UNIVERSIDADE FEDERAL DE SANTA CATARINA – UFSC
UNIVERSIDADE VIRTUAL DO ESTADO DO MARANHÃO - UNIVIMA
DEPARTAMENTO DE MATEMÁTICA E FÍSICA/UFSC
CURSO DE ESPECIALIZAÇÃO EM MATEMÁTICA

FRANCISCO DAS CHAGAS ALVES DOS SANTOS
JAQUELINE SILVA DE SOUZA

SISTEMAS DE COMPUTAÇÃO ALGÉBRICA. APLICAÇÕES

Caxias - MA
2009

FRANCISCO DAS CHAGAS ALVES DOS SANTOS
JAQUELINE SILVA DE SOUZA

SISTEMAS DE COMPUTAÇÃO ALGÉBRICA. APLICAÇÕES

Trabalho de conclusão de curso de Especialização em Matemática, formação de professores como requisito para obtenção do título de Especialista em Matemática pela Universidade Federal de Santa Catarina – UFSC, em conjunto com a Universidade Virtual do Estado do Maranhão – UNIVIMA.

Orientador: Prof. Dr. Daniel Norberto Kozakevich.

Caxias – MA
2009

FRANCISCO DAS CHAGAS ALVES DOS SANTOS
JAQUELINE SILVA DE SOUZA

SISTEMAS DE COMPUTAÇÃO ALGÉBRICA. APLICAÇÕES

Monografia apresentada ao Curso de Especialização em Matemática – Formação de Professores da Universidade Federal de Santa Catarina – UFSC.

Aprovada em: _____/_____/2009.

BANCA EXAMINADORA:

_____

Daniel Norberto Kozakevich
(Orientador)
Universidade Federal de Santa Catarina

_____

2º Examinador:
Universidade Federal de Santa Catarina

_____

3º Examinador:
Universidade Federal de Santa Catarina

Dedicamos este trabalho aos nossos familiares, pelo apoio, incentivo e credibilidade depositados em nós durante toda a vida.

AGRADECIMENTOS

Ao nosso soberano Deus pela plenitude da vida e sabedoria.

Ao professor Daniel Norberto Kozakevich pelo material didático fornecido e orientações concedidas que foram essenciais para a elaboração deste trabalho.

A toda equipe de professores e coordenadores da Universidade Federal de Santa Catarina - UFSC, pela sabedoria e empenho no desenvolvimento deste projeto.

A todos os colaboradores da Universidade Virtual do Estado do Maranhão - UNIVIMA, através do Pólo Tecnológico de Caxias pelo apoio, dedicação e acolhimento durante este curso.

À nossa preciosa família, por todo amor dedicado e incentivo ao longo de toda nossa vida estudantil.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

CAS - Sistemas de Computação Algébrica

CA – Computer Algebra ou Computação Algébrica

SBM - Sociedade Brasileira de Matemática

MuPAD - *Multi Processing Álgebra Data Tool* ou Multiferramenta de Processamento de Dados Álgebra

INRIA - *Institut de Recherche em Informatique et en Automatique*

MÉTALAU - *Méthods, algorithmes et logiciels pour l'automatique*

ENPC - Ecole Nationale des Ponts et Chaussées

CASCD - Computer Aided Control System Design ou Projeto de Sistemas de Controle Auxiliado por Computador

MIT-*Massachusetts Institute of Technology* ou Instituto de Tecnologia de Massachusetts

GNU GPL - General Public License ou Licença Pública Geral

EDO - Equação Diferencial Ordinária

RESUMO

Este trabalho apresenta um estudo sobre Sistemas de Computação Algébrica (CAS – sigla em inglês para *Computer Algebra System*) e aplicações. O estudo foi desenvolvido a partir de pesquisas bibliográficas e subsídios na internet com o objetivo de analisar alguns softwares existentes no mercado, como MuPAD, Scilab e Maxima, que são poderosas ferramentas utilizadas na resolução de problemas de Matemática, Engenharia, Física e áreas afins. São apresentados um histórico, os principais comandos e funcionalidades desses sistemas, bem como representações gráficas, priorizando o uso do sistema Maxima com aplicações práticas envolvendo equações diferenciais. Além disso, este trabalho vem possibilitar informações, que possam contribuir para o processo de ensino-aprendizagem de conceitos matemáticos através da importância do uso de softwares na educação.

Palavras – chave: Computação algébrica, software, matemática.

SUMÁRIO

CAPÍTULO 1 - INTRODUÇÃO

A computação algébrica possui alguns aspectos relacionados à Matemática, com a vantagem de desenvolver algoritmos e aplicações que realizam a manipulação e análise de expressões matemáticas. Os sistemas de Computação Algébrica são também chamados de Sistemas da Álgebra Computacional, ou o termo mais comum, em inglês, *Computer Algebra Systems* (CAS). Tais sistemas podem executar uma grande variedade de manipulações, tais como: simplificação de expressões e funções, substituição de valores numéricos ou simbólicos, entre outras.

A manipulação de expressões matemáticas com o intuito de simplificá-las é uma tarefa lenta e, ao mesmo tempo pode-se tornar cansativo, trabalhoso e às vezes propenso a erros. Desse modo, o ensino de matemática pode se beneficiar desses CAS, capazes de reduzir o tempo na resolução desses problemas, ao passo que torna o resultado mais preciso.

Atualmente existe uma grande variedade de CAS tanto proprietários, quanto livres que auxiliam na resolução de problemas matemáticos, tanto de natureza numérica, quanto simbólica.

Uma das vantagens de se utilizar um CAS e uma calculadora tradicional consiste na habilidade de tratar os problemas de forma simbólica. Outra vantagem da Computação Algébrica é a possibilidade de resolução de problemas literais, ou seja, manipulando letras ao invés de números.

De acordo com BARBOSA (2006), os sistemas podem ser classificados como de propósito geral ou sistemas de propósito específico. Entretanto, não há unanimidade quanto a essa classificação, um sistema de propósito geral, dependendo de seu uso pode ser classificado como específico e vice-versa.

Ainda nesse contexto, o mesmo autor apresenta um histórico dos CAS distribuídos por período, conforme o resumo a seguir:

- Idade da pedra (meados dos anos 50 – Final dos anos 60): aparecem os primeiros trabalhos relacionados com CAS, os primeiros sistemas eram complicados e trabalhavam com áreas restritas;

- Idade do ferro (meados dos anos 60 – Final dos anos 70): surgiram sistemas mais fáceis de utilizar que os anteriores, tratam-se dos sistemas de propósito geral;
- Idade do plástico (final dos anos 80): esse período foi marcado pela pesquisa e desenvolvimento de sistemas interativos, em que o usuário pudesse desenvolver seus próprios algoritmos de manipulação.

Hoje, os sistemas tem boa qualidade, porém, ainda existem dificuldades a serem superadas, dentre as quais destacam-se: melhoria na interação com o usuário; comunicação entre CASs; utilização de bibliotecas em sistemas diversos; crescimento de expressões intermediárias.

Apesar das dificuldades citadas acima, o uso de CAS tornou-se uma ferramenta de apoio ao ensino nas Universidades, sobretudo nos cursos de Matemática, Física, Engenharia, Biologia e áreas afins, onde muitos cálculos são necessários.

Neste trabalho, será feito um estudo de alguns softwares livres, tais como: **MuPAD, Scilab e Maxima**, destacando suas vantagens e desvantagens, bem como a aplicação do CAS Maxima na resolução de Equações Diferenciais e na construção de gráficos. Este sistema apresenta uma sintaxe simples, é compatível com vários sistemas operacionais dentre eles, o Linux e o Windows e, pode ser utilizado na resolução de problemas complexos.

Baseado em pesquisa bibliográfica, reunindo subsídios na internet, como artigos, livros digitalizados e tutoriais, o trabalho consiste em analisar de forma sucinta os sistemas acima citados.

Na primeira parte do trabalho será feito um histórico da Computação Algébrica, destacando suas várias etapas. Apresenta ainda, as características dos softwares livres **MuPAD, Scilab e Maxima,** bem como suas vantagens e desvantagens em relação a sua utilização na resolução de problemas matemáticos.

A segunda parte contém um tutorial do sistema Maxima, com seus principais comandos e operações nele realizados. Na terceira parte, as aplicações do Maxima em Cálculo, especificamente na resolução de Equações Diferenciais e representação gráfica. E por fim, as considerações finais onde serão apresentadas as conclusões, destacando a importância do uso de um CAS no processo ensino de Matemática.

1.1 Objetivos

1.1.1 Geral

Fazer uma abordagem sobre alguns softwares de Computação Algébrica, enfatizando o uso do sistema Maxima na resolução de equações diferenciais.

1.1.2 Específicos

- Analisar os softwares MuPAD, Scilab e Maxima destacando seus principais comandos e funcionalidades;
- Desenvolver habilidades de manipulação de um CAS para aplicação na resolução e simplificação de expressões Matemáticas;
- Utilizar o sistema Maxima para resolver problemas que envolvem equações diferenciais;
- Fazer um resgate histórico da Computação Algébrica.

CAPÍTULO 2 - UMA ANÁLISE DE SOFTWARES DE COMPUTAÇÃO ALGÉBRICA

2.1 Histórico da Computação Algébrica

Com o surgimento do computador e sua aplicação no sistema educacional, houve significativas mudanças no contexto pedagógico. O uso de ferramentas computacionais no processo de ensino-aprendizagem vem ganhando cada vez mais ênfase na resolução de diversos problemas que envolvem as ciências em especial, a Matemática.

De acordo com ANDRADE (2002), historicamente, o verbo computar, significa "fazer cálculos com números". Com a criação da Computação Numérica foi possível solucionar uma série de problemas abrangendo funções e operações matemáticas, cujos dados armazenados são puramente numéricos. Todavia, geralmente, esses resultados apresentados não são exatos.

Coube, então, à Computação Algébrica ou Computação Simbólica, por volta dos anos 1960, obter exatidão desses resultados, através da representação de objetos matemáticos por símbolos, baseados nas regras usuais da Álgebra. Conforme RANGEL (2005):

> Os cálculos realizados no tratamento simbólico são exatos, isto é, tem precisão infinita, em contraste ao correspondente tratamento numérico. Uma operação do tipo 1/3+1/3, que numericamente resultaria em 0.666666, no cálculo simbólico teremos como resultado o valor exato, 2/3.

Sendo assim, a Computação Algébrica, a qual recebe outras denominações como Álgebra Computacional, Manipulação Simbólica ou Manipulação de Fórmulas, constitui-se um inovado ramo de estudo que vem sendo utilizado em diversas áreas da Ciência e da Tecnologia. Dentre elas podemos citar: Mecânica Celeste, Acústica, Teoria dos Grupos e Teoria dos Números, entre outras.

Segundo ANDRADE (2002), "o conjunto de programas de computador relacionados à Computação Algébrica denominou-se Sistemas de Computação Algébrica ou Sistemas de Manipulação Simbólica, entre outros nomes". Esses sistemas realizam cálculos analíticos de forma precisa e rápida, podendo ser proprietários ou livres, de uso específico ou geral para fins comerciais ou educacionais.

Os Sistemas de Computação Algébrica (CAS, em inglês *Computer Algebra System)* de uso específico contemplam problemas relacionados com a Matemática, a Engenharia ou a Física. Os CAS incorporam recursos algébricos, numéricos ou gráficos. Além disso, podem funcionar como linguagens de programação[1] para solucionar os mais diversos tipos de problemas matemáticos. Para citar só alguns deles: cálculos envolvendo trigonometria, logaritmos, polinômios, limites, derivadas, integrais, equações diferenciais, transformadas de Laplace, etc.

A implantação desses sistemas computacionais no processo educacional ocasionou um grande avanço no ensino e aprendizagem da Matemática, tornando-a ainda mais experimental. Essa afirmação pode ser evidenciada por RANGEL (2005):

> O uso de ferramentas computacionais no processo de ensino-aprendizagem permite ao estudante abordar problemas complexos. No caso da computação simbólica, sistemas computacionais permitem tornar a Matemática mais experimental, permitindo analisar diferentes situações com visualização gráfica, dando assim oportunidade ao estudante de aprender fazendo.

A Computação Algébrica começou a ser desenvolvida em 1953. No inicio década de 60, surgiram os primeiros sistemas de computação algébrica, a partir de então, houve um avanço em pesquisas para inteligência artificial. Nessa década, dá-se início a elaboração dos primeiros softwares no campo de manipulação simbólica, dentre estes destacam-se: o Formac, o Lisp e o Alpak, entre 1961 e 1971. Entre 1966 e 1971, surge a segunda geração Reduce, Macsyma, ScratchPad. A partir de 1970, os softwares Reduce e Macsyma ganham popularidade e surge o MuMath, antecessor do Derive. A partir de 80, originam-se os softwares Maple e Mathematica, que atualmente estão entre os líderes no mercado.

Nas últimas décadas diversos tipos de softwares de Computer Algebra (CA) foram desenvolvidos. Dentre eles podemos citar, Axiom, Derive, Reduce, Maple, Mathematica, Mupad, Scilab e Maxima, entre outros.

No Brasil, a Computação Algébrica surgiu por volta da década de 80, quando a Sociedade Brasileira de Matemática (SBM) passou a adotar esse recurso computacional nas universidades.

---

[1] Uma linguagem de programação é um método padronizado para expressar instruções para um computador. É um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador.

Uma das vantagens do uso de um software livre é a economia, já que os usuários não precisarão adquirir licenças dos sistemas proprietários. Outra vantagem é que estes sistemas estão sempre se atualizando, sofrendo alterações de modo que seus usuários podem contribuir com a construção do sistema.

Além disso, esses softwares podem ser copiados, redistribuídos e modificados sem restrição. Para possibilitar estas atividades, os usuários precisam de acesso ao código fonte. Apesar de livre, alguns sistemas não são totalmente gratuitos. Alguns são licenciados apenas uma parte com restrição de memória, como no caso do software MuPAD, através do pacote Light.

Dentre suas desvantagens pode-se citar: alguns softwares não disponibilizam de todos os recursos e a compatibilidade de programas com plataforma Microsoft Windows é muito menor em relação à do sistema Linux.

Nessa perspectiva, os seguintes sistemas serão analisados: Scilab, MuPAD e Maxima, onde enfatiza-se cada um deles, bem como suas características usuais, aplicações e um tutorial básico. Será dada uma maior ênfase ao Sistema Maxima, destacando sua aplicação no uso de Equações Diferenciais.

2.2 MuPAD

O MuPAD[1] (*Multi Processing Álgebra Data Tool*) foi desenvolvido, a partir da década de 1990, por um grupo de pesquisadores, liderado pelo professor Benno Fuchssteiner, da Universidade de Panderborn, na Alemanha.

Com a criação do MuPAD foi fundada a empresa SciFace Software, que desde 1997 comercializa o sistema, fornecendo a versão Light do programa para fins educacionais, podendo ser distribuída entre estudantes e professores.

De acordo com ALEGRE (2006), a palavra "MuPAD" advem da expressão "*mupas*" que foi substituído em 1990 pela expressão atual representando, por um lado a (*Processing Algebra Data Tool* - Ferramenta de processamentos de dados) e por outro o local de origem do projeto (Universidade de Panderborn).

Esse software dispõe de uma ampla biblioteca de operações matemáticas usuais e de uma linguagem de programação em C ou C++, permitindo que seus usuários possam criar seus próprios algoritmos. Além disso, possui manuais,

[1] http:// www.mupad.com

tutoriais (disponíveis em alemão e inglês) e demonstrativos sobre suas funcionalidades.

Está disponível nas versões padrão (proprietário) e light (livre). Na versão padrão pode ser adquirido um *trial* por 30 dias, após esse período o sistema só funciona com uma licença de uso fornecida pela empresa SciFace Software. Essas versões podem ser obtidas através do site oficial do programa ou pela SciFace.

O sistema está disponível para as plataformas *Windows, Macintosh e Linux*. A figura 1 exibe a interface do sistema na versão Windows.

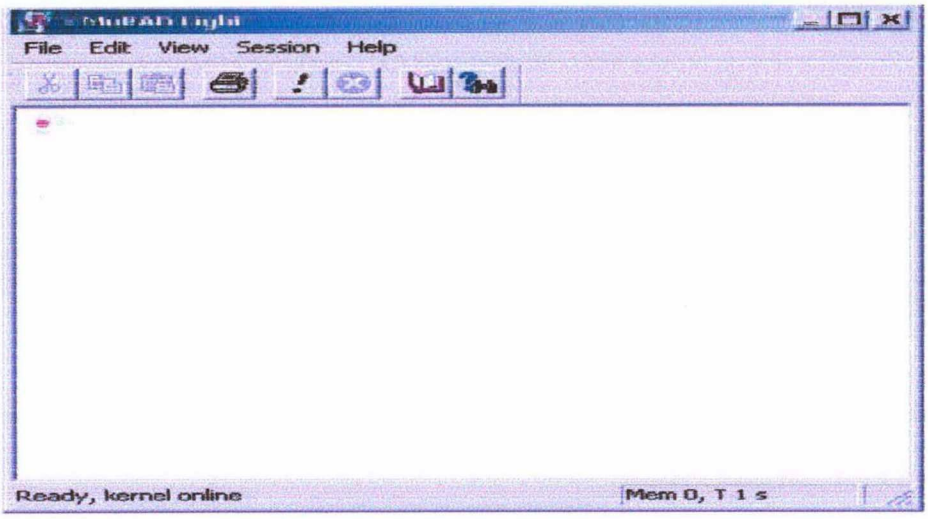

Figura 1: Interface do MuPAD versão 2.5.3
Fonte: http://www.sciface.com/mupad_download/

Abaixo temos os ícones que aparecem na janela:

	Recortar copiar imprimir
	Imprimir
	Executar linha parar computação
	Iniciar tutorial interno procurar na ajuda

Tabela 1: Principais botões do MuPAD

De acordo com VAZ (2001), antes de começar a usar o aplicativo, serão necessárias algumas instruções preliminares:

- O sinal . precede os comandos que deverão ser digitados;
- Pode-se usar o terminador dois pontos (:) após o comando. Este terminador omite a visualização do resultado;

- Sempre que terminar de digitar um comando acione a tecla <enter> para que o MuPAD efetue as operações;
- Todos os comandos do MuPAD são escritos com letras minúsculas;
- Na versão MuPAD Light não pode corrigir erros de digitação na linha de comando. Deve-se copiar a linha ou digitar tudo novamente.

Trata-se de uma ferramenta bastante poderosa e com grande potencial, podendo ter seu uso bastante difundido em Universidades para o aprendizado das disciplinas relacionadas com o ensino da Matemática.

O aplicativo MuPAD trabalha com interatividade, exibindo uma área trabalho na qual serão digitados os comandos e visualizando as respostas. Para a plataforma Windows, a versão MuPAD Light possui uso restrito de memória, mas suficientemente para tratar alguns tópicos da Matemática, tais como: cálculo numérico, resolução de equações, gráficos de funções, limites, derivada, integral e séries , entre outros.

Segundo SANT-ANNA (2006), uma razoável abordagem, com excelente qualidade visual, pode ser conseguida com a utilização conjunta dos softwares texmacs (TEXMACS, 2006), javaview (JAVAVIEW, 2006) e MuPAD (MuPAD, 2006). Estes programas são obtidos gratuitamente pela internet.

2.2.1 Comandos e operações no sistema MuPAD

Os principais comandos existentes no MuPAD são: *expand*, *combine*, *factor*, *normal* e *like*. Além dos comandos de simplificação, existe o comando *testeq*, usado para determinar se duas expressões são equivalentes. Existem também comandos que realizam substituições, são estes: *subs*, *subsex* e *subsop*. Algumas das principais características de cada um, sua área de trabalho, desempenho, como alguns comandos do software e suas utilidades.

Ao ser iniciado, sua tela é simples possuindo poucos botões de comandos, com um espaço para digitação.

A seguir temos algumas operações realizadas no MuPAD:

float – resultado de expressões numéricas

```
> float(sin(90))
  0.8939966636
```

Equações:
solve (equação, variável)

```
> solve(x^2-7*x+6=0)
  {[x = 1], [x = 6]}
```

Funções:
f:=variável->função

```
> f:= x-> 2*x^3 - 4
 x -> 2*x^3 - 4
```

Para obter o resultado, basta atribuir um valor.

```
> f(1)
 -1
```

Limites:
O cálculo de limites é representado de três formas:
Cálculo de limite pela esquerda (se existir) " limit (função, x=a, Left)"

```
> limit(2*x, x=2,Left)
 4
```

Cálculo de limite pela direita (se existir) " limit (função, x=a, Right)"

```
> limit(sqrt(1-x^2),x=-1,Right)
 0
```

Cálculo de limite bidirecional (se existir) "limit (função,x=a)"

```
> limit(1/x,x=0)
  undefined
```

Derivadas:
Dada a função f, basta digitar o comando D, f' ou diff, neste caso para distinguir a função e a variável.

```
> f:= x->2*x^2
   x -> 2*x^2 - 1
> D(f)
  4 id
```

Integração:
Para integração utiliza-se o operador "int".
"int(função , variável)", para integrais indefinidas
"int(função, variável =mínimo..máximo)" para integrais definidas.

```
> int(x^5,x)
```

```
  6
 x
 --
  6
```

2.2.2 Gráficos no MuPAD

O MuPAD gera visualizações gráficas tanto bidimensionais quanto tridimensionais, através dos comandos `plotfunc`. O MuPAD possui uma biblioteca para lidar com os casos mais complexos. Os gráficos aparecem em uma janela para facilitar a visualização.

Exemplo: Construir o gráfico da função $f(x)=x^3$ no intervalo [-5,5].

`plotfunc2d(x^3,x = -5..5,y=-5..5)` pressione < enter >.

O gráfico gerado é apresentado na figura 2 a seguir:

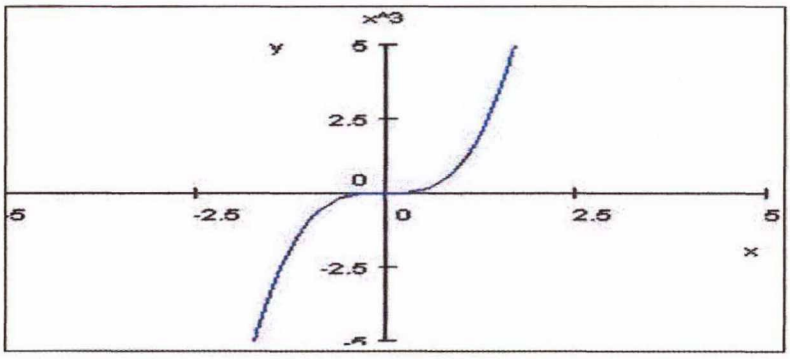

Figura 2: Gráfico 2D da função $f(x)=x^3$
Fonte: http://www.sciface.com/mupad_download/

Gráficos tridimensionais:

Para que o MuPAD desenhe o gráfico da função z = f(x; y) no domínio [a, b]x[c, d] será usado os seguinte comando:

`plotfunc3d (opt, f1, f2..., x=a..b, y=c..d), onde:`

`fi - expressões`

`x, y - identificador das variáveis x e y`

`a, b, c, d - números reais`

`opt - opções do comando`

As opções do comando plotfunc3d são as mesmas do comando plotfunc2d.

Exemplo: Construir o gráfico da função f(x) =x^3 no intervalo [-3,2]x[-3,2].

`plotfunc3d(x^3, x = -3..2, y=-3..2) pressione < enter >.`

O gráfico gerado é apresentado na figura 3 a seguir:

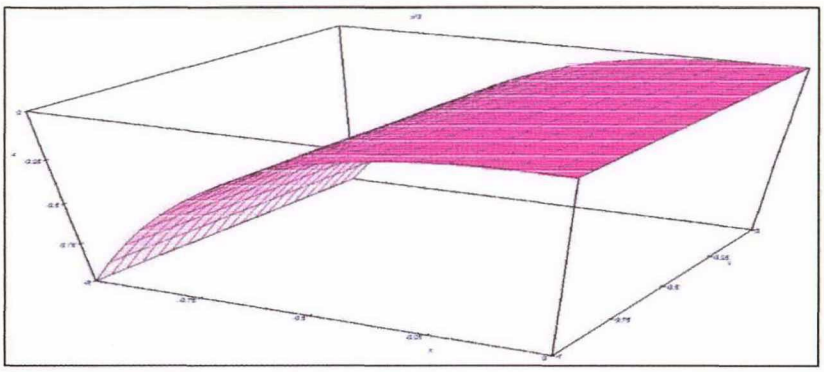

Figura 3: Gráfico 3D da função f(x)=x^3

Fonte: http://www.sciface.com/mupad_download/

As principais características do **MuPAD** são:

- É gratuito de fácil acesso na Internet (versão Light);
- É semelhante ao Mathematica e ao Maple;
- É compatível com os sistemas operacionais, tais como Linux, Unix, Mac e Windows;
- Permite processamento em paralelo;
- Sua instalação torna-se bastante fácil;
- Faz gráficos de funções bi (2D) e tridimensionais (3D) de excelente qualidade e podem ser gerados em vários formatos, incluindo o *eps* (que podem ser incluído nos documentos para o latex) ou formatos com menor definição, tais como *png* ou *tiff*.

Segundo avaliação realizada por BATISTA e BARCELOS (2004), o MuPAD versão 2.5.3 apresenta:

Pontos positivos

- Permite trabalhar com diversos conteúdos matemáticos, o que contribui para promover o relacionamento entre os mesmos;
- Possui interface agradável, com ícones bastante significativos;

- Possibilita realizar desde cálculos simples até os mais complexos, dependendo da necessidade do usuário;
- Pode contribuir para a associação de idéias e para o desenvolvimento do senso crítico;
- Possui um ótimo tutorial auxiliando o usuário.

Pontos negativos:

- Não apresenta versão em português;
- Exige atenção a sua sintaxe (cuidados com as exigências da linguagem de programação);
- Só é possível salvar um arquivo de trabalho em formato de texto (versão Light);
- Impossibilidade de editar uma linha de comando já interpretada (versão Light).

2.3 SCILAB

É um software científico para computação numérica que oferece um ambiente aberto para aplicações científicas e em engenharia. Criado em 1990, e mantido por pesquisadores pertencentes ao INRIA (*Institut de Recherche em Informatique et en Automatique*), através do projeto MÉTALAU (*Méthods, algorithmes et logiciels pour l'automatique*) e ENPC (Ecole Nationale des Ponts et Chaussées).

O programa é distribuído gratuitamente via Internet desde 1994, juntamente com seu código fonte (*open source software*) e, a partir de maio de 2003, passou a ser mantido pelo Consórcio Scilab (Consórcio de empresas e instituições francesas). Possui finalidade educacional e industrial em todo o mundo. Apresenta-se como um software CASCD (Computer Aided Control System Design)[1].

2.3.1 O ambiente gráfico do Scilab

[1] Projeto de Sistemas de Controle Auxiliado por Computador

Após a instalação do Scilab[1] na plataforma Windows, basta dar um duplo clique no ícone Scilab para começar a utilizar o programa. A tela inicial logo irá aparecer, como mostra a figura 4:

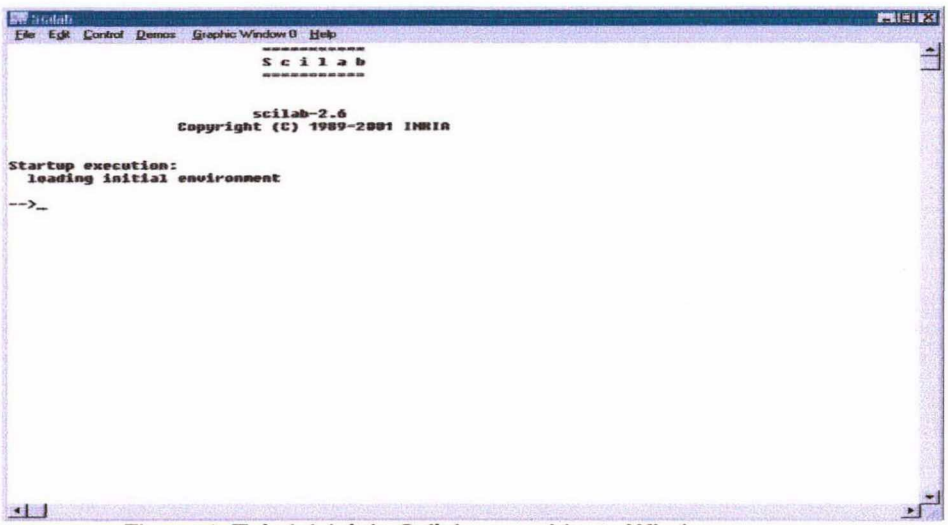

Figura 4: Tela inicial do Scilab no ambiente Windows
Fonte: http://scilabsoft.inria.fr/

O menu principal do Scilab possui seis opções. Veremos a seguir cada uma delas e suas respectivas funções:

File: Gerenciamento de arquivos;

Control: Começa ou interrompe a execução do Scilab;

Demos: Executa rotinas de demonstração do programa;

Graphic Window N: Manipulação de janelas gráficas;

Help: Provê informações sobre as diversas funções e comandos do Scilab;

Editor: Utiliza o editor Scipad para escrever comandos e funções.

Segundo CARRILHO (2004), PEREIRA (2005), as principais características do Scilab são:

- É gratuito, com código livre de fácil acesso e linguagem simples;
- Pode ser usado para o desenvolvimento ou prototipação de software numérico de propósito geral;
- Inclui centenas de funções matemáticas com a possibilidade de adicionar, interativamente, programas de várias linguagens como C, C++, Fortran, etc.;

[1]Página do Scilab: http://scilabsoft.inria.fr/

- Possui estruturas sofisticadas de dados (incluindo listas, nomes compostos, funções racionais, etc.), um interpretador e uma linguagem de programação de alto nível;
- Incluem gráficos em 2D e 3D;
- Inclui Álgebra linear, matrizes, interpolação, aproximação, simulação etc.;
- Possui recursos avançados, como otimização de diferenciáveis e não diferenciáveis processamento de sinais e estatísticas;
- Faz animações;
- Pode ser acessado por programas de Computação Simbólica como o Maple, que é um software comercial, ou o MuPAD, que é livre para uso em instituições de ensino/pesquisa;
- O Scilab fornece ao usuário a possibilidade de criar e usar novas funções;
- Suporta o desenvolvimento de conjuntos de funções voltadas para aplicações específicas denominados *toolboxes;*

2.3.2 Comandos

Símbolos e Constantes:

- `ans`: resposta mais recente.
- `%eps`: precisão numérica corrente. Ex. 2.220E-16
- `%pi`: 3.1415927
- `%i`: parte imaginária de números complexos.
- `%inf`: infinito. Ex: 1/0.
- `%nan`: "not a number"

Funções pré-definidas:

O Scilab é carregado com algumas funções pré-definidas, chamadas primitivas, dentre elas destacam-se:

sqrt(x) // raiz quadrada de x

```
-->sqrt(16)
ans =
4.
```

log(x) // logaritmo neperiano de x

```
-->log(%e)
ans =
1
```

exp(x) // e elevado a x

```
-->exp(2)
ans =
7.3890561
```

sin(x)// seno de x
cos(x) // cosseno de x
tan(x) // tangente de x
cotg(x)// cotangente de x

SCI// mostra o diretório no qual o Scilab foi instalado;

PWD// mostra o diretório no qual o Scilab foi lançado e está rodando;

// comentários (não devem ter acentuação) pwd mostra o diretório no qual estamos trabalhando;

chdir: (nome do diretório) mudamos o diretório em que estamos trabalhando.

As variáveis criadas no ambiente scilab podem ser armazenadas em um arquivo.

Save('dados.dat',a,b) //para salvar os valores de a e b em um arquivo, o comando save cria o arquivo dados.dat no diretório de trabalho;

Para recuperar os valores de a e b usamos o comando load ('dados.dat','a','b');

unix_w('ls') permite a comunicação do scilab com a shell unix.

1-unix_w('ls') //mostra o conteúdo de /home/priscilasouza/software livre;

2-mkdir(nome diretório novo) //cria um novo diretório.

Comandos de edição:

Ctrl p ou seta para cima: recupera o comando digitado anteriormente;

Ctrl n ou seta para baixo: recupera o comando seguinte (se houver);

Ctrl b ou seta para a direita: move o cursor um caracter para trás;

Ctrl f ou seta para a esquerda: move o cursor um caracter para frente;

Delete ou backspace: apaga o caracter anterior;

Ctrl h: mesmo efeito da linha anterior (acima);

Ctrl d: apaga o caracter sob o cursor;

Ctrl a: move o cursor para o inicio da linha;

Ctrl e: move o cursor para o final da linha;

Ctrl k: apaga da posição do cursor até o final da linha;

Ctrl u: cancela a linha;

!prev: recupera a linha de comando que começa com prev;

O ponto e vírgula no final de um comando inibe a apresentação de um resultado;

O Scilab também aceita operações simples como: a+b, a-b, a * b, a/b;

O software também aceita grandezas complexas, para isto, usa-se do seguinte modo %i;

É possível também digitar vários comandos na mesma linha;

Realiza raiz quadrada de valores negativos;

Aceita incremento negativo. Ex: definindo vetor j (j=5:- 1:1) //definindo j como um vetor de 5 posições;

No *Scilab* também há o conceito de ambientes. Muda-se o ambiente através do comando *pause*, as variáveis mantidas no ambiente anterior são mantidas. O retorno ao ambiente anterior se dá através das palavras *return* ou *resume*, porém com este tipo de retorno você perde as variáveis definidas no novo ambiente.

Para preservar o que foi definido no novo ambiente ao retornar para o antigo digite *b=resume(b)*.

2.3.3 Gráficos

O ambiente Scilab permite gerar gráficos bidimensionais que podem ser gerados através do comando plot2d.

A forma mais simplificada de escrever a função é:

```
plot2d([x],y)
```

Exemplo: Gráfico da função sin(x), no intervalo [0,2π]

```
-->//definição de abcissas e ordenadas t=(0:0.5:1);
--> y=sin (2* %pi *t);
--> plot2d (parâmetro1, parâmetro2, [nome do eixo x,
nome do eixo y, nome do gráfico]) plot
(t,y,'tempo','f(t)=sin (2* %pi *t)','seno']);
```

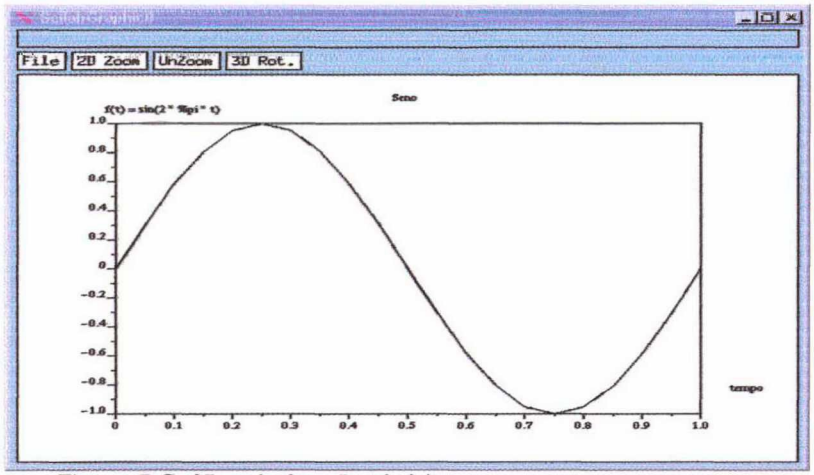

Figura 5 Gráfico da função sin(x)
Fonte: http://scilabsoft.inria.fr/

O Scilab também gera gráficos em 3D, representando superfícies, para isso usa-se o comando plot3d, assim a função equivale: `plot3d(x, y, z)`

Exemplo: A função cos(x)sen(y) no intervalo [0,2π], com incremento igual a 0,1, usando os comandos:

```
-->x = [0:0.1:2*%pi]';
-->y = x;
-->z = cos(x') * sin(x);
-->plot3d(x, y, z)
-->
```

Gera o gráfico da figura 6:

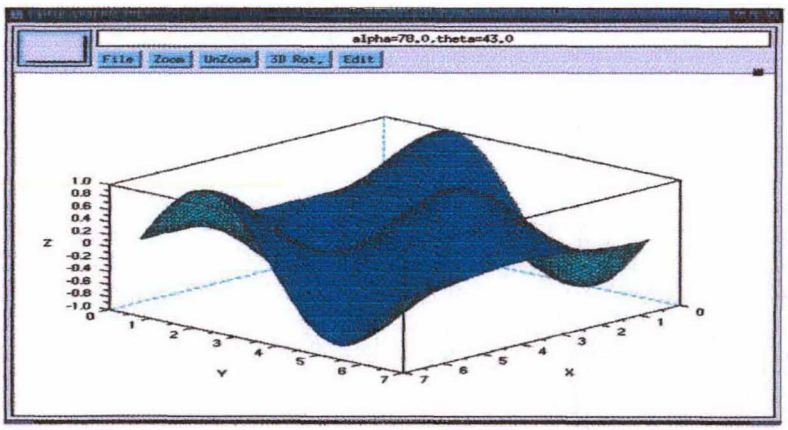

Figura 6: Gráfico da função cos(x)sen(y)
Fonte: http://scilabsoft.inria.fr/

Em resumo as principais características do Scilab, segundo PIRES (2004) são:

VANTAGENS:

- A última versão do software está sempre disponível, geralmente através da Internet;
- O software pode ser legalmente utilizado, copiado, distribuído, modificado;
- Os resultados obtidos podem ser divulgados sem nenhuma restrição;
- Os programas desenvolvidos podem ser transferidos para outras pessoas sem imposições ou constrangimentos de quaisquer natureza;
- O acesso ao código fonte, evitando surpresas desagradáveis;
- O acesso a informação de alta qualidade;
- A certeza de estar participando de uma comunidade cujo principal valor é a irrestrita difusão do conhecimento;
- Possui linguagem simples e de fácil aprendizado.

DESVANTAGENS:

- Não apresenta versão em português;
- Impossibilidade de editar uma linha de comando já interpretados;
- Não permite corrigir erros de digitação.

CAPÍTULO 3 - TUTORIAL DO MAXIMA

Este tutorial apresenta informações básicas desde a instalação até a manipulação de expressões simbólicas, onde serão abordados tópicos relacionados a Limites, Derivadas e Integrais, bem como representação gráfica em duas e em três dimensões.

3.1 Maxima

O Maxima é um Sistema de Computação Algébrica de propósito geral descendente do *Macsyma*[1], desenvolvido no final dos anos 1960, nos laboratórios de Inteligência Artificial do *Massachusetts Institute of Technology* (Instituto de Tecnologia de Massachusetts-MIT) por Carl Engelman (19__- 1981), Joel Moses (1941-) e William Martin(1938-1981).

Segundo Barbosa (2006), "o Macsyma foi o primeiro CAS de propósito geral criado, também foi o primeiro ambiente de CA que pôde ser utilizado facilmente por matemáticos, cientistas, entre outros". Tratava-se de um sistema comercial escrito na linguagem LISP[2].

Esse sistema foi mantido pelo professor da Universidade de Texas William Frederick Schelter (1947-2001) desde 1982, o qual obteve no ano de 1998, a liberação do código fonte sob a GNU[3] General Public License (Licença Pública Geral - GPL). A partir daí, o sistema passou a ser gratuito e chamado de *Maxima.* Após a morte de William, outros colaboradores e grupos de usuários passaram a contribuir para o aperfeiçoamento desse CAS.

De acordo com MACEDO (2008), "esse software é o único sistema baseado em *Macsyma* ainda publicamente disponível e com uma comunidade de usuários ativa". O *Maxima* pode ser compilado nos sistemas *Windows, Linux e MacOS X* através do site oficial do programa.

Possui uma ampla biblioteca matemática com uma vasta gama de funcionalidades de transformações algébricas, incluindo a manipulação de expressões matemáticas envolvendo diferenciação, integração, equações

---

[1] *Macsyma* é um sistema legendário de Álgebra do computador desenvolvido inicialmente para os computadores de grande escala DEC-PDP-10 que eram usados em várias instituições acadêmicas.
[2] Linguagem LISP é uma família de linguagens de programação concebida pelo cientista americano da Computação John McCarthy em 1958.
[3] GNU Gereral Public Licence (Licença Publica Geral) é a designação da licença para software livre idealizada por Richard Stallman no final da década de 80.

diferenciais ordinárias, sistemas de equações lineares, polinômios, vetores, matrizes, entre outros.

Além disso, no site do *Maxima* encontram-se disponíveis manuais e tutoriais em espanhol, inglês e português, que podem ser visualizados nas versões: html (on-line) e pdf. Há ainda, um livro denominado "The Máxima Book", porém, escrito incompletamente.

Suas principais características são:

- É livre;
- Funciona em máquinas obsoletas;
- Possui excelente documentação em português;
- Tem eficiência computacional comparável ao *Maple* e *Mathematica*;
- É mantido regularmente;
- Gera gráficos em 2D E 3D através do programa *Gnuplot*[1] *Graph;*
- Excelente complemento para o Scilab.

O Maxima também possui algumas desvantagens, de acordo com SANT-ANNA (2006), o Maxima possui limitações relacionadas com sua parte gráfica (baseada no Gnuplot). Ainda GUEDES (2009) aponta que, o sistema requer um nível aceitável de competência matemática.

3.2 Download e Instalação

Neste tutorial a versão do MAXIMA apresentada é a versão 5.18.0 para Windows. O Download do programa pode ser feito através do link a seguir: http://ufpr.dl.sourceforge.net/maxima/maxima-5.18.0.

Para outras plataformas pode ser feito o acesso pelo Site Oficial do MAXIMA: http://maxima.sourceforge.net

Após a conclusão do Download, dê um duplo clique no arquivo baixado (MAXIMA 5.18.0), e em seguida aparecerá a tela como mostra a figura 7:

[1] Gnuplot é uma portátil linha de comando interativa que pode plotar gráficos de funções matemáticas em duas ou três dimensões e outros conjuntos de dados. O programa pode ser executado nos sistemas operacionais Linux, Windows, Mac OS X, entre outras plataformas. O software é protegido, mas livremente distribuído. Foi originalmente destinado a cientistas e alunos para visualização de dados e funções matemáticas.

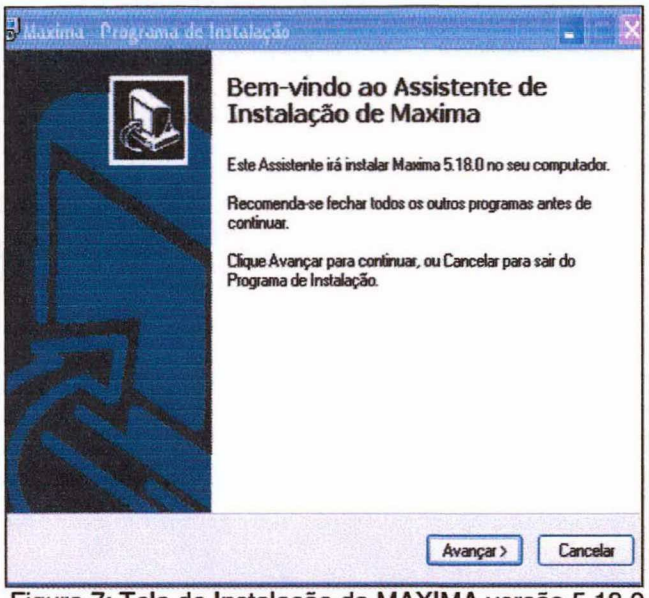

Figura 7: Tela de Instalação do MAXIMA versão 5.18.0
Fonte: http://maxima.sourceforge.net/

Seguindo as recomendações do programa, escolhe-se o local de instalação, e em seguida clica-se na opção Avançar para finalizar a instalação. Aparecendo a tela a seguir clique em Concluir.

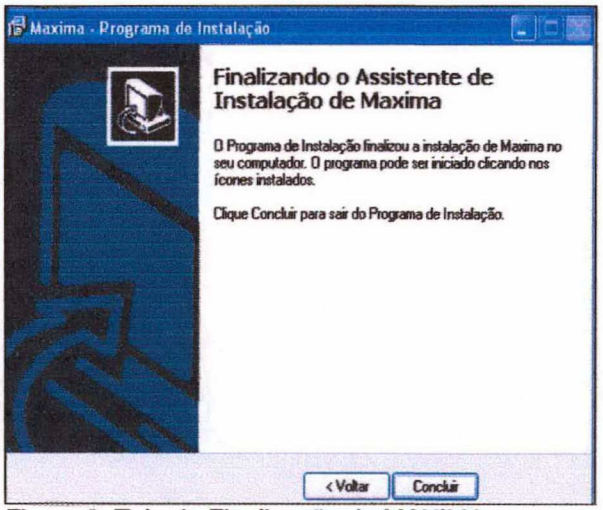

Figura 8: Tela de Finalização do MAXIMA
Fonte: http://maxima.sourceforge.net/

Posteriormente, surgirá um arquivo *readmet.text* que contem informações sobre o MAXIMA e o usuário poderá, então, acessar o programa a partir do menu Iniciar.

3.3 INTERFACE, ENTRADA E SAÍDA DE COMANDOS.

A Interface do **Maxima,** na figura 9, possui na parte superior o terminal com informações sobre o programa, ou seja, é a área de trabalho onde serão digitadas as operações e, na parte inferior um sistema de ajuda, que pode ser ocultado através de um clique no menu Options>Toggle Browser Visibility.

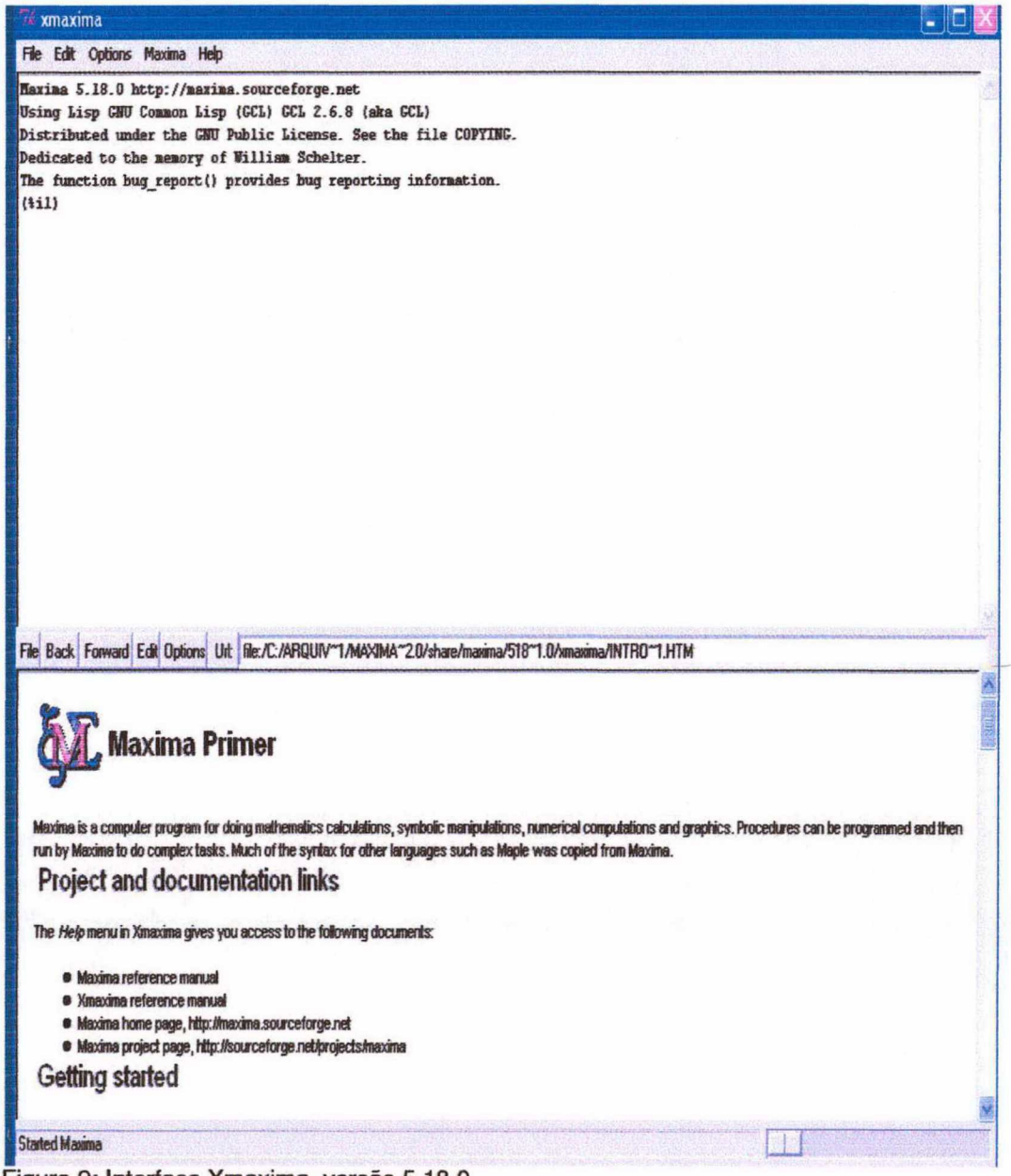

Figura 9: Interface Xmaxima, versão 5.18.0.
Fonte: http://maxima.sourceforge.net/

Algumas instruções preliminares são necessárias para que o usuário comece a usar o aplicativo:

- Na parte superior do terminal, o '(%i1)' representa a posição da memória de cada operação a ser realizada. Após a expressão '(%ix)', as operações devem ser digitadas. O 'i' significa INPUT, ou seja, entrada de informações. Por sua vez, o 'o' designa OUTPUT em '(%ox)', representando a saída do programa, como mostra a figura 10.

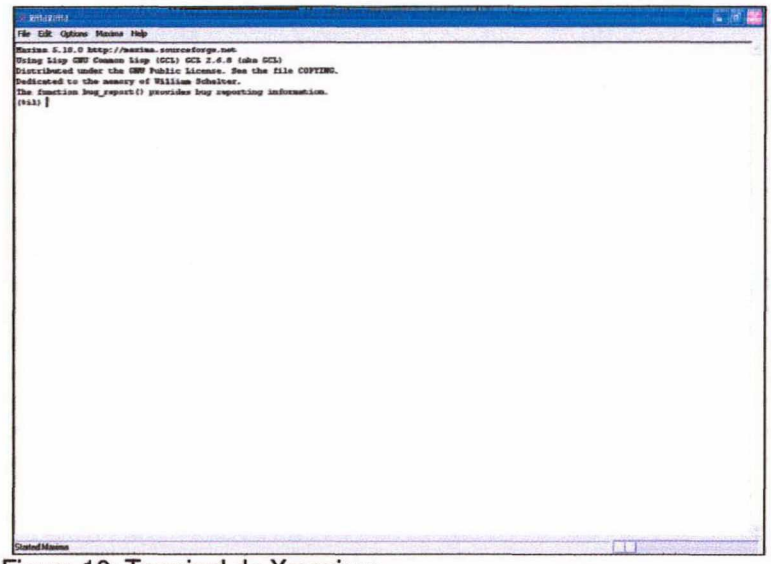

Figura 10: Terminal do Xmaxima.
Fonte: http://maxima.sourceforge.net/

- Na área de trabalho, logo após a digitação das operações, deve ser inserido no final de cada expressão o ';'(ponto e vírgula) para que o programa apresente a resolução, por exemplo:

```
(%i1) 3+2; (INPUT – representa em azul no MAXIMA).
(%o1) 5    (OUTUP – representa em preto no MAXIMA).
```

- As funções no MAXIMA deverão ser escritas com letras minúsculas, bem como seus parâmetros (entre parênteses), por exemplo:

```
(%i1) sin (%pi);
(%o1) 0
```

- Os espaços são desnecessários, auxiliando apenas na leitura.

A tabela abaixo, lista alguns dos principais comandos do MAXIMA.

Operadores aritméticos	
x + y	Adição
x – y	Subtração
x * y	Multiplicação
x / y	Divisão
x ^ y	Exponenciação
Funções	
abs(x)	valor absoluto $\|x\|$
x!	x!
log(x)	ln(x)
sqrt(x)	$\sqrt{x}$
exp(x)	e^x
cos(x)	cos(x)
sin(x)	sin(x)
tan(x)	tan(x)
asin(x)	arcsin(x)
acos(x)	arccos(x)
atan(x)	arctan(x)
Outros comandos úteis:	
float (expr)	Resultado aproximado da expressão **expr** quando ocorre um ponto flutuante.
solve (eq, x)	Resolve a equação (eq) na variável x
describe (assunto a ser pesquisado)	Obter informações sobre determinados comandos
expand(expr)	Expande expressões algébricas
factor(expr)	Fatora expressões algébricas

Tabela 2: Principais comandos do Máxima.

No MAXIMA, a maioria das constantes deve vir precedida com o símbolo % (porcentagem), como mostra a Tabela 3, a seguir:

Constantes	
%pi	$\pi = 3,1415926...$
%i	$i = \sqrt{-1}$ (constante imaginária)
%e	$e = 2,7182818...$
Observação: O comando % permite a manipulação dos resultados já obtidos.	

Tabela 3: Algumas constantes no MAXIMA.

Outro ambiente gráfico do Maxima é o wxMaxima (Figura 11), que é distribuído em conjunto com o executável para Windows.Trata-se de uma interface gráfica multi-plataforma, baseado em wxWidgets[1], para o Maxima.

O wxMaxima disponibiliza um acesso às funções do Maxima através de menus e diálogos.

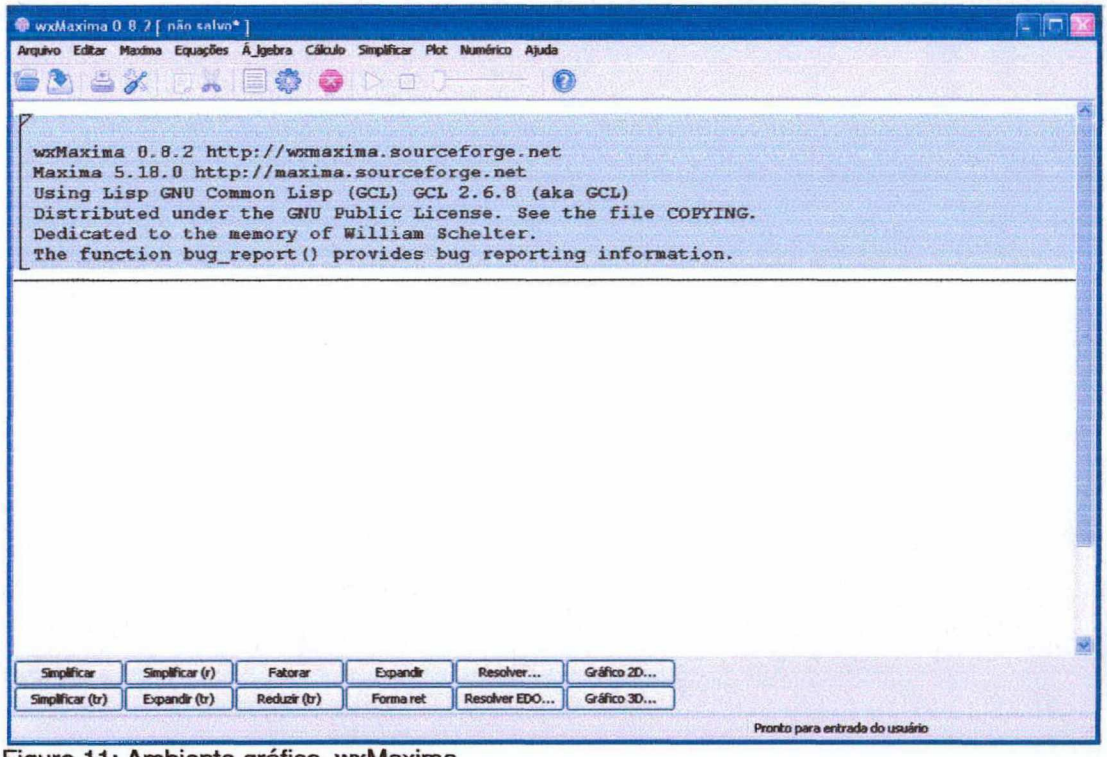

Figura 11: Ambiente gráfico wxMaxima
Fonte: http://wxmaxima.sourceforge.net

[1] Em computação, **wxWidgets** (conhecido antigamente como **wxWindows**) é um utilitário para a criação de *widgets* multi-plataforma e com código livre. É uma biblioteca com elementos básicos para a construção de interfaces gráficas com o usuário, conexão a bancos de dados ODBC e conectividade por sockets.

Veja exemplos com o uso de algumas dessas funções.

```
(%i1) 15!;
(%o1)                        1307674368000
(%i2) 18^23;
(%o2)                        743477136140219279133187768 32
(%i3) x^3+x^2-x-1;
                               3    2
(%o3)                         x  + x  - x - 1
(%i4) factor(%);
                                                          2
(%o4)                                      (x - 1) (x + 1)
(%i5) expand((x-1)^2*(x+3)^3);

                      5      4       3       2
(%o5)                 x  + 7 x  + 10 x  - 18 x  - 27 x + 27
(%i6) cos(%pi);
(%o6)                - 1
(%i7) sqrt(2);
(%o7)               sqrt(2)
(%i8) float(sqrt(2));
(%o8)                1.414213562373095
(%i9) exp(2);
                          2
(%o9)                    %e
(%i10) log(20);
(%o10)                   log (20)
(%i11) float(log(20));
(%o11)                   2.995732273553991
(%i12) asin(1);
                          %pi
(%o12)                    ---
                           2
(%i13) solve (4 * x ^ 2 + 3 * x + 2 = 0)
```

```
                  sqrt(23) %i + 3          sqrt(23) %i - 3
(%o13) [x = - ------------------, x = ------------------]
                        8                        8
(%i14) solve(x + y - 1 = 0,x);
(%o14)                        [x = 1 - y]
(%i15) solve([3*x + 2*y - z=4,4*x - y + z=4,x + y + z=3]);
(%o15)                          [[z = 1, y = 1, x = 1]]
```

A seguir, serão abordados tópicos mais específicos de Cálculo no MAXIMA.

3.4 FUNÇÕES

As funções no Maxima são definidas de forma bem simples. Usa-se o símbolo ': =' e as variáveis são colocadas entre vírgulas, como mostram os exemplos:

```
(%i1) f (x): = x + 3;
(%o1)           f (x): = x+3
(%i2) f (6);
(%o2)           9
(%i3) g (x, y): = x * y + 2
(%o3) g (x, y): = x y + 2
(%i4) g (1, 3);
(%o4)           5
```

3.5 LIMITES

O cálculo de Limites pode ser efetuado facilmente pelo MAXIMA com o seguinte comando:

limit (função, variável, valor que tende a variável);

Veja os exemplos:

```
(%i1) limit ( (1 - cos (x) ) / (x ^ 2) ,  x, 0 ) ;

                      1
(%o1)                 -
                      2

(%i2) limit ( (1 + 1 / n) ^ n ,  n, inf  ) ;
(%o2)            %e
```

Limites trigonométricos:

```
(%i3) limit (sin (x) / x, x, 0) ;
(%o3)           1
(%i4) limit ((sin(tan(x)) - tan (sin(x))) / x ^ 7, x, 0 ) ;

                        1
(%o4)                 - --
                        30
```

Limites laterais:

Respeitando a sintaxe anterior, adiciona-se um quarto parâmetro 'minus' para o limite lateral pela esquerda:

```
(%i5) limit (sqrt (x * (4-x)), x, 4, minus);
(%o5)                         0
(%i6) limit (2 / x, x, 0,  minus) ;
(%o6)                       minf
```

Observação: minf significa menos infinito.

Utiliza-se o parâmetro 'plus' para os limites a direita:

```
(%i7) limit (sqrt ((5*x) / (x-5)), x, 5, plus);
```

```
(%o7)                          inf
(%i8) limit (2 / x, x, 0, plus);
(%o8)                          inf
```

Obs.: inf significa infinito.

3.6 DERIVADAS

Para obter a derivada de uma função, usa-se o comando 'diff':

diff (função, variável) ;

Observe as aplicações:

```
(%i1) diff ((3 * x ^ 2 + 4 * x),  x) ;
(%o1)                      6 x + 4
(%i2) diff (sin (x), x, 2);
(%o2)                    - sin (x)
```

Obs.: O terceiro argumento na sintaxe representa a ordem da derivação.

3.7 INTEGRAIS

Para calcular a integral de uma função, usam-se os seguintes comandos:

integrate (função, variável) ; Para integrais indefinidas integrate (função, variável, início, fim) ; Para integrais definidas

Aplicações:

- Integrais indefinidas

```
(%i1) integrate ( x ^ 2 * ( (1 + x ^  3)^ 4 ),  x ) ;

                          3     5
                       ( x  + 1)
(%o1)                  ----------
                           15

(%i2) integrate ( x ^ 3 * sin (5 * x ^ 4),  x) ;

                                 4
                          cos(5 x  )
(%o2)                   - ----------
                             20
```

-Integrais definidas

```
(%i3) integrate (x * sin(x), x, 0, 1 ) ;

(%o3)          sin(1) - cos(1)

(%i4) integrate (7 - x ^ 2 - x, x, -4, 3 ) ;

                        133
(%o4)                   -----
                          6
```

Observe um exemplo da integral de uma função descontínua:

```
(%i5) integrate (2 / x, x, -1, 0);

defint: integral is divergent.
 -- an error.  To debug this try debugmode (true);
```

Neste último exemplo, a verificação da condição de continuidade da função deve ser feita utilizando-se os recursos de limites.

3.8 Gráficos

A visualização gráfica é um dos mais importantes recursos computacionais do MAXIMA. Nesta seção descrevem-se os principais deste aplicativo.

Neste tutorial, utiliza-se o programa anexo ao MAXIMA chamado *Gnuplot Graph* que geram gráficos bi e tri dimensionais.

3.8.1 Gráficos em duas dimensões

Para traçar gráficos em duas dimensões usa-se o seguinte comando 'plot2d':

```
plot2d(função, [eixo,início,final] ) ;
```

Exemplos:

```
(%i1)  plot2d ( cos(x), [x, 0, 2 * %pi] ) ;
(%o1)
```

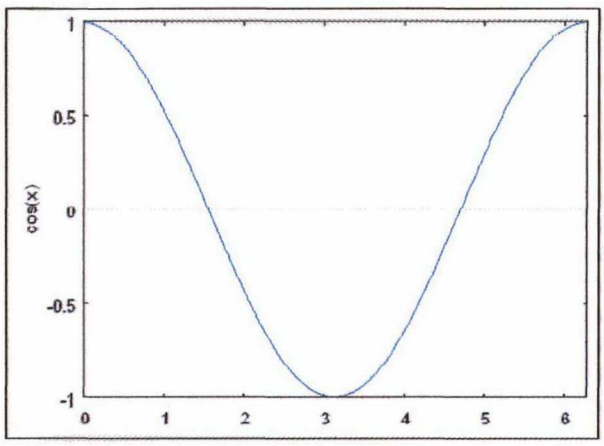

Figura 12: Gráfico da função cos(x)
Fonte: http://maxima.sourceforge.net/

A Interface gráfica do MAXIMA é uma segunda janela que aparecerá logo após o comando gráfico ser executado.

O usuário pode alterar a visualização dos gráficos conforme desejar, por exemplo:

- Como desenhar os eixos;

- Escolher a escala para desenhar as funções;
- Desativando o sistema de manuseio por mouse, pressionando 'm', e posteriormente dando um clique com o botão direito em qualquer parte do gráfico, surgirá um menu onde o usuário pode escolher diversas opções, como cores de linhas, tipo de fonte, copiar a imagem do gráfico, etc;
- Para realizar maiores operações com os gráficos, basta pressionar 'Espaço', onde surgirá a janela principal do *Gnuplot*, com todas as opções disponíveis.

Para desenhar várias funções no mesmo gráfico, colocam-se as funções da seguinte forma:

```
(%i2) plot2d ([sin (x), cos(x)], [x, -2 * %pi, 2 * %pi] ) ;

(%o2)
```

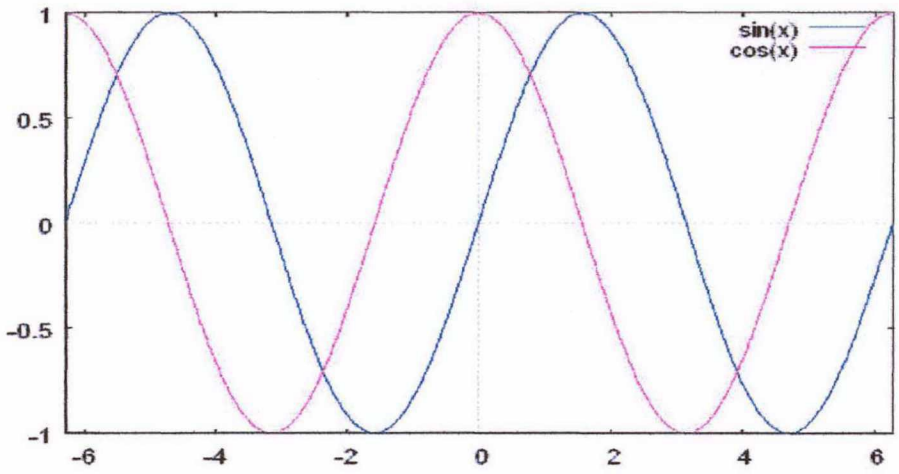

Figura 13: Gráficos das funções sin(x) e cos(x).
Fonte: http://maxima.sourceforge.net/

```
(%i3) plot2d ([sin (2*x), x ^ 2], [x, -%pi, %pi]);
(%o3)
```

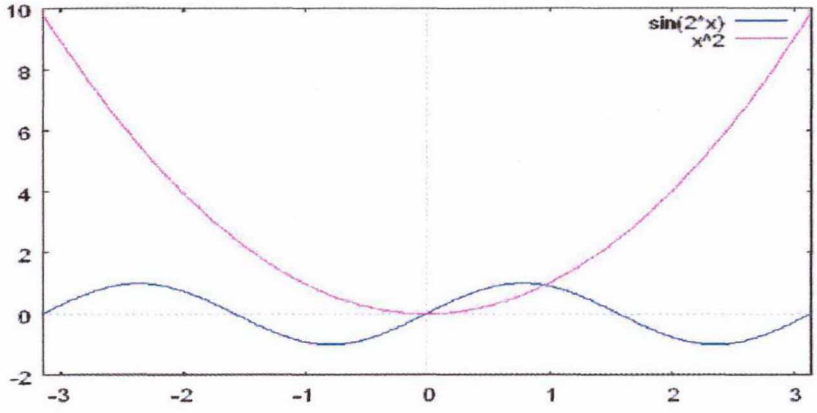

Figura 14: Gráfico das funções sin (2x) e x^2.
Fonte: http://maxima.sourceforge.net/

3.8.2 Gráficos em três dimensões

Para desenhar gráficos em três dimensões utiliza-se a função 'plot3d'. As opções desse comando são semelhantes as do comando 'plot2d'.

O *Gnuplot* gera gráficos tridimensionais permitindo que o usuário possa manuseá-los, ou seja, girá-los, conforme desejar, a partir de um clique em cima do gráfico. Além disso, o usuário pode também remanejar a escala dos gráficos como desejar, clicando no botão 3 do mouse, isto é, o do meio.

Exemplos:

```
(%i1) plot3d (x ^ 2 + y ^ 2, [x, -2, 2], [y, -2, 2]);
(%o1)
```

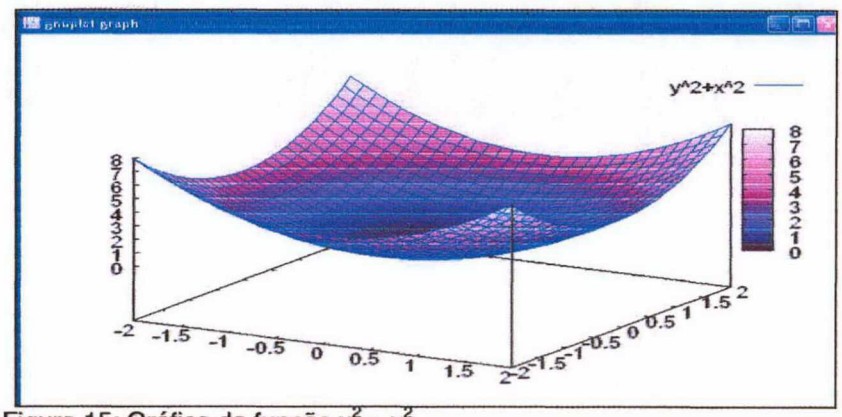

Figura 15: Gráfico da função $x^2 + y^2$.
Fonte: http://maxima.sourceforge.net/

```
(%i2) plot3d(sin(x) * sin(y),[x, 0, 2 * %pi],[y, 0,2 * %pi]);
(%o2)
```

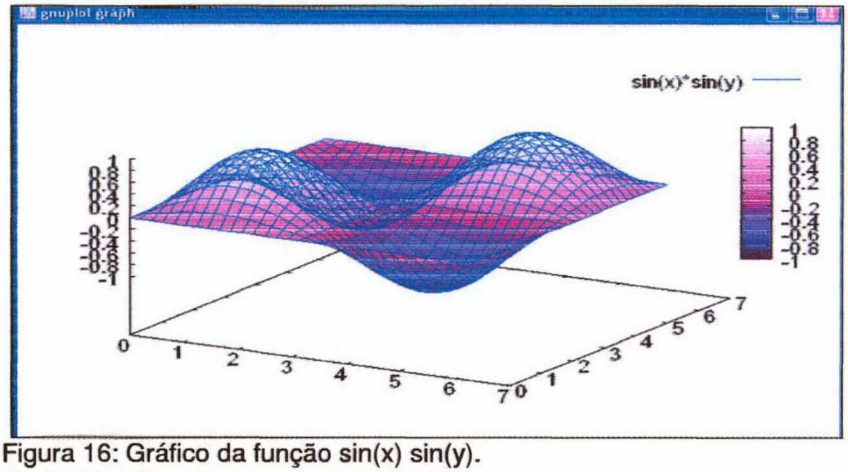

Figura 16: Gráfico da função sin(x) sin(y).
Fonte: http://maxima.sourceforge.net/

CAPÍTULO 4 - APLICAÇÕES DE EQUAÇÕES DIFERENCIAIS NO MAXIMA

Baseada em VILLATE (2007), RIOTORTO (2008), esta seção apresenta as funções disponíveis no MAXIMA para obter a solução analítica de algumas equações diferenciais de primeira e segunda ordem, bem como representações gráficas no espaço de fase, através do pacote adicional plotdf. Além disso, descreve equações em diferenças com uma abordagem prática de sistemas dinâmicos.

4.1 Equações Diferenciais

Equação Diferencial é uma equação que envolve uma função incógnita e suas derivadas, além de variáveis independentes. As Equações Diferenciais tem muitas aplicações práticas nas áreas de Medicina, Física, Engenharia, Química, Biologia, entre outras. Estas equações são úteis, por exemplo, na projeção de aviões, automóveis, circuitos elétricos e etc.

São exemplos de equações diferenciais a segunda Lei de Newton: $\vec{F} = m\vec{a} \Rightarrow \vec{F}(\vec{v},t) = \frac{md^2\vec{v}}{dt^2}$ e a equação de Lotka – Volterra :

$$\begin{aligned} \frac{dx}{dt} &= x(\alpha - \beta y) \\ \frac{dy}{dt} &= -y(\gamma - \delta x) \end{aligned}$$

, entre outras.

4.2 Definições para Equações Diferenciais

O MAXIMA pode resolver analiticamente algumas equações diferenciais ordinárias[1] (EDO) de primeira e segunda ordem utilizando o comando 'ode2'.

Uma EDO de primeira ordem possui a forma geral $F(x,y,,y) = 0$, onde $y' = \frac{dy}{dx}$.

Para resolver uma EDO de primeira ordem é conveniente definir primeiro a equação, antes de se usar o comando 'ode2', por exemplo:

Consideremos a equação:

[1] Equações Diferenciais Ordinárias são equações que contém apenas funções de uma variável e derivadas daquela mesma variável.

$$\frac{dy}{dx} = \frac{9x^2 + y - 1}{4y - x}.$$

Assim, temos:

```
(%i1) eq1: 'diff (y, x) = (9 * x ^ 2 + y -1) / (4 * y - x);
```

$$(\%o1) \qquad \frac{dy}{dx} = \frac{y + 9\,x^2 - 1}{4\,y - x}$$

O uso do apóstrofo (') indica a derivada, sem que esta seja calculada. Para obter a solução geral da equação, usa-se a função 'ode2' e três argumentos: a equação diferencial (eqn), a variável dependente (dvar), e a variável independente (ivar), como mostra a seguir:

Função: ode2(eqn, dvar, ivar)

Então,

```
(%i2) ode2 (eq1, y, x);
```

$$(\%o2) \qquad 2\,y^2 - x\,y = 3\,x^3 + x = \%c$$

onde %c representa uma constante arbitrária de integração, que se ajustará de acordo com a condição inicial que é imposta a equação. Após obter a solução, a variável método denota o método de solução usado, que neste caso, foi o método para equações exactas.

Caso a função 'ode2' não consiga encontrar a solução, aparecerá uma mensagem de erro e retornará 'false'.

Para resolver problemas de valores iniciais (PVI) estão disponíveis no MAXIMA as funções 'ic1' e 'ic2', para equações de primeira e segunda ordem, respectivamente. Na resolução de problemas de valores fronteira (PVF) para equação diferencial de segunda ordem, usa-se a função 'bc2', conforme:

Função: ic1(solução, xval, yval)
Função: ic2(solução, xval, yval, dval)
Função: bc2 (solução, xval1, yval1, xval2, yval2)

Onde:

- solução é uma solução geral para a equação, na forma dada por ode2;
- xval dá o valor inicial para a variável independente, na forma x = x0;
- yval dá o valor inicial para a variável dependente, na forma y = y0;
- xval1 e xval2 definem o valor da variável independente , num primeiro e segundo pontos, respectivamente;
- yval1 e yval2 definem o valor da variável dependente , num primeiro e segundo pontos, respectivamente;
- dval dá o valor inicial para a primeira derivada da variável dependente , em função da variável independente, na forma diff(y,x)= dy0(diff , sem o uso do apóstrofo).

Veja outro exemplo:

Seja a equação de variáveis separadas:

$$(x-1)y^3+(y-1)x^3\left(\frac{dy}{dx}\right)=0.$$

Resolvendo temos,

```
(%i3) eq2 :( x - 1) * y ^ 3 +(y - 1)* x ^ 3 *'diff(y,x)= 0;
```

$$(\%o3)\qquad x^3\,(y-1)\,\frac{dy}{dx}+(x-1)\,y^3=0$$

```
(%i4) ode2(eq2, y, x);
```

$$(\%o4)\qquad \frac{2y-1}{2y^2}=\%c-\frac{2x-1}{2x^2}$$

Nesta equação, com as condições iniciais x = 2 e y = -3, temos:

```
(%i5) ic1(%o4, x=2, y=-3);
```

(%o5) $\dfrac{2y-1}{2y^2} = -\dfrac{x^2+72x-36}{72x^2}$

Uma EDO de segunda ordem possui a forma geral *F*(*x*, *y*, *y'*,*y''*) = 0, sendo *y''* a segunda derivada de *y* em relação a *x*.

Observe o exemplo de equação diferencial ordinária de segunda ordem utilizando as funções na documentação de 'edo2':

Considere a equação:

$$\frac{dy}{dx} = \frac{d^2y}{dx^2} + x.$$

Resolução:

```
(%i6) 'diff (y, x ) = x + 'diff (y, x, 2 ) ;
```

(%o6) $\dfrac{dy}{dx} = \dfrac{d^2y}{dx^2} + x$

```
(%i7) ode2 ( %, y ,x ) ;
```

(%o7) $y = \%k_1 \%e^x + \dfrac{x^2+2x+2}{2} + \%k_2$

Maxima retorna um resultado que depende de dois parâmetros, % k_1 e % k_2, que precisam ser ajustados para fornecer algumas condições iniciais. Sabendo que para x=1, então y = -1 e y'=$\frac{dy}{dx}\Big|_{x=1} = 2$, usa -se o seguinte comando:

```
(%i8) ic2 (%, x = 1, y = - 1, diff (y, x ) = 2 ) ;
```

(%o8) $y = \dfrac{x^2+2x+2}{2} - \dfrac{7}{2}$

Assim, a solução obtida em (%o2) deve passar pelos pontos (-1,3) e (2, $\frac{5}{3}$) . Então,

```
(%i9) bc2 (%o7, x = -1, y = 3, x = 2, y = 5/3);
```

(%o9) $$y=-\frac{35\%\, e^{x+1}}{6\%\, e^3-6}+\frac{x^2+2x+2}{2}+\frac{15\%\, e^3+20}{6\%\, e^3-6}$$

O MAXIMA dispõe da função 'desolve' para resolver sistemas de equações diferenciais ordinárias usando transformada de Laplace. Este comando é descrito da seguinte forma:

Função: desolve (eqn, x)
Função: desolve ([eqn_1,..., eqn_n], [x_1,..., x_n])

As expressões eqn são equações diferenciais nas variáveis dependentes x_1,..., x_n.

Exemplo:

Considere o sistema de equações diferenciais:

$$\begin{cases} \frac{df(x)}{dx} = f(x)+g(x)+3h(x) \\ \frac{dg(x)}{dx} = g(x)-2h(x) \\ \frac{dh(x)}{dx} = f(x)+h(x) \end{cases}$$

sob as condições $f(0)=-1$, $g(0)=3$ e $h(0)=1$.

Resolvendo o sistema temos:

Para definir as equações utiliza-se notação funcional dentro da expressão diff. Assim, antes de fazer chamada à função 'desolve', é necessário introduzir as condições iniciais através da função 'atvalue'.

```
(%i10) 'diff (f(x),x) = f(x) + g(x) + 3 * h(x);
```

```
                    d
(%o10)              --- (f(x)) = 3 h (x)  + g (x)  + f (x)
                    dx
(%i11) 'diff (g(x),x) = g(x)- 2 * h(x);

                     d
(%o11)               --- (g(x)) = g (x) - 2 h (x)
                     dx
(%i12) 'diff (h(x),x) = f(x) + h(x);

                     d
(%o12)               -- (h(x))  = h (x) +  f x)
                     dx
(%i13) atvalue(f(x),x = 0,-1);
(%o13)                                  - 1
(%i14) atvalue(g(x),x = 0,3);
(%o14)                                   3
(%i15) atvalue(h(x),x = 0,1);
(%o15)                                   1
(%i16) desolve([%o6,%o7,%o8],[f(x),g(x),h(x)]);
(%o16)[f(x)=x %e^2x+%e^2x -2 %e^-x, g(x)=- 2 x %e^2x + 2 %e^2x + %e^-x,
                                        h(x) = x %e^2x + %e^-x]
```

4.2.1 Plotdf

O MAXIMA dispõe do pacote adicional plotdf para obter representações gráficas no espaço de fase. Para poder usá-lo deverá primeiro carregá-lo com o comando load("plotdf"), seguido do símbolo $(dólar). Este comando permite gerar campos de direções para uma EDO de primeira ordem, ou para um sistema de duas EDO's autônomas, de primeira ordem.

Para desenhar o campo de direções de uma única EDO de primeira ordem, ou de um sistema de duas EDO's autônomas, as equações deverão ser escritas na seguinte forma, respectivamente:

$$\frac{dy}{dx} = F(x, y)$$

A função *F* representa o argumento para o comando plotdf.

$$\begin{cases} \frac{dx}{dt} = G(x, y) \text{ e } \frac{dy}{dt} = F(x, y) \end{cases}$$

Neste sistema o argumento para o comando plotdf será uma lista de funções F e G. Além disso, as variáveis serão sempre *x*(independente) e y(dependente).

A função plotdf admite várias opções gráficas, onde cada uma delas corresponde a uma lista de dois ou mais elementos. O primeiro elemento equivale ao nome da opção, e o resto representará os argumentos para essa opção.

Veja exemplos de aplicação do comando plotdf.

Exemplo1:

- Mostrar o campo de direções da equação diferencial:

$$\frac{dy}{dx} = 1 + y + y^2.$$

Resolução no Maxima:

```
(%i7) load (plotdf)$
(%i8) plotdf (1 + y + y ^ 2);
```

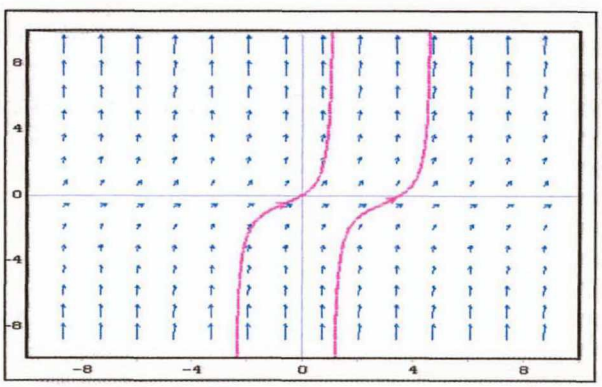

Figura 17: Campo de direções da equação $\frac{dy}{dx} = 1 + y + y^2$.

Fonte: http://maxima.sourceforge.net/

A figura 17 representa duas trajetórias a partir de dois cliques em dois pontos sobre o plano.

No campo de direções as setas representam segmentos das retas tangentes às curvas solução, e o sinal indica quando as soluções crescem ou decrescem(e quanto).

Exemplo 2:

Gerar o campo de direções da equação que representa o modelo predador – presa de Lotka – Volterra[1] que depende dos parâmetros h e k, com valores -1.2 e 0.9, respectivamente:

$$\begin{cases} \frac{dx}{dt} = 2x + hxy \text{ e } \frac{dy}{dt} = -x + kxy \end{cases}$$

Solução:

```
(%i19)plotdf([2*x+k*x*y,-y+h*x*y],
[parameters,"k=-1.2,h=0.9"],[sliders,"k=-2:2,h=-2:2"]);
```

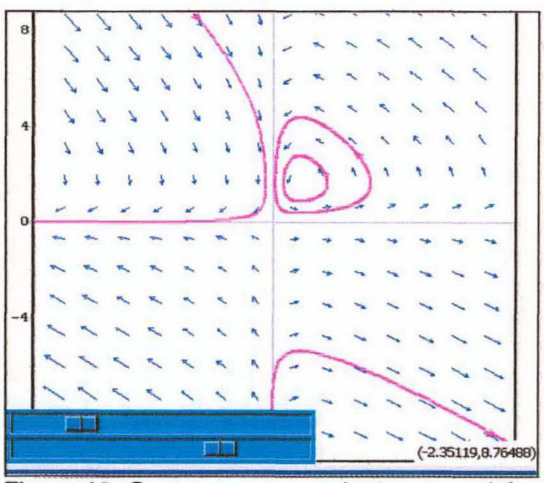

Figura 18: Campo correspondente ao modelo predador-presa.
Fonte: http://maxima.sourceforge.net/

[1] As equações de Lotka-Volterra são um par equações diferenciais, de primeira ordem e não lineares , frequentemente utilizadas para descrever dinâmicas nos sistemas biológicos. Foram propostas independentemente por Alfred J. Lotka em 1925 e Vito Volterra em 1926. Estas equações envolvem interação entre espécies, em especial a interação de uma presa com uma predadora.

Nessa janela de gráficos aparecem duas barras deslizantes para mudar interativamente os parâmetros h e k e mostrar as mudanças no campo, bastando clicar sobre elas.

A função plotdf admite várias opções, neste último caso, foram apresentadas as opções:

- *parameters* estabelece os nomes dos parâmetros e os seus valores, os quais devem escrever-se em formato de cadeia de caracteres como uma sequência de pares nome=valor separados por vírgulas;
- *sliders* estabelece os nomes dos parâmetros e os seus intervalos, os quais devem escrever-se em formato de cadeia de caracteres como uma sequência de pares nome=min:max separados por vírgulas. Estes parâmetros poderão ser alterados interactivamente utilizando barras com sliders, assim como os seus intervalos de variação.

4.3 Equações em Diferenças

As equações em diferenças são equações que contém diferenças. Por exemplo, uma seqüência de números, um sistema dinâmico e uma função de iteração, entre outros.

São exemplos de Equação em diferenças:

- A sequência de Fibonacci

A sequência $\{1, 1, 2, 3, 5, 8, 13, 21, \ldots\}$, gerada através da fórmula $y(k+2) = y(k+1) + y(k)$ para $k = 0, 1, 2, 3, \ldots$ e com valor incial $y(0) = y_0 = 1$.

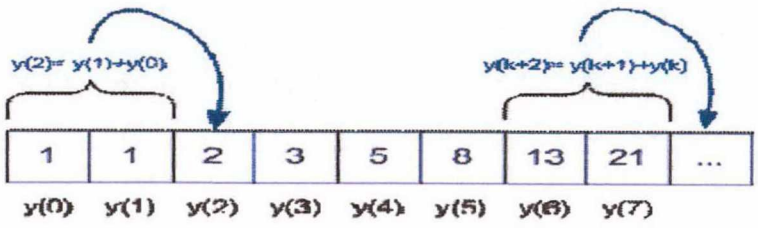

Figura 19: seqüência Fibonacci

- Um sistema dinâmico discreto

Um sistema dinâmico discreto que assume valores de sinais de entradas discretos e produz valores de sinais de saída.

O sistema dinâmico $y(k) = 2y(k-1) - 1.5u(k)$ toma valores de entrada do tipo escada $u(k) = \begin{cases} 0 & for\ k = -1,-2,-3,... \\ 1 & for\ k = 0, 1, 2, 3,... \end{cases}$ para gerar a saída $y(k) = \frac{3}{2}(1 - 2^{k+1})$

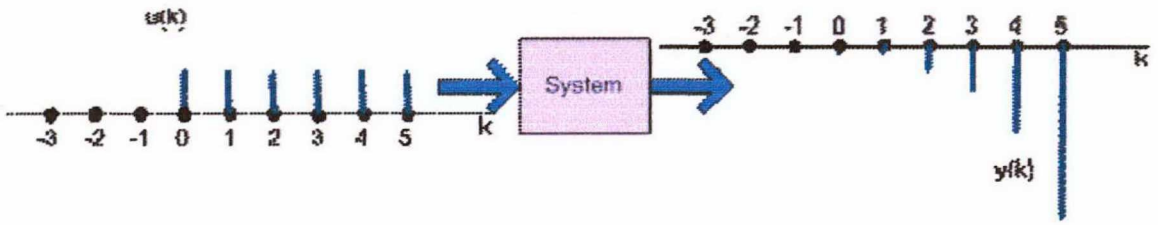

Figura 20: Sistemas dinâmicos

Sistemas dinâmicos são modelos gerais de sistemas que evoluem segundo uma regra que liga o estado presente aos estados passados. Esses modelos podem ser físicos, econômicos, biológicos, entre outros.

De acordo com VILLATE (2006), "Um sistema dinâmico discreto é um sistema em que o seu estado só muda durante os instantes {*t*0, *t*1, *t*2, . . .}. No intervalo de tempo entre dois desses instantes, o estado permanece constante".

Tratando-se de sistemas discretos em uma dimensão determinados por uma variável y, temos que:

- Os valores da variável de estado nos instantes {*t*0, *t*1, *t*2, . . .} será uma seqüência {*y*0, *y*1, *y*2,. . .};
- O intervalo de tempo entre dois instantes sucessivos t_n e t_{n+1} não tem que ser constante.

A equação que permite calcular o estado y_{n+1}, num instante n+1, a partir do estado y_n,
no instante anterior n é chamada equação de evolução:

$$y_{n+1} = F(y_n)$$

onde $F(y)$ é alguma função conhecida. A partir de um estado inicial y_0, aplicações sucessivas de função F obtêm a seqüência de estados y_n.

Aplicações de sistemas dinâmicos com o uso do MAXIMA:

- Modelo Econômico: Pede-se um empréstimo de R$ 500,00 ao banco, a uma taxa de juro anual de 5%, com prazo de 20 meses. A prestação mensal é de R$ 26,11. Qual será o montante em dívida após 10 meses?

Resolução:

A equação que representa esse sistema é:

$$y_n = y_{n-1} + j\, y_{n-1} - p,$$

onde:

n: número mês;

y_n: montante em dívida;

y_{n-1}: montante em dívida no mês anterior;

j: taxa de juro mensal;

p: prestação paga no mês.

Em MAXIMA, aplicando de forma repetida a relação de recorrência[1] temos:

```
(%i1) j: 0.05/12$

(%i2) y: 500$

(%i3) y: y + j * y - 26.11;

(%o3)                          475.9733333333333
(%i4) y: y + j * y - 26.11;

(%o4)                          451.8465555555555
```

[1] Uma relação de recorrência é uma equação em que cada termo de uma sequência é definido em função dos elementos anteriores.

```
(%i5) y: y + j * y - 26.11;

(%o5)                                427.619249537037
```

É necessário repetir o comando (%i3) dez vezes para obter a solução do problema.

Uma forma mais simples de resolver este sistema é definir uma função de argumento inteiro, a partir da relação de recorrência, e usá-la diretamente para calcular y_{10}:

```
(%i6) y[0]: 500$

(%i7) y[n]: = y[n-1]+j * y[n-1] - 26.11;

(%o7)                        y: = y        + j y      - 26.11
                                 n     n - 1       n - 1
(%i8) y[10];

(%o8)                              255.1779109580579
```

Esse último método de resolução utilizado requer cuidados no MAXIMA:

- Quando se calcula y[10], os valores de y[9], y[8],..., y[1] , foram também calculados e armazenados na memória;
- Mudando a relação de recorrência, os valores que já foram calculados não serão atualizados;
- Então, antes de modificar a relação de recorrência, ou o valor inicial y[0], é necessário apagar a seqüência já calculada, usando o comando kill.

Supondo que o valor do empréstimo fosse duplicado para R$1 000,00 e a prestação fosse também duplicada, será que o montante em dívida após o décimo mês também passaria para o dobro?

Veja:

```
(%i9) kill(y) $

(%i10) y [0]:1000$

(%i11) y[n]: = y[n-1] +j * y[n-1] - 52.22;

(%o11)                    y: = y      + j y        - 52.22
                           n     n - 1      n - 1

(%i12) y[10];

(%o12)                              510.3558219161157
```

Portanto, o montante em dívida é também duplicado.

Neste mesmo exemplo, o valor da prestação para um prazo de 40 meses seria resolvido da seguinte forma:

Sendo p a variável e o valor da dívida calculada em função de p, temos

```
(%i13) kill(y)$

(%i14) y[0]: 500$

(%o14)                              500

(%i15) y[n]: = expand (y[n-1] + j * y[n-1] - p);

(%o15)        y: = expand(y      + j y        - p)
                          n - 1      n - 1

(%i16) solve(y[40]=0,p);
```

```
                                                  72970398
(%o16)                                  [p = --------------]
                                                  5366831

(%i17) %,numer;
(%o17)                          [p = 13.59655222979818]
```

Portanto, o valor da prestação seria R$ 13,60.

No MAXIMA a função expand foi utilizada para forçá-lo a calcular os produtos na expressão para y[n]. Assim evitam-se expressões complicadas com muitos parênteses. As mensagens adicionais apresentadas indicam neste caso que alguns números de ponto flutuante foram substituídos por frações, para evitar erros numéricos.

- Queda Livre: Um pára-quedista abre o pára-quedas quando se encontra a 50m de altura, a descer com velocidade de 25 m/s. Quanto tempo demora até à aterragem? Admita uma constante aerodinâmica de 0.8, massa total 70 kg, área da secção transversal do pára-quedas 7m^2, massa volúmica do ar 1.2 kg/m3 e aceleração da gravidade 9.8 m/s^2.

Resolução:
A equação da aceleração em função da velocidade é:

$$a(v) = -9.8 - 0.048\,|v|\,v \qquad \text{(em unidades SI).}$$

Utilizando o método de Euler para calcular a altura em função do tempo, temos:

```
(%i18) numer: true$

(%i19) a(v):= -9.8 - 0.048*abs(v)*v$

(%i20) h:0.01$
```

```
(%i21) y:[50]$

(%i22) v:[-25]$

(%i23) for n thru 305 do
              (y: endcons(last(y) + h*last(v), y),
               v: endcons(last(v) + h*a(last(v)), v))$
```

Para que todos os resultados sejam calculados numericamente, a constante numer assume um valor verdadeiro (true). O número de iterações[1] foi aumentado gradualmente até se obter uma altura final negativa. Observa-se isso para os últimos elementos na lista.

```
(%i24) makelist(y[i], i, 300, 306);
(%o24) [0.74443557017859, 0.60029540282671, 0.45617250209136,
0.3120666290608, 0.16797754815704, 0.023905027088601,
- 0.12015116319614]
```

A função makelist é utilizada quando os elementos de uma lista obedecem a certo critério de construção. Neste caso, uma lista com elementos da forma y[i], de modo que i tome valores inteiros de 300 a 306.
Então, a aterragem foi entre o instante n = 304 e n = 305

$$t = 0.01 \times 304.5 = 3.045$$

a queda do pára-quedista, nos últimos 50m, demorou 3.045 s.
O gráfico resultante da altura em função do tempo pode ser feito com os comandos:

```
(%i25) t: makelist(n * h, n, 0, 305)$
(%i26) load("graph2d")$
(%i27) graph2d(t, y)$
```

[1] Iterações é o processo chamado na programação de repetição de uma ou mais ações.

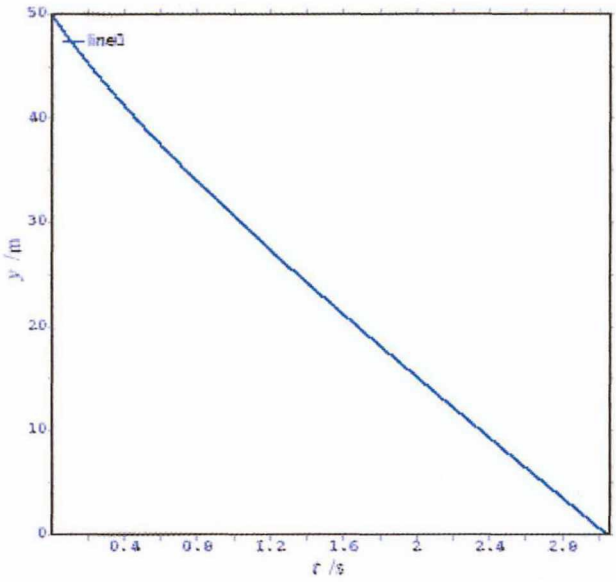

Figura 21: Altura em função do tempo, de um pára-quedista.
Fonte: http://maxima.sourceforge.net/

CAPÍTULO 5 - CONSIDERAÇÕES FINAIS

Nos últimos anos, tem se observado o desenvolvimento de diversas aplicações de sistemas de computação algébrica ao ensino de ciências, engenharia e matemática. A utilização desses sistemas no ensino e aprendizagem de Matemática e áreas afins permitem solucionar uma série de problemas e possibilitam a realização de "experiências matemáticas" que promovem a integração de aspectos geométricos e analíticos e, consequentemente, o desenvolvimento de habilidades para a construção de conceitos matemáticos.

Dentre os vários sistemas disponíveis no mercado, enfatizou-se nesse estudo, o uso dos softwares livres MuPAD,Scilab e Maxima. Priorizou-se o software Maxima por ser uma eficiente ferramenta utilizada para a solução de diversos problemas de cálculo, fácil integração a novas funções, possuir uma sintaxe simples e gerar resultados numéricos usando frações exatas e com alta precisão arbitrária.

Além disso, o Maxima possui boa documentação, representação gráfica de grande qualidade e, possibilita a criação e utilização de pacotes de funções algébricas específicas para determinados conteúdos que se deseja abordar. É também um excelente recurso aplicado na resolução de equações diferenciais, através de uma abordagem prática de sistemas dinâmicos, que representam fenômenos físicos, biológicos, econômicos, entre outros.

Este estudo permitiu sintetizar o que há de melhor entre os CAS existentes e, ao mesmo tempo, fazer uma associação entre a teoria e prática quanto a utilização destes, para um melhor aprendizado de certos conceitos matemáticos e uma aproximação mais eficaz dos resultados.

Diante disso, as expectativas relacionadas com o estudo foram, na sua grande maioria superada pelo fato de se conhecer as funcionalidades, limitações e potencialidades de cada programa da área da matemática computacional. Abrindo novos caminhos e ampliando os conceitos relacionados na área da Matemática e da linguagem da computação.

Nesse sentido, o ensino de Matemática pode acompanhar as evoluções tecnológicas, desde que professores e instituições de ensino estejam abertas a essas mudanças. E aproveitem o que há de melhor em termos de projetos e propostas que auxiliem o professor no desenvolvimento do seu trabalho facilitando o ensino e aprendizagem.

É importante ressaltar que o professor antes de utilizar um software, faça uma avaliação do mesmo, no sentido de conhecê-lo bem e aplicá-lo de acordo com suas necessidades para alcançar seus objetivos.

Portanto, o uso de softwares no processo de ensino-aprendizagem, pode contribuir de forma significativa para tornar a prática pedagógica docente mais dinâmica, bem como colaborar para ampliar o conhecimento científico de estudantes e outros usuários que dispõem destas tecnologias educacionais.

REFERÊNCIAS

ALEGRE, Ezzizis Carvalho Costa. **Tutorial MuPAD Light**. Disponível em: http://www.ime.unicamp.br/~marcio/tut2005/mupad/047085Ezzizis.pdf. Acesso em 14 jan. 2009.

ANDRADE, Lenimar Nunes de. **Computação Algébrica**. Disponível em: http://www.mat.ufpb.br/lenimar/. Acesso em: 10 de fev. 2009.

BARBOSA, Alexandre; JUNIOR, Bernardo Lula; LIMA, Aécio Ferreira de. **Simplificação automática de expressões matemáticas**. Disponível em: http://www.dsc.ufcg.edu.br/~miniblio/RelTec/2006-01.pdf. Acesso em: 06 fev. 2009.

BATISTA, Silvia; BARCELOS, Gilmara. **Projeto "TIC no Processo de Ensino e Aprendizagem de Matemática"**. Disponível em: http://www.es.cefetcampos.br/softmat/softw/mupad.html. Acesso em: 25 fev. 2009.

CARRILHO, Cap. **Introdução ao Scilab**. Disponível em: http://www.ime.eb.br/~aecc/Introducao_Scilab/IntroducaoScilab.pdf . Acesso em: 09 abr. 2009.

GUEDES, Pedro Barbosa. **Maxima. Algebra Computacional no Ensino de Matemática**. Disponível em: http://www4.dei.isep.ipp.pt/etc/workshop-oss/ISEP_CC_SLA_PBG_MAXIMA-13MAR09.pdf. Acesso em: 29 maio 2009.

MACEDO, Bruno F. Milaré de. **Tutorial Maxima 5.9.2 para Windows**. Disponível em: http://www.ime.unicamp.br/~marcio/tut2005/maxima/042290Bruno.pdf. Acesso em: 31 jan. 2009.

PEREIRA, Lucas Campos e PEREIRA, Priscila de Souza. **Software livre Scilab**. Disponível em: http://www.lenep.uenf.br/~nivaldo/disciplinas/softLivre/turma2008/apresentacoes/LucasPriscila/scilab.pdf. Acesso em: 09 abr. 2009.

PIRES, Paulo Sergio da Motta. **Introdução ao Scilab**. Disponível em: http://www.scilab.org/publications/SCIPORT/sciport.pdf. Acesso em: 14 jan. 2009.

RANGEL, Leandro; ROJAS, Alexandre. **Tutorial sobre álgebra computacional com programação**. Disponível em: http://www.ime.uerj.br/cadernos/cadinf/vol18/artigo4.pdf. Acesso em: 28 mar. 2009.

RIOTORTO, Mario Rodríguez. **Primeiros pasos en Maxima**. Disponível em:http:// www.telefonica.net/web2/biomates. Acesso em: 11 de mar. 2009.

_____. Trad. Jorge Barros de Abreu. **Primeiros passos no Maxima**. Disponível em: http://www.professores.uff.br/hjbortol/disciplinas/2006.2/esp00000/arquivos/max_pt.pdf. --- acesso em: 11 de mar. 2009.

SANT-ANA, Eliane da Costa Granadeiro, et al. **Aplicação de um sistema de computação algébrica como ferramenta educacional para apoio ao ensino do calculo diferencial e integral**. Disponível em: http://biblioteca.univap.br:88/inic/inic/01.htm. Acesso em: 14 jan. 2009.

VAZ, Cristina Lucia. **Aprendendo MuPAD**. Disponível em: http://www.projetos.unijui.edu.br/matematica/amem/mupad/MUPAD.pdf. Acesso em: 17 jan. 2009.

VILLATE, Jaime E. **Introdução aos Sistemas Dinâmicos: uma abordagem prática com Maxima**. Disponível em: http://fisica.fe.up.pt/pub/maxima/sistdinam.pdf. Acesso em: 22 fev. 2009.

_____. **Equações diferenciais e equações de diferenças**. Disponível em: http://villate.org/doc/eqdiferenciais/. Acesso em: 1 jun. 2009.