UNIVERSIDADE FEDERAL DO TRIÂNGULO MINEIRO

MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL

Joaquim Barbosa Júnior

Resolução de problemas usando o wxMaxima

Uberaba-MG

2013

JOAQUIM BARBOSA JÚNIOR

Resolução de problemas usando o wxMaxima

Dissertação, apresentada ao curso de Mestrado Profissional em Matemática em Rede Nacional-PROFMAT, como parte das atividades para obtenção do título de Mestre em Matemática da Universidade Federal do Triângulo Mineiro - UFTM, Departamento de Matemática.

Uberaba

2013

Catalogação na fonte: Biblioteca da Universidade Federal do Triângulo Mineiro

B198r

Barbosa Júnior, Joaquim
Resolução de problemas usando o wxMaxima / Joaquim Barbosa Júnior. -- 2013.
56 f. : il., fig., graf., tab.

Dissertação (Mestrado Profissional em Matemática em Rede Nacional) -- Universidade Federal do Triângulo Mineiro, Uberaba, MG, 2012.
Orientador: Prof. Me. Wellington Barros e Barbosa
Coorientador: Prof. Dr. Osmar Aléssio

1. Matemática aplicada à computação. 2. Lógica simbólica e matemática. 3. Solução de problemas. 4. Maxima (Programa de computador). 5. Software livre. I. Barbosa, Wellington Barros. II. Universidade Federal do Triângulo Mineiro. III.Título.

CDU 519.6

Joaquim Barbosa Júnior

Resolução de problemas usando o wxMaxima

Dissertação, apresentada ao curso de Mestrado Profissional em Matemática em Rede Nacional-PROFMAT, da Universidade Federal do Triângulo Mineiro, como parte das atividades para obtenção do título de Mestre em Matemática.

15 de março 2013

Banca Examinadora

Prof. Ms. Welington Barros e Barbosa
Orientador
Universidade Federal do Triângulo Mineiro

Prof. Dr. Osmar Aléssio
Co-orientador
Universidade Federal do Triângulo Mineiro

Moisés R. C. Monte

Prof. Ms. Moisés Rodrigues C. Do Monte
Universidade Federal de Uberlândia

A minha esposa Mária Rúbia Barbosa
e minha filha Mariannah Silva Barbosa.

AGRADECIMENTOS

– Agradeço a Deus acima de tudo.

– Ao meu orientador Ms. Welington Barros e Barbosa.

– Ao meu co-orientador Dr. Osmar Aléssio.

– À coordenadora do PROFMAT polo UFTM, Drª. Marcela Luciano Vilela de Souza

– À CAPES

– Agradeço ao IMPA e a SBM pela criação do PROFMAT, que proporcionou-me a oportunidade de tornar-me um mestre.

– Ao meus companheiros de muitas viagens Mário e Neilon.

A mente que se abre a uma nova ideia jamais voltará ao seu tamanho original.

Albert Einstein

RESUMO

Este trabalho ilustra as potencialidades do software wxMaxima usando o sistema operacional UBUNTU-Linux em um ambiente gráfico KDE. Mostra com trabalhar com variáveis, listas, equações, funções e gráficos de duas ou três dimensões. Outros temas do ensino superior como limites, derivadas, integrais, matrizes, sistemas lineares, autovalores, autovetores e equações diferenciais também são abordados. WxMaxima é um software livre e pode fazer cálculos usando a forma simbólica e/ou numérica. Pode também ser utilizado em sistemas operacionais como MacOS e Windows. A dissertação também apresenta sugestões sobre como e onde o aplicativo pode ser utilizado.

Palavras-chave: Resolução de problemas, Maxima, wxMaxima, Macsyma, Software livre, Matemática computacional simbólica, CAS

ABSTRACT

This work shows the potential of to use the software wxMaxima in UBUNTU-Linux operating system with a graphical environment KDE. Shows how to work with variables, lists, equations, functions and graphs in two or three dimensions. Other issues in higher education as limits, derivatives, integrals, matrices, linear systems, eigenvalues, eigenvectors and differential equations are also discussed. WxMaxima is free software and can perform calculations using the symbolic form and / or numerical. Can also be used on operating systems like Windows and MacOS. The dissertation also presents suggestions on how and where the application can be used.

Keywords: Troubleshooting, Maxima, WxMaxima, Macsyma, Free software, Symbolic computational mathematics, CAS

Conteúdo

Lista de Figuras

1 INTRODUÇÃO

Sabe-se, atualmente, que com os avanços tecnológicos o uso de computadores se tornou indispensável. Devido ao grande número de informações do mundo atual bancos, hospitais, aeroportos e lojas não podem mais funcionar sem o uso de tais máquinas. Na educação, professores podem usar essa tecnologia para enriquecer suas aulas proporcionando aos alunos mais uma ferramenta de aprendizagem que pode ser muito eficaz se bem utilizada. Os computadores também tem extrema importância para muitos profissionais da área de exatas, no que diz respeito a resolução de problemas que envolvam cálculos complexos ou até mesmo muito extensos. Neste sentido, os computadores e seus softwares se tornam poderosas ferramentas de ensino, aprendizagem e resolução de problemas. Pensando nisso, este trabalho apresenta um material complementar a alunos dos cursos de graduação ou pós-graduação, que possuam as disciplinas de Álgebra Linear, Cálculo Diferencial e Integral, Equações Diferenciais e Cálculo Numérico em seus planos de curso. Esta dissertação pretende analisar alguns problemas típicos das disciplinas citadas apresentando resoluções algébricas e/ou numéricas utilizando recursos computacionais.

Para as soluções computacionais será utilizado o software wxMaxima 11.08.0 no sistema operacional Ubuntu-Linux com ambiente gráfico KDE. Para usuários que não são familiarizados com o Linux, existe também versões do wxMaxima para Windows ou Macintosh. Os arquivos do Windows ou MacOS para download e instalação podem ser encontrados no seguinte site http://sourceforge.net/projects/maxima/files/. No Ubuntu-Linux, em uma janela do terminal, digite "sudo apt-get install wxmaxima" junto com a senha de super usuário para instalar o programa. O leitor não precisará pagar licenças para poder utilizá-lo pois é um programa livre, razão pela qual foi escolhido para tal estudo. O autor recomenda que o leitor esteja sempre com o software ao lado para que, na leitura do texto, vá comprovando os comandos e aprendendo a usá-los de forma completamente prática. Maxima é uma linguagem computacional baseada em Lisp para a manipulação de expressões simbólicas e numéricas. Manipula expressões, gráficos, cálculos de integração e diferenciação,matrizes, limites, vetores,etc. Pode obter resultados de forma simbólica,

Figura 1: Logo do Maxima.

sendo seu ponto mais interessante. Para se resolver, por exemplo, a equação $x^2-2=0$ um outro programa qualquer pode retornar como resultado $x=1,4142$ ou $x=-1,4142$. Por mais que se aumente a precisão do resultado, sempre são perdidas infinitas casas decimais no processo. Utilizando a forma simbólica do wxMaxima, para a mesma equação, as raízes seriam $x=\sqrt{2}$ ou $x=-\sqrt{2}$. Neste caso, não há nenhuma perda por arredondamento ou truncamento de resultados.

Maxima surgiu do código fonte do Macsyma desenvolvido pelo MIT (Massachusetts Institute of Technology) entre 1968 e 1982. Em 1982 uma cópia do Macsyma foi entregue ao departamento de energia americano e outra ao professor William F. Schelter, da universidade do Texas. Esta cópia foi chamada de Macsyma DOE (Departament of energy). Três anos antes de sua morte em 2001, professor Schelter conseguiu licença do departamento de energia para tornar livre o código fonte do software . Assim outras pessoas poderiam continuar desenvolvendo o programa. E continuam até hoje. Essa versão livre do Macsyma é conhecida como Maxima.

A dissertação segue assim apresentada: O capítulo 2, trata de regras e comandos básicos para definição de variáveis, listas, equações e funções. No capítulo 3 é feita a plotagem de gráficos bidimensionais e tridimensionais de forma explícita, implícita e paramétrica. No final deste mesmo capítulo, aborda-se um método para se resolver equações utilizando-se gráficos. Temas do cálculo diferencial e integral e álgebra linear são trabalhados no capítulo 4. Neste são calculados limites, derivadas, integrais, operações com matrizes, resolução de sistemas lineares, autovalores, autovetores utilizando auxílio computacional. Há uma breve revisão do conteúdo de equações diferenciais na primeira seção do capítulo 5, nas outras seções são resolvidas equações diferenciais e construídos campos de direções usando o wxMaxima. No último capítulo, são feitas algumas conclusões e sugestões a respeito do software destacando suas potencialidades.

2 REGRAS BÁSICAS DO WXMAXIMA

2.1 VARIÁVEIS E CÁLCULOS ALGÉBRICOS

O wxMaxima recebe comandos ou valores digitados na linha %i(n), os executa e mostra o resultado na linha %o(n), onde n é um número natural. A letra “i” em %i(n) significa input(do inglês: entrada) e a letra “o” em %o(n) significa output (do inglês:saída). Pode-se atribuir valor a uma variável no software através do sinal de dois pontos (:). Logo, b:5, significa que b vale 5 ou b = 5. Todo comando no wxMaxima é executado somente após a combinação de teclas “shift + enter” ser pressionada. A tecla “enter” é utilizada junto com o sinal de ponto e vírgula (;) quando se quer inserir vários comandos ao mesmo tempo. Por exemplo, a:1 ; b:-3 ; c:1/5 diz ao programa que as variáveis a, b e c são respectivamente 1, -3 e $\frac{1}{5}$. Pode-se então fazer cálculos aritméticos usando os símbolos da tabela 1 com a atribuição de valores acima. (figura 2)

A raiz quadrada é feita pelo comando “sqrt(<número>)”. Para o cálculo das raízes com os demais índices, deve ser utilizada a potenciação. Assim,desejando-se calcular $\sqrt[4]{16}$, usa-se a potenciação $16^{\frac{1}{4}}$. Para $\sqrt[3]{216}$,$216^{\frac{1}{3}}$. Generalizando, deve ser digitado “<radicando>**(1/<índice>)”. As figuras 3 e 4, mostram alguns exemplos de cálculos de raízes. Como dito anteriormente, uma das potencialidades do software é o cálculo simbólico, que é exibido em formato LaTex. Note que isso ocorreu em casos de raízes inexatas, mas se o usuário desejar uma resposta numérica, poderá utilizar o comando

Símbolos ou comandos	Operação
+	adição
-	subtração
*	multiplicação
/	divisão
** ou ˆ	potenciação
log(x)	logaritmo neperiano
log(x)/log(b)	log.de x na base b

Tabela 1: Operações básicas no wxMaxima

```
(%i1) a:1; b:-3; c:1/5;
(%o1) 1
(%o2) -3
(%o3) 1/5

(%i4) a+b+c;
(%o4) -9/5

(%i5) 2*a-3*b+5*c;
(%o5) 12
```

Figura 2: Atribuindo variáveis e realizando cálculos.

```
(%i1) sqrt(49);
(%o1) 7

(%i2) 216**(1/3);
(%o2) 6

(%i3) 16**(1/4);
(%o3) 2
```

Figura 3: Cálculo de raízes exatas.

```
(%i4) sqrt(52);
(%o4) 2 sqrt(13)

(%i5) %,numer;
(%o5) 7.211102550927978

(%i6) 32**(1/3);
(%o6) 2 4^(1/3)

(%i7) %,numer;
(%o7) 3.174802103936399
```

Figura 4: Cálculo de raízes não exatas com resultado simbólico ou numérico.

```
(%i1) log(6),numer /* logaritmo de 6 na base e */;
(%o1) 1.791759469228055

(%i2) log(%e)/* %e é a constante de Euler */;
(%o2) 1

(%i3) log(64)/log(4),numer/* logaritmo de 64 na base 4 */;
(%o3) 3.0

(%i4) log(2)/log(10), numer/* logaritmo de 2 na base 10 */;
(%o4) 0.30102999566398
```

Figura 5: Cálculo de logaritmos.

"%,numer" que passa para numérica a última saída simbólica. [1] Para o cálculo de logaritmos neperianos (na base **e**) é utilizado o comando log(<número>). Como o software não possui um comando próprio para logaritmos em outras bases, deve-se utilizar uma relação

[1] O símbolo "%", sozinho, é usado sempre que se queira referir a última saída do programa. Assim na entrada "%i7", (figura 4), "%" se refere a saída "%o6".

bastante conhecida de mudança de base, $log_b(x) = \frac{log_e(x)}{log_e b}$, para solucionar o problema. Na linha de comando: log(<número>)/log(<base>).(figura 5) Os cálculos também podem ser organizados utilizando-se comentários que não são interpretados pelo programa quando escritos entre os sinais /* e */. Por várias vezes, neste texto, o autor utiliza este recurso para melhorar a compreensão dos comandos contidos nas figuras.

2.2 LISTAS

Muitas vezes, como será visto posteriormente, o programa trabalhará com listas. Uma lista é uma coleção de números, variáveis, funções, matrizes, vetores, etc ... Sempre vêm escritos entre colchetes e com todos seu elementos separados por vírgula. Na figura 6 há exemplos de listas e operações que podem ser feitas utilizando-as.

```
(%i1) listaexemplo:[1,sqrt(5),6,1/3,9]/*Lista com nome listaexemplo*/;
(%o1) [1, √5, 6, 1/3, 9]

(%i2) 3*listaexemplo /*O triplo de toda a lista.*/;
(%o2) [3, 3 √5, 18, 1, 27]

(%i3) (1/6)*listaexemplo /*Um sexto de todos de todos os elementos.*/;
(%o3) [1/6, √5/6, 1, 1/18, 3/2]

(%i4) %,numer /*Lista anterior na forma numérica*/;
(%o4) [0.16666666666667, 0.37267799624996, 1, 0.055555555555556, 1.5]
```

Figura 6: Trabalhando com listas.

2.3 RESOLVENDO EQUAÇÕES

O software consegue resolver muitas equações de forma simbólica. Uma equação é solucionada através do comando solve(<equação>, < variável>), digitando no campo <variável> a variável a ser isolada. No campo <equação> os membros da equação devem ser separados pelo sinal de igual, diferentemente da atribuição de valores à variáveis que é feito usando o sinal de dois pontos. Como exemplo, a equação $\frac{p}{4} - \frac{r}{9} + k^2 = 5$ foi resolvida na figura 8 em relação às variáveis p, r e k. Algumas equações do segundo grau também foram resolvidas na figura 7. Para fins agilizar a digitação, a equação contida na figura 8, recebeu um nome, “eq1”, através do sinal de dois pontos. Desta forma, não é necessário ficar digitando a equação inúmeras vezes, bastando apenas chamar por seu nome.

(%i1) `solve(x**2-5*x+6=0,x) /* duas raízes reais */;`

(%o1) $[x=3, x=2]$

(%i2) `solve(x**2+2*x+1=0,x) /* uma raiz real */;`

(%o2) $[x=-1]$

(%i3) `solve(x**2+x+1=0,x)     /* duas raízes complexas */;`

(%o3) $[x=-\frac{\sqrt{3}\,\%i+1}{2},\ x=\frac{\sqrt{3}\,\%i-1}{2}]$

Figura 7: Resolvendo equações do segundo grau

(%i4) `eq1:p/4-r/9+k**2=5 /* equações literais */;`

(%o4) $-\frac{r}{9}+\frac{p}{4}+k^2=5$

(%i5) `solve(eq1,p)    /* solução em relação a p */;`

(%o5) $[p=\frac{4\,r-36\,k^2+180}{9}]$

(%i6) `solve(eq1,r)     /* solução em relação a r */;`

(%o6) $[r=\frac{9\,p+36\,k^2-180}{4}]$

(%i7) `solve(eq1,k)    /* solução em relação a k */;`

(%o7) $[k=-\frac{\sqrt{4\,r-9\,p+180}}{6},\ k=\frac{\sqrt{4\,r-9\,p+180}}{6}]$

Figura 8: Resolvendo equações literais

2.4 DEFININDO FUNÇÕES DE UMA OU MAIS VARIÁVEIS

O wxMaxima possui também inúmeros comandos para trabalhar com funções. Uma função é definida no programa através do sinal de dois pontos (:) junto com o sinal de de igual(=). Assim, escrevendo f(x) := 3*x+1, está definindo-se a função $f(x) = 3x + 1$. Com a função definida, pode-se calcular imagens digitando simplesmente f(1), f(2), f(-5), f(3/4), etc... Além disso, é possível determinar as raízes da função através do comando solve(<função> = 0, <variável>), onde <função> deve ser preenchido com a função f(x) e <variável> com a variável que se pretende isolar. Logo deve ser digitado solve (f(x) = 0, x). (figura 9) Observe também o exemplo envolvendo uma função do segundo grau $g(x) = x^2 + x + 1$ na figura 10. Primeiramente, a função é definida, tem algumas imagens calculadas, determinando-se posteriormente as raízes. Note que foi usado o comando %pi para se referir ao número irracional π. Nas raízes da figura 10, %i significa a unidade imaginária, mostrando que a função g(x) possui raízes complexas. Para funções de mais de uma variável, o processo é feito de forma análoga. Na figura 11, tem-se a função $r(x, y, z) = x^2 + y^3 - \frac{z}{3}$ e o cálculo de algumas imagens. Duas equações foram resolvidas na figura 12 usando a função r, sendo a primeira equação

$$r(s, 2, -9) = 0 \Rightarrow s^2 + 8 + 3 = 0 \Rightarrow s^2 = -11 \Rightarrow s = \pm\sqrt{11}i$$

e a segunda

$$r(\sqrt{t}, t, 0) = 0 \Rightarrow t + t^3 + 0 = 0 \Rightarrow t(1 + t^2) = 0 \Rightarrow t \in \{0, -i, +i\}.$$

O cálculo de limites, derivadas e integrais de funções será detalhado em seções posteriores.

```
(%i1) f(x):=3*x+1;
(%o1) f(x):=3 x+1
(%i2) f(1);
(%o2) 4
(%i3) f(2);
(%o3) 7
(%i4) f(5);
(%o4) 16
(%i5) f(3/4);
(%o5) 13/4
(%i6) solve(f(x)=0,x);
(%o6) [x=-1/3]
```

Figura 9: Cálculo de imagens e zero da função f(x) = 3x + 1

```
(%i1) g(x):=x^2+x+1;
(%o1) g(x):=x^2+x+1
(%i2) g(-1/5);
(%o2) 21/25
(%i3) g(sqrt(3));
(%o3) sqrt(3)+4
(%i4) g(0);
(%o4) 1
(%i5) g(%pi);
(%o5) %pi^2+%pi+1
(%i6) solve(g(x)=0,x);
(%o6) [x=-(sqrt(3)*%i+1)/2, x=(sqrt(3)*%i-1)/2]
```

Figura 10: Cálculo de imagens e zero da função $g(x) = x^2 + x + 1$

```
(%i1) r(x,y,z):=x^2 + y^3 - (z/3);
(%o1) r(x,y,z):=x^2+y^3-z/3
(%i2) r(1,2,3);
(%o2) 8
(%i3) r(-1,0,-6);
(%o3) 3
(%i4) r(sqrt(a),a^(1/3),12*a);
(%o4) -2 a
```

Figura 11: Cálculo das imagens $r(1,2,3)$, $r(-1,0,-6)$ e $r(\sqrt{a},\sqrt[3]{a},12a)$

```
(%i5) r(s,2,-9)=0;
(%o5) s^2+11=0
(%i6) solve(%);
(%o6) [s=-sqrt(11) %i, s=sqrt(11) %i]
(%i7) r(sqrt(t),t,0)=0;
(%o7) t^3+t=0
(%i8) solve(%);
(%o8) [t=-%i, t=%i, t=0]
```

Figura 12: Resolvendo as equações $r(s,2,-9)=0$ e $r(\sqrt{t},t,0)=0$

3 PLOTAGEM DE GRÁFICOS

3.1 DUAS DIMENSÕES

O wxMaxima também constrói gráficos de funções bidimensionais. Para isso há quatro comandos básicos:

(i) plot2d

(ii) wxplot2d

(iii) draw2d

(iv) wxdraw2d

3.1.1 Funções explícitas

O comando plot2d plota gráficos de funções explícitas de duas dimensões em uma janela separada da janela do wxMaxima, utiliza-se a seguinte sintaxe:

plot2d([<lista de funções>],[x,<xmin>,<xmax>][y,<ymin>,<ymax>])

Os campos <xmin>, <xmax>, <ymin> e <ymax> indicam as dimensões da janela de visualização do gráfico. Assim, a janela de visualização é um retângulo com vértices opostos, (<xmin>,<ymin>) e (<xmax>,<ymax>). No campo <lista de funções> devem ser colocadas todas as funções a serem plotadas observando duas coisas:

(a) Todas as funções devem estar entre colchetes e separadas por vírgulas. (Lista)

(b) Só deve ser digitado o segundo membro de cada função.

Na figura 13 foram plotados os gráficos de $f(x) = x \cdot sen(x)$ e $g(x) = sen(x)$ em uma única janela usando o plot2d, já na figura 14 foi usado o wxplot2d para que o leitor note as diferenças. Foi criada uma lista $[x \cdot sen(x)\ ,\ sen(x)]$ e utilizada uma janela com vértices (-4,-2) e (4,2) nos dois casos. A diferença entre os comandos plot2d e wxplot2d é que no segundo o gráfico é exibido dentro do próprio ecrã do wxMaxima.

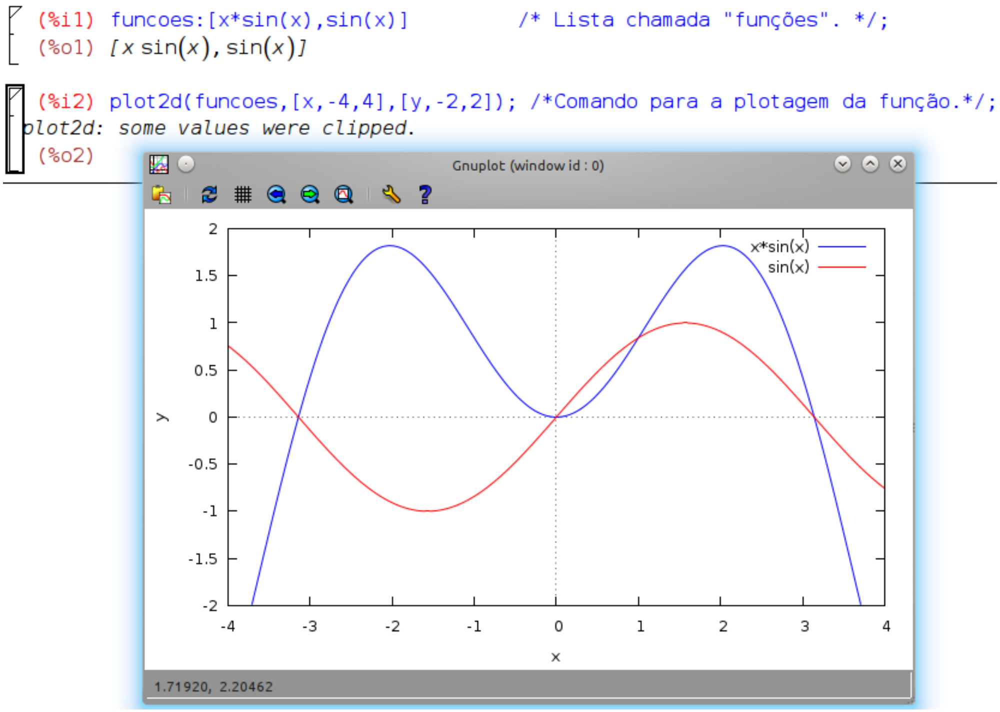


Figura 13: Gráfico de funções explícitas usando o plot2d.

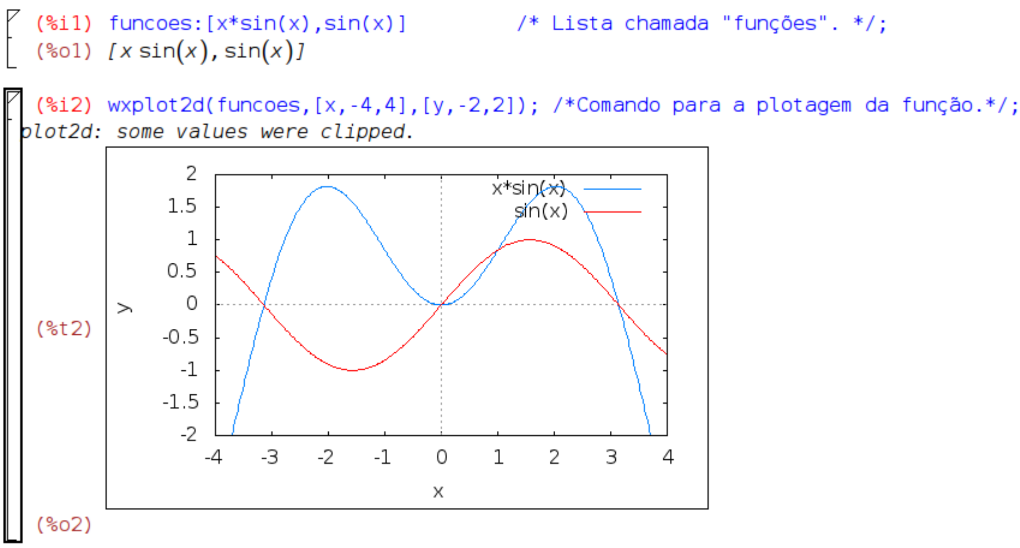


Figura 14: Gráfico de funções explícitas usando o wxplot2d.

3.1.2 Funções implícitas e paramétricas

Para o plotagem de funções implícitas e paramétricas, em duas dimensões, é utilizado o comando draw2d ou wxdraw2d, que são carregados no pacote draw[1] digitando-se "load(draw)". O wxdraw2d possui as seguintes sintaxes:

(i) wxdraw2d(implicit(<função implícita>, x,<xmin>,<xmax>, y,<ymin>, <ymax>))

(ii) wxdraw2d(parametric($< X >$, $< Y >$, t,<tmin>,<tmax>))

Em (i), <função implícita> não é uma lista mas uma única função. Em (ii), $< X >$ e $< Y >$ são funções do parâmetro t. Se o usuário quiser mais gráficos em uma única janela, deve usar um comando implict ou parametric para cada função dentro do mesmo comando draw2d. Os outros parâmetros têm o mesmo significado que aqueles citados no plot2d. As diferenças entre o draw2d e wxdraw2d são análogas às do plot2d e wxplot2d. Resumindo, implicit indica ao wxdraw2d que a função digitada deve ser considerada implícita enquanto parametric(ii) indica que a função deve ser paramétrica. Exemplos do comando draw2d para funções implícitas estão nas figuras 15, 16 , 17. Na figura 18, foi utilizada uma cicloide de equações paramétricas

$$\begin{cases} X(t) = t - sen(t) \\ Y(t) = 1 - cos(t) \end{cases} \qquad t \in [0, 8\pi].$$

Foi utilizado também o comando "nticks" que indica o número de pontos que devem ser calculados para a construção do gráfico. Já "xrange" e "yrange" servem para dar um melhor ajuste a janela.

[1]Um pacote contém um conjunto de comandos nele organizados para um determinado fim. Draw é um pacote especializado em gráficos de duas ou três dimensões.

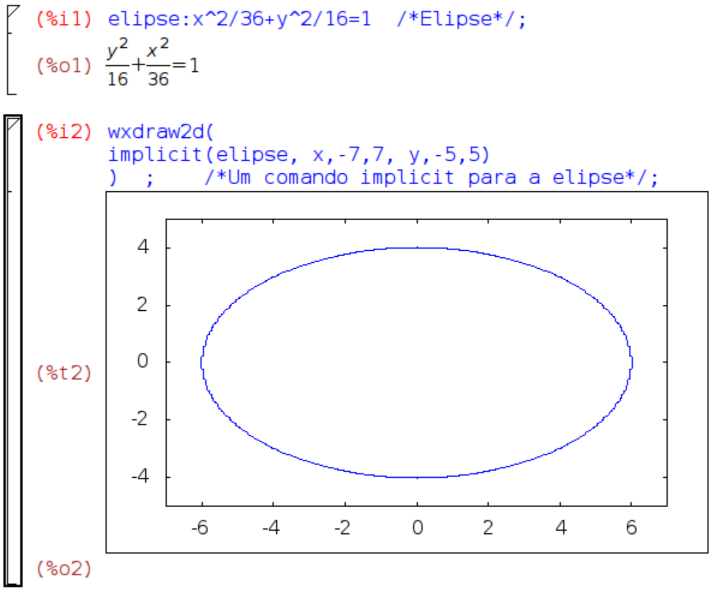


Figura 15: Comando wxdraw2d em uma equação de elípse.

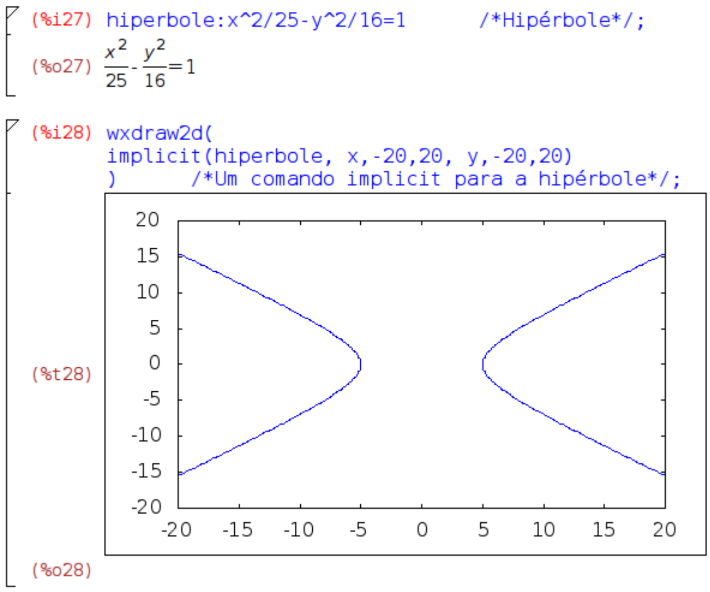


Figura 16: Comando wxdraw2d em uma equação de hipérbole.

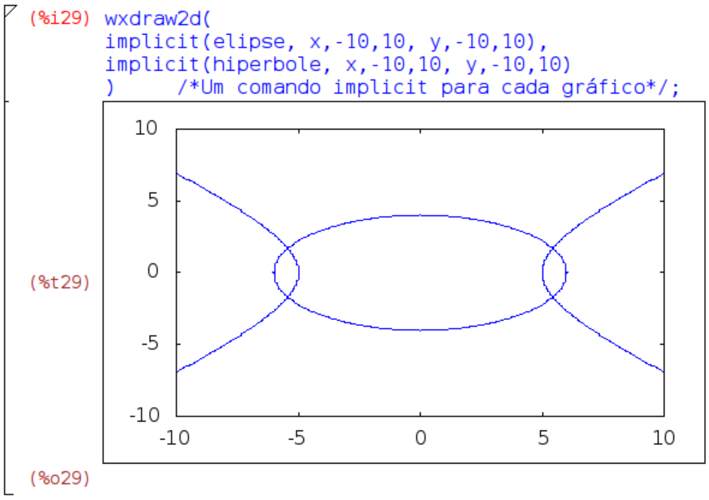


Figura 17: Comando wxdraw2d para duas funções implícitas.

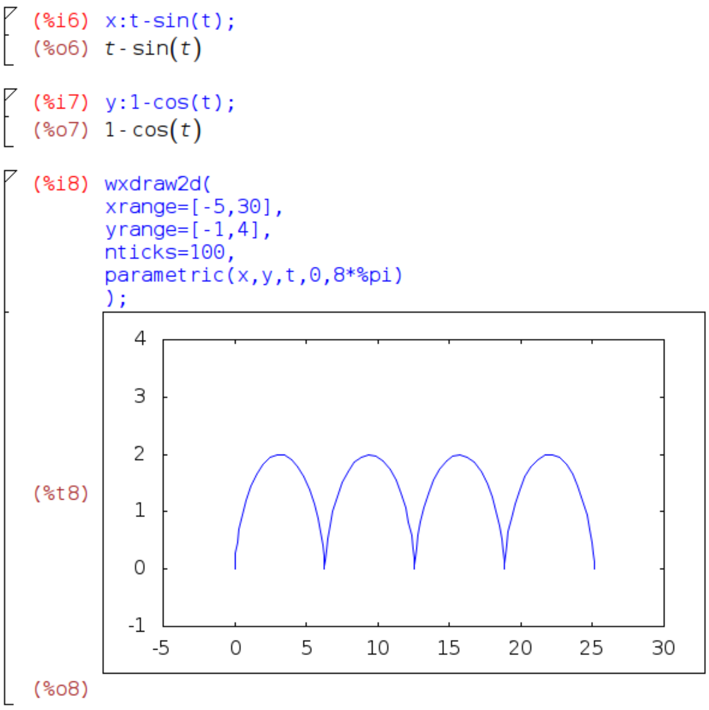


Figura 18: Comando wxdraw2d para função paramétrica.

3.2 TRÊS DIMENSÕES

No caso dos gráficos em três dimensões, usa-se:

(i) plot3d

(ii) wxplot3d

(iii) draw3d

(iv) wxdraw3d

A diferença entre os comandos sem o "wx", no espaço, é que os gráficos podem ser rotacionados, aproximados ou distanciados enquanto no segundo caso isso,com o "wx", isso não acontece. Para o comando draw3d(sem "wx") deve-se carregar sempre o pacote draw através da sintaxe load(draw).

3.2.1 Funções explícitas

Será usado, nesta seção, plot3d[2] para funções explícitas com a seguinte sintaxe que é completamente análoga ao wxplot2d:

plot3d([<Lista de funções>, [x,<xmin>,<xmax>], [y,<ymin>,<ymax>]])

Plot3d dá um bom acabamento em superfícies tridimensionais, atribuindo cores diferentes as partes da superfície dependendo do intervalo da variável z, mas tem suas limitações na hora de desenhar funções implícitas. Como exemplo de função explícita, considere o paraboloide elíptico $P : z = 3x^2 + 4y^2$ secionada pelo plano paralelo ao plano π_{XY}, $\pi : z = 200$ (figura 19). Observe que no argumento, <Lista de funções>, a variável "z" não deve ser digitada, somente o segundo membro. Para o caso de funções paramétricas e implícitas, usa-se o draw3d, que é um comando mais completo e será visto na próxima seção.

[2] A sintaxe do wxplod3d é idêntica.

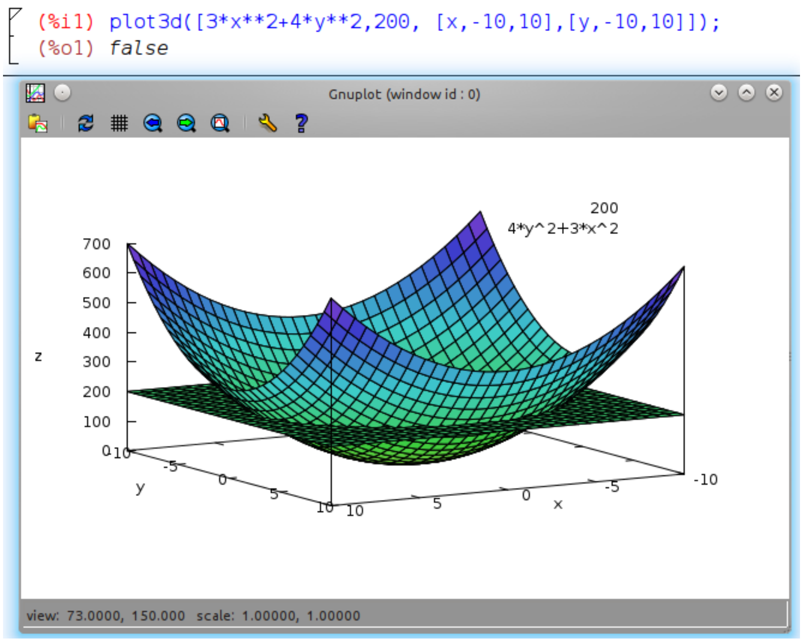


Figura 19: Exemplo plot3d em paraboloide e plano

3.2.2 O draw3d

O comando é mais completo que o plot3d para criar gráficos tridimensionais. Consegue desenhar, em uma mesma janela, varias funções escritas de diversas formas, sejam elas paramétricas, implícitas e até mesmo as explícitas feitas pelo plot3d. Para isso, deve-se entender quatro subcomandos básicos que precisam ser digitados dentro dos parêntesis do draw3d. São eles:

(i) explicit(<função>, x,<xmin>,<xmax>, y,<ymin>,<ymax>)

(ii) parametric($< X >, < Y >, < Z >, t, < tmin >, < tmax >$)

(iii) parametric_surface($< X >, < Y >, < Z >, u, < umin >, < umax >, v, < vmin >,$ $< vmax >$))

(iv) implicit(<função>, x,<xmin>,<xmax>, y,<ymin>,<ymax>,z,<zmin>, <zmax>)

O item (i) é usado para funções explícitas onde <função> é apenas o segundo membro da função. Na figura 20, está representado um paraboloide hiperbólico de equação $z = x^2 - y^2$ e cor laranja selecionada, pelo subcomando "color=orange". A tabela 2 mostra algumas opções de $< cor >$ para "color=$< cor >$" que podem ser usadas no draw 2d ou 3d. Já (ii) serve para funções paramétricas onde <X>, <Y> e <Z> são funções de um único parâmetro t. A figura 21 contem exemplo da hélice de equações paramétricas

$$\begin{cases} X(t) = 2 \cdot cos(t) \\ Y(t) = 3 \cdot sen(t) \qquad t \in [-2\pi, 2\pi] \\ Z(t) = 3 \cdot t \end{cases}$$

usando parametric. O comando parametric_surface (iii), deve ser utilizado quando a superfície tem dois parâmetros. Um toro de equações paramétricas

$$\begin{cases} X(u,v) = (R + r \cdot cos(v)) \cdot cos(u) \\ Y(u,v) = (R + r \cdot cos(v)) \cdot sen(u) \qquad u, v \in [0, 2\pi] \\ Z(u,v) = r \cdot sen(v) \end{cases}$$

foi criado com o parametric_surface na figura 22. Implicit, no item (iv), plota uma função implícita. No campo <função>, dever ser digitado os dois membros da equação a ser plotada. A figura 23 mostra, um cone de equação $z^2 = 3x^2 + 3y^2$, construído com implicit. A figura 24 mostra os quatro subcomandos juntos e mais opções de cores em um único draw3d. Utilizou-se os comandos xrange, yrange e zrange para fixar um cubo no $\mathbb{R}^3$ onde os gráficos serão visualizados. A omissão destes faz com que o wxMaxima escolha as dimensões do cubo automaticamente. Foram usados um cilindro parabólico $z = (x-5)^2 - 7$ no explicit[3], um segmento parametrizado

$$\begin{cases} X(t) = r \\ Y(t) = 0 \qquad r \in [-10, 10] \\ Z(t) = -r \end{cases}$$

no parametric, uma esfera $(x + 5)^2 + y^2 + z^2 = 25$ no implicit e um plano parametrizado

$$\begin{cases} X(u,v) = 3 - u \\ Y(u,v) = 1 + 2u + v \qquad u, v \in [-30, 30] \\ Z(u,v) = -10 + u \end{cases}$$

no parametric_surface. As imagens 25 e 26 mostram janelas resultantes do comando draw3d em perspectivas diferentes.

A qualquer momento, os gráficos construídos neste capítulo podem ser salvos ou copiados para a área de transferência a fim de que possam ser usados, futuramente, em outros aplicativos como editores de texto. Se o comando possuir o prefixo “wx”, basta clicar

[3] Como no explicit do draw3d é necessário apenas o segundo membro da função, usou-se rhs para extrair somente esse membro. Rhs e lhs retornam respectivamente os membros direito e esquerdo de uma equação.

Cor	Opção $< cor >$
branco	white
preto	black
cinza	gray
vermelho	red
amarelo	yellow
verde	green
azul	blue
rosa escuro	magenta
rosa claro	pink
laranja	orange
marrom	brown
violeta	violet
roxo	purple

Tabela 2: Opções de cores para o draw.

com o botão direito sobre a figura e escolher a opção salvar imagem. O gráfico será salvo em um arquivo de imagem separadamente. Se o comando não possuir o prefixo, clica-se no ícone “salvar na área de transferência”, para então usar-se a opção colar em um editor de texto ou imagem.

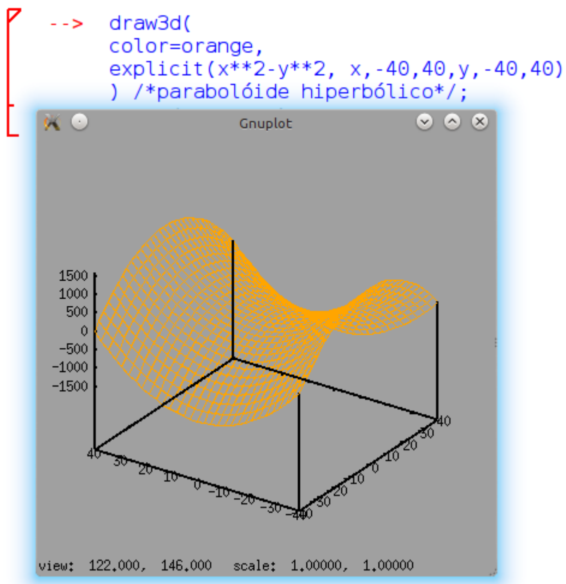


Figura 20: Explicit:Paraboloide hiperbólico

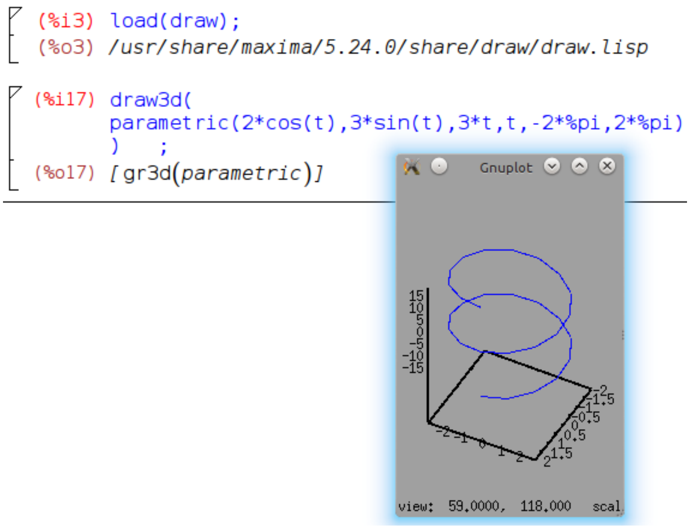


Figura 21: Parametric:Hélice

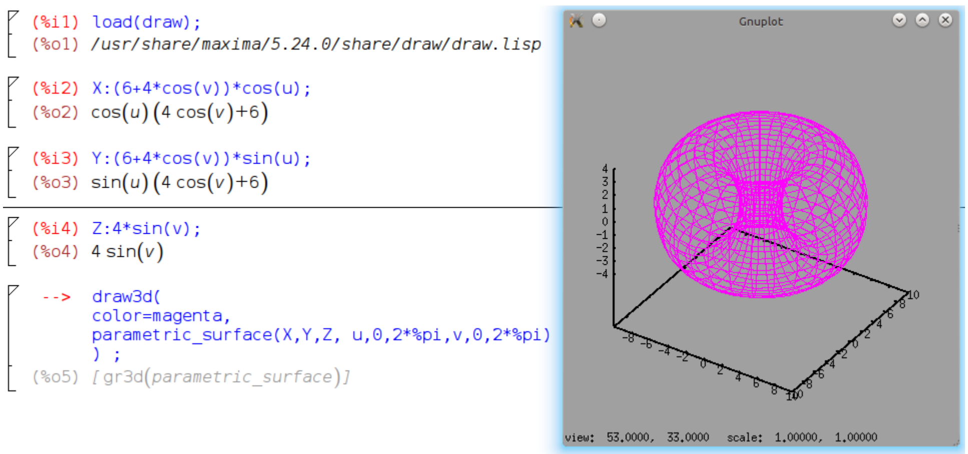


Figura 22: Parametric Surface:Toro

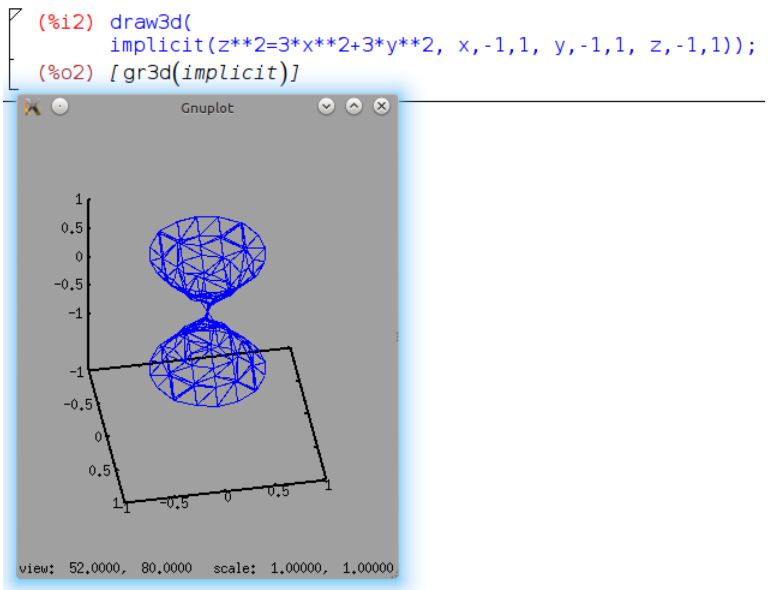


Figura 23: Implicit: Cone

```
(%i1) esfera:(x+5)^2+y^2+z^2=25;
(%o1) z^2+y^2+(x+5)^2=25

(%i2) cil_parabolico:z=(x-5)^2-7;
(%o2) z=(x-5)^2-7

(%i3) load(draw);
(%o3) /usr/share/maxima/5.24.0/share/draw/draw.lisp

(%i4) draw3d(
      xrange=[-10,10],
      yrange=[-10,10],
      zrange=[-10,10],
      color=red,                                          /*cor vermelha*/
      explicit(rhs(cil_parabolico),x,-10,10,y,-10,10), /*cilindro(2ºmembro)*/
      color=orange ,                                      /*cor laranja*/
      parametric(r,0,-r, r,-10,10),                  /*segmento parametrizado*/
      color=black,                                        /*cor preta*/
      implicit(esfera,x,-10,10,y,-10,10,z,-10,10),
      color=magenta,
      parametric_surface(3-u,1+2*u+v,-10+u,u,-30,30,v,-30,30)
      );
(%o4) [gr3d(explicit,parametric,implicit,parametric_surface)]
```

Figura 24: Quatro comandos dentro de um mesmo draw3d

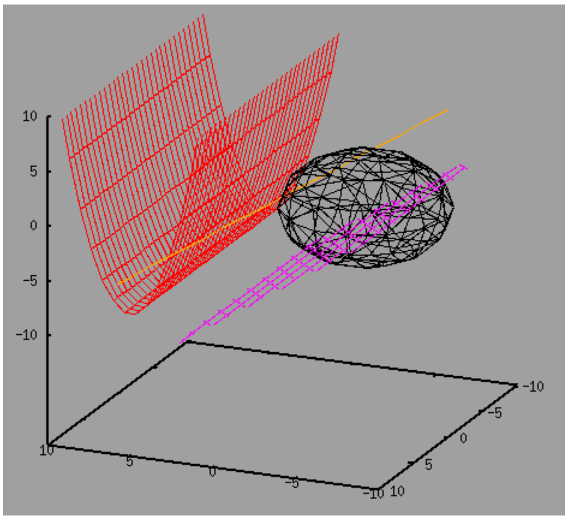


Figura 25: Quatro gráficos em único draw3d.

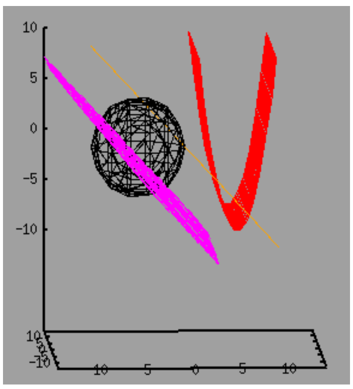


Figura 26: Outra perspectiva para os quatro gráficos

3.3 USANDO GRÁFICOS PARA ENCONTRAR RAÍZES NUMÉRICAS

Dependendo do grau de dificuldade de uma equação, o comando solve, discutido em seções anteriores, não consegue encontrar a raiz de uma equação. Um bom exemplo é $cos(x) = log_e x$ que não pode ser resolvida pelo solve (figura27). Temos que $cos(x) = log_e x \Leftrightarrow cos(x) - log_e x = 0 \Leftrightarrow f(x) = cos(x) - log_e x = 0$. Dessa forma, encontrar a raiz da equação é equivalente a encontrar o zero da função. Plotando-se o gráfico de f(x) (figura 28), que foi obtida pela diferença dos comandos lhs e rhs, encontra-se o intervalo (1,2) no qual a raiz se encontra. Após o intervalo ser determinado, basta utilizar o comando find_ root com a seguinte sintaxe:

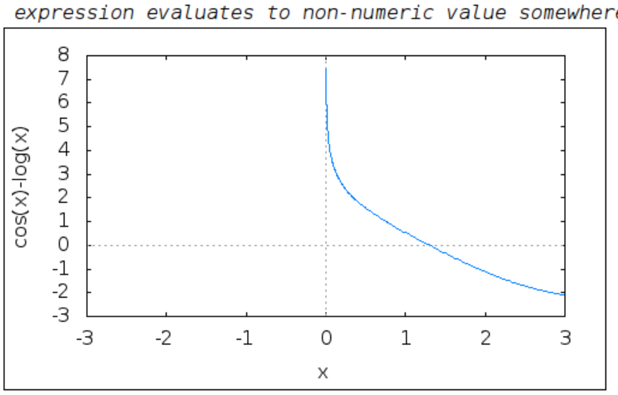
find_ root(f(x)=0,x,<xmin>, <xmax>).

O find_ root encontrará a raiz da equação $cos(x) - log_e x = 0$ no intervalo (1 , 2) encontrando $x = 1,302964001216013$.

```
(%i1) eq:cos(x)=log(x)     /*Equação de nome eq.*/;
(%o1) cos(x)=log(x)

(%i2) solve(eq)          /*Nenhuma resposta significativa.*/;
(%o2) [log(x)=cos(x)]

(%i3) f(x):=lhs(eq)-rhs(eq)    /*Diferença entre os membros de eq1.*/;
(%o3) f(x):=lhs(eq)-rhs(eq)

(%i4) f(x)        /*Testando a função*/;
(%o4) cos(x)-log(x)
```

Figura 27: Ineficácia do comando solve e função f(x).

```
(%i5) wxplot2d([f(x)], [x,-3,3])    /*A raiz está entre 1 e 2.*/;
plot2d: expression evaluates to non-numeric value somewhere in plotting range.
(%t5)
```




```
(%o5)

(%i6) find_root(eq,x,1,2);
(%o6) 1.302964001216013
```

Figura 28: Gráfico de f(x) e utilização do find_ root.

4 CÁLCULO DIFERENCIAL E ÁLGEBRA LINEAR

4.1 LIMITES DE FUNÇÕES

Limites são usados quando se quer saber o comportamento de uma função f(x) quando x está na vizinhança de um valor x_o,

$$\lim_{x \to x_o} f(x)$$

quando x tende ao infinito positivo

$$\lim_{x \to \infty} f(x)$$

ou infinito negativo.

$$\lim_{x \to -\infty} f(x)$$

No caso dos limites tendendo a um valor x_o, ainda podem ser calculados os limites laterais quando x se aproxima de x_o pela direita

$$\lim_{x \to x_o^+} f(x)$$

ou pela esquerda.

$$\lim_{x \to x_o^-} f(x)$$

O limite de $f(x)$ com x tendendo a x_o só existe quando os limites laterais existem e são iguais. O sofware wxMaxima faz o cálculo de limites através do comando *limit*. Para os limites tendendo a um número é real é usado

limit(<função>,<variável>,<número>)

e para limites laterais tendendo a esquerda e a direita

limit(<função>,<variável>,<número>,minus)

e

limit(<função>,<variável>,<número>,plus),

respectivamente. No campo <número> deve ser digitado o número para o qual a variável x tende. Além dos números reais retornados por *limit*, podem aparecer as palavras *und*, *ind* ou *infinity*, significando que o limite é indeterminado. Considere o exemplo contido na figura 29 para saber o valor do limite da função $f(x) = \frac{1}{3x-6}$ quando x tende a 2 pela esquerda, direita e bilateralmente. Definiu-se a função f, como feito em seção anterior, calculando posteriormente os limites.

```
(%i16) f(x):=1/(3*x-6);
(%o16) f(x):= 1/(3x-6)

(%i17) limit(f(x),x,2,plus);
(%o17) ∞

(%i18) limit(f(x),x,2,minus);
(%o18) -∞

(%i19) limit(f(x),x,2)  /*indefinido nos reais*/;
(%o19) infinity
```

Figura 29: Limites laterais e bilateral.

Para os limites tendendo a mais ou a menos infinito usam-se, respectivamente, as sintaxes

limit(<função>,<variável>,inf)

e

limit(<função>,<variável>,minf)

significando *inf*, mais infinito e *minf*, menos infinito. Na figura 30,foi tomada uma função g definida como o quociente entre dois polinômios de 3º grau e calculado seus limites tendendo a mais e menos infinito. Três limites fundamentais do cálculo diferencial estão presentes na figura 31. São eles:

(i) $\lim_{x\to 0} \frac{sen(x)}{x} = 1$

(ii) $\lim_{x\to\infty} \left(1 + \frac{1}{x}\right)^x = e$

(iii) $\lim_{x\to 0} \frac{a^x - 1}{x} = ln(a)$

O resultado do limite (ii), %e , é a constante de Euler e. É um número irracional aproximadamente igual a 2,71828182 . É também importante lembrar que qualquer logaritmo que apareça no programa é neperiano, ou seja, na base e, conforme dito na primeira seção do capítulo 02.

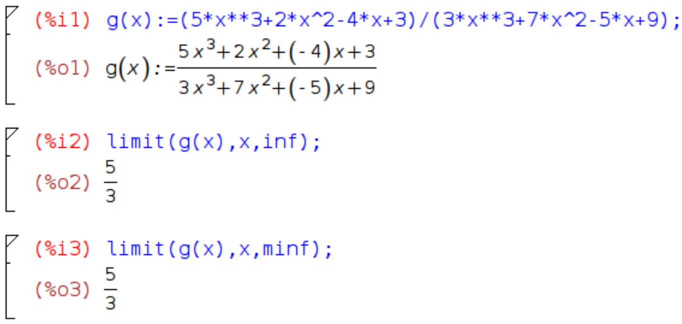


Figura 30: Limites tendendo ao infinito.

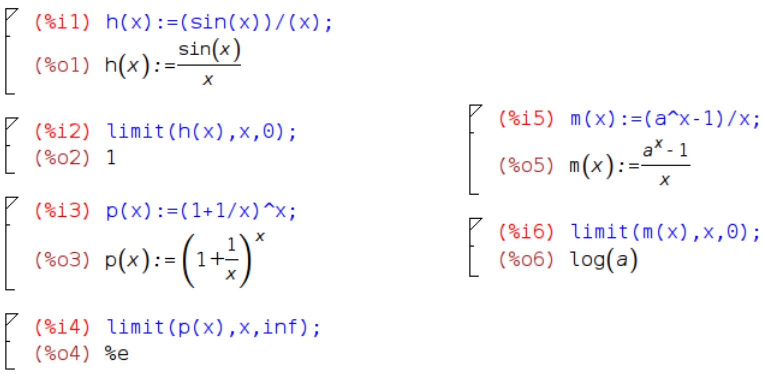


Figura 31: Alguns limites fundamentais.

4.2 DERIVADAS

A derivada f'(x) de uma função f(x) pode ser interpretada como taxa de variação instantânea da função, ou como coeficiente angular da reta tangente ao gráfico de f no ponto (x,f(x)). É definida através do limite

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}.$$

Se esse limite não existir, diz-se que a função não é derivável. O símbolo $\frac{dy}{dx}$ também é outra maneira de se simbolizar a derivada da função f(x)=y em relação a x. Caso f' for derivável, sua derivada f''(x) é chamada derivada de 2ª ordem, se f''(x) existir. Se $f^{(n-1)}(x)$ for derivável, sua derivada $f^{(n)}(x)$ é dita de ordem "n" (se $f^{(n)}(x)$ existir) .

Usando o wxMaxima, a derivada de uma função é calculada através do comando *diff* da seguinte forma:

diff(<função>,<variável>, <ordem>)

Devem ser declarados no comando *diff*, a função, a variável independente e a ordem da derivada desejada. Derivadas de primeira, segunda, terceira e décima ordens foram obtidas para a função composta $f(x) = cos(2t - 5)$ na figura 32.

Para funções de duas ou mais variáveis, podem ser calculadas derivadas parciais que são definidas pelo limite

$$\frac{\partial f}{\partial x_i} = \lim_{\Delta x_i \to 0} \frac{f(x_1, x_2, \cdots, x_i + \Delta x_i, \cdots, x_n) - f(x_1, x_2, \cdots, x_i, \cdots, x_n)}{\Delta x_i}.$$

Resumindo, deriva-se em relação a variável x_i considerando as outras variáveis constantes. Na figura 33, foi definida uma função de duas variáveis $g(w, x) = log_e(x^2 w^5 - x w^2)$. Após isso, foram calculadas derivadas parciais de primeira ordem em relação as variáveis w e x.

```
(%i1) f(t):=cos(2*t-5);
(%o1) f(t):=cos(2 t - 5)

(%i2) diff(f(t),t,1)  /*primeira ordem*/;
(%o2) - 2 sin(2 t - 5)

(%i3) diff(f(t),t,2)  /*segunda ordem*/;
(%o3) - 4 cos(2 t - 5)

(%i4) diff(f(t),t,3)  /*terceira ordem*/;
(%o4) 8 sin(2 t - 5)

(%i5) diff(f(t),t,10) /*décima ordem*/;
(%o5) - 1024 cos(2 t - 5)
```

Figura 32: Derivada de uma função composta.

```
(%i1) g(w,x):=log(x^2*w^5-x*w^2) /*log na base e*/;
(%o1) g(w,x):=log(x^2 w^5 - x w^2)

(%i2) diff(g(w,x),w,1)  /*relação a w*/;
(%o2) (5 w^4 x^2 - 2 w x) / (w^5 x^2 - w^2 x)

(%i3) diff(g(w,x),x,1)  /*relação a x*/;
(%o3) (2 w^5 x - w^2) / (w^5 x^2 - w^2 x)
```

Figura 33: Exemplo de derivada parcial.

4.3 INTEGRAIS

Integrais podem ser calculadas através dos comandos *integrate* e *romberg* com as seguintes sintaxes:

(i) integrate(<função>,<variável>)

(ii) integrate(<função>,<variável>, <lim-inf> , <lim-sup>)

(iii) romberg(<função>,<variável>, <lim-inf> , <lim-sup>)

Para o cálculo de integrais indefinidas usa-se (i) e para as definidas (ii). Os campos <lim-inf> e <lim-sup> são respectivamente os limites de integração inferior e superior. Quando não é possível obter a integral definida por integrate, utiliza-se *romberg*, em (iii), com mesma sintaxe de *integrate* em (ii). Na figura 34, foram calculadas a integral definida e indefinida por *integrate*. A figura 35 mostra uma situação quando o *integrate* não consegue determinar o valor da integral definida

$$\int_0^2 \sqrt{m^4+1}\, dm.$$

Na figura 36, *romberg* é utilizado para resolver a mesma integral, numericamente.[1]

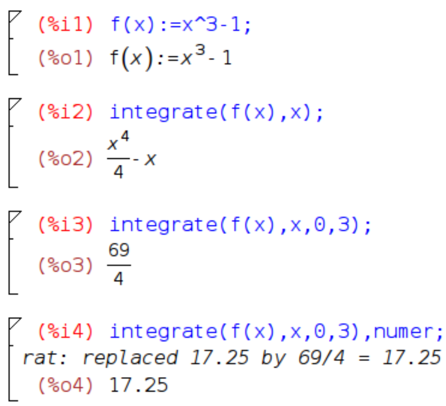


Figura 34: Comando *integrate*.

[1] O comando *romberg* deve ser carregado através de *load(romberg)*

```
(%i1) p(m):=sqrt(1+m^4);
(%o1) p(m):=sqrt(1+m^4)

(%i2) integrate(p(m),m,0,2);
(%o2) integrate(sqrt(m^4+1), m, 0, 2)

(%i3) integrate(p(m),m,0,2),numer;
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced 0.5 by 1/2 = 0.5
(%o3) integrate((m^4+1)^0.5, m, 0, 2)
```

Figura 35: Incapacidade de *integrate* para integrar algumas funções.

```
(%i4) load(romberg);
(%o4) /usr/share/maxima/5.24.0/share/numeric/romberg.lisp

(%i5) romberg(p(m),m,0,2);
(%o5) 3.653485706412626
```

Figura 36: O uso de *romberg* para integrais definidas.

4.4 SISTEMAS LINEARES

Um sistema linear pode ser resolvido através do comando

$$solve([< lista_e >], [< lista_v >])$$

em que $[< lista_e >]$ é uma lista das equações a serem resolvidas e $[< lista_v >]$ um lista das variáveis envolvidas. Determinou-se, através de *solve*, a solução do sistema

$$\begin{cases} 2x - 3y + 4z + 6t + 2w = -\frac{59}{10} \\ 4x - y + 5z + 3t + 3w = -\frac{41}{10} \\ 7x + 9y + 3z - 6t + 11w = \frac{134}{5} \\ 2x - 2y - 3z + 5t + 8w = \frac{877}{30} \\ x + 3y - z + t + 7w = \frac{653}{30} \end{cases}$$

de cinco equações e cinco variáveis na figura 37. Quando o sistema é possível e indeterminado como no sistema,

$$\begin{cases} 2x - 3y + 4z - t = 9 \\ 4x - y + 5z + 3t = 10 \\ 2x + 10y + 3z - 6t = 15 \end{cases}$$

solve irá usar como parâmetro números reais como %r1, %r2, ... , %rn. (figura 38). Quando o sistema é impossível conforme

$$\begin{cases} 2x - 3y + 4z = 7 \\ 4x - y + 5z = 15 \\ -6x + 9y - 12z = 17 \end{cases}$$

o *solve* retornará colchetes vazios (figura 39).

```
(%i1) eq1:2*x-3*y+4*z+6*t+2*w=-59/10;
```
(%o1) $4\,z-3\,y+2\,x+2\,w+6\,t=-\frac{59}{10}$

```
(%i2) eq2:4*x - y + 5*z + 3*t + 3*w =-41/10;
```
(%o2) $5\,z-y+4\,x+3\,w+3\,t=-\frac{41}{10}$

```
(%i3) eq3:7*x + 9*y + 3*z -6*t + 11*w =134/5;
```
(%o3) $3\,z+9\,y+7\,x+11\,w-6\,t=\frac{134}{5}$

```
(%i4) eq4:2*x - 2*y - 3*z + 5*t + 8*w = 877/30;
```
(%o4) $-3\,z-2\,y+2\,x+8\,w+5\,t=\frac{877}{30}$

```
(%i5) eq5:x + 3*y - z + t + 7*w =653/30;
```
(%o5) $-z+3\,y+x+7\,w+t=\frac{653}{30}$

```
(%i6) solve([eq1,eq2,eq3,eq4,eq5],[x,y,z,t,w]);
```
(%o6) $[[x=1, y=\frac{1}{2}, z=-3, t=\frac{1}{6}, w=\frac{23}{10}]]$

Figura 37: Solucionando sistemas lineares com *solve*.

```
(%i1) eq6:2*x-3*y+4*z - t=9;
```
(%o1) $4\,z-3\,y+2\,x-t=9$

```
(%i2) eq7:4*x - y + 5*z + 3*t =10;
```
(%o2) $5\,z-y+4\,x+3\,t=10$

```
(%i3) eq8:2*x + 10*y + 3*z -6*t  =15;
```
(%o3) $3\,z+10\,y+2\,x-6\,t=15$

```
(%i4) solve([eq6,eq7,eq8],[x,y,z,t]);
```
(%o4) $[[x=-\frac{133\,\%r1+76}{34}, y=\frac{10\,\%r1+13}{17}, z=\frac{45\,\%r1+67}{17}, t=\%r1]]$

Figura 38: Sistema possível e indeterminado.

```
(%i1) eq9:2*x-3*y+4*z=7;
(%o1) 4 z - 3 y+2 x=7

(%i2) eq10:4*x  -  y  +  5*z   =15;
(%o2) 5 z - y+4 x=15

(%i3) eq11:-6*x  +  9*y  -12*z  =17;
(%o3) - 12 z+9 y - 6 x=17

(%i4) solve([eq9, eq10, eq11], [x,y,z]);
(%o4) []
```

Figura 39: Sistema impossível.

4.5 MATRIZES

Uma matriz é declarada no wxMaxima através do comando

$$matrix([linha_1], [linha_2], \quad \cdots \quad , [linha_n])$$

sendo cada $[linha_i]$ uma lista dos elementos da linha. A sessão 2.2 contém detalhes sobre como trabalhar com listas. Quando uma matriz é criada, podem ser feitas chamadas de alguns elementos da matriz, como:

(i) $< matriz > [i, j]$ para um elemento da linha i e coluna j.

(ii) $row(< matriz >, i)$ para chamar toda a linha i.

(iii) $col(< matriz >, j)$ para chamar toda a coluna j

Outros comandos podem ser utilizados sobre ela. *Determinant* calcula o determinante da matriz, *invert* a inverte e *echelon* dá sua forma escalonada. (figuras 40, 41, 42 e 43). A sintaxes são, respectivamente,

(iv) $determinant(< matriz >)$

(v) $invert(< matriz >)$

(vi) $echelon(< matriz >)$

Para operações entre matrizes tem-se, o produto de uma matriz por um escalar, a soma e a subtração e produto entre matrizes nas figuras 44, 45 e 46. A parte referente a autovalores e autovetores serão tratados na próxima sessão.

```
(%i1) A:matrix([2,3,1/4],[1,7,-6],[-3,-1/3,2]);
```

$$(\%o1)\quad \begin{bmatrix} 2 & 3 & \frac{1}{4} \\ 1 & 7 & -6 \\ -3 & -\frac{1}{3} & 2 \end{bmatrix}$$

```
(%i2) A    /* Chamando a matriz A. */;
```

$$(\%o2)\quad \begin{bmatrix} 2 & 3 & \frac{1}{4} \\ 1 & 7 & -6 \\ -3 & -\frac{1}{3} & 2 \end{bmatrix}$$

Figura 40: Definindo uma matriz

```
(%i2) A[2,1] ;
(%o2) 1
```

```
(%i3) A[3,2];
```

$$(\%o3)\quad -\frac{1}{3}$$

```
(%i4) row(A,2)/* segunda linha */;
```

$$(\%o4)\quad \begin{bmatrix} 1 & 7 & -6 \end{bmatrix}$$

```
(%i5) col(A,3) /* terceira coluna */;
```

$$(\%o5)\quad \begin{bmatrix} \frac{1}{4} \\ -6 \\ 2 \end{bmatrix}$$

Figura 41: Alguns elementos da matriz A.

```
(%i3) determinant(A);
```

$$(\%o3)\quad \frac{463}{6}$$

```
(%i4) echelon(A);
```

$$(\%o4)\quad \begin{bmatrix} 1 & \frac{3}{2} & \frac{1}{8} \\ 0 & 1 & -\frac{49}{44} \\ 0 & 0 & 1 \end{bmatrix}$$

Figura 42: Determinante e escalonamento.

```
(%i5) invert(A);
```

$$(\%o5)\quad \begin{bmatrix} \frac{72}{463} & -\frac{73}{926} & -\frac{237}{926} \\ \frac{96}{463} & \frac{57}{926} & \frac{147}{926} \\ \frac{124}{463} & -\frac{50}{463} & \frac{66}{463} \end{bmatrix}$$

Figura 43: Matriz inversa.

(%i2) `B:matrix( [6,-5,16], [0,1,-6], [4,-7,2]);`

$$(\%o2)\quad \begin{bmatrix} 6 & -5 & 16 \\ 0 & 1 & -6 \\ 4 & -7 & 2 \end{bmatrix}$$

(%i8) `(1/3)*B;`

$$(\%o8)\quad \begin{bmatrix} 2 & -\frac{5}{3} & \frac{16}{3} \\ 0 & \frac{1}{3} & -2 \\ \frac{4}{3} & -\frac{7}{3} & \frac{2}{3} \end{bmatrix}$$

Figura 44: Matriz B e produto por escalar.

(%i3) `A.B;`

$$(\%o3)\quad \begin{bmatrix} 13 & -\frac{35}{4} & \frac{29}{2} \\ -18 & 44 & -38 \\ -10 & \frac{2}{3} & -42 \end{bmatrix}$$

(%i4) `B.A;`

$$(\%o4)\quad \begin{bmatrix} -41 & -\frac{67}{3} & \frac{127}{2} \\ 19 & 9 & -18 \\ -5 & -\frac{113}{3} & 47 \end{bmatrix}$$

Figura 45: Produto entre matrizes.

(%i5) `A+B;`

$$(\%o5)\quad \begin{bmatrix} 8 & -2 & \frac{65}{4} \\ 1 & 8 & -12 \\ 1 & -\frac{22}{3} & 4 \end{bmatrix}$$

(%i7) `B-A;`

$$(\%o7)\quad \begin{bmatrix} 4 & -8 & \frac{63}{4} \\ -1 & -6 & 0 \\ 7 & -\frac{20}{3} & 0 \end{bmatrix}$$

Figura 46: Adição e diferença.

4.6 POLINÔMIO CARACTERÍSTICO, AUTOVALORES E AUTOVETORES

O polinômio característico $p(\lambda)$ de uma matriz é aquele usado para obter os autovalores desta mesma matriz. É obtido através da equação

$$\begin{vmatrix} a_{1,1}-\lambda & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2}-\lambda & a_{2,3} & \cdots & a_{2,n} \\ a_{3,1} & a_{3,2} & a_{3,3}-\lambda & \cdots & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n}-\lambda \end{vmatrix} = 0\ .$$

Onde $a_{i,j}$ são os elementos de uma matriz quadrada A. No Maxima, o polinômio característico pode ser obtido através de *charpoly*($< matriz >$,$< var >$). O comando depende somente de uma matriz e da variável independente do polinômio. A figura 47 contém exemplo do comando *charpoly* na variável “w” e o uso de *solve* para determinar os autovalores.

Os autovalores de uma matriz também podem ser calculados através de um pacote especializado chamado *eigen* que deve ser carregado através do comando *load(eigen)*. Para o obtenção dos autovalores usa-se

$$eigenvalues(< matriz >)$$

Na figura 48, deve-se observar no resultado de *eigenvalues*, que a primeira lista contém os autovalores e a segunda a multiplicidade de cada um como raiz do polinômio característico.

Quando se deseja também os autovetores da matriz, faz-se uso de outra ferramenta do pacote *eigen*, o *eigenvectors*.

$$eigenvectors(< matriz >)$$

Eigenvectors calcula, além dos autovetores, todos os autovalores e suas respectivas multiplicidades. Retorna uma lista contendo os autovalores, uma contendo as multiplicidades e outra contendo os autovetores nesta ordem. A figura 49 contém exemplo da ferramenta *eigenvectors* e a tabela 3 a interpretação dos resultados.

```
(%i1) C:matrix([3,-1,1],[-1,5,-1],[1,-1,3]);
```

$$
(\%o1)\quad \begin{bmatrix} 3 & -1 & 1 \\ -1 & 5 & -1 \\ 1 & -1 & 3 \end{bmatrix}
$$

```
(%i2) charpoly(C,w);
```

$$
(\%o2)\quad 2\,w+\left((3-w)\,(5-w)-1\right)(3-w)-6
$$

```
(%i3) expand(%)   /*desenvolve o resultado anterior*/;
```

$$
(\%o3)\quad -w^3+11\,w^2-36\,w+36
$$

```
(%i4) solve(%,w) /*autovalores de C*/;
(%o4) [w=2,w=3,w=6]
```

Figura 47: Autovalores da matriz C.

```
(%i7) load(eigen);
(%o7) /usr/share/maxima/5.24.0/share/matrix/eigen.mac

(%i9) eigenvalues(C);
(%o9) [[2,3,6],[1,1,1]]
```

Figura 48: Autovalores usando o pacote *eigen*.

```
(%i4) eigenvectors(C);
(%o4) [[[2,3,6],[1,1,1]],[[[1,0,-1]],[[1,1,1]],[[1,-2,1]]]]
```

Figura 49: Autovetores da matriz C.

Autovalor	Multiplicidade	Autovetor
2	1	(1,0,-1)
3	1	(1,1,1)
6	1	(1,-2,1)

Tabela 3: Interpretação do resultado da figura 49.

5 EQUAÇÕES DIFERENCIAIS ORDINÁRIAS

5.1 REVISANDO A TEORIA

Segundo Zill e Cullen [8], uma equação contendo derivadas de uma ou mais variáveis dependentes, em relação a uma ou mais variáveis independentes é dita equação diferencial. Se a equação diferencial possui apenas derivadas de uma ou mais variáveis dependentes, em relação a uma só variável independente é classificada como equação diferencial ordinária.

O objetivo de se resolver uma equação diferencial é determinar as funções incógnitas (variáveis dependentes). A ordem de uma EDO é a maior ordem entre as derivadas que figuram na equação. Por exemplo, sendo $y = f(x)$, considere a EDO

$$a_n y^{(n)} + a_{n-1} y^{(n-1)} + \cdots + a_1 y' + a_0 y = b$$

onde $a_n \neq 0$. Diz-se que n é a ordem da equação diferencial e cada a_i e b constantes ou funções de x.

5.1.1 Equações de de 1ª ordem

5.1.1.1 Equações de variáveis separáveis

É toda equação diferencial da forma

$$\frac{dy}{dx} = p(x)q(y). \tag{5.1}$$

O interessante desse tipo de equações é que suas variáveis podem ser separadas em cada membro da equação, o que torna simples sua resolução. Assim 5.1 pode ser escrita como

$$\frac{1}{q(y)}dy = p(x)dx$$

que quando tem os membros integrados chega-se a

$$\int \frac{1}{q(y)}dy = \int p(x)dx + C$$

que muitas vezes pode ser resolvida utilizando-se os métodos de integração aprendidos no cálculo básico.

5.1.1.2 Equações exatas

É qualquer equação da forma

$$M(x,y) + N(x,y)\frac{dy}{dx} = 0 \tag{5.2}$$

com

$$M_y(x,y) = N_x(x,y) \tag{5.3}$$

sendo $M(x,y)$ e $N(x,y)$ funções de x e y. As funções $M_y(x,y)$ e $N_x(x,y)$ são derivadas parciais e y é função de x. Se a condição 5.3 for satisfeita, existe um função $\psi(x,y)$ tal que:

$$\frac{\partial\psi}{\partial x} = M(x,y)$$
$$\frac{\partial\psi}{\partial y} = N(x,y)$$

Assim 5.2 pode ser escrita como

$$\frac{\partial\psi}{\partial x} + \frac{\partial\psi}{\partial y}\frac{dy}{dx} = \frac{d\psi}{dx} \Rightarrow \frac{d\psi}{dx} = 0 \Rightarrow d\psi = 0dx$$

que, através de integração nos dois membros, possui solução implícita $\psi(x,y) = c$.

5.1.1.3 Equações lineares e os fatores integrantes

A teoria do fator integrante $\mu(t)$ é usada para se resolver uma EDO linear de primeira ordem, na sua forma geral, ou seja

$$\frac{dy}{dt} + m(t)y = n(t) \tag{5.4}$$

onde $m(t)$ e $n(t)$ são funções da variável independente t. O fator integrante para a equação 5.4 deve ser

$$\mu(t) = e^{\int m(t)dt} \tag{5.5}$$

pois se multiplicado em ambos os membros da equação 5.4, tem-se

$$\frac{dy}{dt} + m(t)y = n(t)$$

$$\Rightarrow \mu(t)\frac{dy}{dt} + \mu(t)m(t)y = n(t)\mu(t)$$

$$\Rightarrow (e^{\int m(t)dt})\frac{dy}{dt} + (e^{\int m(t)dt})m(t)y = n(t)(e^{\int m(t)dt})$$

onde o primeiro membro da última equação é uma derivada do produto, então

$$\frac{d[y(e^{\int m(t)dt})]}{dt} = n(t)(e^{\int m(t)dt})$$

$$\Rightarrow d[y(e^{\int m(t)dt})] = [n(t)(e^{\int m(t)dt})]dt$$

integrando-se os dois membros

$$\int d[y(e^{\int m(t)dt})] = \int [n(t)(e^{\int m(t)dt})]dt$$

$$\Rightarrow y(e^{\int m(t)dt}) - C = \int [n(t)(e^{\int m(t)dt})]dt$$

$$\Rightarrow y = \frac{\int [n(t)(e^{\int m(t)dt})]dt + C}{(e^{\int m(t)dt})}$$

$$\Rightarrow y = \frac{1}{\mu(t)}\left[\int n(t)\mu(t)dt + C\right] \tag{5.6}$$

Assim a equação 5.4 pode ser resolvida usando o fator integrante 5.5 e possui solução geral 5.6.

5.1.2 EDOs de 2ª ordem com coeficientes constantes

5.1.2.1 Equação característica e resolução da equação homogênea

Uma equação geral de segunda ordem com coeficientes constantes a,b e c tem a forma

$$ay'' + by' + cy = f(x) \tag{5.7}$$

e é resolvida determinando-se, primeiro, a solução da equação homogênea

$$ay'' + by' + cy = 0 \tag{5.8}$$

e somando-se a ela uma solução particular. [1] Observando 5.8 nota-se que, quando c é não nulo, a função incógnita y é uma combinação linear de suas derivadas primeira e segunda. Caso c seja nulo e b não, a derivada primeira é múltipla da derivada segunda. Uma das funções que satisfaz essas propriedades, é a função exponencial

$$y = e^{rx}, r \in \mathbb{R} \tag{5.9}$$

[1] O caso da solução particular não será tratado neste texto, mas pode ser encontrado em qualquer bom livro de equações diferenciais. Vide bibliografia [2] no final deste trabalho.

pois $y' = re^{rx}$ e $y'' = r^2 e^{rx}$. Substituindo 5.9 e suas derivadas em 5.8 tem-se

$$ay'' + by' + cy = 0 \Rightarrow ar^2 e^{rx} + bre^{rx} + ce^{rx} = 0 \Rightarrow (ar^2 + br + c)e^{rx} = 0$$

como $e^{rx} \neq 0$,

$$ar^2 + br + c = 0 \tag{5.10}$$

O expressão 5.10 é chamada equação característica da equação diferencial 5.8. Por ser uma equação do segundo grau, temos 3 casos a considerar nas próximas seções.

5.1.2.2 Duas raízes reais distintas

Se a equação característica 5.10 tem duas raízes reais distintas r_1 e r_2, o que acontece quando $b^2 - 4ac > 0$, a solução geral da equação 5.8 será

$$y = C_1 e^{r_1 x} + C_2 e^{r_2 x} \tag{5.11}$$

pois substituindo 5.11 em 5.8

$$ay'' + by' + cy = a(C_1 r_1^2 e^{r_1 x} + C_2 r_2^2 e^{r_2 x}) + b(C_1 r_1 e^{r_1 x} + C_2 r_2 e^{r_2 x}) + c(C_1 e^{r_1 x} + C_2 e^{r_2 x}) =$$

$$= C_1 e^{r_1 x}(ar_1^2 + br_1 + c) + C_2 e^{r_2 x}(ar_2^2 + br_2 + c) = C_1 e^{r_1 x}.0 + C_2 e^{r_2 x}.0 = 0$$

pois r_1 e r_2 são raízes da equação característica.

5.1.2.3 Duas raízes complexas

Se $b^2 - 4ac < 0$, a equação característica 5.10 possuirá duas raízes complexas

$$r_1 = p \pm qi, \quad p, q \in \mathbb{R}\ .$$

Tem-se então, como solução, a função 5.11 com expoentes complexos. Neste ponto, faz-se necessária a fórmula de Euler

$$e^{it} = cos(t) + isen(t)$$

para potências de números complexos. Assim, a solução com raízes complexas fica

$$y = C_1' e^{r_1 x} + C_2' e^{r_2 x} = C_1' e^{(p+qi)x} + C_2' e^{(p-qi)x} = C_1' e^{px} e^{qix} + C_2' e^{px} e^{-qix} =$$

$$= C_1' e^{px}(cos(qx) + isen(qx)) + C_2' e^{px}(cos(qx) - isen(qx)) =$$

$$= (C_1' e^{px} + C_2' e^{px})cos(qx) + i(C_1' e^{px} - C_2' e^{px})sen(qx) =$$

$$= (C_1' + C_2')e^{px}cos(qx) + i(C_1' - C_2')e^{px}sen(qx) = C_1e^{px}cos(qx) + C_2e^{px}sen(qx)$$

Logo a solução é

$$y = C_1e^{px}cos(qx) + C_2e^{px}sen(qx)\ . \tag{5.12}$$

5.1.2.4 Uma raiz real

Quando o discriminante $b^2 - 4ac$ é nulo, não é possível utilizar nenhuma das soluções anteriores, pois tem-se uma única raiz $r = -\frac{b}{2a}$ para a equação característica 5.10. As soluções 5.11 e 5.12 não são apropriadas pois $r_1 = r_2$ e não existem soluções complexas, respectivamente. Neste caso, procura-se uma segunda solução, y_2 que seja um produto da conhecida exponencial $e^{-\frac{bx}{2a}}$ e uma função desconhecida $d(x)$.

$$y_2 = d(x)e^{-\frac{bx}{2a}} \tag{5.13}$$

Calculando as derivadas primeira e segunda de 5.13

$$y_2' = d'(x)e^{-\frac{bx}{2a}} - \frac{b}{2a}d(x)e^{-\frac{bx}{2a}}$$

$$y_2'' = d''(x)e^{-\frac{bx}{2a}} - \frac{b}{2a}d'(x)e^{-\frac{bx}{2a}} - \frac{b}{2a}d'(x)e^{-\frac{bx}{2a}} + \frac{b^2}{4a^2}d(x)e^{-\frac{bx}{2a}}$$

e substituindo-as na equação 5.8, tem-se

$$a\left(d''(x)e^{-\frac{bx}{2a}} - \frac{b}{2a}d'(x)e^{-\frac{bx}{2a}} - \frac{b}{2a}d'(x)e^{-\frac{bx}{2a}} + \frac{b^2}{4a^2}d(x)e^{-\frac{bx}{2a}}\right) +$$

$$+b\left(d'(x)e^{-\frac{bx}{2a}} - \frac{b}{2a}d(x)e^{-\frac{bx}{2a}}\right) + c\left(d(x)e^{-\frac{bx}{2a}}\right) = 0 \Rightarrow$$

$$e^{-\frac{bx}{2a}}\left\{a\left(d''(x) - \frac{b}{2a}d'(x) - \frac{b}{2a}d'(x) + \frac{b^2}{4a^2}d(x)\right) + b\left(d'(x) - \frac{b}{2a}d(x)\right) + cd(x)\right\} = 0 \Rightarrow$$

$$e^{-\frac{bx}{2a}}\left\{ad''(x) + \left(-\frac{b}{2} - \frac{b}{2} + b\right)d'(x) + \left(\frac{b^2}{4a} - \frac{b^2}{2a} + c\right)d(x)\right\} = 0 \Rightarrow$$

$$e^{-\frac{bx}{2a}}\left\{ad''(x) + 0.d'(x) + \left(\frac{b^2 - 2b^2 + 4ac}{4a}\right)d(x)\right\} = 0 \Rightarrow$$

$$e^{-\frac{bx}{2a}}\left\{ad''(x) + 0.d'(x) - \left(\frac{b^2 - 4ac}{4a}\right)d(x)\right\} = 0 \Rightarrow$$

$$e^{-\frac{bx}{2a}}\left\{ad''(x) + 0.d'(x) + \frac{0}{4a}.d(x)\right\} = 0 \Rightarrow d''(x) = 0$$

Concluindo-se então que a derivada segunda de d(x) é nula, a derivada primeira uma constante e a função d(x) uma função do primeiro grau da forma $d(x) = C_1' + C_2x$. Logo a solução geral será

$$y = C_1''y_1 + C_2y_2 = C_1''y_1 + C_2d(x)y_1 = C_1''e^{-\frac{bx}{2a}} + C_2(C_1' + C_2x)e^{-\frac{bx}{2a}} =$$

$$= (C_1'' + C_2C_1')e^{-\frac{bx}{2a}} + C_2xe^{-\frac{bx}{2a}} = C_1e^{-\frac{bx}{2a}} + C_2xe^{-\frac{bx}{2a}} \Rightarrow$$

$$y = C_1e^{-\frac{bx}{2a}} + C_2xe^{-\frac{bx}{2a}} \tag{5.14}$$

5.2 USANDO O WXMAXIMA PARA SOLUCIONAR EDOS

O wxMaxima usa o comando *"ode2"* para resolver equações de primeira ou segunda ordens da seguinte forma:

ode2(<equação>,<vardep>,<varindep>)

Sendo <equação>, a equação diferencial, <vardep>, a variável dependente (função) e <varindep>, a variável independente. Quando se deseja resolver problemas envolvendo condições iniciais, utiliza-se

ic1(<solução>,$< x_1 >$,$< y_1 >$)

para soluções de equações de primeira ordem, onde é necessário um ponto, ($< x_1 >$,$< y_1 >$), e

bc2(<solução>,$< x_1 >$,$< y_1 >$,$< x_2 >$,$< y_2 >$)

para soluções de equações de segunda ordem, onde precisa-se de dois pontos, ($< x_1 >$,$< y_1 >$,$< x_2 >$,$< y_2 >$). Como exemplo, tem-se as equações diferenciais

$$y'(x) + 10y = 0,$$

$$2x + y^2 + 2xyy' = 0$$

e

$$y''(x) - 5y'(x) + 6y = 0$$

resolvidas com suas respectivas condições de iniciais nas figuras 50, 51 e 52.

O apóstrofo, visto antes do comando *diff*, faz com que o programa não tente calcular a derivada de y, conforme visto na seção 4.2. Com seu uso, wxMaxima não calculas as derivadas mas sim as entende como derivações da função incógnita, y' e y''.

```
(%i1) edo1:'diff(y,x,1)+10*y=0;
(%o1) d/dx y+10 y=0

(%i2) ode2(edo1,y,x);
(%o2) y=%c %e^(-10 x)

(%i3) ic1(%,x=%pi,y=3);
(%o3) y=3 %e^(10 %pi-10 x)
```

Figura 50: Equação diferencial de 1ª ordem com condição inicial $(\pi, 3)$.

```
(%i1) edo2:2*x+y^2+2*x*y*'diff(y,x,1)=0;
(%o1) 2 x y (d/dx y)+y^2+2 x=0

(%i2) ode2(edo2,y,x);
(%o2) x y^2+x^2=%c

(%i3) ic1(%,x=-2,y=sqrt(5));
(%o3) x y^2+x^2=-6
```

Figura 51: Equação diferencial de 1ª ordem com condição inicial $(-2, \sqrt{5})$

```
(%i1) edo3:'diff(y,x,2)-5*'diff(y,x)+6*y=0;
(%o1) d^2/dx^2 y-5 (d/dx y)+6 y=0

(%i2) ode2(edo3,y,x);
(%o2) y=%k1 %e^(3 x)+%k2 %e^(2 x)

(%i3) bc2(%,x=1.5,y=0.5,x=2,y=6),numer;
rat: replaced 0.5 by 1/2 = 0.5
rat: replaced -90.0171313005218 by -42038/467 = -90.017130620985
rat: replaced -20.0855369231877 by -12915/643 = -20.0855365474339
rat: replaced -403.428793492735 by -257791/639 = -403.428794992175
rat: replaced -54.5981500331442 by -53124/973 = -54.5981500513875
(%o3) y=0.02923617779987 %e^(3 x)-0.1061339253166 %e^(2 x)
```

Figura 52: Equação diferencial de 2ª ordem com condições de contorno $(1,5; 0,5)$ e $(2; 6)$.

5.3 CAMPO DE DIREÇÕES

O campo de direções de uma EDO $\frac{dy}{dx} = f(x, y)$ é um gráfico que relaciona, a cada ponto do plano cartesiano (x_P, y_P), um segmento de reta com inclinação igual ao valor da derivada primeira $\frac{dy}{dx}$ neste ponto. O campo direcional sugere a aparência que a solução da EDO deve ter, pois se a função solução passa por um determinado ponto do plano, deve ser tangente ao segmento relativo a este mesmo ponto.

A plotagem de campos direcionais pode ser feita no wxMaxima através de

plotdf(<equação>,[trajectory_at,$< x_o >, < y_o >$])

sendo <equação> o segundo membro da EDO quando esta tem isolada a derivada $\frac{dy}{dx}$ no primeiro membro. Para usar-se *plotdf* deve-se antes carregar seu devido pacote através de *load(plotdf)*. A opção *trajectory_at* serve para informar uma condição inicial $y(x_o) = y_o$. A título de exemplo,nas figuras 53,54 e 55 foram criados os campos de direções das seguintes equações com condições iniciais:

$$\frac{dy}{dx} = x \quad , \quad y(-1) = -2$$

$$\frac{dy}{dx} = e^x \quad , \quad y(0) = 1$$

$$\frac{dy}{dx} = y^2 - 1 \quad , \quad y(2) = 4$$

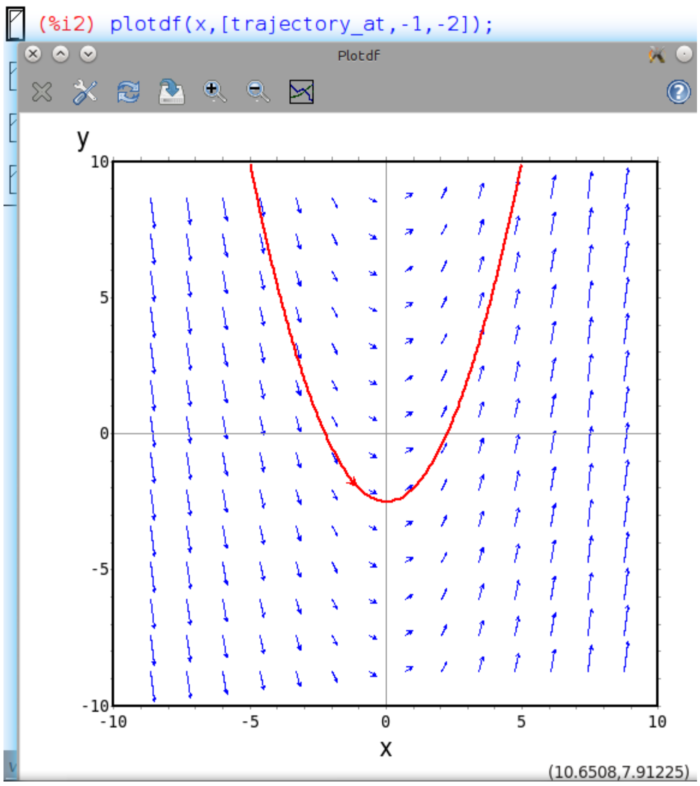


Figura 53: Campo de direções com condição inicial.

Para a equação

$$\frac{dy}{dx} = y^2x - e^y(y^2 - 2y + 2)$$

não foi usada uma condição inicial através de *trajectory_at*, mas usou-se o mouse para escolher diversas condições iniciais e plotar várias curvas em uma mesma janela. Para isso, basta clicar no par ordenado do plano cartesiano correspondente a condição inicial desejada.(figura 56)

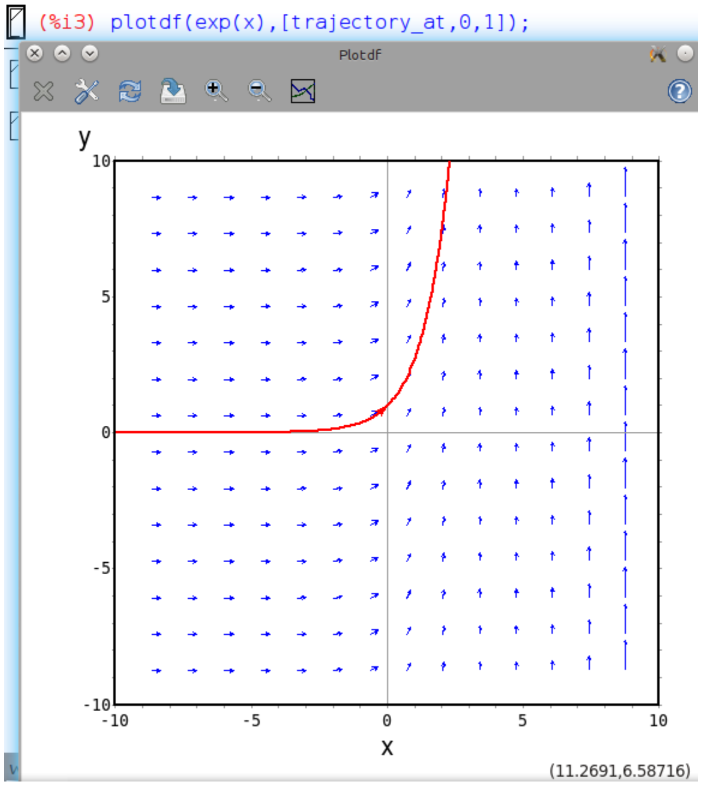


Figura 54: Campo de direções com condição inicial.

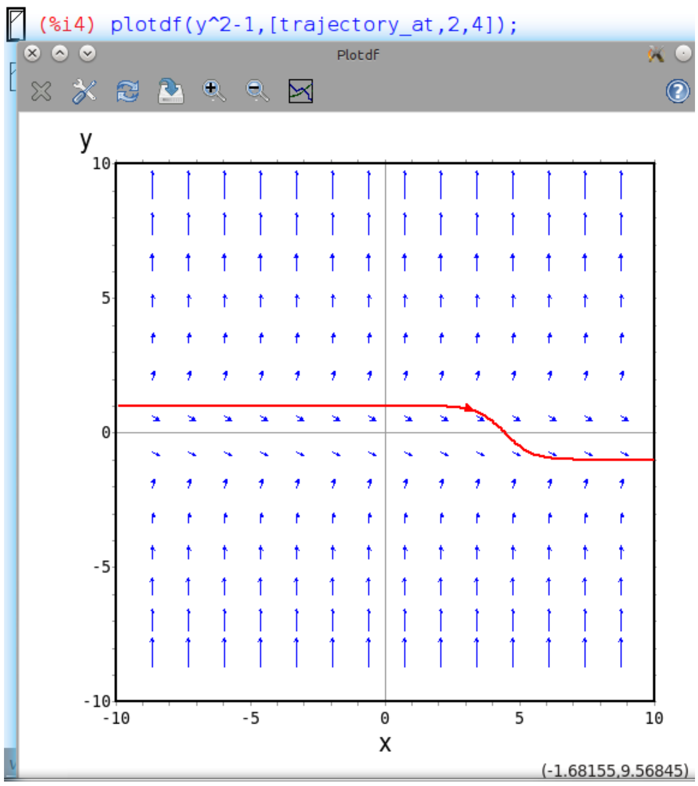


Figura 55: Campo de direções com condição inicial.

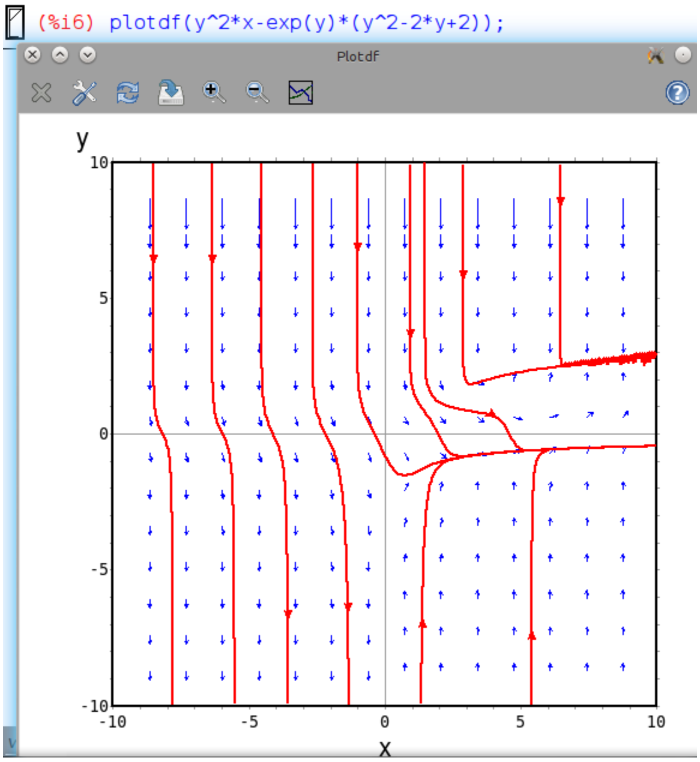


Figura 56: Condições iniciais escolhidas através de “clics” do mouse

6 CONSIDERAÇÕES FINAIS

Como visto, o software analisado neste trabalho, possui inúmeras ferramentas que podem ser utilizadas na apresentação, ilustração/representação e complementação da teoria de diversos conteúdos matemáticos. Entretanto é importante ressaltar, que não se pretende, aqui, substituir o tratamento formal baseado em demostrações, que são tão importantes para o estudo da Matemática, mas apresentar formas de comprovação computacional que facilitem a compreensão de fatos que devem também ser vistos sobre o olhar da mais pura Matemática.

O programa é diferencial tecnológico que torna o estudo da disciplina envolvida mais interessante para os alunos na hora dos estudos, podendo comprovar resultados já conhecidos. Já para professores, há a possibilidade de conferir resoluções de questões que, por exemplo, serão usadas em uma avaliação ou explanação.

O software também tem muita utilidade para profissionais, que possuem o conhecimento da teoria matemática e precisam apenas de uma rápida solução para determinado problema. Em outros casos, a resolução pode necessitar grande quantidade de cálculos, o que torna inviável sua resolução sem auxílio computacional.

Apesar do software possuir uma imensidade de comandos e aplicações, é de fácil aprendizagem, podendo ser operado com poucas horas de dedicação. O aplicativo também permite resolver problemas pontuais, bastando para isso apenas o entendimento do capítulo 2 e a consulta dos comandos específicos necessários para realizar a tarefa desejada. Pode-se citar, por exemplo, a plotagem de gráficos ou resolução de uma EDO.

Outro ponto forte do Maxima é sua gratuidade, pois pode ser distribuído livremente e utilizado amplamente em instituições de ensino ou para uso pessoal sem qualquer custo de instalação ou direitos autorais. Vale lembrar que o aplicativo está disponível nos três mais conhecidos sistemas operacionais: Linux, Windows e MacOS.

Referências

[1] BOLDRINI, José Luiz .**Álgebra linear**. $3^{a.}$ edição. São Paulo:HARBRA, 1986 .405p

[2] BOYCE, William E. e DiPrima, Richard C. **Equações Diferenciais Elementares E Problemas De Valores De Contorno.** 9^{a} edição. Rio de Janeiro:LTC , 2010. 607p

[3] LEITHOLD, Louis . **O Cálculo Com Geometria Analítica**:Volume 1. $3^{a.}$ edição. São Paulo:HARBRA, 1994 . 685p

[4] LEITHOLD, Louis . **O Cálculo Com Geometria Analítica**:Volume 2. $3^{a.}$ edição. São Paulo:HARBRA, 1994 . 1178p

[5] SANTOS, Bruna . **Introdução ao Software MAXIMA**. Porto, 2009. Disponível em: <http://cmup.fc.up.pt/cmup/v2/include/filedb.php?id=289&table=publicacoes &field=file>. Acesso em: 15 out. 2012.

[6] SWOKOSWSKI, Earl Willian. **Cálculo com geometria analítica**. 2 ed. São Paulo: Makron Books, 1994. 744p

[7] VILLATE, Jaime E. **Introdução aos Sistemas Dinâmicos:** uma abordagem prática com Maxima. Porto, 2007. Disponível em: <http://villate.org/doc/ sistemas-dinamicos/ sistdinam-1_2.pdf>. Acesso em: 12 out. 2012.

[8] ZILL, Dennis G.; CULLEN, Michael R. **Equações diferenciais**. 3 ed. São Paulo: Makron Books, 2010. 473p