Universidade Estadual Vale do Acaraú-UVA
Centro de Ciências Exatas e Tecnológicas-CCET
Curso de Licenciatura em Matemática

Usando o Software Maxima na Resolução de EDO's[1] de 1ª e 2ª Ordem

Maria Ilsangela da Silva

Sobral-Ceará
Junho-2009

[1] Equações Diferenciais Ordinárias

Maria Ilsangela da Silva

Usando o Software Maxima na Resolução de EDO's de 1ª e 2ª Ordem

Monografia apresentada à Universidade Estadual Vale do Acaraú como requisito parcial para obtenção do título de graduação com Licenciatura Plena em Matemática sob a orientação do professor Pós-Doutor Tarcisio Praciano Pereira.

Sobral-Ceará
Junho-2009

Monografia apresentada como requisito parcial para obtenção do título de graduação com Licenciatura Plena em Matemática.

Orientanda: Maria Ilsangela da Silva

Monografia aprovada em: ——-/ 06 / 2009

Orientador: Prof. Phd Tarcisio Praciano Pereira (UVA)

1º Examinador: Prof. Ms. Delano Klinger Alves de Souza (UVA)

2º Examinador: Prof. Esp. Francisco Antônio de Alencar Menezes (UVA)

Coordenador do curso

Agradecimentos

1. A Deus por ter me dado força e persistência nos momentos difícies que passei durante o curso.

2. Ao professor Tarcisio Praciano Pereira que tanto me ajudou nessa jornada e que considero como um amigo sincero e verdadeiro. Em qualquer lugar que eu esteja, lebrarei sempre dessa figura que marcou minha vida.

3. A meus pais: Francisco Edmar da Silva e Benedita Rosa da Silva, por todo apoio e dedicação.

4. Aos professores que conquistaram minha confiança.

5. A meus colegas pelos momentos de convivência durante esses anos.

Dedicatória

Esta monografia é dedicada ao meu avô
e ao professor Tarcisio Praciano Pereira.

A vida é como uma partida de xadrez,
um lance errado, pode ser fatal.

Resumo

Este trabalho é apenas uma pequena introdução sobre o software Maxima, dando ênfase a uma de suas potencialidades de cálculo, resolver EDO's de 1ª e 2ª ordem. Escrevi de forma bem simples, possibilitando aos leitores, o entendimento do texto e a reprodução dos exemplos no ambiente Maxima. Da mesma forma, contribuir, para que posteriormente outros alunos possam ter o mesmo interesse que tive e, partindo deste pequeno trabalho, desenvolvam outros mais aprofundados. O capítulo 1 faz uma apresentação do software, mostra o ambiente Maxima, principais comando, construção de gráficos 2D e 3D, entre outas facilidades disponíveis, as quais, são essenciais para o desenvolvimento dos próximos capítulo. O capítulo 2, mostra como usar o software Maxima para resolver equações diferenciais de primeira ordem, apresenta as funções disponíveis no software para resolver analiticamente estas equações diferenciais e paralelamente, aborda um pouco da teoria sobre EDO's de primeira ordem. O capítulo 3 mostra como usar o Maxima para resolver EDO's de segunda ordem, com abordagem da teoria.

Sumário

Intodução **10**

1 Maxima, um Ambiente de Computação Simbólica **11**
1.1 O Ambiente do Maxima 12
1.2 Alguns Comandos Básicos 14
1.3 Derivando com o Maxima 18
1.4 Integrando com o Maxima 18
1.5 Desenhando Gráficos com o maxima 19

2 Usando Maxima para Resolver EDO's de 1ª Ordem **23**
2.1 Equações Diferenciais Ordinárias de 1ª Ordem 23
2.2 Campo de Direções e Trajetórias das Equações de 1ª Ordem 24
2.3 Equações Lineares de 1ª Ordem 27
2.4 Equações Separáveis 28
2.5 Equações Diferenciais Exatas 38
2.5.1 Fator Integrante 39

3 Usando Maxima para Resolver EDO's de 2ª Ordem **44**
3.1 Equações Diferenciais de 2ª Ordem 44
3.2 Equações Lineares de 2ª Ordem 45
3.2.1 Equações Homogênias com Coeficientes Constantes 46
3.3 Equações Não-Homogêneas- Métodos dos Coeficientes Indeterminados 49
3.4 Séries de Potência no Maxima 49
3.4.1 Solução Usando Série de Potência 50

Consideração Final **52**

Referências Bibliográficas **53**

Apêndices A **54**

Apêndices B **56**

Lista de Figuras

1.1 maxima-interface por linha de comandos. 13
1.2 xmaxima- interface gráfica do maxima. 14
1.3 wxmaxima- outra alternativa para interface gráfica do maxima. 14
1.4 Gráfico de duas curvas gerado com o openmath. 20
1.5 Gráfico da função $f(x) = -x^2$. 20
1.6 Gráfico de várias funções em um mesmo plano. 20
1.7 Gráfico gerado no Maxima com o openmath. 21
1.8 Gráfico gerado no Maxima com o Gnuplot. 21
1.9 Gráfico da função $f(x, y) = sin(x)sin(y)$ gerado no Maxima com o Gnuplot. 22

2.1 Campo de direções da equação y'=2x 24
2.2 Argumentos da função `plotdf` 24
2.3 Trajetória de uma solução da Eq.(3.32) no ponto $x = 0$ e $y = 0$. 25
2.4 Algumas Soluções da Eq.(3.32). 27
2.5 Forças atuando no objeto em queda livre. 29
2.6 Campo de direções da Eq.(2.11). 31
2.7 Campo de direções e solução de equilíbrio da Eq.(2.11). 31
2.8 Trajetória de uma solução da Eq.(2.11). 32
2.9 Algumas curvas integrais da Eq.(2.11). 34
2.10 Trajetória de uma solução da Eq.(2.13). 37
2.11 Algumas curvas integrais da Eq.(2.13). 37

3.1 Algumas soluções de $y'' + 5y' + 6y = 0$. 48

Introdução

A maioria dos princípios, ou leis, que regem o comportamento do mundo físico são relações(equações) que envolvem a taxa(derivada) segundo a qual as coisas acontecem. Portanto, para compreender e investigar problemas envolvendo a queda de um objeto na atmosfera, o aumento ou diminuição de populações, o movimento dos fluidos, entre outros, é necessário ter noções de equações diferenciais. Se uma equação diferencial é capaz de descrever algum processo físico, químico, biológico, etc, é chamada *modelo matemático* do processo em questão e, chegar a esta equação a partir das descrições desses processos é o que chamamos *modelagem do problema*.

O uso de um programa de computador é necessário na investigação de soluções de equações diferenciais, desde os problemas mais simples, que envolve equações diferenciais, aos mais complexos, os quais, só são possíveis com a ajuda da tecnologia computacional, principalmente quando precisa de gráficos, cálculos complicados e manipulação simbólica.

O Maxima é um exemplo de tal porgrama de computador, este software é considerado um ambiente de computação simbólica muito poderoso. É capaz de resolver equações diferenciais rapidamente, mesmo as mais complicadas, que demoraria muito tempo se resolvidas manualmente. Como se não bastasse, ainda é possível vizualizarmos os gráficos das soluções, assim como, os capos de direções.

Sendo este trabalho uma introdução do software com ênfase nas soluções de equações diferenciais, procuramos as EDO's de 1ª e 2ª ordem mais simples possíveis de resolver manualmente. Sabendo que, mesmo as mais simples, precisamos de muito tempo para resolvê-las e os cálculos são cansativos pois, na maioria das vezes, são fáceis e repetitivo. Nestas condições, nos deparamos com a necessidade de usar o Maxima.

Capítulo 1

Maxima, um Ambiente de Computação Simbólica

Uma categoria de software muito útil para o ensino da Matemática são os denominados *sistemas de computação simbólica*, ao contrário da computação numérica, tais sistemas permitem manipulação analítica de símbolos matemáticos, possibilitando a construção de sequência de cálculos algébricos. A característica mais importante de um sistema de computação simbólica, como o Maxima, é a habilidade que o programa tem de lidar com símbolos e obter respostas exatas para muitos problemas matemáticos e ainda permitir gerar e exportar gráficos em duas e três dimensões. Todas estas qualidades fazem do Maxima um laboratório excepcional para estudar matemática.

Foi desenvolvido na década de sessenta pelo grupo de Matemática Aplicada e Computação do MIT[1]. Antes era chamado de Macsyma[2] além do mais era um produto comercial. Alguns anos depois o Professor da Universidade do Texas, William F. Shelter, obteve uma autorização para estudar e ao mesmo tempo desenvolver o código do programa original, feito isso, tornou o programa open-source com o nome que conhecemos até hoje, *Maxima*. Com a morte de Shelter, um grupo de utilizadores e programadores juntaram-se para continuar o projeto.
Está disponível na forma GPL[3] e possui versão para os sistemas linux e windows.

O Maxima é uma ferramenta matemática muita usada para a manipulação de expressões algébricas que envolvem constantes, variáveis ou funções além de permitir diferenciar, integrar, fatorar polinômios, resolver sistemas lineares, expandir séries de Taylor ou Laurent, resolver equações diferenciais com ou sem condições iniciais, fazer manipulação de matrizes, plotar gráficos em duas e três dimensões, entre outras possibilidades de cálculo. Esse software é bem completo e ao mesmo tempo, uma ótima opção para quem precisa e não quer gastar com programas pagos.

[1] Massachusetts Institute of Technology
[2] MACs SYmbolic MAnipulator
[3] General Public License(Licença Pública Geral)

Nesse capítulo mostraremos o ambiente Maxima, principais comando, construção de gráficos 2D e 3D, entre outas facilidades disponíveis, as quais, achamos essenciais para o desenvolvimento dos próximos capítulo, que vamos dedicar a soluções de EDO's de 1ª e 2ª ordem.

O objetivo é mostrar um pouco as potencialidades desse software, capaz de resolver cálculos considerados tediosos, se fossem resolvidos manualmente.

1.1 O Ambiente do Maxima

Sua interface é por linha de comandos. No sistema linux, executando o comando *maxima* em um terminal veremos a tela (1.1), que usamos para incluir e executar os comandos do maxima, ou seja, temos acesso ao *prompt*. Para sair desse prompt é só digitar *quit();*, assim:

```
ilsangela@math:~$ maxima

Maxima 5.10.0 http://maxima.sourceforge.net
Using Lisp GNU Common Lisp (GCL) GCL 2.6.7 (aka GCL)
Distributed under the GNU Public License. See the file COPYIN    G.
Dedicated to the memory of William Schelter.
This is a development version of Maxima. The function bug_rep    ort()
provides bug reporting information.
(%i1) diff(x*sin(x),x);
(%o1)                                sin(x)+x cos(x)
(%i2) diff(%o1,x);
(%o2)                                2cos(x)-x sin(x)
(%i3) quit();
ilsangela@math:~$
```

A parte superior da tela (1.1) é como a que vemos acima, mostra a versão do maxima que estamos usando, informações sobre a licença-GNU-GPL- e a página oficial do programa. No canto esquerdo verá que o *prompt* do programa é algo do tipo (%i1), (%i2) ... (%iN) onde N é um número natural. O 'i' deve ser lembrado como input, que no inglês é usado para designar algo como "entrada". Já a resposta é dada em linhas com nomes da seguinte forma: (%o1), (%o2) ... (%oN). O 'o' deve ser lembrado como output, termo que lembra "saída". Estes nomes são variáveis também, que estão com as "saídas" de cada comando. A vantagem é que pode-se fazer referência a um resultado que foi obtido há algum tempo. É só fazer referência à saída, como mostra o exemplo abaixo:
Digitamos a expressão

```
(%i1) diff(x*sin(x),x);
```

para que o Maxima calcule a derivada, em seguida, aparece na tela como resposta

```
(%o1)                          sin(x) + x cos(x)
```

com o comando

```
(%i2) diff(%o1,x);
```

Maxima entende que queremos a derivada da expressão que ele mostrou como resposta à primeira equação, e retorna

```
(%o2)                          2 cos(x) - x sin(x)
```

Podemos ter usado simplismente o símbolo % em vez de %o1, quando queremos fazer referência ao último resultado.
Quando queremos atribuir um valor a uma variável, devemos fazer essa atribuição com dois pontos e não com o sinal de igualdade.

```
(%i3) x:3$
(%i4) y:7$
(%i5) x+y;
(%o5)      10
```

Essas atribuições permanecem valendo até finalizarmos o Maxima com `quit();` portanto, temos que tomar cuidado para não cometer algum tipo de erro.

Temos acima um exemplo simples de como usar maxima para calcularmos derivada de uma função. Como os comandos são lidos linha a linha, o ponto e vírgula serve para executar e imprimir na tela o resultado da operação ao apertar a tecla enter. Caso não queira que o resultado da operação seja impresso na tela, pode-se usar o $ (cifrão) ao final da sentença.

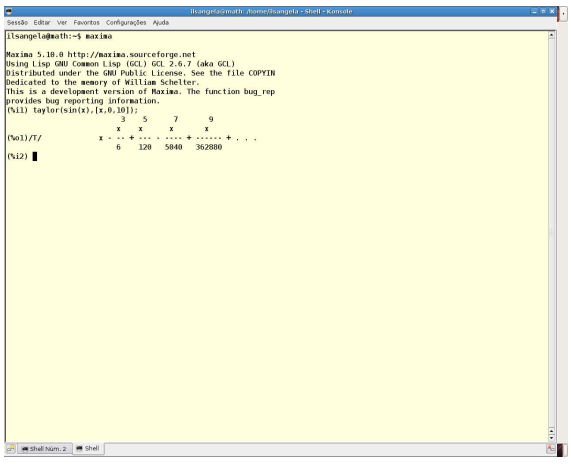

Figura 1.1: maxima-interface por linha de comandos.

A saída do Maxima é escrita usando caracteres, ou melhor, a imagem que aparece na tela não é muito boa mas, além dessa interface por linha de comandos, podemos ter uma interface gráfica, melhorando assim, o acesso ao prgrama. O

Maxima disponibiliza de interfaces graficas como *xMaxima* e *wxMaxima*.

O *wxMaxima* e *xmaxima* são plataformas gráficas onde o software *Maxima* pode ser executado sem o uso de linhas de comando. Esses programas fazem com que o *Maxima* fique com uma aparência gráfica, da forma que a maioria gosta de usar. Digitando *xmaxima* em um terminal aparecerá a tela (1.2) e *wxmaxima* aparecerá uma imagem como a Figura (1.3).
Como podemos observar, a interface gráfica do *xMaxima* apresenta duas divisões:

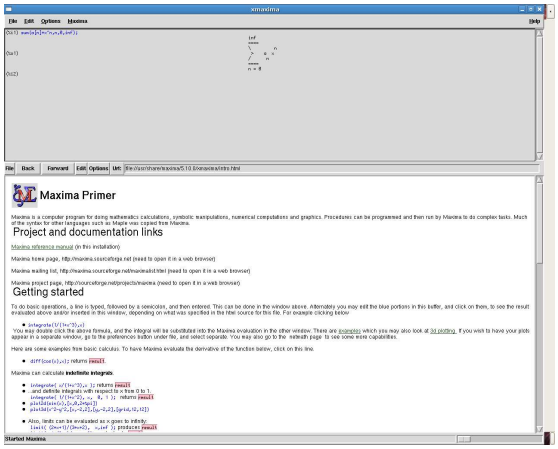

Figura 1.2: xmaxima- interface gráfica do maxima.

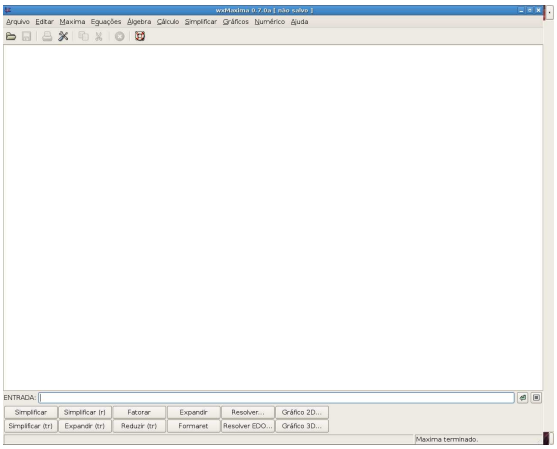

Figura 1.3: wxmaxima- outra alternativa para interface gráfica do maxima.

1ª-tela para digitar os comandos; 2ª- tela para termos acesso ao manual do maxima como se fosse diretamente pela internet.

1.2 Alguns Comandos Básicos

Definindo Variáveis

O nome atribuido a uma variável poderá ser qualquer combinação de letras, números e os símbolos % e _. O primeiro caracter não pode ser um número e uma variável não pode ter o mesmo nome de alguns comandos do Maxima. O Maxima diferencia entre letras maiúsculas e minúsculas.
Para definir variáveis usamos ':'(dois pontos). A lógica é a seguinte: Quando digitamos por exemplo $x : 2$, isso quer dizer que queremos atribuir o valor '2' à variável 'x'. Essa atribuição pode não ser números.

```
%i1) /*Definindo Variáveis*/
x:5$ /*atribui 5 a x*/
(%i2) y:4$ /*atribui 4 a y*/

(%i3) x+y; /*mostra o valor de x+y*/
(%o3)         9
(%i4) a:3*2$
```

```
(%i5) b:(6+4)$

(%i6) a+b;
(%o6)       16
```

Operações Aritméticas

Os operadores usados no maxima são os normalmentes usados em outros software como o octave, scilab, matlab, gnuplot. São os simbolos, $+$, $-$, $*$, / e $**$ que são respectivamente adição, subtração, multiplicação, divisão e potenciação.
A prioridade nos cálculos das operações é a mesma mesma usada na Matemática, parênteces, potenciações, multiplicações e divisões e por último, as adições e subtrações. Maxima não aceita agrupar as expressões com colchetes [ ] e chaves { }, temos que usar os parênteses ().

```
(%i9) /*Operações Aritméticas*/
(4+5)*3;
(%o9)                27
(%i10) (4+5)*3^2;
(%o10)               81
(%i11) ((4+5)*3)^2;
(%o11)              729
```

Manipulação Algébrica

Uma das capacidades mais importantes do Maxima é sua facilidade de manipular expressões algébricas. Vamos mostrar através do próximo exemplo esse processo: Vamos atribuir a variável $q1$ uma expressão e ver o que acontece

```
(%i1) q1: (x+a)^2+(x-4)^4;
                                      2           4
(%o1)                          (x + a)  + (x - 4)
```

Se observa que o Maxima não realizou nenhum cálculo. A função `expand` desenvolve as potências,

```
%i2) expand(q1);
                   4       3       2                     2
(%o2)             x  - 16 x  + 97 x  + 2 a x - 256 x + a  + 256
```

ainda é possível substituir alguma letra por outra expressão, por exemplo, substituir a por 3

```
(%i3) %,a=3;
                         4       3       2
(%o3)                   x  - 16 x  + 97 x  - 250 x + 265
```

Equações

O Maxima resolve uma equação de muitas formas diferentes, temos a equação

```
(%i4) eq: 4*x=3+x;
(%o4)                              4 x = x + 3
(%i5) %-x;
(%o5)                             3 x =  3
(%i6) %/3;

(%o6)                              x = 1
```

que subtraimos x nos dois lados da equação e a seguir dividimos o reto por 3, obtendo desta forma a solução da equação como se estivéssemos feito manualmente. A mesma equação pode ser resolvida de um modo mais rápido, usando a função `solve`

```
(%i9) solve(eq);
(%o9)                                [x = 1]
```

É possível resolver equações polinomiais de grau ≤ 4, porém, como é de se esperar, Maxima não dispõe de um método algébrico que permita resolver equações polinomiais de grau maior que quatro, retornando a mesma equação sem resolver.

```
%i15) solve(x^5+2*x^4+6=0);
                                       5      4
(%o15)                          [0 = x  + 2 x  + 6]
```

Existe outras funções diferentes de `solve` que resolve equações e sistemas, como por exemplo, `algsys`.

Funções Matemáticas

No Maxima estão definidas muitas funções matemáticas. Temos na tabela a seguir algumas delas.

sqrt(x)	$\sqrt{(x)}$
exp(x)	e^x
log(x)	$ln(x)$
sin(x)	$sin(x)$
cos(x)	$cos(x)$
tan(x)	$tan(x)$
sec(x)	$sec(x)$
cot(x)	$cot(x)$
asin(x)	$arcsin(x)$
acos(x)	$arccos(x)$
atan(x)	$arctan(x)$
abs(x)	$abs(x)$
x!	$x!$
sinh(x)	$sinh = \frac{1}{2}(e^x - e^{-x})$
cosh(x)	$cosh = \frac{1}{2}(e^x + e^{-x})$
tanh(x)	$\frac{sinh(x)}{cosh(x)}$

O Maxima sempre devolve resultados exatos em seus cálculos, mas podemos solicitar em formato decimal, com uma determinada precissão,

```
(%i4) asin(1);
                                        %pi
(%o4)                                   ---
                                         2

(%i5) %,numer;
(%o5)                          1.570796326794897
\end{vrbatim}
ou, definir a precissão com
\begin{verbatim}
(%i6) fpprec:60$ bfloat(%i4);
(%o7)   1.57079632679489661923132169163975144209858469968755291048747b0
```

Não terminam aqui as funções do Maxima, temos ainda as funções de Airy, elípticas, de Bessel entre outras.

Estrutura de Programação do Maxima

O Maxima possui uma estrutura de programação muito versátil para repetições *for/do*. Exemplo:

```
(%i1) for i:1 thru 8 do print(i^3);
1
8
27
64
125
216
```

```
343
512
(%o1)                                  done
(%i2) /* A variável de repetição "i", é uma variável local. Essa variável
varia de 1 a 8 com passo 1. Esse programa simplismente imprime o valor de
"i" ao cubo.*/
```

1.3 Derivando com o Maxima

O Maxima é muito útil para o cálculo diferencial e integral. A função *diff* é usada para calcular derivadas. A sintaxe usada no Maxima para derivar é a seguinte `diff(expressão,variável);`. Para uma derivada de ordem superior a sintaxe será: `diff(expressão, variável, ordem);`. Se as funções, as quias queremos derivar forem colocadas de forma genérica, o maxima simplismente explita as derivadas como veremos abaixo:

```
(%i22) diff(f(x)*g(x)*h(x),x);
                  d                          d                          d
(%o22) f(x) g(x) (-- (h(x))) + f(x) h(x) (-- (g(x))) + g(x) h(x) (-- (f(x)))
                  dx                         dx                         dx
```

Se tinvéssemos colocado ′ na função `diff` teria ficado assim:

```
(%i23)        'diff(f(x)*g(x)*h(x),x);
                           d
(%o23)                     -- (f(x) g(x) h(x))
                           dx
```

1.4 Integrando com o Maxima

A função *integrate* é usada para calcular integrais definidas e indefinidas. Quando queremos calcular a integral indefinida de uma função no Maxima devemos digitar os seguintes comandos no `prompt`: `integrate(expressão,variável);`. Integrando a função $y = x^2$ temos:

```
(%i24)                   integrate(x^2,x);
                                  3
                                 x
(%o24)                           --
                                 3
```

Mas também podemos calcular a integral definida usando a sintaxe: `integrate(expressão, variável,início,fim);`. Para a mesma função no intervalo $[0, 1]$,

```
%i25)                   integrate(x^2,x,0,1);

                                 1
(%o25)                           --
                                 3
```

1.5 Desenhando Gráficos com o Maxima

O software Maxima não está totalmente habilitado para construir gráficos, portanto, faz-se uso de um programa externo que realize esta tarefa. Nós, a partir do Maxima, nos encarregaremos de ordenar que tipo de gráfico nos interessa e Maxima se encarregará de comunicá-lo à aplicação gráfica que está ativa no momento, que por padrão, será o Gnuplot[4]. Veremos alguns exemplos de como gerar gráficos a partir do Maxima com Gnuplot e depois trataremos brevemente sobre como podemos modificar algumas das opções padrão do ambiente gráfico.

O controle das opções gráficas se consegue manipulando a variável global `plot_options`, cujo estado padrão é,

```
%i7) plot_options;
(%o7) [[x, - 1.75555970201398E+305, 1.75555970201398E+305],
[y, - 1.75555970201398E+305, 1.75555970201398E+305], [t, - 3, 3],
[grid, 30, 30], [view_direction, 1, 1, 1], [colour_z, false],
[transform_xy, false], [run_viewer, true], [plot_format, gnuplot],
[gnuplot_term, default], [gnuplot_out_file, false], [nticks, 10],
[adapt_depth, 10], [gnuplot_pm3d, false], [gnuplot_preamble, ],
[gnuplot_curve_titles, [default]], [gnuplot_curve_styles,
[with lines 3, with lines 1, with lines 2, with lines 5, with lines 4,
with lines 6, with lines 7]], [gnuplot_default_term_command, ],
[gnuplot_dumb_term_command, set term dumb 79 22], [gnuplot_ps_term_command,
se\
t size 1.5, 1.5;set term postscript eps enhanced color solid 24],
[logx, false], [logy, false], [plot_realpart, false]]
```

A função *plot2d* faz gráficos de $f(x)$ versus x, bidimensionais, que são exibidos em uma janela separada.
plot2d([f(x1),f(x2),...,f(xn)],[x,a,b],[y,c,d],opções)
sendo:

f(x1),f(x2),...,f(xn) funções de uma variável;
[x,a,b] intervalo de variação da variável x;
[y,c,d] intervalo de variação da variável y;
$a, b, c, d \in \mathbf{R}$.

A Figura 1.4 mostra o gráfico de duas curvas usando pontos grandes.

```
openplot_curves ([["{plotpoints 1} {pointsize 6}
      {label curva1}"],
      [1, 2, 3, 4, 5, 6, 7, 8], ["{label curva2} {color pink }"],
      [3, -1, 4, 2, 5, 7]]);
```

Este gráfico não foi gerado com o gnuplot.

[4] http://www.gnuplot.info

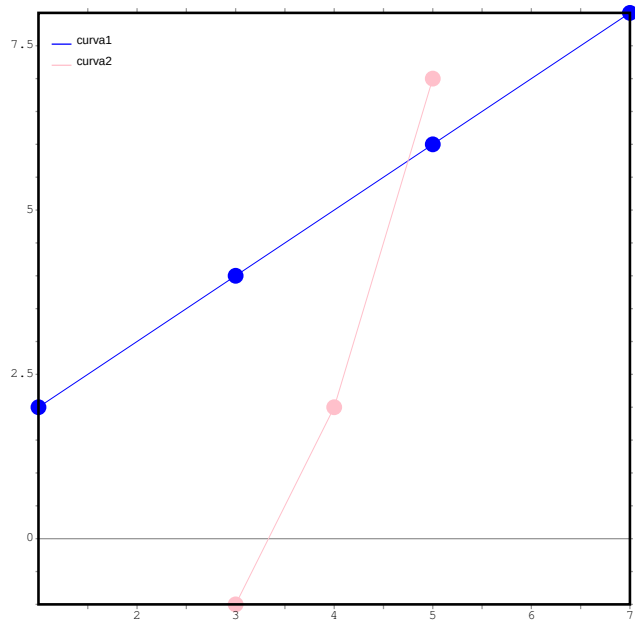

Figura 1.4: Gráfico de duas curvas gerado com o openmath.

Na Figura 1.5 temos o gráfico da função $f(x) = -x^2$ gerado no Maxima com o gnuplot e na Figura 1.6, o gráfico de várias funções em um mesmo plano cartesiano.

```
plot2d(exp(-x^2),[x,-2,5]);
```

```
plot2d([exp(-x^2),-x^2,sin(x),0],[x,-2,5],[y,-10,2]);
```

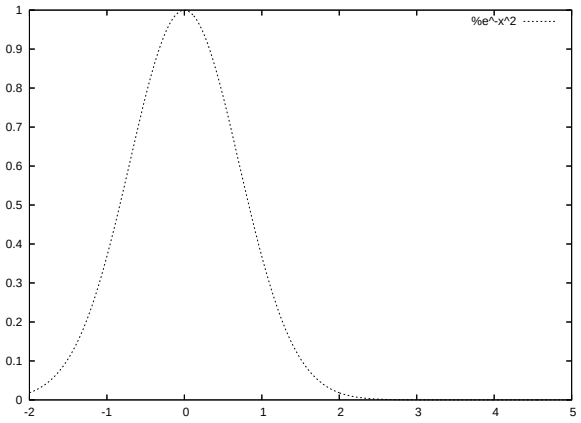

Figura 1.5: Gráfico da função $f(x) = -x^2$.

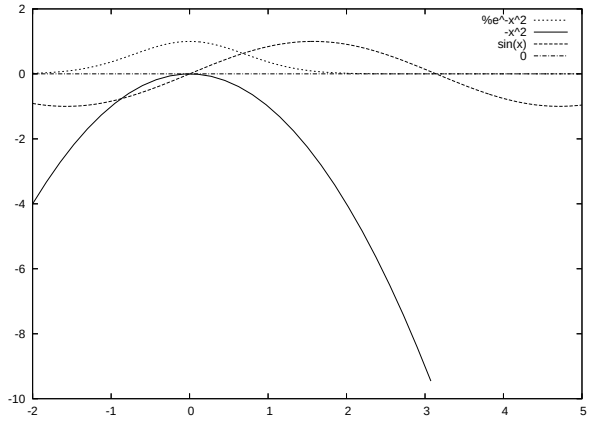

Figura 1.6: Gráfico de várias funções em um mesmo plano.

Já a função *plot3d* faz gráficos de $f(x, y)$ versus (x, y), sendo superfícies tridimensionais. O comando *plot3d* não admite várias funções simultaneamente. Existem dois aplicativos responsáveis para desenhar o gráfico, o Gnuplot(padrão no Maxima) e o Openmath(opcional no Maxima).
A opção *plot_format* com o valor *openmath*, permite escolher uma janela 3D mais interativa:

```
plot3d(x^2-y^2,[x,-4,4],[y,-4,4],[plot\_format,openmath])\$
```

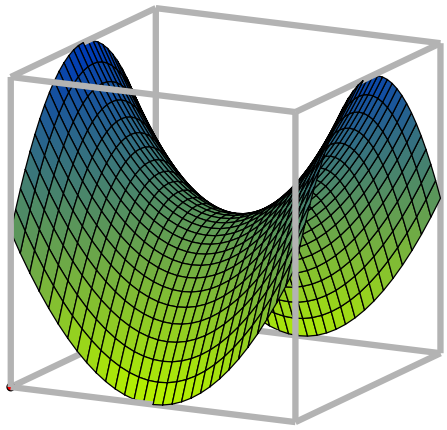

Figura 1.7: Gráfico gerado no Maxima com o openmath.

Os gráficos que estão sendo mostrados neste trabalho foram gerado por um arquivo gráfico em Postscript, da seguinte forma:

```
(%i16) plot3d(exp(-x^2-y^2),[x,-2,2],[y,-2,0],
                  [gnuplot_preamble,"set terminal postscript eps;
                                     set out 'graf05.eps'"]);
```

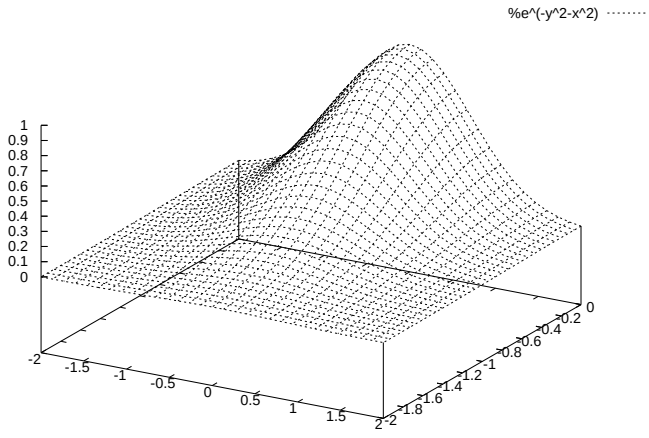

Figura 1.8: Gráfico gerado no Maxima com o Gnuplot.

salvos no formato *.eps*. Podemos também construir gráficos tridimencionais com a malha definida através o Gnuplot como mostra a Figura 1.9:

```
plot3d(sin(x)*sin(y), [x, 0, 2*%pi], [y, 0, 2*%pi],
[grid,50,50],[gnuplot_preamble,"set terminal postscript eps;set out
'graf06.eps'"]);
```

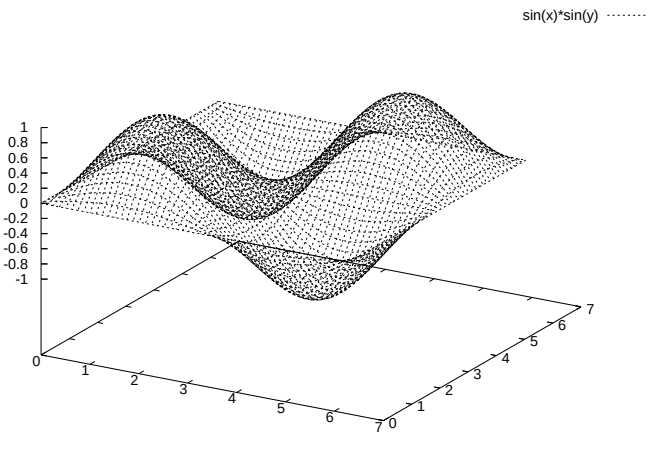

Figura 1.9: Gráfico da função $f(x, y) = sin(x)sin(y)$ gerado no Maxima com o Gnuplot.

O parâmetro gnuplot_preamble permite passar a este programa uma série de comandos que aperfeiçoam os detalhes; estes comandos devem ir separados por ponto e vírgula e devem ser os próprios da linguagem do Gnuplot.

Também podemos fazer com que o programa gráfico gere diretamente um arquivo PNG em lugar de mostrar no monitor com:

```
(%i15) plot3d(exp(-x^2-y^2),[x,-2,2],[y,-2,0],
                [gnuplot_preamble,"set terminal png size 420,320;
                                   set out 'graf05.png'"])\$
```

Para um melhor domínio destes detalhes é aconselhável recorrer à documentação deste programa (www.gnuplot.info).

Capítulo 2

Usando Maxima para Resolver EDO's de 1ª Ordem

O objetivo desse capítulo é usar o software Maxima para resolver equações diferenciais de primeira ordem, entender tais soluções e paralelamente, apresentar as funções disponíveis no software para resolver analiticamente essas equações diferenciais, nos livrando assim, de cálculos monótonos e cansativos. Para obter uma solução numérica de um sistema de equações diferenciais, temos um pacote chamado `dynamics` e para as representações gráficas o pacote `plotdf`.

2.1 Equações Diferenciais Ordinárias de 1ª Ordem

Uma equação diferencial de primeira ordem possui a forma

$$\frac{dy}{dx} = f(x, y). \tag{2.1}$$

Existem vários métodos para resolver equações diferenciais de primeira ordem. Essas equações foram divididas em várias subclasses com métodos de solução diferente. Como por exemplo, equações lineares, equações separáveis e equações exatas.

Para resolver analiticamente uma equação diferencial ordinária de primeira e segunda ordem usando o software *Maxima*, faz-se o uso da função `ode2(eq, var-dep, var-ind)`. Esta função usa três argumentos: a equação diferencial, a variável dependente, e a variável independente. Quando `ode2` encontra uma solução, retorna uma solução explícita ou implícita. Se a equação for de primeira ordem, $\%c$ é usado para representar a constante de integração, e $\%k1$ e $\%k2$ as constantes para equações de segunda ordem. Ao usar a função `ode2` para encontrar a solução de equações diferenciais, a variável independente deve sempre ser fornecida como o terceiro argumento.

No caso de `ode2` não conseguir obter uma solução por qualquer razão, retorna `false`, ao imprimir uma mensagem de erro. Os métodos para equações de primeira ordem na sequência em que vamos apresentar são: linear, separável, exato - fator de integração.

2.2 Campo de Direções e Trajetórias das Equações de 1ª Ordem

O campo de direção é fundamental para a avaliação de equações diferenciais, pois, apresentam uma melhor visão dos valores que as soluções dessas equações tendem. Esses campos são formados por vetores tangentes às soluções nos pontos do plano xy. Podemos usar o Maxima para desenhar tais campos direcionais, para isso é necessário ativar a biblioteca `plotdf` utilizando o comando `load(plotdf)`.

Examplo: Traçar o campo de direções para a Eq.(3.32) usando o Maxima.

O primeiro passo é ativar a biblioteca `plotdf`. Como a função que desenha os campos de direções é `plotdf` temos:

```
(%i1) load(plotdf);
(%o1)          /usr/share/maxima/5.10.0/share/contrib/plotdf.lisp
(%i2) plotdf(2*x);
```

Que tem como resultado a Figura (2.1). Na Figura (2.2) podemos ver os argumentos da função `plotdf`. A função `plotdf` possui alguns argumentos que servem

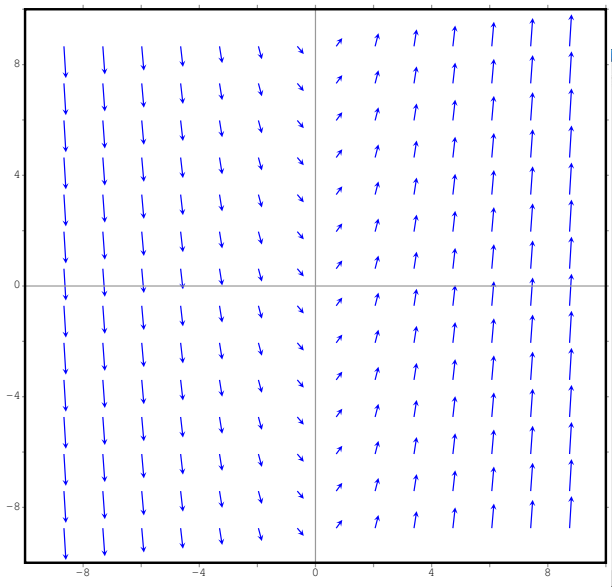

Figura 2.1: Campo de direções da equação y'=2x

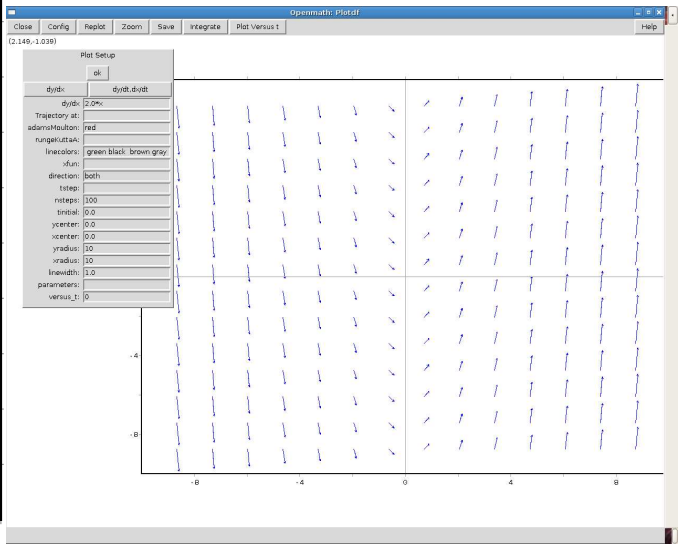

Figura 2.2: Argumentos da função `plotdf`

para a construção de campos de direções como,

Argumento	Parâmetros	Valor Padrão	Significado
xradius	1(real)	0	Magnitude do eixo x.
yradius	1(real)	0	Magnitude do eixo y
xcenter	1(real)	0	Centro do eixo x
ycenter	1(real)	0	Centro do eixo y
nsteps	1(inteiro)	100	Nº de passos p/ traçar a trajetória
trajectory_at	2(real,real)	vazio	Ponto em que é traçado a trajetória

Usando esses argumento podemos construir a trajetória de uma, ou mais, soluções da Eq.(3.32) que será mostrado na Figura 2.3.

```
(%i1) /*Trajetória de uma solução da equação dy/dx=2x no
ponto x=0 e y=0. */
eq1:'diff(y,x)=2*x;
                                      dy
(%o1)                                 -- = 2 x
                                      dx
(%i2) load(plotdf);
(%o2)          /usr/share/maxima/5.10.0/share/contrib/plotdf.lisp
(%i3) plotdf(2*x,[trajectory_at,0,0]);
pt.utf8
```

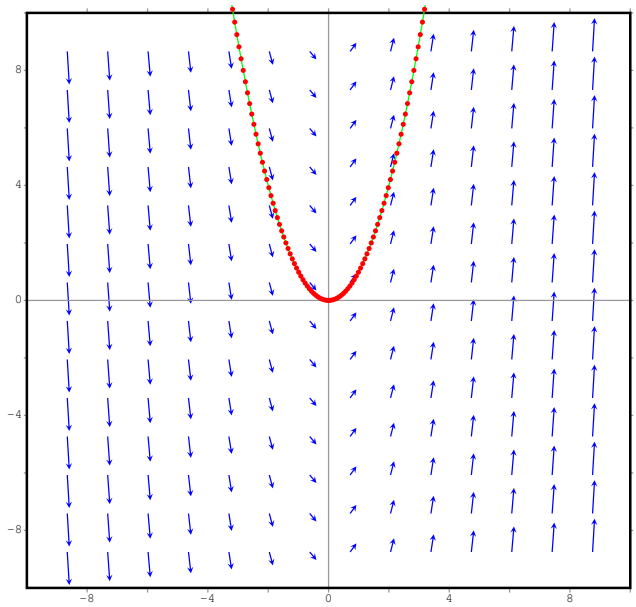

Figura 2.3: Trajetória de uma solução da Eq.(3.32) no ponto $x = 0$ e $y = 0$.

Para mostrarmos mais de uma solução no Maxima, devemos traçar o campo de direções e, ao clicar em `Config` podemos ver o campo `Trajectory at`, ao inserirmos os valores da coordenada do ponto desejado mostramos soluções adicionais.

Geometricamente, a solução da Eq.(3.32) é uma família de parábolas paralelas, com curvatura constante. Ver Figura (2.4). Com `plot2d`, construiremos essas parábolas para algumas condições iniciais dadas,

```
(%i1) /*Solução da equação dy/dx=2x e algumas curvas integrais.*/
eq1:''diff(y,x)=2*x;
                                    dy
(%o1)                               -- = 2 x
                                    dx
(%i2) ode2(''eq1,y,x);
                                        2
(%o2)                              y = x  + %c
(%i3) /*Uma solução particular. */
ic1(%o2,x=0,y=0);
                                       2
(%o3)                                y = x
(%i4) y1:rhs(%o3);
                                       2
(%o4)                                  x
(%i5) ic1(%o2,x=0,y=1);
                                       2
(%o5)                              y = x  + 1
(%i6) y2:rhs(%o5);
                                     2
(%o6)                                x  + 1
(%i7) ic1(%o2,x=0,y=2);
                                       2
(%o7)                              y = x  + 2
(%i8) y3:rhs(%o7);
                                     2
(%o8)                                x  + 2
(%i9) ic1(%o2,x=0,y=3);
                                       2
(%o9)                              y = x  + 3
(%i10) y4:rhs(%o9);
                                     2
(%o10)                               x  + 3
(%i11) ic1(%o2,x=-1,y=-2);
                                       2
(%o11)                             y = x  - 3
(%i12) y5:rhs(%o11);
                                     2
(%o12)                               x  - 3
(%i17) plot2d([y1,y2,y3,y4,y5],[x,-10,10],[y,-10,10],
[gnuplot_preamble," set terminal postscript eps;set out 'campo03.eps'"])\$
```

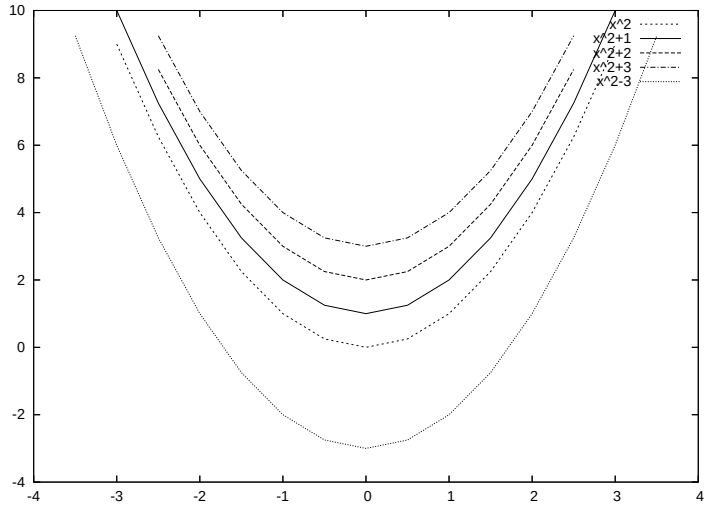

Figura 2.4: Algumas Soluções da Eq.(3.32).

2.3 Equações Lineares de 1ª Ordem

Podemos dizer que a Eq.(2.1) é linear de primeira ordem se f depender linearmente da variável depentente y. O movimeto de um objeto em queda livre na atmosfera pode ser modelado por uma equação diferencial linear

$$\frac{dy}{dx} = -ay + b \tag{2.2}$$

onde a e b são coeficientes constantes. De forma mais geral uma equação diferencial linear de primeira ordem é da forma

$$\frac{dy}{dx} + p(x)y = q(x), \tag{2.3}$$

onde $p(x)$ e $q(x)$ são funções da variável independente x.
O problema consiste em resolver essas equações. A Eq.(2.2) pode ser facilmente resolvida por um método de integração direta[1] mas, o mesmo método não pode ser aplicado para resolver a Eq.(2.3) pois, nem toda equação linear é separável. Diante disto, temos que recorre a outro método para encontrar a solução, que vamos mostrar por meio de um exemplo, como o Maxima faz isso.

A equação diferencial (2.4) pode ser calculada facilmente no Maxima, veja:

$$\frac{dy}{dx} + 2y = xe^{-2x} \tag{2.4}$$

```
(%i1) /*equação diferencial linear de primeira ordem*/
ode2('diff(y,t)+2*y=t*exp(-2*t),y,t);
                                    2
                                   t          - 2 t
(%o1)                        y = (-- + %c) %e
                                   2
```

[1] assunto da próxima subseção

O maxima apresenta a solução geral da equação, com isso podemos construir o campo vetorial atribuindo arbitrariamente valores para a constante c que dependem das condições iniciais do problema.

A função `ic1 (solução, xini, yini)`, de *initial conditions (condições iniciais)*, resolve problemas de valor inicial para equações diferenciais de primeira ordem. Aqui `solução` é uma solução geral para a equação, como a que encontramos acima, usando `ode2`. `xini` fornece um valor inicial para a variável independente na forma $x = x0$, e `yini` fornece o valor inicial para a variável dependente na forma $y = y0$.
Sendo as condições iniciais $t = 2$ e $y = 1$ para a equação resolvida acima, podemos encontrar o valor da constante c através da função `ic1`, da seguinte forma:
`(solução, valor inicial de t, valor incial de y)`

```
%i3) ic1(y=(t^2/2+%c)*%e^(-2*t),y=1,t=2);
                                  2       4         - 2 t
                              (t  + 2 %e  - 4) %e
(%o3)                    y = -----------------------
                                       2
```

2.4 Equações Separáveis

Como vimos na seção anterior nem toda equação linear de primeira ordem é separável. Mas existe equações de primeira ordem que podem ser resolvidas por um método de integração direta.
È possível reescrever a Eq.(2.1) na forma

$$P(x,y) + Q(x,y)\frac{dy}{dx} = 0 \tag{2.5}$$

supondo $P(x,y) = -f(x,y)$ e $Q(x,y) = 1$. Quando a função P depende apenas de x e a função Q depende apenas de y, chamamos de equações separáveis e são expressas na forma

$$P(x) + Q(y)\frac{dy}{dx} = 0 \tag{2.6}$$

Estas equações podem ser resolvidas por integração direta, integrando ambos os lados pois, podemos separar sem nenhum problema as variáveis para que fique:

$$P(x)dx = -Q(y)dy \tag{2.7}$$

A solução geral destas equações é uma família de curvas. O Maxima será bastante útil na construção dessas curvas, assim como nos cálculos, que são fáceis de fazer sem o uso da tecnologia computacional mas, cansativos.

Um exemplo bem simples destas equações diferenciais encontramos na Física, a que descreve o movimento de um corpo em queda livre.

Exemplo 1: Um Objeto em Queda Livre-[2]

*D*eixe cair um corpo de massa m de uma certa altura, considerando a resistência do ar e chamando de v sua velocidade, podemos estabelecer a lei de variação da velocidade da queda v, sendo a resistência do ar proporcional à velocidade(coeficiente de proporcionalidade k), isto é, encontrar $v = f(t)$.

Pela segunda lei de Newton

$$m\frac{dv}{dt} = F, \tag{2.8}$$

sendo $\frac{dv}{dt}$ a aceleração do corpo em movimento e F é a força que atua sobre o corpo no sentido do movimento. Nesse caso temos duas forças atuando sobre o corpo, portanto a força F é formada pela força da gravidade (mg) e pela resistência do ar (kv), nesse caso vamos considerar que a resistência do ar é proporcional a velocidade. Sendo assim:

$$m\frac{dv}{dt} = mg - kv, \tag{2.9}$$

Fazendo $g = 10m/s^2$

$$\frac{dv}{dt} = 10 - \frac{k}{m}v, \tag{2.10}$$

A Eq.(2.9) é considerada a equação de um objeto caindo na atmosfera, cuja solução de equilíbrio é $v(t) = mg/k$.

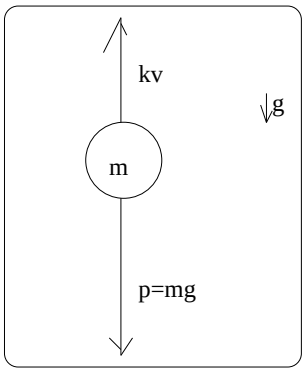

Figura 2.5: Forças atuando no objeto em queda livre.

Vamos determinar a solução dessa equação diferencial com o apoio do software Maxima. Temos na Eq.(2.9) três constantes sendo que m e k dependem unicamente do objeto que está caindo. Supondo que o objeto em queda tem $m = 10$ e $k = 2$ a Eq.(2.10) fica

$$\frac{dv}{dt} = 9.8 - \frac{v}{5}. \tag{2.11}$$

Então, com o apoio do Maxima vamos gerar o campo de direções (Figura 2.6), a solução de equilíbrio (Figura 2.7) e finalmente encontrar as soluções da Eq.(2.11).

```
(%i1) /* objeto em queda livre- no Maxima vamos usar
```

```
y no lugar de v e x no lugar de t pois, não é possível gerar os gráficos
usando as variáveis v e t  nesse software.*/
m:10$

(%i2) k:2$

(%i3) eq: 'diff(v,t)=10-k/m*v;
                          dv        v
(%o3)                     -- = 10 - -
                          dt        5
(%i6) /*vamos tentar entender o comportamento das soluções dessa
equação diferencial sem encontrar de fato, a solução. Supondo
v=40 e portanto, encontrar dv/dt.*/
v:40$
(%i7) eq1:rhs(%o3);
                                      v
(%o7)                           10 - -
                                      5
(%i8) /*rhs é usado no maxima quando queremos separar o lado direito
 da equação, para resolver.*/

(%i9) eq1: 10-k/m*v;
(%o9)                          2
(%i8) /*isso significa que o coeficiente angular da reta
tangente ao gráfico de uma solução v=v(t) vale 2 para v=40.
Para outros valores de v: */
(%i9) v:45;
(%o9)                               45
(%i44) eq1: 10-k/m*v;
(%o44)                               1
(%i41) v:49;
(%o41)                               49
(%i42) eq1: 10-k/m*v;
(%o42)                               1
                                     -
                                     5
v:50$

(%i10) eq1: 10-k/m*v;
(%o10)                               0.0
(%i11) v:60$

(%i12) eq1: 10-k/m*v;
(%o12)                        - 2
(%i13) v:65$
```

```
(%i14) eq1: 10-k/m*v;
(%o14)                          - 3
(%i15) /* Quando v<50 todos os segmentos de reta têm coeficiente
angulares positivos e v>50, coeficientes angulares negativos.
Quando v=50, teremos uma solução de equilíbrio entre a gravidade
e a resistência do ar, já que v(t)=50 não varia com o tempo. Com
 esses valores de v e consequentemente dv/dt, podemos formar um
campo de direção que, mesmo sem saber a solução da equação, podemos
analisar o comportamento das mesmas. Gerar o campo de direção */
load(plotdf);
(%o15)        /usr/share/maxima/5.10.0/share/contrib/plotdf.lisp
(%i37) plotdf(10-y/5,[xradius, 15 ],[yradius, 80],[ycenter, 50]);
pt.utf8
(%o37)                                    0
/* Analisando o campo de direção, conclui-se que todas as soluções parecem
convergir para a solução de equilíbrio na proporção em que t aumenta. Gerar
a solução de equilíbrio.*/

(%i40) plotdf(10-y/5,[xradius, 15 ],[yradius, 80],
[ycenter, 50],[trajectory_at, 0,50]);
pt.utf8

(%i41)/*Uma trajetória de uma solução no ponto (-4,69) e desenhada por:*/
plotdf(10-y/5,[xradius, 15 ],[yradius, 80],
[ycenter, 50],[trajectory_at, -4,69]);
pt.utf8
```

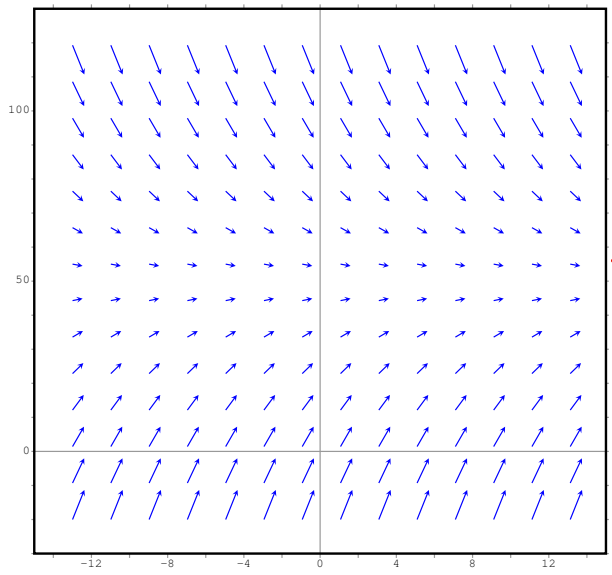

Figura 2.6: Campo de direções da Eq.(2.11).

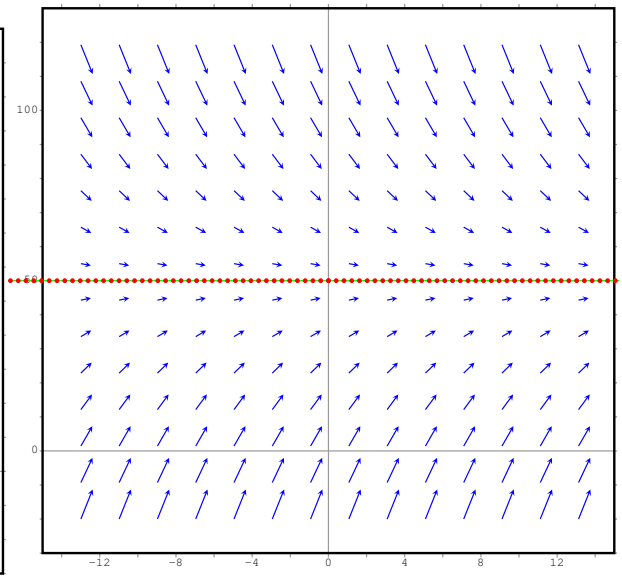

Figura 2.7: Campo de direções e solução de equilíbrio da Eq.(2.11).

Até agora o que fizemos foi uma análise do comportamento das soluções da Eq.(2.11) por meio dos campos de direções. Vamos então, encontrar a solução

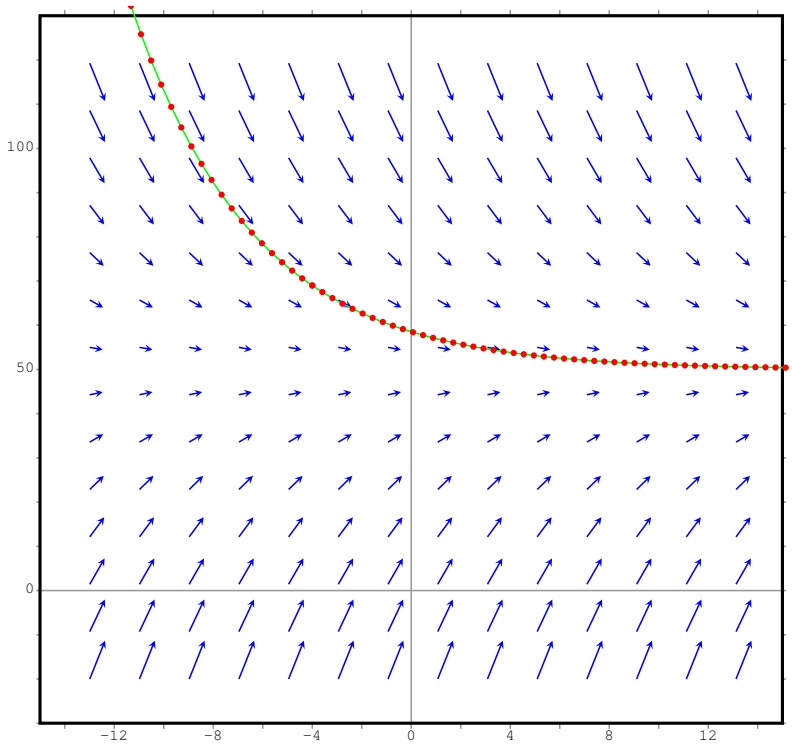

Figura 2.8: Trajetória de uma solução da Eq.(2.11).

geral dessa equação diferencial que é uma família infinita de curvas, chamadas curvas integrais[2]. Cada curva integral está associada a um valor particular de c, que somos capazes de descobrir a partir de uma condição inicial dada.

```
(%i1) /*Solução da equação de um objeto em queda livre*/
m:10$

(%i2) k:2$

(%i3) eq: 'diff(v,t)=10-k/m*v;
                                    dv         v
(%o3)                               -- = 10 - -
                                    dt         5
(%i4) ode2(''eq,v,t);
                                  - t/5        t/5
(%o4)                     v = %e       (50 %e    + %c)
```

O Maxima tem uma função que permite exportar expressões tex. Quando estamos usando o Maxima para fazer algum cálculo, o qual, queremos exportar para um documento em latex, fazemos o uso da função `tentex` da seguinte forma: primeiro devemos ativá-la com `load(tentex)` e em seguida usar `tex(expressao)` para exportar a expressão para o latex. Por exemplo: vamos exportar a solução geral da Eq.(2.11) para latex da seguinte forma,

[2]Representação geométrica da soluçõa geral de uma EDO.

```
(%i8) load(tentex);
(%o8)          /usr/share/maxima/5.10.0/share/tensor/tentex.lisp
(%i9) tex(%o4);
$$v=e^ {- {{t}\over{5}} }\,\left(50\,e^{{{t}\over{5}}}+{\it \%c}
 \right)\leqno{\tt (\%o4)}$$
(%o9)                                  (\%o4)
```

que fica assim

$$(\%o4) \qquad v = e^{-\frac{t}{5}} \left(50\, e^{\frac{t}{5}} + \%c\right)$$

e podemos representar por,

$$v = 50 + ce^{-\frac{t}{5}}. \tag{2.12}$$

já que o software Maxima não imprime as soluções das equações diferenciais de forma simplificada. Nos resta encontra as soluções da Eq.(2.11) e representá-las graficamente com a ajuda do Maxima. Ver Figura (2.9),

```
(%i3) eq1: 'diff(v,t)=10-v/5;
                                    dv        v
(%o3)                               -- = 10 - -
                                    dt        5
(%i5) /*Solução geral*/
ode2(''eq1,v,t);
                                  - t/5       t/5
(%o5)                      v = %e      (50 %e    + %c)
(%i6) /*Soluções particulares para certas condições iniciais dadas. */
ic1(%o5,t=0,v=0);
                                  - t/5       t/5
(%o6)                      v = %e      (50 %e    - 50)
(%i7) y1:rhs(%o6);
                                - t/5       t/5
(%o7)                         %e      (50 %e    - 50)
(%i8) ic1(%o5,t=0,v=1);
                                  - t/5       t/5
(%o8)                      v = %e      (50 %e    - 49)
(%i9) y2:rhs(%o8);
                                - t/5       t/5
(%o9)                         %e      (50 %e    - 49)
(%i10) ic1(%o5,t=0,v=3);
                                  - t/5       t/5
(%o10)                     v = %e      (50 %e    - 47)
(%i11) y3:rhs(%o10);
                                - t/5       t/5
(%o11)                        %e      (50 %e    - 47)

(%i27) ic1(%o5,t=1,v=3);
                                - t/5       1/5        t/5
```

```
(%o27)                 v = - %e       (47 %e    - 50 %e   )
(%i28) y4:rhs(%o27);
                             - t/5        1/5         t/5
(%o28)                   - %e       (47 %e    - 50 %e   )
 ic1(%o5,t=-10,v=6);
                             - t/5 - 2        t/5 + 2
(%o30)                 v = %e           (50 %e        - 44)
y5:rhs(%o30);
                             - t/5 - 2        t/5 + 2
(%o33)                   %e           (50 %e        - 44)
ic1(%o5,t=-6,v=6);
                             - t/5 - 6/5        t/5 + 6/5
(%o35)               v = %e             (50 %e          - 44)
(%i36) y6:rhs(%o35);
                             - t/5 - 6/5        t/5 + 6/5
(%o36)                 %e             (50 %e          - 44)
ic1(%o5,t=-3,v=6);
                             - t/5 - 3/5        t/5 + 3/5
(%o38)               v = %e             (50 %e          - 44)
(%i39) y7:rhs(%o38);
                             - t/5 - 3/5        t/5 + 3/5
(%o39)                 %e             (50 %e          - 44)

(%i44) plot2d([y1,y2,y3,y4,y5,y6,y7],[t,-10,15],[y,-10,60],[gnuplot_preamble,
"set terminal postscript eps;set out 'Bcurvas01.eps'"])\$
```

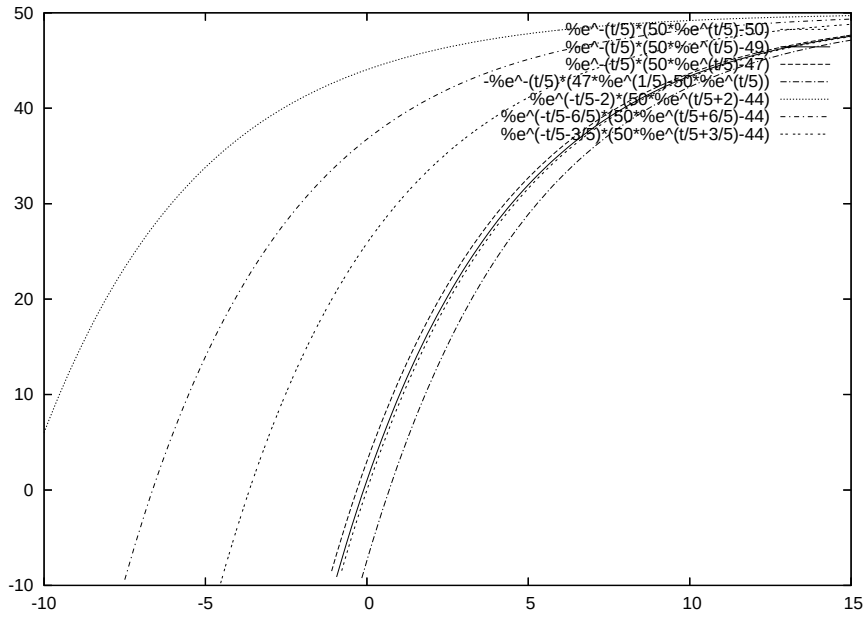

Figura 2.9: Algumas curvas integrais da Eq.(2.11).

Exemplo 2: O Maxima resolve a equação 2.13, traça o campo de direções e algumas curvas integrais para as condições de valor inicial dadas. Na Figura (2.10) temos o campo de direções e a trajetória de uma possível solução para $x = 0$ e $y = 1$. E, na Figura (2.11) temos algumas curvas integrais.

$$\frac{dy}{dx} = xy \tag{2.13}$$

```
(%i1) /*Solução da equação dy/dx=xy.*/
eq1: 'diff(y,x)=x*y;
                                    dy
(%o1)                               -- = x y
                                    dx
(%i2) /**Solução geral*/
ode2(''eq1,y,x);
                                          2
                                         x
                                         --
                                         2
(%o2)                           y = %c %e
(%i3) /*Campo de direções e trajetória de uma solução para x=0 e y=1.*/
load(plotdf);
(%o3)         /usr/share/maxima/5.10.0/share/contrib/plotdf.lisp
(%i4) plotdf(x*y,[trajectory_at,0,1]);
pt.utf8
(%o4)                                0
(%i5) /*Solução particular*/
ic1(%o2,x=0,y=0);
(%o5)                              y = 0
(%i6) y1:rhs(%o5);
(%o6)                                0
(%i7) ic1(%o2,x=0,y=1);
                                        2
                                       x
                                       --
                                       2
(%o7)                           y = %e
(%i8) y2:rhs(%o7);
                                        2
                                       x
                                       --
                                       2
(%o8)                                %e
(%i9) ic1(%o2,x=0,y=2);
```

```
                                          2
                                         x
                                         --
                                         2
(%o9)                           y = 2 %e
(%i10) y3:rhs(%o9);
                                          2
                                         x
                                         --
                                         2
(%o10)                            2 %e
(%i11) ic1(%o2,x=1,y=2);
                                          2
                                         x
                                         -- - 1/2
                                         2
(%o11)                          y = 2 %e
(%i12) y4:rhs(%o11);
                                         2
                                        x
                                        -- - 1/2
                                        2
(%o12)                            2 %e
(%i13)
ic1(%o2,x=-1,y=-2);
                                          2
                                         x
                                         -- - 1/2
                                         2
(%o13)                         y = - 2 %e
(%i14) y5:rhs(%o13);
                                         2
                                        x
                                        -- - 1/2
                                        2
(%o14)                           - 2 %e
(%i15) ic1(%o2,x=-2,y=-2);
                                          2
                                         x
                                         -- - 2
                                         2
(%o15)                          y = - 2 %e
(%i16) y6:rhs(%o15);
```

```
                                        2
                                       x
                                       -- - 2
                                       2
(%o16)                          - 2 %e
(%i17) ic1(%o2,x=-2,y=-3);
                                         2
                                        x
                                        -- - 2
                                        2
(%o17)                        y = - 3 %e
(%i18) y7:rhs(%o17);
                                         2
                                        x
                                        -- - 2
                                        2
(%o18)                          - 3 %e
(%i19) /*Gráfico das curvas */
plot2d([y1,y2,y3,y4,y5,y6,y7],[x,-5, 5],[y,-5, 5],[gnuplot_preamble,"set
terminal postscript eps;set out 'Bcurvas02.eps'"])\$
```

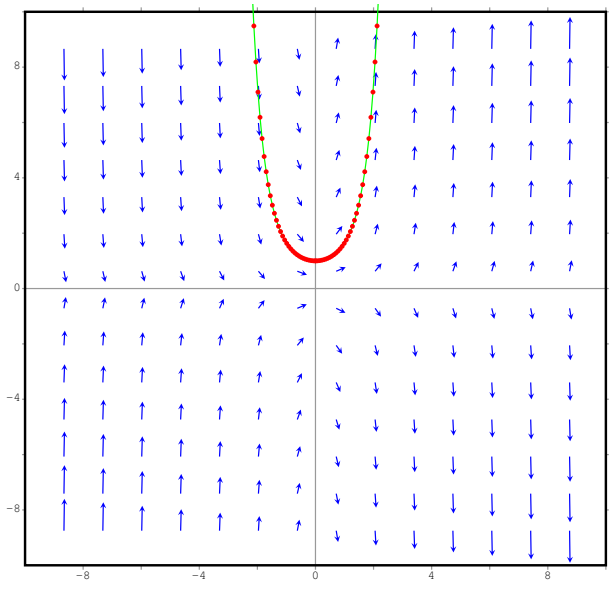

Figura 2.10: Trajetória de uma solução da Eq.(2.13).

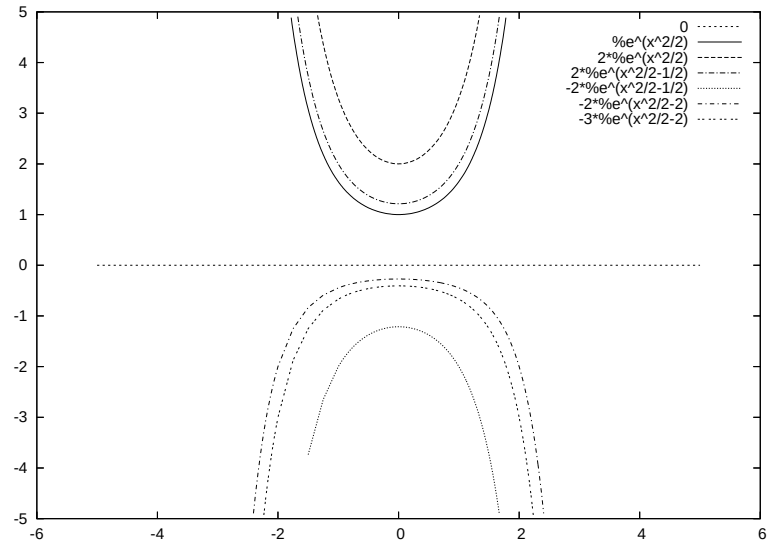

Figura 2.11: Algumas curvas integrais da Eq.(2.13).

Para expressar essa equação usamos o comando `diff`, sendo obrigatório o uso do apóstrofo (') antes de `diff` com o objetivo de evitar o cálculo da derivada, que por outro lado daria zero por não ter declarado a variável y como dependente de x. Para determinar a solução particular que satisfaça a condição inicial dada, usamos o comando `ic1`.

2.5 Equações Diferenciais Exatas

[5]

Nesta seção vamos resolver as chamadas equações diferenciais exatas com a ajuda do software Maxima.

Resolver equações diferenciais exatas, embora as mais simples, pode ser extremamente trabalhoso, por exemplo a determinação de um fator integrante que nos conduz a resolver uma outra equação diferencial. E por outro lado, não há absolutamente nenhum segredo ou novidade nas soluções destas equações, é puro cálculo. Neste ponto, o Maxima se transforma numa excelente ferramenta complementar para o ensino das equações diferenciais uma vez que nos libera dos cálculos mecânicos que poderiamos levar horas para fazer, não esquecendo as múltiplas possibilidades de erro. Com Maxima, ou qualquer outro programa de Computação Algébrica, podemos encontrar a solução e passar ao principal que é entender o que representam as soluções de equações diferenciais exatas - curvas, superfícies de nível, ou variedades diferenciáveis de dimensão n-1 quando a equação diferencial representar uma variedade[3] diferenciável de dimensão n.

Uma equação diferencial exata de primeira ordem é da forma

$$P(x,y) + Q(x,y)\frac{dy}{dx} = 0 \tag{2.14}$$

em que

$$P_y(x,y) = Q_x(x,y) \tag{2.15}$$

Caso a condição acima seja satisfeita, a solução geral da equação é dada pela função F, de forma que

$$F_x(x,y) = P(x,y) \quad \text{e} \quad F_y(x,y) = Q(x,y) \tag{2.16}$$

Exemplo: [2] Resolver, com a ajuda do Maxima, a equação

$$2x + y^2 + 2xyy' = 0. \tag{2.17}$$

A Eq.(2.17) não é linear nem separável, logo os métodos que resolve esses tipos de equações não são aplicáveis. Vamos verificar como o Maxima resolve:

```
/*equação exata*/

(%i1) ode2((2*x+y^2)+2*x*y*'diff(y,x)=0,y,x);
                                    2    2
(%o1)                             x y  + x  = %c
```

Como podemos perceber, o Maxima imprime na tela simplismente a solução geral da equação, não temos acesso aos cálculos que geraram esta solução. Portanto, para usar o software é necessário saber a teoria.

[3] generalidade para as noções geométricas

2.5.1 Fator Integrante

[3]

Em muitos casos a equação não se encontra na forma exata, e em muitas vezes, podemos trasformá-la em uma equação exata, multiplicando a equação por uma função $\mu(x, y)$

Exemplo: Vamos consirerar a equação

$$(y^2 + 3xy) + (xy + x^2)\frac{dy}{dx} = 0 \tag{2.18}$$

e analisar se a mesma é uma equação diferencial exata. Neste exemplo temos que

$$P(x, y) = y^2 + 3xy \Rightarrow \frac{\partial P}{\partial y} = 2y + 3x \tag{2.19}$$

$$Q(x, y) = xy + x^2 \Rightarrow \frac{\partial Q}{\partial x} = y + 2x \tag{2.20}$$

$$\frac{\partial P}{\partial y} \neq \frac{\partial Q}{\partial x} \tag{2.21}$$

concluíndo assim, que a Eq.(2.18) não é exata. Este exemplo serve para mostra que ao verificarmos se uma determinada equação é exata, o resultado pode não ser positivo simplismente pela falta de um fator que multiplique a equação, tornando-a em uma equação exata. Tal fator chamamos de *fator integrante*. Nos resta agora mostrar como encontrá-lo, a fim de tornar uma equação diferencial

$$P(x, y) + Q(x, y)\frac{dy}{dx} = 0 \tag{2.22}$$

numa equação diferencial exata.

Como não sabemos qual é o fator, vamos expressá-lo por $\mu(x, y)$ e escrever a Eq.(2.22) na forma

$$\mu(x, y)P(x, y) + \mu(x, y)Q(x, y)\frac{dy}{dx} = 0 \tag{2.23}$$

e fazer com que a Eq.(2.23) seja exata e assim, encontrar $\mu(x, y)$. Pelo que vimos anteriormente, a Eq.(2.22) é exata se, e somente se,

$$(\mu P)_y = (\mu Q)_x \tag{2.24}$$

então temos:

$$\frac{\partial \mu P}{\partial y} = \mu_y P + \mu P_y \tag{2.25}$$

$$\frac{\partial \mu Q}{\partial x} = \mu_x Q + \mu Q_x \tag{2.26}$$

então

$$\mu_y P + \mu P_y = \mu_x Q + \mu Q_x \tag{2.27}$$

$$\mu_y P + \mu P_y - \mu_x Q - \mu Q_x = 0 \tag{2.28}$$

$$\mu_y P - \mu_x Q + (P_y - Q_x)\mu = 0 \tag{2.29}$$

Sendo P e Q funções dadas, a Eq.(2.23) diz que o fator integrante μ tem que satisfazer a Eq.(2.29). Se for possível encontrar essa função μ, então a Eq.(2.23) é exata.

A Eq.(2.29) pode ter mais de uma solução, se esse for o caso, qualquer uma dessas soluções pode ser usada como fator integrante para a Eq.(2.22). Isso acontece devido a possibilidade de não unicidade do fator integrante.
Diante disto, podemos fazer por exemplo, $\mu_y = 0$, que torna μ uma função apenas de x. Se isto não for uma boa opção, tentamos para $\mu_x = 0$, que torna μ uma função apenas de y. Existe casos em que a solução será encontrada de forma bem complicada. Infelizmente a Eq.(2.23), que determina o fator integrante μ, tem a mesma dificuldade para resolver quanto a equação original(2.22). Embora fatores integrantes sejam ferramentas poderosas para a solução de equações diferenciais, os mesmos só podem ser encontrados em casos especiais. Podemos encontrar fatores integrantes simples quando μ é uma função de apenas uma das variáveis x ou y, em vez de depender de ambas.

Supondo que $\mu_y = 0$ temos,

$$\mu_y P - \mu_x Q + (P_y - Q_x)\mu = 0 \tag{2.30}$$

$$(P_y - Q_x)\mu - \mu_x Q = 0 \tag{2.31}$$

$$(P_y - Q_x)\mu = \mu_x Q \tag{2.32}$$

$$\frac{\mu_x}{\mu} = \frac{P_y - Q_x}{Q} \tag{2.33}$$

$$\frac{d\mu}{\mu} = \frac{(P_y - Q_x)dx}{Q} \tag{2.34}$$

$$ln(\mu) = \int \frac{(P_y(x,y) - Q_x(x,y))dx}{Q(x,y)} \tag{2.35}$$

A Eq.(2.35), calculada exatamente, expressa a função μ.
Agora é possível continuar a resolver a Eq.(2.18)

$$P(x,y) = y^2 + 3xy \quad \text{e} \quad Q(x,y) = xy + x^2 \tag{2.36}$$

$$\frac{\partial P}{\partial y} = 2y + 3x \neq \frac{\partial Q}{\partial x} = y + 2x \tag{2.37}$$

Queremos encontrar uma função μ tal que

$$P\mu + Q\mu\frac{dy}{dx} = 0 \tag{2.38}$$

seja exata.

$$\frac{\partial}{\partial y}P\mu = \frac{\partial P}{\partial y}\mu + P\frac{\partial \mu}{\partial y} = \tag{2.39}$$

$$\frac{\partial}{\partial y}[(y^2+3xy)\mu] = (2y+3x)\mu + (y^2+3xy)\frac{\partial \mu}{\partial y} \tag{2.40}$$

$$\frac{\partial}{\partial x}Q\mu = \frac{\partial Q}{\partial x}\mu + Q\frac{\partial \mu}{\partial x} = \tag{2.41}$$

$$\frac{\partial}{\partial x}[(xy+x^2)\mu] = (y+2x)\mu + (xy+x^2)\frac{\partial \mu}{\partial x} \tag{2.42}$$

Pela Eq.(2.24) temos

$$(2y+3x)\mu + (y^2+3xy)\frac{\partial \mu}{\partial y} = (y+2x)\mu + (xy+x^2)\frac{\partial \mu}{\partial x} \tag{2.43}$$

$$\Rightarrow (2y+3x)\mu - (y+2x)\mu + (y^2+3xy)\frac{\partial \mu}{\partial y} - (xy+x^2)\frac{\partial \mu}{\partial x} \Rightarrow \tag{2.44}$$

$$(y+x)\mu + (y^2+3xy)\frac{\partial \mu}{\partial y} - (xy+x^2)\frac{\partial \mu}{\partial x} \tag{2.45}$$

È nesse ponto que temos uma complicação na resolução da Eq.(2.45), para facilitar, temos que supor μ, uma função apenas de x ou apenas de y para que possamos anular uma das parcelas.

Supondo $\mu_y = 0$(função apenas de x) temos,

$$(y+x)\mu + (y^2+3xy)\frac{\partial \mu}{\partial y} - (xy+x^2)\frac{\partial \mu}{\partial x} \Rightarrow \tag{2.46}$$

$$(y+x)\mu - (xy+x^2)\frac{\partial \mu}{\partial x} \Rightarrow \tag{2.47}$$

$$(xy+x^2)\mu' = (y+x)\mu \tag{2.48}$$

$$\frac{\mu'}{\mu} = \frac{x+y}{xy+x^2} = \frac{x+y}{x(y+x} = \frac{1}{x} \Rightarrow \tag{2.49}$$

$$\frac{\mu'}{\mu} = \frac{1}{x} \Rightarrow \tag{2.50}$$

$$\frac{d\mu}{\mu dx} = \frac{1}{x} \tag{2.51}$$

Logo, existe um fator integrante μ que é função só de x e satisfaz a equação diferencial

$$\frac{d\mu}{\mu} = \frac{dx}{x} \tag{2.52}$$

integrando a Eq.(2.52) temos,

$$ln(\mu) = ln(x) \tag{2.53}$$

Portanto,

$$\mu(x) = x \tag{2.54}$$

Multiplicando a Eq.(2.18) por esse fator integrante, obtemos

$$(xy^2 + 3x^2y) + (x^2y + x^3)\frac{dy}{dx} = 0. \tag{2.55}$$

A Eq.(2.55) é uma equação diferencial exata com solução da forma

$$F(x, y) = c \tag{2.56}$$

O Maxima resolve normalmente a Eq.(2.18), ou melhor, a Eq.(2.55) como veremos abaixo.

```
%i1) ode2((3*x*y+y^2)+(x^2+x*y)*'diff(y,x)=0,y,x);
                                  2  2       3
                                 x  y  + 2 x  y
(%o1)                            -------------- = %c
                                       2
```

Para saber se o Maxima usou um fator integrante para fazer a transformação da Eq.(2.18) em uma equação exata, usa-se a função `intfactor`. Nesse caso, a função `intfactor` retorna x, ou melhor, multiplica a equação diferencial por $\mu(x) = x$. Veja o que acontece no Maxima quando usamos a função `intfactor`

```
(%i2) intfactor (%i1);
                                  2  dy    2
(%o2)            x(ode2((x y + x ) -- + y  + 3 x y = 0, y, x))
                                    dx
```

Ou seja, a equação exata que o maxima resolveu foi

$$(x^2y + x^3)\frac{dy}{dx} + xy^2 + 3x^2y = 0. \tag{2.57}$$

Com este exemplo podemos notar o quanto é importante saber usar um software como o Maxima, que nos auxili na solução de equações diferenciais para não perder tempo nos cálculos que, por muitas vezes são fáceis de fazer mas, cansativos.

Vamos mostrar outros exemplos de equações diferenciais que não são exatas mas o Maxima encontra um fator integrante e em seguida multiplica pelos termos da equação original e produz uma equação diferencial exata, a qual ele resolve.

$$y + (2x - ye^y)\frac{dy}{dx} = 0 \tag{2.58}$$

A solução encontrada usando o Maxima foi

```
(%i41) ode2(y+(2*x-y*exp^y)*'diff(y,x)=0,y,x);
```

```
                2       2                              log(exp) y      3         2
           (log (exp) y  - 2 log(exp) y + 2) %e                - log (exp) x y
(%o41) - ------------------------------------------------------------------- =
%c
                                                3
                                           log (exp)
```

e o fator de integração para tornar a Eq.(2.58) exata é $\mu = y$. Na verdade o Maxima resolveu a equação

```
(%i42) intfactor (%i41);

                                                 y      dy
(%o42)                 y(ode2((2 x - exp  y) -- + y = 0, y, x))
                                                dx
```

Vamos resolver a equação

$$(2y^2 + 3x) + 2xy\frac{dy}{dx} = 0. \tag{2.59}$$

Esta equação também não é exata e o procedimento é o mesmo, o Maxima vai encontrar o fator integrante e depois resolver a equação.

```
(%i6) ode2(2*y^2+3*x+(2*x*y)*'diff(y,x)=0,y,x);
                                    2  2    3
(%o6)                              x  y  + x  = %c
```

Quando digitamos `intfactor` no prompt do Maxima temos como resposta

```
(%i7) intfactor (%i6);
                                    dy      2
(%o7)                 x(ode2(2 x y -- + 2 x  + 1 = 0, y, x))
                                    dx
```

que é a equação resolvida pelo Maxima.

Capítulo 3

Usando Maxima para Resolver EDO's de 2ª Ordem

Nesse capítulo vamos resolver equações diferenciais ordinárias de 2ª ordem com a ajuda do software Maxima.

3.1 Equações Diferenciais de 2ª Ordem

Uma equação diferencial é dita de segunda ordem quando o maior grau de suas derivadas é 2. Esse tipo de equação tem sua importância, possui muitos métodos de resoluções com um nível matemático relativamente elementar. São essenciais para qualquer investigação séria nas áreas clássicas da física matemática, como mecânica dos fluídos, condução de calor, movimento ondulatório entre outros.

A forma de uma equação diferencial de segunda ordem é

$$\frac{d^2y}{dx^2} = f(x, y, \frac{dy}{dx}). \tag{3.1}$$

Alguns tipos de equações de segunda ordem que podem ser resolvidos com a utilização do software Maxima são: coeficientes constantes, exato, linear homogêneo com coeficientes não constantes que podem ser transformados em coeficientes constantes, equações resolvíveis pelo método de variação de parâmetros e equações que possam ser reduzidas a duas equações lineares de primeira ordem para serem resolvidas em seguida.

Nessa seção vamos apresentar apenas as equações homogêneas com coeficientes constantes e as equações não-homogêneas com o método dos coeficiéntes indeterminados.

O Maxima também faz o uso da função `ode2` para resolver, equações de segunda ordem. A função: *ic2 (solução, xval, yval, dval)* resolve o problema de valor inicial

para equação diferencial de segunda ordem. Aqui, solução é uma solução geral para a equação, `xval` é uma equação para a variável independente na forma $x = x0$, `yval` é uma equação para a variável dependente na forma $y = y0$, e `dval` é uma equação para a derivada da variável dependente com relação à variável independente avaliada no ponto `xval`.

3.2 Equações Lineares de 2ª Ordem

Sendo f uma função dada. A Eq.(3.1) é linear se f assume a forma:

$$f(x, y, \frac{dy}{dx}) = g(x) - p(x)\frac{dy}{dx} - q(x)y \Rightarrow \tag{3.2}$$

$$\frac{d^2y}{dx} = g(x) - p(x)\frac{dy}{dx} - q(x)y \Rightarrow \tag{3.3}$$

$$y'' + p(x)y' + q(x)y = g(x) \tag{3.4}$$

ou seja, se f é linear em y e y' as funções g, p e q dependem apenas da variável independente x.

Podemos encontrar também a equação

$$P(x)y'' + Q(x)y' + R(x)y = G(x) \tag{3.5}$$

Nesse caso, se $P(x) \neq 0$, podemos dividir a Eq.(3.5) por $P(x)$ e obter a Eq.(3.4) com

$$p(x) = \frac{Q(x)}{P(x)}, \quad q(x) = \frac{R(x)}{P(x)}, \quad g(x) = \frac{G(x)}{P(x)}, \tag{3.6}$$

obs: Vamos considerar apenas os casos em intervalos que as funções g, p e q são contínuas.
As condições iniciais para uma equação de segunda ordem indicam um ponto particular (x_0, y_0) pertencente ao gráfico da solução e o coeficiente angular y'_0 da reta tangente ao gráfico naquele ponto. Então, um problema de valor inicial consite em uma equação diferencial, como por exemplo a Eq.(3.1), com um par de condições iniciais

$$y(x_0) = y_0, \quad y'(x_0) = y'(x_0), \tag{3.7}$$

onde y_0 e y'_0 são números dados. Uma explicação para isso é que uma equação de segunda ordem precisa de duas integrações para se encontrar a solução e cada integração produz uma constante arbitrária. Portanto, espera-se que duas condições iniciais sejam suficientes para determinar os valores dessas suas constantes.

3.2.1 Equações Homogênias com Coeficientes Constantes

Uma equação diferencial linear de segunda ordem é homogênea quando a função $g(x) = 0$. Reescrevendo a Eq.(3.5), fazendo $g(x) = 0$, temos uma equação homogênea que escrevemos na forma:

$$P(x)y'' + Q(x)y' + R(x)y = 0 \tag{3.8}$$

Quando adquirirmos mais experiência vamos perceber que o problema de resolver uma equação homogênea é o mais fundamental, ou melhor, é o passo inicial para resolvermos a não-homogênea.

Nesse primeiro momento vamos considerar apenas equações que possuem as funções P, Q e R constantes. Sendo assim a Eq.(3.8) fica

$$ay'' + by' + cy = 0 \tag{3.9}$$

com a, b e c constantes.

A Eq.(3.9) não é muito difícil de ser resolvida por meio de funções elementares. Mas, se os coeficientes dessa equação não são constantes, temos dificuldades de encontrar a solução sendo preciso usar séries de potências.

Exemplo: Vamos usar o Maxima para resolver a equação

$$6y'' - y' - y = 0 \tag{3.10}$$

```
(%i1) /*equação homogênea*/
eq: 6*'diff(y,x,2)-'diff(y,x)-y=0;
                                  2
                                 d y   dy
(%o1)                          6 --- - -- - y = 0
                                   2   dx
                                 dx
(%i2) /*solução geral equação homogênea*/
ode2(''eq,y,x);
                                       x/2          - x/3
(%o2)                    y = %k1 %e    + %k2 %e
(%i3) method(%i2);
                                    2
                                   d y   dy
(%o3)          constcoeff(ode2(6 --- - -- - y = 0, y, x))
                                     2   dx
                                   dx
(%i4) /*solução particular da equação homogênea*/
ic2(%o2,x=0,y=1,diff(y,x)=2);
```

```
                                      x/2       - x/3
                                  14 %e      9 %e
(%o4)                        y = -------- - ---------
                                    5           5
```

Como podemos perceber, o Maxima usa a função `ode2` para encontrar a solução geral de equações diferenciais de primeira e segunda ordem. No caso que vimos acima a função `method` retorna `constcoeff`, método usado pelo Maxima para resolver a equação diferencial.
Resolver a Eq.(3.11) e encontrar algumas curvas soluções.

$$y'' + 5y' + 6y = 0 \tag{3.11}$$

```
(%i1) /*equação homogênea*/
eq: 'diff(y,x,2)+5*'diff(y,x)+6*y=0;
                                2
                               d y     dy
(%o1)                          --- + 5 -- + 6 y = 0
                                 2     dx
                               dx
(%i2) /*solução geral da equação homogênea*/
ode2(''eq,y,x);
                                     - 2 x         - 3 x
(%o2)                  y = %k1 %e      + %k2 %e
(%i3) /*método usado pelo Maxima para resolver a equação*/
method(%i2);
                                   2
                                  d y     dy
(%o3)          constcoeff(ode2(--- + 5 -- + 6 y = 0, y, x))
                                   2     dx
                                  dx
(%i4) /*solução particular da equação homogênea*/
ic2(%o2,x=0,y=2,diff(y,x)=3);
                                     - 2 x       - 3 x
(%o4)                  y = 9 %e      - 7 %e
(%i5) y1:rhs(%o4);



                                    - 2 x       - 3 x
(%o5)                         9 %e      - 7 %e
(%i6) ic2(%o2,x=0,y=1,diff(y,x)=3);



                                     - 2 x       - 3 x
(%o6)                  y = 6 %e      - 5 %e
```

```
(%i7) y2:rhs(%o6);
                                  - 2 x       - 3 x
(%o7)                          6 %e      - 5 %e
(%i8) ic2(%o2,x=0,y=1,diff(y,x)=2);
                                    - 2 x       - 3x
(%o8)                        y = 5 %e      - 4 %e
(%i9) y3:rhs(%o8);
                                  - 2 x       - 3 x
(%o9)                          5 %e      - 4 %e
(%i10) ic2(%o2,x=1,y=1,diff(y,x)=2);
                                  2 - 2 x       3 -3x
(%o10)                     y = 5 %e        - 4 %e
(%i11) y4:rhs(%o10);
                                2 - 2 x       3 - 3x
(%o11)                       5 %e        - 4 %e
(%i12) /*família de curvas soluções da equação*/
plot2d([y1,y2,y3,y4],[x,0, 5],[y,-3, 3],[gnuplot_preamble,"set terminal
postscript eps;set out 'Dcurvas01.eps'"])\$
```

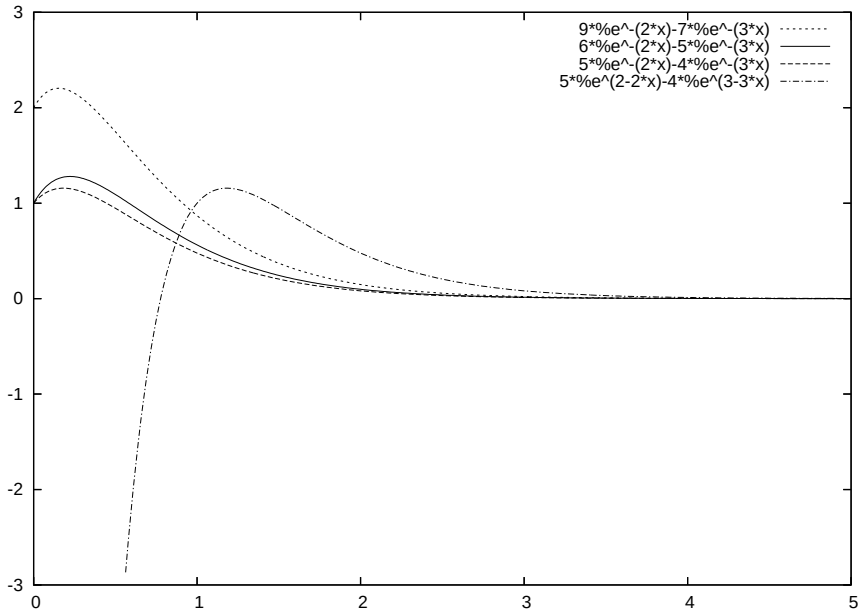

Figura 3.1: Algumas soluções de $y'' + 5y' + 6y = 0$.

3.3 Equações Não-Homogêneas- Métodos dos Coeficientes Indeterminados

São consideradas equações não-homogêneas aquelas que tem a forma da Eq.(3.4), onde, p, q e g são funções contínuas no intervalo considerado e, $g(x) \neq 0$. As equações não-homogêneas tem uma equação homogênea associada, da forma:

$$y'' + p(x)y' + q(x)y = 0 \tag{3.12}$$

A solução geral de uma equação não-homogênea pode ser escrita na forma:

$$y = c_1 y_1(x) + c_2 y_2(x) + Y(x). \tag{3.13}$$

em que y_1 e y_2 formam um conjunto de soluções da equação homogênea associada (3.12), c_1, c_2 são constantes e Y é uma solução específica da solução da equação original, não-homogênea.

3.4 Séries de Potência no Maxima

Uma funcionalidade bastante útil das séries de potências, é a utilização das mesmas como um método alternativo para resolver equações diferenciais. O Maxima disponibiliza muitas funções para tratar expansões em série, como por exemplo, `taylor`(série de taylor) e `powerseries`(série de potência).

Muitas equações diferenciais lineares de segunda ordem, como por exemplo

$$y'' + xy = 0, \tag{3.14}$$

possuem soluções que não podem ser expressas de forma simples, usando funções algébricas, logarítmicas ou trigonométricas. Para obter uma representação que seja válida das soluções de muitas dessas equações, usamos as séries de potências.

Em alguns cursos de Matemática a expansão em série de Taylor é programa da disciplina cálculo II, no curso que fiz, o primeiro contato com essa expansão foi na disciplina cálculo numérico computacional.

Nesse seção vamos usar a série de Taylor para resolver equações diferenciais ordinárias lineares de 2ª ordem. Podemos observar no livro de cálculo avançado do autor Wilfred Kaplan uma pequena explicação sobre as séries de Taylor, onde está desenvolvida uma aplicação sobre equação diferencial ordinária não-linear. Analisando o exemplo dado por Kaplan,

$$y'' = x + y^2 \tag{3.15}$$

em que sua solução em série de Taylor é:

$$y = 1 + 2(x-1) + (x-1)^2 + \frac{5}{6}(x-1)^3 + \frac{1}{2}(x-1)^4 + ..., \tag{3.16}$$

onde as condições iniciais são: $y = 1$ e $y' = 2$ no instante $x = 1$, temos que esse método depende das condições iniciais ou de contornos, ou melhor, do ponto $(x_0, y(x_0))$ e da derivada nesse ponto $(x'_0, y(x'_0))$.

3.4.1 Solução Usando Série de Potência

Seja a equação diferencial linear de segunda ordem homogênea

$$y'' - y = 0 \tag{3.17}$$

Queremos encontrar uma solução em forma de série de potência (polinômio de Taylor) em torno do ponto $x = 0$,

$$y = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + ... + a_nx^n + ... = \sum_{n=0}^{\infty} a_nx^n \tag{3.18}$$

Diferenciando a Eq.(3.18) termo a termo, obtemos

$$y' = a_1 + 2a_2x + 3a_3x^2 + 4a_4x^3 + ... + na_nx^{n-1} + ... \tag{3.19}$$

que também podemos representar por:

$$\sum_{n=1}^{\infty} na_nx^{n-1} \tag{3.20}$$

Diferenciando pela segunda vez a Eq.(3.18), temos

$$y'' = 2a_2 + 6a_3x + 12a_4x^2... + n(n-1)a_nx^{n-2} + ... = \sum_{n=2}^{\infty} n(n-1)a_nx^{n-2} \tag{3.21}$$

Substituindo as Eq.(3.18) e (3.21) na Eq.(3.17), obtemos:

$$(2a_2+6a_3x+12a_4x^2+...+n(n-1)a_nx^{n-2}+...)-(a_0+a_1x+a_2x^2+a_3x^3+a_4x^4+...+a_nx^n+...) = 0 \tag{3.22}$$

ou

$$\sum_{n=2}^{\infty} n(n-1)a_nx^{n-2} + \sum_{n=0}^{\infty} a_nx^n = 0. \tag{3.23}$$

Para associarmos as duas séries, precisamos reescrever de forma que ambas tenham o mesmo termo geral. Assim, devemos mudar o índice de somatório na primeira série substituido n por $n + 2$ e começar em 0 em vez de 2.

$$\sum_{n=0}^{\infty}(n+2)(n+1)a_{n+2}x^n - \sum_{n=0}^{\infty} a_nx^n = 0 \Rightarrow \sum_{n=0}^{\infty}[(n+2)(n+1)a_{n+2} - a^n]x^n = 0. \tag{3.24}$$

Como a soma dessa série(polinômio de Taylor) é nula, então todos os coeficientes da série serão nulos para que essa equação seja satisfeita para todo x, temos aquir uma combinação linear independente. Logo, podemos obter as equações

$$2a_2 - a_0 = 0, \quad 6a_3 - a_1 = 0, \quad 12a_4 - a_2 = 0, ... \tag{3.25}$$

dado pela equação

$$(n+2)(n+1)a_{n+2} - a^n = 0 \tag{3.26}$$

quando $n = 0, 1, 2, 3, 4, ...$ que expressa uma fórmula de recorrência para os coeficientes:

$$a_{n+2} = \frac{a_n}{(n+2)(n+1)} \tag{3.27}$$

Assim,

$$2a_2 - a_0 = 0 \Rightarrow a_2 = \frac{a_0}{2}, \quad 6a_3 - a_1 = 0 \Rightarrow a_3 = \frac{a_1}{6}, \quad 12a_4 - a_2 = 0 \Rightarrow a_4 = \frac{a_0}{24}, ... \tag{3.28}$$

Podemos perceber que a_0 e a_1 são constantes arbitrárias, na verdade, são valores iniciais de y e y' em $x = 0$. As soluções podem ser escritas na seguinte forma:

$$y = a_0[1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} + ... + \frac{x^{2n}}{(2n)! + ...}] + a_1[x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} + ... + \frac{x^{2n+1}}{(2n+1)! + ...}] \tag{3.29}$$

$$\sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!} + \sum_{n=0}^{\infty} \frac{x^{2n+1}}{(2n+1)!} \tag{3.30}$$

Aqui temos a solução geral. Para encontrar a solução particular devemos ter ondições iniciais para a equação.

Mesmo para uma equação diferencial simples, é necessário muito cálculo para resolvê-la manualmente por série. Digo-lhe que é possível e, sem dúvidas, muito interessante, usar o Software Maxima para fazer estes cálculos. Então, deixo como uma curiosidade, para que alguém tenha interesse em desenvolver algo a respeito.

Consideração Final

De forma alguma um programa[1] de computador pode ser visto como uma forma de substituir por completo os cálculos manuais e, principalmente, o esforço de compreendermos os conceito matemáticos. Tal programa deve ser visto como uma ferramenta de grande ajuda no processo de aprendizagem. Porém, o domínio do computador assim como um programa computacional, *Maxima*, são fundamentais quando nos deparamos com problemas que não podem ser resolvidos simplesmente com lápis e papel por demorar muito tempo para serem resolvidos se comparado com o tempo que um programa de computador levaria para resolver o mesmo problema.
Erros também estão presentes em qualquer software, possuem seus limites e, quando alcançados, é possível obtermos respostas incorretas.

[1] Maxima

Referências Bibliográficas

[1] BUTKOV, E.,Mathematical Physics. New York:Addison-Wesley Publishing Company,1968.

[2] Boyce, William. E e Diprima, Richard C. *Equações Diferenciais Elementares e Problema de Valores de Contorno*. 8ª ed. Rio de janeiro: LTC, 2006.

[3] Praciano-Pereira, T. *Notas de Aula de Equações Diferenciais Ordinárias* http://www.edo-metodos.sobralmatematica.org/textos

[4] Silva, M. Ilsangela. Primeiros Exemplos de Equações Diferenciais. Disponível em: www.sobralmatematica.org/preprints/ilsangela02.pdf Acesso em: 12 maio.2009

[5] Silva, M. Ilsangela; Oliveira, F. Vagner. *Equações Diferenciais Exatas.* http://www.sobralmatematica.org/preprints

[6] http://maxima.sourceforge.net/

Apêndices A

Notas Históricas[2]

As equações diferenciais começaram com o estudo de cálculo por *Isaac Newton(1642-1727)* e *Gottfried W. Leibniz(1646-1716)* durante o século XVII. Newton atuou pouco na área de equações diferenciais, mas, seu desenvolvimento do cálculo e a elucidação dos princípios básicos da mecânica forneceram a base para a aplicação das equações diferenciais no século XVIII, especialmente por *Euler*.

Newton desenvolveu um método para resolver a equação $\frac{dy}{dx} = f(x, y)$, no caso em que $f(x, y)$ é um polinômio em x e y, usando séries infinitas.

Leibniz descobriu o método de separação de variáveis em 1691, a redução de equações homogêneas a equações separáveis em 1691 e o procedimento para resolver equações lineares de primeira ordem em 1694.

Os irmão Bernoulli, *Jakob (1654-1705) e Johann (1667-1748)*, contribuiram muito para o desenvolvimento de métodos para resolver equações diferenciais e ampliar o campo de suas aplicações. Com a ajuda do cálculo, resolveram diversos problemas em mecânica, formulando-os como equações diferenciais. *Jakob Bernoulli* resolveu a equação diferencial $y' = [a^3/(b^2y - a^3)]^{1/2}$ em 1690. No ano de 1694 *Johann Bernoulli* foi capaz de resolver a equação $dy/dx = y/ax$.

Um problema que ambos os irmão resolveram e gerou atrito entre eles foi o problema da braquistócrona resolvido também por Newton, Leibniz e L'Hôpital.

Equações Diferenciais

Equações diferenciais é um tópico da matemática que tem aplicações em praticamente todos os ramos da ciência. Muitos problemas que encontramos na Física, Quimica, Economia e Biologia podem ser reduzidos a equações diferenciais. Por serem equações que contém derivadas de funções, precisamos ter uma boa noção de alguns tópicos estudados no Cálculo Diferencial e Integral para estudá-las. Na escola, quando estudamos álgebra, trigonometria somos levados a resolver equações do tipo

$$x^2 + 6x + 9 = 0 \tag{3.31}$$

na variável x, ou seja, encontrar um valor real ou complexo para a variável x. Este

processo é parecido quando queremos encontrar a solução de uma equação diferencial, sendo que, em vez de encontrarmos números para ser a solução, encontramos uma função. Por exemplo, seja a equação diferencial

$$y' = 2x \tag{3.32}$$

a solução geral dessa equação é a função

$$y = x^2 + c. \tag{3.33}$$

Esta equação é considerada uma das mais simples possíveis de encontrar a solução.

Definição:

Equação Diferencial é uma equação que envolve uma função desconhecida, incógnita da equação, e uma ou mais de suas próprias derivadas.

Nosso primeiro contato com equações diferenciais foi quando resolvemos a seguinte equação estudada no Cálculo I:

$$F' = f \Leftrightarrow \int f(x)dx = F(x) \tag{3.34}$$

Resolvendo a integral de $f(x)$ encontramos uma primitiva F da função dada, ou melhor, encontramos uma função F tal que, $F' = f$. Sendo assim, a primitiva F nada mais é que uma solução para esta equação diferencial.
Para ilustrar temos:

$$F'(x) = sin(x) \tag{3.35}$$

que é equivalente a

$$F(x) = \int sin(x) = -cos(x) + c, \tag{3.36}$$

devido a constante c, esta equação diferencial possui infinitas soluções.

Apêndices B

Séries no Maxima

A Função `sum()`

A função `sum()` é usada para representar uma expanssão em série num somatório com a sintaxe: `sum(expressao, var, ini, fim)`. No Maxima representamos assim:

```
%i1) sum(c[n]*x^n,n,0,inf);
                                    inf
                                    ====
                                    \          n
(%o1)                                >    c  x
                                    /      n
                                    ====
                                    n = 0
```

A Função `diff()`

A função `diff()` como vimos acima, é usada no Maxima para diferenciar expressões mas também, podemos usar a mesma para diferenciar expansões em séries com a sintaxe: `diff(expressao, var, ordem da derivada)`. Vamos diferenciar a série

$$\sum_{n=0}^{\infty} c_n x^n$$

no Maxima:

```
%i1) sum(c[n]*x^n,n,0,inf);
                                  inf
                                  ====
                                  \          n
(%o1)                              >    c  x
                                  /      n
                                  ====
                                  n=0
(%i2) diff(%o1,x,2);
                             inf
```

```
                              ====
                              \                    n - 2
(%o2)                          >     (n - 1) n c   x
                              /                 n
                              ====
                              n = 0
(%i3) diff(%o1,x,3);
                         inf
                         ====
                         \                             n - 3
(%o3)                     >     (n - 2) (n - 1) n c   x
                         /                          n
                         ====
                         n = 0
```

Que são as derivadas de segunda e terceira ordem.

A Função `taylor()`

A função `taylor()` é usada quando queremos apresentar uma expressão em série de Taylor para uma dada função. A sintaxe é `taylor(f(x), var, ini, fim)`. Encontramos a expansão da função $sin(x)$ no ponto 0 e grau 9 e da seguinte forma:

```
(%i1) taylor(sin(x),x,0,9);
                            3    5      7       9
                           x    x      x       x
(%o1)/T/              x - -- + --- - ---- + ------ + . . .
                           6    120   5040   362880
```

A Função `solve_rec()`

Essa função é usada para resolver a relação de recorrência no processo de resolução de uma equação diferencial por meio de séries de potências. Antes de utilizarmos tal função é preciso ativá-la com o auxílio da função `load`. A sintaxe é `solve_rec(relacao de recorrencia, a[n])`.