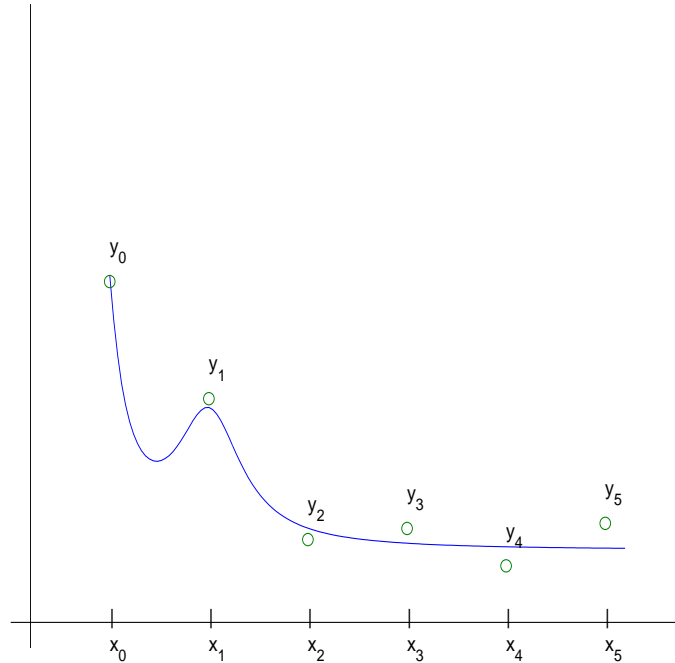
### 7.1 Método Euler

Dado o problema de valor inicial

$$\begin{cases} y' = f(x, y) \\ y(x_0) = y_0 \end{cases}$$

Uma forma de aproximar a derivada de uma função no ponto  $x_1$  é dado por

$$y'(x_0) = \lim_{h \rightarrow 0} \frac{y(x_0 + h) - y(x_0)}{h} \approx \frac{y(x_0 + h) - y(x_0)}{h} \quad (7.1)$$



Como  $x_1 = x_0 + h$  e pelo P.V.I. segue que

$$\frac{y_1 - y_0}{h} = f(x_0, y_0) \Rightarrow y_1 = y_0 + hf(x_0, y_0)$$

Com isto relacionamos o ponto  $y_1$  com  $y_0$ , um valor dado no P.V.I. Assim obtemos uma aproximação  $y(x_1) \approx y_1$ . De forma análoga podemos obter  $y_2$  em função de  $y_1$ , sendo que de uma forma geral teremos

$$y_{k+1} = y_k + hf(x_k, y_k)$$

Este método é conhecido como Método de Euler.

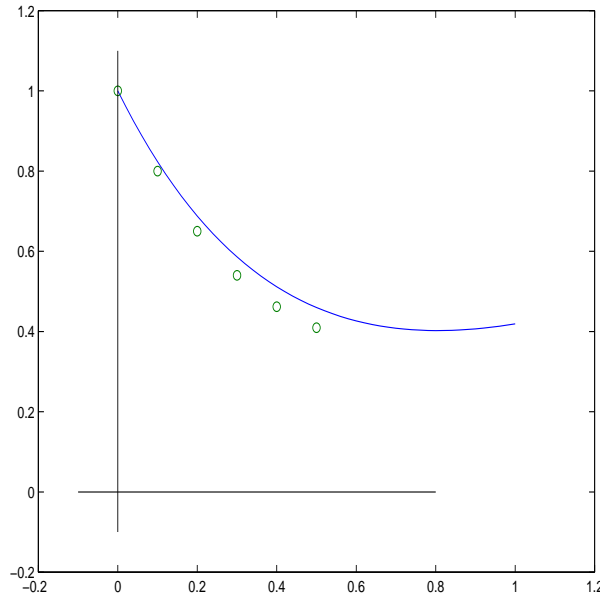
**Exemplo 7.1.1** Consideremos o seguinte problema de valor inicial

$$\begin{cases} y' = x - 2y \\ y(0) = 1 \end{cases}$$

Neste caso temos que  $x_0 = 0$  e  $y_0 = 1$ . Vamos usar o Método de Euler para obter uma aproximação para  $y(0.5)$ , usando  $h = 0.1$ . Desta forma temos

$$\begin{aligned} y_1 &= y_0 + h(x_0 - 2y_0) = 1 + 0.1(0 - 2 * 1) = 0.8 \approx y(x_1) = y(0.1) \\ y_2 &= y_1 + h(x_1 - 2y_1) = 0.8 + 0.1(0.1 - 2 * 0.8) = 0.65 \approx y(x_2) = y(0.2) \\ y_3 &= y_2 + h(x_2 - 2y_2) = 0.65 + 0.1(0.2 - 2 * 0.65) = 0.54 \approx y(x_3) = y(0.3) \\ y_4 &= y_3 + h(x_3 - 2y_3) = 0.54 + 0.1(0.3 - 2 * 0.54) = 0.462 \approx y(x_4) = y(0.4) \\ y_5 &= y_4 + h(x_4 - 2y_4) = 0.462 + 0.1(0.4 - 2 * 0.462) = 0.4096 \approx y(x_5) = y(0.5) \end{aligned}$$

A solução analítica do P.V.I. é dada por  $y(x) = (5e^{-2x} + 2x - 1)/4$ . No gráfico abaixo comparamos a solução exata com os valores calculados pelo Método de Euler. Note que em cada valor calculado o erro aumenta. Isto se deve porque cometemos um "erro local" na aproximação da derivada por (7.1) e este erro vai se acumulando a cada valor calculado. Na próxima seção daremos uma forma geral do erro local.



## 7.2 Métodos da Série de Taylor

Vamos considerar o problema de valor inicial

$$\begin{cases} y' = f(x, y) \\ y(x_0) = y_0 \end{cases}$$

Aplicando a série de Taylor para  $y(x)$  no ponto  $x_k$ , temos

$$y(x) = y(x_k) + \frac{y'(x_k)}{1!}(x-x_k) + \frac{y''(x_k)}{2!}(x-x_k)^2 + \dots + \frac{y^{(n)}(x_k)}{n!}(x-x_k)^n + \frac{y^{(n+1)}(\xi)}{(n+1)!}(x-x_k)^{n+1}$$

Calculando no ponto  $x_{k+1}$  e considerando que  $x_{k+1} - x_k = h$  temos que

$$y(x_{k+1}) = y(x_k) + \frac{y'(x_k)}{1!}h + \frac{y''(x_k)}{2!}h^2 + \dots + \frac{y^{(n)}(x_k)}{n!}h^n + \frac{y^{(n+1)}(\xi)}{(n+1)!}h^{n+1} \quad (7.2)$$

Como  $y'(x_k) = f(x_k, y_k)$  podemos relacionar as derivadas de ordem superior com as derivadas da função  $f(x, y)$ . Como exemplo consideremos

$$y''(x_k) = \frac{d}{dx}f(x_k, y_k)$$

$$\begin{aligned}
&= f_x + f_y y' \\
&= f_x + f_y f \\
y'''(x_k) &= f_y(f_x + f_y f) + f^2 f_{yy} + 2f f_{xy} + f_{xx}
\end{aligned}$$

Desta forma podemos obter uma aproximação para o cálculo do P.V.I., substituindo as relações do tipo acima na série de Taylor. Os métodos podem ser classificados de acordo com o termo de maior ordem que usamos na série de Taylor, sendo

**Definição 7.2.1** Dizemos que um método para a solução de P.V.I. é de ordem  $n$  se este coincide com a série de Taylor até o  $n$ -ésimo termo. O erro local cometido por esta aproximação será da forma

$$E_{loc}(x_{k+1}) = \frac{y^{(n+1)}(\xi)}{(n+1)!} h^{n+1} \quad \xi \in [x_k, x_{k+1}]$$

Como exemplo temos que o Método de Euler é um método de 1<sup>o</sup> ordem, pois este coincide com a série de Taylor até o primeiro termo, logo o erro local é dado por

$$E_{loc}(x_{k+1}) = \frac{y''(\xi)}{2!} h^2 \quad \xi \in [x_k, x_{k+1}]$$

Em geral, podemos determinar a ordem de um método pela fórmula do erro. Se o erro depende da  $n$ -ésima derivada dizemos que o método é de ordem  $n - 1$ .

**Exemplo 7.2.1** Vamos utilizar o método de 2<sup>o</sup> ordem para aproximar  $y(0.5)$ , usando  $h = 0.1$ , para o P.V.I.

$$\begin{cases} y' = x - 2y \\ y(0) = 1 \end{cases}$$

Neste caso temos que  $f(x, y) = x - 2y$ , logo segue que

$$y'' = f_x + f_y f = 1 - 2(x - 2y)$$

Substituindo em (7.2) temos que

$$\begin{aligned}
y(x_{k+1}) &= y(x_k) + \frac{y'(x_k)}{1!} h + \frac{y''(x_k)}{2!} h^2 \\
&= y(x_k) + h(x_k - 2y(x_k)) + \frac{h^2}{2}(1 - 2x_k + 4y(x_k))
\end{aligned}$$

Portanto o método é dado por

$$y_{k+1} = y_k + h(x_k - 2y_k) + \frac{h^2}{2}(1 - 2x_k + 4y_k)$$

Sendo  $x_0 = 0$  e  $y_0 = 1$  obtemos que

$$\begin{aligned}
 y_1 &= y_0 + h(x_0 - 2y_0) + \frac{h^2}{2}(1 - 2x_0 + 4y_0) \\
 &= 1 + 0.1(0 - 2 * 1) + \frac{0.1^2}{2}(1 - 2 * 0 + 4 * 1) = 0.825 \\
 y_2 &= y_1 + h(x_1 - 2y_1) + \frac{h^2}{2}(1 - 2x_1 + 4y_1) \\
 &= 0.825 + 0.1(0.1 - 2 * 0.825) + \frac{0.1^2}{2}(1 - 2 * 0.1 + 4 * 0.825) = 0.6905 \\
 y_3 &= y_2 + h(x_2 - 2y_2) + \frac{h^2}{2}(1 - 2x_2 + 4y_2) \\
 &= 0.6905 + 0.1(0.2 - 2 * 0.6905) + \frac{0.1^2}{2}(1 - 2 * 0.2 + 4 * 0.6905) = 0.58921 \\
 y_4 &= y_3 + h(x_3 - 2y_3) + \frac{h^2}{2}(1 - 2x_3 + 4y_3) \\
 &= 0.58921 + 0.1(0.3 - 2 * 0.58921) + \frac{0.1^2}{2}(1 - 2 * 0.3 + 4 * 0.58921) = 0.515152 \\
 y_5 &= y_4 + h(x_4 - 2y_4) + \frac{h^2}{2}(1 - 2x_4 + 4y_4) \\
 &= 0.515152 + 0.1(0.4 - 2 * 0.515152) + \frac{0.1^2}{2}(1 - 2 * 0.4 + 4 * 0.515152) = 0.463425
 \end{aligned}$$

O gráfico na figura 7.1 compara os resultados obtidos por este método, com os resultados obtidos pelo método de Euler.

Os resultados obtidos pelo método de 2<sup>o</sup> ordem são mais precisos. Quanto maior a ordem do método melhor será a aproximação. A dificuldade em se tomar métodos de alta ordem é o cálculo da relação de  $y^{(n+1)}(x) = [f(x, y)]^{(n)}$ .

### 7.3 Métodos de Runge-Kutta

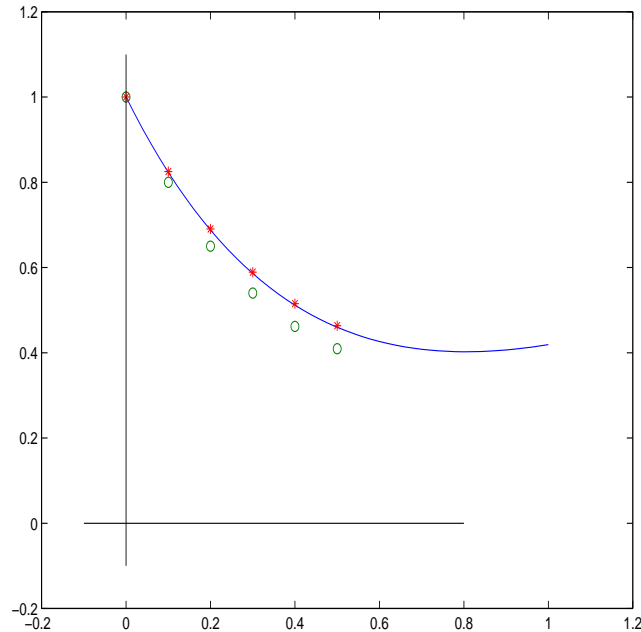
A estratégia dos métodos de Runge-Kutta é aproveitar as qualidades dos métodos da Série de Taylor (escolher a precisão) sem ter que calcular as derivadas totais de  $f(x, y)$ .

O método de Runge-Kutta de 1<sup>o</sup> ordem é o Método de Euler, que coincide com o método da Série de Taylor de 1<sup>o</sup> ordem. Vamos determinar um método de 2<sup>o</sup> ordem, conhecido como método de Euler Melhorado ou método de Heun. Como o próprio nome diz, a idéia é modificar o método de Euler de tal forma que podemos melhorar a precisão. Na figura 7.2-a temos a aproximação  $ye_{n+1} = y_n + hf(x_n, y_n)$  que é obtida pela aproximação do método de Euler.

Montamos a reta  $L_1$  que tem coeficiente angular dado por  $y'(x_n) = f(x_n, y_n)$ .

$$\begin{aligned}
 L_1(x) &= y_n + (x - x_n)y'_n = y_n + (x - x_n)f(x_n, y_n) \\
 L_1(x_{n+1}) &= y_n + (x_{n+1} - x_n)y'_n = y_n + hf(x_n, y_n) = ye_{n+1}
 \end{aligned}$$




 Figura 7.1: Método de Euler 'o' — Método de 2<sup>o</sup> ordem '\* '

Montamos a reta  $L_2$  com coeficiente angular dado por  $f(x_{n+1}, ye_{n+1}) = f(x_n, y_n + hy')$  e passa pelo ponto  $P$  ( Ver figura 7.2-b ).

$$L_2(x) = ye_{n+1} + (x - x_{n+1})f(x_{n+1}, ye_{n+1})$$

Montamos a reta  $L_0$  que passa por  $P$  e tem como coeficiente angular a média dos coeficientes angular de  $L_1$  e  $L_2$  (Ver figura 7.2-c). Finalmente a reta que passa pelo ponto  $(x_n, y_n)$  e é paralela a reta  $L_0$  tem a forma

$$L(x) = y_n + (x - x_n)\frac{1}{2}(f(x_n, y_n) + f(x_{n+1}, y_n + hy'_n))$$

Calculando no ponto  $x_{n+1}$  temos

$$y_{n+1} = y_n + h\frac{1}{2}(f(x_n, y_n) + f(x_{n+1}, y_n + hy'_n))$$

Podemos observar que o valor de  $y_{n+1}$  (Ver figura 7.2-d) está mais próximo do valor exato que o valor de  $ye_{n+1}$ . Este esquema numérico é chamado de método de Euler Melhorado, onde uma estimativa do erro local é dado por

$$|E_{loc}(x_n)| \leq \frac{h^3}{6} \max_{\xi \in [x_n, x_{n+1}]} |y'''(\xi)|$$

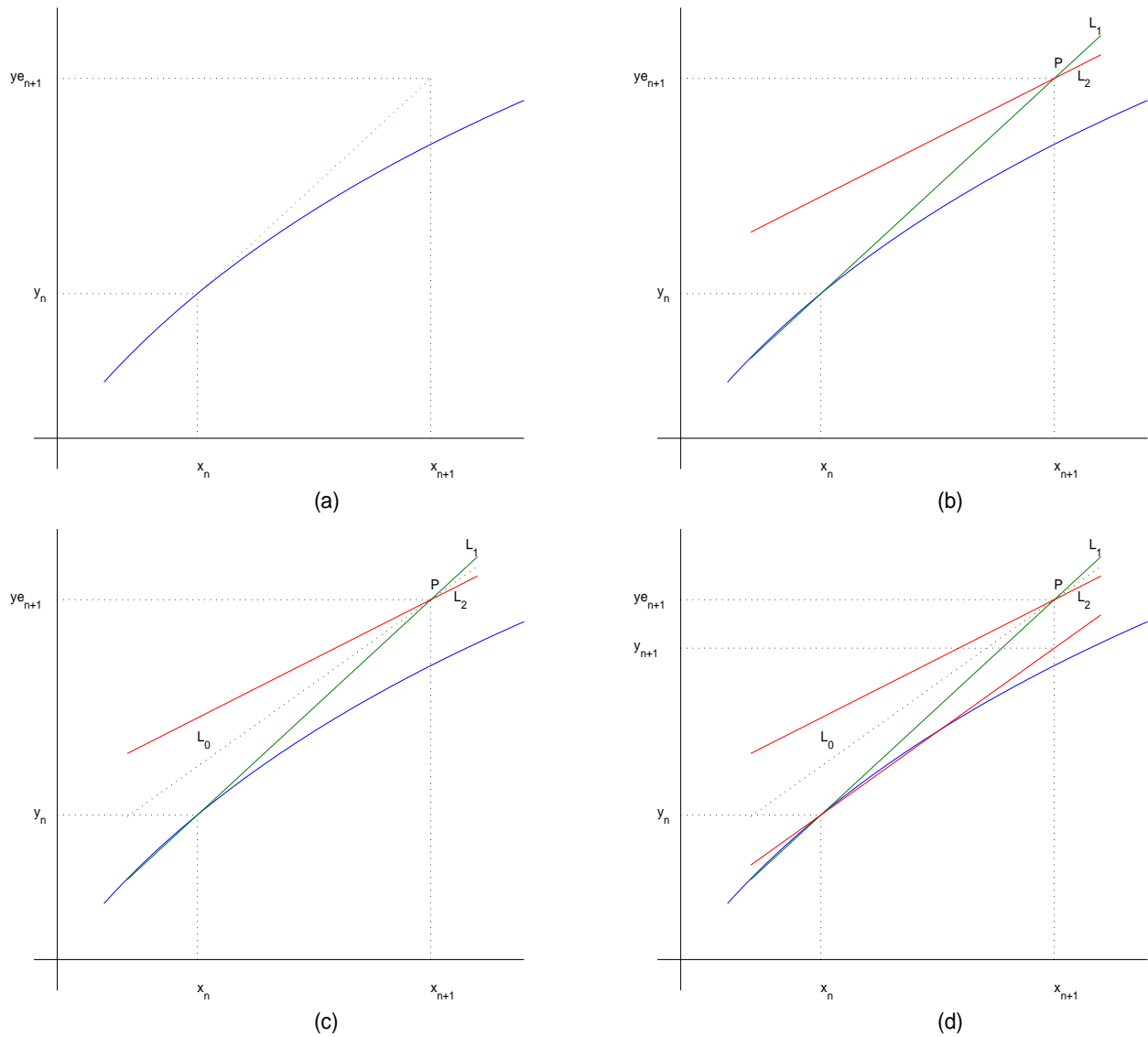


Figura 7.2: Método de Euler Melhorado

Determinamos o método de Euler Melhorado por uma construção geométrica. Também podemos obter uma demonstração analítica. Desenvolvemos a série de Taylor da função  $f(x, y)$  e calculando no ponto  $(x_{n+1}, y_n + hy'_n)$ . A expressão encontrada deve concordar com a Série de Taylor até a segunda ordem. Em geral um método de Runge-Kutta de segunda ordem é dado por

$$y_{n+1} = y_n + a_1 h f(x_n, y_n) + a_2 h f(x_n + b_1 h, y_n + b_2 h y'_n),$$

onde

$$\begin{cases} a_1 + a_2 = 1 \\ a_2 b_1 = 1/2 \\ a_2 b_2 = 1/2 \end{cases} \quad (7.3)$$

O método de Euler Melhorado é obtido com  $a_1 = a_2 = 1/2$  e  $b_1 = b_2 = 1$ .

Métodos de ordem superior são obtidos seguindo o mesmo procedimento. Abaixo apresentamos um método de 3<sup>o</sup> e 4<sup>o</sup>

**R-K 3<sup>o</sup> ordem:**

$$\begin{aligned} y_{n+1} &= y_n + \frac{2}{9}K_1 + \frac{1}{3}K_2 + \frac{4}{9}K_3 \\ K_1 &= hf(x_n, y_n) \\ K_2 &= hf(x_n + h/2, y_n + K_1/2) \\ K_3 &= hf(x_n + 3h/4, y_n + 3K_2/4) \end{aligned}$$

**R-K 4<sup>o</sup> ordem:**

$$\begin{aligned} y_{n+1} &= y_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 &= hf(x_n, y_n) \\ K_2 &= hf(x_n + h/2, y_n + K_1/2) \\ K_3 &= hf(x_n + h/2, y_n + K_2/2) \\ K_4 &= hf(x_n + h, y_n + K_3) \end{aligned}$$

## 7.4 Métodos de Adans-Bashforth

São métodos de passo múltiplos baseados na integração numérica. A estratégia é integrar a equação diferencial no intervalo  $[x_n, x_{n+1}]$ , isto é

$$\int_{x_n}^{x_{n+1}} y'(x)dx = \int_{x_n}^{x_{n+1}} f(x, y(x))dx \Rightarrow \quad (7.4)$$

$$y(x_{n+1}) = y(x_n) + \underbrace{\int_{x_n}^{x_{n+1}} f(x, y(x))dx}_{\text{Integração Numérica}} \quad (7.5)$$

A integral sobre a função  $f(x, y)$  é aproximada pela integral de um polinômio interpolador que pode utilizar pontos que não pertencem ao intervalo de integração. Dependendo da escolha dos pontos onde vamos aproximar a função  $f(x, y)$  os esquemas podem ser classificados como:

**Explícito:** São obtidos quando utilizamos os pontos  $x_n, x_{n-1}, \dots, x_{n-m}$  para interpolar a função  $f(x, y)$ .

**Implícito:** São obtidos quando no conjunto de pontos, sobres os quais interpolamos a função  $f(x, y)$ , temos o ponto  $x_{n+1}$ .

### 7.4.1 Métodos Explícitos

Vamos considerar o caso em que a função  $f(x, y)$  é interpolada sobre os pontos  $(x_n, f_n)$  e  $(x_{n-1}, f_{n-1})$ , onde  $f_n = f(x_n, y_n)$ . Considerando o polinômio interpolador na forma de Newton temos

$$f(x, y) \approx p(x) = f_{n-1} + f[x_{n-1}, x_n](x - x_{n-1})$$

Integrando sobre o intervalo  $[x_n, x_{n+1}]$  temos

$$\begin{aligned} \int_{x_n}^{x_{n+1}} p(x) dx &= \int_{x_n}^{x_{n+1}} f_{n-1} + f[x_{n-1}, x_n](x - x_{n-1}) dx \\ &= f_{n-1}(x_{n+1} - x_n) + f[x_{n-1}, x_n] \left( \frac{x_{n+1}^2}{2} - x_{n-1}x_{n+1} - \frac{x_n^2}{2} + x_{n-1}x_n \right) \\ &= hf_{n-1} + \frac{f_n - f_{n-1}}{h} \frac{1}{2} (x_{n+1}^2 - 2(x_n - h)x_{n+1} - x_n^2 + 2(x_n - h)x_n) \\ &= hf_{n-1} + \frac{f_n - f_{n-1}}{h} \frac{1}{2} (x_{n+1}^2 - 2x_nx_{n+1} + x_n^2 + 2h(x_{n+1} - x_n)) \\ &= hf_{n-1} + \frac{f_n - f_{n-1}}{h} \frac{1}{2} ((x_{n+1} - x_n)^2 + 2h^2) \\ &= \frac{h}{2} (2f_{n-1} + 3f_n) \end{aligned}$$

Substituindo a aproximação da integral em (7.5) obtemos o seguinte método

$$y_{n+1} = y_n + \frac{h}{2} (2f_{n-1} + 3f_n)$$

Este método é um método explícito, de passo dois. Isto significa que  $y_{n+1}$  depende de  $y_n$  e  $y_{n-1}$ . Logo necessitamos de dois valores para iniciar o método:  $y_0$  que é dado no P.V.I.;  $y_1$  que deve ser obtido por um método de passo simples. De uma forma geral, os métodos de  $k$ -passos necessitam de  $k$ -valores iniciais que são obtidos por métodos de passo simples, de ordem igual ou superior a ordem do método utilizado.

Obtemos o método, aproximando a função pelo polinômio interpolador de grau um. Assim o erro local, cometido por esta aproximação será

$$\begin{aligned} \int_{x_n}^{x_{n+1}} E_1(x) dx &= \int_{x_n}^{x_{n+1}} (x - x_{n-1})(x - x_n) \frac{f''(\xi, y(\xi))}{2!} dx \\ &= h^3 \frac{5}{12} y'''(\xi) \quad \xi \in [x_n, x_{n+1}] \end{aligned}$$

Com isto temos a seguinte estimativa para o erro local

$$|E_{loc}(x_{n+1})| \leq h^3 \frac{5}{12} \max_{\xi \in [x_n, x_{n+1}]} |y'''(\xi)|$$

**Exemplo 7.4.1** Considere o seguinte P.V.I.

$$\begin{cases} y' = -2xy \\ y(0.5) = 1 \end{cases}$$

Vamos achar uma aproximação para  $y(1.1)$  pelo Método de Adams-Bashforth explícito, de passo dois, usando  $h = 0.2$ .

Do P.V.I segue que  $y_0 = 1$  e  $x_0 = 0.5$ . Este é um método de passo dois e vamos ter que calcular  $y_1$  por um método de passo simples que tenha ordem igual ou superior ao do Método de Adams-Bashforth. Como o erro local depende da 3ª derivada de  $y(x)$  temos que este método é de 2ª ordem. Assim vamos utilizar o método de Euler Melhorado, que é um método de 2ª ordem.

$$\begin{aligned} y_1 &= y_0 + h \frac{1}{2} (f(x_0, y_0) + f(x_0, y_0 + hy_0')) \\ &= 1 + \frac{0.2}{2} (-2 * 0.5 * 1 + (-2) * 0.5 * (1 + 0.2 * (-2 * 0.5 * 1))) = 0.82 \approx y(0.7) \end{aligned}$$

Tendo o valor de  $y_0$  e  $y_1$  podemos iniciar o Método de Adams-Bashforth, sendo

$$\begin{aligned} y_2 &= y_1 + \frac{h}{2} (2f_0 + 3f_1) \\ &= 0.82 + \frac{0.2}{2} (2 * (-2) * 0.5 * 1 + 3 * (-2) * (0.7) * 0.82) = 0.2756 \approx y(0.9) \\ y_3 &= y_2 + \frac{h}{2} (2f_1 + 3f_2) \\ &= 0.2756 + \frac{0.2}{2} (2 * (-2) * 0.7 * 0.82 + 3 * (-2) * (0.9) * 0.2756) = -0.102824 \approx y(1.1) \end{aligned}$$

Se tivéssemos aproximado a função  $f(x, y)$  por um polinômio de grau 3, sobre os pontos  $(x_n, f_n), (x_{n-1}, f_{n-1}), (x_{n-2}, f_{n-2}), (x_{n-3}, f_{n-3})$  obteríamos o método de passo 4 e ordem 4 dado por

$$y_{n+1} = y_n + \frac{h}{24} [55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}] \quad E_{loc}(x_{n+1}) = h^5 \frac{251}{720} y^{(v)}(\xi) \quad \text{com } \xi \in [x_n, x_{n+1}]$$

Neste caso necessitamos de quatro valores iniciais,  $y_0, y_1, y_2$  e  $y_3$ , que deve ser calculados por um método de passo simples de ordem maior ou igual a quatro (Ex. Runge-Kutta de 4ª ordem).

## 7.4.2 Métodos Implícitos

Neste caso o ponto  $(x_{n+1}, f_{n+1})$  é um dos pontos, onde a função  $f(x, y)$  será interpolada. Vamos considerar o caso em que a função  $f(x, y)$  é interpolada sobre os pontos  $(x_n, f_n)$  e  $(x_{n+1}, f_{n+1})$ . Considerando o polinômio interpolador na forma de Newton temos

$$f(x, y) \approx p(x)f_n + f[x_n, x_{n+1}](x - x_n)$$

Integrando sobre o intervalo  $[x_n, x_{n+1}]$  temos

$$\begin{aligned} \int_{x_n}^{x_{n+1}} p(x)dx &= \int_{x_n}^{x_{n+1}} f_n + f[x_n, x_{n+1}](x - x_n)dx \\ &= f_n(x_{n+1} - x_n) + f[x_n, x_{n+1}] \left( \frac{x_{n+1}^2}{2} - x_n x_{n+1} - \frac{x_n^2}{2} + x_n x_n \right) \\ &= hf_n + \frac{f_{n+1} - f_n}{h} \frac{1}{2} (x_{n+1}^2 - 2x_n x_{n+1} + x_n^2) \\ &= hf_n + \frac{f_{n+1} - f_n}{h} \frac{1}{2} (x_{n+1} - x_n)^2 \\ &= \frac{h}{2} (f_n + f_{n+1}) \end{aligned}$$

Substituindo a aproximação da integral em (7.5) obtemos o seguinte método

$$y_{n+1} = y_n + \frac{h}{2} (f_n + f_{n+1})$$

O erro local, cometido por esta aproximação será

$$\begin{aligned} \int_{x_n}^{x_{n+1}} E_1(x)dx &= \int_{x_n}^{x_{n+1}} (x - x_{n+1})(x - x_n) \frac{f''(\xi, y(\xi))}{2!} dx \\ &= -h^3 \frac{1}{12} y'''(\xi) \quad \xi \in [x_n, x_{n+1}] \end{aligned}$$

Note que o cálculo de  $y_{n+1}$  depende de  $f_{n+1} = f(x_n, y_{n+1})$ . Em geral a  $f(x, y)$  não permite que isolemos  $y_{n+1}$ . Desta forma temos que usar um esquema Preditor-Corretor. Por um método explícito encontramos uma primeira aproximação para  $y_{n+1}$  (Preditor) e este valor será corrigido através do método implícito (Corretor).

**Exemplo 7.4.2** *Vamos considerar o seguinte problema:*

$$\begin{cases} y' = -2xy - y^2 \\ y(0) = 1 \end{cases}$$

Usando  $h = 0.1$  vamos achar uma aproximação para  $y(0.2)$ , usando o esquema

$$\begin{aligned} P &: y_{n+1} = y_n + hf(x_n, y_n) \\ C &: y_{n+1} = y_n + \frac{h}{2} (2f_n + f_{n+1}) \end{aligned}$$

Sendo  $x_0 = 0$  e  $y_0 = 1$  temos

$$P : y_1 = y_0 + hf(x_0, y_0) = 1 + 0.1(-2 * 0 * 1 - 1^2) = 0.9$$

$$C : y_1 = y_0 + \frac{h}{2} (2f_0 + f_1) = 1 + \frac{0.1}{2} (-2 * 0 * 1 - 1^2 - 2 * 0.1 * 0.9 - 0.9^2) = 0.9005 \approx y(0.1)$$

$$P : y_2 = y_1 + hf(x_1, y_1) = 0.9005 + 0.1(-2 * 0.1 * 0.9005 - (0.9005)^2) = 0.8013$$

$$\begin{aligned} C : y_2 &= y_1 + \frac{h}{2} (f_1 + f_2) = 0.9005 + \frac{0.1}{2} (-2 * 0.1 * 0.9005 - (0.9005)^2 - 2 * 0.2 * 0.8013 - 0.8013^2) \\ &= 0.8018 \approx y(0.2) \end{aligned}$$

## 7.5 Equações de Ordem Superior

Uma equação de ordem  $m$  pode ser facilmente transformadas em um sistema de  $m$  equações de primeira ordem. Este sistema pode ser visto como uma equação vetorial de primeira ordem e assim poderemos aplicar qualquer um dos métodos analisados nas seções anteriores. Como exemplo vamos considerar o problema de terceira ordem dado por

$$\begin{cases} y''' = x^2 + y^2 - y' - 2y''x \\ y(0) = 1 \\ y'(0) = 2 \\ y''(0) = 3 \end{cases}$$

Para transformar num sistema de primeira ordem devemos usar variáveis auxiliares. Fazemos  $y' = w \Rightarrow y'' = w'$  e fazemos  $y'' = w' = z \Rightarrow y''' = w'' = z'$ . Com isto a equação acima pode ser representada por:

$$\begin{cases} y' = w \\ w' = z \\ z' = x^2 + y^2 - w - 2zx \\ y(0) = 1 \\ w(0) = 2 \\ z(0) = 3 \end{cases}$$

O sistema acima pode ser escrito na forma matricial, onde

$$\underbrace{\begin{pmatrix} y' \\ w' \\ z' \end{pmatrix}}_{Y'} = \underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ y & -1 & -2x \end{pmatrix}}_A \underbrace{\begin{pmatrix} y \\ w \\ z \end{pmatrix}}_Y + \underbrace{\begin{pmatrix} 0 \\ 0 \\ x^2 \end{pmatrix}}_X$$

Desta forma temos a equação vetorial

$$\begin{cases} Y' = AY + X \\ Y(0) = Y_0 \end{cases}$$

onde  $Y_0 = (1, 2, 3)^T$ . Vamos aplicar o método de Euler na equação acima obtendo

$$Y_{n+1} = Y_n + h(A_n Y_n + X_n)$$

ou seja

$$\begin{pmatrix} y_{n+1} \\ w_{n+1} \\ z_{n+1} \end{pmatrix} = \begin{pmatrix} y_n \\ w_n \\ z_n \end{pmatrix} + h \left[ \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ y_n & -1 & -2x_n \end{pmatrix} \begin{pmatrix} y_n \\ w_n \\ z_n \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ x_n^2 \end{pmatrix} \right]$$

Tomando  $h = 0.1$ , vamos calcular uma aproximação para  $y(0.2)$ . Calculando  $Y_1 \approx Y(0.1)$  temos que

$$\begin{pmatrix} y_1 \\ w_1 \\ z_1 \end{pmatrix} = \begin{pmatrix} y_0 \\ w_0 \\ z_0 \end{pmatrix} + h \left[ \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ y_0 & -1 & -2x_0 \end{pmatrix} \begin{pmatrix} y_0 \\ w_0 \\ z_0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ x_0^2 \end{pmatrix} \right] \Rightarrow$$

$$\begin{pmatrix} y_1 \\ w_1 \\ z_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} + 0.1 \left[ \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & -1 & -2 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0^2 \end{pmatrix} \right] = \begin{pmatrix} 1.2 \\ 2.3 \\ 2.9 \end{pmatrix}$$

Calculando  $Y_2 \approx Y(0.2)$  temos

$$\begin{pmatrix} y_2 \\ w_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ w_1 \\ z_1 \end{pmatrix} + h \left[ \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ y_1 & -1 & -2x_1 \end{pmatrix} \begin{pmatrix} y_1 \\ w_1 \\ z_1 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ x_1^2 \end{pmatrix} \right] \Rightarrow$$

$$\begin{pmatrix} y_2 \\ w_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} 1.2 \\ 2.3 \\ 2.9 \end{pmatrix} + 0.1 \left[ \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1.2 & -1 & -2 * 0.1 \end{pmatrix} \begin{pmatrix} 1.2 \\ 2.3 \\ 2.9 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0.1^2 \end{pmatrix} \right] = \begin{pmatrix} 1.430 \\ 2.590 \\ 2.757 \end{pmatrix}$$

Note que neste caso não estamos achando apenas o valor aproximado de  $y(0.2)$ , mas também de  $y'(0.2)$  e  $y''(0.2)$ , sendo

$$\begin{pmatrix} y(0.2) \\ y'(0.2) \\ y''(0.2) \end{pmatrix} \approx \begin{pmatrix} 1.430 \\ 2.590 \\ 2.757 \end{pmatrix}$$

## 7.6 Exercícios

**Exercício 7.1** Dado o P.V.I.

$$\begin{cases} y' = x - y \\ y(1) = 2.2 \end{cases}$$

- a) Considerando  $h = 0.2$ , ache uma aproximação para  $y(2.6)$ , usando um método de segunda ordem.
- b) Se tivéssemos usado o método de Euler, com o mesmo passo  $h$ , o resultado obtido seria mais preciso? Justifique.

**Exercício 7.2** O esquema numérico abaixo representa um método para solução de E.D.O.

$$y_{n+1} = y_n + \frac{h}{24} [9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2}] \quad \text{com} \quad E_{loc}(x_{n+1}) = h^5 \frac{251}{720} y^{(5)}(\xi)$$

Classifique o método de acordo com o número de passos, se este é implícito ou explícito, a ordem do método, procurando sempre dar justificativas as suas respostas.

**Exercício 7.3** Determine o método de Runge-Kutta, de 2<sup>o</sup> ordem, obtido com  $a_1 = 0$ ,  $a_2 = 1$  e  $b_1 = b_2 = 1/2$  (ver (7.3)). Aplique o método para o P.V.I.

$$\begin{cases} \frac{y'}{y} = -2\sqrt{y} \\ y(1) = 0.25 \end{cases}$$



**Exercício 7.4** Considere o P.V.I.

$$\begin{cases} y' = y \\ y(0) = 1 \end{cases}$$

a) Mostre que quando aplicamos o método de Euler Melhorado ao problema temos

$$y_{n+1} = \left(1 + h + \frac{h^2}{2}\right)^{n+1}$$

b) Comparando com a solução exata do problema, você esperaria que o erro tivesse sempre o mesmo sinal? Justifique.

**Exercício 7.5** Considere o P.V.I. abaixo.

$$\begin{cases} y' = x - 2y \\ y(0) = 1 \end{cases}$$

a) Ache as aproximações, para  $y(x)$  no intervalo  $[0, 2]$ , usando  $h = 0.2$  e o esquema numérico baixo

$$P : y_{n+1} = y_n + hf(x_n + h/2, y_n + h/2y')$$

$$C ; y_{n+1} = y_n + \frac{h}{2}(f_{n+1} + f_n)$$

b) Sabendo que a solução exata é dada por  $y(x) = (5e^{-2x} + 2x - 1)/4$ , plote (novo verbo?) um gráfico com os valores obtidos pelo esquema Preditor e pelo esquema Corretor. Compare os resultados.

**Exercício 7.6** Determine uma aproximação para  $y(1)$  utilizando o método de Euler com  $h = 0.1$ , para o P.V.I. abaixo.

$$\begin{cases} y'' - 3y' + 2y = 0 \\ y(0) = -1, y'(0) = 0 \end{cases}$$