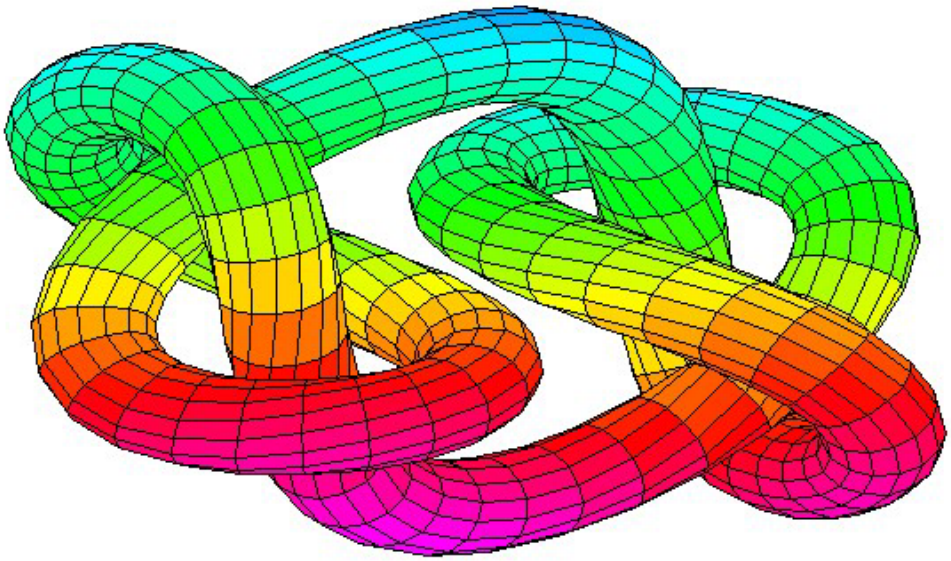
Maxima: um programa para as aulas de Matemática



Lenimar Nunes de Andrade
numerufpb@gmail.com
versão 1.5 – 14/agosto/2015

Sumário

Prefácio

Este texto corresponde às notas de aula resumidas de uma parte da disciplina *"Recursos Computacionais para o Ensino de Matemática"* do PROFMAT que vem sendo ministrada por mim na Universidade Federal da Paraíba desde fevereiro de 2014.

O objetivo geral é introduzir o *Maxima*, interessante programa de Computação Algébrica de uso geral que pode ser uma ferramenta valiosa em aulas de Matemática. Seu uso pode ser incorporado à rotina diária, podendo ser muito útil na resolução de problemas, elaboração de provas e listas de exercícios.

No capítulo 1, fazemos uma rápida apresentação do programa. O capítulo 2 é sobre fatoração, simplificação e expansão de expressões algébricas e o capítulo 3 é sobre listas e conjuntos. Esses três primeiros capítulos são os que tratam de assuntos mais básicos e que podem ser abordados no ensino médio.

O capítulo 4 é sobre a construção de gráficos. O *Maxima* repassa a tarefa de construir gráficos para outro programa chamado GnuPlot, instalado juntamente com ele. Podem ser construídos todos os tipos de gráficos, planos ou tridimensionais, dos mais simples aos mais sofisticados. Também é possível a construção de animações gráficas.

O capítulo 5 é sobre matrizes e Álgebra Linear. Uma diversidade de conceitos e operações está implementada nesse programa de modo que ele consegue realizar desde as mais simples operações como adição, multiplicação ou cálculo de determinantes e matrizes inversas, até operações mais complicadas como cálculo da forma de Jordan e funções de matrizes.

O capítulo 6 é sobre alguns conceitos do Cálculo Diferencial e Integral tais como limites, derivadas, integrais, séries e equações diferenciais ordinárias. As funções envolvidas podem ser de uma ou várias variáveis e as variáveis podem ser reais ou complexas.

O capítulo 7 é sobre noções de programação. O *Maxima* pode ser utilizado também como uma linguagem de programação na qual podem ser elaborados programas ou implementados algoritmos diversos. Esse é um importante aspecto do programa e que exige uma maior dedicação e um pouco mais de estudo por parte do usuário.

No final, diversos assuntos foram acrescentados na forma de apêndice. Esses assuntos são os seguintes: funções anônimas, propriedades de funções, opera-

dores, operadores lógicos, números complexos e inequações.

Este texto pode ser copiado e distribuído livremente. Está à disposição em mat.ufpb.br/lenimar/textos.htm . Foi elaborado usando-se exclusivamente programas livres e gratuitos que podem ser facilmente encontrados à disposição na Internet tais como *Maxima*, LaTeX e FastStone Image Viewer.

João Pessoa, 15 de julho de 2015

Lenimar Nunes de Andrade

Capítulo 1

Introdução

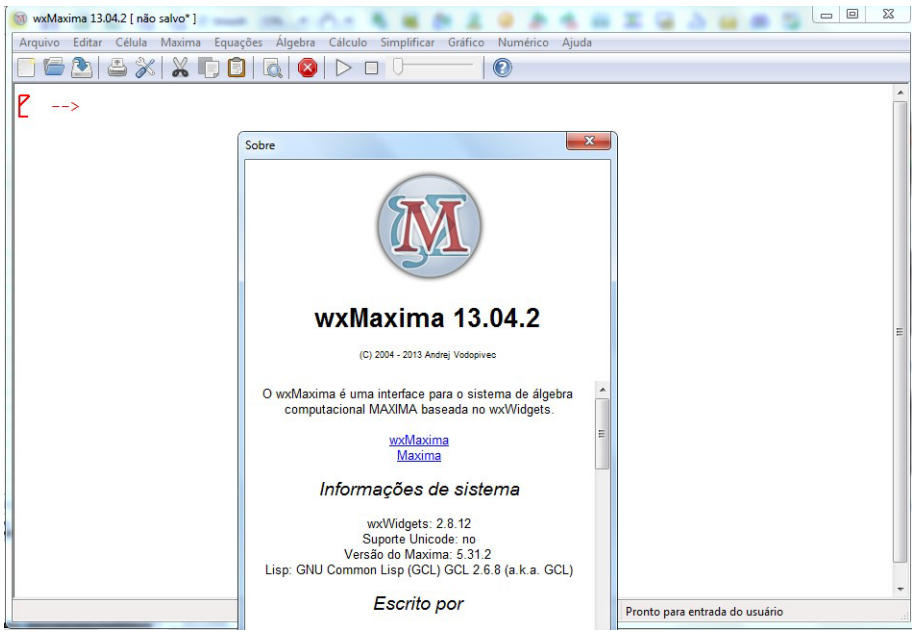
Maxima é um programa que executa cálculos numéricos e simbólicos, em desenvolvimento desde 1969. Seu nome original era Macsyma e foi elaborado nos laboratórios do MIT, nos Estados Unidos.

É capaz de simplificar expressões algébricas e trigonométricas, efetuar cálculos com matrizes e com números complexos, construir diversos tipos de gráficos, fatorar polinômios, resolver diversos tipos de equações e sistemas etc. Pode ser usado como apoio computacional para os mais diversos tipos de disciplinas, tanto as mais elementares quanto as mais avançadas.

É considerado um Sistema de Computação Algébrica de uso geral e pode ser usado em diversos sistemas operacionais como Windows e Linux.

Trata-se de um programa livre, distribuído gratuitamente e que pode ser copiado a partir do endereço **maxima.sourceforge.net**.

1.1 Interface wxMaxima



São várias as formas pelas quais o *Maxima* se comunica com o usuário. Aqui,

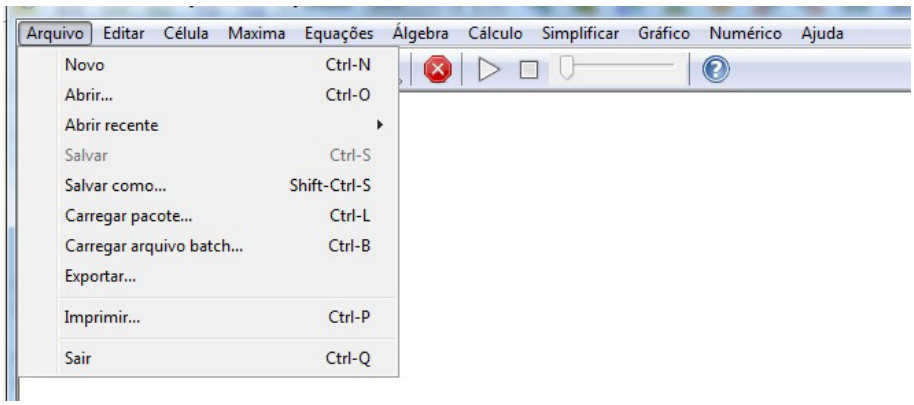
citamos apenas a interface denominada wxMaxima, que é bastante amigável, intuitiva e fácil de se usar.

O wxMaxima é instalado juntamente com o programa e sua tela inicial é uma das primeiras que aparecem quando o *Maxima* é executado. Na sua tela inicial, aparece uma linha com o seguinte menu principal:

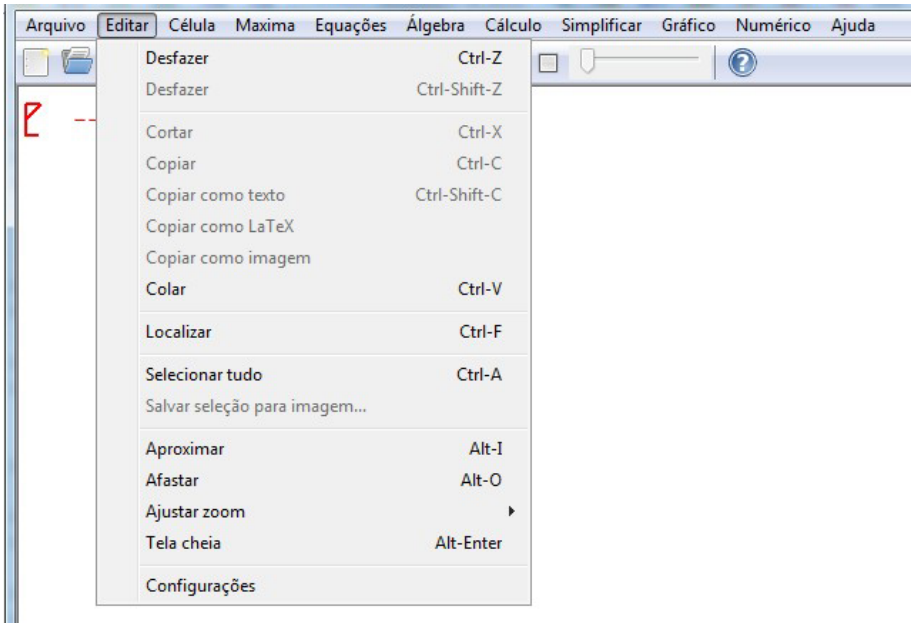
Arquivo Editar Célula Maxima Equações Álgebra Cálculo Simplificar Gráfico Numérico Ajuda

Diversas ações e comandos podem ser executados através dessas opções do menu principal. Ao se clicar com o mouse em uma dessas palavras, abrem-se outras janelas com mais opções conforme mostrado na seguinte listagem:

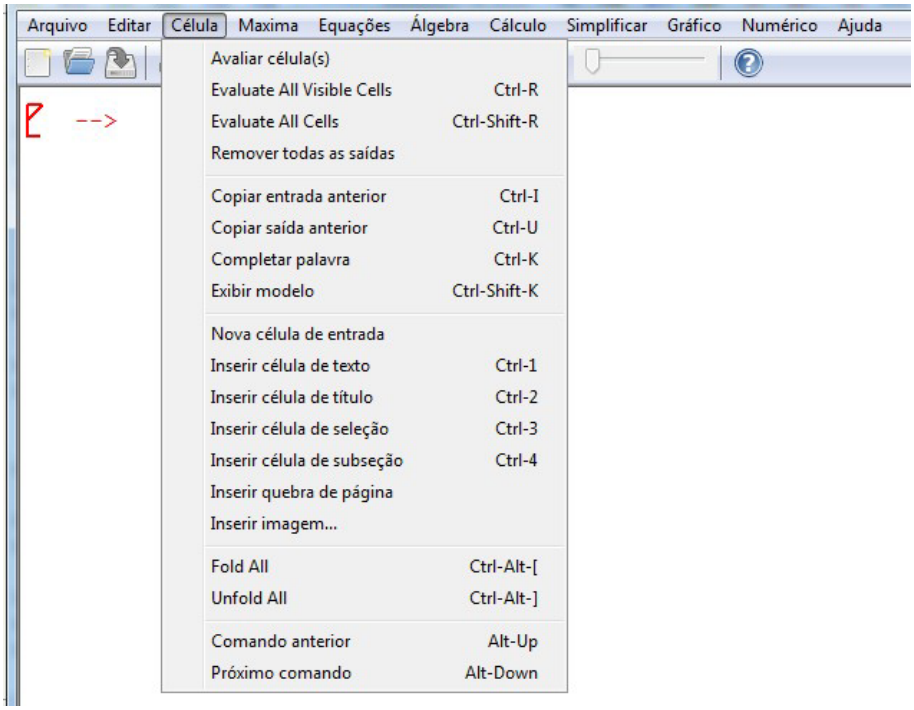
- Arquivo: Novo, Abrir, Abrir recente, Salvar, Salvar como, Carregar pacote, Carregar arquivo batch, Exportar, Imprimir, Sair;



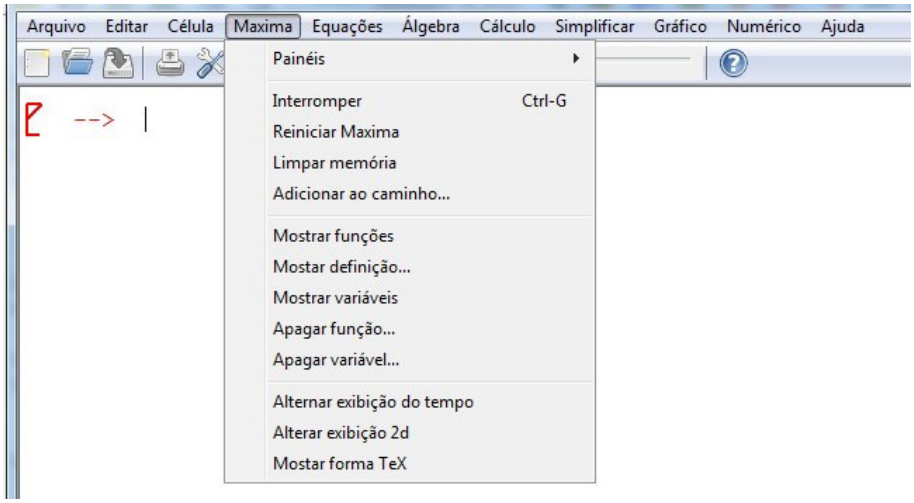
- Editar: Desfazer, Cortar, Copiar, Copiar como texto, Copiar como LaTeX, Copiar como imagem, Colar, Localizar, Selecionar tudo, Salvar seleção para imagem, Aproximar, Afastar, Ajustar zoom, Tela cheia, Configurações;



- Célula: Avaliar célula, Avaliar todas as células visíveis, Avaliar todas as células, Remover todas as saídas, Copiar entrada anterior, Copiar saída anterior, Completar palavra, Exibir modelo, Nova célula de entrada, Inserir célula de texto, Inserir célula de título, Inserir célula de seção, Inserir célula de subseção, Inserir quebra de página, Inserir imagem, Agrupar todas, Desagrupar todas, Comando anterior, Próximo comando;

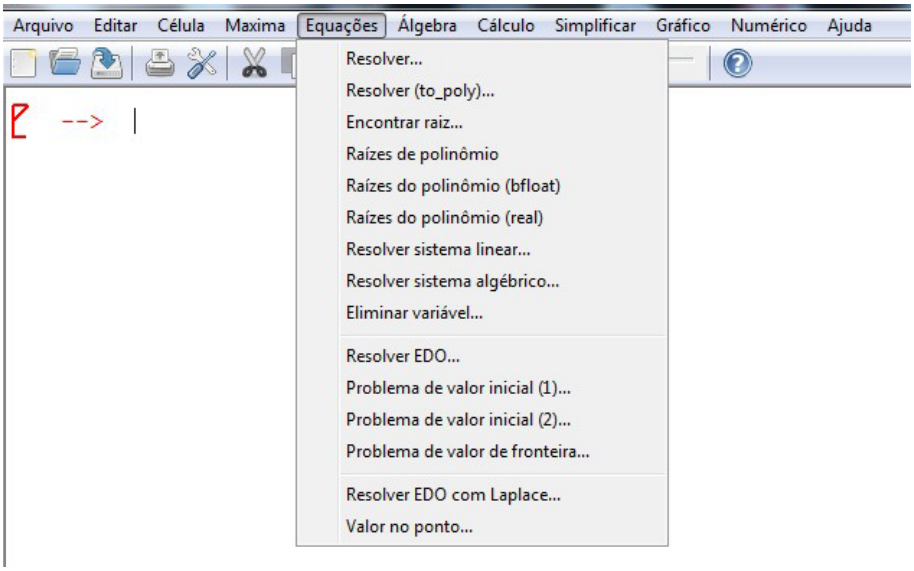


- Maxima: Painéis, Interromper, Reiniciar Maxima, Limpar memória, Adicionar ao caminho, Mostrar funções, Mostrar definição, Mostrar variáveis, Apagar função, Apagar variável, Alternar exibição do tempo, Alterar exibição 2d, Mostrar forma TeX;

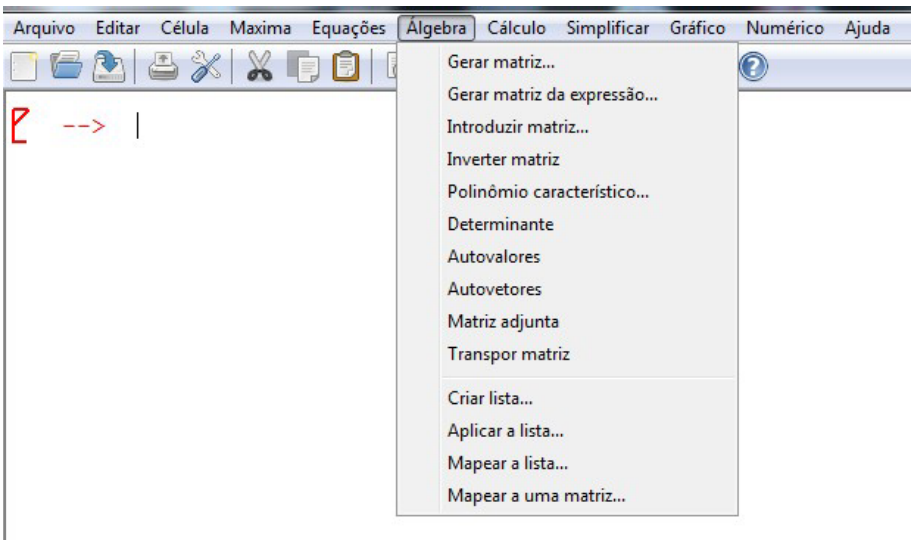


- Equações: Resolver, Resolver (to_poly), Encontrar raiz, Raízes de polinômio, Raízes de polinômio (bfloat), Raízes de polinômio (real), Resolver

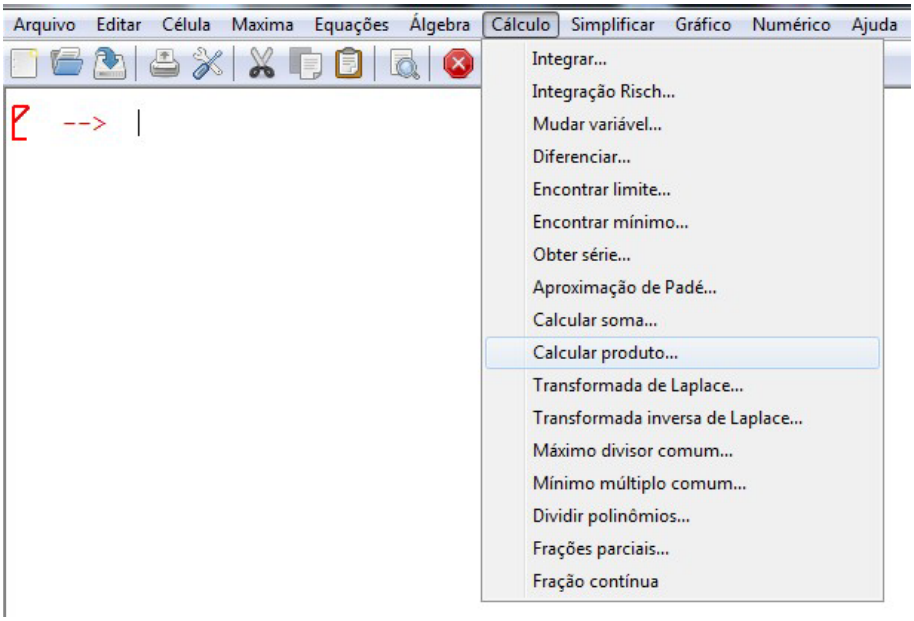
sistema linear, Resolver sistema algébrico, Eliminar variável, Resolver EDO, Problema de valor inicial (1), Problema de valor inicial (2), Problema de valor de fronteira, Resolver EDO com Laplace, Valor no ponto;



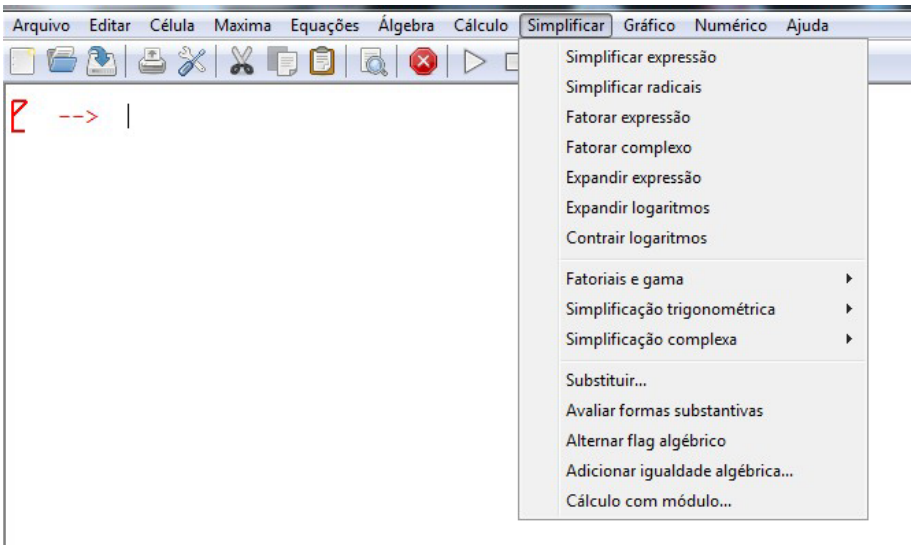
- Álgebra: Gerar matriz, Gerar matriz da expressão, Introduzir matriz, Inverter matriz, Polinômio característico, Determinante, Autovalores, Autovetores, Matriz adjunta, Transpor matriz, Criar lista, Aplicar a lista, Mapear a lista, Mapear a uma matriz;



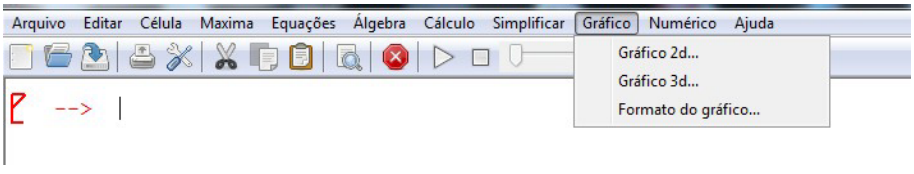
- Cálculo: Integrar, Integração Risch, Mudar variável, Diferenciar, Encontrar limite, Encontrar mínimo, Obter série, Aproximação de Padé, Calcular soma, Calcular produto, Transformada de Laplace, Transformada inversa de Laplace, Máximo divisor comum, Mínimo múltiplo comum, Dividir polinômios, Frações parciais, Fração contínua;



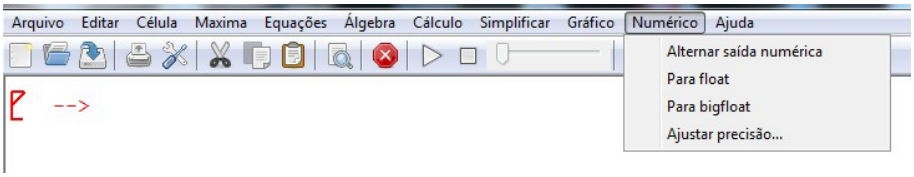
- Simplificar: Simplificar expressão, Simplificar radicais, Fatorar expressão, Fatorar complexo, Expandir expressão, Expandir logaritmos, Contrair logaritmos, Fatoriais e gama, Simplificação trigonométrica, Simplificação complexa, Substituir, Avaliar formas substantivas, Alternar flag algébrico, Adicionar igualdade algébrica, Cálculo com módulo;



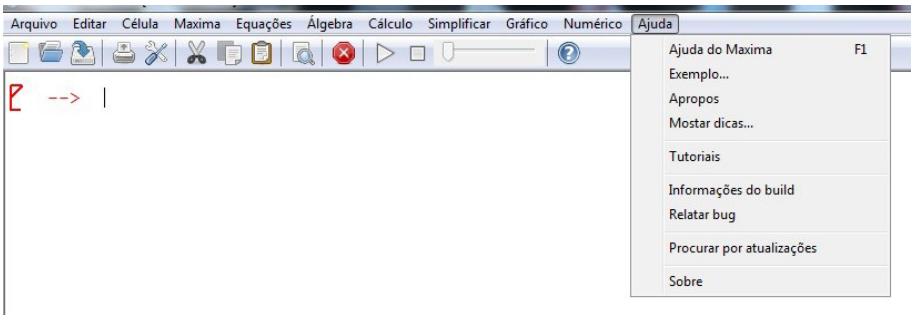
- Gráfico: Gráfico 2d, Gráfico 3d, Formato do gráfico;



- Numérico: Alternar saída numérica, Para float, Para bigfloat, Ajustar precisão;



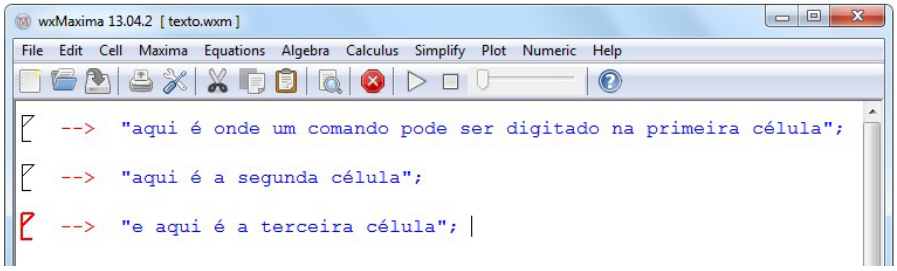
- Ajuda: Ajuda do Maxima, Exemplo, Apropos, Mostrar dicas, Tutoriais, Informações do build, Relatar bug, Procurar atualizações, Sobre.



1.2 Iniciando a digitação de comandos

Logo abaixo da linha do menu principal, vem uma barra de ferramentas que tem ícones que são atalhos para diversas ações tais como "`Novo documento`", "`Abrir documento`", "`Salvar documento`", "`Imprimir documento`", "`Configurar o wxMaxima`", entre outras.

Abaixo da barra de ferramentas, tem uma janela grande que é onde os comandos podem ser digitados ao lado de um prompt que tem o seguinte aspecto: `-->` . Diversos comandos podem ser agrupados através de um símbolo [que parece um colchete. Esse agrupamento de comandos é denominado *célula*.



A linha de comando deve se encerrar com um ponto e vírgula ou com um cifrão. Se for encerrada com um ponto e vírgula, o resultado obtido é mostrado imediatamente. Se for encerrada com um cifrão, o resultado não será mostrado de imediato, ficando guardado internamente.

No final de cada linha, depois do ponto e vírgula ou cifrão, deve-se pressionar a tecla [Enter] ou a combinação de teclas [Shift][Enter] para que o programa execute o comando digitado.

À medida que os comandos vão sendo digitados, o Maxima vai colocando uma numeração (%i1), (%i2), (%i3), ... no início de cada linha e numerando as respectivas respostas do programa com (%o1), (%o2), (%o3), ...

As operações aritméticas básicas são indicadas pelos símbolos $+$, $-$, $*$ (multiplicação), / (divisão) e ^ (potenciação). As prioridades das operações são as mesmas da Matemática. Parênteses podem ser utilizados para dar prioridade a algum cálculo.

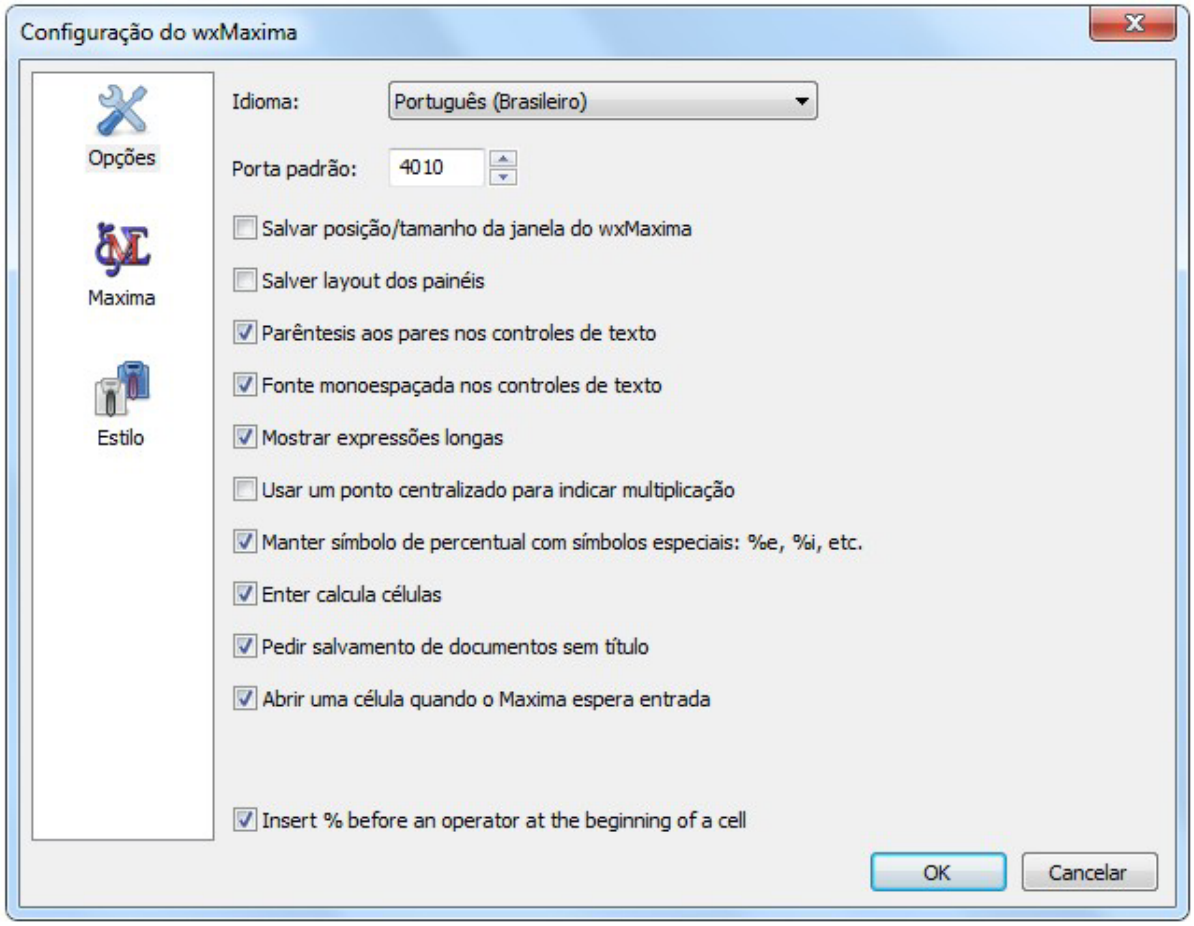
Exemplo 1.1 A seguir, alguns exemplos de comandos digitados no *Maxima*, bem como suas respectivas respostas. Calculamos $1-2^2-2^4+2^5$, $30\times50+8\times10$, $30\times(50+8)\times10$, $\frac{1}{4}+\frac{1}{18}+\frac{1}{468}$ e $\left(1+\frac{1}{30}\right)^{30}$. Em cada caso, terminamos a linha com um ponto e vírgula e pressionamos a tecla [Enter] ou a combinação de teclas [Shift][Enter].

```
(%i1)  30*50 + 8*10;
(%o1)  1580

(%i2)  30*(50 + 8)*10;
(%o2)  17400

(%i3)  1 - 2^2 - 2^4 + 2^5;
(%o3)  13

(%i4)  1/4 + 1/18 + 1/468;
(%o4)  4/13

(%i5)  (1 + 1/30)^30
(%o5)  550618520345910837374536871905139185678862401/
       205891132094649000000000000000000000000000000
```

Geralmente, o *Maxima* apresenta suas respostas da forma mais simplificada possível. Quando a resposta envolve frações, elas são mostradas sempre na forma irredutível.

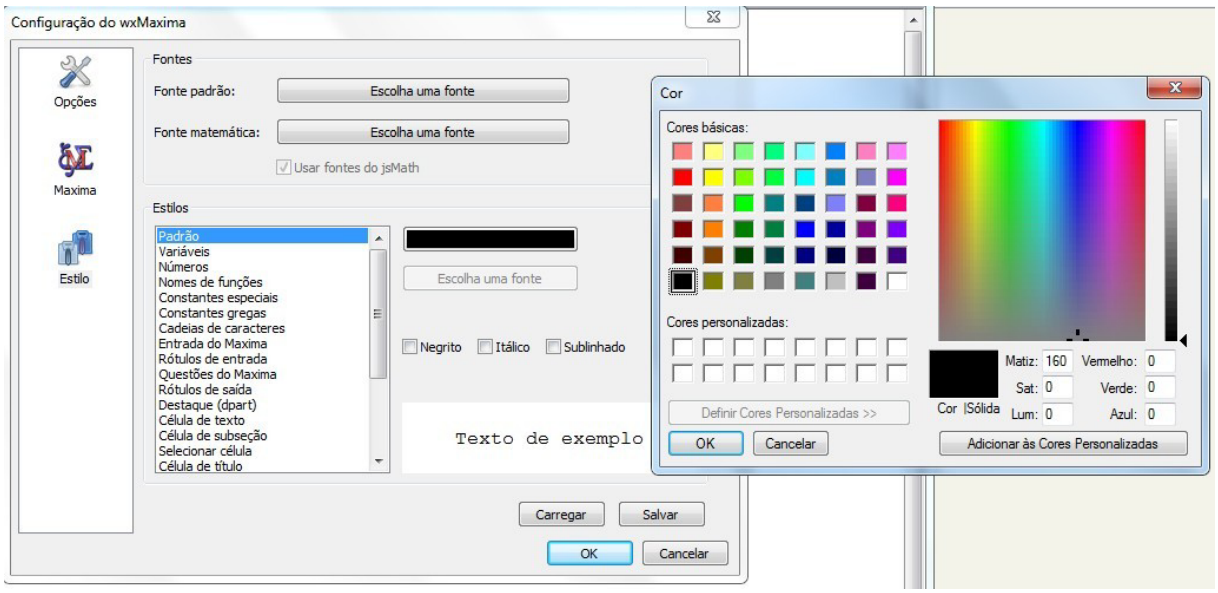
1.3 Configuração básica do programa

O wxMaxima pode ser configurado através de `Editar/Configurações` do menu principal ou pressionando-se no ícone "`Configurar o wxMaxima`" da barra de ferramentas. Em qualquer um desses casos, aparecerá a seguinte janela:



Nessa janela, pode-se configurar o idioma e diversos outros itens. Um item que é conveniente de se configurar clicando-se com o mouse em um quadradinho é o "`Enter calcula células`"

Pode-se também configurar o estilo de apresentação do programa: cores, tamanhos e tipos de letras etc. Para isso, basta usar a opção `Estilo` da janela de configuração.



1.4 Funções matemáticas básicas

sqrt(x)	Raiz quadrada de x	abs(x)	Valor absoluto de x
exp(x)	Exponencial e^x	n!	Fatorial de n
log(x)	Logaritmo natural de x	factorial(n)	Fatorial de n
sin(x)	Seno de x	cos(x)	Cosseno de x
tan(x)	Tangente de x	cot(x)	Cotangente de x
sec(x)	Secante de x	csc(x)	Cossecante de x
asin(x)	Arco-seno de x	acos(x)	Arco-cosseno de x
atan(x)	Arco-tangente de x	sinh(x)	Seno-hiperbólico de x
cosh(x)	Cosseno-hiperbólico de x	floor(x)	Maior inteiro $\leq x$
round(x)	Arredondamento de x	ceiling(x)	Menor inteiro $\geq x$
lcm(m, n)	Mínimo múltiplo comum	min(x, y)	Mínimo de $\{x, y\}$
gcd(m, n)	Máximo divisor comum	max(x, y)	Máximo de $\{x, y\}$
divide(m, n)	Quociente e resto de $m : n$	mod(m, n)	resto da divisão $m : n$

1.5 Constantes matemáticas básicas

As constantes matemáticas $\pi = 3,14159\ldots$, $e = 2,71828\ldots$, $i = \sqrt{-1}$ e $\phi = \frac{1+\sqrt{5}}{2}$ são representadas no *Maxima* por %pi, %e, %i e %phi, respectivamente.

Exemplo 1.2 A seguir, alguns exemplos de utilização de algumas funções e constantes básicas. Calculamos 30!, $\cos(\frac{\pi}{3}) + \text{sen}(\frac{\pi}{3})$, $\sqrt{11+\sqrt{25}}$ e $e^{\pi i} + i^{10}$.

```
(%i6)  30!;
(%o6)  265252859812191058636308480000000

(%i7)  cos(%pi/3) + sin(%pi/3);
(%o7)  sqrt(3)/2 + 1/2

(%i8)  sqrt(11 + sqrt(25));
(%o8)  4

(%i9)  exp(%pi * %i) + %i^10;
(%o9)  -2
```

1.6 Variáveis e atribuições

Uma variável pode ter seu nome formado por uma única letra como x, y, z, ... ou ter um nome longo onde apareçam várias letras, algarismos e caracter de sublinhado como em $expr1$, $expr2$, $result\_1$, $result\_2$, ... É feita distinção entre letras maiúsculas e minúsculas.

Podemos atribuir valor a qualquer variável digitando-se o seu nome seguido de dois pontos e do valor da variável como em $x : 2$, $y : 4$, $z : -1$ ...

Se for feita mais de uma atribuição à mesma variável, então a que importa é a última realizada.

Para se cancelar qualquer definição previamente realizada, deve-se usar um comando kill(variável) como em kill(x), kill(y) ... ou usar um kill(all) para eliminar todas as definições anteriores.

Não confunda uma atribuição como $x : 2$ com uma equação da forma $x = 2$ porque são operações completamente diferentes.

Exemplo 1.3 Neste exemplo, são definidas as variáveis x, y, z, juntamente com seus valores 5, 3, -10, respectivamente. As três atribuições podem ser feitas em uma única linha, desde que separadas por ponto e vírgula (ou cifrão). Depois disso, são calculados $\cos\frac{\pi}{x} + \text{sen}\,\frac{\pi}{y} + |z^3|$ cujo resultado é atribuído a uma nova variável chamada a, e $x + y + z$ cujo resultado é atribuído à variável w. A variável y é eliminada e w é recalculada. No final, é calculado o valor de $|aw|$.

```
(%i10)  x: 5; y: 3; z: -10;
(%o10)  5

(%o11)  3

(%o12)  −10

(%i13)  a: cos(%pi/x) + sin(%pi/y) + abs(z^3);
(%o13)  cos(π/5) + √3/2 + 1000

(%i14)  w: x + y + z;
(%o14)  −2

(%i15)  kill(y);
(%o15)  done

(%i16)  w: x + y + z;
(%o16)  y − 5

(%i17)  abs(a*w);
(%o17)  10|y − 5|
```

O último resultado calculado pode ser referenciado por um símbolo de porcentagem (%). Qualquer resultado anterior pode ser referenciado através do seu respectivo número na forma %i1, %i2, %i3, ... ou %o1, %o2, %o3, ...

Exemplo 1.4 A seguir, efetuamos os cálculos dos valores do seno, cosseno e tangente de 1,0 radiano. Depois, é calculado o valor de $\text{sen}(1.0)/\cos(1.0)$, $\text{sen}^2(1.0) + \cos^2(1.0)$ e é criada uma variável k cujo valor é $4(\text{sen}^2(1.0) + \cos^2(1.0))$.

```
(%i18)  sin(1.0);
(%o18)  0.8414709848079

(%i19)  cos(1.0);
(%o19)  0.54030230586814

(%i20)  tan(1.0);
(%o20)  1.557407724654902

(%i21)  %o18/%o19;
(%o21)  1.557407724654902

(%i22)  %o18^2 + %o19^2;
(%o22)  1.0

(%i23)  k: 4 * % ;
(%o23)  4.0
```

1.7 Criando apelidos

É possível definir um nome alternativo (apelido) para qualquer variável, função ou comando do *Maxima*. Para isso, basta digitar um comando **alias***(novo nome, velho nome)*. Podem ser definidos vários apelidos com um único comando conforme mostrado no exemplo `alias(sen, sin, tg, tan)`. Um comando **aliases** pode ser usado para mostrar a lista de todos os apelidos previamente definidos.

Exemplo 1.5 Definimos um apelido para sqrt chamado raiz e calculamos o valor de $\sqrt{21 + \sqrt{16}}$. Depois, definimos arctg e arcsen como sendo apelidos de atan e asin, respectivamente, e calculamos o valor de $\text{arctg}(1) + \text{arcsen}(\frac{\sqrt{3}}{2})$.

Definimos também mdc como sendo um apelido de gcd, mmc como apelido de lcm e calculamos $\text{mdc}(25, 10) \times \text{mmc}(25, 10)$. No final, listamos a relação de todos os apelidos previamente definidos.

```
(%i24)  alias(raiz, sqrt);
(%o24)  [raiz]

(%i25)  raiz(21 + raiz(16));
(%o25)  5

(%i26)  alias(arctg, atan, arcsen, asin);
(%o26)  [arctg, arcsen]

(%i27)  arctg(1) + arcsen(raiz(3)/2);
(%o27)  7π/12

(%i28)  alias(mdc, gcd, mmc, lcm);
(%o28)  [mdc, mmc]

(%i29)  mdc(25, 10)*mmc(25, 10);
(%o29)  250

(%i30)  aliases;
(%o30)  [raiz, arctg, arcsen, mdc, mmc]
```

Os apelidos permanecem ativos apenas durante a sessão atual do programa. Ao reiniciar, os apelidos precisam ser redefinidos.

1.8 Variáveis globais

O *Maxima* define suas ações baseado nos valores de certas variáveis que são consideradas padrões para o programa. Por exemplo, tem uma variável chamada `showtime` cujo valor padrão é `false`. Se o valor de `showtime` for trocado para `true`, então o programa passa a mostrar o tempo gasto para executar cada comando.

Exemplo 1.6 Neste exemplo, mostramos o tempo gasto na execução do comando factor(2^(2^7) + 1) que fatora o número inteiro $2^{2^7} + 1$. O programa mostra que demorou 34,43 segundos. Depois, desativamos essa exibição do tempo gasto.

```
(%i31)  showtime: true
(%o31)  true

(%i32)  factor(2^(2^7) + 1);
Evaluation took 34.4300 seconds (34.4300 elapsed)
(%o32)  59649589127497217  5704689200685129054721

(%i34)  showtime: false
(%o34)  false
```

Outra variável global interessante é a `logexpand`. Como padrão, o *Maxima* não expande logaritmo de produtos, expande apenas potências. No entanto, se for atribuído o valor `all` a essa variável, então o programa passa a usar as propriedades do logaritmo do produto, mas não usa o logaritmo do quociente. Se for atribuído o valor `super` à variável `logexpand`, então o programa passa a usar todas as propriedades conhecidas, incluindo o logaritmo do quociente.

Exemplo 1.7 Usamos o *Maxima* para desenvolver $\log(abc)$, $\log x^y$, $\log \frac{4}{3}$, $\log \frac{1}{x}$ e $\log(\frac{4}{3}\pi R^3)$. Note que a resposta do programa depende do valor que tiver sido atribuído à variável logexpand.

```
(%i35)  logexpand;
(%o35)  true

(%i36)  log(a*b*c);
(%o36)  log(a b c)

(%i37)  log(x^y);
(%o37)  log(x) y

(%i38)  logexpand: all;
(%o38)  all

(%i39)  log(a*b*c);
(%o39)  log(c) + log(b) + log(a)

(%i40)  log(4/3);
(%o41)  log(4/3)

(%i42)  logexpand: super;
(%o42)  super

(%i43)  log(4/3);
(%o43)  log(4) - log(3)
```

```
(%i44)  log(1/x);
(%o44)  - log(x)

(%i45)  log(4/3 * %pi * R^3);
(%o45)  3 log(R) + log(π) + log(4) - log(3)
```

1.9 Valor numérico de uma expressão

Usamos o comando $float(x)$ para obtermos a representação decimal de x. Se aparecer algum número com ponto decimal na definição de x, então a representação decimal é mostrada automaticamente. São utilizados 16 algarismos significativos como padrão.

Exemplo 1.8 Definimos x, y, z, w como sendo $\frac{1}{3}$, $\frac{1,0}{3}$, $\sqrt{5}$ e $\sqrt{5,0}$, respectivamente, e calculamos seus valores numéricos.

```
(%i46)  x: 1/3;
(%o46)  1/3

(%i47)  y: 1.0/3;
(%o47)  0.33333333333333

(%i48)  z: sqrt(5);
(%o48)  √5

(%i49)  w: sqrt(5.0);
(%o49)  2.23606797749979

(%i50)  float(x);
(%o50)  0.33333333333333

(%i51)  float(z);
(%o51)  2.23606797749979
```

Assim como ocorre com vários outros comandos do *Maxima*, o `float` pode ser utilizado depois do valor a ser mostrado. Por exemplo, $float(z)$; é considerado equivalente a $z, float$;

Exemplo 1.9 Definimos $x1$, $x2$ e $x3$ como sendo $\left(\frac{4}{7}\right)^{\ln 11} \cdot \left(\frac{7}{11}\right)^{\ln 4} \cdot \left(\frac{11}{4}\right)^{\ln 7}$, $\frac{355}{113}$ e $\cos(\ln(\pi + 20))$, respectivamente, calculamos seus valores numéricos e obser-

vamos que $x1$ é inteiro, $x2$ é uma aproximação do valor de π e que $x3$ é quase inteiro.

```
(%i52)  x1: (4/7)^log(11) *(7/11)^log(4) * (11/4)^log(7) ;
(%o52)  11^(log(7)-log(4)) 7^(log(4)-log(11)) 4^(log(11)-log(7))

(%i53)  x2: 355/113;
(%o53)  355/113

(%i54)  x3: cos(log(%pi + 20));
(%o54)  cos(log(%pi+20))

(%i55)  float(x1);
(%o55)  1.0

(%i56)  float(x2);
(%o56)  3.141592920353983

(%i57)  x3, float;
(%o57)  -0.99999999924368
```

1.10 Ajustando a precisão numérica

A quantidade de algarismos significativos usados nas representações decimais é controlado pela variável global chamada `fpprec` cujo valor padrão é 16. Para efetuar cálculos com grande quantidade de casas decimais, é só atribuir a `fpprec` um valor desejado e ajustar a saída dos dados para `ascii` através do menu "Maxima / Alterar exibição 2d" ou digitando um comando do tipo `set_display('ascii)`. Nesses casos, é preciso usar um comando `bfloat(...)` no lugar do `float(...)`.

No final de cada valor mostrado aparece um "b0" ou "b-1" que representam produtos por potências de dez $\times 10^0$ e $\times 10^{-1}$.

Exemplo 1.10 Neste exemplo, mostramos os valores numéricos de $\frac{1}{7}$, $\sqrt{2}$, π e e com 200 algarismos significativos.

```
(%i58)  fpprec;
(%o58)  16

(%i59)  fpprec: 200
(%o59)  200
```

```
(%i60)  set_display(ascii);
(%o60)  ascii
```

```
(%i61)  bfloat(1/7);
(%o61)  1.4285714285714285714285714285714285714285714285714285714285714285\
714285714285714285714285714285714285714285714285714285714285714285714285\
714285714285714285714285714285714285714285714285714285714285714b-1
```

```
(%i62)  bfloat(sqrt(2));
(%o62)  1.4142135623730950488016887242096980785696718753769480731766797379907324\
784621070388503875343276415727350138462309122970249248360558507372126441214970\
999358314132226659275055927557999505011527820605715b0
```

```
(%i63)  bfloat(%pi);
(%o63)  3.1415926535897932384626433832795028841971693993751058209749445923078164\
062862089986280348253421170679821480865132823066470938446095505822317253594081\
284811174502841027019385211055596446229489549303820b0
```

```
(%i64)  bfloat(%e);
(%o64)  2.7182818284590452353602874713526624977572470936999595749669676277240766\
303535475945713821785251664274274663919320030599218174135966290435729003342952\
605956307381323286279434907632338298807531952510190b0
```

```
(%i65)  fpprec: 16;
(%o65)  16
```

```
(%i66)  set_display(xml);
(%o66)  xml
```

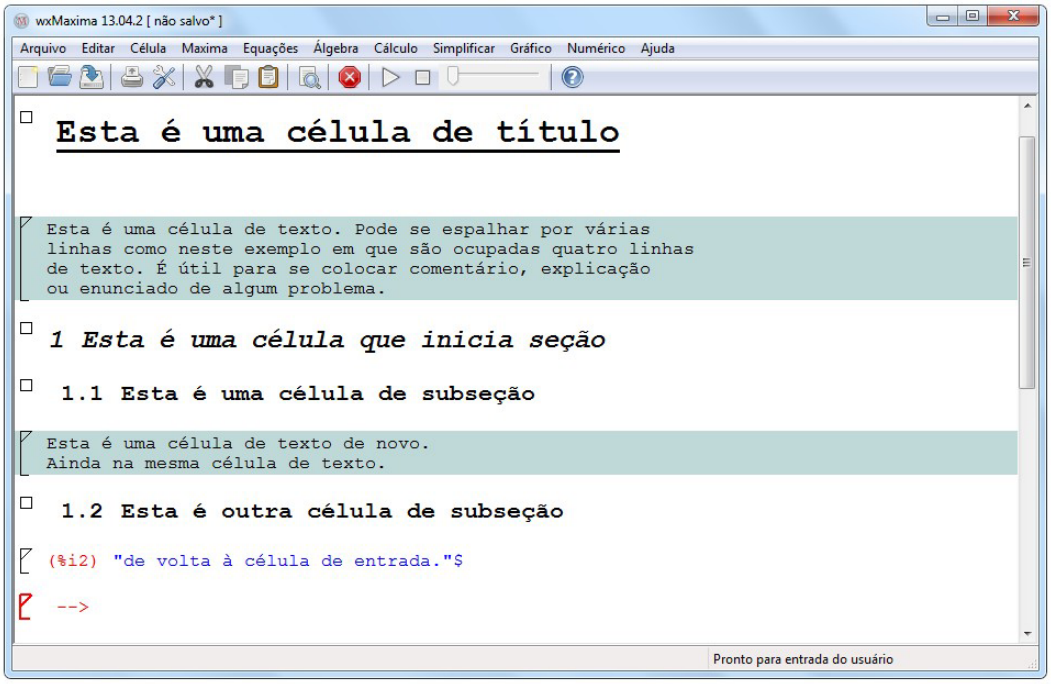
Uma barra invertida (\) colocada no final da linha em que é mostrado um valor númerico significa que a apresentação continua na linha seguinte.

1.11 Células

São possíveis vários tipos de células: entrada, texto, título, seção, subseção.

Os comandos devem ser executados em células de entrada, as células de texto são úteis para comentários e enunciados e as células de título e seção são úteis para estruturação do texto.

Para mudar o tipo de célula deve-se usar a opção "Célula / Inserir célula texto" ou "Célula / Nova célula de entrada" do menu principal.



1.12 Obtendo ajuda do programa

O *Maxima* possui uma ajuda muito interessante para o usuário. Basta digitar dois interrogações, seguido de uma parte do nome do comando, para obter uma listagem de todos comandos que são resultados daquela pesquisa. Daí, é só fornecer o número da opção selecionada, responder `all` se quiser informações sobre todos os itens ou `none` para não prosseguir com as informações.

Exemplo 1.11 Aqui, pedimos ajuda ao *Maxima* a respeito do `gcd`. A resposta dele é uma lista de comandos que têm `gcd` como parte do nome e fica esperando que o usuário faça sua opção.

```
(%i67)  ??gcd
```

```
0: ezgcd  (Funções e Variáveis Definidas para Polinômios)
1: gcd  (Funções e Variáveis Definidas para Polinômios)
2: gcdex  (Funções e Variáveis Definidas para Polinômios)
3: gcdivide  (Funções e Variáveis Definidas para simplification)
4: poly_gcd  (Funções e Variáveis Definidas para grobner)
```

Enter space-separated numbers, 'all' or 'none':

1.13 Os pacotes do Maxima

Novos comandos adicionais que vão sendo criados ao longo do tempo são agrupados em *pacotes* de acordo com seus objetivos e funções. Alguns desses pacotes são: diag, distrib, draw, dynamics, finance, fractals, functs, graphs, grobner, impdiff, lapack, linearalgebra, lsquares, mnewton, orthopoly, plotdf, stats, unit, entre outros.

Uma vez que se decida usar algum comando de determinado pacote, basta fazer uma chamada uma única vez ao pacote com um comando **load(...)** e, depois, usar o comando normalmente.

Exemplo 1.12 Ao consultar a ajuda do programa sobre o pacote **functs** com um comando `??functs`, observamos que ele possui os seguintes comandos (além de outros):

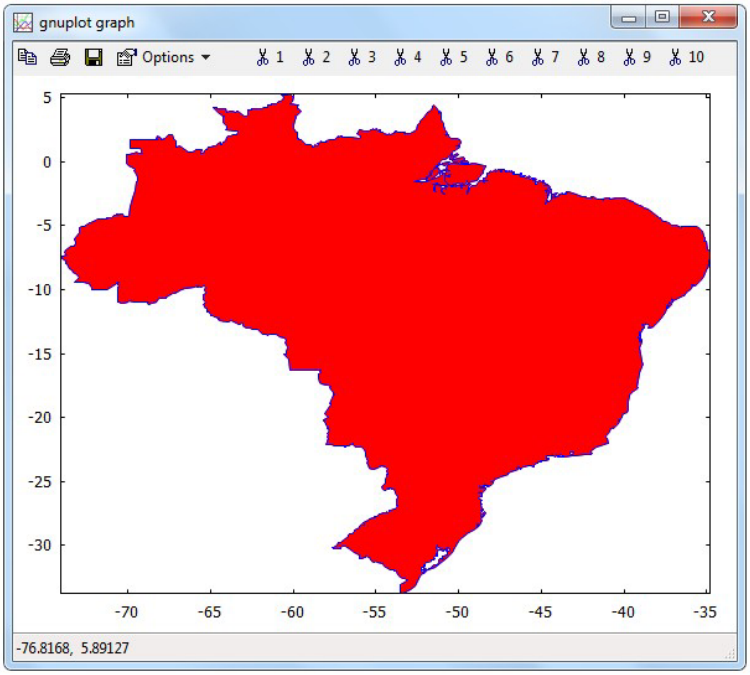
- **arithmetic(a, r, n)** enésimo termo da progressão aritmética de razão r e primeiro termo a
- **geometric(a, q, n)** enésimo termo da progressão geométrica de razão q e primeiro termo a
- **arithsum(a, r, n)** soma dos n primeiros termos da progressão aritmética de razão r e primeiro termo a
- **geomsum(a, q, n)** soma dos n primeiros termos da progressão geomética de razão q e primeiro termo a
- **combination(n, p)** número de combinações de n elementos p a p
- **permutation(n, p)** arranjos (permutações) de n elementos p a p

Exemplificamos aqui o cálculo de $C_{10,5}$, $A_{10,5}$ e a soma dos 10 primeiros termos de uma PG de razão $\frac{1}{2}$ e primeiro termo igual a 5.

```
(%i68)  load(functs)$
(%i69)  combination(10, 5);
(%o69)  252

(%i70)  permutation(10, 5);
(%o70)  30240

(%i71)  geomsum(5, 1/2, 10);
(%o71)  5115/512
```

Exemplo 1.13 O *Maxima* tem um pacote chamado worldmap que é um mapa mundi onde todos os países estão definidos na forma de polígonos do plano cartesiano. Neste exemplo, chamamos esse pacote com um comando load e, depois, desenhamos um mapa do Brasil com os comandos `draw2d` e `make_poly_country(Brazil)`.

```
(%i72)  load(worldmap)$
(%i73)  draw2d(make_poly_country(Brazil));
```



1.14 Gravando e recuperando uma sessão

Toda sessão de uso do *Maxima* pode ser gravada para ser recuperada posteriormente através da opção "Arquivo / Salvar" ou "Arquivo / Salvar como..." do menu principal.

Uma sessão que foi previamente salva pode ser carregada no programa através da opção "Arquivo / Abrir..." ou "Arquivo / Abrir recente". Nesses casos, a extensão do arquivo considerada padrão é a .wxm.

É possível também geral um arquivo .tex para ser usado pelo Latex com os comandos da sessão atual. Para isso, é só usar a opção "Arquivo / Exportar... / Arquivo pdfLaTeX" do menu principal.

Capítulo 2

Fatoração, simplificação e expansão

Neste capítulo, fazemos uma introdução a alguns dos mais importantes recursos do programa: fatoração e simplificação de expressões algébricas. O *Maxima* é muito útil não só na manipulação de polinômios e quocientes de polinômios, mas também com outros tipos de funções como logarítmicas e trigonométricas.

2.1 Fatoração

O *Maxima* é bastante eficiente na fatoração dos mais diversos tipos de expressão. Tanto na fatoração de números inteiros, quanto na de números complexos ou na de polinômios de uma ou várias variáveis, o programa tem um excelente desempenho na execução dessa difícil atividade.

O principal comando de fatoração do *Maxima* é o **factor(...)**. Pode ser usado para fatorar os mais diversos tipos de "objetos" algébricos tais como números inteiros, polinômios e frações que envolvam polinômios (funções racionais). Os resultados apresentados são sempre em termos de números primos ou polinômios irredutíveis.

Exemplo 2.1 Fatorar os seguintes números 2015, 2016, 2017 e $\frac{55}{36}$.

```
(%i1)  factor(2015);
(%o1)  5 13 31

(%i2)  factor(2016);
(%o2)  2^5 3^2 7

(%i3)  factor(2017);
(%o3)  2017

(%i4)  factor(55/36);
(%o4)  5 11 / 2^2 3^2
```

Os resultados apresentados mostram que $2015 = 5 \times 13 \times 31$, $2016 = 2^5 \times 3^2 \times 7$, 2017 é primo e que $\frac{55}{36} = \frac{5\times 11}{2^2\times 3^2}$.

Exemplo 2.2 Escrever cada um dos inteiros 40! e $2^{2^6} + 1$ como produtos de primos.

```
(%i5)  factor(40!);
(%o5)  2^38 3^18 5^9 7^5 11^3 13^3 17^2 19^2 23 29 31 37

(%i6)  factor(2^2^6 + 1);
(%o6)  274177  67280421310721
```

Os resultados acima mostram que $2^{2^6} + 1 = 274177 \times 67280421310721$ e que $40! = 2^{38} \cdot 3^{18} \cdot 5^9 \cdot 7^5 \cdot 11^3 \cdot 13^3 \cdot 17^2 \cdot 19^2 \cdot 23 \cdot 29 \cdot 31 \cdot 37$.

Exemplo 2.3 Fatorar os polinômios $f = x^4 - 81$, $g = 3x^9 - 11x^8 - 159x^5 + 583x^4 + 588x - 2156$ e $h = x^{21} + 1$.

```
(%i7)  f: x^4 - 81; g: 3*x^9 - 11*x^8 - 159*x^5 + 583*x^4 + 588*x - 2156;
h = x^21 + 1;
(%o7)  x^4 - 81

(%o8)  3*x^9 - 11*x^8 - 159*x^5 + 583*x^4 + 588*x - 2156

(%o9)  x^21 + 1

(%i10)  factor(f);
(%o10)  (x - 3)(x + 3)(x^2 + 9)

(%i11)  factor(g);
(%o11)  (3x - 11)(x^2 - 7)(x^2 - 2)(x^2 + 2)(x^2 + 7)

(%i12)  factor(h);
(%o12)  (x + 1)(x^2 - x + 1)(x^6 - x^5 + x^4 - x^3 + x^2 - x + 1)(x^12 + x^11 - x^9 -
x^8 + x^6 - x^4 - x^3 + x + 1)
```

Obtivemos assim as seguintes fatorações:

- $x^4 - 81 = (x - 3)(x + 3)(x^2 + 9)$,

- $3x^9 - 11x^8 - 159x^5 + 583x^4 + 588x - 2156 = (3x - 11)(x^2 - 7)\ (x^2 - 2)\ (x^2 + 2)(x^2 + 7)$,
- $x^{21} + 1 = (x + 1)(x^2 - x + 1)(x^6 - x^5 + x^4 - x^3 + x^2 - x + 1)(x^{12} + x^{11} - x^9 - x^8 + x^6 - x^4 - x^3 + x + 1)$.

2.1.1 Polinômios com várias variáveis e funções racionais

Os polinômios que o `factor(...)` decompõe em fatores podem ter várias variáveis. Além disso, se a expressão algébrica for um quociente de polinômios, então o numerador e o denominador são fatorados separadamente.

Exemplo 2.4 Fatorar as seguintes expressões algébricas:

- $-6yz^2 - 3xz^2 + 2y^2z - 23xyz - 12x^2z + 8xy^2 + 4x^2y$
- $\dfrac{x^2 - 5x + 6}{x^2 + 7x + 12}$
- $\dfrac{x^4 - 44x^2 - 245}{x^4 - 6x^2 - 7}$
- $\dfrac{x^4 - 2x^2y^2 + y^4}{16x^4 - 81y^4}$

```
(%i13) -6*y*z^2 - 3*x*z^2 + 2*y^2*z - 23*x*y*z - 12*x^2*z + 8*x*y^2 +
4*x^2*y;
```

(%o13) $-6yz^2 - 3xz^2 + 2y^2z - 23xyz - 12x^2z + 8xy^2 + 4x^2y$

```
(%i14) factor(%);
```

(%o14) $-(2y + x)(z + 4x)(3z - y)$

```
(%i15) (x^2 - 5*x + 6)/(x^2 + 7*x + 12);
```

(%o15) $\dfrac{x^2 - 5x + 6}{x^2 + 7x + 12}$

```
(%i16) factor(%);
```

(%o17) $\dfrac{(x - 3)(x - 2)}{(x + 3)(x + 4)}$

```
(%i18) (x^4 - 44*x^2 - 245)/(x^4 - 6*x^2 - 7);
```

(%o18) $\dfrac{x^4 - 44x^2 - 245}{x^4 - 6x^2 - 7}$

```
(%i19) factor(%);
```

(%o20) $\dfrac{(x - 7)(x + 7)(x^2 + 5)}{(x^2 - 7)(x^2 + 1)}$

```
(%i21) (x^4 - 2*x^2*y^2 + y^4)/(16*x^4 - 81*y^4);
```

$$(\%o21)\quad \frac{x^4-2x^2y^2+y^4}{16x^4-81y^4}$$

```
(%i22) factor(%);
```

$$(\%o22)\quad -\frac{(y-x)^2(y+x)^2}{(3y-2x)(3y+2x)(9y^2+4x^2)}$$

Obtivemos dessa forma as seguintes fatorações:

- $-6yz^2-3xz^2+2y^2z-23xyz-12x^2z+8xy^2+4x^2y = -(2y+x)(z+4x)(3z-y)$
- $\dfrac{x^2-5x+6}{x^2+7x+12}=\dfrac{(x-3)(x-2)}{(x+3)(x+4)}$
- $\dfrac{x^4-44x^2-245}{x^4-6x^2-7}=\dfrac{(x-7)(x+7)(x^2+5)}{(x^2-7)(x^2+1)}$
- $\dfrac{x^4-2x^2y^2+y^4}{16x^4-81y^4}=-\dfrac{(y-x)^2(y+x)^2}{(3y-2x)(3y+2x)(9y^2+4x^2)}$

2.1.2 Fatoração nos números complexos

O *Maxima* também fatora polinômios nos números complexos. Para isso, deve-se usar um comando **gfactor(...)**.

Exemplo 2.5 Fatorar $x^4+8x^3+41x^2+128x+400$ de duas maneiras: nos inteiros e nos complexos.

```
(%i23) p: x^4 + 8*x^3 + 41*x^2 + 128*x + 400;
```

$$(\%o23)\quad x^4+8x^3+41x^2+128x+400$$

```
(%i24) factor(p);
```

$$(\%o24)\quad (x^2+16)(x^2+8x+25)$$

```
(%i25) gfactor(p);
```

$$(\%o25)\quad (x-4\%i)(x-3\%i+4)(x+3\%i+4)(x+4\%i)$$

O resultado obtido é: $x^4+8x^3+41x^2+128x+400=(x^2+16)(x^2+8x+25)=(x-4i)(x-3i+4)(x+3i+4)(x+4i)$

Exemplo 2.6 Fatorar $x^{10}+1$ de duas maneiras: nos inteiros e nos complexos.

Assim como ocorre com muitos comandos, o `factor` e o `gfactor` podem ser digitados no final da linha, depois da expressão a ser fatorada, seguida de uma vírgula.

```
(%i26)  x^10 + 1, factor;
```
$$(\%o26) \quad (x^2+1)(x^8-x^6+x^4-x^2+1)$$

```
(%i27)  x^10+1, gfactor;
```
$$(\%o27) \quad (x-\%i)(x+\%i)(x^4-\%ix^3-x^2+\%ix+1)(x^4+\%ix^3-x^2-\%i*x+1)$$

As fatorações obtidas foram: $x^{10}+1 = (x^2+1)(x^8-x^6+x^4-x^2+1) = (x-i)(x+i)(x^4-ix^3-x^2+ix+1)(x^4+ix^3-x^2-ix+1)$.

Exemplo 2.7 Fatorar $x^4+(6+i)x^3+(12+31i)x^2+(72+135i)x+(193+109i)$.

```
(%i28)  poli: x^4 + (6+%i)*x^3 + (12 + 31*%i)*x^2 + (72 + 135*%i)*x
+ (193+109*%i)
```
$$(\%o28) \quad x^4+(\%i+6)x^3+(31\%i+12)x^2+(135\%i+72)x+109\%i+193$$

```
(%i29)  gfactor(poli);
```
$$(\%o29) \quad (x-2\%i+1)(x-\%i+4)^2(x+5\%i-3)$$

O resultado obtido foi: $x^4+(6+i)x^3+(12+31i)x^2+(72+135i)x+(193+109i) = (x-2i+1)(x-i+4)^2(x+5i-3)$.

2.1.3 Fatoração em $\mathbb{Q}[w]$

Normalmente, o `factor(...)` escreve um polinômio de coeficientes inteiros como produto de polinômios irredutíveis com coeficientes inteiros, sempre que isso for possível.

Podemos ampliar o conjunto que contém os coefientes dos polinômios fatorados, permitindo que os coeficientes sejam combinações potências de um número `w`. Para isso, é só usar um comando `factor(polinômio, p(w) = 0)` ou `factor(polinômio, p(w))` onde `p` é um polinômio do qual `w` é uma raiz.

Exemplo 2.8 Escrever x^2+9 como produto de polinômios com coeficientes que sejam combinações de w satisfazendo a equação $w^2+1=0$.

```
(%i30)  factor(x^2 + 9, w^2 + 1 = 0);
```
$$(\%o30) \quad (x-3w)(x+3w)$$

Dessa forma, obtivemos que $x^2+9=(x-3w)(x+3w)$ onde w é raiz de $w^2+1=0$, ou seja, $x^2+9=(x-3i)(x+3i)$.

Exemplo 2.9 Escrever $x^2-6x-19$ como produto de polinômios com coeficientes que sejam combinações de s satisfazendo a equação $s^2-7=0$.

```
(%i31)  factor(x^2 - 6*x - 19, s^2 - 7 = 0);
```
(%o31) $(x-2s-3)(x+2s-3)$

Dessa forma, obtivemos que $x^2-6x-19=(x-2s-3)(x+2s-3)$ onde s é raiz de $s^2-7=0$, ou seja, $x^2-6x-19=(x-2\sqrt{7}-3)(x+2\sqrt{7}-3)$.

Exemplo 2.10 Sendo b uma raiz da equação $x^4+1=0$, escrever $x^{12}+1$ como produto de polinômios com coeficientes em $\mathbb{Q}[b]$, ou seja, com coeficientes que sejam combinações de b ou potências de b. No final, conferir a resposta apresentada efetuando os produtos levando em conta a restrição $b^4+1=0$.

```
(%i32)  factor(1 + x^12, b^4 + 1);
```
(%o32) $(x-b)(x+b)(x-b^3)(x+b^3)(x^2-bx+b^2)(x^2+bx+b^2)(x^2-b^3x-b^2)(x^2+b^3x-b^2)$

```
(%i33)  expr: %;
```
(%o33) $(x-b)(x+b)(x-b^3)(x+b^3)(x^2-bx+b^2)(x^2+bx+b^2)(x^2-b^3x-b^2)(x^2+b^3x-b^2)$

```
(%i34)  expand(expr);
```
(%o35) $x^{12}-2b^6x^{10}-2b^2x^{10}+b^{12}x^8+2b^8x^8+b^4x^8-b^{10}x^6-b^6x^6+2b^{12}x^4+2b^8x^4-b^{18}x^2-2b^{14}x^2-b^{10}x^2+b^{16}$

```
(%i36)  scsimp(expr, b^4 + 1 = 0);
```
(%o36) $x^{12}+1$

O `expand(expressão)` expande a expressão efetuando todas as multiplicações e potências indicadas.

O `scsimp(expressão, restrição)` simplifica uma expressão levando em conta alguma restrição em suas variáveis.

Assim, os resultados obtidos foram:
$x^{12}+1=(x-b)(x+b)(x-b^3)(x+b^3)(x^2-bx+b^2)(x^2+bx+b^2)(x^2-b^3x-b^2)(x^2+b^3x-b^2)=x^{12}-2b^6x^{10}-2b^2x^{10}+b^{12}x^8+2b^8x^8+b^4x^8-b^{10}x^6-b^6x^6+2b^{12}x^4+2b^8x^4-b^{18}x^2-2b^{14}x^2-b^{10}x^2+b^{16}=x^{12}+1$.

2.1.4 Fatoração em um corpo finito

O *Maxima* realiza qualquer operação em um corpo finito $\mathbb{Z}_p$. Para isso, basta atribuir um valor inteiro p à variável global `modulus`. Se `modulus` for `false`, então é utilizada a aritmética usual em $\mathbb{Z}$.

Exemplo 2.11 Neste exemplo, fatoramos o polinômio $p = x^6 + 3x^4 + x^3 + 2$ em $\mathbb{Z}[x]$, $\mathbb{Z}_5[x]$ e em $\mathbb{Z}_{11}[x]$.

```
(%i37)  p: x^6 + 3x^4 + x^3 + 2;
(%o37)  x^6 + 3*x^4 + x^3 + 2

(%i38)  factor(p);
(%o38)  x^6 + 3*x^4 + x^3 + 2

(%i39)  modulus: 5;
(%o39)  5

(%i40)  factor(p);
(%o40)  (x + 1)^3*(x^3 + 2*x^2 - x + 2)

(%i41)  modulus: 11;
(%o41)  11

(%i42)  factor(p);
(%o42)  (x - 3)*(x^2 - 3*x + 3)*(x^3 - 5*x^2 + 5*x + 1)

(%i43)  modulus: false
(%o43)  false
```

Obtemos dessa forma que p não pode ser fatorado em $\mathbb{Z}[x]$ (ou seja, é irredutível) e que $p = (x+1)^3(x^3+2x^2-x+2)$ em $\mathbb{Z}_5[x]$. e $p = (x-3)(x^2-3x+3)(x^3-5x^2+5x+1)$ em $\mathbb{Z}_{11}[x]$.

2.2 Simplificação

O *Maxima* possui vários comandos para simplificação, entre os quais, podemos destacar os seguintes:

ratsimp(...) para expressões algébricas que envolvam frações;

radcan(...) para expressões com radicais, potências, exponenciais ou logaritmos;

scsimp(...) para expressões com restrições;

trigsimp(...) para expressões que envolvam funções trigonométricas ou hiperbólicas;

Exemplo 2.12 Simplificar as seguintes expressões:

- $\dfrac{a^3}{(a-b)(a-c))} + \dfrac{b^3}{(b-a)(b-c)} + \dfrac{c^3}{(c-a)(c-b)}$
- $\dfrac{(3x^2+4x+1)^2-(3x^2+10x+1)^2}{(3x^2+11x+1)^2-(3x^2+3x+1)^2}$
- $\dfrac{3x^2+x-4}{3x^3-8x^2-247x-308}$

Se tiver alguma operação de adição, subtração ou multiplicação no numerador ou denominador de uma fração ou no expoente de uma potência, então deve-se ter o cuidado de usar parênteses na hora da entrada do comando. Deve-se ter cuidado de não esquecer de escrever um asterisco sempre que tiver alguma operação de multiplicação envolvida. Por exemplo, uma fração como $\dfrac{a+2b}{5a-4b+1}$ deve ser introduzida na forma (a + 2*b)/(5*a - 4*b + 1).

```
(%i44)  expr1: a^3/((a-b)*(a-c)) + b^3/((b-a)*(b-c)) + c^3/((c-a)*(c-b));
```

$$(\%o44)\quad \frac{c^3}{(c-a)(c-b)}+\frac{b^3}{(b-a)(b-c)}+\frac{a^3}{(a-b)(a-c)}$$

```
; (%i45)  ratsimp(expr1);
```

$$(\%o45)\quad c+b+a$$

```
(%i46)  expr2: ((3*x^2 + 4*x + 1)^2 - (3*x^2 + 10*x + 1)^2)/((3*x^2 +
11*x + 1)^2 - (3*x^2 + 3*x + 1)^2);
```

$$(\%o46)\quad \frac{(3x^2+4x+1)^2-(3x^2+10x+1)^2}{(3x^2+11x+1)^2-(3x^2+3x+1)^2}$$

```
(%i47)  ratsimp(expr2);
```

$$(\%o47)\quad -\frac{3}{4}$$

```
(%i48)  expr3: (3*x^2 + x - 4)/(3*x^3 - 8*x^2 - 247*x - 308)
```

$$(\%o48)\quad \frac{3x^2+x-4}{3x^3-8x^2-247x-308}$$

```
(%i49)  ratsimp(expr3);
```

(%o49) $$\frac{x-1}{x^2-4x-77}$$

Dessa forma, vemos que as expressões expr1, expr2 e expr3 simplificadas são iguais a $c+b+a$, $-\frac{3}{4}$ e $\frac{x-1}{x^2-4x-77}$, respectivamente.

Expressões trigonométricas simples que envolvem apenas redução ao primeiro quadrante como $\cos(x+\pi)$, $\operatorname{sen}(x+\frac{3\pi}{2})$ são automaticamente simplificadas, assim que forem digitadas. Se a simplificação for um pouco mais complicada, envolvendo fórmulas básicas como $\cos^2 x+\operatorname{sen}^2 x=1$ ou $\cosh^2 x-\operatorname{senh}^2 x=1$, então deve-se usar um `trigsimp(...)`.

Deve-se ter o cuidado de se digitar potências como $\cos^2 x$ na forma $\cos(x)^2$ e, se for o caso, não esquecer de colocar um símbolo de porcentagem antes de pi. Para o *Maxima*, o pi sem o símbolo de porcentagem é apenas um símbolo genérico, sem valor numérico.

Exemplo 2.13 Simplificar as seguintes expressões trigonométrica e hiperbólica $\frac{\cos^5 x+\cos^5(x+\frac{\pi}{2})}{\cos x+\cos(x+\frac{\pi}{2})}$ e $1+(\cosh x+\operatorname{senh} x)^2-(\operatorname{senh} x)(\cosh x)$.

```
(%i50) cos(x + %pi/2);
```
(%o50) $-\sin(x)$

```
(%i51) cos(x + pi/2);
```
(%o51) $\cos(x+\frac{\pi}{2})$

```
(%i52) expr5: (cos(x)^5 + cos(x + %pi/2)^5)/(cos(x) + cos(x + %pi/2));
```
(%o52) $$\frac{\cos(x)^5-\sin(x)^5}{\cos(x)-\sin(x)}$$

```
(%i53) trigsimp(expr5);
```
(%o53) $\cos(x)\sin(x)+\cos(x)^4-\cos(x)^2+1$

```
(%i54) expr6: 1+ (cosh(x) + sinh(x))^2 - sinh(x)*cosh(x);
```
(%o54) $1+(\cosh(x)+\sinh(x))^2-\sinh(x)\cosh(x)$

```
(%i55) trigsimp(expr6);
```
(%o55) $\cosh(x)\sinh(x)+2\cosh(x)^2$

Obtivemos nessas simplificações os seguintes resultados:

- $\frac{\cos^5 x-\operatorname{sen}^5 x}{\cos x-\operatorname{sen} x} = \cos x\ \operatorname{sen} x+\cos^4 x-\cos^2 x+1$
- $1+(\cosh x+\operatorname{senh} x)^2-\operatorname{senh} x\cosh x = \cosh x\ \operatorname{senh} x+2\cosh^2 x$

Exemplo 2.14 Simplificar $\dfrac{x+2\sqrt{x}+1}{\sqrt{x}+1}$.

Apesar da expressão ser bem simples, devido à presença dos radicais, o `ratsimp(...)` não funciona neste caso. Deve-se usar o `radcan(...)`.

```
(%i56)  expr7: (x + 2*sqrt(x)+1)/(sqrt(x)+ 1);
```

$$(\%o56)\quad \frac{x+2\sqrt{x}+1}{\sqrt{x}+1}$$

```
(%i57)  radcan(expr7);
```

$$(\%o57)\quad \sqrt{x}+1$$

Obtivemos que $\dfrac{x+2\sqrt{x}+1}{\sqrt{x}+1} = \sqrt{x}+1$.

Exemplo 2.15 Simplificar $\dfrac{\sqrt[3]{\ln(x+x^2+x^3)-\ln(x)}}{\sqrt[8]{\ln(1+x+x^2)}}$.

Devido à presença dos radicais e dos logaritmos na expressão, devemos simplificá-la com o `radcan(...)`. O `ratsimp(...)` não funciona neste caso.

```
(%i58)  expr10: (log(x + x^2 + x^3) - log(x))^(1/3)/log(1 + x + x^2)^(1/8);
```

$$(\%o58)\quad \frac{(\log(x+x^2+x^3)-\log(x))^{1/3}}{\log(1+x+x^2)^{1/8}}$$

```
(%i59)  radcan(expr10);
```

$$(\%o59)\quad \log(x^2+x+1)^{5/24}$$

Exemplo 2.16 Simplificar a expressão

$$A = \left(\frac{x^{x^{-x}}+x^{-x^x}}{x^{-x^{-x}}+x^{x^x}}\right)^{\frac{x^x}{1-x^{2x}}}$$

O uso de parênteses em potências como $a^{(b^c)}$ é opcional. O programa considera que a^{b^c} é equivalente a $a^{(b^c)}$. Devido às várias potências envolvidas, a simplificação deve ser realizada com um `radcan(...)`.

```
(%i60)  A: (( x^x^-x + x^-x^x)/(x^-x^-x + x^x^x))^(x^x/(1-x^(2*x)));
```

$$(\%o60)\quad \left(\frac{\frac{1}{x^{x^x}}+x^{\frac{1}{x^x}}}{x^{x^x}+\frac{1}{x^{\frac{1}{x^x}}}}\right)^{\frac{x^x}{1-x^{2x}}}$$

```
(%i61)  radcan(A);
(%o61)  x
```

Note que a base da potência anterior é da forma $\frac{a+\frac{1}{b}}{\frac{1}{a}+b}$ com $a = x^{x^{-x}}$ e $b = x^{x^x}$ e essa base pode ser simplificada para $\frac{a}{b}$ que equivale a $x^{x^{-x}-x^x} = x^{\frac{1-x^{2x}}{x^x}}$. Assim, a expressão anterior A é equivalente a $\left(x^{\frac{1-x^{2x}}{x^x}}\right)^{\frac{x^x}{1-x^{2x}}} = x$.

Exemplo 2.17 Simplificar $ab^2c^2d^2 + a^3c^2d^2 + b^2cd + a^2cd - 22ab^2 - 2b^2 - 22a^3 - 2a^2$ supondo válidas as seguintes relações entre as variáveis: $a^2 + b^2 = 1$ e $cd = 5$.

Aqui, temos uma expressão a ser simplificada juntamente com duas restrições. Esse tipo de simplificação pode ser realizada com a utilização de um comando `scsimp(expressão, restrição1, restrição2, ...)`.

```
(%i62)  expr11: a*b^2*c^2*d^2+a^3*c^2*d^2+b^2*c*d+a^2*c*d-22*a*b^2-
2*b^2-22*a^3-2*a^2
(%o62)  a*b^2*c^2*d^2+a^3*c^2*d^2+b^2*c*d+a^2*c*d-22*a*b^2-2*b^2-22*a^3-2*a^2

(%i63)  r1: a^2 + b^2 = 1;
(%o63)  a^2+b^2=1

(%i64)  r2: c*d = 5;
(%o64)  c*d=5

(%i65)  scsimp(expr11, r1, r2);
(%o65)  3*a+3
```

Portanto, $ab^2c^2d^2 + a^3c^2d^2 + b^2cd + a^2cd - 22ab^2 - 2b^2 - 22a^3 - 2a^2$ com as restrições $a^2 + b^2 = 1$ e $cd = 5$ é equivalente a $3a + 3$.

2.2.1 A variável simp

Assim como muitas outras ações do *Maxima*, a simplificação é controlada por determinadas variáveis globais como a variável `simp` cujo valor pode ser `true` ou `false`. Se for `false`, então o *Maxima* não faz nenhum tipo de simplificação automática. Se simp for igual a `true` (o padrão), então algumas simplificações são realizadas automaticamente.

Exemplo 2.18 Calculamos $\sqrt{9}$ e $\frac{20}{16}$ de duas maneiras: com simp igual a `false` e, depois, igual a `true`.

```
(%i66)  simp: false;
(%o66)  false

(%i67)  sqrt(9);
(%o67)  √9

(%i68)  20/16;
(%o68)  20/16

(%i69)  simp: true;
(%o69)  true

(%i70)  sqrt(9);
(%o70)  3

(%i71)  20/16;
(%o71)  5/4
```

2.2.2 A variável radexpand

A variável global `radexpand` controla algumas simplificações de expressões com radicais. Pode assumir os valores `true`, `false` ou `all`. Seu valor padrão (*default*) é `true`.

- Se `radexpand` for `true`, então uma expressão como $\sqrt{x^2}$ é simplificada para $|x|$.
- Se `radexpand` for `all`, então $\sqrt{x^2}$ é simplificada para x. Esse caso é como se tivesse sido acrescentada uma hipótese de que $x > 0$ através de um comando `assume(x > 0)`;
- Se `radexpand` for `false`, então nenhuma simplificação é realizada.

Exemplo 2.19 Calculamos $\sqrt{(x-5)^2}$ de quatro maneiras. No último cálculo realizado, acrescentamos uma hipótese de que $x < 5$.

```
(%i72)  radexpand: false;
(%o72)  false

(%i73)  sqrt((x - 5)^2);
```

(%o73) $\sqrt{(x-5)^2}$

(%i74) radexpand: all;
(%o74) *all*

(%i75) sqrt((x - 5)^2);
(%o75) $x-5$

(%i76) radexpand: true;
(%o76) *true*

(%i77) sqrt((x - 5)^2);
(%o77) $|x-5|$

(%i78) assume(x < 5);
(%o78) $[x<5]$

(%i79) sqrt((x - 5)^2);
(%o79) $5-x$

Dependendo dos valores de `radexpand` e de x, obtemos as respostas $\sqrt{(x-5)^2}$, $x-5$, $|x-5|$ e $5-x$ para o cálculo de $\sqrt{(x-5)^2}$.

2.3 Substituição

Uma troca de x por y em uma expressão expr que contenha x pode ser efetuada com um comando **subst(x = y, expr)**. Aqui, x ou y podem ser variáveis ou expressões algébricas.

Para que a troca se torne permanente, é necessário redefinir a expressão expr como por exemplo com um comando do tipo **expr: subst(x = y, expr)** ou semelhante.

Uma troca múltipla de x por u, y por v, z por w etc. em uma expressão expr pode ser efetuada com um **sublis([x = u, y = v, z = w, . . .], expr)**. Com esse comando, $x, y, z, \ldots$ não podem ser expressões, têm que ser nomes de variáveis (símbolos).

Exemplo 2.20 Na expressão $\dfrac{x^2+y^2}{z^2+1}$, substituir x por $a+b$, y por $a-b$, z por $3a+2b+c$.

(%i80) sublis([x = a+b, y = a-b, z = 3*a+2*b+c], (x^2 + y^2)/(z^2 + 1));

(%o80) $\dfrac{(b+a)^2+(a-b)^2}{(c+2b+3a)^2+1}$

Exemplo 2.21 Efetuar as trocas de variáveis $x = r\cos\theta$ e $y = r\,\mathrm{sen}\,\theta$ na expressão $\sqrt{x^2+y^2}\cos(x^2+y^2)$. Supondo $r > 0$, simplificar a expressão obtida.

```
(%i81)  expr1: sqrt(x^2 + y^2) * cos(x^2 + y^2);
```
$$(\%o81)\quad \sqrt{y^2+x^2}\ \cos(y^2+x^2)$$
```
(%i82)  expr1: sublis([ x = r*cos(theta), y = r*sin(theta)], expr1);
```
$$(\%o82)\quad \sqrt{r^2\sin(\theta)^2+r^2\cos(\theta)^2}\ \cos(r^2\sin(\theta)^2+r^2\cos(\theta)^2)$$
```
(%i83)  assume(r > 0);
```
$$(\%o83)\quad [r > 0]$$
```
(%i84)  trigsimp(expr1);
```
$$(\%o84)\quad r\cos(r^2)$$

Obtemos $\sqrt{x^2+y^2}\cos(x^2+y^2) = r\cos r^2$, depois da substituição e simplificação.

Exemplo 2.22 Trocar $\mathrm{sen}\,x$ por $\sqrt{1-\cos^2 x}$ na expressão $z = 2\,\mathrm{sen}\,x\cos x$.

```
(%i85)  z: 2*sin(x)*cos(x);
```
$$(\%o85)\quad 2\cos(x)\sin(x)$$
```
(%i86)  z: subst(sin(x) = sqrt(1 - cos(x)^2), z);
```
$$(\%o86)\quad 2\cos(x)\sqrt{1-\cos(x)^2}$$

Seria errado usar um `sublis([sin(x) = sqrt(1 - cos(x)^2)], z)` porque `sin(x)` não é um nome de variável.

2.4 Expansão

O comando **expand(...)** pode ser usado para desenvolver potências ou produtos de expressões algébricas tais como binômios, trinômios etc.

Se for usado na forma **expand(**`expr, expopos, exponeg`**)** então são desenvolvidos apenas as potências de expoentes positivos menores ou iguais a `expopos` e as de expoentes negativos cujos módulos sejam no máximo iguais a `exponeg`.

Exemplo 2.23 Desenvolver as potências $(a-b)^{10}$ e $(a+b+c)^5$.

```
(%i87)  expand((a - b)^10);
```
$$(\%o87)\quad b^{10}-10ab^9+45a^2b^8-120a^3b^7+210a^4b^6-252a^5b^5+210a^6b^4-$$

$120a^7b^3 + 45a^8b^2 - 10a^9b + a^{10}$

```
(%i88)  expand((a + b + c)^5);
```
(%o88) $c^5 + 5bc^4 + 5ac^4 + 10b^2c^3 + 20abc^3 + 10a^2c^3 + 10b^3c^2 + 30ab^2c^2 + 30a^2bc^2 + 10a^3c^2 + 5b^4c + 20ab^3c + 30a^2b^2c + 20a^3bc + 5a^4c + b^5 + 5ab^4 + 10a^2b^3 + 10a^3b^2 + 5a^4b + a^5$

Exemplo 2.24 Efetuar os seguintes produtos:

- $(x^2 + 2)(x^4 - 2x^2 + 4)$,
- $(x^2 + x + 1)(x^6 - x^5 + x^3 - x^2 + 1)$,
- $(x^2 - x + \sqrt{3})(x^2 - x - \sqrt{3})$,
- $(x - 3 + 2i)(x - 3 - 2i)$.

```
(%i89)  (x^2 + 2)*(x^4 - 2*x^2 + 4), expand;
```
(%o89) $x^6 + 8$

```
(%i90)  (x^2 + x + 1)*(x^6 - x^5 + x^3 - x^2 + 1), expand;
```
(%o90) $x^8 + x + 1$

```
(%i91)  (x^2 - x + sqrt(3))*(x^2 - x - sqrt(3)), expand;
```
(%o91) $x^4 - 2x^3 + x^2 - 3$

```
(%i92)  (x - 3 + 2*%i)*(x - 3 - 2*%i), expand;
```
(%o92) $x^2 - 6x + 13$

Exemplo 2.25 Calcular $(-3 + 2i)^5$ e $(\sqrt{3} - 1)^6$.

```
(%i93)  expand( (-3 + 2*%i)^5);
```
(%o93) $122i + 597$

```
(%i94)  expand( (sqrt(3) -1)^6);
```
(%o94) $208 - 40\ 3^{3/2}$

Exemplo 2.26 Expandir $(a + 1)^3(b + 5)^2$ de várias maneiras, variando o valor máximo do expoente da potência desenvolvida.

```
(%i95)  expr2: (a + 1)^3*(b + 5)^2;
```
(%o95) $(a + 1)^3(b + 5)^2$

```
(%i96)  expand(expr2, 1);
```
(%o96) $(a+1)^3(b+5)^2$

```
(%i97)  expand(expr2, 2);
```
(%o97) $(a+1)^3b^2+10(a+1)^3b+25(a+1)^3$

```
(%i98)  expand(expr2, 3);
```
(%o98) $a^3b^2+3a^2b^2+3ab^2+b^2+10a^3b+30a^2b+30ab+10b+25a^3+75a^2+75a+25$

Exemplo 2.27 Expandir $\dfrac{(x+1)^4}{(y+2)^2}$ de várias maneiras, variando os valores máximos dos expoentes das potências desenvolvidas.

```
(%i99)  expr3: (x + 1)^4/(y + 2)^2;
```
(%o99) $\dfrac{(x+1)^4}{(y+2)^2}$

```
(%i100)  expand(expr3, 1, 1);
```
(%o100) $\dfrac{(x+1)^4}{(y+2)^2}$

```
(%i101)  expand(expr3, 1, 2);
```
(%o101) $\dfrac{(x+1)^4}{y^2+4y+4}$

```
(%i102)  expand(expr3, 4, 1);
```
(%o102) $\dfrac{x^4}{(y+2)^2}+\dfrac{4x^3}{(y+2)^2}+\dfrac{6x^2}{(y+2)^2}+\dfrac{4x}{(y+2)^2}+\dfrac{1}{(y+2)^2}$

```
(%i103)  expand(expr3, 4, 2);
```
(%o103) $\dfrac{x^4}{y^2+4y+4}+\dfrac{4x^3}{y^2+4y+4}+\dfrac{6x^2}{y^2+4y+4}+\dfrac{4x}{y^2+4y+4}+\dfrac{1}{y^2+4y+4}$

```
(%i104)  expand(expr3);
```
(%o104) $\dfrac{x^4}{y^2+4y+4}+\dfrac{4x^3}{y^2+4y+4}+\dfrac{6x^2}{y^2+4y+4}+\dfrac{4x}{y^2+4y+4}+\dfrac{1}{y^2+4y+4}$

2.4.1 Expandindo expressões trigonométricas e hiperbólicas

O comando **trigexpand(...)** pode ser usado para expandir expressões que contenham funções trigonométricas ou hiperbólicas de somas ou diferenças de ângulos ou ângulos múltiplos. A ação desse comando é controlada pelas seguintes variáveis:

trigexpand Se for `true`, então todas as somas e produtos serão expandidas, assim que forem digitados. Note que essa variável tem o mesmo nome do comando. O valor padrão (*default*) é `false`.

trigexpandplus Se for `true`, as funções de somas ou diferenças (por exemplo, $\cos(a+b)$) serão expandidas. O valor padrão é `true`.

trigexpandtimes Se for `true`, as funções de produtos (ex. $\cos 3x$) serão expandidos. O valor padrão é `true`.

halfangles Se for `true`, as funções de arcos metades somas (ex. $\cos(x/2)$) serão expandidas. O valor padrão é `false`.

Exemplo 2.28 Expandir $\operatorname{sen}(a+b)$, $\cos(a+b)$ e $\operatorname{tg}(a-b)$.

```
(%i105) trigexpandplus: true;
(%o105) true

(%i106) sin(a + b), trigexpand;
(%o106) cos(a) sin(b) + sin(a) cos(b)

(%i107) trigexpand(cos(a + b));
(%o107) cos(a) cos(b) - sin(a) sin(b)

(%i108) tan(a - b), trigexpand;
                    tan(a) - tan(b)
(%o108)             ---------------
                    tan(a) tan(b) + 1
```

Exemplo 2.29 Expandir $\cos(7x-3y)$

```
(%i109) trigexpand: false;
(%o109) false

(%i110) trigexpandplus: true;
(%o110) true

(%i111) trigexpand(cos(7*x - 3*y));
(%o111) sin(7x) sin(3y) + cos(7x) cos(3y)

(%i112) trigexpand: true;
(%o112) true

(%i113) cos(7*x - 3*y);
(%o113) (-7 cos(x) sin(x)^6+35 cos(x)^3 sin(x)^4-21 cos(x)^5 sin(x)^2+cos(x)^7)
(cos(y)^3-3 cos(y) sin(y)^2)-(-sin(x)^7+21 cos(x)^2 sin(x)^5-35 cos(x)^4 sin(x)^3
+7 cos(x)^6 sin(x))(sin(y)^3-3 cos(y)^2 sin(y))
```

```
(%i114)  expand(%);
```
$$(\%o114)\quad \sin(x)^7\sin(y)^3-21\cos(x)^2\sin(x)^5\sin(y)^3+35\cos(x)^4\sin(x)^3\sin(y)^3-7\cos(x)^6\sin(x)\sin(y)^3+21\cos(x)\sin(x)^6\cos(y)\sin(y)^2-105\cos(x)^3\sin(x)^4\cos(y)\sin(y)^2+63\cos(x)^5\sin(x)^2\cos(y)\sin(y)^2-3\cos(x)^7\cos(y)\sin(y)^2-3\sin(x)^7\cos(y)^2\sin(y)+63\cos(x)^2\sin(x)^5\cos(y)^2\sin(y)-105\cos(x)^4\sin(x)^3\cos(y)^2\sin(y)+21\cos(x)^6\sin(x)\cos(y)^2\sin(y)-7\cos(x)\sin(x)^6\cos(y)^3+35\cos(x)^3\sin(x)^4\cos(y)^3-21\cos(x)^5\sin(x)^2\cos(y)^3+\cos(x)^7\cos(y)^3$$

Note que neste caso não há necessidade de usar um comando como `trigexpand(cos(7*x - 3*y))` porque a variável `trigexpand` teve seu valor modificado para `true` no início. O comando `expand` usado no final efetua apenas algumas multiplicações.

Exemplo 2.30 Expandir tg $\frac{x}{2}$, cos $\frac{x}{2}$ e sen $\frac{x}{2}$.

```
(%i115)  halfangles: true;
```
$$(\%o115)\quad true$$

```
(%i116)  tan(x/2);
```
$$(\%o116)\quad \frac{1-\cos(x)}{\sin(x)}$$

```
(%i117)  cos(x/2);
```
$$(\%o117)\quad \frac{(-1)^{\mathrm{floor}\left(\frac{x+\pi}{2\pi}\right)}\sqrt{\cos(x)+1}}{\sqrt{2}}$$

```
(%i118)  assume(x > 0, x < %pi/2);
```
$$(\%o118)\quad [x>0,\frac{\pi}{2}>x]$$

```
(%i119)  cos(x/2);
```
$$(\%o119)\quad \frac{\sqrt{\cos(x)+1)}}{\sqrt{2}}$$

```
(%i120)  sin(x/2);
```
$$(\%o120)\quad \frac{\sqrt{1-\cos(x)}}{\sqrt{2}}$$

```
(%o121)  halfangles: false;
```
$$(\%o121)\quad false$$

Note que para evitar uma resposta genérica que dependa de uma potência de (-1), o ângulo x foi definido como sendo do primeiro quadrante através de um comando `assume(x > 0, x < %pi/2)`. Com a variável `halfangles` com valor `true`, não há necessidade de usar comandos `trigexpand(...)` para

expandir os arcos-metades, eles são expandidos automaticamente.

2.4.2 O comando trigreduce

O comando **trigreduce(...)** pode ser utilizado para escrever potências de $\cos(x)$, $\text{sen}(x)$, $\cosh(x)$ ou $\text{senh}(x)$ como combinação de $\cos(px)$, $\text{sen}(qx)$, $\cosh(mx)$, $\text{senh}(nx)$.

Exemplo 2.31 Escrever $\cos^8 x$ e $\text{sen}^8 x$ como combinação de senos ou cossenos de ângulos múltiplos.

```
(%i122) trigreduce(cos(x)^8);
```

$$(\%o122)\quad \frac{\cos(8x)+8\cos(6x)+28\cos(4x)+56\cos(2x)+35}{128}$$

```
(%i123) trigreduce(sin(x)^8);
```

$$(\%o123)\quad \frac{\cos(8x)-8\cos(6x)+28\cos(4x)-56\cos(2x)+35}{128}$$

Exemplo 2.32 Escrever $\text{sen}^4 x - 6\cos^2 x\,\text{sen}^2 x + \cos^4 x$ como combinação de senos ou cossenos de ângulos múltiplos.

```
(%i124) trigreduce( sin(x)^4 - 6*cos(x)^2*sin(x)^2 + cos(x)^4);
```

$$(\%o124)\quad \frac{3\cos(4x)-3}{4}+\frac{\cos(4x)+4\cos(2x)+3}{8}+\frac{\cos(4x)-4\cos(2x)+3}{8}$$

```
(%i125) ratsimp(%);
(%o125) cos(4x)
```

2.4.3 Separação em frações parciais

Se o numerador e o denominador de uma fração são polinômios e o denominador pode ser fatorado como produto de polinômios relativamente primos, então ela pode ser decomposta em uma soma de frações mais simples, denominadas *frações parciais*. Para se obter essa decomposição, deve-se usar um comando **partfrac(***fração, variável***)**.

Exemplo 2.33 Decompor $\dfrac{x^3+2x^2-x+11}{x^4-16}$ e $\dfrac{x}{x^8+x+1}$ em frações parciais.

```
(%i126) expr1: (x^3 + 2*x^2 - x + 11)/(x^4 - 16);
```

$$(\%o126)\quad \frac{x^3+2x^2-x+11}{x^4-16}$$

```
(%i127) partfrac(expr1, x);
```

$$(\%o127)\quad \frac{5x-3}{8(x^2+4)}-\frac{13}{32(x+2)}+\frac{25}{32(x-2)}$$

```
(%i128) expr2: x/(x^8 + x + 1);
```

$$(\%o128)\quad \frac{x}{x^8+x+1}$$

```
(%i129) partfrac(expr2, x);
```

$$(\%o129)\quad \frac{5x+2}{19(x^2+x+1)}-\frac{5x^5-8x^4+x^3+12x^2-16x+2}{19(x^6-x^5+x^3-x^2+1)}$$

```
(%i130) ratsimp(%);
```

$$(\%o130)\quad \frac{x}{x^8+x+1}$$

Obtivemos dessa forma as seguintes decomposições:

- $\dfrac{x^3+2x^2-x+11}{x^4-16}=\dfrac{5x-3}{8(x^2+4)}-\dfrac{13}{32(x+2)}+\dfrac{25}{32(x-2)}$
- $\dfrac{x}{x^8+x+1}=\dfrac{5x+2}{19(x^2+x+1)}-\dfrac{5x^5-8x^4+x^3+12x^2-16x+2}{19(x^6-x^5+x^3-x^2+1)}$

Se for usado um `ratsimp(...)` no lado direito de cada igualdade anterior, então recuperamos a fração original, que está do lado esquerdo.

2.5 Racionalização

Dada uma fração com radicais, é possível encontrar outra equivalente sem radicais no denominador. Essa transformação costuma ser denominada *racionalização*.

O *Maxima* efetua racionalização de uma expressão algébrica com radicais com os comandos **ratsimp(...)** ou **rat(...)**, se for atribuído o valor `true` à variável global `algebraic`.

Se p for um polinômio de coeficientes inteiros, um comando **tellrat(p)** pode ser usado para adicionar as raízes de p ao anel dos inteiros algébricos conhecidos do *Maxima*. Os números assim acrescentados são levados em consideração nas simplificações efetuadas com o `ratsimp(...)` ou o `rat(...)`.

Exemplo 2.34 Racionalizar $\dfrac{16}{5-\sqrt{3}}$.

```
(%i131) ratsimp(16/(5 - sqrt(3)));
```

(%131) $-\frac{16}{\sqrt{3}-5}$

(%132) `ratsimp(16/(5 - sqrt(3))), algebraic;`

(%132) $\frac{8\sqrt{3}+40}{11}$

Usar um "`, algebraic`" ao final da linha, equivale a usar temporariamente uma atribuição do tipo `algebraic: true`.

Exemplo 2.35 Racionalizar $\dfrac{1}{\sqrt{1+\sqrt{1+\sqrt{2}}}}$.

(%133) `ratsimp( 1/sqrt(1 + sqrt(1 + sqrt(2)))), algebraic;`

(%133) $$\frac{\sqrt{\sqrt{\sqrt{2}+1}+1}\left(\sqrt{2}\sqrt{\sqrt{2}+1}-\sqrt{2}\right)}{2}$$

Exemplo 2.36 Determinar $a_1, a_2, a_3, a_4, a_5 \in \mathbb{Q}$ tais que

$$\frac{1}{7-\sqrt[5]{2}} = a_1 + a_2\sqrt[5]{2} + a_3\sqrt[5]{2^2} + a_4\sqrt[5]{2^3} + a_5\sqrt[5]{2^4}.$$

(%134) `ratsimp( 1/(7-2^(1/5)) ), algebraic;`

(%134) $$\frac{2^{4/5} + 7\ 2^{3/5} + 49\ 2^{2/5} + 343\ 2^{1/5} + 2401}{16805}$$

Observando a resposta do programa, chegamos à seguinte conclusão:
$a_1 = \frac{2401}{16805}$, $a_2 = \frac{343}{16805}$, $a_3 = \frac{49}{16805}$, $a_4 = \frac{7}{16805}$, e $a_5 = \frac{1}{16805}$.

Exemplo 2.37 Seja s uma raiz da equação $x^5 + 3x^2 + 1 = 0$, racionalizar as expressões $\dfrac{s+1}{s^2+s+3}$ e $\dfrac{s^9+5s+4}{s^9-3s^2-1}$.

(%135) `tellrat(s^5 + 3*s^2 + 1);`

(%135) $[s^5 + 3s^2 + 1]$

(%136) `algebraic: true$`

(%137) `ratsimp((s + 1)/(s^2 + s + 3));`

(%137) $$\frac{7s^4 + 2s^3 - 23s^2 + 38s + 58}{165}$$

(%138) `ratsimp((s^9 + 5*s + 4)/(s^9 - 3*s^2 - 1));`

(%138) $$-\frac{2906s^4 - 1141s^3 - 904s^2 + 9153s - 3214}{89}$$

Capítulo 3

Listas e Conjuntos

Listas e conjuntos são dois conceitos básicos do *Maxima* que estão entre os mais importantes. Eles aparecem nas situações mais diversas tais como: solução de uma equação, solução de um sistema de equações, construção de um gráfico, definição de uma matriz, divisão de polinômios etc.

3.1 Conjuntos

Um conjunto é um agrupamento de elementos no qual a ordem não é importante. Podem ser criados com um comando **set(...)** ou com chaves { **...** }. O conjunto vazio pode ser criado com um `set()` ou com { }.

Diversas operações e funções foram implementadas para conjuntos:

- **union(A, B)** $A \cup B$, união dos conjuntos A e B
- **intersecton(A, B)** $A \cap B$, interseção dos conjuntos A e B
- **setdifference(A, B)** $A - B$, diferença dos conjuntos A e B
- **cartesian_product(A, B)** $A \times B$, produto cartesiano dos conjuntos A e B
- **powerset(A)** $\wp(A)$, conjunto das partes de A
- **cardinality(A)** número de elementos de A
- **elementp(x, A)** verifica se o elemento x pertence ao conjunto A, retorna `true` ou `false`
- **subsetp(A, B)** verifica se A é subconjunto de B, retorna `true` ou `false`
- **subset(A, propriedade)** subconjunto de A cujos elementos satisfazem determinada propriedade.

Exemplo 3.1 Verificar se os conjuntos $A = \{a, b, c\}$, $B = \{a, b, x\}$ e $C = \{c, c, b, a, a\}$ são iguais. O comando **is(**`predicado`**)** verifica se o predicado dado é verdadeiro (`true`) ou falso (`false`). Por exemplo, `is(4 = 4)` é `true`, enquanto que `is(5 > 9)` é `false`.

```
(%i1)  A: {a, b, c};
(%o1)  {a, b, c}

(%i2)  B: {a, b, x};
(%o2)  {a, b, x}

(%i3)  C: {c, c, b, a, a};
(%o3)  {a, b, c}

(%i4)  is(A = B);
(%o4)  false

(%i5)  is(A = C);
(%o5)  true

(%i6)  is(B = C);
(%o6)  false
```

As respostas do programa significam que $A \neq B$, $A = C$ e $B \neq C$.

Exemplo 3.2 Sejam $A = \{-5, -2, 0, 1, 3, 4\}$ e $B = \{-2, -1, 1, 3, 7, 8\}$. Calcule $U = A \cup B$, $I = A \cap B$, $D_1 = A - B$, $D_2 = B - A$ e verifique se $(A \cup B) - (A \cap B) = (A - B) \cup (B - A)$, ou seja, se $U - I = D_1 \cup D_2$.

```
(%i7)  A: set(-5, -2, 0, 1, 3, 4);
(%o7)  {-5, -2, 0, 1, 3, 4}

(%i8)  B: set(-2, -1, 1, 3, 7, 8);
(%o8)  {-2, -1, 1, 3, 7, 8}

(%i9)  U: union(A, B);
(%o9)  {-5, -2, -1, 0, 1, 3, 4, 7, 8}

(%i10)  I: intersection(A, B);
(%o10)  {-2, 1, 3}

(%i11)  D1: setdifference(A, B);
(%o11)  {-5, 0, 4}

(%i12)  D2: setdifference(B, A);
(%o12)  {-1, 7, 8}
```

```
(%i13)  is(setdifference(U, I) = union(D1, D2));
(%o13)  true
```

As respostas do programa significam que

- $A \cup B = \{-5, -2, -1, 0, 1, 3, 4, 7, 8\}$;
- $A \cap B = \{-2, 1, 3\}$;
- $A - B = \{-5, 0, 4\}$;
- $B - A = \{-1, 7, 8\}$;
- É verdade que $(A \cup B) - (A \cap B) = (A - B) \cup (B - A)$.

Exemplo 3.3 Calcular o conjunto das partes de $X = \{p, q, r, s\}$ e a sua cardinalidade.

```
(%i14)  X: {p, q, r, s} ;
(%o14)  {p, q, r, s}

(%i15)  Y: powerset(X);
(%o15)  {{}, {p}, {p, q}, {p, q, r}, {p, q, r, s}, {p, q, s}, {p, r}, {p, r, s}, {p, s},
{q}, {q, r}, {q, r, s}, {q, s}, {r}, {r, s}, {s}}

(%i16)  cardinality(Y);
(%o15)  16
```

Exemplo 3.4 Sejam $M = \{\}$ e $N = \{\}$. Calcular os produtos cartesianos $M \times N$ e $N \times M$, verificar se $M \times N = N \times M$ e calcular a quantidade de elementos de um desses produtos. No *Maxima*, um par ordenado (x, y) é denotado por $[x, y]$. Como o comando `cartesian_product` tem um nome longo, criamos um apelido `cart` para esse nome.

```
(%i17)  alias(cart, cartesian_product);
(%o17)  [cart]

(%i18)  M: set(4, 6, 8);
(%o18)  {4, 6, 8}

(%i19)  N: set(a, b, c);
(%o19)  {a, b, c}

(%i20)  cart(M, N);
(%o20)  {[4, a], [4, b], [4, c], [6, a], [6, b], [6, c], [8, a], [8, b], [8, c]}
```

```
(%i21)  cart(N, M);
(%o21)  {[a, 4], [a, 6], [a, 8], [b, 4], [b, 6], [b, 8], [c, 4], [c, 6], [c, 8]}

(%i22)  cardinality(%);
(%o22)  9

(%i23)  is(cart(M, N) = cart(N, M));
(%o23)  false
```

Exemplo 3.5 Seja $A = \{1, 2, 3, 4, \{4\}, \{5\}, \{4, 5, 6\}\}$. Verificar se é verdade que $1 \in A$, $5 \in A$, $\{4\} \in A$, $\{6\} \in A$ e $\{4, 5, 6\} \in A$.

```
(%i24)  A = { 1, 2, 3, 4, {4}, {5}, {4, 5, 6} };
(%o24)  {1, 2, 3, 4, {4}, {4, 5, 6}, {5}}

(%i25)  elementp(1, A);
(%o25)  true

(%i26)  elementp(5, A);
(%o26)  false

(%i27)  elementp({4}, A);
(%o27)  true

(%i28)  elementp({6}, A);
(%o28)  false

(%i29)  elementp({4, 5, 6}, A);
(%o29)  true
```

As respostas do programa a essas perguntas significam que $1 \in A$, $5 \notin A$, $\{4\} \in A$, $\{6\} \notin A$ e $\{4, 5, 6\} \in A$.

Exemplo 3.6 Seja $A = \{1, 2, 3, 4, \{4\}, \{5\}, \{4, 5, 6\}\}$. Verificar se é verdade que $\emptyset \subset A$, $\{1\} \subset A$, $\{5\} \subset A$, $\{2, 3, 4\} \subset A$, $\{4, 5, 6\} \subset A$ e calcular o número de elementos de A.

```
(%i30)  A = { 1, 2, 3, 4, {4}, {5}, {4, 5, 6} };
(%o30)  {1, 2, 3, 4, {4}, {4, 5, 6}, {5}}

(%i31)  subsetp({ }, A);
(%o31)  true

(%i32)  subsetp({1}, A);
```

```
(%o32)  true

(%i33)  subsetp({5}, A);
(%o33)  false

(%i34)  subsetp({2, 3, 4}, A);
(%o34)  true

(%i35)  subsetp({4, 5, 6}, A);
(%o35)  false

(%i36)  cardinality(A);
(%o36)  7
```

Essas respostas do programa significam que $\emptyset \subset A$, $\{1\} \subset A$, $\{5\} \not\subset A$, $\{2,3,4\} \subset A$, $\{4,5,6\} \not\subset A$ e $n(A) = 7$.

Exemplo 3.7 Considerando o conjunto $A = \{0,1,2,3,4,5,6,7,8,9,10\}$, definir os seguintes subconjuntos: B números pares de A, C números ímpares de A, D múltiplos de 3 de A e E dos múltiplos de 5 de A. A propriedade "x deixa resto igual a b quando é dividido por a" pode ser codificada no *Maxima* da seguinte forma: `lambda([x], is(mod(x, a) = b))`.

```
(%i37)  A: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
(%o37)  {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

(%i38)  B: subset(A, evenp);
(%o38)  {0, 2, 4, 6, 8, 10}

(%i39)  C: subset(A, oddp);
(%o39)  {1, 3, 5, 7, 9}

(%i40)  D: subset(A, lambda([x], is(mod(x, 3) = 0)));
(%o41)  {0, 3, 6, 9}

(%i42)  E: subset(A, lambda([x], is(mod(x, 5) = 0)));
(%o42)  {0, 5, 10}
```

3.2 Listas

Uma lista é um conjunto ordenado de elementos, separados entre si por vírgulas e envolvidos por colchetes, como nos seguintes exemplos: $[1,2,3,4,5]$, $[10,12,14]$ e $[0,0,a,b,x+1,y/3]$.

O número de coordenadas de uma lista é denominado comprimento da lista. Se L for uma lista de comprimento n, então a enésima coordenada dessa lista é denotada por $L[n]$. Por exemplo, $L = [x, y, z, 3, 8]$ é uma lista de comprimento 5 na qual $L[1] = x$, $L[2] = y$, $L[3] = z$, $L[4] = 3$ e $L[5] = 8$.

3.2.1 Operações e funções com listas

Diversas operações e funções são implementadas para as listas L e M. Algumas delas são as seguintes:

- **L+M** adição de listas de mesmo comprimento;
  Exemplo: [8, 4, z] + [1, 3, 6] = [9, 7, z+6]
- **L-M** subtração de listas de mesmo comprimento;
  Exemplo: [8, 4, z] - [1, 3, 6] = [7, 1, z-6]
- **L*M** multiplicação de listas de mesmo comprimento;
  Exemplo: [x, y, 3] * [2, 5, 7] = [2x, 5y, 21]
- **L/M** divisão de listas de mesmo comprimento;
  Exemplo: [x, y, z] / [2, 5, 7] = [x/2, y/5, z/7]
- **L.M** produto interno de listas de mesmo comprimento;
  Exemplo: [2, 3, 5] . [a, b, c] = 2a + 3b + 5c
- **k*L** multiplicação de uma lista por um mesmo valor;
  Exemplo: 4*[2, 3, 5] = [8, 12, 20]
- **1/L** lista dos inversos multiplicativos;
  Exemplo: 1/[2, 3, 5] = [1/2, 1/3, 1/5]
- **L^a** lista de potências com mesmo expoente;
  Exemplo: [2, 3, 5]^2 = [4, 9, 25]
- **a^L** lista de potências com mesma base;
  Exemplo: a^[2, 3, 5] = [a^2, a^3, a^5]
- **sqrt(L)** raízes quadradas;
  Exemplo: sqrt([9, 25, 81]) = [3, 5, 9]
- **exp(L)** lista das exponenciais de base e;
  Exemplo: exp([2, 3, x]) = [%e^2, %e^3, %e^x]
- **log(L)** lista de logaritmos naturais;
  Exemplo: log[1, %e, x] = [0, 1, log(x)]

- **cos(L)** lista de cossenos;
  Exemplo: cos([0, %pi/3, a]) = [1, 1/2, cos(a)]

- **sin(L)** lista de senos;
  Exemplo: sin([0, %pi/3, a]) = [0, $\frac{\sqrt{3}}{2}$, sin(a)]

- **lenght(L)** comprimento da lista;
  Exemplo: lenght([a, b, c, d, e]) = 5

- **first(L)** primeiro elemento da lista;
  Exemplo: first([a, b, c, d, e]) = a

- **last(L)** último elemento da lista;
  Exemplo: last([a, b, c, d, e]) = e

- **reverse(L)** lista na ordem inversa;
  Exemplo: reverse([a, b, c, d, e]) = [e, d, c, b, a]

- **sort(L)** classifica em ordem crescente ou alfabética;
  Exemplo: sort([4, x, 7, b, a, 1, c]) = [1, 4, 7, a, b, c, x]

- **permutations(L)** conjunto das permutações de uma lista;
  Exemplo: permutations([1, 2, 3]) = { [1,2,3], [1,3,2], [2,1,3], [2,3,1], [3,1,2], [3,2,1] }

- **listp(L)** Verifica se L é uma lista;
  Exemplos: listp([a, b, c]) = true, listp({a, b, c}) = false

- **emptyp(L)** Verifica se L é uma lista vazia ou um conjunto vazio;
  Exemplos: emptyp([ ]) = true, emptyp([1, 2, 3]) = false

- **member(x, L)** Verifica se x é membro da lista L;
  Exemplos: member(2, [1, 2, 3]) = true, member(4, [1, 2, 3]) = false

- **append(L1, L2)** emenda as listas L1 e L2;
  Exemplo: append([1, 2, 3, 4], [a, b, c]) = [1, 2, 3, 4, a, b, c]

- **cons(x, L)** lista formada por x seguido dos elementos de L;
  Exemplo: cons(5, [1, 2, 3]) = [5, 1, 2, 3]

- **endcons(x. L)** lista formada pelos elementos de L seguido de x;
  Exemplo: endcons(5, [1, 2, 3]) = [1, 2, 3, 5]

Exemplo 3.8 Dadas as listas L = [4, 5, 6, 7] e M = [7, 4, 6, 5], verificar se essas listas são iguais, criar uma nova lista S formada pela emenda das listas

L e M, mostrar o quarto e o sexto elementos de S e escrever S na ordem inversa.

```
(%i43)  L: [4, 5, 6, 7];
(%o43)  [4, 5, 6, 7]

(%i44)  M: [7, 4, 6, 5];
(%o44)  [7, 4, 6, 5]

(%i45)  is(L = M);
(%o46)  false

(%i47)  S: append(L, M);
(%o47)  [4, 5, 6, 7, 7, 4, 6, 5]

(%i48)  S[4];
(%o48)  7

(%i49)  S[6];
(%o49)  4

(%i50)  reverse(S);
(%o50)  [5, 6, 4, 7, 7, 6, 5, 4]
```

Exemplo 3.9 Seja L a lista [-1, 1, -2, 2]. Adicionar 2 a cada coordenada de L e determinar o conjunto P de todas as permutações de L e o número de elementos de P.

```
(%i51)  L: [-1, 1, -2, 2];
(%o51)  [1, −1, −2, 2]

(%i52)  L: L + [2, 2, 2, 2];
(%o52)  [1, 3, 0, 4

(%i53)  P: permutations(L);
(%o53)  {[0, 1, 3, 4], [0, 1, 4, 3], [0, 3, 1, 4], [0, 3, 4, 1], [0, 4, 1, 3], [0, 4, 3, 1],
[1, 0, 3, 4], [1, 0, 4, 3], [1, 3, 0, 4], [1, 3, 4, 0], [1, 4, 0, 3], [1, 4, 3, 0],
[3, 0, 1, 4], [3, 0, 4, 1], [3, 1, 0, 4], [3, 1, 4, 0], [3, 4, 0, 1], [3, 4, 1, 0],
[4, 0, 1, 3], [4, 0, 3, 1], [4, 1, 0, 3], [4, 1, 3, 0], [4, 3, 0, 1], [4, 3, 1, 0]}
(%i54)  cardinality(P);
(%o54)  24
```

3.2.2 Múltiplas atribuições

Uma lista pode ser usada para fazer atribuições simultaneas a várias variáveis. Para isso, basta colocar uma lista de variáveis, seguida de dois pontos e da lista de valores atribuídos.

Exemplo 3.10 Neste exemplo, atribuímos os valores $2, -11, 9$ a x, y, z, respectivamente. Depois, adicionamos 1 a cada uma dessas variáveis.

```
(%55)  [x, y, z]: [2, -11, 9];
(%55)  [2, −11, 9]

(%56)  x;
(%56)  2

(%57)  y;
(%57)  −11

(%58)  z;
(%58)  9

(%59)  [x, y, z]: [x, y, z] + [1, 1, 1]
(%59)  [3, −10, 10]
```

3.2.3 Aplicando uma função a uma lista

Os comandos **map(f, lista)** e **apply(f, lista)** podem ser usados para aplicarem uma função ou uma operação aos elementos de uma ou várias listas.

- Se f for uma função, então `map(`f`, [`$a_1, a_2, \ldots, a_n$`])` fornece como resultado $[f(a_1), f(a_2), \ldots, f(a_n)]$, enquanto que `apply(`$f$`, [`$a_1, a_2, \ldots, a_n$`])` resulta em $f(a_1, a_2, \ldots, a_n)$.
- Se op for uma operação, então `apply("op", [`$a_1, a_2, \ldots, a_n$`])` resulta em a_1 `op` a_2 `op ...op` a_n.
- Se F for uma função, então `map(`F`, [`$a_1, \ldots, a_n$`], [`$b_1, \ldots, b_n$`], ..., [`$z_1, \ldots, z_n$`])` resulta em `[`$F(a_1, b_1, \ldots, z_1), F(a_2, b_2, \ldots, z_2), \ldots, F(a_n, b_n, \ldots, z_n)$` ]`.

Exemplo 3.11 Aplicar uma função f às listas $[a]$, $[a, b]$ e $[a, b, c]$ usando o `map` e o `apply`.

```
(%i60) map(f, [a]);
(%o60) [f(a)]

(%i61) apply(f, [a]);
(%o61) f(a)

(%i62) map(f, [a, b]);
(%o62) [f(a), f(b)]

(%i63) apply(f, [a, b]);
(%o63) f(a, b)

(%i64) map(f, [a, b, c]);
(%o64) [f(a), f(b), f(c)]

(%i65) apply(f, [a, b, c]);
(%o65) f(a, b, c)
```

Exemplo 3.12 Usando o `map`, aplicar as seguintes funções às seguintes listas:

função	lista(s)
log	$[1, e, e^2, e^3]$
g	$[a, b, c]$, $[x, y, z]$
h	$[a, b, c]$, $[x, y, z]$, $[r, s, t]$
cos	$[x, y, z, w, t]$
cos	$[0, \frac{\pi}{6}, \frac{\pi}{4}, \frac{\pi}{3}, \pi]$
binomial	$[a, b, c, d]$, $[m, n, p, q]$
binomial	$[10, 9, 8, 7]$, $[2, 3, 4, 5]$
factor	$[x^2 - 9,\ x^2 - 16,\ x^2 - 5x,\ x^4 - 5x^2 + 4]$

```
(%i66) map(log, [1, %e, %e^2, %e^3]);
(%o66) [0, 1, 2, 3]

(%i67) map(g, [a, b, c], [x, y, z]);
(%o67) [g(a, x), g(b, y), g(c, z)]

(%i68) map(h, [a, b, c], [x, y, z], [r, s, t]);
(%o68) [h(a, x, r), h(b, y, s), h(c, z, t)]

(%i69) map(cos, [x, y, z, w, t]);
(%o69) [cos(x), cos(y), cos(z), cos(w), cos(t)]

(%i70) map(cos, [0, %pi/6, %pi/4, %pi/4, %pi]);
(%o70) [1, sqrt(3)/2, 1/sqrt(2), 1/sqrt(2), -1]

(%i71) map(binomial, [a, b, c, d], [m, n, p, q]);
```

```
(%o71)  [binomial(a, m), binomial(b, n), binomial(c, p), binomial(d, q)]
(%i72)  map(binomial, [10, 9, 8, 7], [2, 3, 4, 5]);
(%o72)  [45, 84, 70, 21]
(%i73)  map(factor, [x^2 -9, x^2 - 16, x^2 - 5*x, x^4 - 5*x^2 + 4]);
(%o73)  [(x-3)(x+3), (x-4)(x+4), (x-5)x, (x-2)(x-1)(x+1)(x+2)]
```

Exemplo 3.13 Usando o `apply`, aplicar as operações +, − e * à lista [a, b, c, d].

```
(%i74)  apply("+", [a, b, c, d]);
(%o74)  d + c + b + a
(%i75)  ordergreat(a, b, c, d);
(%o75)  done
(%i76)  apply("+", [a, b, c, d]);
(%o76)  a + b + c + d
(%i77)  apply("-", [a, b, c, d]);
(%o77)  a - b - c - d
(%i78)  apply("*", [a, b, c, d]);
(%o78)  abc
```

3.2.4 Criando uma lista a partir do termo geral

Uma sequência finita cujo termo geral é f(k), com k variando de k1 a k2, pode ser criada com um comando **makelist(f(k), k, k1, k2)**. Alternativamente, um comando **create_list(f(k), k, k1, k2)** também pode ser usado para definir esse tipo de sequência. Esses comandos são semelhantes, mas o `create_list(...)` admite que o termo geral da sequência tenha vários índices.

Exemplo 3.14 Listar os 15 primeiros termos das sequências cujos termos gerais são $a_k = 3k - 19$, $b_k = \frac{40}{2^k}$ e $c_k = \cos(\frac{2k\pi}{5k-4})$.

```
(%i79)  a: makelist(3*k - 19, k, 1, 15);
(%o79)  [-16, -13, -10, -7, -4, -1, 2, 5, 8, 11, 14, 17, 20, 23, 26]
```

```
(%i80)  b: makelist(40/2^k, k, 1, 15);
```
(%o80) $[20, 10, 5, \frac{5}{2}, \frac{5}{4}, \frac{5}{8}, \frac{5}{16}, \frac{5}{32}, \frac{5}{64}, \frac{5}{128}, \frac{5}{256}, \frac{5}{512}, \frac{5}{1024}, \frac{5}{2048}, \frac{5}{4096}]$

```
(%i81)  c: makelist(cos(2*k*%pi/(5*k - 4)), k, 1, 15);
```
(%o81) $[1, -\frac{1}{2}, \cos(\frac{6\pi}{11}), 0, \cos(\frac{10\pi}{21}), \cos(\frac{6\pi}{13}), \cos(\frac{14\pi}{31}, \cos(\frac{4\pi}{9}), \cos(\frac{18\pi}{41}),$
$\cos(\frac{10\pi}{23}), \cos(\frac{22\pi}{51}), \cos(\frac{3\pi}{7}), \cos(\frac{26\pi}{61}), \cos(\frac{14\pi}{33}), \cos(\frac{30\pi}{71})]$

Exemplo 3.15 Listar todos os termos da sequência cujo termo geral é x^j+x^k+1, com $1 \leq j \leq 3$ e $1 \leq k \leq 4$.

Como temos dois índices, j e k, devemos usar um `create_list(...)`.

```
(%i82)  create_list(x^j + x^k + 1, j, 1, 3, k, 1, 4);
```
(%o82) $[2x+1, x^2+x+1, x^3+x+1, x^4+x+1, x^2+x+1, 2x^2+1, x^3+x^2+1, x^4+x^2+1, x^3+x+1, x^3+x^2+1, 2x^3+1, x^4+x^3+1]$

Exemplo 3.16 Verificar se o inteiro 7491000 é igual a n^3-n^2-10 para algum n de 1 a 200.

Neste caso, basta criar a sequência cujo termo geral $f(n)$ é dado e verificar se o inteiro citado é membro dessa sequência.

```
(%i83)  L: makelist(n^3 - n^2 - 10, n, 1, 200)$
(%i84)  member(7491000, L);
(%o84)  false
```

Pela resposta do programa, concluímos que 7491000 não é da forma n^3-n^2-10 para n tal que $1 \leq n \leq 200$.

3.3 Equações e sistemas de equações

O *Maxima* possui vários comandos para resolução de equações. Entre eles, citaremos agora apenas os seguintes comandos:

- **solve(equação, variável)** ou **solve(equação)** resolve uma equação polinomial. Se houver mais de uma variável envolvida, então devem ser fornecidas as variáveis cujos valores serão calculados.
- **solve([lista de equações], [lista de variáveis])** resolve um sistema de equações polinomiais.

- **allroots(equação)** determina numericamente as raízes reais ou complexas de uma equação polinomial de uma variável e coeficientes reais.
- **realroots(equação, erro)** determina as raízes reais de uma equação polinomial de uma variável e coeficientes reais, usando um erro de aproximação fornecido.
- **find_root(f(x), a, b)** determina uma raiz de $f(x) = 0$ no intervalo $[a, b]$. A função f deve ser contínua e $f(a)$ e $f(b)$ devem ter sinais contrários.

Se houver multiplicidade de raízes, ela pode ser mostrada através da variável **multiplicities**.

Normalmente, as soluções encontradas são mostradas no formato de lista de equações. Se a solução for única, então ela também pode ser mostrada no formato de lista de atribuições, se a variável global `globalsolve` for ajustada em `true`.

Exemplo 3.17 Resolver a equação $x^4 + x^3 - 8x^2 - 7x + 7 = 0$.

```
(%i85)  solve(x^4 + x^3 - 8*x^2 - 7*x + 7 = 0);
```
(%o85) $[x = -\sqrt{7},\ x = \sqrt{7},\ x = -\frac{\sqrt{5}+1}{2},\ x = \frac{\sqrt{5}-1}{2}]$

Obtemos assim uma lista com as quatro raízes da equação: $-\sqrt{7}$, $\sqrt{7}$, $-\frac{\sqrt{5}+1}{2}$, e $\frac{\sqrt{5}-1}{2}$. No caso deste exemplo, `solve(x^4+x^3-8*x^2-7*x+7=0)` é equivalente a `solve(x^4+x^3-8*x^2-7*x+7=0, x)`.

Exemplo 3.18 Considerando a equação $yx^2 - yx - 4 = 0$, escreva x em função de y e, depois, y em função de x.

```
(%i86)  eq2: y*x^2 - y*x - 4 = 0;
```
(%o86) $x^2y - xy - 4 = 0$

```
(%i87)  solve(eq2, x);
```
(%o87) $[x = -\frac{\sqrt{y^2+16y}-y}{2y},\ x = \frac{\sqrt{y^2+16y}+y}{2y}]$

```
(%i88)  solve(eq2, y);
```
(%o88) $[y = \frac{4}{x^2-x}]$

Exemplo 3.19 Determine as raízes da equação

$$625x^5 + 2750x^4 - 27650x^3 + 66640x^2 - 66199x + 24010 = 0$$

e suas respectivas multiplicidades.

```
(%i89)  eq: 625*x^5+2750*x^4-27650*x^3+66640*x^2-66199*x+24010=0;
```

(%o89) $625x^5 + 2750x^4 - 27650x^3 + 66640x^2 - 66199x + 24010 = 0$

```
(%i90)  solve(eq);
```
(%o90) $[x = -10, x = \frac{7}{5}]$

```
(%i91)  multiplicities;
```
(%o91) $[1, 4]$

Concluímos dessa forma que $x = -10$ é uma raiz de multiplicidade 1 e que $x = \frac{7}{5}$ é raiz de multiplicidade 4.

Exemplo 3.20 Determine as raízes da equação $x^8 + 4x^3 - 100 = 0$.

```
(%i92)  eq: x^8 + 4*x^3 - 100 = 0;
```
(%o92) $x^8 + 4x^3 - 100 = 0$

```
(%i93)  solve(eq);
```
(%o93) $[0 = x^8 + 4x^3 - 100]$

```
(%i94)  realroots(eq, 1e-8);
```
(%o94) $[x = -\frac{490567143}{268435456},\ x = \frac{463766039}{268435456}]$

```
(%i95)  float(%)
```
(%o95) $[x = -1.827505018562079,\ x = 1.72766312584281]$

```
(%i96)  allroots(eq);
```
(%o96) $[x = 1.258004092917\%i + 1.3069310214045, x = 1.3069310214045 - 1.258004092917\%i,\ x = 1.2578459713267\%i - 1.2069310215045,\ x = -1.2578459713267\%i - 1.2069310215045,\ x = 1.7789900329062\%i - 0.0500790530371,\ x = -1.7789900329062\%i - 0.0500790530371,\ x = 1.7276631227920,\ x = -1.8275050165177]$

Como se vê, o `solve(...)` não consegue achar nenhuma raiz, o `realroots(...)` encontra duas raízes reais com um erro da ordem de 10^{-8} (1e-8) e o `allroots(...)` encontra todas as raízes reais ou complexas.

Exemplo 3.21 Determinar a solução do sistema

$$\begin{cases} x+y+z &= 3 \\ xy+yz+xz &= -18 \\ x^3+y^3+z^3 &= 189 \end{cases}$$

```
(%i97)  q1: x+y+z=3; q2: x*y+y*z+x*z = -18; q3: x^3+y^3+z^3 = 189;
```
(%o97) $z+y+x=3$

(%o98) $yz+xz+xy=-18$

(%o98) $z^3+y^3+x^3=189$

```
(%i99)  solve([q1, q2, q3], [x, y, z]);
```
(%o99) $[[x=0,y=6,z=-3],[x=0,y=-3,z=6],[x=6,y=0,z=-3],[x=6,y=-3,z=0],[x=-3,y=0,z=6],[x=-3,y=6,z=0]]$

A resposta é apresentada no formato de lista de listas de equações. Poderíamos também ter usado um comando `solve([q1, q2, q3])` que teríamos obtido a mesma resposta do programa.

Exemplo 3.22 Resolver o seguinte sistema linear:

$$\begin{cases} 2x-3y+5z &= 10 \\ 4x-y+2z &= 11 \\ 5x-2y+z &= -15 \end{cases}$$

```
(%i100) ordergreat(x, y, z);
```
(%o100) *done*

```
(%i101) s1: 2*x - 3*y + 5*z = 10; s2: 4*x - y + 2*z = 11; s3: 5*x - 2*y +
z = -15;
```
(%o101) $2x-3y+5z=10$

(%o102) $4x-y+2z=11$

(%o103) $5x-2y+z=-15$

```
(%i104) solve([s1, s2, s3]);
```
(%o104) $[[x=\frac{32}{27},\ y=\frac{433}{27},\ z=\frac{301}{27}]]$

```
(%i105) x;
```
(%o105) x

```
(%i106) y;
```
(%o106) y

```
(%i107) globalsolve: true;
```

```
(%o107)  true

(%i108)  solve([s1, s2, s3]);
(%o108)  [[x : 32/27, y : 433/27, z : 301/27]]

(%i109)  x;
(%o109)  32/27

(%i110)  y;
(%o110)  433/27

(%i111)  unorder();
(%o111)  [z, y, x]
```

Note que quando a resposta é apresentada como lista de equações, as variáveis continuam sem valores definidos. No entanto, se `globalsolve` for ajustada em `true`, a resposta é apresentada como lista de atribuições e, nesse caso, cada variável tem seu valor definido de acordo com a solução do problema.

Através do comando `ordergreat(...)` podemos especificar uma ordem na qual as variáveis devem ser mostradas. Para cancelar essa ordem, devemos usar um `unorder()`.

Exemplo 3.23 Determinar uma raiz de cada uma das equações $4^x + 6^x = 9^x$, $2^x = x^2$ e $\cos x = x^2 - 9$ que pertençam aos intervalos $[1,\ 2]$, $[-2,\ 0]$ e $[0,\ 5]$, respectivamente.

```
(%i112)  find_root(4^x + 6^x = 9^x, 1, 2);
(%o112)  1.186814390280981

(%i113)  find_root(2^x = x^2, -2, 0);
(%o113)  −0.76666469596212

(%i114)  find_root(cos(x) = x^2-9, 0, 5);
(%o114)  2.836575103492521
```

3.4 Transformando conjuntos em listas e vice-versa

Uma lista é um agrupamento de elementos no qual a ordem é importante. Um conjunto é um agrupamento semelhante no qual a ordem dos elementos não importa. Algumas operações são possíveis apenas com um desses conceitos. Por

exemplo, calcular uma interseção só é possível se for entre dois conjuntos. Não há interseção de listas. Por outro lado, o elemento de determinada ordem (por exemplo, o primeiro elemento) só é possível de ser calculado se for em uma lista. Não há primeiro elemento de um conjunto.

O *Maxima* possui dois comandos que permite transformar um conjunto em uma lista ou vice-versa. Esses comandos são os seguintes:

- **listify(...)** transforma conjunto em lista;
  Exemplo: listify({a, b, c}) = [a, b, c];
- **setify(...)** transforma lista em conjunto.
  Exemplo: setify([a, b, c]) = {a, b, c}.

Exemplo 3.24 Considerando duas listas L1= [3, 2, 1, 0, 7, 10, 3, 3, 7, 10] e L2 = [11, 3, 4, 10, 11, 1, 0] dadas, construir uma lista L formada pelos elementos que aparecem em uma das listas e não aparecem na outra.

As listas L1 e L2 são transformadas em conjuntos S1 e S2. Nessa transformação, a repetição de elementos é automaticamente eliminada. A partir daí, calculamos o conjunto $(S1 \cup S2) - (S1 \cap S2)$ e, no final, transformamos esse conjunto em uma lista L.

```
(%i115)  L1: [3, 2, 1, 0, 7, 10, 3, 3, 7, 10];
(%o115)  [3, 2, 1, 0, 7, 10, 3, 3, 7, 10]

(%i116)  L2: [11, 3, 4, 10, 11, 1, 0];
(%o116)  [11, 3, 4, 10, 11, 1, 0]

(%i117)  S1: setify(L1);
(%o118)  {0, 1, 2, 3, 7, 10}

(%i119)  S2: setify(L2);
(%o119)  {0, 1, 3, 4, 10, 11}

(%i120)  U: union(S1, S2);
(%0120)  {0, 1, 2, 3, 4, 7, 10, 11}

(%i121)  I: intersection(S1, S2);
(%o121)  0, 1, 3, 10

(%i122)  Delta: setdifference(U, I);
(%o122)  {2, 4, 7, 11}

(%i123)  L: listify(Delta);
(%o123)  [2, 4, 7, 11]
```

Portanto, a lista L procurada é igual a [2, 4, 7, 11].

Exemplo 3.25 Determine o conjunto-solução da equação

$$x^6 - 9x^5 + 53x^4 - 405x^3 + 684x^2 - 2916x + 2592 = 0.$$

O *Maxima* apresenta as raízes de uma equação na forma de lista-solução onde cada coordenada é uma equação do tipo `variável = raiz`. Logo, é preciso mapear a função `rhs` para obter somente o lado direito de cada equação e, depois, transformar a lista em conjunto.

```
(%i124)  eq: x^6 - 9*x^5 + 53*x^4 - 405*x^3 + 684*x^2 - 2916*x + 2592
= 0
(%o124)  x^6 - 9*x^5 + 53*x^4 - 405*x^3 + 684*x^2 - 2916*x + 2592 = 0

(%i125)  solve(eq);
(%o125)  [x = 1, x = 8, x = -3%i, x = 3%i, x = -6%i, x = 6%i]

(%i126)  map(rhs, %);
(%o126)  [1, 8, -3%i, 3%i, -6%i, 6%i]

(%i127)  S: setify(%);
(%o127)  {1, 8, -6%i, -3%i, 3%i, 6%i}
```

Portanto, o conjunto-solução obtido é $S = \{1, 8, -6i, -3i, 3i, 6i\}$.

Exemplo 3.26 Entre os 200 primeiros termos das progressões aritméticas cujos termos gerais são $a_k = 5k - 11$ e $b_k = 3k + 7$, determine quantos elas têm em comum.

Criamos as progressões aritméticas como sendo as listas L1 e L2, transformamos as listas em conjuntos S1 e S2 e calculamos a cardinalidade da sua interseção.

```
(%i128)  L1: makelist(5*k - 11, k, 1, 200)$
(%i129)  L2: makelist(3*k + 7, k, 1, 200)$
(%i130)  S1: setify(L1)$
(%i131)  S2: setify(L2)$
(%i132)  S: intersection(S1, S2);
(%o11)   {19, 34, 49, 64, 79, 94, 109, 124, 139, 154, 169, 184, 199, 214, 229,
244, 259, 274, 289, 304, 319, 334, 349, 364, 379, 394, 409, 424, 439, 454, 469,
484, 499, 514, 529, 544, 559, 574, 589, 604}
```

```
(%i133)  cardinality(S);
(%o133)  40
```

Concluímos que as progressões aritméticas dadas têm 40 termos em comum entre seus 200 primeiros termos.

3.5 Relações e classes de equivalência

Uma relação em um conjunto A é qualquer subconjunto de $A \times A$. Para definir uma relação R em um conjunto A podemos usar um comando do tipo `R(x, y) := is(...)` e definir a relação entre os elementos x e y do conjunto A como parâmetro do comando `is`. Se for assim definida, então x estará relacionado com y, em símbolos xRy, quando `R(x, y)` for verdadeira.

Por exemplo, a relação de igualdade em um conjunto pode ser definida com `R(x, y) := is(x = y)`. Nesse caso, `R(4, 4)` retorna um valor `true` (significando que `4 R 4`, ou seja, $4 = 4$) e `R(1, 3)` retorna `false` (significando que $1 \not R 3$, ou seja, $1 \neq 3$).

O *Maxima* calcula as classes de equivalência de uma relação R em um conjunto A através de um comando **equiv_classes(A, R)**. Se R é uma relação de equivalência, o conjunto das classes é conhecido pelo nome de *conjunto-quociente de A por R* e costuma ser denotado por A/R.

Exemplo 3.27 Definir a seguinte relação: $xRy \Leftrightarrow x - y$ é múltiplo de 5. Para isso, podemos usar o comando `remainder(m, n)` (ou `mod(m, n)`) que calcula o resto da divisão de m por n. Sendo $A = \{1, 2, 3, 4, 9, 11, 12, 14, 19, 20\}$, verificar se $4R19$, $11R1$ e $9R20$ e calcular as classes de equivalência de R em A.

```
(%i134)  A: {1, 2, 3, 4, 9, 11, 12, 14, 19, 20};
(%o134)  {1, 2, 3, 4, 9, 11, 12, 14, 19, 20}

(%i135)  R(x, y) := is(remainder(x - y, 5) = 0);
(%o135)  R(x, y) := is(remainder(x - y, 5) = 0)

(%i136)  R(4, 19);
(%o136)  true

(%i137)  R(11, 1);
(%o137)  true

(%i138)  R(9, 20);
(%o138)  true
```

```
(%i139)  equiv_classes(A, R);
(%o139)  {{1, 11}, {2, 12}, {3}, {4, 9, 14, 19}, {20}}
```

De acordo com as respostas do programa, temos que $4\,R\,19$, $11\,R\,1$, $9\ \not R\,20$ e o conjunto das classes de equivalência da relação R no conjunto A é dado por $A/R = \{\{1, 11\}, \{2, 12\}, \{3\}, \{4, 9, 14, 19\}, \{20\}\}$.

Exemplo 3.28 Sejam $B = \{\,(1, 2), (3, 4), (2, 2), (\sqrt{5}, 0), (0, 5), (-2, -2), (2, -2), (\sqrt{2}, \sqrt{3})\}$ e a seguinte relação S definida em B:

$$(x_1, x_2)\, S\, (y_1, y_2) \;\Leftrightarrow\; x_1^2 + x_2^2 = y_1^2 + y_2^2.$$

Verifique se $(0, 5)$ está relacionado com $(3, 4)$ e determine o conjunto das classes de equivalência.

```
(%i140)  B: { [1, 2], [3, 4], [2, 2], [sqrt(5), 0], [0, 5], [-2, -2], [2, -2],
             [sqrt(2), sqrt(3)] };
(%o140)  {[−2, −2], [0, 5], [1, 2], [2, −2], [2, 2], [3, 4], [√2, √3], [√5, 0]}

(%i141)  S(x, y) := is(x[1]^2 + x[2]^2 = y[1]^2 + y[2]^2);
(%o141)  S(x, y) := is(x[1]^2 + x[2]^2 = y[1]^2 + y[2]^2);

(%i142)  S([0, 5], [3, 4]);
(%o142)  true

(%i143)  equiv_classes(B, S);
(%o143)  {{[−2, −2], [2, −2], [2, 2]}, {[0, 5], [3, 4]}, {[1, 2], [√2, √3], [√5, 0]}}
```

Capítulo 4

Operações com matrizes

O *Maxima* possui uma grande quantidade de recursos para a manipulação de matrizes. Esses recursos vão desde os mais básicos (como adição e multiplicação de matrizes, cálculo de determinantes) aos mais avançados (autovalores, autovetores, formas canônicas).

4.1 Introduzindo uma matriz

Uma matriz pode ser definida de várias maneiras. Os principais modos de definição são através dos seguintes comandos:

- **matrix([linha1], [linha2], ...)** Define uma matriz cujas linhas são dadas. Equivale ao "`Introduzir matriz ...`" do menu "`Álgebra`".
- **genmatrix(função, N. linhas, N. colunas)** Gera uma matriz de ordem especificada e termo geral dado. Equivale ao "`Gerar matriz ...`" do menu "`Álgebra`".
- **entermatrix(N. linhas, N. colunas)** Entra com uma matriz elemento por elemento, à medida que vão sendo solicitados.

Exemplo 4.1 Definir uma matriz A fornecendo as linhas na forma de listas de mesmo comprimento.

```
(%i1)  A: matrix( [3, 2, -1], [8, 0, 7], [a, b, c], [0, 0, x] );
```

$$\text{(\%o1)} \quad \begin{bmatrix} 3 & 2 & -1 \\ 8 & 0 & 7 \\ a & b & c \\ 0 & 0 & x \end{bmatrix}$$

Exemplo 4.2 Definir uma matriz B de ordem 3×4 cujo termo geral seja $b_{ij} = \sqrt{i} + \frac{1}{j}$.

```
(%i2)  b[i, j] := sqrt(i) + 1/j;
```
(%o2) $b_{i,j} := \sqrt{i} + \frac{1}{j}$

```
(%i3)  B: genmatrix(b, 3, 4);
```
(%o3)
$$\begin{bmatrix} 2 & \sqrt{2}^{\frac{3}{2}} & \sqrt{2}^{\frac{4}{3}} & \sqrt{2}^{\frac{5}{4}} \\ \sqrt{2}+1 & \sqrt{2}+\frac{1}{2} & \sqrt{2}+\frac{1}{3} & \sqrt{2}+\frac{1}{4} \\ \sqrt{3}+1 & \sqrt{3}+\frac{1}{2} & \sqrt{3}+\frac{1}{3} & \sqrt{3}+\frac{1}{4} \end{bmatrix}$$

Note que a função do termo geral é definida usando-se colchetes e ":=".

Exemplo 4.3 A função `random(n)` retorna um inteiro pseudoaleatório de 0 a $n-1$. Usando essa função, gere uma matriz M de ordem 5×5 aleatoriamente.

```
(%i4)  a[i, j] := random(100);
```
(%o4) $a_{i,j} := random(100)$

```
(%i5)  M: genmatrix(a, 5, 5);
```
(%o5)
$$\begin{bmatrix} 26 & 36 & 53 & 60 & 71 \\ 12 & 9 & 29 & 18 & 26 \\ 95 & 53 & 93 & 87 & 28 \\ 44 & 6 & 95 & 20 & 97 \\ 76 & 45 & 34 & 7 & 2 \end{bmatrix}$$

Exemplo 4.4 Definir uma matriz M de ordem 2×3, elemento por elemento.

```
(%i5)  M: entermatrix(2, 3);
Row 1 Column 1:    8;
Row 1 Column 2:    9;
Row 1 Column 3:    -11;
Row 2 Column 1:    1;
Row 2 Column 2:    0;
Row 2 Column 3:    20;
Matrix entered.
```
(%o5)
$$\begin{bmatrix} 8 & 9 & -11 \\ 1 & 0 & 20 \end{bmatrix}$$

4.2 Matrizes especiais

O *Maxima* possui vários tipos pré-definidos de matrizes. Entre eles, citamos os seguintes:

- **ident(n)** Matriz identidade $n \times n$.
- **zeromatrix(m, n)** Matriz nula $m \times n$.
- **ematrix(m, n, k, i, j)** Matriz $m \times n$ com todos os elementos nulos, exceto o da linha i e coluna j que é igual a k.
- **diagmatrix(n, k)** Matriz $n \times n$ com elementos da diagonal iguais a k.
- **diag_matrix($x_1, x_2, \ldots, x_n$)** Matriz cujos elementos da diagonal são dados e os outros elementos são nulos.
- **vandermonde_matrix($[x_1, x_2, \ldots, x_n]$)** Matriz de Vandermonde definida pela lista de elementos dados.
- **hilbert_matrix(n)** Matriz de Hilbert de ordem $n \times n$, $h_{ij} = \frac{1}{i+j-1}$.

Exemplo 4.5 Definimos vários tipos de matrizes especiais, todas de ordem 4×4.

```
(%i6)  I: ident(4);
```

$$\text{(\%o6)} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
(%i7)  Z: zeromatrix(4, 4);
```

$$\text{(\%o7)} \quad \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
(%i8)  A: diagmatrix(4, -3);
```

$$\text{(\%o8)} \quad \begin{bmatrix} -3 & 0 & 0 & 0 \\ 0 & -3 & 0 & 0 \\ 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & -3 \end{bmatrix}$$

```
(%i9)  B: diag_matrix(5, -11, 13, sqrt(2));
```

$$\text{(\%o10)} \quad \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & -11 & 0 & 0 \\ 0 & 0 & 13 & 0 \\ 0 & 0 & 0 & \sqrt{2} \end{bmatrix}$$

```
(%i10)  C: ematrix(4, 4, -23, 3, 3);
```

```
(%o10)
```
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -23 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
(%i11) D: ematrix(4, 4, -23, 2, 1);
```
```
(%o11)
```
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ -23 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
(%i12) V: vandermonde_matrix([a, b, 2, 3]);
```
```
(%o12)
```
$$\begin{bmatrix} 1 & a & a^2 & a^3 \\ 1 & b & b^2 & b^3 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \end{bmatrix}$$

```
(%i13) H: hilbert_matrix(4);
```
```
(%o13)
```
$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{bmatrix}$$

Exemplo 4.6 O comando `sum(f(k), k, m, n)` calcula o somatório da expressão `f(k)`, com k variando de m até n. Usando os comandos `sum(...)` e `ematrix(...)`, defina uma matriz M de ordem 3×5 que tenha todos os seus elementos iguais a 11.

```
(%i14) M: sum(sum(ematrix(3, 5, 11, i, j), i, 1, 3), j, 1, 5);
```
```
(%o14)
```
$$\begin{bmatrix} 11 & 11 & 11 & 11 & 11 \\ 11 & 11 & 11 & 11 & 11 \\ 11 & 11 & 11 & 11 & 11 \end{bmatrix}$$

De um modo geral,

```
sum(sum(ematrix(m, n, k, i, j), i, 1, m), j, 1, n)
```

gera uma matriz de ordem $m \times n$ com todos os elementos iguais a k.

Exemplo 4.7 Usando `sum(...)`, `random(...)` e `ematrix(...)`, defina uma matriz P de ordem 3×7 cujos elementos sejam gerados aleatoriamente.

```
(%i15)  P: sum(sum(ematrix(3, 7, random(50), i, j), i, 1, 3), j, 1, 7);
```

$$(\%o15)\quad \begin{bmatrix} 22 & 13 & 18 & 44 & 31 & 30 & 38 \\ 25 & 0 & 3 & 28 & 24 & 28 & 35 \\ 28 & 33 & 33 & 8 & 12 & 15 & 3 \end{bmatrix}$$

De um modo geral,

```
sum(sum(ematrix(m, n, random(p), i, j), i, 1, m), j, 1, n)
```

gera uma matriz de ordem $m \times n$ com todos os seus elementos inteiros de 0 a p-1 gerados aleatoriamente.

4.3 Extraindo linhas, colunas e elementos de uma matriz

Dada uma matriz M, os seguintes elementos podem ser calculados:

- **M[i, j]** ou **M[i][j]** Elemento da linha i e coluna j de M.
- **M[k]** ou **row(M, k)** Linha k da matriz M.
- **col(M, j)** Coluna j da matriz M.
- **minor(M, i, j)** Matriz obtida pela retirada da linha i e da coluna j de M.

Exemplo 4.8 Dada uma matriz M, mostrar os valores de alguns elementos, linhas, colunas e alguns menores.

```
(%i16)  m[i, j] := random(20)$   M: genmatrix(m, 4, 4);
```

$$(\%o16)\quad \begin{bmatrix} 12 & 2 & 14 & 5 \\ 4 & 11 & 9 & 5 \\ 18 & 3 & 15 & 5 \\ 0 & 6 & 19 & 0 \end{bmatrix}$$

```
(%i17)  M[3, 1];
(%o17)  18

(%i18)  M[1, 3];
(%o18)  14

(%i19)  M[1,1]+M[2,2]+M[3,3]+M[4,4];
(%o20)  38

(%i21)  M[1]
```

```
(%o22)  [12, 2, 14, 5]

(%i23)  M[2]
(%o23)  [4, 11, 9, 5]

(%i24)  col(M, 3);
(%o24)  [ 14 ]
        [  9 ]
        [ 15 ]
        [ 19 ]

(%i25)  minor(M, 2, 2);
(%o25)  [ 12  14  5 ]
        [ 18  15  5 ]
        [  0  19  0 ]

(%i26)  minor(M, 4, 3);
(%o26)  [ 12   2  5 ]
        [  4  11  5 ]
        [ 18   3  5 ]
```

4.4 Operações básicas

Dadas M e N matrizes, são definidas as seguintes operações e funções:

- **matrix_size(M)** Ordem da matriz M no formato de lista $[m, n]$.
- **M + N** Adição de matrizes de mesma ordem.
- **M - N** Subtração de matrizes de mesma ordem.
- **k*M** Multiplicação de uma matriz M por uma constante k.
- **M.N** Multiplicação de matrizes, se o número de colunas de M for igual ao número de linhas de N.
- **transpose(M)** Matriz transposta de M.
- **invert(M)** Matriz inversa de M, se M for quadrada.
- **M^^n** Matriz M elevada à enésima potência. M^^0 é a matriz identidade e M^^-1 é a matriz inversa de M.
- **echelon(M)** Escalonamento da matriz M.
- **rank(M)** Posto da matriz M

- **mat_trace(M)** Traço da matriz quadrada M, ou seja, soma dos elementos da diagonal principal.

Observação: Além das operações básicas usuais, o *Maxima* possui definidas algumas operações que não são usuais tais como a multiplicação M*N e a potenciação M^n. M*N é a multiplicação termo a termo de M por N e M^n é a matriz das enésimas potências dos elementos de M. Por exemplo, se $M = \begin{bmatrix} x & z \\ r & s \end{bmatrix}$ e $N = \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}$, então $M * N = N * M = \begin{bmatrix} 3x & 4z \\ 5r & 6s \end{bmatrix}$ e $M\text{^}2 = \begin{bmatrix} x^2 & z^2 \\ r^2 & s^2 \end{bmatrix}$. Deve-se ter cuidado para não confundir M.N com M*N, nem M^^n com M^n.

Exemplo 4.9 Dadas as matrizes M e N, calcular $M + N$, $M - N$, MN, NM, $2M$, M^0, M^{-1}, M^2, M^t e o traço de M.

```
(%i27) M: matrix([1,1,1], [3,5,4], [-2,-1,-2]);
```

$$(\%o27)\quad \begin{bmatrix} 1 & 1 & 1 \\ 3 & 5 & 4 \\ -2 & -1 & -2 \end{bmatrix}$$

```
(%i28) N: matrix([1,2,3], [1,3,4], [1,3,5]);
```

$$(\%o28)\quad \begin{bmatrix} 1 & 2 & 3 \\ 1 & 3 & 4 \\ 1 & 3 & 5 \end{bmatrix}$$

```
(%i29) M + N;
```

$$(\%o29)\quad \begin{bmatrix} 2 & 3 & 4 \\ 4 & 8 & 8 \\ -1 & 2 & 3 \end{bmatrix}$$

```
(%i30) M - N;
```

$$(\%o30)\quad \begin{bmatrix} 0 & -1 & -2 \\ 2 & 2 & 0 \\ -3 & -4 & -7 \end{bmatrix}$$

```
(%i31) M.N;
```

$$(\%o31)\quad \begin{bmatrix} 3 & 8 & 12 \\ 12 & 33 & 49 \\ -5, -13, -20 & & \end{bmatrix}$$

```
(%i32) N.M;
```

```
(%o32)  [ 1   8   3 ]
        [ 2  12   5 ]
        [ 0  11   3 ]

(%i33)  2*M;
        [  2   2   2 ]
(%o33)  [  6  10   8 ]
        [ -4  -2  -4 ]

(%i34)  M^^0;
        [ 1  0  0 ]
(%o34)  [ 0  1  0 ]
        [ 0  0  1 ]

(%i35)  M^^-1;
        [  6  -1   1 ]
(%o35)  [  2   0   1 ]
        [ -7   1  -2 ]

(%i36)  invert(M);
        [  6  -1   1 ]
(%o36)  [  2   0   1 ]
        [ -7   1  -2 ]

(%i37)  M^^2;
        [  2   5   3 ]
(%o37)  [ 10  24  15 ]
        [ -1  -5  -2 ]

(%i38)  transpose(M);
        [ 1  3  -2 ]
(%o38)  [ 1  5  -1 ]
        [ 1  4  -2 ]

(%i39)  mat_trace(M);
(%o39)  4
```

Exemplo 4.10 Dada uma matriz M gerada aleatoriamente e de ordem 4×4, escaloná-la de duas maneiras: diretamente, com o comando `echelon(M)`, e também um escalonamento passo a passo, usando operações elementares nas linhas.

```
(%i40)  M: sum(sum(ematrix(4, 4, random(20), i, j), i,1,4), j,1,4);
```

(%o40)
$$\begin{bmatrix} 7 & 0 & 16 & 19 \\ 2 & 6 & 7 & 1 \\ 18 & 2 & 14 & 6 \\ 13 & 6 & 6 & 8 \end{bmatrix}$$

(%i41) P: echelon(M);

(%o41)
$$\begin{bmatrix} 1 & 0 & \frac{16}{7} & \frac{19}{7} \\ 0 & 1 & \frac{17}{42} & -\frac{31}{42} \\ 0 & 0 & 1 & \frac{869}{587} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(%i42) M[1] : M[1]/M[1, 1];

(%o42) $[1, 0, \frac{16}{7}, \frac{19}{7}]$

(%i43) M[2] : M[2] - M[2, 1]*M[1];

(%o43) $[0, 6, \frac{17}{7}, -\frac{31}{7}]$

(%i44) M[3] : M[3] - M[3, 1]*M[1];

(%o44) $[0, 2, -\frac{190}{7}, -\frac{300}{7}]$

(%i45) M[4] : M[4] - M[4, 1]*M[1];

(%o45) $[0, 6, -\frac{166}{7}, -\frac{191}{7}]$

(%i46) M[2] : M[2]/M[2, 2];

(%o46) $[0, 1, \frac{17}{42}, -\frac{31}{42}]$

(%i47) M[3] : M[3] - M[3, 2]*M[2];

(%o47) $[0, 0, -\frac{587}{21}, -\frac{869}{21}]$

(%i48) M[4] : M[4] - M[4, 2]*M[2];

(%o48) $[0, 0, -\frac{183}{7}, -\frac{160}{7}]$

(%i49) M[3] : M[3]/M[3, 3];

(%o49) $[0, 0, 1, \frac{869}{587}]$

(%i50) M[4] : M[4] - M[4, 3]*M[3];

(%o50) $[0, 0, 0, \frac{9301}{587}]$

(%i51) M[4] : M[4]/M[4, 4];

(%o51) $[0, 0, 0, 1]$

(%i52) M;

(%o52)
$$\begin{bmatrix} 1 & 0 & \frac{16}{7} & \frac{19}{7} \\ 0 & 1 & \frac{17}{42} & -\frac{31}{42} \\ 0 & 0 & 1 & \frac{869}{587} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(%i53) is(M = P);

(%o53) *true*

Note que o resultado fornecido com o escalonamento passo a passo é o mesmo que se obtém com um comando `echelon(M)`.

4.5 Determinantes

O determinante de uma matriz quadrada M é calculado com um comando **determinant(M)**.

Exemplo 4.11 Calcule o determinante da matriz $M = \begin{bmatrix} a & b & c & d \\ -b & a & -d & c \\ -c & d & a & -b \\ -d & -c & b & a \end{bmatrix}$.

```
(%i54) M: matrix([a, b, c, d], [-b, a, -d, c], [-c, d, a, -b], [-d, -c, b, a]);
```

(%o54) $$\begin{bmatrix} a & b & c & d \\ -b & a & -d & c \\ -c & d & a & -b \\ -d & -c & b & a \end{bmatrix}$$

```
(%i55) determinant(M)$
(%i56) expand(%);
```

(%o56) $d^4 + 2c^2d^2 + 2b^2d^2 + 2a^2d^2 + c^4 + 2b^2c^2 + 2a^2c^2 + b^4 + 2a^2b^2 + a^4$

```
(%i57) factor(%);
```

(%o57) $(d^2 + c^2 + b^2 + a^2)^2$

Exemplo 4.12 Consideremos a matriz A de ordem 5×5 cujo termo geral é $a_{ij} = i - j + 1 + \max(i,j)(\max(i,j) - 1)$. Calcular os determinantes de A e de sua inversa.

```
(%i58) alias(inv, invert, det, determinant);
```

(%o58) $[inv,\ det]$

```
(%i59) a[i, j] := i - j + 1 + max(i, j)*(max(i, j) - 1);
```

(%o59) $a_{i,j} := i - j + 1 + max(i,j) * (max(i,j) - 1)$

```
(%i60) A: genmatrix(a, 5, 5);
```

```
(%o61)
```
$$\begin{bmatrix} 1 & 2 & 5 & 10 & 17 \\ 4 & 3 & 6 & 11 & 18 \\ 9 & 8 & 7 & 12 & 19 \\ 16 & 15 & 14 & 13 & 20 \\ 25 & 24 & 23 & 22 & 21 \end{bmatrix}$$

```
(%i62) B: inv(A);
```
```
(%o62)
```
$$\begin{bmatrix} \frac{-491}{958} & \frac{485}{958} & \frac{1}{479} & \frac{1}{958} & \frac{-21}{958} \\ \frac{485}{958} & \frac{-1443}{1916} & \frac{477}{1916} & \frac{-1}{1916} & \frac{21}{1916} \\ \frac{1}{479} & \frac{477}{1916} & \frac{-799}{1916} & \frac{319}{1916} & \frac{7}{1916} \\ \frac{1}{958} & \frac{-1}{1916} & \frac{319}{1916} & \frac{-559}{1916} & \frac{243}{1916} \\ \frac{27}{958} & \frac{-27}{1916} & \frac{-9}{1916} & \frac{235}{1916} & \frac{-145}{1916} \end{bmatrix}$$

```
(%i63) det(A);
(%o63) 7664
(%i64) det(B);
(%o64) 1/7664
```

Exemplo 4.13 Dada uma matriz A (gerada aleatoriamente), calcule M, a matriz dos cofatores de A, e a transposta de M dividida pelo determinante de A. Por fim, verifique que essa última matriz obtida é a matriz inversa de A.

```
(%i65) a[i, j] := random(15)$  A: genmatrix(a, 5, 5);
```
```
(%o65)
```
$$\begin{bmatrix} 2 & 12 & 14 & 5 & 9 \\ 4 & 1 & 14 & 5 & 4 \\ 3 & 13 & 10 & 5 & 7 \\ 0 & 6 & 14 & 0 & 1 \\ 14 & 4 & 8 & 4 & 6 \end{bmatrix}$$

```
(%i66) m[i, j] := (-1)^(i + j)*determinant(minor(A, i, j));
(%o66) m_{i,j} := (-1)^(i+j) determinant(minor(A, i, j))
(%i67) M: genmatrix(m, 5, 5);
```
```
(%o67)
```
$$\begin{bmatrix} 1856 & 2696 & -392 & 7624 & -10688 \\ 692 & 1420 & -798 & -6224 & 2652 \\ -732 & -4204 & 1162 & -8992 & 8956 \\ -728 & -1176 & -1624 & 3528 & 2296 \\ -2270 & 110 & 35 & 2616 & -1150 \end{bmatrix}$$

```
(%i68) B: 1/determinant(A)*transpose(M);
```

```
(%o68)  [ -232/3437   -173/6874   183/6874   13/491   1135/13748 ]
        [ -337/3437   -355/6874   1051/6874  21/491   -55/13748  ]
        [ 7/491       57/1964     -83/1964   29/491   -5/3928    ]
        [ -953/3437   778/3437    1124/3437  -63/491  -327/3437  ]
        [ 1336/3437   -663/6874   -2239/6874 -41/491  575/13748  ]

(%i69)  A.B;
        [ 1  0  0  0  0 ]
        [ 0  1  0  0  0 ]
(%o69)  [ 0  0  1  0  0 ]
        [ 0  0  0  1  0 ]
        [ 0  0  0  0  1 ]

(%i70)  B.A;
        [ 1  0  0  0  0 ]
        [ 0  1  0  0  0 ]
(%o71)  [ 0  0  1  0  0 ]
        [ 0  0  0  1  0 ]
        [ 0  0  0  0  1 ]

(%i72)  is(B = invert(A));
(%o72)  true
```

Assim, fica mostrado que a matriz B obtida é a inversa de A.

4.6 Polinômio característico, autovalores e autovetores

Dada uma matriz quadrada M, o determinante de $xI - M$ é denominado polinômio característico de M. Suas raízes λ são chamados autovalores e as soluções não nulas v do sistema $Mv = \lambda v$ são chamados autovetores de M. Esses itens são calculados pelo *Maxima* com os seguintes comandos:

- **charpoly(M, x)** Polinômio característico de M na variável x.
- **eigenvalues(M)** Autovalores λ_i e suas respectivas multiplicidades m_i no formato de lista de listas $[[\lambda_1, \lambda_2, \ldots], [m_1, m_2, \ldots]]$.
- **eigenvectors(M)** Autovetores v_i da matriz M no formato de lista de listas $[[[\lambda_1, \lambda_2, \ldots], [m_1, m_2, \ldots]], [[[v_1]], [[v_2]], \ldots]]$.

Exemplo 4.14 Calcular o polinômio característico da matriz $A = \begin{bmatrix} 0 & 0 & 0 & -d \\ 1 & 0 & 0 & -c \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & -a \end{bmatrix}$

e também da sua transposta $B = A^t$.

```
(%i73) A: matrix( [0, 0, 0, -d], [1, 0, 0, -c], [0, 1, 0, -b], [0, 0, 1, -a]);
```

$$(\%o73)\quad \begin{bmatrix} 0 & 0 & 0 & -d \\ 1 & 0 & 0 & -c \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & -a \end{bmatrix}$$

```
(%i74) B: transpose(A);
```

$$(\%o74)\quad \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -d & -c & -b & -a \end{bmatrix}$$

```
(%i75) p: determinant(x*ident(4) - A);
```

$$(\%o75)\quad x(x(x(x+a)+b)+c)+d$$

```
(%i76) p: expand(p);
```

$$(\%o76)\quad x^4 + ax^3 + bx^2 + cx + d$$

```
(%i77) q: charpoly(B, x);
```

$$(\%o77)\quad d - x(-x(b-(-x-a)x)-c)$$

```
(%i78) q: expand(q);
```

$$(\%o78)\quad x^4 + ax^3 + bx^2 + cx + d$$

Obtivemos dessa forma que o polinômio característico de A e da sua transposta é igual a $x^4 + ax^3 + bx^2 + cx + d$.

Exemplo 4.15 Determinar o polinômio característico, os autovalores e autovetores da matriz $M = \begin{bmatrix} -3 & -5 & 10 & -4 \\ -1 & -14 & 21 & -8 \\ -1 & -19 & 30 & -12 \\ -1 & -19 & 29 & -11 \end{bmatrix}$.

```
(%i79) M: matrix([-3,-5,10,-4], [-1,-14,21,-8], [-1,-19,30,-12],
                 [-1,-19,29,-11] );
```

$$(\%o79)\quad \begin{bmatrix} -3 & -5 & 10 & -4 \\ -1 & -14 & 21 & -8 \\ -1 & -19 & 30 & -12 \\ -1 & -19 & 29 & -11 \end{bmatrix}$$

```
(%i80) charpoly(M, x)$
(%i81) expand(%);
```

$$(\%o81)\quad x^4 - 2x^3 - 15x^2 - 4x + 20$$

```
(%i82)  solve(%);
(%o82)  [x = 1, x = 5, x = -2]

(%i83)  eigenvalues(M);
(%o83)  [[1, 5, -2], [1, 1, 2]]

(%i84)  eigenvectors(M);
(%o84)  [[[1, 5, -2], [1, 1, 2]], [[[1, 2, 3, 4]], [[1, 2, 3, 3]], [[1, 1, 1, 1]]]]
```

Observando essas respostas, temos que o polinômio característico de M é $p(x) = x^4 - 2x^3 - 15x^2 - 4x + 20$, os autovalores são 1, 5, -2 com multiplicidades 1, 1, 2, respectivamente, e os autovetores são $(1, 2, 3, 4)$, $(1, 2, 3, 3)$ e $(1, 1, 1, 1)$.

4.7 Polinômio mínimo e forma de Jordan

O pacote `diag` do *Maxima* possui vários comandos que permitem executar diversas atividades, algumas bastante complexas:

- **jordan(M)** calcula uma lista de listas que corresponde à forma canônica de Jordan de uma matriz M.
- **dispJordan(lista)** mostra a lista obtida com o comando jordan(M) no formato de matriz.
- **minimalPoly(lista)** mostra o polinômio mínimo a partir da lista obtida com jordan(M).
- **JF(λ, n)** bloco de Jordan de ordem n associado ao autovalor λ.
- **diag([M1, M2, ...])** matriz de blocos cuja diagonal é formada pelas matrizes M1, M2, ...
- **ModeMatrix(M)** ou **ModeMatrix(M, lista)** matriz P tal que (P^^-1) . M . P é igual à forma de Jordan de M. A lista obtida com jordan(M) pode ser passada como segundo parâmetro para reduzir o tempo gasto nos cálculos.
- **mat_function(f, M)** retorna f(M), o valor da função f aplicada a M.

Exemplo 4.16 Sendo $M = \begin{bmatrix} 184 & 365 & 505 & 599 & 646 \\ -451 & -869 & -1193 & -1408 & -1515 \\ 374 & 712 & 971 & 1137 & 1220 \\ -65 & -121 & -159 & -177 & -189 \\ -31 & -60 & -85 & -104 & -111 \end{bmatrix}$, deter-

mine sua forma de Jordan, seu polinômio mínimo e uma matriz P tal que $P^{-1}MP = J$.

```
(%i85) load(diag)$
(%i86) M: matrix([184,365,505,599,646], [-451,-869,-1193,-1408,-1515],
                 [374,712,971,1137,1220], [-65,-121,-159,-177,-189],
                 [-31,-60,-85,-104,-111]);
```

$$(\%o86)\quad \begin{bmatrix} 184 & 365 & 505 & 599 & 646 \\ -451 & -869 & -1193 & -1408 & -1515 \\ 374 & 712 & 971 & 1137 & 1220 \\ -65 & -121 & -159 & -177 & -189 \\ -31 & -60 & -85 & -104 & -111 \end{bmatrix}$$

```
(%i87) lista: jordan(M);
```

$$(\%o87)\quad [[-7, 2], [4, 3]]$$

```
(%i88) dispJordan(lista);
```

$$(\%o88)\quad \begin{bmatrix} -7 & 1 & 0 & 0 & 0 \\ 0 & -7 & 0 & 0 & 0 \\ 0 & 0 & 4 & 1 & 0 \\ 0 & 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 0 & 4 \end{bmatrix}$$

```
(%i89) minimalPoly(lista);
```

$$(\%o89)\quad (x-4)^3(x+7)^2$$

```
(%i90) P: ModeMatrix(M, lista);
```

$$(\%o90)\quad \begin{bmatrix} \frac{10}{7} & 1 & 10 & 4 & 1 \\ -\frac{8}{7} & 0 & -40 & -6 & 0 \\ \frac{2}{7} & -\frac{5}{7} & 60 & -16 & 0 \\ 0 & \frac{2}{7} & -40 & 44 & -10 \\ 0 & 0 & 10 & -26 & 9 \end{bmatrix}$$

```
(%i91) P^^-1 . M . P;
```

$$(\%o91)\quad \begin{bmatrix} -7 & 1 & 0 & 0 & 0 \\ 0 & -7 & 0 & 0 & 0 \\ 0 & 0 & 4 & 1 & 0 \\ 0 & 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 0 & 4 \end{bmatrix}$$

```
(%i92) is (P^^-1 . M . P = dispJordan(jordan(M)));
(%i92) true
```

Exemplo 4.17 Sendo $M = \begin{bmatrix} 6 & 2 & 1 & 1 \\ 0 & 5 & 1 & 0 \\ 1 & 2 & 8 & 5 \\ -1 & -2 & -3 & 1 \end{bmatrix}$, calcule $E = \exp(M)$ e $L = \log(E)$.

```
(%i93)  load(diag)$
(%i94)  M: matrix([6,2,1,1], [0,5,1,0], [1,2,8,5], [-1,-2,-3,1])
```

$$(\%o94)\quad \begin{bmatrix} 6 & 2 & 1 & 1 \\ 0 & 5 & 1 & 0 \\ 1 & 2 & 8 & 5 \\ -1 & -2 & -3 & 1 \end{bmatrix}$$

```
(%i95)  E: mat_function(exp, M);
```

$$(\%o96)\quad \begin{bmatrix} \frac{17\%e^5}{6} & \frac{11\%e^5}{3} & \frac{7\%e^5}{2} & \frac{10\%e^5}{3} \\ \frac{\%e^5}{3} & \frac{5\%e^5}{3} & 2\%e^5 & \frac{11\%e^5}{6} \\ \frac{\%e^5}{2} & \%e^5 & \frac{5\%e^5}{2} & 3\%e^5 \\ -\%e^5 & -2\%e^5 & -3\%e^5 & -3\%e^5 \end{bmatrix}$$

```
(%i97)  L: mat_function(log, E);
```

$$(\%o97)\quad \begin{bmatrix} 6 & 2 & 1 & 1 \\ 0 & 5 & 1 & 0 \\ 1 & 2 & 8 & 5 \\ -1 & -2 & -3 & 1 \end{bmatrix}$$

Exemplo 4.18 Sejam A, B, C blocos de Jordan de ordem 3 associados aos autovalores α, β e λ, respectivamente. Construir uma matriz M de ordem 9×9 cuja diagonal de blocos é $[A, B, C]$ e calcule seu polinômio mínimo.

```
(%i98)  load(diag)$
(%i99)  A: JF(%alpha, 3);
```

$$(\%o99)\quad \begin{bmatrix} \alpha & 1 & 0 \\ 0 & \alpha & 1 \\ 0 & 0 & \alpha \end{bmatrix}$$

```
(%i100)  B: JF(%beta, 3);
```

$$(\%o100)\quad \begin{bmatrix} \beta & 1 & 0 \\ 0 & \beta & 1 \\ 0 & 0 & \beta \end{bmatrix}$$

```
(%i101)  C: JF(%lambda, 3);
```

$$(\%o101)\quad \begin{bmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{bmatrix}$$

```
(%i102) M: diag([A, B, C]);
```

$$
\begin{bmatrix}
\alpha & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \alpha & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \alpha & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \beta & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \beta & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \beta & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \lambda & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda
\end{bmatrix}
$$

(%o102)

```
(%i103) r: jordan(M);
```

(%o103) $[[\lambda, 3], [\beta, 3], [\alpha, 3]]$

```
(%i104) minimalPoly(r);
```

(%o104) $(x-\alpha)^3(x-\beta)^3(x-\lambda)^3$

4.8 Ortogonalização de Gram-Schmidt

O comando **gramschimdt(...)** do pacote `eigen` obtém uma base ortogonal a partir de uma outra base definida como uma lista de listas ou como uma matriz. Os resultados mostrados normalmente contém fatoração de inteiros.

Exemplo 4.19 Aplicar o processo de ortogonalização à base

$$\{(1,2,3,4),\ (5,4,3,5),\ (1,1,1,1),\ (3,7,8,-19)\}.$$

```
(%i105) load(eigen);
```

(%o105) "C : /PROGRA2/MAXIMA1.2/5.31.2/matrix/eigen.mac"

```
(%i106) A: [[1, 2, 3, 4], [5, 4, 3, 5], [1, 1, 1, 1], [3, 7, 8, -19]];
```

(%o106) $[[1,2,3,4],[5,4,3,5],[1,1,1,1],[3,7,8,-19]]$

```
(%i107) B: gramschmidt(A);
```

(%o108) $[[1,2,3,4],[\frac{2\ 3^2}{5},\frac{2\ 3}{5},-\frac{2\ 3}{5},-\frac{3}{5}],[0,\frac{1}{3^2},\frac{2}{3^2},-\frac{2}{3^2}],[-\frac{1}{2},1,-\frac{1}{2},0]]$

```
(%i109) B: expand(B);
```

(%o109) $[[1,2,3,4],[\frac{18}{5},\frac{6}{5},-\frac{6}{5},-\frac{3}{5}],[0,\frac{1}{9},\frac{2}{9},-\frac{2}{9}],[-\frac{1}{2},1,-\frac{1}{2},0]]$

```
(%i110) B[1].B[2];
```

(%o110) 0

```
(%i111) B[3].B[4];
```

```
(%o111)  0
```

Obtivemos desse modo a base ortogonal $B = \{B_1, B_2, B_3, B_4\} = \{(1, 2, 3, 4), (\frac{18}{5}, \frac{6}{5}, -\frac{6}{5}, -\frac{3}{5}), (0, \frac{1}{9}, \frac{2}{9}, -\frac{2}{9}), (-\frac{1}{2}, 1, -\frac{1}{2}, 0)\}$ e mostramos também que B_1 é ortogonal a B_2 e que B_3 é ortogonal a B_4.

Capítulo 5

Gráficos

Com a ajuda de outros programas que são instalados juntamente com ele, o *Maxima* consegue construir os mais diversos tipos de gráficos. Se não for fornecido outro nome, ele usa como padrão um programa chamado Gnuplot para construir os gráficos.

Os gráficos construídos pelos programas que trabalham em conjunto com o *Maxima* são os mais variados: gráficos de pontos isolados, gráficos de funções $f(x)$, gráficos em coordenadas polares, gráficos definidos por equações paramétricas, gráficos tridimensionais de funções do tipo $z = f(x, y)$ ou de superfícies parametrizadas, além de outros tipos.

5.1 Gráficos planos

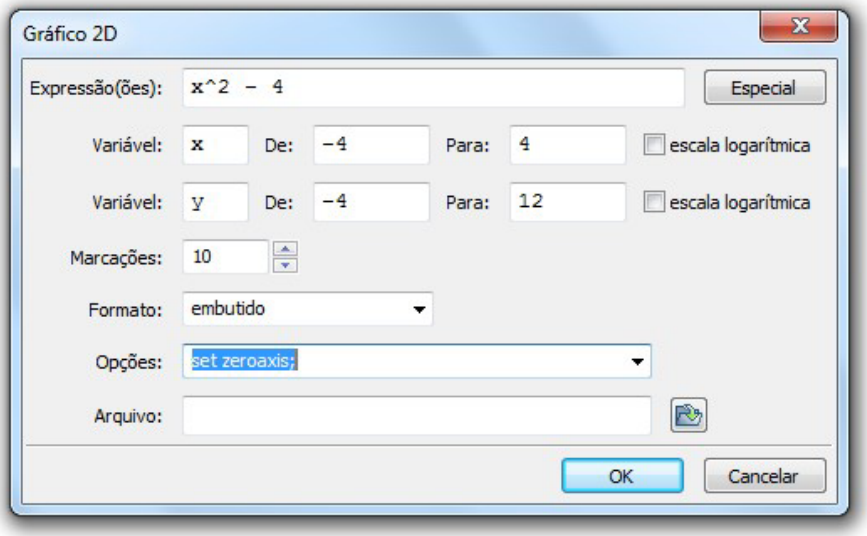
Diversos tipos de gráficos planos podem ser construídos com um comando **plot2d(expressão, domínio, opções, ...)** onde `expressão` pode ser uma expressão algébrica, um nome de função ou uma lista de expressões algébricas ou de pontos isolados, `domínio` é uma lista que contém o nome da variável e seus valores mínimo e máximo, e `opções` são listas que definem a apresentação do gráfico tais como cores, largura, escalas etc.

No lugar do `plot2d(...)` pode ser usado um comando `wxplot2d(...)`. A diferença entre um e o outro é que o `plot2d(...)` faz o gráfico em uma janela exclusiva, enquanto que o `wxplot2d(...)` faz o gráfico "embutido" na janela principal do *Maxima*.

Um gráfico plano também pode ser construído através do "Gráfico 2d" do item "Gráfico" do menu principal. Na janelinha que se abre, podem ser definidos:

- A expressão algébrica que define a função da qual será construída o gráfico;
- Se o gráfico é paramétrico ou discreto;
- Se a escala do gráfico é logarítmica;
- A variação do x;

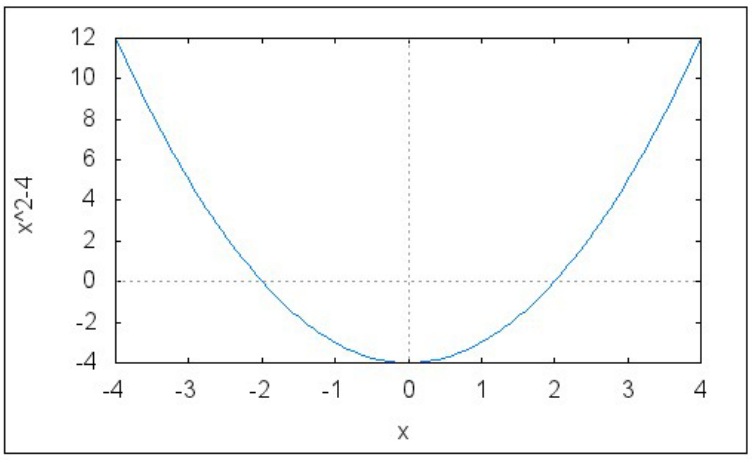
- A variação do y;
- O número de pontos utilizados na construção do gráfico;
- O formato do gráfico (padrão, embutido, gnuplot ou openmath);
- As opções para construção o gráfico: `set zero axis` para mostrar os eixos, `set size ratio 1` para usar a mesma escala nos eixos `set grid` para desenhar uma grade quadriculada como fundo do gráfico, entre outras.



Exemplo 5.1 Construímos como nosso primeiro exemplo o gráfico da função $f(x) = x^2 - 4$, com $-4 \leq x \leq 4$.

```
(%i1)  wxplot2d(x^2 - 4, [x, -4, 4]);
```

(%t1)

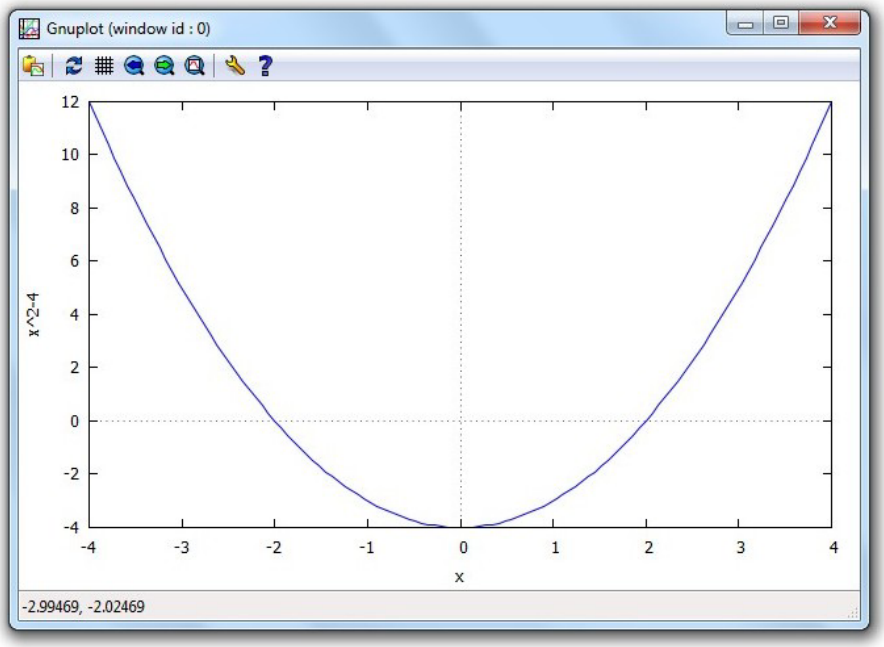


É possível a função ser definida previamente, antes do comando plot2d(...). Sendo assim, o gráfico anterior também pode ser construído da seguinte forma:

```
(%i2)  f(x) := x^2 - 4$
(%i3)  wxplot2d(f(x), [x, -4, 4]);
```

O mesmo gráfico anterior também pode ser construído em uma janela a parte, em formato maior. Para isso, é só retirar o "wx" do comando, digitando apenas `plot2d(...)`. Com esse tipo de gráfico, não podem ser abertas várias janelas simultaneamente, tem que ser construído um gráfico de cada vez.

```
(%i4)  plot2d(x^2 - 4, [x, -4, 4]);
```



5.2 Opções para construção de gráficos

Um gráfico pode ser construído usando-se vários estilos. Podem ser alteradas a cor, a largura do traço e vários outros itens. Para isso, deve ser acrescentado à linha do comando `plot2d(...)` ou `wxplot2d(...)` uma ou várias listas com opções:

- **[x,** x_{min}**,** x_{max}**]** Variação do x, $x_{min} \leq x \leq x_{max}$.
- **[y,** y_{min}**,** y_{max}**]** variação do y, $y_{min} \leq y \leq y_{max}$.
- **[style, [lines, largura, cor]]** Gráfico formado por pequenos segmentos de retas de `largura` e número de `cor` especificados.
- **[color, nome]** Cor utilizada; nome pode ser `blue`, `red`, `green`, `magenta`, `black` ou `cyan`.

- **[style, [points, tamanho, cor, formato]]** gráfico formado por pontos isolados, `tamanho` é um inteiro positivo, `cor` é um inteiro de 1 a 6 e `formato` é um inteiro de 1 a 13 que corresponde ao estilo do ponto.

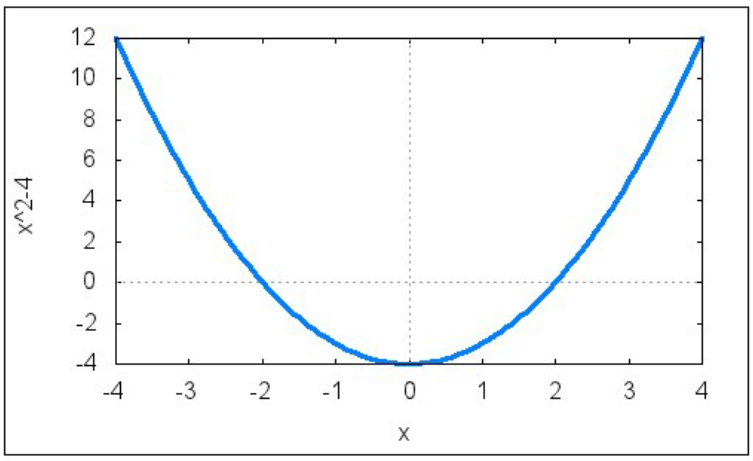
- **[gnuplot_term, tipo]** Tipo de terminal (arquivo) onde o gráfico é gerado; pode ser jpeg, pdf, gif, png, eps, entre outros.

- **[gnuplot_out_file, "arquivo"]** Nome do arquivo onde o gráfico é gravado; deve estar entre aspas; pode incluir um caminho como nos exemplos: `"c:\\temp\\grafico.jpg"`, `"c:\\usuário\\imagens\\textogr.gif"`, `"d:\\begin{exemplo}los\\arquivo.pdf"`.

Exemplo 5.2 Construímos várias vezes o gráfico da função $f(x) = x^2 - 4$, alterando em cada caso a cor do gráfico, a largura do traço ou o estilo do ponto utilizado.

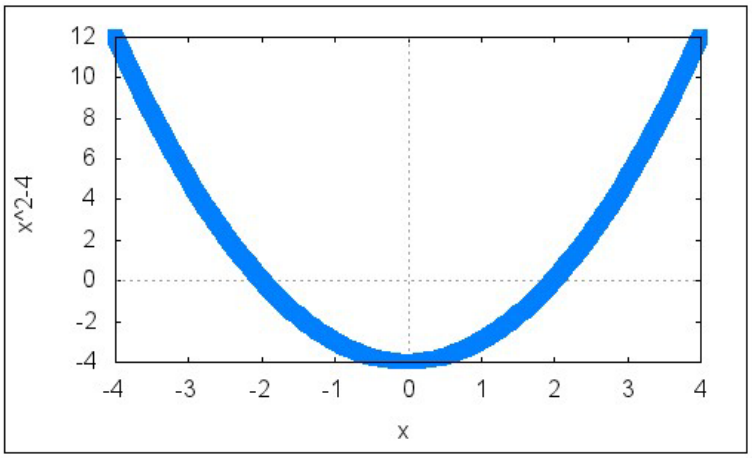
```
(%i5)  wxplot2d(x^2 - 4, [x, -4, 4], [style, [lines, 3]]);
```

(%t5)



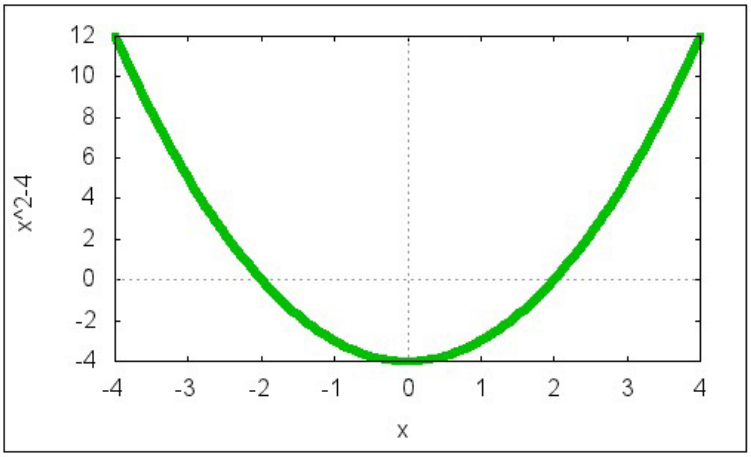

```
(%i6)  wxplot2d(x^2 - 4, [x, -4, 4], [style, [lines, 10, 1]]);
```

(%t6)



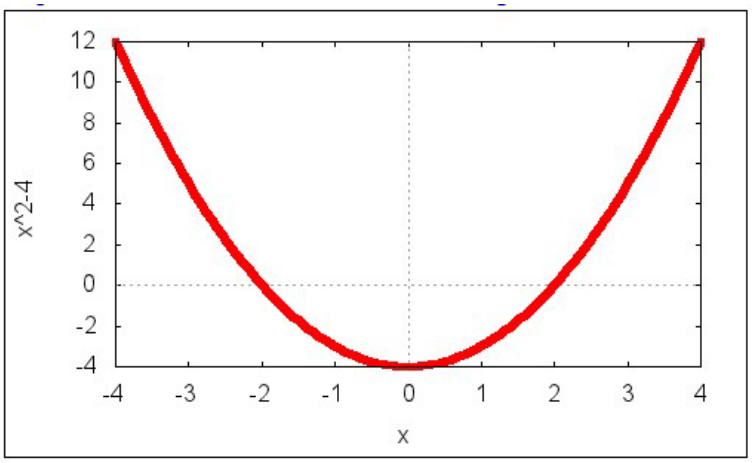

```
(%i7)  wxplot2d(x^2 - 4, [x, -4, 4], [style, [lines, 5]], [color, green]);
```

(%t7)



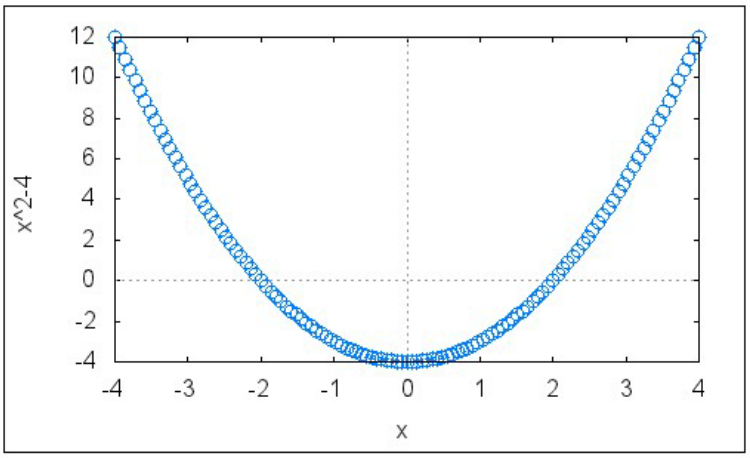
(%i8) wxplot2d(x^2 - 4, [x, -4, 4], [style, [lines, 5]], [color, red]);

(%t8)



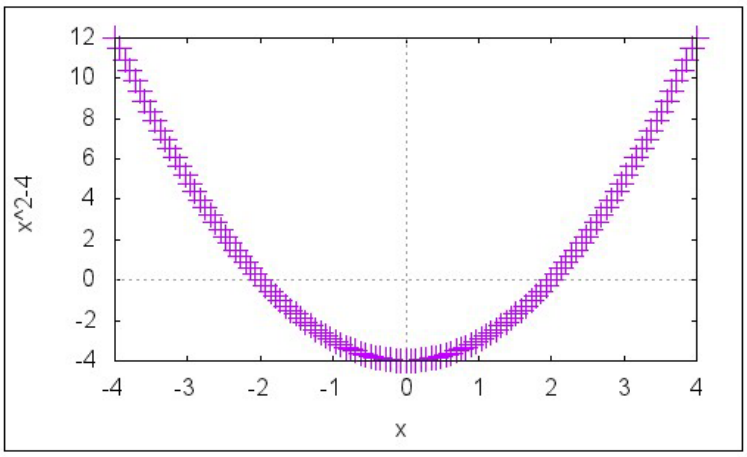
(%i9) wxplot2d(x^2 - 4, [x, -4, 4], [style, [points, 3, 1, 2]]);

(%t9)



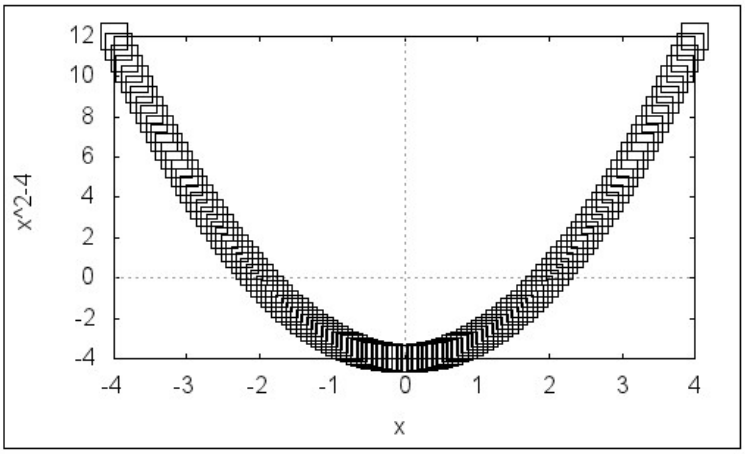
(%i10) wxplot2d(x^2 - 4, [x, -4, 4], [style, [points, 5, 4, 3]]);

(%t10)




```
(%i11)  wxplot2d(x^2 - 4, [x, -4, 4], [style, [points, 6, 5, 7]]);
```

(%t11)



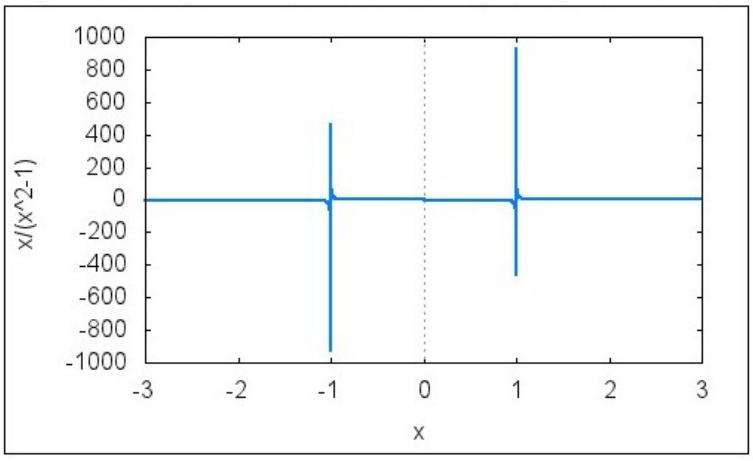
Em vez ser mostrado na tela, o gráfico pode ser gravado em um arquivo. Por exemplo, para gravá-lo em um arquivo de nome EXEMPLO.JPG, basta usar o seguinte comando:

```
(%i12)  wxplot2d(x^2 - 4, [x, -4, 4], [style, [lines, 3]], [gnuplot_term, jpeg],
[gnuplot_out_file, "EXEMPLO.JPG"]);
(%o12)  exemplo.jpg
```

Exemplo 5.3 Construímos o gráfico de $f(x) = \frac{x}{x^2-1}$ com x no intervalo $[-3, 3]$. Inicialmente, não é definida variação para o y e o gráfico fica muito ruim. Depois, o gráfico é construído novamente limitando-se o y ao intervalo $[-10, 10]$ e a apresentação do gráfico melhora significativamente.

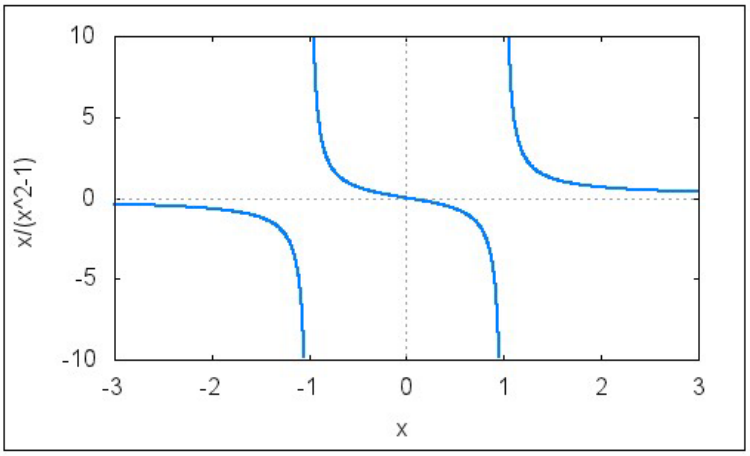
```
(%i13)  wxplot2d(x/(x^2 - 1), [x, -3, 3], [style, [lines, 2]]);
```

(%t13)



(%i14) wxplot2d(x/(x^2 - 1), [x, -3, 3], [y, -10, 10], [style, [lines, 2]]);

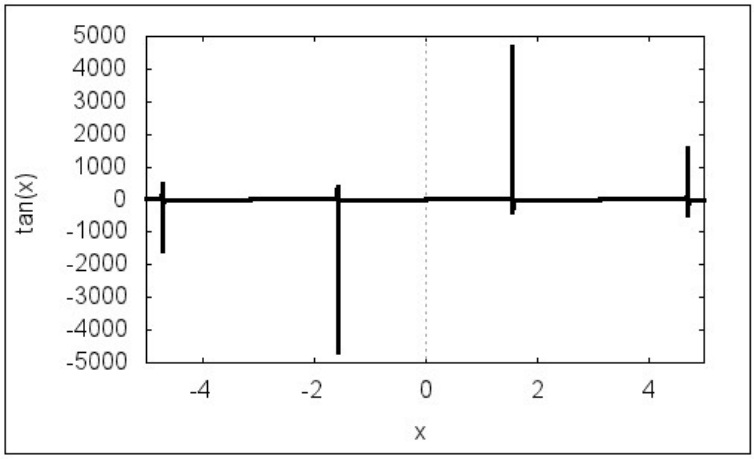
(%t14)



Exemplo 5.4 Construímos o gráfico da tangente trigonométrica no intervalo $[-5, 5]$. O gráfico só fica com uma boa aparência se o y for limitado a algum intervalo.

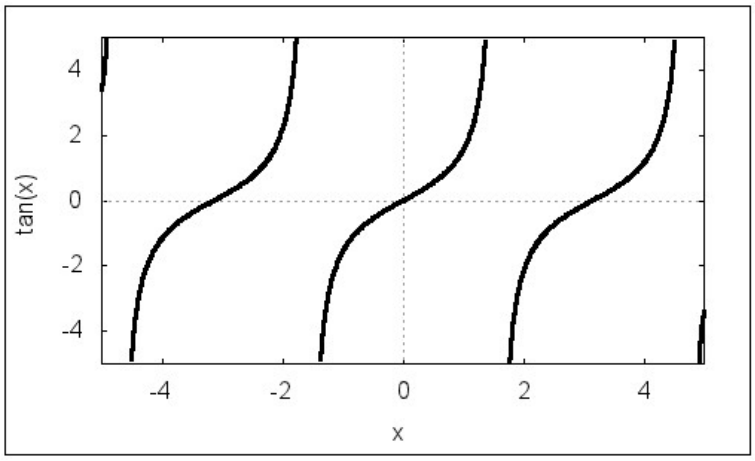
(%i15) wxplot2d(tan(x), [x, -5, 5], [style, [lines, 3]], [color, black]);

(%t15)



(%i16) wxplot2d(tan(x), [x, -5, 5], [y, -5, 5], [style, [lines, 3]], [color, black]);

(%t16)



5.3 Mais opções de construção de gráficos

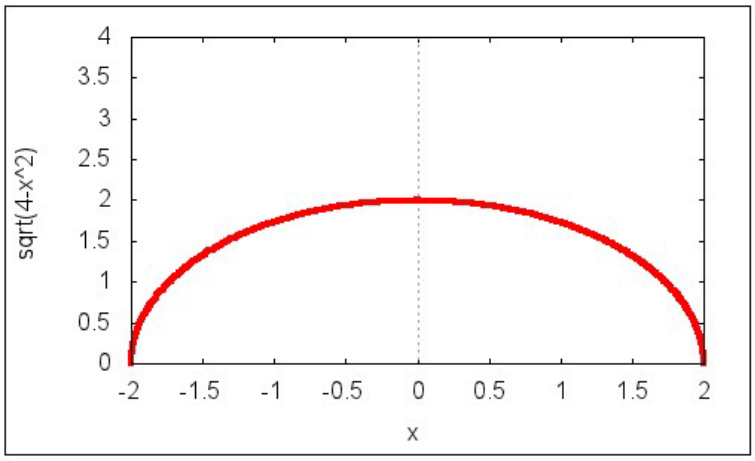
- **[legend, nome]** Acrescenta uma ou várias legendas ao gráfico. Se nome for false, então é mostrado um gráfico sem legendas.
- **[box, teste]** Envolve o gráfico em uma moldura se teste for true, ou não envolve se teste for false.
- **[nticks, n]** Contrói um gráficos usando n pontos no seu traçado se $n \geq 29$.
- **[xlabel, nome]** Rótulo para o eixo dos x.
- **[ylabel, nome]** Rótulo para o eixo dos y
- **[gnuplot_preamble, "set polar")]** Constrói gráfico usando coordenadas polares.
- **[gnuplot_preamble, "set size ratio 1"])** Mesma escala nos eixos.
- **[gnuplot_preamble, "set grid"])** Desenha uma grade como fundo do gráfico.

Exemplo 5.5 Construir o gráfico de $f(x) = \sqrt{4 - x^2}$ com $-2 \leq x \leq 2$. O gráfico é uma semicircunferência que aparece deformada se não for usado uma opção "set size ratio 1". No final, acrescentamos uma grade como desenho de fundo.

```
(%i17)  f(x) := sqrt(4 - x^2);
```
(%o17) $f(x) := \sqrt{4 - x^2}$

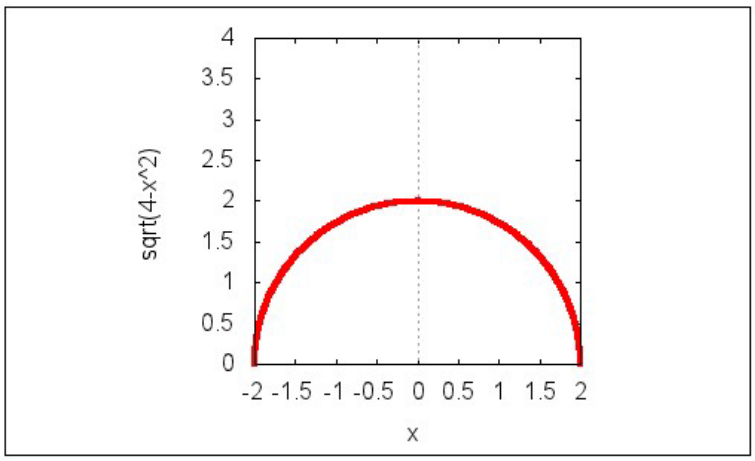
```
(%i18) wxplot2d(f(x), [x, -2, 2], [y, 0, 4], [style, [lines, 4, 2]]);
```

(%t18)



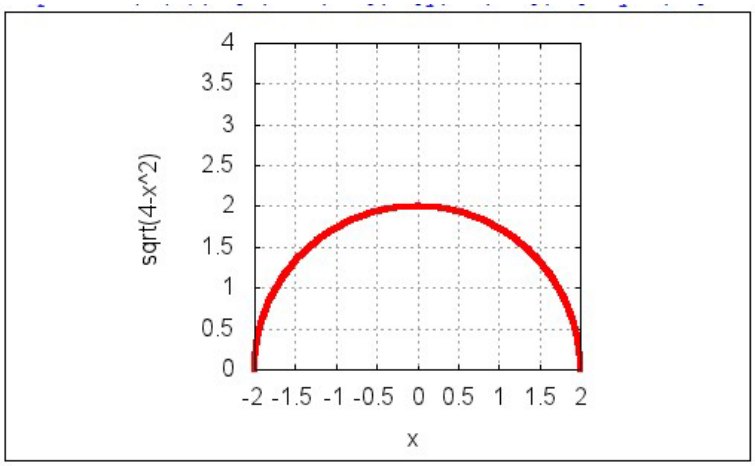

```
(%i19) wxplot2d(f(x), [x, -2, 2], [y, 0, 4], [style, [lines, 4, 2]],
       [gnuplot_preamble, "set size ratio 1"]);
```

(%t19)




```
(%i20) wxplot2d(f(x), [x, -2, 2], [y, 0, 4], [style, [lines, 4, 2]],
       [gnuplot_preamble, "set size ratio 1; set grid"]);
```

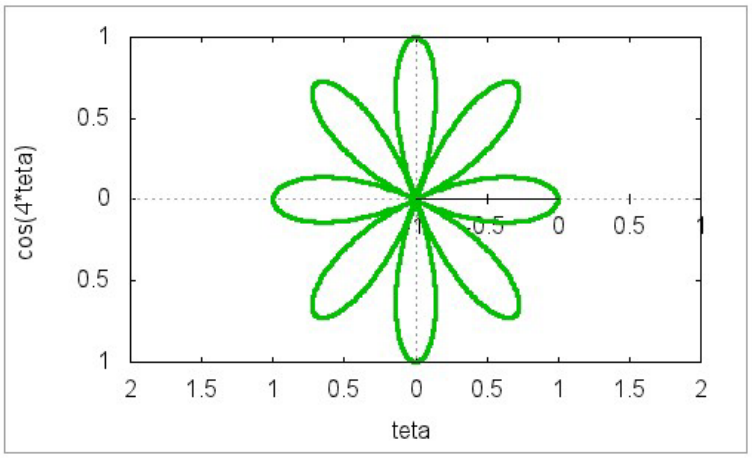
(%t20)



Exemplo 5.6 Usando coordenadas polares, construir o gráfico de $r = \cos(4\theta)$, com $0 \leq \theta \leq 2\pi$.

```
(%i21)  wxplot2d(cos(4*teta), [teta, 0, 2*%pi], [x, -2, 2],
         [style, [lines, 3, 3]], [gnuplot_preamble, "set polar"]);
```

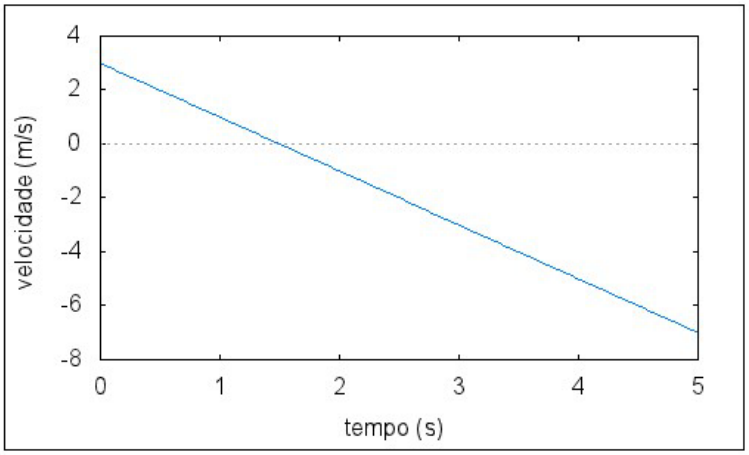
(%t21)



Exemplo 5.7 Construir o gráfico de $v(t) = 3 - 2t$, com $0 \leq t \leq 5$, usando o rótulo *"tempo (s)"* no eixo dos x e *"velocidade (m/s)"* no eixo dos y.

```
(%i22)  v(t) := 3 - 2*t$
(%i23)  wxplot2d(v(t), [t, 0, 5], [xlabel, "tempo (s)"],
                                  [ylabel, "velocidade (m/s)"]);
```

(%to23)



5.4 Construindo vários gráficos simultaneamente

Os gráficos $G_1, G_2, \ldots, G_n$ podem ser construídos simultaneamente em um único sistema de coordenadas cartesianas. Para isso, basta colocá-los no formato de lista no comando plot2d ou wxplot2d. Nesse caso, diversas outras opções podem aparecer também no formato de lista:

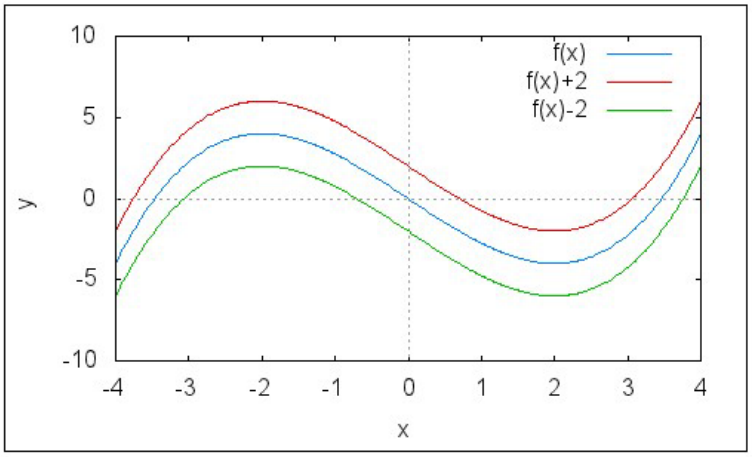
plot2d($[G_1, G_2, \ldots, G_n]$, [style, estilo$_1$, estilo$_2$, $\ldots$, estilo$_n$], [color, cor$_1$, cor$_2$, $\ldots$, cor$_n$], [legend, legenda$_1$, legenda$_2$, $\ldots$, legenda$_n$], $\ldots$).

Exemplo 5.8 Sendo $f(x) = \frac{x^3}{3} - 4x$, construir os gráficos de $f(x)$, $f(x)+2$ e $f(x)-2$ em um mesmo sistema de eixos. Criar uma legenda para os gráficos e, depois, eliminar a moldura que os envolve.

```
(%i24)  f(x) := x^3/3 - 4*x;
(%o24)  f(x) := x^3/3 - 4x
```

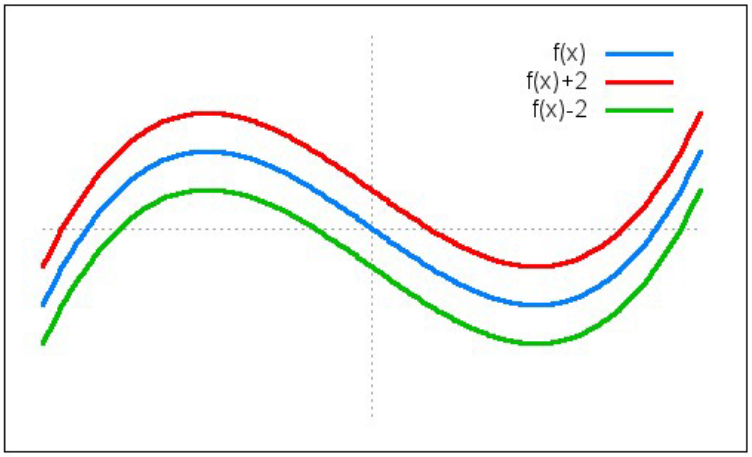
```
(%i25)  wxplot2d([f(x), f(x)+2, f(x)-2], [x, -4, 4], [y, -10, 10],
                 [legend, "f(x)", "f(x)+2", "f(x)-2"] );
```

(%t25)




```
(%i26)  wxplot2d([f(x),f(x)+2,f(x)-2], [x,-4,4], [y,-10,10], [style,[lines,3]],
                 [legend, "f(x)", "f(x)+2", "f(x)-2"], [box, false]);
```

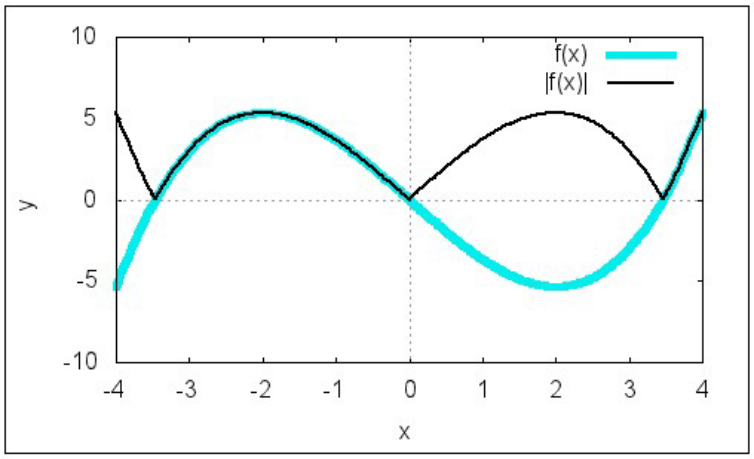
(%t26)



Exemplo 5.9 Ainda com relação a $f(x) = \frac{x^3}{3} - 4x$, construir em um mesmo sistema de eixos os gráficos das funções $f(x)$ e $|f(x)|$. Usar legendas e estilos diferentes para cada gráfico.

```
(%i27)  f(x) := x^3/3 - 4*x$
(%i28)  wxplot2d([f(x), abs(f(x))], [x, -4, 4], [y, -10, 10],
          [legend,"f(x)","|f(x)|"],[style,[lines,5],[lines,2]],[color,cyan,black]);
```

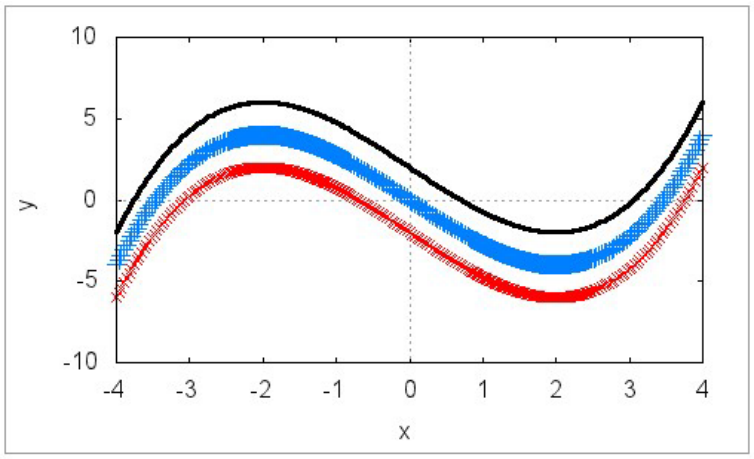
(%t28)



Exemplo 5.10 Construir a mesma lista de gráficos do exemplo anterior sem legendas, usando diferentes estilos para cada gráfico.

```
(%i29)  wxplot2d([f(x), f(x) + 2, f(x) - 2], [x, -4, 4], [y, -10, 10],
                 [style, [points,4,1,3], [lines,3,5], [points,2,2,4]], [legend,false]);
```

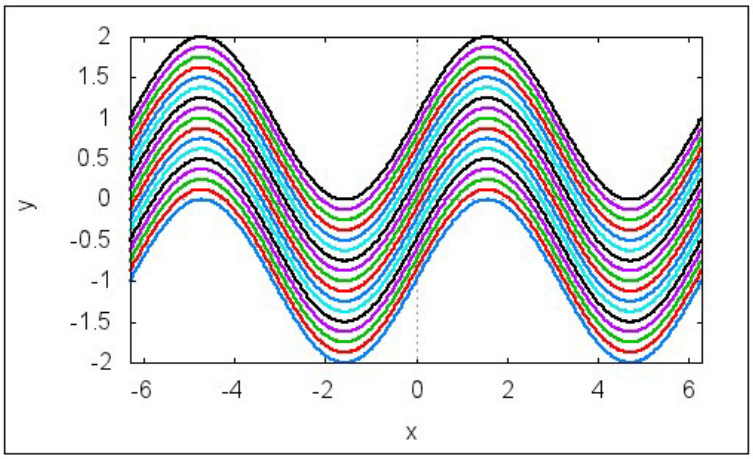
(%t29)



Exemplo 5.11 Construir os gráficos de $\text{sen}(x) + \frac{n}{8}$, com $-8 \leq n \leq 8$, $-2\pi \leq x \leq 2\pi$ e $-2 \leq y \leq 2$. Um comando makelist(...) pode ser usado para criar a lista de gráficos.

```
(%i30)  wxplot2d(makelist(sin(x)+n/8, n, -8, 8), [x, -2*%pi, 2*%pi],
                 [y, -2, 2], [legend, false], [style, [lines, 2]]);
```

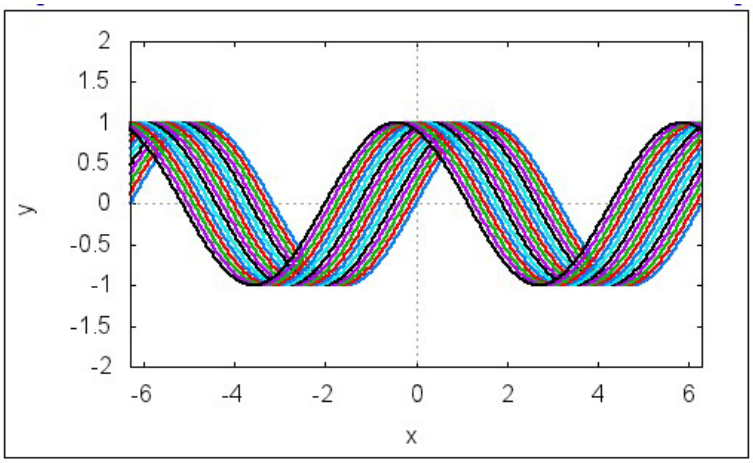
(%t30)



Exemplo 5.12 Construir os gráficos de $\text{sen}(x + \frac{n}{8})$, com $0 \leq n \leq 16$, $-2\pi \leq x \leq 2\pi$ e $-2 \leq y \leq 2$. Um comando makelist(...) pode ser usado para criar a lista de gráficos.

```
(%i31) wxplot2d(makelist(sin(x+n/8), n, 0, 16), [x, -2*%pi, 2*%pi],
                [y, -2, 2], [legend, false], [style, [lines, 2]]);
```

(%t31)



5.5 Gráficos discretos

Um *"gráfico discreto"* é aquele formado por pontos isolados (x_1, y_1), (x_2, y_2), ..., (x_n, y_n). A construção desse tipo de gráfico no *Maxima* é feita com um comando do tipo plot2d([discrete, listagem], ...) ou wxplot2d([discrete, listagem], ...), onde listagem é a lista de pontos $[[x_1, y_1], [x_2, y_2], \ldots, [x_n, y_n]]$ ou as listas de coordenadas $[x_1, x_2, \ldots, x_n]$, $[y_1, y_2, \ldots, y_n]$. O padrão é desenhar uma poligonal que passa pelos pontos dados, a não ser que seja adicionada uma opção do tipo [style, points] ou [style, [points, tamanho, cor, tipo]].

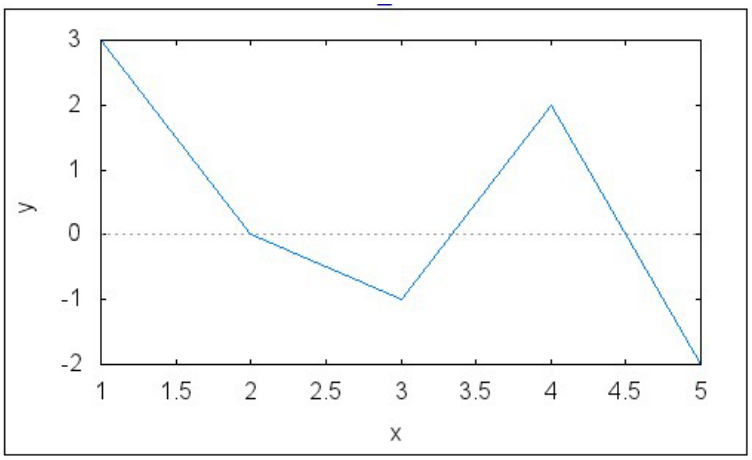
Exemplo 5.13 Fazer um gráfico discreto com os pontos $(1, 3)$, $(2, 0)$, $(3, -1)$,

$(4, 2)$ e $(5, -2)$.

```
(%i32)  lista_pontos: [[1,3],[2,0],[3,-1],[4,2],[5,-2]];
(%o32)  [[1, 3], [2, 0], [3, −1], [4, 2], [5, −2]]
```

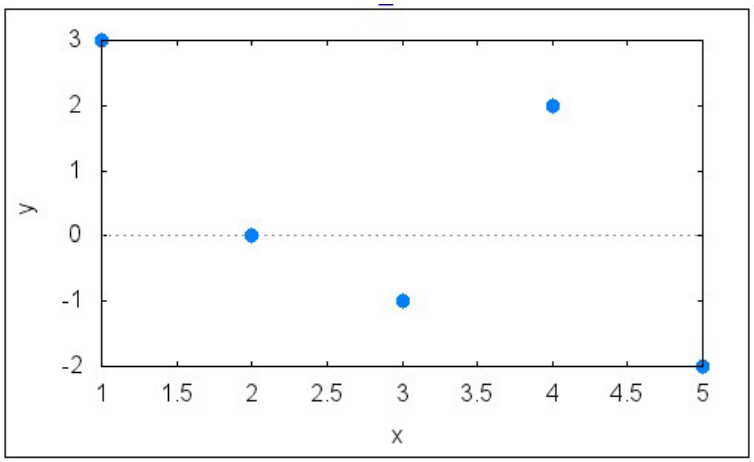
```
(%i33)  wxplot2d([discrete, lista_pontos]);
```

(%t33)



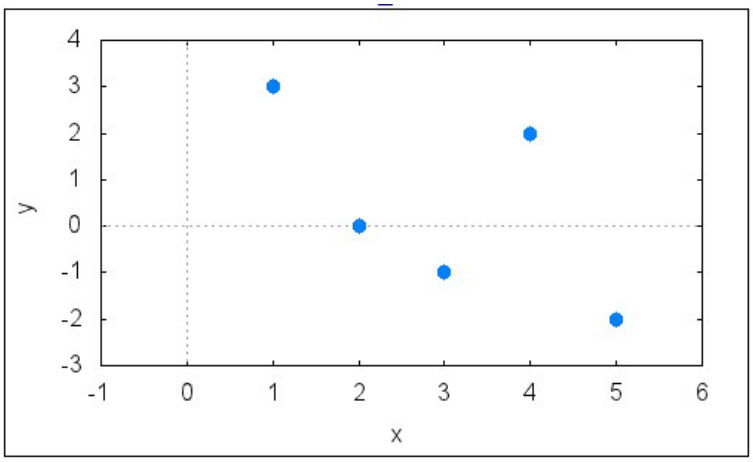

```
(%i34)  wxplot2d([discrete, lista_pontos], [style, points]);
```

(%t34)



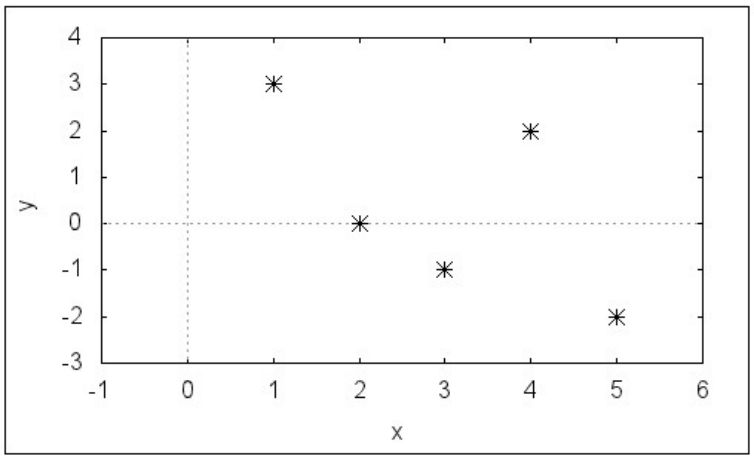

```
(%i35)  wxplot2d([discrete, lista_pontos], [x, -1, 6], [y, -3, 4],
                 [style, points, 3, 1, 1]);
```

(%t35)



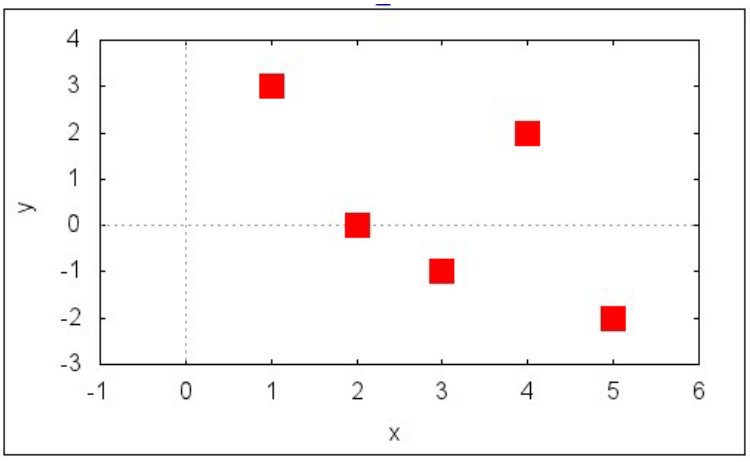

```
(%i36) wxplot2d([discrete, lista_pontos], [x, -1, 6], [y, -3, 4],
                [style, [points, 3, 5, 5]]);
```

(%t36)




```
(%i37) wxplot2d([discrete, lista_pontos], [x, -1, 6], [y, -3, 4],
                [style, [points, 5, 2, 6]]);
```

(%t37)



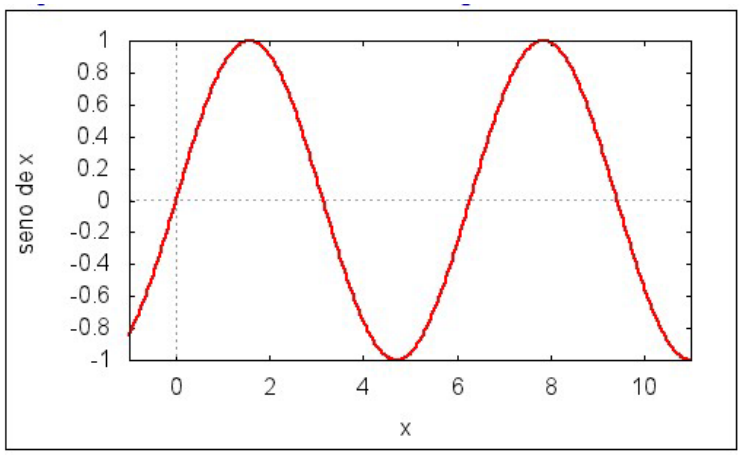
Exemplo 5.14 Construir o gráfico de $f(x) = \text{sen}(x)$ no intervalo $[-1, 11]$ dando destaque aos pontos cujas abscissas são inteiros de 0 a 10.

```
(%i38) f(x) := sin(x)$
```

```
(%i39) L: makelist([n, f(n)], n, 0, 10)$
```

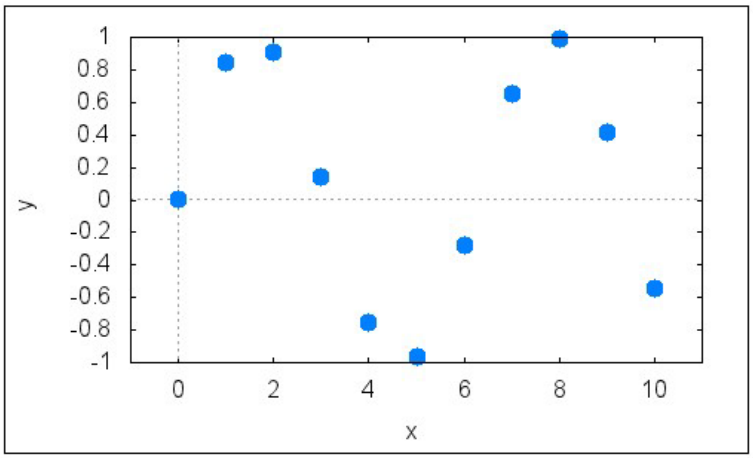
```
(%i40) wxplot2d(f(x), [x, -1, 11], [style, [lines, 2, 2]], [ylabel, "seno de x"]);
```

(%t40)



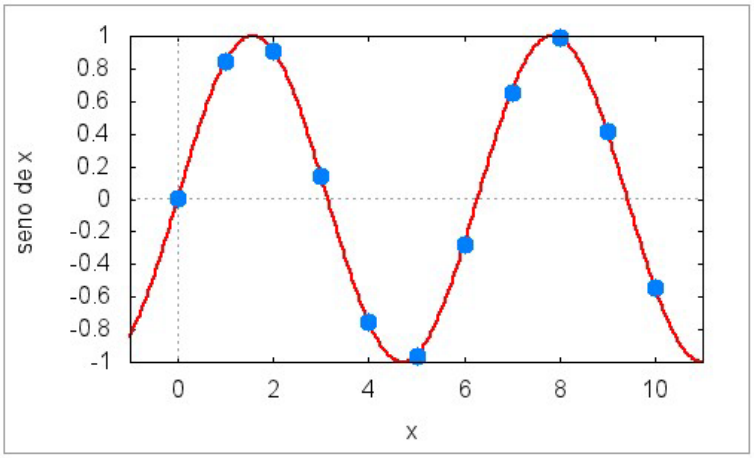

```
(%i41) wxplot2d([discrete, L]], [x, -1, 11], [style, [points, 4, 1, 1]]);
```

(%t41)




```
(%i42) wxplot2d([f(x), [discrete, L]], [x, -1, 11], [style, [lines, 2, 2],
       [points, 4, 1, 1]], [legend, false], [ylabel, "seno de x"]);
```

(%t42)



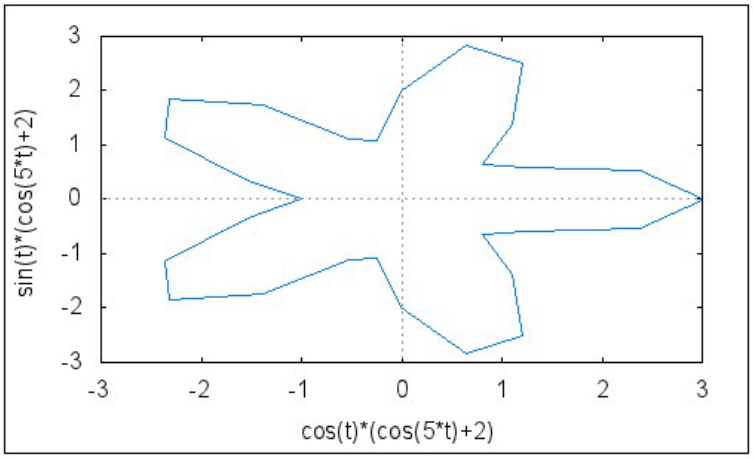
5.6 Gráficos paramétricos

Um gráfico definido por equações paramétricas $x = f(t)$, $y = g(t)$, $a \leq t \leq b$ pode ser construído com um comando wxplot2d([parametric, f(t), g(t)], [t, a, b], ...) ou plot2d([parametric, f(t), g(t)], [t, a, b], ...).

Exemplo 5.15 Construir o gráfico da curva definida por equações paramétricas: $x = (2 + \cos 5t)\cos t$, $y = (2 + \cos 5t)\,\mathrm{sen}\, t$, $0 \leq t \leq 2\pi$. Na primeira tentativa de construção, o gráfico fica muito ruim. Por isso, foi acrescentada uma opção [nticks, 200] para aumentar a quantidade de pontos do gráfico. No final, foi acrescentada uma [gnuplot_preamble, "set size ratio 1"] para uniformizar as escalas dos eixos.

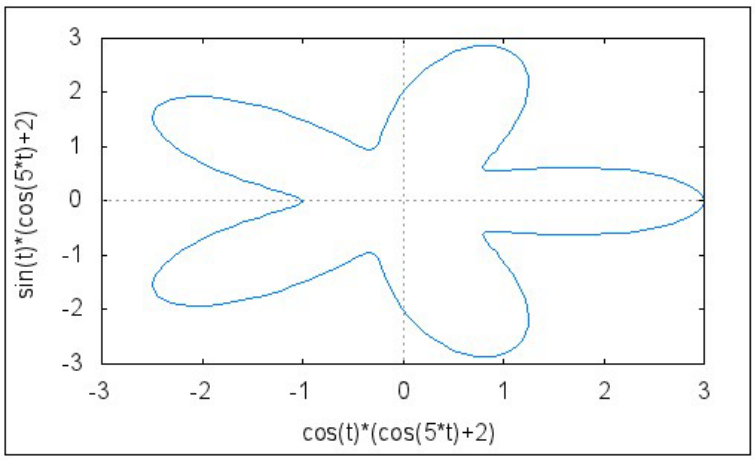
```
(%i43) wxplot2d([parametric, (2+cos(5*t))*cos(t), (2+cos(5*t))*sin(t)],
         [t, 0, 2*%pi]);
```

(%t43)



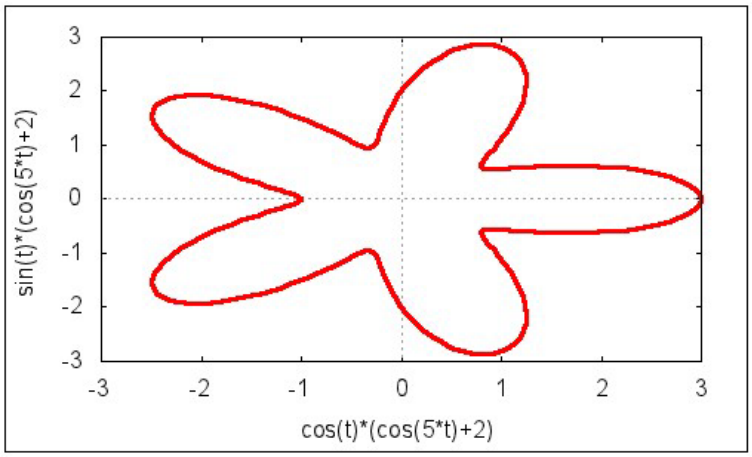

```
(%i44) wxplot2d([parametric, (2+cos(5*t))*cos(t), (2+cos(5*t))*sin(t)],
         [t, 0, 2*%pi], [nticks, 200]);
```

(%t44)



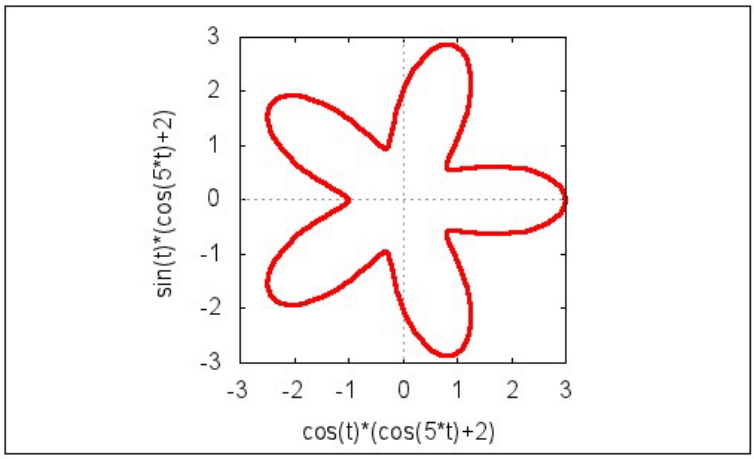

```
(%i45) wxplot2d([parametric, (2+cos(5*t))*cos(t), (2+cos(5*t))*sin(t)],
         [t, 0, 2*%pi], [style, [lines, 3, 2]], [nticks, 200]);
```

(%t45)




```
(%i46) wxplot2d([parametric, (2+cos(5*t))*cos(t), (2+cos(5*t))*sin(t)],
         [t, 0, 2*%pi], [style, [lines, 3, 2]], [nticks, 200],
         [gnuplot_preamble, "set size ratio 1"]);
```

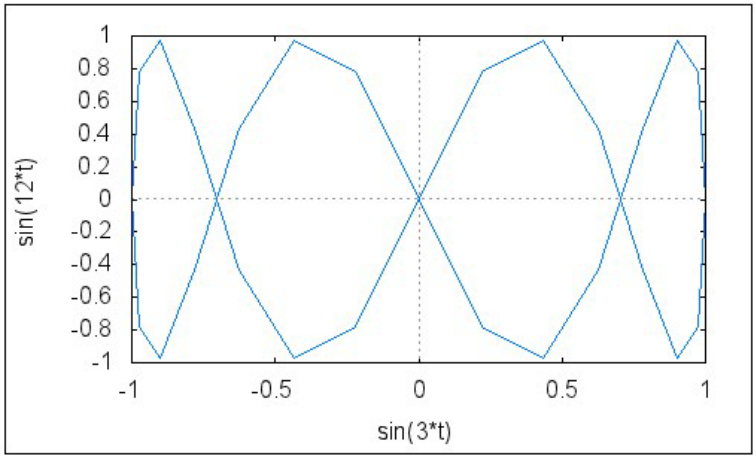
(%t46)



Exemplo 5.16 Construir o gráfico de $x = \operatorname{sen} 3t$, $y = \operatorname{sen} 12t$, $0 \leq t \leq \frac{2\pi}{3}$. Na primeira tentativa de construção, o gráfico é muito ruim. Por isso, o número de pontos foi aumentado com um [nticks, 200]. No final, vários estilos de construção com pontos isolados são utilizados.

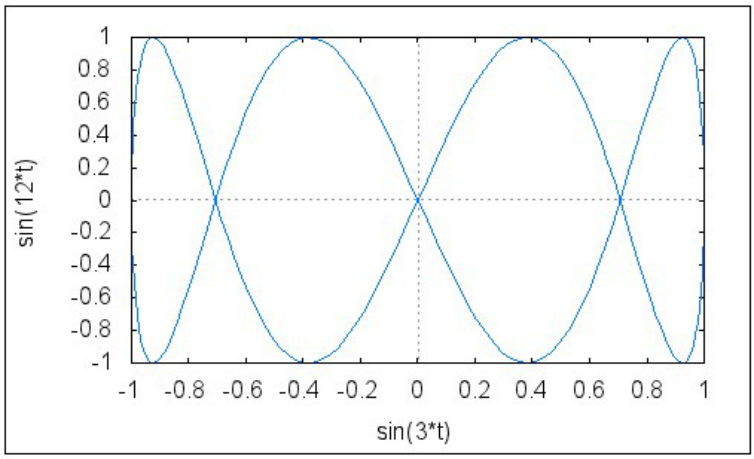
```
(%i47) wxplot2d([parametric, sin(3*t), sin(12*t)], [t, 0, 2*%pi/3]);
```

(%t47)



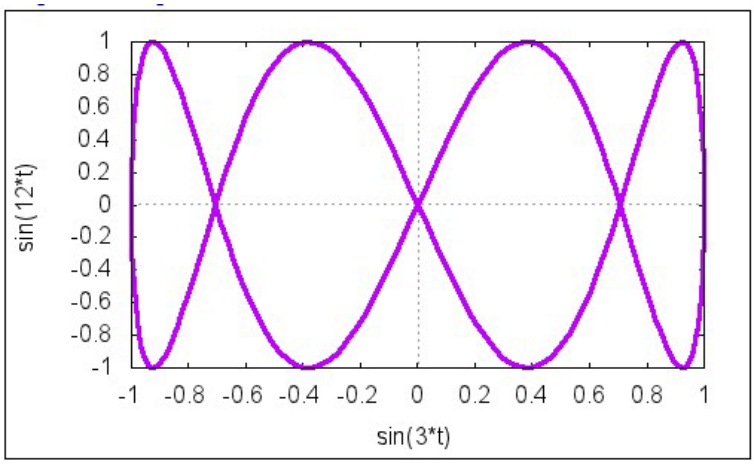

```
(%i48) wxplot2d([parametric, sin(3*t), sin(12*t)], [t, 0, 2*%pi/3],
         [nticks, 200]);
```

(%t48)



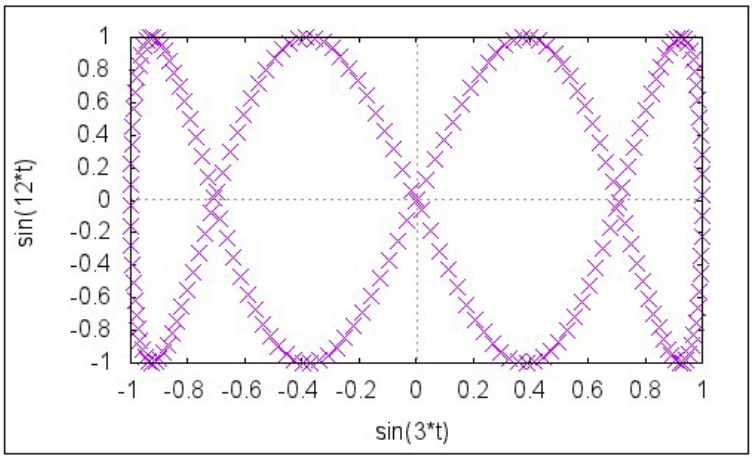

```
(%i49) wxplot2d([parametric, sin(3*t), sin(12*t)], [t, 0, 2*%pi/3],
         [nticks, 200], [style, [lines, 3, 4]]);
```

(%t49)



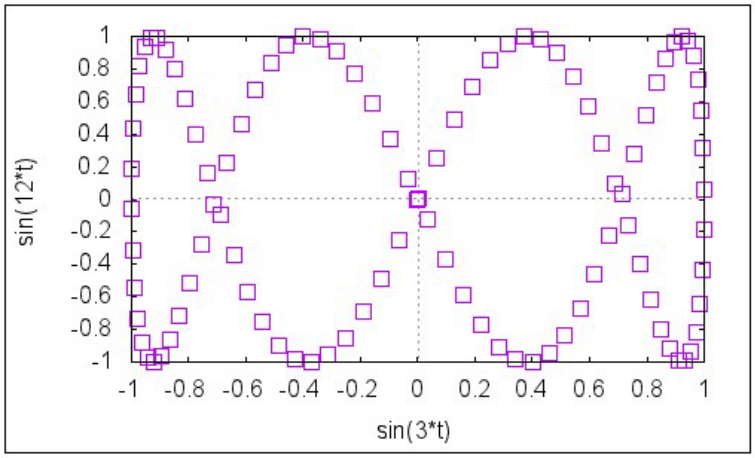

```
(%i50) wxplot2d([parametric, sin(3*t), sin(12*t)], [t, 0, 2*%pi/3],
         [nticks, 200], [style, [points, 3, 4, 4]]);
```

(%t50)



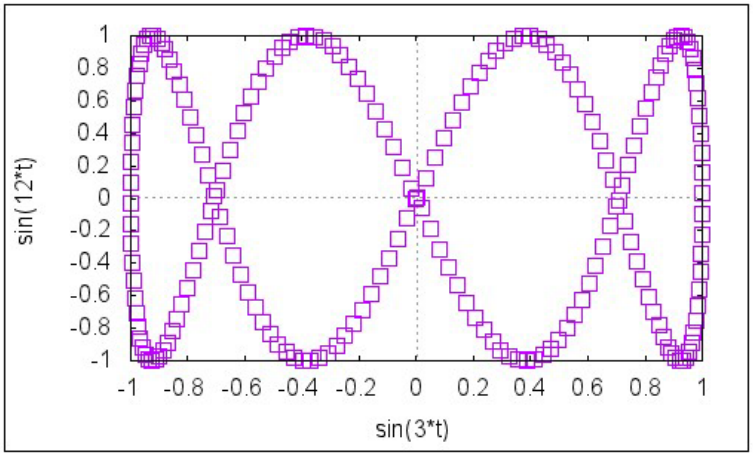
(%i51) wxplot2d([parametric, sin(3*t), sin(12*t)], [t, 0, 2*%pi/3], [nticks, 100], [style, [points, 3, 4, 7]]);

(%t51)



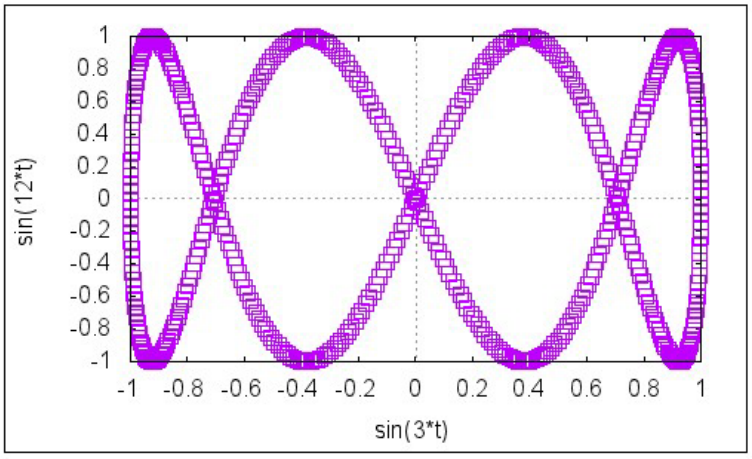
(%i52) wxplot2d([parametric, sin(3*t), sin(12*t)], [t, 0, 2*%pi/3], [nticks, 200], [style, [points, 3, 4, 7]]);

(%t52)



(%i53) wxplot2d([parametric, sin(3*t), sin(12*t)], [t, 0, 2*%pi/3], [nticks, 500], [style, [points, 3, 4, 7]]);

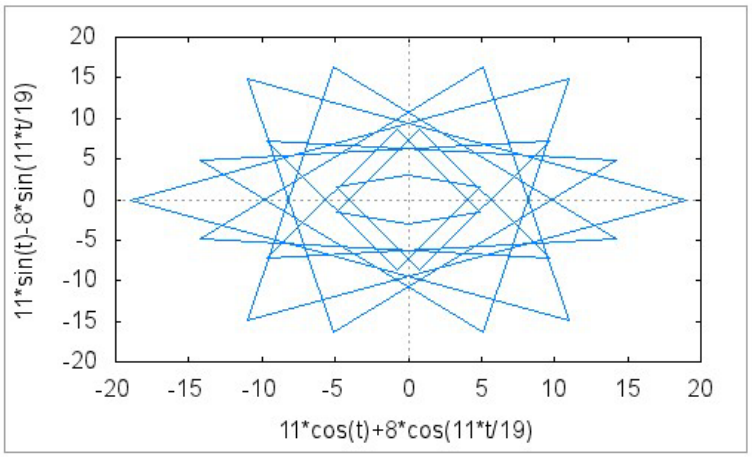
(%t53)



Exemplo 5.17 Considere um círculo de raio s girando dentro de outro círculo maior de raio r, tangentes interiormente. Se P for um ponto qualquer no círculo menor situado a uma distância a do seu centro, então a trajetória descrita por P quando o círculo menor se desloca tangenciando interiormente o círculo maior, sem deslizar, é uma curva conhecida pelo nome de *hipotrocóide* e que pode ser descrita pelas seguintes equações paramétricas: $x = (r-s)\cos(t) + a\cos((\frac{r}{s} - 1)t)$, $y = (r-s)\,\mathrm{sen}(t) - a\,\mathrm{sen}((\frac{r}{s} - 1)t)$, $0 \leq t \leq \mathrm{denom}(\frac{r}{s})$, onde $\mathrm{denom}(\frac{r}{s})$ é o denominador da fração $\frac{r}{s}$. Variando-se os valores de a, r e s, obtém-se uma interessante família de curvas. A primeira tentativa de construção do gráfico fica muito ruim. Por isso, aumentamos o número de pontos com [nticks, 500] e uniformizamos as escalas dos eixos x e y.

```
(%i54)  f(t, a, r, s) := (r-s)*cos(t) + a*cos((r/s - 1)*t)$
(%i55)  g(t, a, r, s) := (r-s)*sin(t) - a*sin((r/s - 1)*t)$
(%i56)  a: 8$ r: 30$ s: 19$
(%i57)  wxplot2d([parametric, f(t,a,r,s), g(t,a,r,s)], [t,0,2*%pi*denom(r/s)]);
```

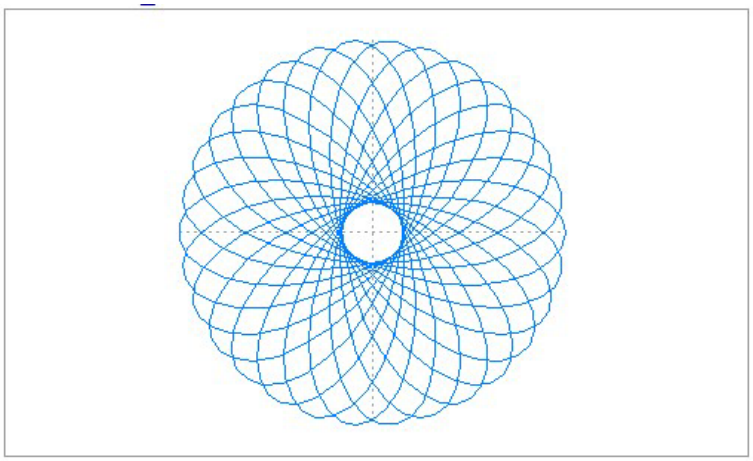
(%t57)




```
(%i58)  a: 8$ r: 30$ s: 19$
(%i59)  wxplot2d([parametric, f(t,a,r,s), g(t,a,r,s)], [t,0,2*%pi*denom(r/s)],
```

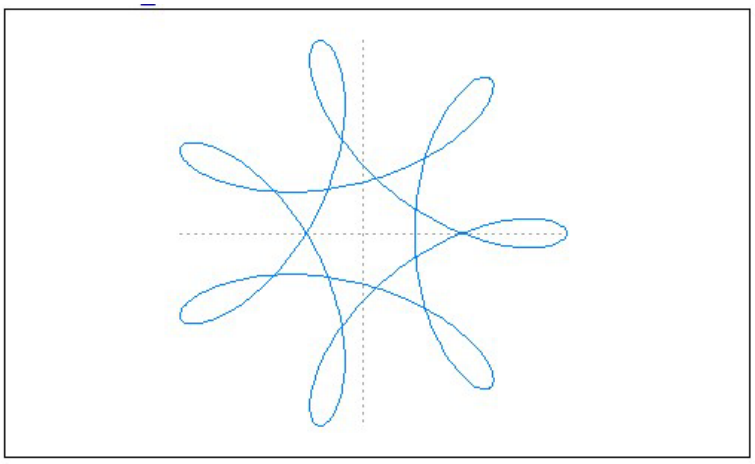
```
                [nticks, 500], [gnuplot_preamble, "set size ratio 1"], [box, false]);
```

(%t59)



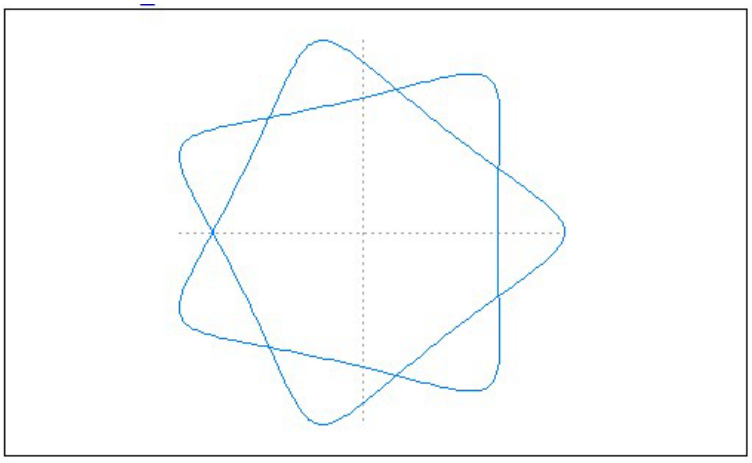
```
(%i60)  a: 2$ r: 21/5$ s: 3$
(%i61)  wxplot2d([parametric, f(t,a,r,s), g(t,a,r,s)], [t,0,2*%pi*denom(r/s)],
                [nticks, 500], [gnuplot_preamble, "set size ratio 1"], [box, false]);
```

(%t61)



```
(%i62)  a: 6$ r: 21/5$ s: 3$

(%i63)  wxplot2d([parametric, f(t,a,r,s), g(t,a,r,s)], [t,0,2*%pi*denom(r/s)],
                [nticks, 500], [gnuplot_preamble, "set size ratio 1"], [box, false]);
```

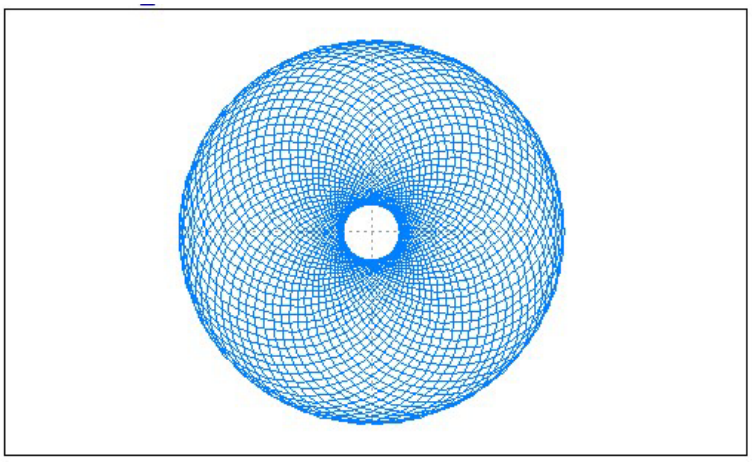
(%t63)



```
(%i64)  a: 40$ r: 30$ s: 1/2$
```

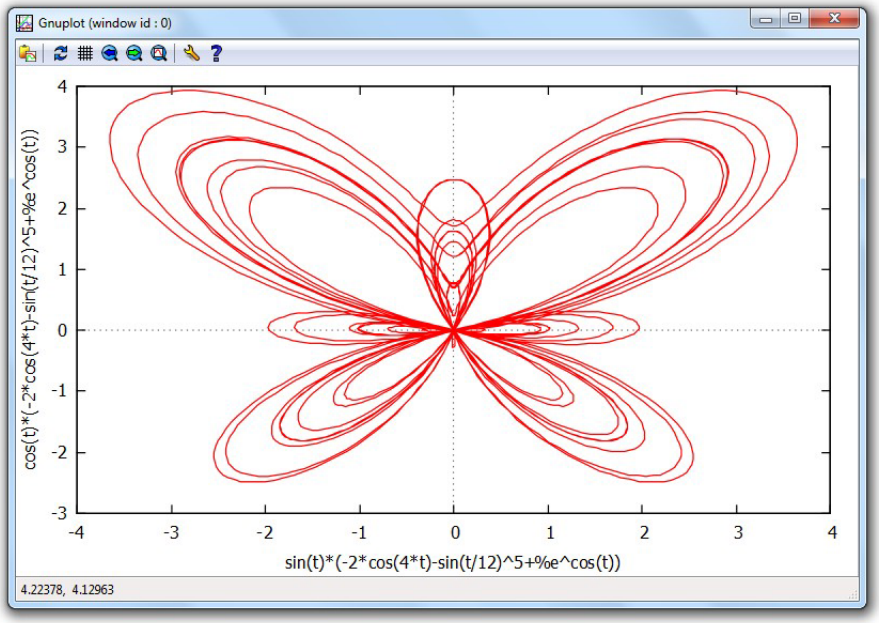
```
(%i65)  wxplot2d([parametric, f(t,a,r,s), g(t,a,r,s)], [t,0,2*%pi*denom(r/s)],
          [nticks, 1200], [gnuplot_preamble, "set size ratio 1"], [box, false]);
```

(%t66)



Exemplo 5.18 Construir o gráfico da "borboleta" definida pelas equações paramétricas $x = r(t)\operatorname{sen} t$, $y = r(t)\cos t$, $-8\pi \leq t \leq 8\pi$ e $r(t) = e^{\cos t} - 2\cos 4t - \operatorname{sen}^5 \frac{t}{12}$.

```
(%i67)  r(t) := exp(cos(t)) - 2*cos(4*t) - sin(t/12)^5 $
(%i68)  plot2d([parametric, r(t)*sin(t), r(t)*cos(t)], [t, -8*%pi, 8*%pi],
          [nticks, 2500], [color, red]);
```



5.7 Gráficos de funções implícitas

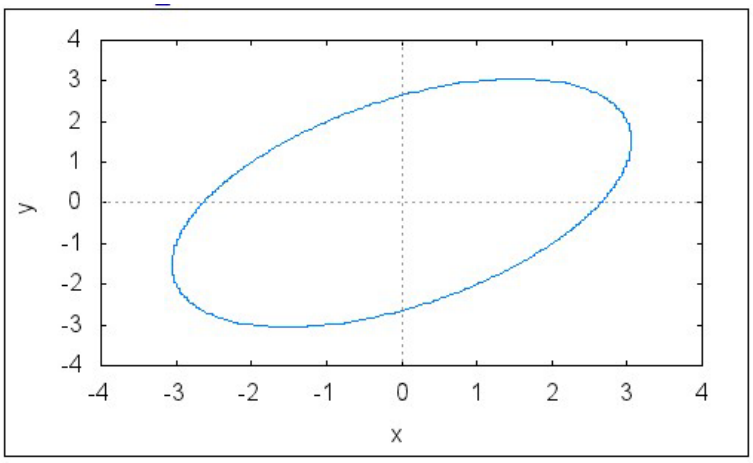
Uma equação em duas variáveis, digamos x e y, com $a \leq x \leq b$ e $c \leq y \leq d$, define y como função implícita de x. Seu gráfico pode ser construído com um comando implicit_plot(equação, [x, a, b], [y, c, d], ...) ou com wximplicit_plot(equação, [x, a, b], [y, c, d], ...). Para tornar esse comando disponível para uso, é preciso usar um load(implicit_plot) pelo menos uma vez.

Vários gráficos podem ser construídos em um mesmo sistema de coordenadas se suas equações forem fornecidas como lista: [equação$_1$, equação$_2$, ...].

Exemplo 5.19 Construir o gráfico da curva definida implicitamente pela equação $x^2 - xy + y^2 = 7$, com $-4 \leq x \leq 4$ e $-4 \leq y \leq 4$.

```
(%i69)  load(implicit_plot)$
(%i70)  wximplicit_plot(x^2 - x*y + y^2 = 7, [x, -4, 4], [y, -4, 4]);
```

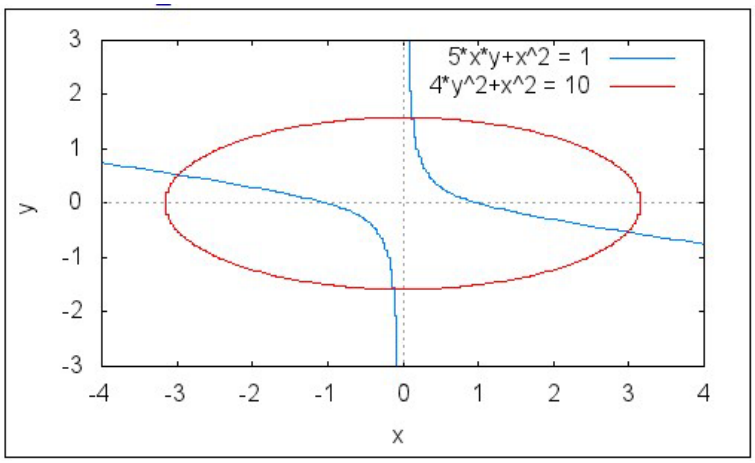
(%t70)



Exemplo 5.20 Construir os gráficos de $x^2 + 5xy = 1$, $x^2 + 4y^2 = 10$, com $-4 \leq x \leq 4$ e $-3 \leq y \leq 3$ em um mesmo sistema de coordenadas.

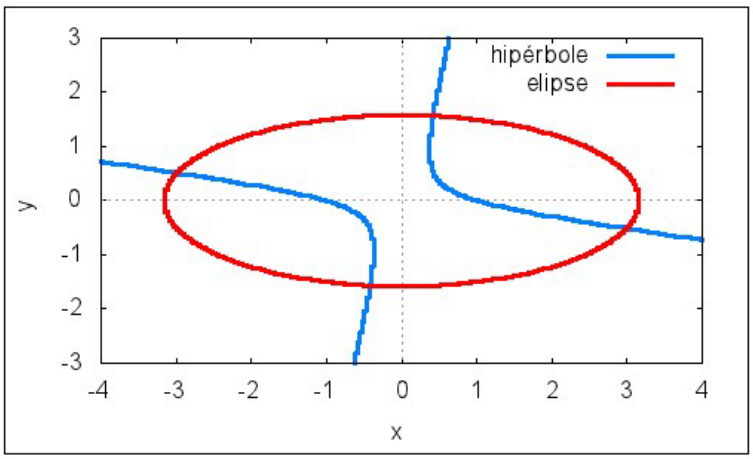
```
(%i71)  load(implicit_plot)$
(%i72)  wximplicit_plot([x^2 + 5*x*y = 1, x^2 + 4*y^2 = 10],
            [x, -4, 4], [y, -3, 3]);
```

(%t72)



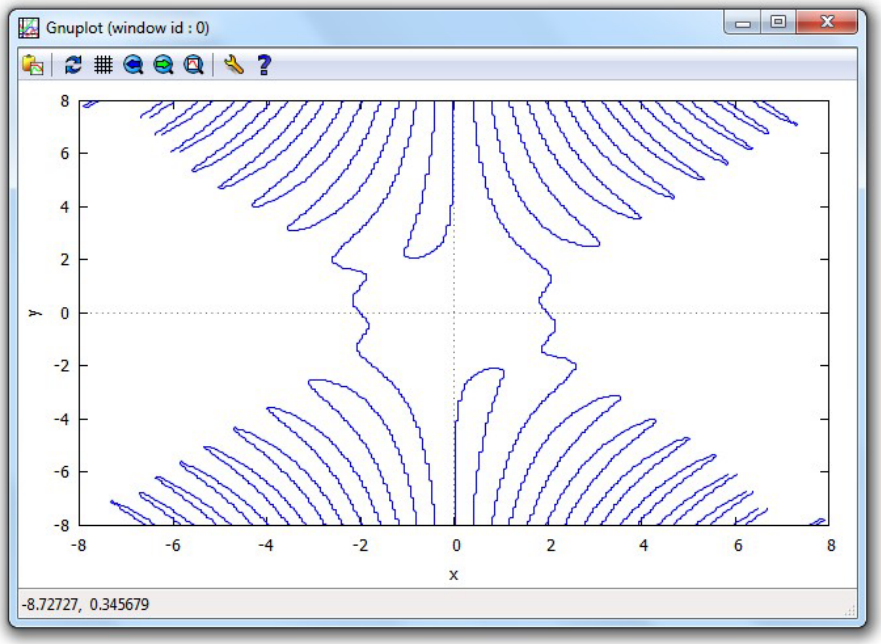

```
(%i73) wximplicit_plot([x^2 + 5*x*y - y^2 = 1, x^2 + 4*y^2 = 10],
         [x,-4,4], [y,-3,3], [legend,"hipérbole","elipse"], [style,[lines,3]]);
```

(%t73)



Exemplo 5.21 Construir o gráfico da função definida implicitamente por $(y^2-1)\,\mathrm{sen}(xy) = x^2 - 4$, $-8 \leq x \leq 8$, $-8 \leq y \leq 8$.

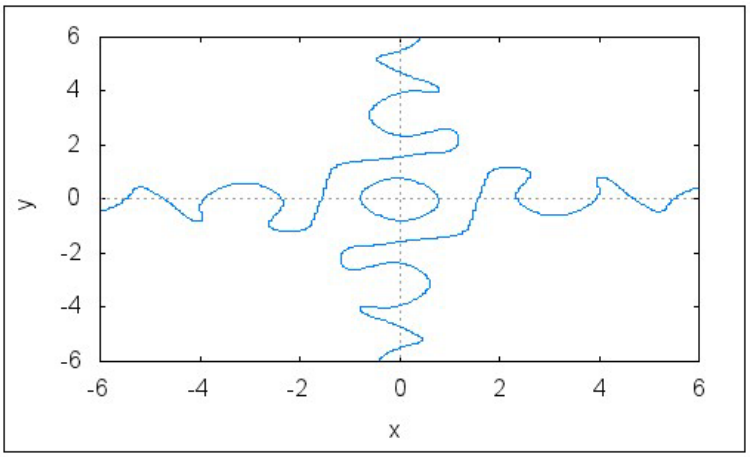
```
(%i74) implicit_plot( (y^2 - 1)*sin(x*y) = x^2 - 4, [x, -8, 8], [y, -8, 8]);
```



Exemplo 5.22 Construir os gráficos definidos pela equação $xy - a\cos y \cos ax - a\cos x \cos ay = 0$, $-6 \leq x \leq 6$, $-6 \leq y \leq 6$, nos seguintes casos: $a = 3$, $a = 5$ e $a = 10$.

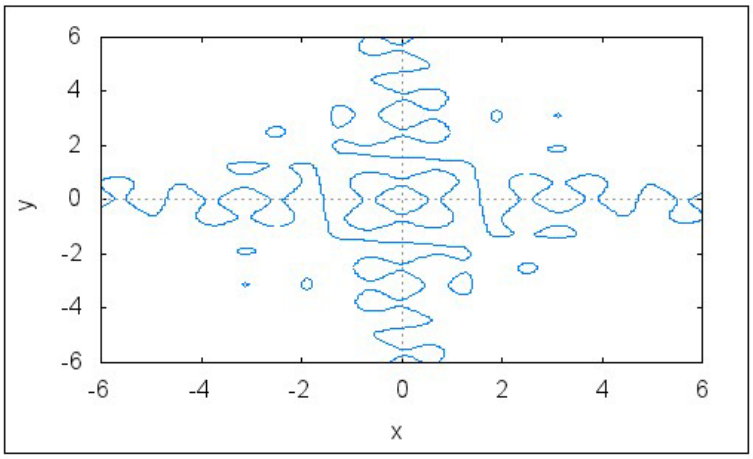
```
(%i75)  load(implicit_plot)$
(%i76)  a: 3$ wximplicit_plot(x*y - a*cos(y)*cos(a*x)
                              - a*cos(x)*cos(a*y) = 0, [x, -6,6], [y,-6,6]);
```

(%t76)



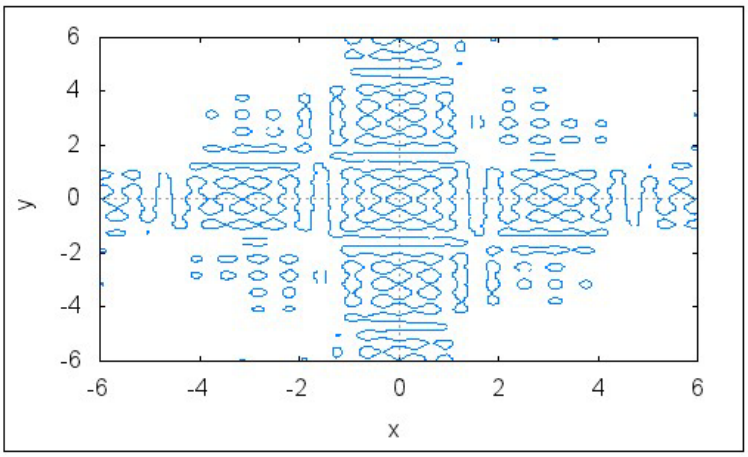

```
(%i77)  a: 5$ wximplicit_plot(x*y - a*cos(y)*cos(a*x)
                              - a*cos(x)*cos(a*y) = 0, [x, -6,6], [y,-6,6]);
```

(%t77)




```
(%i78)  a: 10$ wximplicit_plot(x*y - a*cos(y)*cos(a*x)
                               - a*cos(x)*cos(a*y) = 0, [x, -6,6], [y,-6,6]);
```

(%t78)



5.8 Gráficos tridimensionais

Uma superfície é um gráfico tridimensional que pode ser definido de várias maneiras:

- Através de uma função de duas variáveis $z = f(x, y)$ e seu domínio $x_{min} \leq x \leq x_{max}$, $y_{min} \leq y \leq y_{max}$. Por exemplo, $z = e^{-x^2-y^2}$.
- Através de três funções de duas variáveis $x = f(u, v)$, $y = g(u, v)$ e $z = h(u, v)$ e seus domínios $u_{min} \leq u \leq u_{max}$ e $v_{min} \leq v \leq v_{max}$. Por exemplo, $x = 5 \cos u \operatorname{sen} v$, $y = 5 \operatorname{sen} u \operatorname{sen} v$, $z = 5 \cos v$.
- Através de uma equação de três variáveis $f(x, y, z) = 0$. Por exemplo, $x^2 + y^2 - z^2 - 4 = 0$.

O comando `plot3d(...)` permite que o *Maxima* construa gráficos tridimensionais:

- **plot3d (f(x, y), [x, x_{min}, x_{max}], [y, y_{min}, y_{max}], opções, ...)** para construir o gráfico de uma função de duas variáveis $z = f(x, y)$.

- **plot3d ([f(u, v), g(u, v), h(u, v)], [u, u_{min}, u_{max}], [v, v_{min}, v_{max}], opções, ...)** para construir o gráfico de uma superfície definida por equações paramétricas $x = f(u, v)$, $y = g(u, v)$ e $z = h(u, v)$.

- **plot3d ([f_1(x, y), f_2(x, y), ..., f_n(x, y), [x, x_{min}, x_{max}], [y, y_{min}, y_{max}]], opções, ...)** para construir simultaneamente os gráficos de várias funções de duas variáveis $f_1, f_2, \ldots, f_n$.

No lugar de `plot3d(...)` pode ser usado um `wxplot3d(...)`. A diferença é que o `plot3d` usa uma janela exclusiva, enquanto o `wxplot3d` constrói gráficos embutidos no texto da janela principal do *Maxima*.

Algumas opções de construção do gráfico podem ser as mesmas do `plot2d`. No entanto, tem opções que são exclusivas do `plot3d`:

- **[grid, m, n]** Usa uma grade com $m \times n$ pontos no domínio.

- **[azimuth, ângulo]** Medida em graus do ângulo de rotação do gráfico em torno do eixo x. Só funciona com o plot3d.

- **[elevation, ângulo]** Medida em graus do ângulo de rotação do gráfico em torno do eixo z. Só funciona com o plot3d.

- **[zlabel, nome]** Rótulo para o eixo dos z.

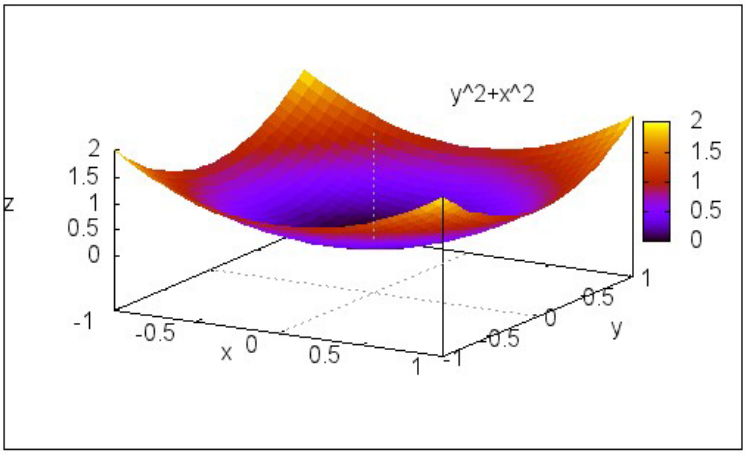
- **[palette, false]** Se for usado em um plot3d(...), então o gráfico é construído usando-se duas cores que podem ser definidas com um comando [color, cor_de_baixo, cor_de_cima]. Se for usado em um wxplot3d(...), então é usada uma única cor.

- **[palette, [nome,** n_1, n_2, n_3, n_{max}**]]** Usa uma variedade de cores baseando-se nos valores dos números n_1, n_2, n_3, n_{max}, todos de 0 e 1. O nome pode ser **hue**, **saturation** ou **value** e define qual desses itens vai variar ao longo do gráfico até atingir o valor máximo n_{max}. Os valores n_1, n_2 e n_3 são os mínimos de hue, saturation e value da cor, respectivamente. Exemplos: [palette, [hue, 0.65, 0.8, 0.9, 0.85]], [palette, [saturation, 0.2, 0.5, 0.7, 0.9]]. Só funciona com o plot3d.

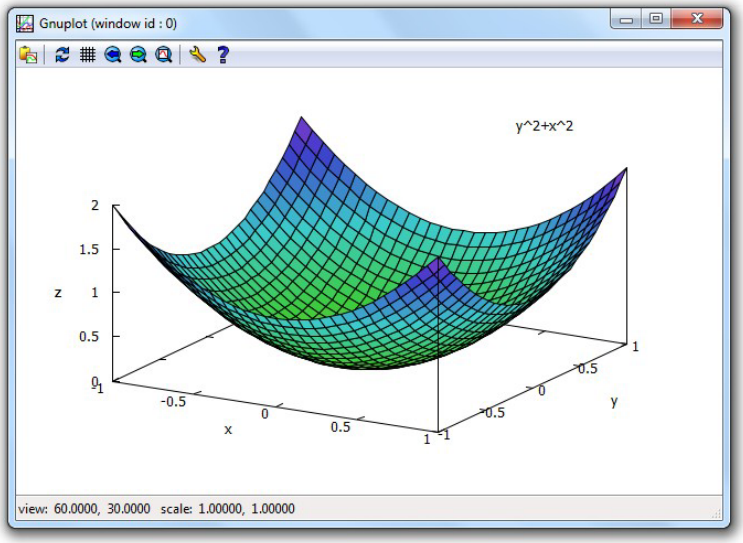
Exemplo 5.23 Usando os comandos wxplot3d e plot3d, construir duas vezes o gráfico de $z = x^2 + y^2$, $-1 \leq x \leq 1$, $-1 \leq y \leq 1$.

```
(%i79)  wxplot3d( x^2+ y^2, [x, -1, 1], [y, -1, 1]);
```

(%t79)



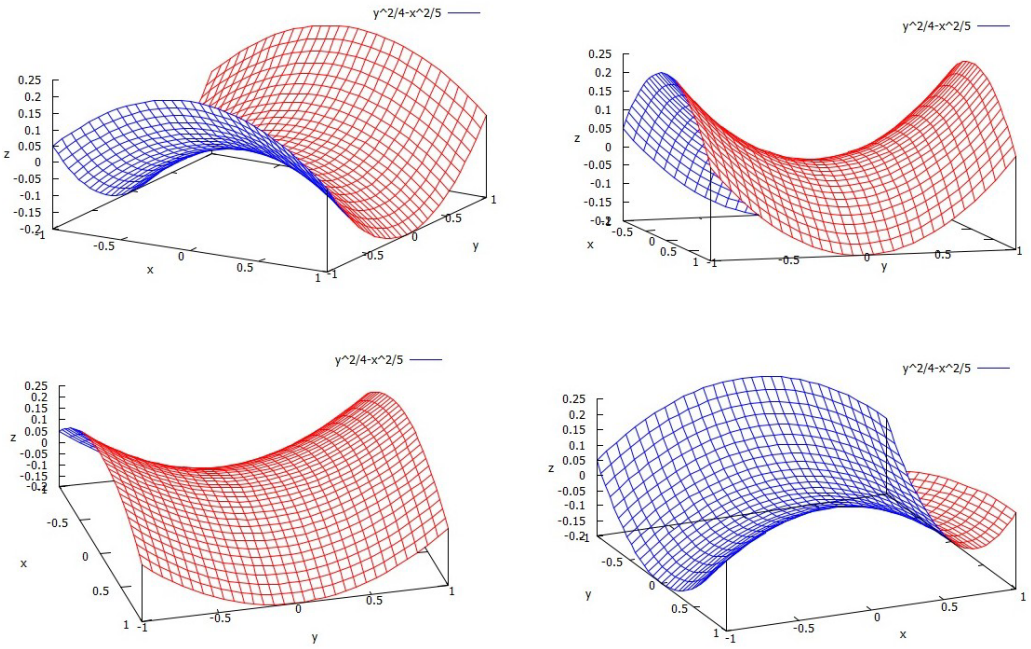

```
(%i80)  plot3d( x^2+ y^2, [x, -1, 1], [y, -1, 1]);
```



A janela aberta com o plot3d precisa ser fechada para tornar possível a construção de um outro gráfico posteriormente.

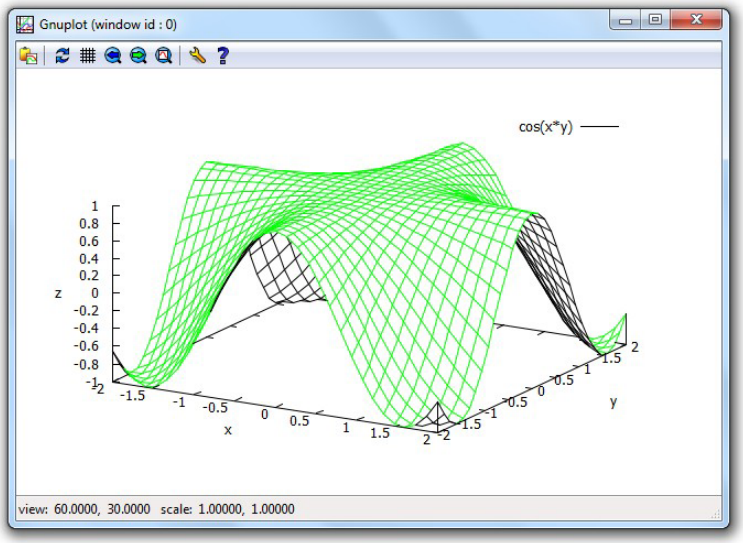
Exemplo 5.24 Construir o gráfico de $z = -\frac{x^2}{4} + \frac{y^2}{5}$ sem paleta (ou seja, com a opção [palette, false]) e com o plot3d. Note que, neste caso, são usadas apenas duas cores para o gráfico e que ele pode ser arrastado com o mouse para diversas posições.

```
(%i81)  plot3d( -x^2/4+ y^2/5, [x, -1, 1], [y, -1, 1], [palette, false]);
```

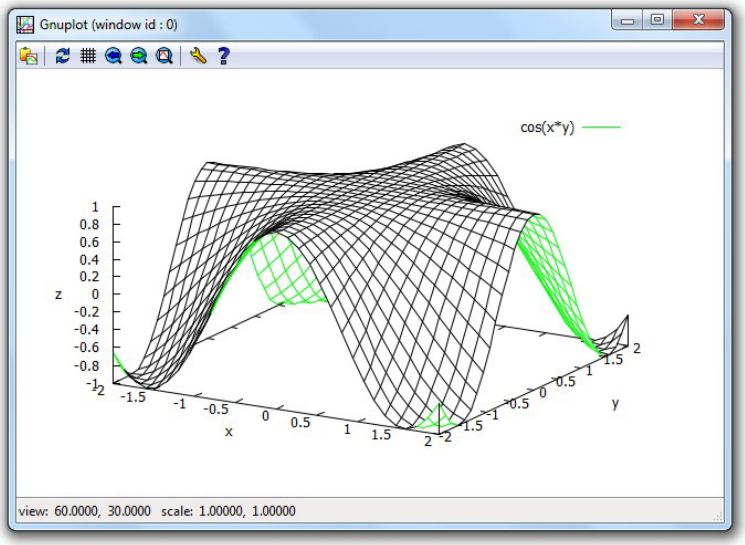


Exemplo 5.25 Construir o gráfico de $z = cos(xy)$ com o plot3d, sem paleta. Primeiramente, usando as cores na ordem black e green, e depois, green e black.

```
(%i82)  plot3d(cos(x*y), [x, -2, 2], [y, -2, 2], [palette, false],
                [color, black, green]);
```

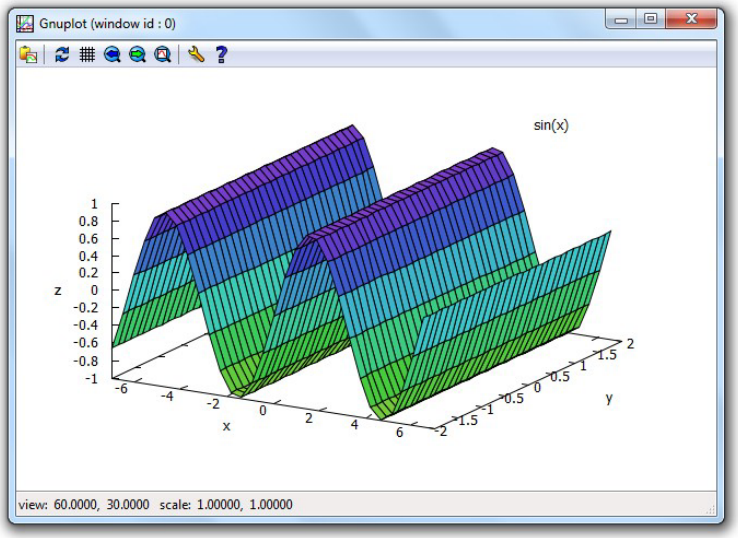



```
(%i83)  plot3d(cos(x*y), [x, -2, 2], [y, -2, 2], [palette, false],
                [color, green, black]);
```

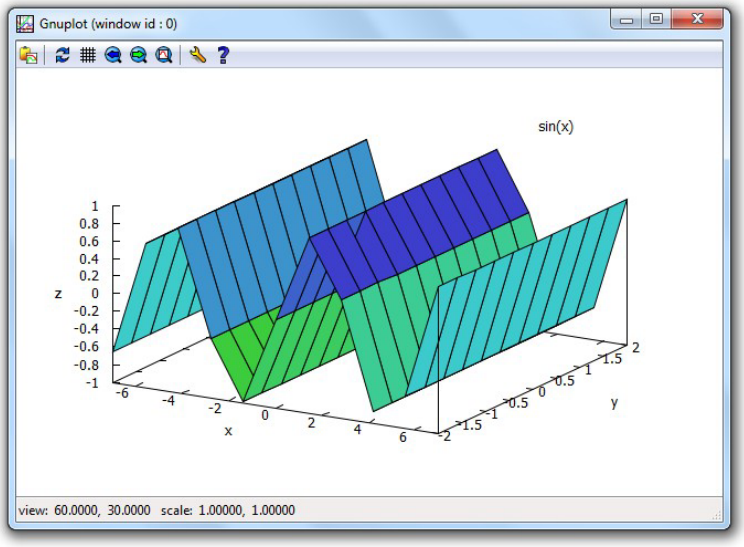


Exemplo 5.26 Construir o gráfico de $z = \operatorname{sen} x$ com o plot3d usando uma grade padrão e, depois, grades 10×10 e 60×60.

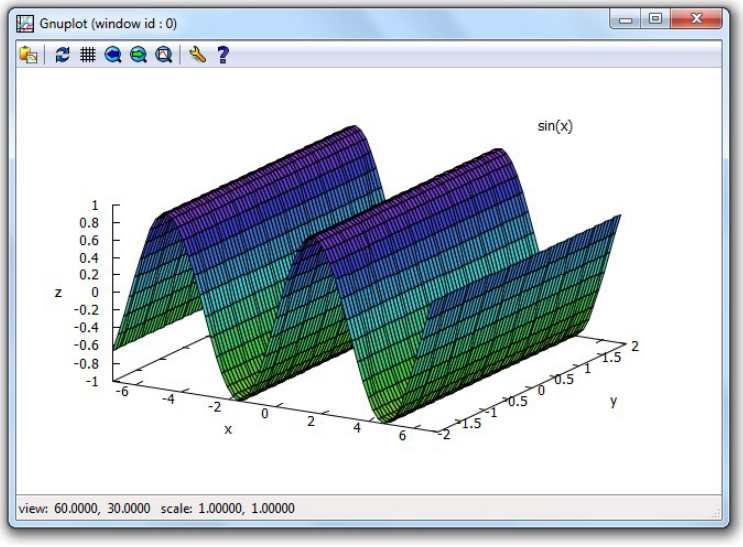
```
(%i84)  plot3d(sin(x), [x, -7, 7], [y, -2, 2]);
```




```
(%i85)  plot3d(sin(x), [x, -7, 7], [y, -2, 2], [grid, 10, 10]);
```

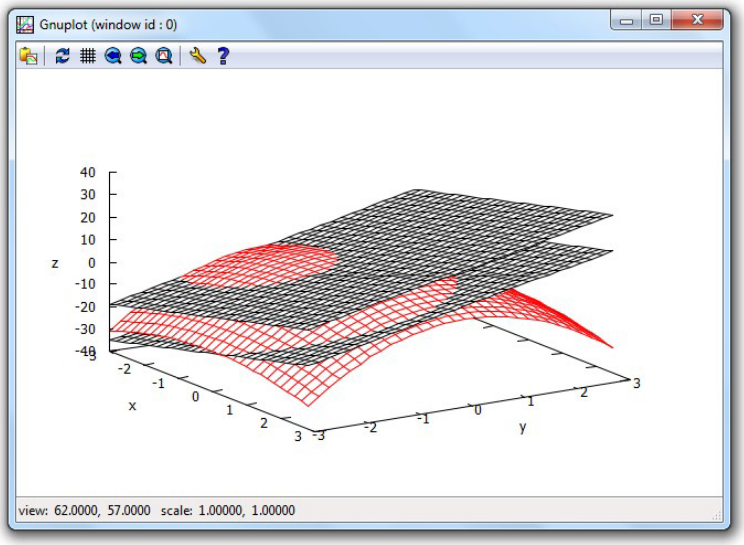


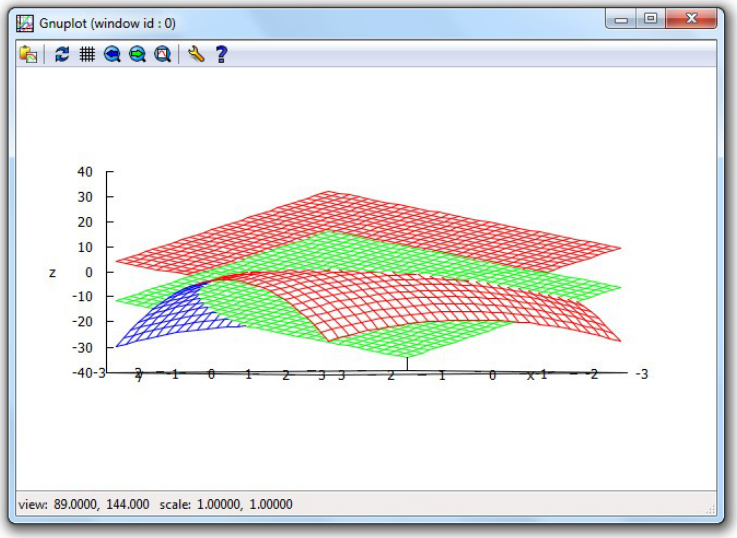

```
(%i86)  plot3d(sin(x), [x, -7, 7], [y, -2, 2], [grid, 60, 60]);
```



Exemplo 5.27 Construir os gráficos do parabolóide $z = 5 - x^2 - 3y^2$ e dos planos paralelos $z = 4x + 5y - 8$ e $z = 4x + 5y + 8$ com $-3 \leq x \leq 3$, $-3 \leq y \leq 3$, em um mesmo sistema de coordenadas cartesianas, sem paleta e sem legenda.

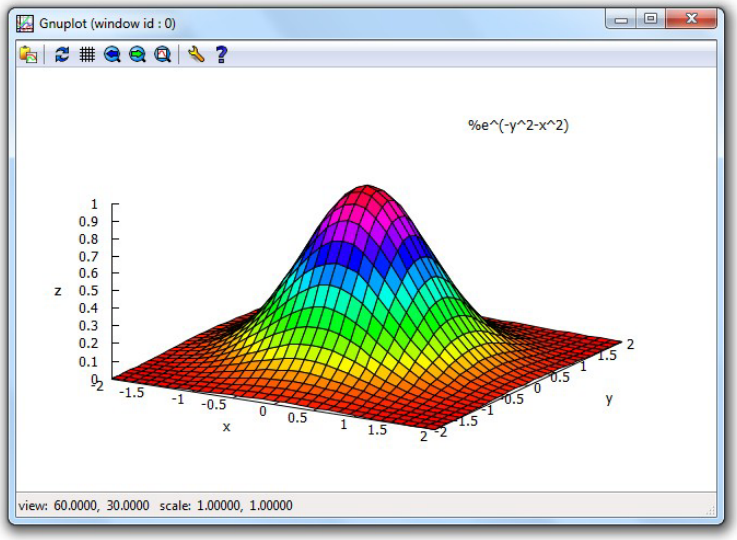
```
(%i87)  plot3d([5-x^2-3*y^2, 4*x + 5*y+8, 4*x + 5*y - 8,
                [x, -3, 3], [y, -3, 3]], [palette, false], [legend, false]);
```



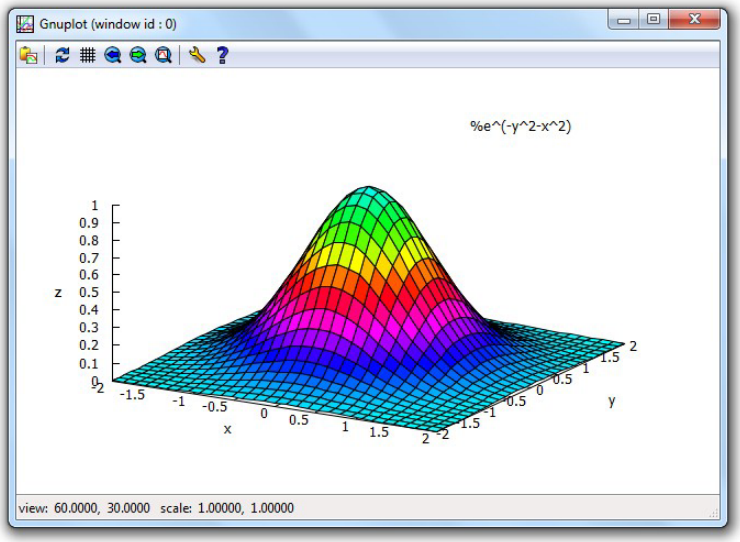


Exemplo 5.28 Construir o gráfico de $z = e^{-x^2-y^2}$, com $-2 \leq x \leq 2$ e $-2 \leq y \leq 2$, usando vários valores da opção palette.

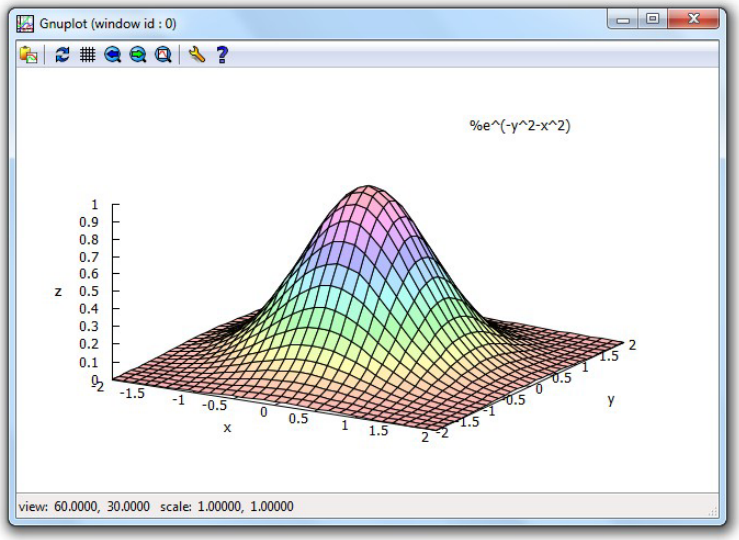
```
(%i88)  plot3d(exp(-x^2-y^2), [x,-2,2], [y,-2,2], [palette, [hue, 0, 1, 1, 1]]);
```



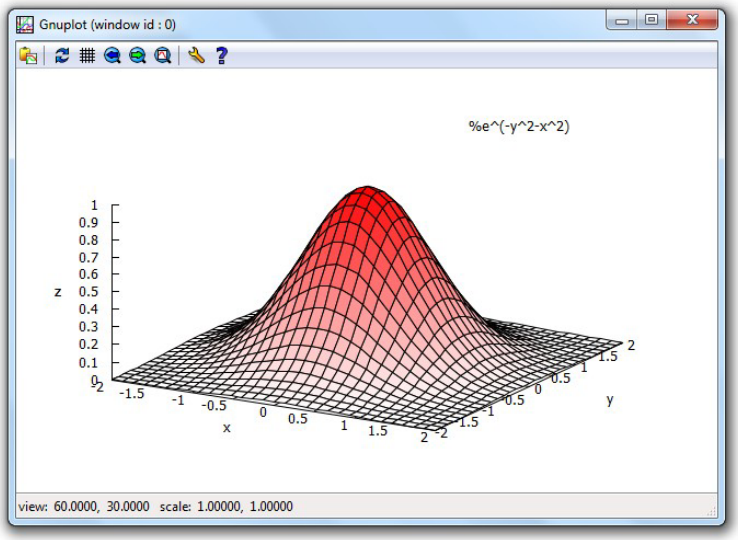

```
(%i89)  plot3d(exp(-x^2-y^2), [x,-2,2], [y,-2,2], [palette, [hue, 0.5, 1, 1, 1]]);
```



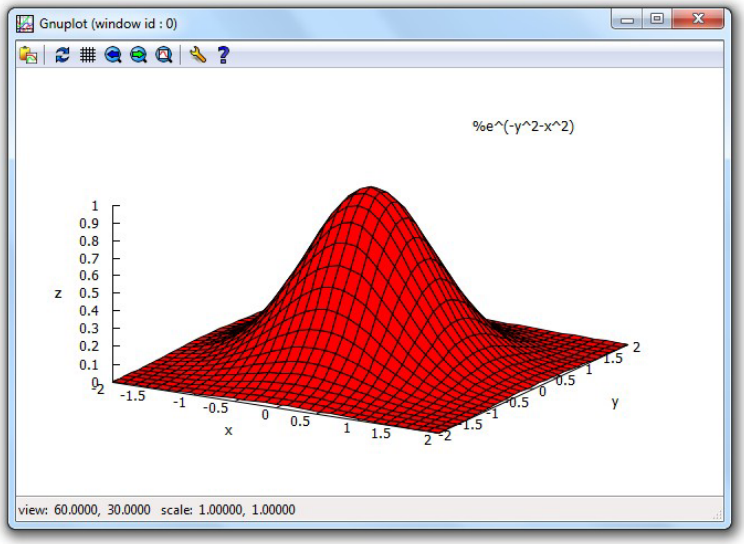

```
(%i90)  plot3d(exp(-x^2-y^2), [x,-2,2], [y,-2,2], [palette, [hue, 0, 0.3, 1, 1]]);
```



```
(%i91)  plot3d(exp(-x^2-y^2), [x,-2,2], [y,-2,2], [palette, [saturation,0,1,1,1]]);
```



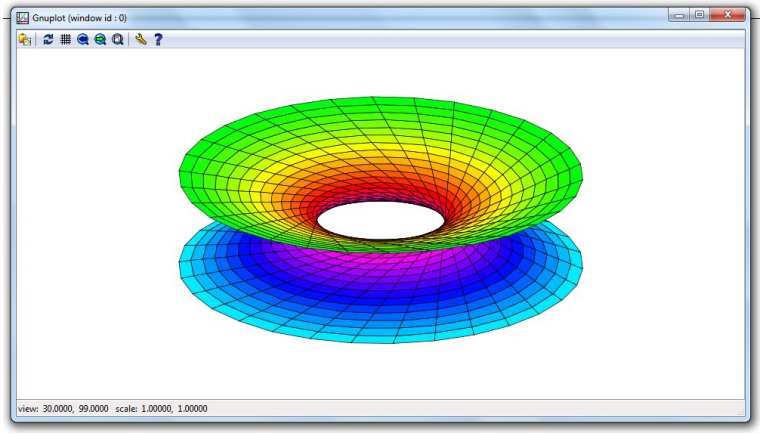

```
(%i92)  plot3d(exp(-x^2-y^2), [x,-2,2], [y,-2,2], [palette, [value, 0, 1, 1, 1]]);
```



5.9 Superfícies parametrizadas

Exemplo 5.29 Construir o gráfico do hiperbolóide de uma folha parametrizado por $x = \cos u - v \operatorname{sen} u$, $y = \operatorname{sen} u + v \cos u$, $z = v$, $0 \leq u \leq 2\pi$, $-3 \leq v \leq 3$.

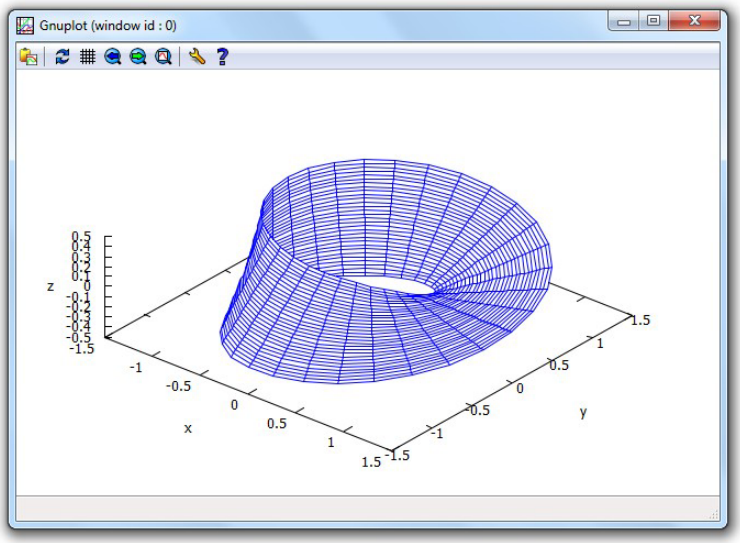
```
(%i93)  x: cos(u) - v*sin(u)$  y: sin(u) + v*cos(u)$  z: v$
(%i94)  plot3d([x, y, z], [u, 0, 2*%pi], [v, -3, 3], [box, false],
         [legend, false], [palette, [hue, 0.5, 1, 1, 1]], [elevation, 30]);
```



Exemplo 5.30 Construir o gráfico da Faixa de Möbius parametrizada por

$$\begin{aligned} x &= (1+\tfrac{v}{2}\cos\tfrac{u}{2})\cos u \\ y &= (1+\tfrac{v}{2}\cos\tfrac{u}{2})\,\mathrm{sen}\,u \\ z &= \tfrac{v}{2}\,\mathrm{sen}\,\tfrac{u}{2},\ 0\le u\le 2\pi,\ -1\le v\le 1. \end{aligned}$$

```
(%i95)  x(u,v) := (1 + v/2*cos(u/2))*cos(u)$
(%i96)  y(u,v) := (1 + v/2*cos(u/2))*sin(u)$
(%i97)  z(u,v) := v/2*sin(u/2)$
(%i98)  dominio_u: [u, 0, 2*%pi]$
(%i99)  dominio_v: [v, -1, 1]$
(%i100)  plot3d([x(u,v),y(u,v),z(u,v)], dominio_u, dominio_v, [palette,false],
          [color, blue, blue], [legend, false], [azimuth, 40], [elevation, 30]);
```



Exemplo 5.31 A Garrafa de Klein é uma famosa superfície parametrizada pelas

seguintes equações:

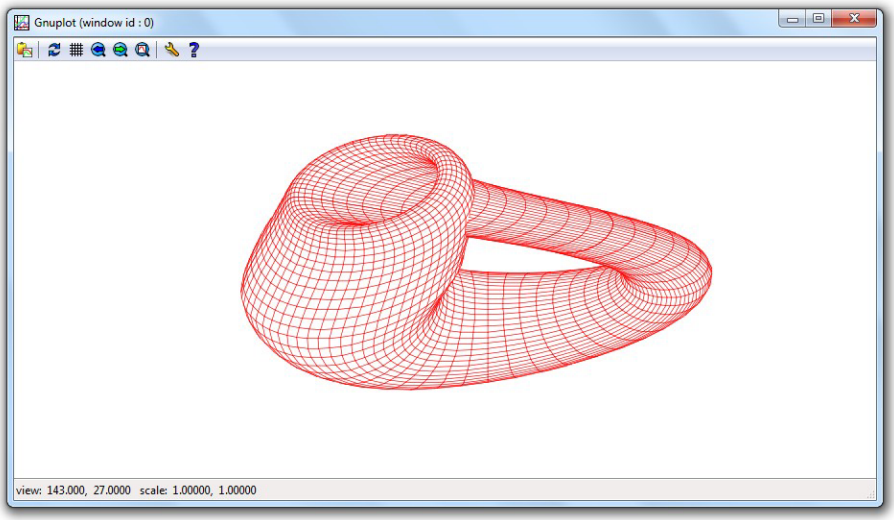
$$f_1(u,v) = \begin{cases} 6\cos u(1+\operatorname{sen} u) + 4(1-\frac{\cos u}{2})\cos u\cos v, & \text{se} \quad 0 \le u \le \pi \\ 6\cos u(1+\operatorname{sen} u) + 4(1-\frac{\cos u}{2})\cos(v+\pi), & \text{se} \quad \pi \le u \le 2\pi \end{cases},$$

$$f_2(u,v) = \begin{cases} 16\operatorname{sen} u + 4(1-\frac{\cos u}{2})\cos v\operatorname{sen} u, & \text{se} \quad 0 \le u \le \pi \\ 16\operatorname{sen} u & \text{se} \quad \pi \le u \le 2\pi \end{cases},$$

$$f_3(u,v) = 4(1-\tfrac{\cos u}{2})\operatorname{sen} v.$$

Seu domínio é o retângulo definido por $0 \le u \le 2\pi$, $0 \le v \le 2\pi$. Desenhar esse gráfico usando uma grade 60×60 no domínio, sem legenda, sem eixos e sem paleta de cores.

```
(%i101)  f1(u, v) := if 0 <= u and u <= %pi then 6*cos(u)*(1 + sin(u))
         + 4*(1 - cos(u)/2)*cos(u)*cos(v) else if %pi <= u and u <= 2*%pi

         then 6*cos(u)*(1 + sin(u)) + 4*(1 - cos(u)/2)*cos(v + %pi)$

(%i102)  f2(u, v) := if 0 <= u and u <= %pi then 16*sin(u)
                     + 4*(1 - cos(u)/2)*cos(v)*sin(u) else if %pi <= u
                     and u <= 2*%pi then 16*sin(u)$

(%i103)  f3(u, v) := 4*(1 - cos(u)/2)*sin(v)$
(%i104)  plot3d([f1(u, v), f2(u, v), f3(u, v)], [u, 0, 2*%pi], [v, 0, 2*%pi],
         [palette, false], [color, red], [grid, 60, 60], [legend, false], [box, false]);
```



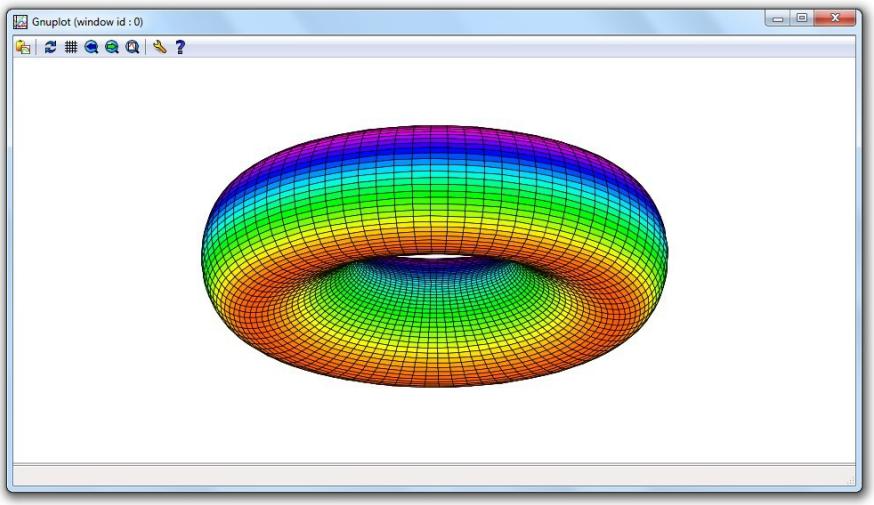
Para o *Maxima*, o condicional "se . . . então . . . senão . . . " deve ser codificado na forma "**if . . . then . . . else . . .** ", e uma dupla desigualdade como $a \le u \le b$ deve ser codificada na forma **a <= u and u <= b** .

Exemplo 5.32 Construir o gráfico do toro definido pelas seguintes equações paramétricas:

$$\begin{aligned} x &= (2+\operatorname{sen}\theta)\cos\phi, \\ y &= (2+\operatorname{sen}\theta)\operatorname{sen}\phi, \\ z &= \cos\theta,\ 0 \le \theta \le 2\pi,\ 0 \le \phi \le 2\pi \end{aligned}$$

Usar uma grade 80×80, sem legenda e sem eixos.

```
(%i105)  x(teta, fi) := (2+sin(teta))*cos(fi)$
(%i106)  y(teta, fi) := (2+sin(teta))*sin(fi)$
(%i107)  z(teta, fi) := cos(teta)$
(%i108)  plot3d([x(teta,fi), y(teta,fi), z(teta,fi)],[teta, 0, 6.29], [fi, 0, 6.29],
         [palette, [hue,0,1,1,1]], [grid,80,80], [legend, false], [box, false]);
```



É o mesmo que

```
(%i109)  x: (2 + sin(u))*cos(v)$
(%i110)  y: (2 + sin(u))*sin(v)$
(%i111)  z: cos(u)$
(%i112)  plot3d([x, y, z], [u, 0, 6.29], [v, 0, 6.29], [palette, [hue,0,1,1,1]],
                [grid,80,80], [legend, false], [box, false]);
```

5.10 Gráficos usando o pacote draw

O *Maxima* possui um pacote chamado "draw" que contém uma grande variedade de comandos para a construção de gráficos. O objetivo desta seção é mostrar apenas uma pequena parte desses recursos.

5.10.1 Principais comandos do draw

Os comandos do pacote draw ficam disponíveis depois de uma chamada com o load ao nome do pacote: load(draw). Nesse pacote, os principais comandos para construção de gráficos são:

- **draw2d(opções, objeto gráfico 1, opções, objeto gráfico 2, ...)** desenha um ou vários gráficos planos usando as opções dadas. Para construir o gráfico embutido no próprio texto, deve-se acrescentar um prefixo "wx" ao nome do comando: **wxplot2d(...).**
- **draw3d(opções, objeto gráfico 1, opções, objeto gráfico 2, ...)** desenha um ou vários gráficos tridimensionais usando as opções dadas. Para construir o gráfico embutido no próprio texto, deve-se acrescentar um prefixo "wx": **wxplot3d(...).**

Os comandos draw2d(...), draw3d(...), wxdraw2d(...) e wxdraw3d(...) são considerados equivalentes a draw(gr2d(...)), draw(gr3d(...)), wxdraw(gr2d(...)), wxdraw(gr3d(...)), respectivamente.

5.10.2 Objetos gráficos planos

Alguns objetos gráficos planos disponíveis são:

- **points([**$x_1, x_2, \ldots, x_n$**], [**$y_1, y_2, \ldots, y_n$**])** Lista de pontos (x_1, y_1), (x_2, y_2), ..., (x_n, y_n)
- **explicit(**$f(x), x, x_{min}, x_{max}$**)** Gráfico da função $f(x)$ com $x_{min} \leq x \leq x_{max}$
- **implicit(equação,** $x, x_{min}, x_{max}, y, y_{min}, y_{max}$**)** Gráfico de função definida implicitamente pela equação dada com $x_{min} \leq x \leq x_{max}$ e $y_{min} \leq y \leq y_{max}$
- **parametric(**$f(t), g(t), t, t_{min}, t_{max}$**)** Curva cujas equações paramétricas são $x = f(t)$, $y = g(t)$, $t_{min} \leq t \leq t_{max}$
- **polar(**$f(\theta), \theta, \theta_{min}, \theta_{max}$**)** Gráfico de $r = f(\theta)$ em coordenadas polares com $\theta_{min} \leq \theta \leq \theta_{max}$

5.10.3 Opções para construção de gráficos

As opções usadas nos comandos draw2d, draw3d etc. devem estar no formato de equações: opção1 = valor1, opção2 = valor2, ... Algumas delas são:

- **color=nome** Define a cor a ser utilizada.
- **background_color=nome** Define a cor de fundo utilizada

- **xrange=[xmin, xmax]** Intervalo do eixo x
- **yrange=[ymin, ymax]** Intervalo do eixo y
- **grid=true/false** Desenha ou não desenha uma grade como fundo
- **title="nome"** Titulo do gráfico
- **nticks=n** Número de pontos utilizados
- **line_width=n** Largura do traço do gráfico
- **line_type = solid/dots** Traço sólido ou pontilhada
- **point_size=n** Tamanho do desenho do ponto
- **point_type=n** Tipo do desenho do ponto, $n \in \{-1, 0, 1, 2, \ldots, 13\}$
- **xaxis=true/false** Desenha ou não o eixo dos x
- **yaxis=true/false** Desenha ou não o eixo dos y
- **xaxis_type=solid/dots** Define se o eixo x é sólido ou pontilhado
- **yaxis_type=solid/dots** Define se o eixo y é sólido ou pontilhado
- **xaxis_color=nome** Cor do eixo dos x
- **yaxis_color=nome** Cor do eixo dos y
- **xaxis_width=n** Largura do traço do eixo dos x
- **yaxis_width=n** Largura do traço do eixo dos y
- **user_preamble="set size ratio 1"** Mantém as mesmas escalas nos eixos
- **proportional_axes=xy** Mantém as mesmas escalas nos eixos
- **terminal=nome** Onde o gráfico é gerado, nome pode ser screen, jpg, png, pdf, eps, eps_color, gif, animated_gif
- **dimensions=[m, n]** Dimensões da imagem do gráfico
- **file_name="arquivo"** Nome do arquivo onde o gráfico é gerado. Pode incluir um caminho como por exemplo "`c:\\temp\\arquivos\\graf1.jpg`".
- **delay=n** Demora em centésimos de segundos entre as imagens de um GIF animado.

Muitas dessas opções só funcionam se forem utilizadas antes do objeto gráfico, ou seja, escritas à esquerda.

Um comando do tipo **set_draw_defaults(opções, ...)** pode ser usado para tornar padrão determinadas opções.

5.10.4 Objetos gráficos tridimensionais

Alguns objetos gráficos tridimensionais disponíveis são:

- **explicit(**$f(x,y), x, x_{min}, x_{max}, y, y_{min}, y_{max}$**)** Gráfico da função $f(x,y)$ com $x_{min} \leq x \leq x_{max}$ e $y_{min} \leq y \leq y_{max}$
- **implicit(equação,** $x, x_{min}, x_{max}, y, y_{min}, y_{max}, z, z_{min}, z_{max}$**)** Gráfico de função definida implicitamente pela equação dada com $x_{min} \leq x \leq x_{max}$, $y_{min} \leq y \leq y_{max}$ e $z_{min} \leq z \leq z_{max}$
- **parametric(**$f(t), g(t), h(t), t, t_{min}, t_{max}$**)** Curva cujas equações paramétricas são $x = f(t)$, $y = g(t)$, $z = h(t)$, $t_{min} \leq t \leq t_{max}$
- **parametric_surface(**$f(u,v), g(u,v), h(u,v), u_{min}, u_{max}, v_{min}, v_{max}$**)** Superfície cujas equações paramétricas são $x = f(u,v)$, $y = g(u,v)$, $z = h(u,v)$, $u_{min} \leq u \leq u_{max}$, $v_{min} \leq v \leq v_{max}$.

5.10.5 Opções para gráficos tridimensionais

Opções exclusivamente para gráficos tridimensionais:

- **surface_hide = true/false** Não mostra ou mostra parte escondida do gráfico
- **enhanced3d = true/false** Controla o modo de colorir a superfície
- **view = [**ϕ, θ**]** Ângulos de rotação em torno dos eixos x e z
- **axis_3d = true/false** Mostra ou não os eixos
- **xu_grid = n** Número de subdivisões no eixo dos x
- **yv_grid = n** Número de subdivisões no eixo dos y
- **zrange=[zmin, zmax]** Intervalo do eixo z
- **zaxis=true/false** Desenha ou não o eixo dos z
- **zaxis_type=solid/dots** Define se o eixo z é sólido ou pontilhado
- **zaxis_color=nome** Cor do eixo dos z
- **zaxis_width=n** Largura do traço do eixo dos z
- **palette=[m,n,p]** Paleta de cores, m, n, p são inteiros de -36 a 36
- **proportional_axes=xyz** Mantém as mesmas escalas nos eixos

5.10.6 As cores do pacote draw

O pacote "draw" torna disponível uma grande variedade de cores que podem ser usadas nos gráficos. São cores que podem ser referenciadas com os seguintes nomes: white, black, gray, light-gray, dark-gray, red, light-red, dark-red, yellow, dark-yellow, green, light-green, dark-green, spring-green, forest-green, sea-green, blue, light-blue, dark-blue, midnight-blue, navy, medium-blue, royalblue, skyblue, cyan, light-cyan, dark-cyan, magenta, light-magenta, dark-magenta, turquoise, light-turquoise, dark-turquoise, pink, light-pink, dark-pink, coral, light-coral, orange-red, salmon, light-salmon, dark-salmon, aquamarine, khaki, dark-khaki, goldenrod, light-goldenrod, dark-goldenrod, gold, beige, brown, orange, dark-orange, violet, dark-violet, plum, purple.

As cores também podem ser fornecidas no formato de número hexadecimal "#RRGGBB", onde RR são dois algarismos hexadecimais (0, 1, . . . , 9, a, b, c, d, e, f) que correspondem à componente vermelha da cor, GG é a componente verde e BB é a azul. Por exemplo, color = "#ff293c", color = "#34ab6d", color = "#0000ef".

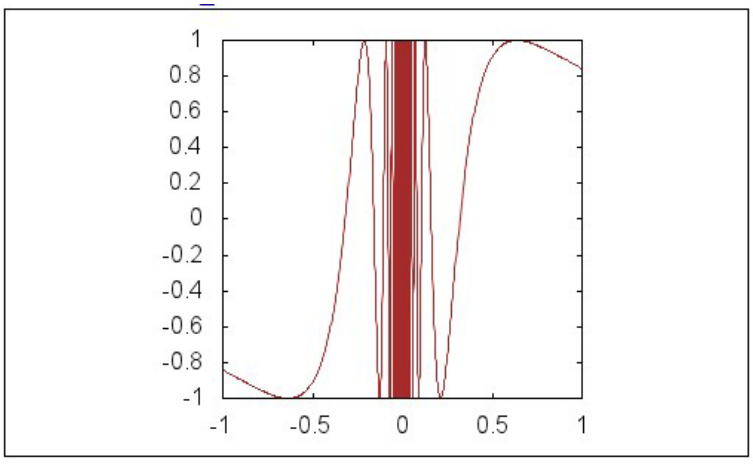
5.10.7 Exemplos de gráficos planos

Para executar qualquer um dos exemplos mostrados a seguir, é preciso que tenha sido digitado antes um comando **load(draw)**, pelo menos uma vez.

Exemplo 5.33 Construir o gráfico da função $f(x) = \text{sen}\,\frac{1}{x}$, com $-1 \leq x \leq 1$.

```
(%i113)  load(draw)$
(%i114)  wxdraw2d(line_width=1, color = brown, explicit(sin(1/x), x, -1, 1));
```

(%t114)



Para o gráfico ser gerado em um arquivo, basta acrescentar opções terminal e file_name na linha de comando, como por exemplo

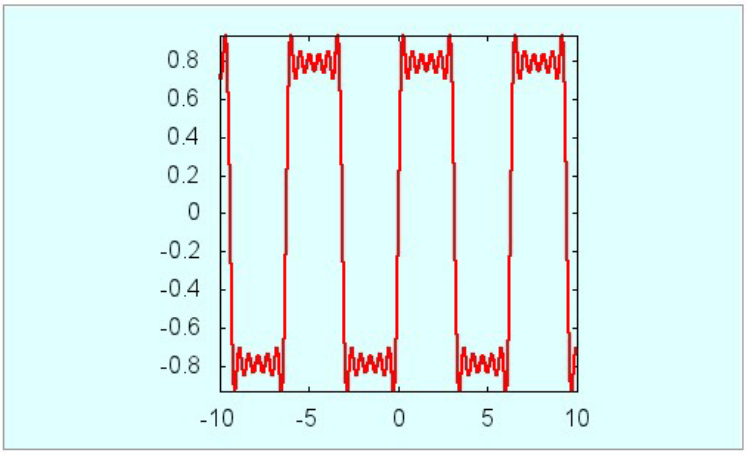
- wxdraw2d(terminal=jpg, file_name="`c:\\temp\\grafico`", line_width=1, color = brown, explicit(sin(1/x), x, -1, 1)); para gerar um arquivo `GRAFICO.JPG` na pasta `C:\TEMP`

- wxdraw2d(terminal=pdf, file_name="`d:\\arquivos\\exemp\\testegr`", line_width=1, color = brown, explicit(sin(1/x), x, -1, 1)); para gerar um arquivo `TESTEGR.PDF` na pasta `D:\ARQUIVOS\exemp`.

Exemplo 5.34 Construir o gráfico de $f(x) = \operatorname{sen} x + \frac{\operatorname{sen} 3x}{3} + \frac{\operatorname{sen} 5x}{5} + \frac{\operatorname{sen} 7x}{7} + \frac{\operatorname{sen} 9x}{9} + \frac{\operatorname{sen} 11x}{11}$, com $-10 \leq x \leq 10$. Usar no gráfico uma cor de fundo ciano claro e um traço de cor vermelha e largura 2.

```
(%i115) wxdraw2d(background_color=light-cyan, line_width=2, color = red,
          explicit(sin(x) + sin(3*x)/3 + sin(5*x)/5 + sin(7*x)/7
          + sin(9*x)/9 + sin(11*x)/11, x, -10, 10));
```

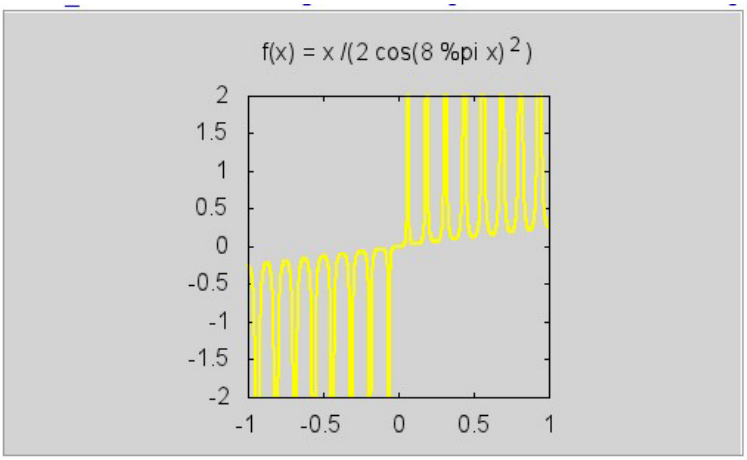
(%t115)



Exemplo 5.35 Construir o gráfico de $f(x) = \frac{x}{2\cos^2 8\pi x}$, $-1 \leq x \leq 1$, usando um traço de largura 2 e cor amarela em um fundo de cor cinza claro. Limitar o eixo dos y ao intervalo $[-2,\ 2]$.

```
(%i116) wxdraw2d(title = "f(x) = x /(2 cos(8 pi x)^2)",
        background_color=light-gray, line_width=2, color=yellow,
        explicit(x/(2*cos(8*%pi*x))^2, x, -1, 1), [yrange = [-2, 2]]);
```

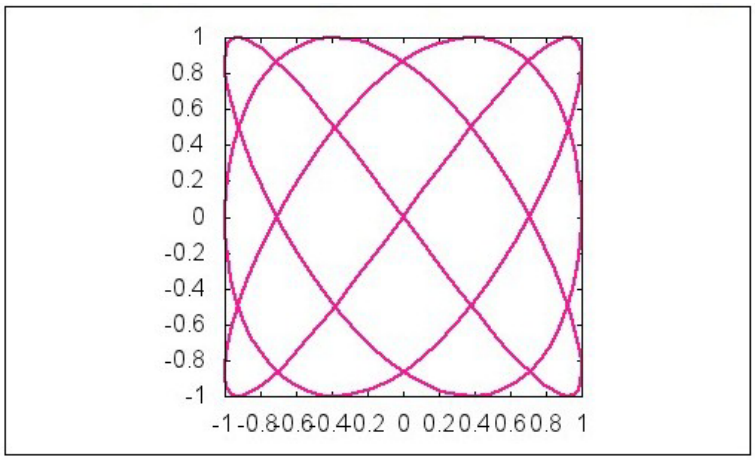
(%t116)



Exemplo 5.36 A família de curvas cujas equações paramétricas são $x = \operatorname{sen} mt$, $y = \operatorname{sen} nt$, $0 \leq t \leq 2\pi$ são conhecidas como curvas de Lissajous. Construir o gráfico da curva de Lissajous em que $m = 3$ e $n = 4$.

```
(%i117)  wxdraw2d(nticks=200, proportional_axes=xy, line_width=2,
         color=dark-pink, parametric(sin(3*t), sin(4*t), t, 0, 2*%pi));
```

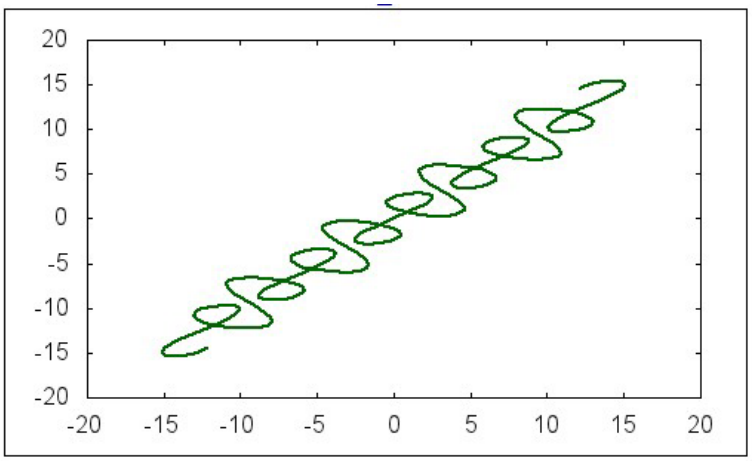
(%t117)



Exemplo 5.37 Construir o gráfico da curva cujas equações paramétricas são $x = t + 2\operatorname{sen} 3t$, $y = t + 2\operatorname{sen} 2t$, $-14 \leq t \leq 14$.

```
(%i118)  wxdraw2d(nticks=200, line_width=2, color=dark-green,
         parametric(t + 2*sin(3*t), t + 2*sin(2*t), t, -14, 14),
         xrange=[-20,20], yrange=[-20,20]);
```

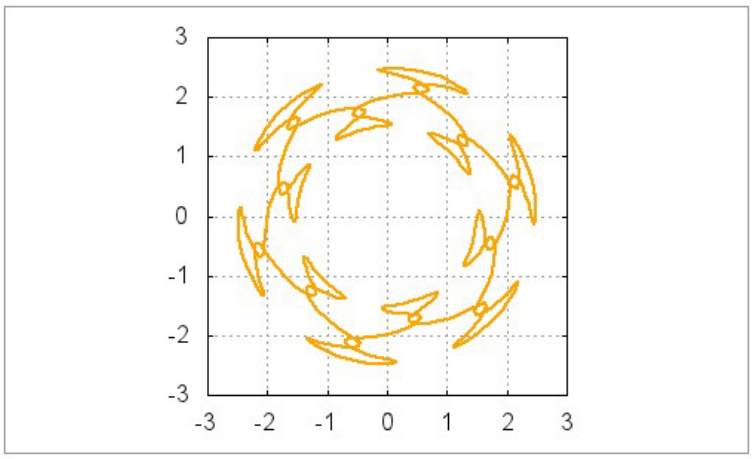
(%t118)



Exemplo 5.38 A família de curvas definidas pelas equações paramétricas $x = (2 + \frac{\text{sen } at}{2})\cos(t + \frac{\text{sen } bt}{c})$, $y = (2 + \frac{\text{sen } at}{2})\,\text{sen}(t + \frac{\text{sen } bt}{c})$, $0 \leq t \leq 2\pi$, possui uma grande variedade de formatos curiosos. Construir os gráficos de duas dessas curvas: uma com $a = 6$, $b = 24$ e $c = 4$ e outra com $a = 11$, $b = 45$, $c = 7$.

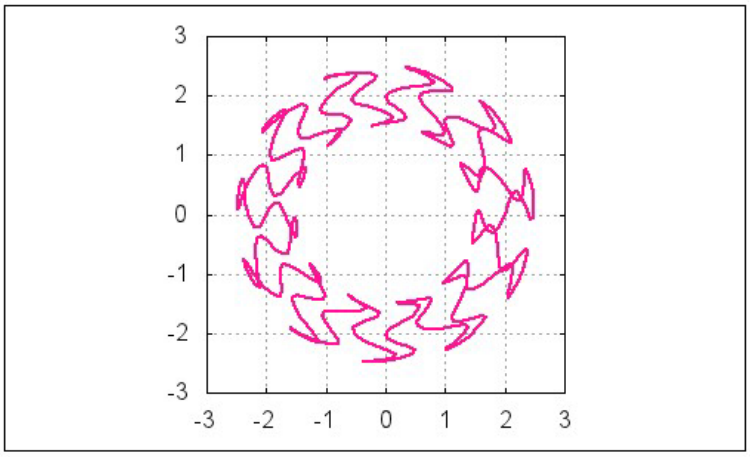
```
(%i119) a: 6; b: 24; c: 4; wxdraw2d(nticks=400, line_width=2,
color=orange, grid=true, user_preamble="set size ratio 1",

parametric((2 + sin(a*t)/2)*cos(t + sin(b*t)/c),
                (2 + sin(a*t)/2)*sin(t + sin(b*t)/c), t, 0, 2*%pi),
xrange=[-3,3], yrange=[-3,3]);
```

(%t119)




```
(%i120) a: 11; b: 45; c: 7; wxdraw2d(nticks=400, line_width=2,
color=dark-pink, grid=true, user_preamble="set size ratio 1",

parametric((2 + sin(a*t)/2)*cos(t + sin(b*t)/c),

                (2 + sin(a*t)/2)*sin(t + sin(b*t)/c), t, 0, 2*%pi),
xrange=[-3,3], yrange=[-3,3]);
```

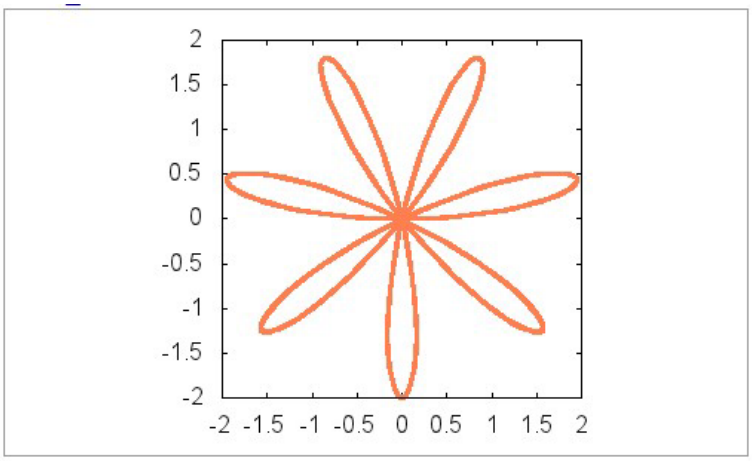
(%t120)



Exemplo 5.39 Uma equação do tipo $r = a \operatorname{sen} n\theta$ ou $r = a \cos n\theta$, em coordenadas polares, corresponde a um rosáceo de n pétalas se n for ímpar ou $2n$ pétalas se n for par. Construir o gráfico do rosáceo de 7 pétalas cuja equação é dada por $r = 2 \operatorname{sen} 7\theta$, $0 \leq \theta \leq 2\pi$.

```
(%i121)  graf2: polar(2*sin(7*teta), teta, 0, 2*%pi)$
(%i122)  wxdraw2d(nticks=300, xrange=[-2, 2], yrange=[-2,2], color=coral,
         line_width=3, user_preamble="set size ratio 1", graf2);
```

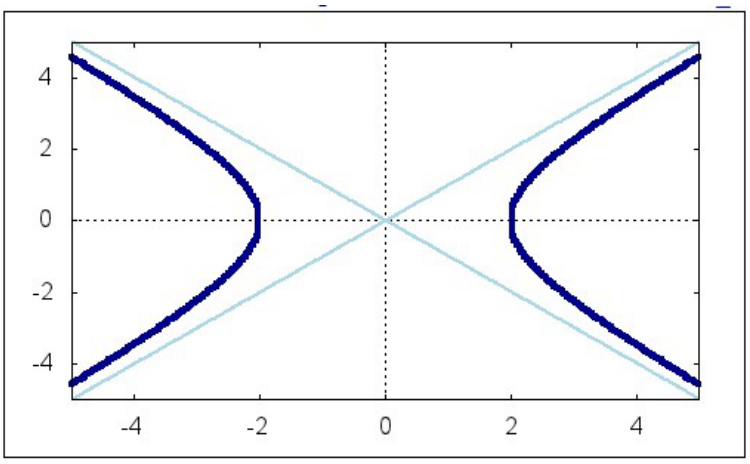
(%t122)



Exemplo 5.40 Usando um traço de largura 4 e cor azul escura, desenhar o gráfico da hipérbole definida implicitamente pela equação $x^2 - y^2 = 4$, com $-4 \leq x \leq 4$ e $-5 \leq y \leq 5$. Desenhar também no mesmo sistema de coordenadas as retas assíntotas $y = x$ e $y = -x$ com traço de largura 2 e cor azul clara.

```
(%i123)  graf1: implicit( x^2 - y^2 = 4, x, -5, 5, y, -5, 5)$
         reta1: implicit(y = x, x, -5, 5, y, -5, 5)$
         reta2: implicit(y = -x, x, -5, 5, y, -5, 5)$
(%i24)   wxdraw2d(xaxis=true, yaxis=true, nticks=300, line_width=4,
         color=dark-blue, graf1, line_width=2,color=light-blue, reta1, reta2);
```

(%t124)

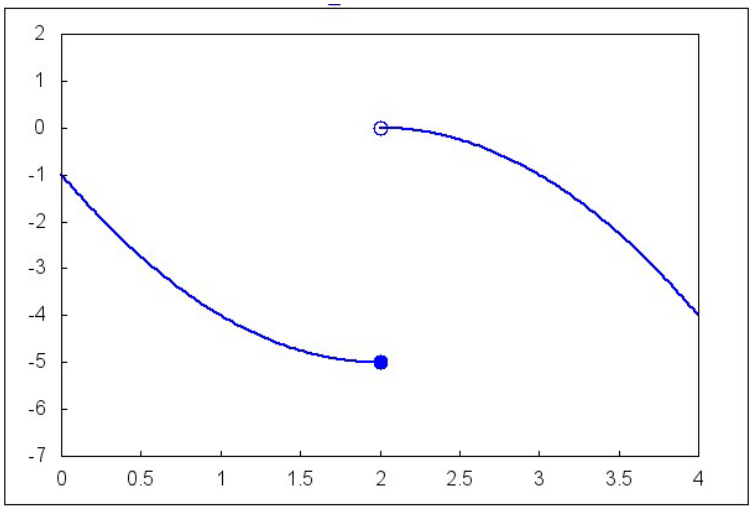


Exemplo 5.41 Construir o gráfico de $F(x) = \begin{cases} x^2 - 4x - 1 & \text{, se} \quad x \leq 2 \\ -4 + 4x - x^2 & \text{, se} \quad x > 2 \end{cases}$ no intervalo $[0,\ 4]$.

Um gráfico de uma função definida por várias sentenças pode ser visto como sendo os gráficos de várias funções construídos em um mesmo sistema de eixos. O gráfico deste exemplo pode ser visto como sendo os gráficos de $f(x) = x^2 - 4x - 1$ em $[0,\ 2]$ e $g(x) = -4 + 4x - x^2$ em $]2,\ 4]$.

```
(%i125)  load(draw)$
(%i126)  f(x) := x^2 -4*x - 1$   g(x) := -4 + 4*x - x^2$
(%i127)  a: 2$  xmin: 0$  xmax: 4$
(%i128)   wxdraw2d(yrange=[-7, 2], line_width=2, g1, g2, point_size = 2,
point_type = 7, points([a], [f(a)]), point_type=6, points([a], [g(a)]));
```

(%t128)

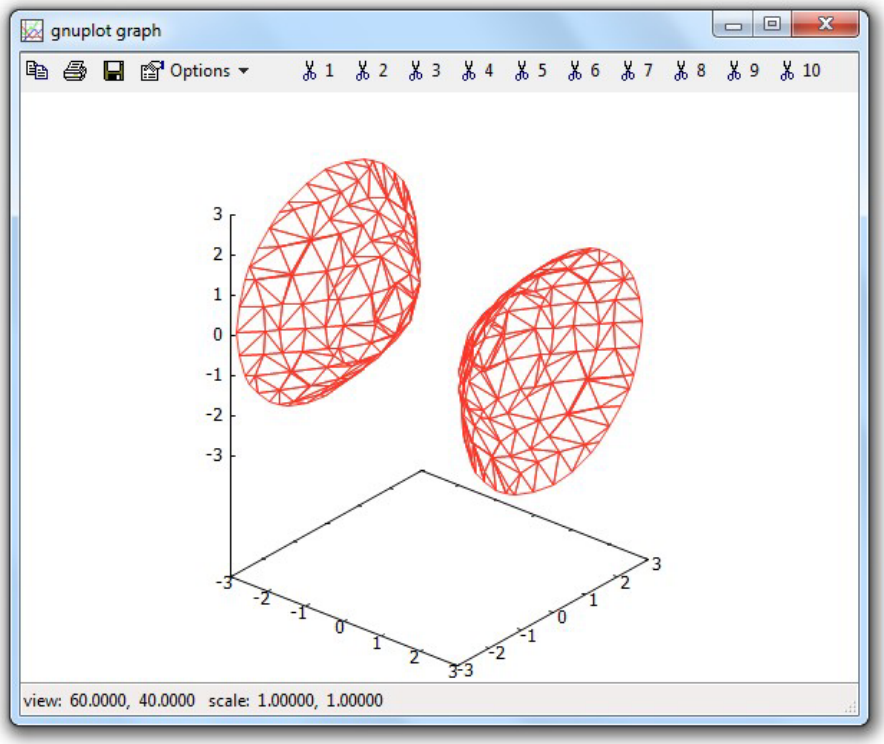


5.10.8 Exemplos de gráficos tridimensionais

A seguir, alguns exemplos de gráficos tridimensionais. Em todos os exemplos, é necessário que tenha sido carregado o pacote draw anteriormente. Para isso, basta executar um comando **load(draw)** no início.

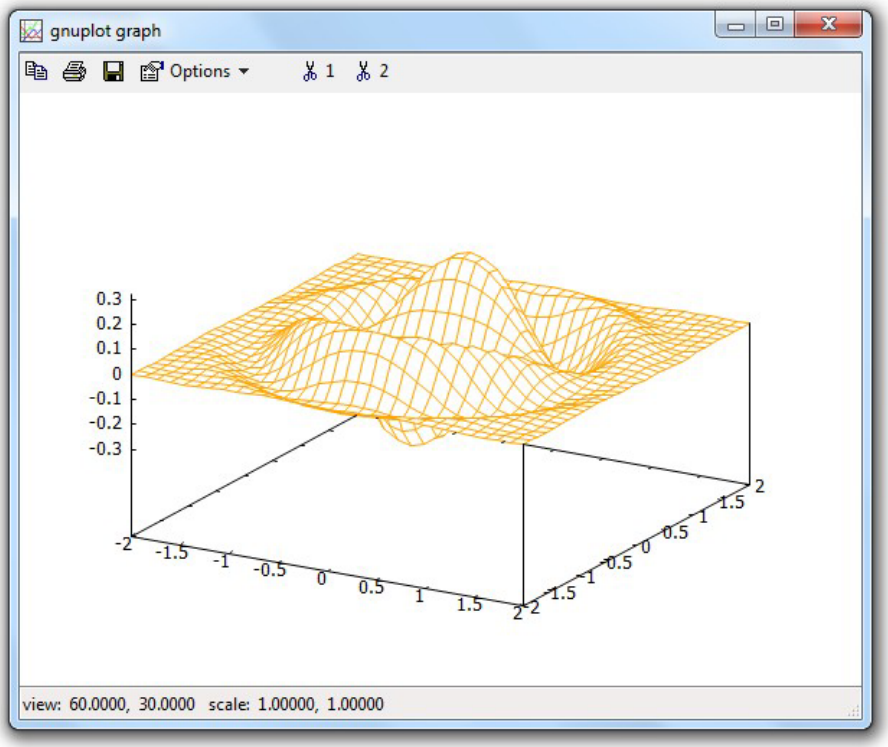
Exemplo 5.42 Desenhar o gráfico do hiperbolóide de duas folhas definido implicitamente pela equação $x^2 - y^2 - z^2 = 1$, com $-3 \leq x \leq 3$, $-3 \leq y \leq 3$, $-3 \leq z \leq 3$. Usar eixos com mesma escala proporcional, não mostrar a parte escondida do gráfico (ou seja, a parte que fica atrás ou abaixo de outras), ângulos de rotações iguais a 60° e 40° e usar uma cor cujo número hexadecimal é #fb3527.

```
(%i125)  load(draw)$
(%i126)  hiperboloide: implicit( x^2 - y^2 - z^2 = 1, x,-3,3, y,-3,3, z,-3,3) $
(%i127)  draw3d(proportional_axes = xyz, surface_hide=true,
         color = "#fb3527", hiperboloide, view = [60, 40]);
(%o127)  [gr3d(implicit)]
```



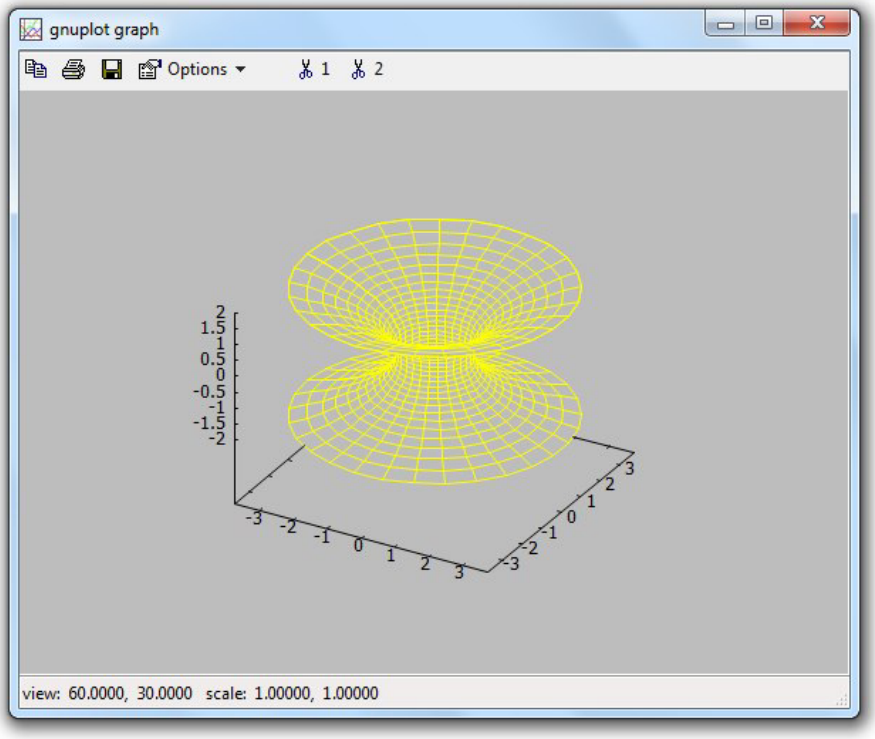
Exemplo 5.43 Usando uma cor laranja e não mostrando a parte escondida, desenhar o gráfico da função $f(x, y) = e^{-x^2-y^2} \cos \frac{x}{4} \operatorname{sen} y \cos 2(x^2 + y^2)$, no domínio $-2 \leq x \leq 2$, $-2 \leq y \leq 2$.

```
(%i128)  load(draw)$
(%i129)  grafico: explicit(exp(-x^2-y^2)*cos(x/4)*sin(y)*cos(2*(x^2+y^2)),
                           x, -2, 2, y, -2, 2)$
(%i130)  draw3d(surface_hide = true, color = orange, grafico);
(%o130)  [gr3d(explicit)]
```



Exemplo 5.44 Desenhar o gráfico do catenóide parametrizado por $x = \cosh v \cos u$, $y = \cosh v \operatorname{sen} u$, $z = v$, no domínio $-\pi \leq u \leq \pi$, $-2 \leq v \leq 2$. Usar um traço de cor amarela em fundo de cor cinza claro.

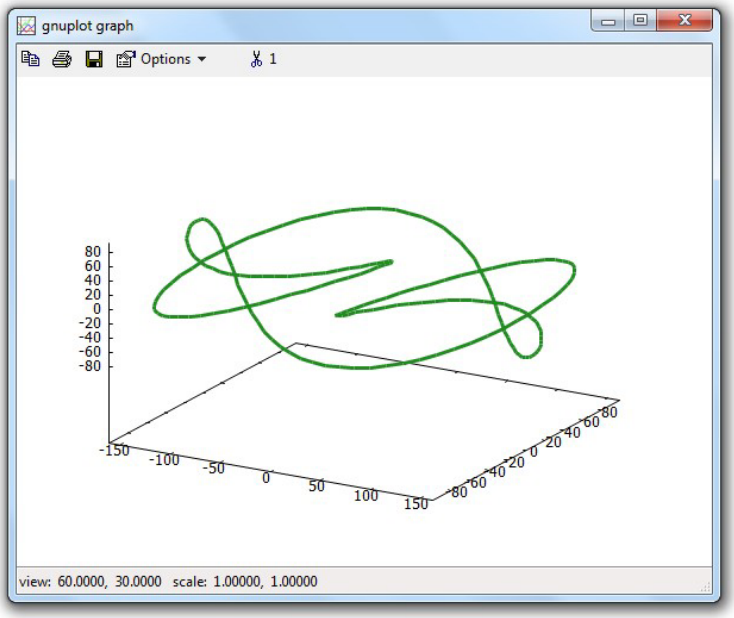
```
(%i131)  load(draw)$
(%i132)  x: cosh(v)*cos(u)$ y: cosh(v)*sin(u)$ z: v$
(%i133)  catenoide: parametric_surface(x, y, z, u, -%pi, %pi, v, -2, 2)$
(%i134)  draw3d(color=yellow, background_color=gray, surface_hide=true,
         proportional_axes=xyz, catenoide);
(%o134)  [gr3d(parametric_surface)]
```



Exemplo 5.45 Construir o gráfico da curva parametrizada por

$$\begin{aligned} x &= -22\cos t - 128\,\mathrm{sen}\, t - 44\cos 3t - 78\,\mathrm{sen}\, 3t, \\ y &= 11\cos t - 43\cos 3t + 34\cos 5t - 39\,\mathrm{sen}\, 5t, \\ z &= 70\cos 3t - 40\,\mathrm{sen}\, 3t + 8\cos 5t - 9\,\mathrm{sen}\, 5t,\ 0 \leq t \leq 2\pi. \end{aligned}$$

```
(%i135)  load(draw);
(%i136)  x: -22*cos(t) - 128*sin(t) - 44*cos(3*t) - 78*sin(3*t) $
         y: 11*cos(t) - 43*cos(3*t)+34*cos(5*t) - 39*sin(5*t)$
         z: 70*cos(3*t) - 40*sin(3*t)+8*cos(5*t) - 9*sin(5*t)$
(%i137)  draw3d(nticks=200, line_width=3, color=forest-green,
                                    parametric(x, y, z, t, 0, 2*%pi));
(%O137)  [gr3d(parametric)]
```



5.10.9 Animações gráficas

Uma animação gráfica pode ser facilmente criada com o pacote draw. Para isso, basta criar um aquivo do tipo GIF animado e, depois, visualizar o arquivo através de um programa conveniente como um programa navegador de internet, por exemplo.

Uma animação pode ser pensada como sendo uma sequência de imagens, mostradas uma após a outra, com intervalo de tempo muito pequeno entre as apresentações de cada uma delas.

Exemplo 5.46 Neste exemplo é criado um GIF animado de nome animacao.gif que é formado pelos gráficos de $\operatorname{sen} x$, $\operatorname{sen} 2x$, $\operatorname{sen} 3x$, $\operatorname{sen} 4x$, mostrados um após o outro com um intervalo de tempo de 50 centésimos de segundo entre eles. Para criar o arquivo em uma pasta específica, digamos que de nome `c:\usuários\imagens`, então o nome deveria ser digitado da seguinte forma: file_name="`c:\\usuários\\imagens\\animacao`".

```
(%i138)  load(draw)$
(%i139)  draw(delay=50, file_name="animacao", terminal=animated_gif,
             gr2d(explicit(sin(x), x, -1,1)), gr2d(explicit(sin(2*x),x,-1,1)),
             gr2d(explicit(sin(3*x),x,-1,1)), gr2d(explicit(sin(4*x),x,-1,1)));
(%o139)  [gr2d(explicit), gr2d(explicit), gr2d(explicit), gr2d(explicit)]
```

Exemplo 5.47 Os gráficos do tipo $y = x^k$ com $0 \leq x \leq 1$, $0 \leq y \leq 1$ e

$k \in \{\frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, 1, 2, 3, 4, 5\}$ são gravados em um arquivo GIF animado e é colocado um intervalo de tempo de 30 centésimos de segundo entre dois deles. O resultado final é um arquivo de nome teste2.gif que pode ser visualizado através de outros programas.

```
(%i140)  load(draw)$
(%i141)  h(k) := gr2d(line_width=3, color=orange, explicit(x^k, x, 0, 1),
                      xrange=[0,1], yrange=[0,1])$
(%i142)  draw(delay=30, file_name="teste2", terminal=animated_gif,
           h(1/5), h(1/4), h(1/3), h(1/2), h(1), h(2), h(3), h(4), h(5))$
```

Exemplo 5.48 Os gráficos do hiperbolóide definido implicitamente pela equação $x^2-y^2-z^2 = 1$ são contruídos várias vezes com os ângulos de rotações $[60, 20k]$, com k variando de 0 a 9. Assim, os ângulos de rotações são iguais a [60, 0], [60, 20], [60, 40], ..., [60, 180]. Cada um desses gráficos é gravado em um arquivo GIF animado de nome hiperboloide.gif e é colocado um intervalo de tempo de 10 centésimos de segundo entre dois deles. A animação tridimensional assim criada pode ser visualizada através de outros programas. Dependo do computador utilizado, a criação desse arquivo pode demorar um pouco, uma vez que são criados 10 gráficos tridimensionais definidos implicitamente.

```
(%i143)  load(draw)$
(%i144)  graf(k):=gr3d(proportional_axes=xyz, surface_hide=true,color=violet,
        view=[60, 20*k], implicit(x^2 - y^2 - z^2 = 1, x,-3,3, y,-3,3, z,-3,3))$

(%i145)  draw(delay=10, file_name="hiperboloide", terminal=animated_gif,
                        graf(0), graf(1), graf(2), graf(3), graf(4),
                        graf(5), graf(6), graf(7),graf(8), graf(9))$
```

A linha anterior poderia ter sido substituída por

```
(%i146)  draw(delay=10, file_name="hiperboloide", terminal=animated_gif,
                        makelist(graf(k), k, 0, 9))$
```

Por fim, execute novamente os comandos desse último exemplo trocando a equação do hiperbolóide por $x^2 + y^2 - z^2 = 1$ e os ângulos de rotações por $[20k, 60]$.

Capítulo 6

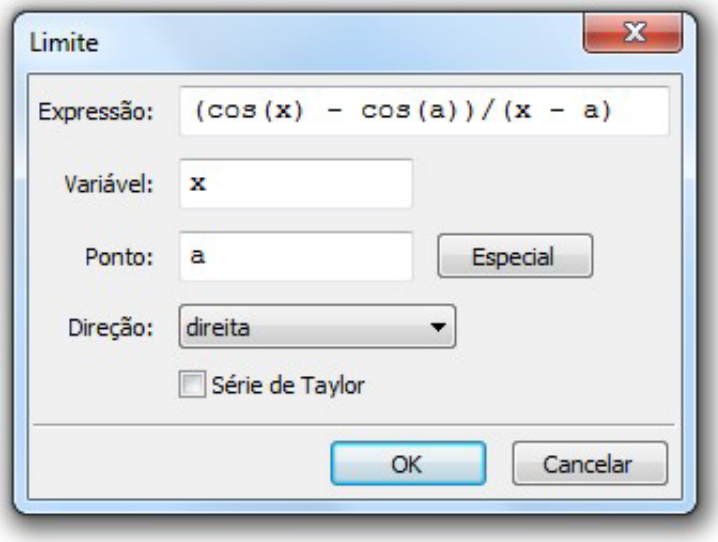
Cálculo Diferencial e Integral

O *Maxima* está habilitado a realizar os mais diversos cálculos envolvendo limites, derivadas e integrais. Neste capítulo, fazemos uma rápida introdução a esses recursos tão interessantes.

6.1 Limites

Um comando **limit(f(x), x, a)** pode ser usado para calcular $\lim_{x \to a} f(x)$. Como ancontece em outros casos, se for colocado um apóstrofo antes do nome do comando, então ele não será avaliado e será apenas enunciado.

Um limite também pode ser calculada através da opção "Cálculo → Encontrar limite" do menu principal.



Exemplo 6.1 Neste primeiro exemplo, vamos calcular $\lim_{x \to 0} \dfrac{\sin(4x)}{x}$.

```
(%i1)  limit(sin(4*x)/x, x, 0);
(%o1)  4
```

```
(%i2)  'limit(sin(4*x)/x, x, 0);
```

(%o2) $\lim_{x\to 0} \dfrac{\sin(4x)}{x}$

Exemplo 6.2 Calcular $\lim_{x\to a} \dfrac{\cos x - \cos a}{x - a}$. Usar comando sem apóstrofo e também com apóstrofo.

```
(%i3)  limit( (cos(x) - cos(a))/(x - a), x, a);
```

(%o3) $-\sin(a)$

```
(%i4)  'limit( (cos(x) - cos(a))/(x - a), x, a);
```

(%o4) $\lim_{x\to a} \dfrac{\cos(x) - \cos(a)}{x - a}$

```
(%i5)  % = ev(%, nouns);
```

(%o5) $\lim_{x\to a} \dfrac{\cos(x) - \cos(a)}{x - a} = -\sin(a)$

O comando **ev('expressão, nouns)** faz com que a expressão digitada depois do apóstrofo seja efetivamente avaliada e calculada.

Exemplo 6.3 Calcular os seguintes limites:

- $\lim_{x\to 2} \dfrac{x^3 - 8x + 8}{3x^3 - 15x^2 + 16x + 4}$
- $\lim_{t\to 2} \dfrac{\sqrt{t^2 - 4t + 5} - \sqrt{-t^2 + 4t - 3}}{(t-2)^2}$
- $\lim_{\theta\to 0} \dfrac{1 - \cos 7\theta}{\theta^2}$
- $\lim_{x\to 0} \dfrac{\sqrt[m]{1 + ax} - \sqrt[n]{1 + bx}}{x}$
- $\lim_{\beta\to 1} \dfrac{\sqrt[n]{\beta} - 1}{\sqrt[m]{\beta} - 1}$
- $\lim_{x\to 0} \dfrac{e^x + \operatorname{sen} x - 1}{\ln(1 + x)}$
- $\lim_{x\to 1} (1 + \operatorname{sen} \pi x)^{\operatorname{cotg} \pi x}$
- $\lim_{x\to 0} \dfrac{\alpha^{\operatorname{sen} x} - 1}{x}$

```
(%i6)  'limit( (x^3-8*x+8)/(3*x^3-15*x^2+16*x+4), x, 2)$ % = ev(%, nouns);
```

$$\lim_{x\to 2}\frac{x^3-8x+8}{3x^3-15x^2+16x+4}=-\frac{1}{2}$$

```
(%i7)  'limit( (sqrt(t^2 - 4*t + 5) - sqrt(-t^2 + 4*t - 3))/(t-2)^2, t, 2)$
       % = ev(%, nouns);
```

$$\lim_{t\to 2}\frac{\sqrt{t^2-4t+5}-\sqrt{-t^2+4t-3}}{(t-2)^2}=1$$

```
(%i8)  'limit((1 - cos(7*theta))/theta^2, theta, 0)$ % = ev(%, nouns);
```

$$\lim_{\theta\to 0}\frac{1-\cos(7\theta)}{\theta^2}=\frac{49}{2}$$

```
(%i9)  'limit(((1+a*x)^(1/m) - (1+b*x)^(1/n))/x, x, 0)$ %=ev(%, nouns);
```

$$\lim_{x\to 0}\frac{(1+ax)^{1/m}-(1+bx)^{1/n}}{x}=\frac{an-bm}{nm}$$

```
(%i10)  'limit((beta^(1/n)-1)/(beta^(1/m)- 1), beta, 1)$ % = ev(%, nouns);
```

$$\lim_{\beta\to 1}\frac{\beta^{1/n}-1}{\beta^{1/m}-1}=\frac{m}{n}$$

```
(%i11)  'limit( (exp(x) + sin(x) - 1)/log(1 + x), x, 0)$ % = ev(%, nouns);
```

$$\lim_{x\to 0}\frac{\sin(x)+\%e^x-1}{\log(x+1)}=2$$

```
(%i12)  'limit ((1 + sin(%pi*x))^cot(%pi*x), x, 1)$ % = ev(%, nouns);
```

$$\lim_{x\to 1}(\sin(\pi x)+1)^{\cot(\pi x)}=\%e^{-1}$$

```
(%i13)  'limit( (alpha^sin(x) - 1)/x, x, 0)$ % = ev(%, nouns);
```

$$\lim_{x\to 0}\frac{\alpha^{\sin(x)}-1}{x}=\log(\alpha)$$

6.1.1 Limites no infinito

Para o *Maxima*, o símbolo ∞ é codificado como sendo **inf** e $-\infty$ como **minf**.

Exemplo 6.4 Calcular os limites:

- $\lim_{x\to-\infty}(x^3+4x+5)$
- $\lim_{x\to-\infty}\left(1+\frac{11}{x}\right)^x$

- $\lim_{x\to-\infty}\left(\frac{x^2+1}{x^2-2}\right)^{x^2}$
- $\lim_{x\to-\infty}(x^2\sqrt{4x^4+5}-2x^4)$
- $\lim_{x\to\infty}\left(\sqrt{x^2-x-1}-\sqrt{x^2-7x+3}\right)$
- $\lim_{x\to\infty}\left(\sqrt{x+\sqrt{x}}-\sqrt{x}\right)$
- $\lim_{x\to\infty}\frac{\sqrt{x+\sqrt{x+\sqrt{x}}}}{\sqrt{x+1}}$
- $\lim_{x\to\infty}\frac{\sqrt{x}+\sqrt[3]{x}+\sqrt[4]{x}}{\sqrt{2x+1}}$

```
(%i14)  'limit(x^3 + 4*x + 5, x, minf)$ % = ev(%, nouns);
```
(%o14) $\lim_{x\to-\infty} x^3+4x+5 = -\infty$

```
(%i15)  'limit((1 + 11/x)^x, x, minf)$ % = ev(%, nouns);
```
(%o15) $\lim_{x\to-\infty}\left(\frac{11}{x}+1\right)^{x} = \%e^{11}$

```
(%i16)  'limit(((x^2 + 1)/(x^2 - 2))^(x^2), x, minf)$ % = ev(%, nouns);
```
(%o16) $\lim_{x\to-\infty}\frac{(x^2+1)^{x^2}}{(x^2-2)^{x^2}} = \%e^3$

```
(%i17)  'limit(x^2*sqrt(4*x^4 + 5) - 2*x^4, x, minf)$ % = ev(%, nouns);
```
(%o17) $\lim_{x\to-\infty} x^2\sqrt{4x^4+5}-2x^4 = \frac{5}{4}$

```
(%i18)  'limit(sqrt(x^2 - x - 1) - sqrt(x^2-7*x+3), x, inf)$ % = ev(%, nouns);
```
(%o18) $\lim_{x\to\infty}\sqrt{x^2-x-1}-\sqrt{x^2-7x+3} = 3$

```
(%i19)  'limit((sqrt(x + sqrt(x))-sqrt(x)), x, inf) $ % = ev(%, nouns);
```
(%o19) $\lim_{x\to\infty}\sqrt{x+\sqrt{x}}-\sqrt{x} = \frac{1}{2}$

```
(%i20)  'limit(sqrt(x+sqrt(x+sqrt(x)))/sqrt(x+1), x, inf)$ % = ev(%,nouns);
```
(%o20) $\lim_{x\to\infty}\frac{\sqrt{\sqrt{x+\sqrt{x}}+x}}{\sqrt{x+1}} = 1$

```
(%i21)  'limit((x^(1/2) + x^(1/3) + x^(1/4))/sqrt(2*x + 1), x, inf)$
        % = ev(%, nouns);
```

(%o21) $$\lim_{x\to\infty} \frac{\sqrt{x}+x^{1/3}+x^{1/4}}{\sqrt{2x+1}} = \frac{1}{\sqrt{2}}$$

6.1.2 Limites laterais

Limites laterais podem ser calculados se for acrescentado **plus** ou **minus** como parâmetro adicional na linha de comando do limite, conforme o limite seja lateral à direita ou à esquerda:

- **limit(f(x), x, a, plus)** para calcular $\lim_{x\to a^+} f(x)$
- **limit(f(x), x, a, minus)** para calcular $\lim_{x\to a^-} f(x)$

Exemplo 6.5 Calcular os seguintes limites:

- $\lim_{x\to\frac{\pi}{2}^+} \operatorname{tg} x$
- $\lim_{x\to\frac{\pi}{2}^-} \operatorname{tg} x$
- $\lim_{x\to 2^+} \frac{x}{x^2-4}$
- $\lim_{x\to 2^-} \frac{x}{x^2-4}$

```
(%i22) 'limit(tan(x), x, %pi/2, plus)$ % = ev(%, nouns);
```
(%o22) $$\lim_{x\to\frac{\pi}{2}+} \tan(x) = -\infty$$

```
(%i23) 'limit(tan(x), x, %pi/2, minus)$ % = ev(%, nouns);
```
(%o23) $$\lim_{x\to\frac{\pi}{2}-} \tan(x) = \infty$$

```
(%i24) 'limit(x/(x^2 - 4), x, 2, plus)$ % = ev(%, nouns);
```
(%o24) $$\lim_{x\to 2+} \frac{x}{x^2-4} = \infty$$

```
(%i25) 'limit(x/(x^2 - 4), x, 2, minus)$ % = ev(%, nouns);
```
(%o25) $$\lim_{x\to 2-} \frac{x}{x^2-4} = -\infty$$

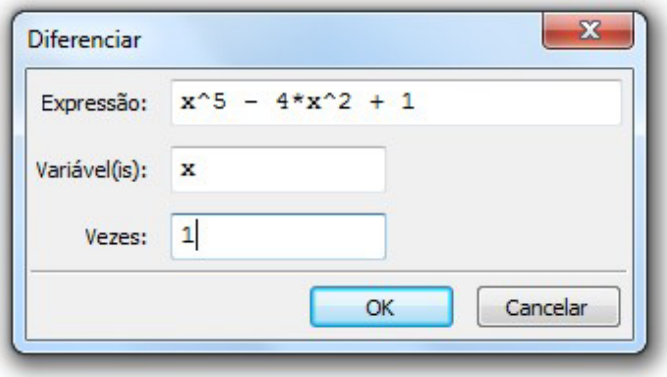
6.2 Derivadas

O *Maxima* é muito eficiente no cálculo de derivadas de funções de uma ou várias variáveis. Para realizar esse tipo de cálculo, ele possui a função **diff(...)** que conhece todas as regras usuais de derivação.

- **diff(expressão, variável)** Derivada da função definida pela expressão com relação à variável dada. Exemplo: diff(log(x), x).
- **diff(expressão, variável, n)** Derivada de ordem n da função definida pela expressão com relação à variável dada. Exemplo: diff(sin(9*x), x, 4).
- **diff(expressão, var$_1$, n$_1$, var$_2$, n$_2$, . . . , var$_s$, n$_s$)** Derivada parcial de ordem $n_1 + n_2 + \cdots + n_s$ da função definida pela expressão dada com relação às variáveis var$_1$, var$_2$, . . . , var$_s$. A derivada com relação à variável var$_k$ é de ordem n_k. Exemplo: diff(x^8 + y^5, x, 2, y, 4).
- **diff(expressão)** Diferencial total da função definida pela expressão dada. Exemplo: diff(x^8*y*z^5)
- **gradef(f(x), g(x))** Define a derivada de f(x) como sendo a função g(x). Exemplo: gradef(f(x), cos(x^2))
- **gradef(expressão, df$_1$, df$_2$, . . . , df$_s$)** Define as derivadas parciais da função de várias variáveis definida pela expressão algébrica dada como sendo df$_1$, df$_2$, . . . , df$_s$. Exemplo: gradef(f(x, y), g(x), h(y))
- **gradefs** Mostra a lista de funções com derivadas definidas

Assim como acontece em outras situações, um apóstrofo antes do nome como **'diff(. . .)** faz com que o cálculo da derivada fique apenas enunciado.

A derivada de uma função também pode ser calculada através da opção Cálculo → Diferenciar do menu principal.



Exemplo 6.6 Calcular a derivada de $x^5 - 4x^2 + 1$ com relação a x.

```
(%i26)  diff(x^5 - 4*x^2 + 1, x);
(%o26)  5x^4 - 8x

(%i27)  'diff(x^5 - 4*x^2 + 1, x);
(%o27)  d/dx (x^5 - 4x^2 + 1)
```

```
(%i28)  'diff(x^5 - 4*x^2 + 1, x) = diff(x^5 - 4*x^2 + 1, x);
```
(%o28) $\frac{d}{dx}(x^5 - 4x^2 + 1) = 5x^4 - 8x$

Exemplo 6.7 Calcule as derivadas das seguintes funções:

- $f(x) = \frac{\ln(x+1)}{1+\sqrt{3x-1}}$
- $g(\theta) = \cos 4\theta - \sec^5 \theta$
- $h(t) = e^{t^3-4t+2} - \cosh(t^4 + t)$
- $j(x) = x^{\cos x}$

```
(%i29)  'diff(log(x+1)/(1+sqrt(3*x+1)), x)$   % = ev(%, nouns);
```
(%o29) $\frac{d}{dx}\frac{\log(x+1)}{\sqrt{3x+1}+1} = \frac{1}{(x+1)(\sqrt{3x+1}+1)} - \frac{3\log(x+1)}{2\sqrt{3x+1}(\sqrt{3x+1}+1)^2}$

```
(%i30)  'diff(cos(4*theta) - sec(theta)^5, theta)$   % = ev(%, nouns);
```
(%o30) $\frac{d}{d\theta}(\cos(4\theta) - \sec(\theta)^5) = -4\sin(4\theta) - 5\sec(\theta)^5\tan(\theta)$

```
(%i31)  h(t) := exp(t^3 - 4*t+2) - cosh(t^4 + t)$
(%o31)  'diff('h(t), t) = diff(h(t), t);
```
(%i32) $\frac{d}{dt}h(t) = (3t^2 - 4)\%e^{t^3-4t+2} - (4t^3 + 1)\sinh(t^4 + t)$

```
(%o32)  j(x) := x^cos(x)$
(%i33)  'diff('j(x), x) = diff(j(x), x);
```
(%o33) $\frac{d}{dx}j(x) = x^{\cos(x)}\left(\frac{\cos(x)}{x} - \log(x)\sin(x)\right)$

Exemplo 6.8 Calcular as derivadas $f'(x)$ nos seguintes casos:

- f(x) = $g(x)h(x)$
- f(x) = $g(x)/h(x)$
- f(x) = $g(x^2)^{h(x^3)}$

```
(%i34)  'diff(g(x)*h(x), x)$   % = ev(%, nouns);
```
(%o34) $\frac{d}{dx}(g(x)h(x)) = g(x)\left(\frac{d}{dx}h(x)\right) + h(x)\left(\frac{d}{dx}g(x)\right)$

```
(%i35)  'diff(g(x)/h(x), x)$   % = ev(%, nouns);
```
(%o35) $\frac{d}{dx}\frac{g(x)}{h(x)} = \frac{\frac{d}{dx}g(x)}{h(x)} - \frac{g(x)\left(\frac{d}{dx}h(x)\right)}{h(x)^2}$

```
(%i36)  ratsimp(%);
```

```
(%o36)  d/dx g(x)/h(x) = -(g(x)(d/dx h(x)) - h(x)(d/dx g(x)))/h(x)^2
(%i37)  'diff(g(x^2)^h(x^3), x)$  % = ev(%, nouns);
(%o37)  d/dx g(x^2)^h(x^3) = g(x^2)^h(x^3) (log(g(x^2)) (d/dx h(x^3)) + h(x^3)(d/dx g(x^2))/g(x^2))
```

Exemplo 6.9 Sabendo que a derivada de uma função $f(x)$ é igual a $\sqrt{x^5+1}$, calcule a derivada de $g(x) = x^4 f(x^3) + \ln f(x)$.

```
(%i38)  gradef(f(x), sqrt(x^5 + 1));
(%o38)  f(x)
(%i39)  diff(x^4*f(x^3) + log(f(x)), x);
(%o39)  4x^3 f(x^3) + sqrt(x^5+1)/f(x) + 3x^6 sqrt(x^15 + 1)
```

Exemplo 6.10 Sabendo que a derivada de uma função $f(x)$ é igual a $g(x) + 3x^2 + 4$ e que a derivada de $g(x)$ é igual a $h(x)^2 + \operatorname{arctg} f(x)$, calcule as derivadas segunda e terceira de $f(x)$.

```
(%i40)  gradef(f(x), g(x) + 3*x^2 + 4);
(%o40)  f(x)
(%i41)  gradef(g(x), h(x)^2 + atan(f(x)));
(%o41)  g(x)
(%i42)  gradefs;
(%o42)  [f(x), g(x)]
(%i43)  'diff(f(x), x, 2) = diff(f(x), x, 2);
(%o43)  d^2/dx^2 f(x) = atan(f(x)) + h(x)^2 + 6x
(%i44)  'diff(f(x), x, 3) = diff(f(x), x, 3);
(%o44)  d^3/dx^3 f(x) = 2h(x) (d/dx h(x)) + (g(x)+3x^2+4)/(f(x)^2+1) + 6
```

Exemplo 6.11 Calcule e simplifique a derivada da função

$$f(x) = 2\sqrt{2}\operatorname{arctg}\frac{\sqrt{2}x^2}{x^4-1} + \sqrt{2}\ln\left(\frac{x^4+\sqrt{2}x^2+1}{x^4-\sqrt{2}x^2+1}\right).$$

```
(%i45)  f(x) := 2*sqrt(2)*atan(sqrt(2)*x^2/(x^4-1)) +
                sqrt(2)*log((x^4+sqrt(2)*x^2+1)/(x^4-sqrt(2)*x^2+1))$
(%i46)  diff(f(x), x);
```

(%o46) $$\frac{\sqrt{2}(x^4-\sqrt{2}x^2+1)\left(\frac{4x^3+2^{3/2}x}{x^4-\sqrt{2}x^2+1}-\frac{(4x^3-2^{3/2}x)(x^4+\sqrt{2}x^2+1)}{(x^4-\sqrt{2}x^2+1)^2}\right)}{x^4+\sqrt{2}x^2+1}+\frac{2^{3/2}\left(\frac{2^{3/2}x}{x^4-1}-\frac{2^{5/2}x^5}{(x^4-1)^2}\right)}{\frac{2x^4}{(x^4-1)^2}+1}$$

```
(%i47) ratsimp(%);
```

(%o47) $$-\frac{16x^5}{x^8+1}$$

Exemplo 6.12 Calcule a derivada de ordem 10 da função $f(x)=\frac{1}{x^2+1}$

```
(%i48) f(x) := 1/(x^2 + 1)$
(%i49) diff(f(x), x, 10);
```

(%o49) $$-\frac{3628800}{(x^2+1)^6}+\frac{217728000x^2}{(x^2+1)^7}-\frac{2032128000x^4}{(x^2+1)^8}+\frac{6502809600x^6}{(x^2+1)^9}-\frac{8360755200x^8}{(x^2+1)^{10}}+\frac{3715891200x^{10}}{(x^2+1)^{11}}$$

```
(%i50) ratsimp(%);
```

(%o50) $$\frac{39916800x^{10}-598752000x^8+1676505600x^6-1197504000x^4+199584000x^2-3628800}{x^{22}+11x^{20}+55x^{18}+165x^{16}+330x^{14}+462x^{12}+462x^{10}+330x^8+165x^6+55x^4+11x^2+1}$$

```
(%i51) factor(%);
```

(%o51) $$\frac{3628800(11x^{10}-165x^8+462x^6-330x^4+55x^2-1)}{(x^2+1)^{11}}$$

Exemplo 6.13 Calcule a derivada de ordem 8 da funçaõ $g(x)=\operatorname{tg} x$ e simplifique o resultado, substituindo $\sec x$ por $\sqrt{\operatorname{tg}^2 x+1}$.

```
(%i52) g(x) := tan(x)$
(%o52) diff(g(x), x, 8);
```

(%i53) $$128\sec(x)^2\tan(x)^7+7680\sec(x)^4\tan(x)^5+24576\sec(x)^6\tan(x)^3+7936\sec(x)^8\tan(x)$$

```
(%i54) subst(sec(x)=sqrt(tan(x)^2+1), %);
```

(%o54) $$7936\tan(x)(\tan(x)^2+1)^4+24576\tan(x)^3(\tan(x)^2+1)^3+7680\tan(x)^5(\tan(x)^2+1)^2+128\tan(x)^7(\tan(x)^2+1)$$

```
(%i55) expand(%);
```

(%o56) $$40320\tan(x)^9+120960\tan(x)^7+129024\tan(x)^5+56320\tan(x)^3+7936\tan(x)$$

Exemplo 6.14 Sendo $w = x^6\cos(y^4+5z^2)$, calcule $\frac{\partial^3 w}{\partial x\partial y\partial z}$, $\frac{\partial^7 w}{\partial x^5\partial y^2}$ e $\frac{\partial^5 w}{\partial x^2\partial y\partial z^2}$

```
(%i57)  w: x^6*cos(y^4 + 5*z^2)$
(%o57)  'diff('w, x, 1, y, 1, z, 1)$   % = ev(%, nouns);
(%i58)  d^3/(dx dy dz) w = -240*x^5*y^3*z*cos(5*z^2 + y^4)
(%i59)  'diff('w, x, 5, y, 2)$   % = ev(%, nouns);
(%o59)  d^7/(dx^5 dy^2) w = -8640*x*y^2*sin(5*z^2 + y^4) - 11520*x*y^6*cos(5*z^2 + y^4)
(%i60)  'diff('w, x, 2, y, 1, z, 2)$   % = ev(%, nouns);
(%o60)  d^5/(dx^2 dy dz^2) w = 12000*x^4*y^3*z^2*sin(5*z^2 + y^4) - 1200*x^4*y^3*cos(5*z^2 + y^4)
```

Exemplo 6.15 Sejam $\alpha = 2\,\text{arctg}\,\dfrac{1+x}{1-x}$ e $\beta = \text{arcsen}\,\dfrac{1-x^2}{1+x^2}$, onde $0 \le x < 1$. Mostre que $\alpha+\beta = \pi$.

Definindo $f(x)$ como sendo $\alpha+\beta$, vamos mostrar que f é constante, ou seja, $f'(x) = 0$ para todo x pertencente a $[0,1[$.

```
(%i61)  alpha: 2*atan((1+x)/(1-x));
(%o61)  2*atan((x+1)/(1-x))
(%i62)  beta: asin((1-x^2)/(1+x^2));
(%o62)  asin((1-x^2)/(x^2+1))
(%i63)  define(f(x), alpha + beta);
(%o63)  f(x) := asin((1-x^2)/(x^2+1)) + 2*atan((x+1)/(1-x))
(%i64)  assume(x >= 0 and x < 1);
(%o64)  [x >= 0, x < 1]
(%i65)  y: diff(f(x), x);
```

$$\text{(\%o65)}\quad \frac{2\left(\dfrac{x+1}{(1-x)^2}+\dfrac{1}{1-x}\right)}{\dfrac{(x+1)^2}{(1-x)^2}+1} + \frac{-\dfrac{2x}{x^2+1}-\dfrac{2x(1-x^2)}{(x^2+1)^2}}{\sqrt{1-\dfrac{(1-x^2)^2}{(x^2+1)^2}}}$$

```
(%i66)  radcan(y);
(%o66)  0
(%i67)  f(0);
(%o67)  %pi
```

Fica mostrado assim que f é constante porque sua derivada é nula em todo

o seu domínio $[0,1[$. Como $f(0)=\pi$, temos que $f(x)=\pi$ para todo x do seu domínio, ou seja, $\alpha+\beta=\pi$.

Exemplo 6.16 Seja $f(x,y,z)=\left(\frac{x}{y}\right)^{\ln z}\cdot\left(\frac{y}{z}\right)^{\ln x}\cdot\left(\frac{z}{x}\right)^{\ln y}$ definida para $x>0$, $y>0$ e $z>0$. Mostre que essa função é constante.

```
(%i68)  f(x, y, z) := (x/y)^log(z)*(y/z)^log(x)*(z/x)^log(y)$
(%i69)  diff(f(x, y, z), x);
```

$$(\%o69)\quad \frac{\left(\frac{x}{y}\right)^{\log(z)}\left(\frac{y}{z}\right)^{\log(x)}\left(\frac{z}{x}\right)^{\log(y)}\log(z)}{x}+\frac{\left(\frac{x}{y}\right)^{\log(z)}\log\left(\frac{y}{z}\right)\left(\frac{y}{z}\right)^{\log(x)}\left(\frac{z}{x}\right)^{\log(y)}}{x}-\frac{\left(\frac{x}{y}\right)^{\log(z)}\log(y)\left(\frac{y}{z}\right)^{\log(x)}\left(\frac{z}{x}\right)^{\log(y)}}{x}$$

```
(%i70)  radcan(%);
(%o70)  0

(%i71)  radcan(diff(f(x, y, z), y));
(%o71)  0

(%i72)  radcan(diff(f(x, y, z), z));
(%o72)  0

(%i73)  f(1, 1, 1);
(%o73)  1
```

Fica mostrado assim que as derivadas de f com relação às variáveis x, y e z são todas nulas. Logo, f é constante. Como $f(1,1,1)=1$, temos que $f(x,y,z)=1$, ou seja,

$$\left(\frac{x}{y}\right)^{\ln z}\cdot\left(\frac{y}{z}\right)^{\ln x}\cdot\left(\frac{z}{x}\right)^{\ln y}=1$$

para quaisquer x, y, z positivos.

6.3 Derivadas de funções definidas implicitamente

Uma equação em duas variáveis $f(x,y)=0$ pode definir y como função implícita de x e, neste caso, a derivada é dada por $\frac{dy}{dx}=\frac{-f_x(x,y)}{f_y(x,y)}$. Isso é válido

também no caso da função possuir mais variáveis. Por exemplo, se $f(x, y, z) = 0$ define z como função implícita de x e y, então $\frac{\partial z}{\partial x} = \frac{-f_x(x,y,z)}{f_z(x,y,z)}$ e $\frac{\partial z}{\partial y} = \frac{-f_y(x,y,z)}{f_z(x,y,z)}$.

No *Maxima*, uma equação `eq` pode ser escrita na forma

```
lhs(eq) - rhs(eq) = 0,
```

onde `lhs(eq)` e `rhs(eq)` representam o lado esquerdo e direito da equação, respectivamente. Portanto, se `eq` for uma equação que inclui as variáveis v1 e v2, e se `v1` for função implícita de `v2`, então sua derivada é dada por:

```
impdif(eq, v1, v2) := -diff(lhs(eq) - rhs(eq), v2)/
                       diff(lhs(eq) - rhs(eq), v1).
```

Exemplo 6.17 Considerando que a equação $y^2 = x^2 + \text{sen}(xy)$ defina y como função de x, calcular $\frac{dy}{dx}$.

```
(%i74) impdif(eq,v1,v2) := -diff(lhs(eq)-rhs(eq),v2)/diff(lhs(eq)-rhs(eq),v1)$

(%i75) impdif(y^2 = x^2 + sin(x*y), y, x);
```

$$(\%o75)\quad \frac{y\cos(xy) + 2x}{2y - x\cos(xy)}$$

Portanto, $\dfrac{dy}{dx} = \dfrac{y\cos(xy) + 2x}{2y - x\cos(xy)}$.

Exemplo 6.18 Considerando que a equação $x^3 + z^2 + ye^{xz} = -z\cos y$ defina z como função de x e de y, calcular $\frac{\partial z}{\partial x}$ e $\frac{\partial z}{\partial y}$.

```
(%i76) impdif(eq,v1,v2) := -diff(lhs(eq)-rhs(eq),v2)/diff(lhs(eq)-rhs(eq),v1)$

(%i77) equacao: x^3 + z^2 + y*exp(x*z)= -z*cos(y)$
(%i78) impdif(equacao, z, x);
```

$$(\%o78)\quad \frac{-yz\%e^{xz} - 3x^2}{xy\%e^{xz} + 2z + \cos(y)}$$

```
(%i79) impdif(equacao, z, y);
```

$$(\%o79)\quad \frac{\sin(y)z - \%e^{xz}}{xy\%e^{xz} + 2z + \cos(y)}$$

Observando a resposta do programa concluímos que $\dfrac{\partial z}{\partial x} = \dfrac{-yze^{xz} - 3x^2}{xye^{xz} + 2z + \cos y}$ e que $\dfrac{\partial z}{\partial y} = \dfrac{z\,\text{sen}\,y - e^{xz}}{xye^{xz} + 2z + \cos y}$.

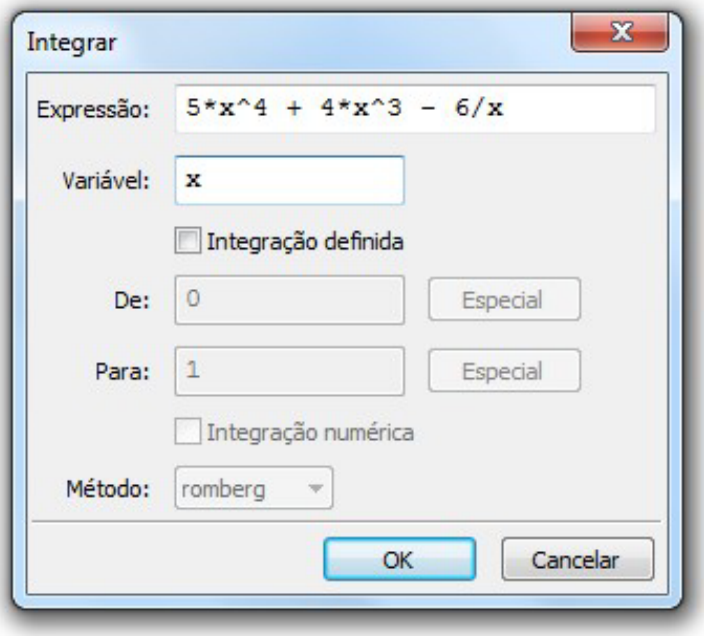
6.4 Integrais

O *Maxima* possui diversos comandos para o cálculo de integrais definidas, indefinidas ou impróprias. Ele consegue calcular desde as mais simples às mais complicadas.

- **integrate(função, variável)** Calcula a integral indefinida da função com relação à variável dada.
  Exemplo: `integrate(log(x), x)` para calcular $\int \log(x)\,dx$.
- **integrate(função, variável, a, b)** Calcula a integral definida da função com relação à variável e no intervalo $[a, b]$ dado.
  Exemplo: `integrate( x^4 + 3, x, -5, 5)` para calcular $\int_{-5}^{5} (x^4+3)\,dx$.
- **changevar(expressão, equação, var$_1$, var$_2$)** Efetuada a mudança de variável var$_2$ por var$_1$ na integral dada pela expressão, com regra de substituição definida pela equação dada.
  Exemplo: `changevar('integrate(f(x^2), x), u=x^2, u, x)` para fazer a troca de variável x por u na integral $\int f(x^2)\,dx$, baseado na equação $u = x^2$.

Assim como acontece em outras situações, um apóstrofo antes do nome como **'integrate(...)** faz com que o cálculo da integral não seja efetuado.

Uma integral também pode ser calculada através da opção Cálculo → Integrar do menu principal.



Exemplo 6.19 Calcular a integral indefinida de $5x^4 + 4x^3 - \frac{6}{x}$.

```
(%i80) integrate(5*x^4 + 4*x^3 - 6/x, x);
```
$$(\%o80)\quad -6\log(x) + x^5 + x^4$$

```
(%i81) 'integrate(5*x^4 + 4*x^3 - 6/x, x);
```
$$(\%o81)\quad \int 5x^4 + 4x^3 - \frac{6}{x}\,dx$$

```
(%i82) 'integrate(5*x^4+4*x^3-6/x, x) = integrate(5*x^4+4*x^3-6/x, x);
```
$$(\%o82)\quad \int 5x^4 + 4x^3 - \frac{6}{x}\,dx = -6\log(x) + x^5 + x^4$$

Exemplo 6.20 Calcular a integral definida de $\dfrac{x+1}{x^2+1}$ no intervalo $[0,\ 1]$

```
(%i83) integrate((x + 1)/(x^2 + 1), x, 0, 1);
```
$$(\%o83)\quad \frac{\log(2)}{2} + \frac{\pi}{4}$$

```
(%i84) 'integrate((x + 1)/(x^2 + 1), x, 0, 1);
```
$$(\%o85)\quad \int_0^1 \frac{x+1}{x^2+1}\,dx$$

```
(%i86) 'integrate((x+1)/(x^2+1), x,0,1) = integrate((x+1)/(x^2+1), x,0,1);
```
$$(\%o86)\quad \int_0^1 \frac{x+1}{x^2+1}\,dx = \frac{\log(2)}{2} + \frac{\pi}{4}$$

Exemplo 6.21 Calcular as seguintes integrais definidas:

- $\displaystyle\int_0^{\frac{\pi}{4}} \frac{1}{1+\operatorname{sen}^2 x}\,dx$
- $\displaystyle\int_1^3 \frac{1}{x^8+x^6}\,dx$
- $\displaystyle\int_a^b x^3 \operatorname{arctg} x\,dx$

```
(%i87) 'integrate(1/(1+sin(x)^2), x, 0, %pi/4)$ %=ev(%,nouns);
```
$$(\%o87)\quad \int_0^{\frac{\pi}{4}} \frac{1}{1+\operatorname{sen}^2 x}\,dx = \frac{\operatorname{atan}(\sqrt{2})}{\sqrt{2}}$$

```
(%i88) 'integrate(1/(x^8+x^6), x, 1, 3)$ %=ev(%, nouns);
```
$$(\%o88)\quad \int_1^3 \frac{1}{x^8+x^6}\,dx = \frac{15\pi+52}{60} - \frac{1215\operatorname{atan}(3)+391}{1215}$$

```
(%i89)  integrate(x^3*atan(x), x, a, b);
Is b − a positive, negative or zero?  positive
```

$$(\%o89)\quad \frac{3b^4-3\,\mathrm{atan}(b)-b^3+3b}{12}-\frac{3a^4-3\,\mathrm{atan}(a)-a^3+3a}{12}$$

Sempre que alguma extremidade do intervalo de integração não for constante, o *Maxima* perguntará se a diferença das extremidades é positiva, negativa ou nula.

Exemplo 6.22 Calcular as integrais indefinidas:

- $\displaystyle\int \frac{1}{\sqrt{x^3}\sqrt[3]{1+\sqrt[4]{x^3}}}\,dx$
- $\displaystyle\int \frac{x^2}{\sqrt{9-x^2}}\,dx$
- $\displaystyle\int \frac{4x^5-1}{x^5+x+1)^2}dx$

```
(%i90)  'integrate( 1/(sqrt(x^3)*(1 + x^(3/4))^(1/3)), x)$ %=ev(%,nouns);
```

$$(\%o90)\quad \int \frac{1}{(x^{3/4}+1)^{1/3}x^{3/2}}\,dx=-\frac{2(x^{3/4}+1)^{2/3}}{\sqrt{x}}$$

```
(%i91)  'integrate(x^2/sqrt(9 - x^2), x)$ % = ev(%, nouns);
```

$$(\%o91)\quad \int \frac{x^2}{\sqrt{9-x^2}}\,dx=\frac{9\,\mathrm{asin}\left(\frac{x}{3}\right)}{2}-\frac{x\sqrt{9-x^2}}{2}$$

```
(%i92)  'integrate( (4*x^5 - 1)/(x^5+x+1)^2, x)$ % = ev(%, nouns);
```

$$(\%o92)\quad \int \frac{4x^5-1}{(x^5+x+1)^2}\,dx=-\frac{x}{x^5+x+1}$$

Exemplo 6.23 Calcular $\displaystyle\int \frac{1}{(x^2+1)^4}\,dx$ e verificar se a resposta obtida está correta, calculando e simplificando sua derivada.

```
(%i93)  integrate(1/(x^2 + 1)^4, x);
```

$$(\%o93)\quad \frac{5\,\mathrm{atan}(x)}{16}+\frac{15x^5+40x^3+33x}{48x^6+144x^4+144x^2+48}$$

```
(%i94)  diff(%, x);
```

$$(\%o94)\quad \frac{75x^4+120x^2+33}{48x^6+144x^4+144x^2+48}-\frac{(15x^5+40x^3+33x)(288x^5+576x^3+288x)}{(48x^6+144x^4+144x^2+48)^2}+\frac{5}{16(x^2+1)}$$

```
(%i95)  ratsimp(%);
```

$$(\%o95)\quad \frac{1}{x^8+4x^6+6x^4+4x^2+1}$$

```
(%i96)  factor(%);
```

$$(\%o96)\quad \frac{1}{(x^2+1)^4}$$

Exemplo 6.24 Calcular $\int \sqrt{x^2-5x+6}\,dx$, derivar e simplificar a resposta obtida.

```
(%i97)  define(f(x), integrate(sqrt(x^2 - 5*x + 6), x));
```

$$(\%o97)\quad f(x) := -\frac{\log(2\sqrt{x^2-5x+6}+2x-5)}{8}+\frac{x\sqrt{x^2-5x+6}}{2}-\frac{5\sqrt{x^2-5x+6}}{4}$$

```
(%i98)  define(g(x), diff(f(x), x);
```

$$(\%o98)\quad g(x) := -\frac{\frac{2x-5}{\sqrt{x^2-5x+6}}+2}{8(2\sqrt{x^2-5x+6}+2x-5)}+\frac{\sqrt{x^2-5x+6}}{2}+\frac{(x(2x-5)}{4\sqrt{x^2-5x+6}}-\frac{5(2x-5)}{8\sqrt{x^2-5x+6}}$$

```
(%i99)  radcan(g(x)*sqrt(x^2 - 5*x+6));
```

$$(\%o99)\quad x^2-5x+6$$

```
(%i100)  ratsimp(%/sqrt(x^2 - 5*x + 6));
```

$$(\%o100)\quad \sqrt{x^2-5x+6}$$

Note que, nesse caso, o *Maxima* não consegue simplificar $g(x)$ diretamente. Por causa disso, usamos o artifício de simplificar $g(x)\sqrt{x^2-5x+6}$ e, no final, dividir o resultado obtido por $\sqrt{x^2-5x+6}$.

Exemplo 6.25 Calcule $F'(x)$, $G'(x)$ e $H'(x)$, sabendo que $F(x)=\int_1^x t^2\,dt$, $G(x)=\int_{4x+1}^{x^2+3} e^{t^2}\,dt$ e $H(x)=\int_{\cos x}^{x^2-\operatorname{sen} x}\sqrt{t^3+1}\,dt$.

```
(%i117)  F(x) := 'integrate(t^2, t, 1, x);
```

$$(\%o117)\quad F(x) := \int_1^x t^2\,dt$$

```
(%i118)  diff(F(x), x);
```

$$(\%o118)\quad x^2$$

```
(%i119)  G(x) := 'integrate( exp(t^2), t, 4*x+1, x^2+3);
```

$$(\%o119)\quad G(x) := \int_{4x+1}^{x^2+3} \exp(t^2)\,dt$$

```
(%i120)  diff(G(x), x);
```

(%120) $2x\%e^{(x^2+3)^2} - 4\%e^{(4x+1)^2}$

(%121) H(x) := 'integrate(sqrt(t^3 + 1), t, cos(x), x^2 - sin(x));

(%121) $H(x) := \int_{\cos(x)}^{x^2-\sin(x)} \sqrt{t^3+1}\,\mathrm{d}t$

(%122) diff(H(x), x);

(%122) $\sqrt{\cos(x)^3+1}\sin(x) + (2x-\cos(x))\sqrt{(x^2-\sin(x))^3+1}$

6.4.1 Mudança de variável

Exemplo 6.26 Aplicar as mudanças de variável $u = x^2$ e $v = x^4$ na integral $\int 4x^3 f(x^4+5)\,\mathrm{d}x$.

(%101) I: 'integrate(4*x^3*f(x^4 + 5), x);

(%101) $4\int x^3 f(x^4+5)\,\mathrm{d}x$

(%102) changevar(I, u=x^2, u, x);

(%102) $2\int uf(u^2+5)\,\mathrm{d}u$

(%103) changevar(I, v=x^4, v, x);

(%103) $\int f(v+5)\,\mathrm{d}v$

Exemplo 6.27 Calcular $\int_1^3 \frac{x^4}{1+x^{10}}\,\mathrm{d}x$ através da mudança de variável $y = x^5$.

(%104) I: 'integrate(x^4/(1 + x^10), x, 1, 3);

(%104) $\int_1^3 \frac{x^4}{x^{10}+1}\,\mathrm{d}x$

(%105) I: changevar(I, y = x^5, y, x);

(%105) $\int_1^{243} \frac{1}{5y^2+5}\,\mathrm{d}y$

(%106) ev(I, nouns);

(%106) $\frac{\mathrm{atan}(243)}{5} - \frac{\pi}{20}$

Exemplo 6.28 Denotando o logaritmo natural de x por $\log(x)$, calcular

$\int \frac{1}{x\log(x)\log(\log(x))}\,dx$ através da mudança de variável $u = \log(\log(x))$.

```
(%107)  I: 'integrate(1/(x*log(x)*log(log(x))), x);
```

$$(\%107)\quad \int \frac{1}{x\log(x)\log(\log(x))}\,dx$$

```
(%108)  changevar(I, u = log(log(x)), u, x);
```

$$(\%108)\quad \int \frac{1}{u}\,du$$

```
(%109)  ev(%, nouns);
```

$$(\%109)\quad \log(u)$$

```
(%110)  subst(u = log(log(x)), %)
```

$$(\%110)\quad \log(\log(\log(x)))$$

6.4.2 Integrais impróprias

Integrais impróprias podem ser calculadas da mesma forma da integrais definidas. Neste caso, podemos usar `inf` para representar ∞ e `minf` para representar $-\infty$ na digitação dos comandos.

Exemplo 6.29 Calcular as seguintes integrais impróprias:

- $\int_{-\infty}^{0} \frac{x}{1+x^4}\,dx$
- $\int_{-\infty}^{\infty} \frac{x^2}{1+x^6}\,dx$
- $\int_{-2}^{0} \frac{1}{\sqrt{x+2}}\,dx$
- $\int_{0}^{\infty} x^3 e^{-x^4}\,dx$
- $\int_{0}^{\infty} x^2 e^{-x^4}\,dx$

```
(%111)  'integrate(x/(1+x^4), x, minf, 0)$ % = ev(%, nouns);
```

$$(\%111)\quad \int_{-\infty}^{0} \frac{x}{x^4+1}\,dx = -\frac{\pi}{4}$$

```
(%112)  'integrate(x^2/(1 + x^6), x, minf, inf)$ % = ev(%, nouns);
```

(%112) $$\int_{-\infty}^{\infty} \frac{x^2}{x^6+1}\,dx = \frac{\pi}{3}$$

(%113) 'integrate(1/sqrt(x + 2), x, -2, 0)$ % = ev(%, nouns);

(%113) $$\int_{-2}^{0} \frac{1}{\sqrt{x+2}}\,dx = 2^{3/2}$$

(%114) 'integrate(x^3*exp(-x^4), x, 0, inf)$ % = ev(%, nouns);

(%114) $$\int_{0}^{\infty} x^3 \% e^{-x^4}\,dx = \frac{1}{4}$$

(%115) 'integrate(x^2*exp(-x^4), x, 0, inf)$ % = ev(%, nouns);

(%115) $$\int_{0}^{\infty} x^2 \% e^{-x^4}\,dx = \frac{\Gamma(\frac{3}{4})}{4}$$

(%116) float(%);

(%116) $$\int_{0.0}^{\infty} \frac{x^2}{2.718281828459045^{x^4}}\,dx = 0.30635417561629$$

6.5 Séries, somatórios e produtórios

O *Maxima* possui uma variedade de comandos para o cálculo de somatórios, produtórios e desenvolvimento de uma função em séries de potências. Entre eles, destacamos os seguintes:

- **sum(**$f(k)$**,** k**,** n_1**,** n_2**)** Somatório de $f(k)$, com k variando de n_1 a n_2. Exemplo: sum(k^3-k, k, 1, 10).
- **nusum(**$f(k)$**,** k**,** n_1**,** n_2**)** Somatório de $f(k)$, com k variando de n_1 a n_2. Exemplo: nusum(k^2+2^k, k, 0, r).
- **unsum(**$s(n)$**,** n**)** Retorna o valor $s(n+1)-s(n)$ que corresponde ao termo geral a_n tal que $\sum a_n = s(n)$. Exemplo: unsum(3^n, n).
- **product(**$f(k)$**,** k**,** n_1**,** n_2**)** Produtório de $f(k)$, com k variando de n_1 a n_2. Exemplo: product((x - 1/k), k, 1, 8).
- **powerseries(expressão, x, a)** Retorna a forma geral da expansão de séries de potência para a expressão na variável x, em torno do ponto a. Exemplo: powerseries(log(x), x, 2).
- **taylor(expressão, x, a, n)** Expande a expressão em uma série truncada de Taylor ou de Laurent na variável x em torno do ponto a, contendo termos até $(x-a)^n$. Exemplo: taylor(cos(x), x, 3, 10).

A variável global **simpsum** pode ser `true` ou `false` e controla a simplificação ou não de algumas somas. Da mesma forma, a variável **simpproduct** controla a simplificação ou não de alguns produtos, dependendo dela valer true ou false.

Se for utilizado um comando **niceindices(...)**, então em vez de mostrar índices i1, i2, i3, ... nos resultados, são mostrados i, j, k, ...

Séries, somatórios e produtórios também podem ser calculados através das opções `Cálculo` → `Obter série`, `Cálculo` → `Calcular soma` e `Cálculo` → `Calcular produto` do menu principal.

Se for colocado um apóstrofo antes do nome do comando e `simpsum` for `false`, então ele tem o seu cálculo retardado, ficando apenas o enunciado.

Exemplo 6.30 Calcular os seguinte somatórios: $\sum_{k=0}^{m}\binom{m}{k}$, $\sum_{k=0}^{n-m}\binom{m+k}{m}$ e $\sum_{k=0}^{m}\binom{m}{k}$.

```
(%123) sum( binomial(n+k, k), k, 0, m);
```

$$(\%123)\quad \sum_{k=0}^{m}\binom{m}{k}$$

```
(%124) sum( binomial(n+k, k), k, 0, m), simpsum;
```

$$(\%124)\quad \binom{n+m+1}{n+1}$$

```
(%125) sum( binomial(m + k, m), k, 0, n-m);
```

$$(\%125)\quad \sum_{k=0}^{n-m}\binom{m+k}{m}$$

```
(%126) sum( binomial(m + k, m), k, 0, n-m), simpsum;
```

$$(\%126)\quad \binom{n+1}{m+1}$$

```
(%127) sum(binomial(m, k), k, 0, m);
```

$$(\%127)\quad \sum_{k=0}^{m}\binom{m}{k}$$

```
(%128) sum(binomial(m, k), k, 0, m), simpsum;
```

$$(\%128)\quad 2^{m}$$

A variável simpsum tem o valor `false` como padrão (default). Se seu nome for acrescentado no final da linha de comando, então ela passa a ser temporariamente `true` e, por causa disso, o *Maxima* consegue calcular os somatórios mos-

trados neste exemplo. Para deixar o valor dessa variável `true` durante a sessão atual do programa, basta fazer uma atribuição da forma `simpsum: true;`

Exemplo 6.31 Calcular os somatórios $\sum_{k=1}^{n} k^6$, $\sum_{k=1}^{n} k^7$ e $\sum_{k=1}^{n} k^8$.

```
(%129) simpsum: true;
(%130) sum(k^6, k, 1, n);
```

$$(\%130)\quad \frac{6n^7+21n^6+21n^5-7n^3+n}{42}$$

```
(%131) sum(k^7, k, 1, n);
```

$$(\%131)\quad \frac{3n^8+12n^7+14n^6-7n^4+2n^2}{24}$$

```
(%132) sum(k^8, k, 1, n);
```

$$(\%132)\quad \frac{10n^9+45n^8+60n^7-42n^5+20n^3-3n}{90}$$

Exemplo 6.32 Calcular as somas das séries $\sum_{k=1}^{\infty} \frac{1}{k^4}$, $\sum_{k=1}^{\infty} \frac{1}{k^6}$ e $\sum_{k=1}^{\infty} \frac{1}{k^8}$.

```
(%133) simpsum: true;
(%134) sum(1/k^4, k, 1, inf);
```

$$(\%134)\quad \frac{\pi^4}{90}$$

```
(%135) sum(1/k^6, k, 1, inf);
```

$$(\%135)\quad \frac{\pi^6}{945}$$

```
(%136) sum(1/k^8, k, 1, inf);
```

$$(\%136)\quad \frac{\pi^8}{9450}$$

Obtemos assim que $\sum_{k=1}^{\infty} \frac{1}{k^4} = \frac{\pi^4}{90}$, $\sum_{k=1}^{\infty} \frac{1}{k^6} = \frac{\pi^6}{945}$ e $\sum_{k=1}^{\infty} \frac{1}{k^8} = \frac{\pi^8}{9450}$.

Exemplo 6.33 Calcular os somatórios $\sum_{k=1}^{n} k \cdot k!$, $\sum_{k=1}^{n} k \cdot 2^k$ e $\sum_{k=1}^{n} k^2 \cdot 2^k$.

```
(%137) nusum(k*k!, k, 1, n);
```

$$(\%137)\quad (n+1)!-1$$

```
(%138) nusum(k*2^k, k, 1, n);
```

$$(\%138)\quad (n-1)2^{n+1}+2$$

```
(%139) nusum(k^2*2^k, k, 1, n);
```
$$(\%139)\quad (n^2-2n+3)2^{n+1}-6$$

O comando `sum(...)` não consegue calcular esses somatórios.

Exemplo 6.34 Determinar a soma da série $\displaystyle\sum_{n=1}^{\infty}\frac{n^4}{4^n}$.

```
(%140) define(f(k), nusum(n^4/4^n, n, 1, k));
```
$$(\%140)\quad f(k):=\frac{95\ 2^2}{3^4}-\frac{27k^4+144k^3+360k^2+528k+380}{81\ 4^k}$$

```
(%141) 'limit('f(k), k, inf) = limit(f(k), k, inf);
```
$$(\%141)\quad \lim_{k\to\infty} f(k)=\frac{380}{81}$$

Concluímos dessa forma que $\displaystyle\sum_{n=1}^{\infty}\frac{n^4}{4^n}=\frac{380}{81}$.

Exemplo 6.35 Determinar $f(k)$ e $g(k)$ tais que $\displaystyle\sum_{k=1}^{n} f(k)=\frac{n(n+1)(2n+1)}{6}$ e $\displaystyle\sum_{k=1}^{n} g(k)=\frac{n^2(n+1)^2}{4}$.

```
(%142) unsum(n*(n+1)*(2*n+1)/6, n);
```
$$(\%142)\quad n^2$$

```
(%143) unsum(n^2*(n+1)^2/4, n);
```
$$(\%143)\quad n^3$$

Observando as respostas do programa, concluímos que $f(k)=k^2$ e $g(k)=k^3$.

Exemplo 6.36 Calcular os produtórios $\displaystyle\prod_{k=1}^{5}\left(x+\frac{k(k+1)}{2}\right)$, $\displaystyle\prod_{k=1}^{n} k$ e $\displaystyle\prod_{1}^{6}(2k)$.

```
(%144) product(x + k*(k+1)/2, k, 1, 5);
```
$$(\%144)\quad (x+1)(x+3)(x+6)(x+10)(x+15)$$

```
(%145) expand(%)
```
$$(\%145)\quad x^5+35x^4+427x^3+2193x^2+4500x+2700$$

```
(%146) product(k, k, 1, n);
```
$$(\%146)\quad \prod_{k=1}^{n} k$$

```
(%147)  product(k, k, 1, n), simpproduct;
```
$$(\%147)\quad n!$$

```
(%148)  product(2*k, k, 1, 6);
```
$$(\%148)\quad 46080$$

```
(%149)  'product(2*k, k, 1, 6) = product(2*k, k, 1, 6);
```
$$(\%149)\quad \prod_{k=1}^{6} 2k = 46080$$

Exemplo 6.37 Obter os primeiros termos do desenvolvimento em série de Taylor (ou Laurent) das funções $f(x) = e^x$, $g(x) = x^3 \operatorname{tg} x$ e $h(x) = \frac{\operatorname{sen} x}{x^8}$, em torno de alguns pontos como $x = 0$, $x = a$, $x = \frac{\pi}{4}$ ou $x = \pi$.

```
(%160)  taylor(exp(x), x, 0, 8);
```
$$(\%160)/T/\quad 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \frac{x^6}{720} + \frac{x^7}{5040} + \frac{x^8}{40320} + \ldots$$

```
(%161)  taylor(exp(x), x, a, 4);
```
$$(\%161)/T/\quad \%e^a + \%e^a(x-a) + \frac{\%e^a(x-a)^2}{2} + \frac{\%e^a(x-a)^3}{6} + \frac{\%e^a(x-a)^4}{24} + \ldots$$

```
(%162)  taylor(x^2*tan(x), x, %pi/4, 3);
```
$$(\%162)/T/\quad \frac{\pi^2}{16} + \frac{(\pi^2+4\pi)(x-\frac{\pi}{4})}{8} + \frac{(\pi^2+8\pi+8)(x-\frac{\pi}{4})^2}{8} + \frac{(\pi^2+6\pi+12)(x-\frac{\pi}{4})^3}{6} + \ldots$$

```
(%163)  taylor(x^2*tan(x), x, %pi, 4);
```
$$(\%163)/T/\quad \pi^2(x-\pi) + 2\pi(x-\pi)^2 + \frac{(\pi^2+3)(x-\pi)^3}{3} + \frac{2\pi(x-\pi)^4}{3} + \ldots$$

```
(%164)  taylor(sin(x)/x^8, x, 0, 5);
```
$$(\%164)/T/\quad \frac{1}{x^7} - \frac{1}{6x^5} + \frac{1}{120x^3} - \frac{1}{5040x} + \frac{x}{362880} - \frac{x^3}{39916800} + \frac{x^5}{6227020800} + \ldots$$

O /T/ mostrado em cada caso indica que a série está "truncada".

Exemplo 6.38 Obter o desenvolvimento em série de potências das seguintes funções:

- $f(x) = \ln x$, em torno de $x = 2$
- $g(x) = x \operatorname{arctg} x + x^3 e^x$, em torno de $x = 0$
- $h(x) = \cos x$, em torno de $x = 3$

```
(%165)  powerseries(log(x), x, 2);
```
$$(\%165)\quad \sum_{i1=0}^{\infty} \frac{2^{-i1-1}(-1)^{i1}(x-2)^{i1+1}}{i1+1}$$

```
(%166)  niceindices(%);
```

(%166) $$\sum_{i=0}^{\infty} \frac{2^{-i-1}(-1)^i(x-2)^{i+1}}{i+1}$$

```
(%167) powerseries(x*atan(x)+x^3*exp(x), x, 0);
```

(%o167) $$x\left(\sum_{i2=0}^{\infty} \frac{(-1)^{i2}x^{2\,i2+1}}{2\,i2+1}\right) + x^3 \sum_{i2=0}^{\infty} \frac{x^{i2}}{i2!}$$

```
(%168) powerseries(cos(x), x, 3);
```

(%168) $$\cos(3)\left(\sum_{i3=0}^{\infty} \frac{(-1)^{i3}(x-3)^{2\,i3}}{(2\,i3)!}\right) - \sin(3)\sum_{i3=0}^{\infty} \frac{(-1)^{i3}(x-3)^{2\,i3+1}}{(2\,i3+1)!}$$

```
(%169) niceindices(%);
```

(%169) $$\cos(3)\left(\sum_{i=0}^{\infty} \frac{(-1)^{i}(x-3)^{2\,i}}{(2\,i)!}\right) - \sin(3)\sum_{i=0}^{\infty} \frac{(-1)^{i}(x-3)^{2\,i+1}}{(2\,i+1)!}$$

O comando `powerseries(...)` só consegue resolver os casos mais simples de desenvolvimento em séries de potências.

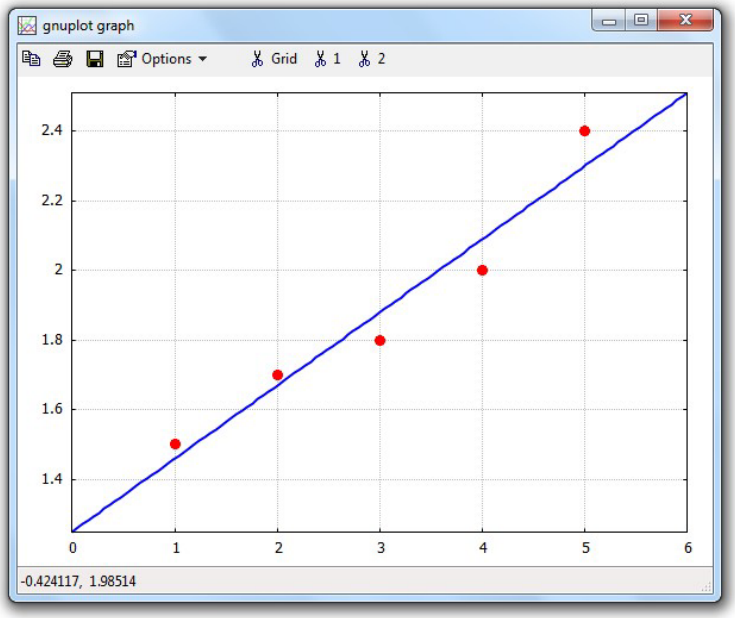
Exemplo 6.39 Dado um conjunto de pontos (1, 1, 5), (2, 1, 7), (3, 1, 8), (4, 2) e (5, 2, 4), determinar a reta dos mínimos quadrados que se ajusta a eles e o seu gráfico.

Segundo o método dos mínimos quadrados, a reta que mais se aproxima dos n pontos (x_i, y_i), com $i = 1, 2, \ldots, n$, tem equação $y = ax + b$, onde $a = \dfrac{n\sum_{i=1}^{n} x_i y_i - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$ e $b = \dfrac{\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i}{n}$.

```
(%150) Px: [1, 2, 3, 4, 5]$
(%151) Py: [1.5, 1.7, 1.8, 2, 2.4]$
(%152) n: length(Px)$
(%153) sx: sum(Px[i], i, 1, n)$ sy: sum(Py[i], i, 1, n)$
(%154) sxy: sum(Px[i]*Py[i], i, 1, n)$ sx2: sum(Px[i]^2, i, 1, n)$
(%155) a: (n*sxy - sx*sy)/(n*sx2 - sx^2)$ b: (sy - a*sx)/n$
(%156) define(f(t), a*t+b);
```

(%156) $f(t) := 0.21t + 1.25$

```
(%157) load(draw)$
(%158) gr: explicit(f(t), t, 0, 6)$ pontos: points(Px, Py)$
(%159) draw2d(grid=true, line_width=2, gr, color=red, point_size=1,
                point_type=7, pontos)$
```



6.6 Equações diferenciais

Uma equação diferencial pode ser definida no *Maxima* usando-se um apóstrofo antes de cada comando que calcula derivadas. Sendo assim, a derivada enésima de y é fornecida na forma `'diff(y, x, n)`. Por exemplo,

$$y''' - 2y'' + y' = 3y + 5$$

deve ser fornecida na forma

```
'diff(y, x, 3) - 2*'diff(y, x, 2) + 'diff(y, x) = 3*y + 5
```

Este assunto pode se estender bastante, mas, aqui, vamos usar apenas os seguintes comandos do *Maxima* para resolver equações diferenciais ordinárias:

- **ode2(equação, y, x)** Resolve uma equação diferencial ordinária de ordem 1 ou 2, onde y é função de x. Quando bem sucedido, esse comando retorna uma equação nas variáveis x, y e constantes genéricas $\%c$, $\%k1$ ou $\%k2$. Exemplo: `sol: ode2('diff(y,x)+y=3*x+1, y, x)`

- **ic1(solução,** $x = x_0, y = y_0$**)** Calcula o valor da constante apresentada na solução de uma EDO de primeira ordem baseando-se no fato de que $y = y_0$ quando $x = x_0$, ou seja, $y(x_0) = y_0$. Exemplo: `ic1(sol, x=0, y=-1)`

- **ic2(solução,** $x = x_0, y = y_0, diff(y, x) = y_1$**)** Calcula os valores das constantes apresentadas na solução de uma EDO de segunda ordem baseando-se no fato de que $y = y_0$ e $y' = y_1$ quando $x = x_0$, ou seja, $y(x_0) = y_0$ e $y'(x_0) = y_1$. Exemplo: `ic2(sol, x=0, y=5, diff(y,x)=4)`

- **bc2(solução,** $x = x_0, y = y_0, x = x_1, y = y_1$**)** Calcula os valores das constantes apresentadas na solução de uma EDO de segunda ordem baseando-se no fato de que $y = y_0$ quando $x = x_0$ e que $y = y_1$ quando $x = x_1$, que é o mesmo que $y(x_0) = y_0$ e $y(x_1) = y_1$.
  Exemplo: `bc2(sol, x=1, y=2, x=3, y=-1)`

Exemplo 6.40 Determinar a solução do problema de valor inicial $y'' + 4y = 0$, $y(\pi/8) = 0$, $y(\pi/6) = 1$.

```
(%170)  eq: 'diff(y,x,2) + 4*y = 0;
```
(%171) $\frac{d^2}{dx^2}y + 4y = 0$

```
(%172)  s: ode2(eq, y, x);
```
(%172) $y = \%k1 \sin(2x) + \%k2 \cos(2x)$

```
(%173)  bc2(s, x=%pi/8, y=0, x=%pi/6, y=1);
```
(%173) $y = (\sqrt{3} + 1)\sin(2x) + (-\sqrt{3} - 1)\cos(2x)$

Exemplo 6.41 Resolver: $y' = \dfrac{x^2 + y^2}{xy}$, $y(1) = -2$.

```
(%174)  eq2: 'diff(y, x) = (x^2 + y^2)/(x*y);
```
(%174) $\frac{d}{dx}y = \frac{y^2 + x^2}{xy}$

```
(%175)  s2: ode2(eq2, y, x);
```
(%175) $\frac{y^2 - 2x^2 \log(x)}{2x^2} = \%c$

```
(%176)  ic1(s2, x=1, y=-2);
```
(%176) $\frac{y^2 - 2x^2 \log(x)}{2x^2} = 2$

Exemplo 6.42 Resolver: $y'' + 10y' + 21y = 0$, $y(0) = -3$, $y'(0) = 41$.

```
(%177)  eq3: 'diff(y,x,2) + 10*'diff(y,x) + 21*y=0;
```
(%177) $\frac{d^2}{dx^2}y + 10\left(\frac{d}{dx}y\right) + 21y = 0$

```
(%178)  s3: ode2(eq3, y, x);
```
(%178) $y = \%k1\%e^{-3x} + \%k2\%e^{-7x}$

```
(%179)  ic2(s3, x=0, y=-3, diff(y,x)=41);
```
(%179) $y = 5\%e^{-3x} - 8\%e^{-7x}$

Exemplo 6.43 Determinar a solução geral da equação $y'' + y = x \operatorname{sen} x$.

```
(%180)  eq4: 'diff(y, x, 2) + y = x*sin(x);
```
$$(\%180)\quad \frac{d^2}{dx^2}y + y = x\sin(x)$$
```
(%181)  ode2(eq4, y, x);
```
$$(\%181)\quad y = \frac{2x\sin(x) + (1-2x^2)\cos(x)}{8} + \%k1\sin(x) + \%k2\cos(x)$$

Portanto, a solução geral encontrada é

$$y = \frac{2x\operatorname{sen} x + (1-2x^2)\cos x}{8} + k_1 \operatorname{sen} x + k_2 \cos x,$$

onde k_1 e k_2 são constantes genéricas.

Testando agora essa solução encontrada:

```
(%182)  define(f(x), rhs(%));
```
$$(\%182)\quad f(x) := \frac{2x\sin(x) + (1-2x^2)\cos(x)}{8} + \%k1\sin(x) + \%k2\cos(x)$$
```
(%183)  f(x) + diff(f(x), x, 2);
```
$$(\%183)\quad \frac{6x\sin(x) - (1-2x^2)cos(x)}{8} + \frac{2x\sin(x) + (1-2x^2)\cos(x)}{8}$$
```
(%184)  ratsimp(%);
```
$$(\%184)\quad x\sin(x)$$

que é o segundo membro da equação diferencial dada, e, assim, podemos concluir que a solução geral encontrada está correta.

6.7 Séries de Fourier

Consideremos uma função $f(x)$ definida em um intervalo $[-L, L]$. A *série de Fourier* de $f(x)$ é a série trigonométrica

$$f(x) = a_0 + \sum_{n=1}^{\infty} \left(a_n \cos \frac{n\pi}{L}x \ + \ b_n \operatorname{sen} \frac{n\pi}{L}x \right),$$

onde

$$a_0 = \frac{1}{2L}\int_{-L}^{L} f(x)\,\mathrm{d}x,$$

$$a_n = \frac{1}{L}\int_{-L}^{L} f(x)\cos\frac{n\pi}{L}x\,dx,$$

$$b_n = \frac{1}{L}\int_{-L}^{L} f(x)\,\text{sen}\,\frac{n\pi}{L}x\,dx.$$

O pacote "fourie" do *Maxima* possui vários comandos que podem ser usados para determinar esse tipo de série:

- **fourier(f(x), x, L)** Coeficientes de Fourier de $f(x)$ no intervalo $[-L, L]$.
- **fourcos(f(x), x, L)** Coeficientes a_n da extensão par de $f(x)$ no intervalo $[-L, L]$.
- **foursin(f(x), x, L)** Coeficientes b_n da extensão ímpar de $f(x)$ no intervalo $[-L, L]$.
- **fourexpand(coef, x, L, n)** Os n primeiros termos da série de Fourier usando os coeficientes obtidos com os comandos `fourier(...)`, `fourcos(...)` ou `foursin(...)`.

Exemplo 6.44 Obter o desenvolvimento em série de Fourier de cossenos da extensão par da função $f(x) = x$ no intervalo $[-\pi, \pi]$.

```
(%i185) f(x) := x$
(%i186) load(fourie)$
(%i187) S: fourcos(f(x), x, %pi);
```

(%t188) $a_0 = \dfrac{\pi}{2}$

(%t189) $a_n = \dfrac{2\left(\frac{\pi\sin(\pi n)}{n} + \frac{\cos(\pi n)}{n^2} - \frac{1}{n^2}\right)}{\pi}$

(%o190) [%t188, %t189]

```
(%i191) fourexpand(S, x, %pi, inf);
```

(%o191) $\dfrac{2\displaystyle\sum_{n=1}^{\infty}\left(\frac{(-1)^n}{n^2} - \frac{1}{n^2}\right)\cos(nx)}{\pi} + \dfrac{\pi}{2}$

```
(%i192) fourexpand(S, x, %pi, n);
```

(%o192) $\dfrac{2\displaystyle\sum_{n=1}^{n}\left(\frac{(-1)^n}{n^2} - \frac{1}{n^2}\right)\cos(nx)}{\pi} + \dfrac{\pi}{2}$

```
(%i193) fourexpand(S, x, %pi, 10);
```

(%o193) $$-\frac{4\cos(9x)}{81\pi}-\frac{4\cos(7x)}{49\pi}-\frac{4\cos(5x)}{25\pi}-\frac{4\cos(3x)}{9\pi}-\frac{4\cos(x)}{\pi}+\frac{\pi}{2}$$

Exemplo 6.45 Desenvolver a extensão ímpar da função $f(x) = x$ em série de Fourier de senos no intervalo $[-\pi, \pi]$.

```
(%i194) f(x) := x$
(%i195) load(fourie)$
(%i196) T: foursin(f(x), x, %pi);
```

(%t197) $$b_n = \frac{2\left(\frac{\sin(\pi n)}{n^2}-\frac{\pi\cos(\pi n)}{n}\right)}{\pi}$$

(%o198) [%t197]

```
(%i199) fourexpand(T, x, %pi, inf);
```

(%o199) $$-2\sum_{n=1}^{\infty}\frac{(-1)^n\sin(nx)}{n}$$

```
(%i200) fourexpand(T, x, %pi, 10);
```

(%o200) $$-\frac{\sin(10x)}{5}+\frac{2\sin(9x)}{9}-\frac{\sin(8x)}{4}+\frac{2\sin(7x)}{7}-\frac{\sin(6x)}{3}+\frac{2\sin(5x)}{5}-\frac{\sin(4x)}{2}+\frac{2\sin(3x)}{3}-\sin(2x)+2\sin(x)$$

Exemplo 6.46 Considerando a função $f(x) = x - 3$ no intervalo $[-6, 6]$, obtenha o desenvolvimento em série de Fourier de $f(x)$ e construa os gráficos de $f(x)$ e duas somas parciais da série ($k = 5$ e $k = 30$ termos).

```
(%i201) load(fourie)$
(%i202) f(x) := x - 3$
(%i203) coef: fourier(f(x), x, 6);
```

(%t203) $$a_0 = -3$$

(%t204) $$a_n = -\frac{6\sin(\pi n)}{\pi n}$$

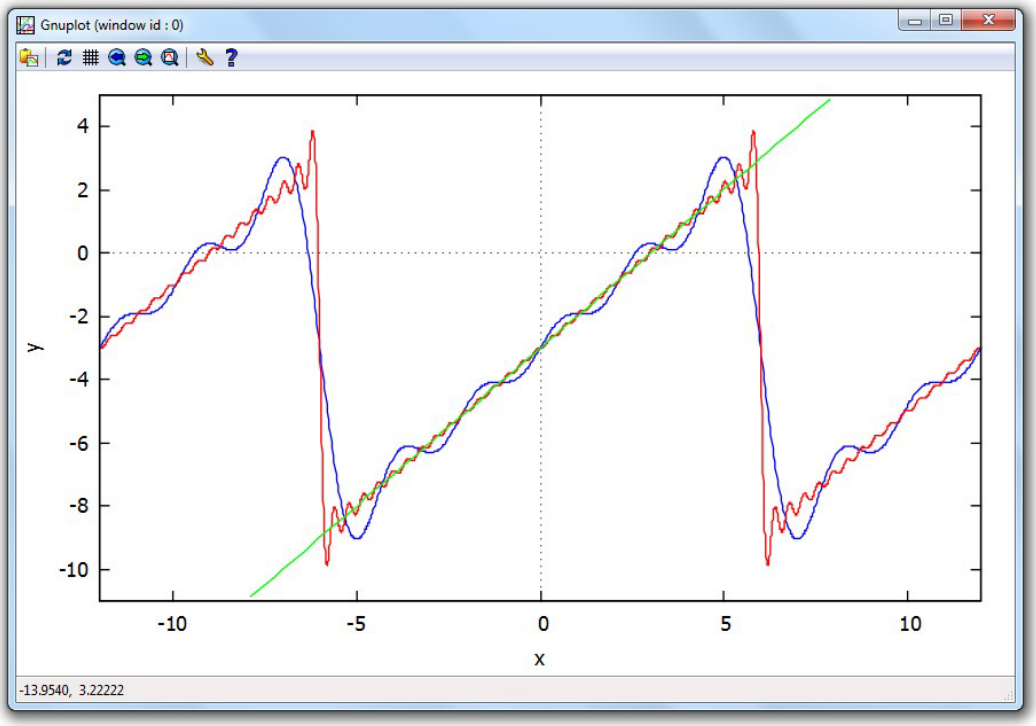
(%t205) $$b_n = \frac{\frac{72\sin(\pi n)}{\pi^2 n^2}-\frac{72\cos(\pi n)}{\pi n}}{6}$$

(%o205) [%t203, %t204, %t205]

```
(%i206) define(g(x, k), fourexpand(coef, x, 6, k));
```

$$\text{(\%o206)}\quad g(x,n) := -\frac{12\sum_{n=1}^{k}\frac{(-1)^n \sin(\frac{\pi n x}{6})}{n}}{\pi} - 3$$

```
(%i207) plot2d([g(x,5), g(x,30), f(x)], [x,-12,12], [y,-11,5], [legend, false]);
```



Exemplo 6.47 Considerando a função $f(x) = 1$ no intervalo $[-5, 5]$, obtenha o desenvolvimento em série de Fourier de senos de $f(x)$ e construa os gráficos de duas somas parciais da série ($k = 5$ e $k = 40$ termos).

```
(%i208) load(fourie)$
(%i209) f(x) := 1$
(%i210) senos: foursin(f(x), x, 5);
```

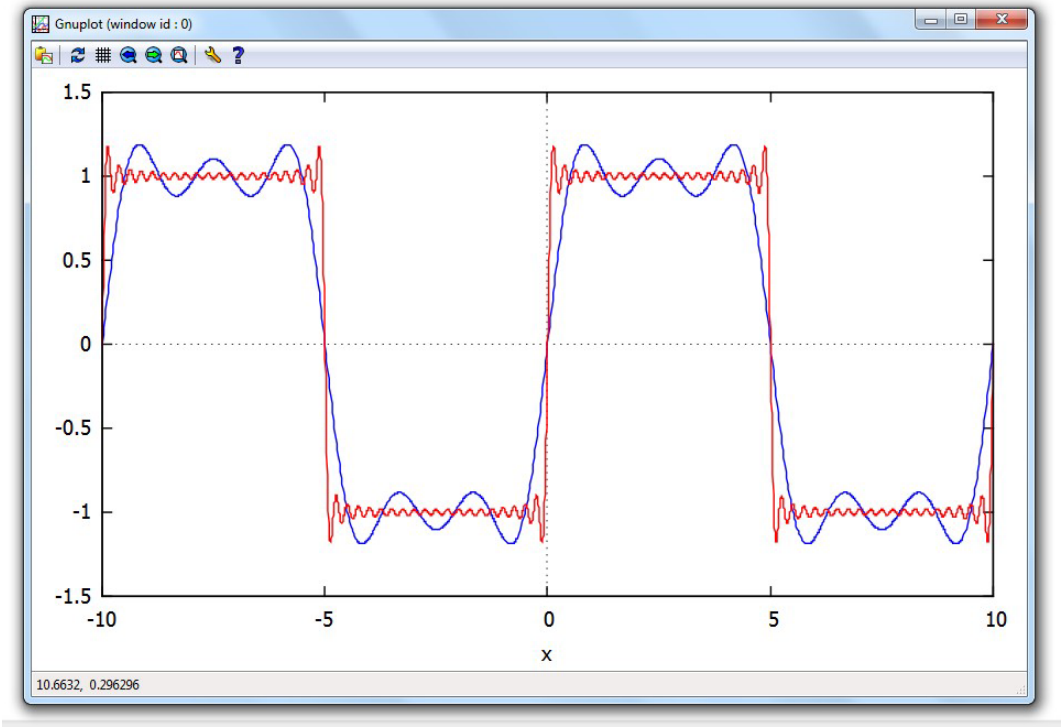
$$\text{(\%t210)}\quad b_n = \frac{2\left(\frac{5}{\pi n} - \frac{5\cos(\pi n)}{\pi n}\right)}{5}$$

$$\text{(\%o210)}\quad [\%t210]$$

```
(%i211) define(h(x, k), fourexpand(senos, x, 5, k));
```

$$\text{(\%o211)}\quad h(x,n) := \frac{2\sum_{n=1}^{k}\left(\frac{5}{\pi n} - \frac{5(-1)^n}{\pi n}\right)\sin\left(\frac{\pi n x}{5}\right)}{5}$$

```
(%i212) plot2d([h(x, 5), h(x, 40)], [x, -10, 10], [legend, false]);
```



6.8 Transformada de Laplace

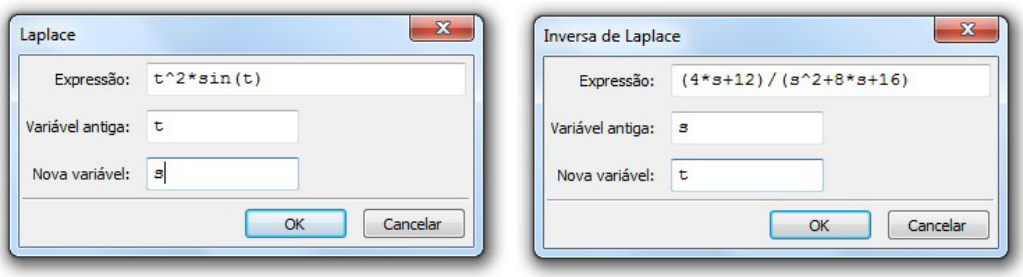
A *transformada de Laplace* de uma função $F(t)$, denotada por $\mathscr{L}(F(t))$ é uma função $f(s)$ definida por

$$\mathscr{L}(F(t)) = f(s) = \int_0^\infty e^{-st} F(t)\,\mathrm{d}t$$

Nesse caso, dizemos que F(t) é a *transformada inversa de Laplace* de f(s) e denotamos por $F(t) = \mathscr{L}^{-1}(f(s))$. O *Maxima* tem esses conceitos implementados nos seguintes comandos:

- **laplace(F(t), t, s)** Transformada de Laplace de $F(t)$ na variável s.
- **ilt(f(s), s, t)** Transformada inversa de Laplace de $f(s)$ na variável t.

A transformada de Laplace também pode ser calculada através da opção `Cálculo` → `Transformada de Laplace` do menu principal.



Exemplo 6.48 Calcular a transformada de Laplace de cada uma das sguintes funções: $e^{3t}\cos 5t$, $t^2 \operatorname{sen} t$ e $t^5 + 7t^3 - 11 + 4e^t$.

```
(%i213) laplace( exp(3*t)*cos(5*t), t, s);
```
$$(\%o213)\quad \frac{s-3}{s^2-6s+34}$$

```
(%i214) laplace(t^2*sin(t), t, s);
```
$$(\%o214)\quad \frac{8s^2}{(s^2+1)^3} - \frac{2}{(s^2+1)^2}$$

```
(%i215) laplace(t^5 + 7*t^3 - 11 + 4*exp(t), t, s);
```
$$(\%o215)\quad -\frac{11}{s} + \frac{42}{s^4} + \frac{120}{s^6} + \frac{4}{s-1}$$

Exemplo 6.49 Sendo f uma função de uma variável, calcular as transformadas $\mathscr{L}(\frac{d}{dt}f(t))$, $\mathscr{L}(\frac{d^2}{dt^2}f(t))$ e $\mathscr{L}(\int_0^t f(x)\,dx)$.

```
(%i216) laplace('diff(f(t), t), t, s);
```
$$(\%o216)\quad s\ \text{laplace}(\text{f(t)},\text{t},\text{s}) - f(0)$$

```
(%i217) laplace('diff(f(t), t, 2), t, s);
```
$$(\%o217)\quad -\frac{d}{dt}f(t)\,|_{t=0} + s^2\ \text{laplace}(\text{f(t)},\text{t},\text{s}) - f(0)s$$

```
(%i218) laplace('integrate(f(x), x, 0, t), t, s);
```
$$(\%o218)\quad \frac{\text{laplace}(\text{f(t)},\text{t},\text{s})}{s}$$

Exemplo 6.50 Calcular as seguintes transformadas inversas de Laplace: $\mathscr{L}^{-1}\left(\frac{4s+12}{s^2+8s+16}\right)$, $\mathscr{L}^{-1}\left(\frac{6s-4}{s^2-4s+20}\right)$ e $\mathscr{L}^{-1}\left(\frac{s+1}{s^2+s+1}\right)$.

```
(%i219) ilt((4*s + 12)/(s^2 + 8*s + 16), s, t);
```
$$(\%o219)\quad 4\%e^{-4t} - 4t\%e^{-4t}$$

```
(%i220) ilt((6*s - 4)/(s^2 - 4*s + 20), s, t);
```
$$(\%o220)\quad \%e^{2t}(2\sin(4t) + 6\cos(4t))$$

```
(%i221) ilt((s + 1)/(s^2 + s + 1), s, t);
```
$$(\%o221)\quad \%e^{-\frac{t}{2}}\left(\frac{\sin\left(\frac{\sqrt{3}t}{2}\right)}{\sqrt{3}} + \cos\left(\frac{\sqrt{3}t}{2}\right)\right)$$

No cálculo da transformada de Laplace da derivada de uma função, aparecem valores da função ou suas derivadas calculadas em um ponto: $f(0)$, $f'(0)$, $f''(0)$ etc. Esses valores podem ser definidos se for usado um comando **atvalue(função, equação, valor)**. Para definir $f(a) = b$, por exemplo, podemos usar um comando `atvalue(f(x), x=a, b)`, e para definir $f^{(n)}(a) = c$ podemos usar um `atvalue('diff(f(x), x, n), x=a, c)`.

Exemplo 6.51 Seja $g(t)$ uma função na qual $g(0) = \alpha$, $g'(0) = \beta$, $g''(0) = \gamma$ e $g'''(0) = \delta$. Calcular $\mathscr{L}(g'''(t))$ e $\mathscr{L}(g^{(4)}(t))$.

```
(%i222) atvalue(g(t), t=0, %alpha)$
(%i223) atvalue('diff(g(t), t), t=0, %beta)$
(%i224) atvalue('diff(g(t), t, 2), t=0, %gamma)$
(%i225) atvalue('diff(g(t), t, 3), t=0, %delta)$
(%i226) laplace('diff(g(t), t, 3), t, s);
```

(%o226) $s^3\,\text{laplace}(g(t), t, s) - \alpha s^2 - \beta s - \gamma$

```
(%i227) laplace('diff(g(t), t, 4), t, s);
```

(%o227) $s^4\,\text{laplace}(g(t), t, s) - \alpha s^3 - \beta s^2 - \gamma s - \delta$

Exemplo 6.52 Resolver a equação diferencial

$$y''(t) + 2y'(t) + 5y(t) = e^{-t}\,\text{sen}\,t$$

com as seguintes restrições: $y(0) = 0$ e $y'(0) = 1$.

Para resolver um problema de valor inicial como esse, podemos aplicar a transformada de Laplace aos dois membros da equação, depois isolar o valor de $\mathscr{L}(y(t)) = f(s)$ e, no final, calcular a transformada inversa $\mathscr{L}^{-1}(f(s)) = y(t)$ que é a solução procurada.

```
(%i228) eqdif: 'diff(y(t), t, 2) + 2*'diff(y(t), t) + 5*y(t) = exp(-t)*sin(t);
```

(%o228) $\frac{d^2}{dt^2}y(t) + 2(\frac{d}{dt}y(t)) + 5y(t) = \%e^{-t}\sin(t)$

```
(%i229) atvalue(y(t), t=0, 0)$
(%i230) atvalue('diff(y(t), t), t=0, 1)$
(%i231) laplace(eqdif, t, s);
```

(%o231) $s^2\,\text{laplace(y(t),t,s)} + 2s\,\text{laplace(y(t),t,s)} + 5\,\text{laplace(y(t),t,s)} - 1$

$$= \frac{1}{s^2 + 2s + 2}$$

```
(%i232) solve(%, laplace(y(t),t,s));
```

(%o232) $[\text{laplace(y(t),t,s)} = \dfrac{s^2 + 2s + 3}{s^4 + 4s^3 + 11s^2 + 14s + 10}]$

```
(%i233) ladodireitoeq: rhs(%[1]);
```

$$(\%o233)\quad \frac{s^2+2s+3}{s^4+4s^3+11s^2+14s+10}$$

```
(%i234) ilt(ladodireitoeq, s, t);
```

$$(\%o234)\quad \frac{\%e^{-t}\sin(2t)}{3}+\frac{\%e^{-t}\sin(t)}{3}$$

Portanto, a solução procurada é $y(t)=\dfrac{e^{-t}\operatorname{sen}2t}{3}+\dfrac{e^{-t}\operatorname{sen}t}{3}$.

Capítulo 7

Noções de Programação

O *Maxima* possui uma variedade razoável de comandos que podem ser usados para programação nas mais diversas situações. Esses comandos podem ser classificados em várias categorias:

- Blocos e procedimentos: o comando **block(...)** permite o agrupamento de comandos dando origem a novos comandos, com todas as estruturas básicas de programação como por exemplo variáveis locais.
- Comandos de entrada e saída de dados: **read, print, display** podem ser usados para ler ou imprimir mensagens, associadas a outros comandos ou dentro de blocos.
- Estruturas condicionais: o comando **if ... then ... else ...** pode ser usado na construção das mais diversas estruturas condicionais, executadas somente se determinadas hipóteses forem satisfeitas.
- Estruturas de repetição: comandos como **for, while, unless, do ...** podem ser usados para criar os mais diversos tipos de estruturas de repetição que são comandos que podem ser executados repetidamente várias vezes.

Com tudo isso, o *Maxima* também pode ser usado como uma linguagem de programação na qual podem ser elaborados os mais diversos tipos de programas.

7.1 Comentários

Para o *Maxima*, tudo o que for digitado entre **/*** e ***/** será considerado um comentário e não será avaliado pelo programa, servindo apenas para orientação do usuário.

Exemplo 7.1 Exemplificamos aqui dois comentários: "o valor de x é 1" e "e o valor de y é 2".

```
(%i1)  x: 1 /* o valor de x é 1 */ ;
(%o1)  1

(%i2)  y: 2 /* e o valor de y é 2 */ ;
(%o2)  2
```

7.2 Os comandos print e display

- **print(expr1, expr2, ...)** Avalia e mostra expr1, expr2, ... uma após a outra. As mensagens mostradas devem ser fornecidas entre aspas. Por exemplo, `print("O valor de x é ", 2 + 3 + 5)` mostra a mensagem: O valor de x é 10.
- **display(expr1, expr2, ...)** Mostra expr1, expr2, ... no formato de equação. Do lado esquerdo de cada equação mostrada aparece a expressão não avaliada, e do lado direito, a expressão avaliada. Por exemplo, se x valer 3 e y valer 8, então `display(x, y)` mostra as equações $x = 3$ e $y = 8$.

O comando `print` devolve o valor da última expressão mostrada, enquanto que `display` finaliza sempre com palavra "`done`".

Exemplo 7.2 Sejam x, y e z variáveis cujos valores são 2, 4 e 6, respectivamente. Usando o `display` e, depois, o `print`, mostrar os valores de x, y, z, $x+y+z$ e $\cos(x+y)$.

```
(%i1)  x: 2$ y: 4$ z: 6$
(%i2)  display(x, y, z, x+y+z, cos(x+y));
```

$x = 2$
$y = 4$
$z = 6$
$2 + 4 + 6 = 12$
$\cos(6) = \cos(6)$

```
(%o2)  done

(%i3)  print(x, y, z, x+y+z, cos(x+y));
2 4 6 12  cos(6)
(%o3)  cos(6)

(%i4)  print("O valor de x é ", x);
O valor de x é 2
(%o4)  2
```

```
(%i5)  print("y vale ", y, "e z vale ", z);
```
y vale 4 e z vale 6
```
(%o5)  6
```

```
(%i6)  print("O valor de x + y + z é ", x + y + z);
```
O valor de $x + y + z$ é 12
```
(%o6)  12
```

```
(%i7)  print("O valor de cos(x+y) é ", cos(x+y));
```
O valor de $\cos(x + y)$ é cos(6)
```
(%o7)  cos(6)
```

7.3 O comando read

O comando **read(expr1, expr1, ...)** avalia e mostras as expressões expr1, expr2, ..., lê uma expressão fornecida e retorna o seu valor avaliado. Exemplo: `a:  read("Forneça o valor de a:  ");`

Exemplo 7.3 O valor de uma variável x é igual a 3. Mostrar uma mensagem citando esse valor e solicitar que seja fornecido pelo teclado um novo valor.

```
(%i8)  x: 3$
(%i9)  x: read("O valor atual de x é ", x, ". Forneça um novo valor: ");
O valor atual de x é 3. Forneça um novo valor:  7;
(%o9)  7

(%i10)  x;
(%o10)  7
```

Exemplo 7.4 Solicitar que sejam fornecidos dois polinômios p e q pelo teclado e calcular sua soma.

```
(%i11)  p: read("Polinômio p? ");
Polinômio p?  x^3 + 5*x^2 - 4*x + 11;
(%o11)  x^3 + 5*x^2 - 4*x + 11

(%i12)  q: read("Polinômio q? ");
Polinômio q?  x^2 + 10*x - 5;
(%o12)  x^2 + 10*x - 5

(%i13)  p + q;
```

(%o13) $x^3 + 6x^2 + 6x + 6$

7.4 O comando for

Para executar um comando ou uma sequência de comandos repetidamente, podemos usar um comando for que possui algumas variantes:

- **for** `variável` **from** `valor_inicial` **thru** `valor_final` **do** `comando`**;**
  O `comando` à direita de `do` é executado para cada valor da variável do `valor_inicial` até o `valor_final`.
- **for** `variável` **from** `valor_inicial` **while** `condição` **do** `comando`**;**
  O `comando` à direita de `do` é executado para cada valor da variável a partir do `valor_inicial`, enquanto a `condição` for verdadeira.
- **for** `variável` **from** `valor_inicial` **unless** `condição` **do** `comando`**;**
  O `comando` à direita de `do` é executado para cada valor da variável a partir do `valor_inicial`, até que a `condição` seja verdadeira.
- **for** `variável` **in** `lista` **do** `comando`**;** O `comando` à direita de `do` é executado para cada valor da variável na lista de valores fornecida.

O padrão (default) do comando for é incrementar a variável de 1 em 1 unidade, para que ela possa assumir todos os valores do valor inicial até o valor final fornecidos. Para usar um passo diferente de 1, basta acrescentar um **step** `incremento` à linha de comando. Se `incremento` for negativo, então a sequência de valores assumidos pela variável será decrescente.

Onde aparece um comando à direita de `do`, também pode aparecer uma sequência de comandos entre parênteses e separados por vírgulas como em (`comando`$_1$`, comando`$_2$`, ..., comando`$_n$).

Exemplo 7.5 Listar as raízes quadradas dos inteiros de 4 a 10.

```
(%i8)  for n from 4 thru 10 do display(sqrt(n));
```

$\sqrt{4} = 2$
$\sqrt{5} = \sqrt{5}$
$\sqrt{6} = \sqrt{6}$
$\sqrt{7} = \sqrt{7}$
$\sqrt{8} = 2^{3/2}$
$\sqrt{9} = 3$
$\sqrt{10} = \sqrt{10}$

```
(%o8)  done
```

```
(%i9)  for n from 4 thru 10 do print(sqrt(n));
```

$$2$$
$$\sqrt{5}$$
$$\sqrt{6}$$
$$\sqrt{7}$$
$$2^{3/2}$$
$$3$$
$$\sqrt{10}$$

```
(%o9)  done
```

```
(%i10)  for n from 4 thru 10 do print(float(sqrt(n)));
2.0
2.23606797749979
2.449489742783178
2.645751311064591
2.828427124746191
3.0
3.16227766016838
(%o10)  done
```

```
(%i11)  for n from 4 thru 10 do
          print("A raiz quadrada de ", n, "é ", float(sqrt(n)));
A raiz quadrada de 4 é 2.0
A raiz quadrada de 5 é 2.23606797749979
A raiz quadrada de 6 é 2.449489742783178
A raiz quadrada de 7 é 2.645751311064591
A raiz quadrada de 8 é 2.828427124746191
A raiz quadrada de 9 é 3.0
A raiz quadrada de 10 é 3.16227766016838
(%o11)  done
```

Exemplo 7.6 Listar os inteiros de -17 a 10 com passo 3, ou seja, de 3 em 3.

```
(%i12)  for x from -17 step 3 while x < 10 do display(x);
```

$$x = -17$$
$$x = -14$$
$$x = -11$$
$$x = -8$$
$$x = -5$$

$x = -2$
$x = 1$
$x = 4$
$x = 7$
(%o12) *done*

O comando anterior é equivalente aos seguintes:

- `for x from -17 while x < 10 step 3 do display(x);`
- `for x from -17 step 3 unless x >= 10 do display(x);`
- `for x from -17 unless x >= 10 do step 3 display(x);`

Exemplo 7.7 Fatorar os polinômios $x^n + x + 1$, com n variando de 10 a 20.

```
(%i13) for n from 10 thru 20 do display(factor(x^n+x+1));
```
$\text{factor}(x^{10} + x + 1) = x^{10} + x + 1$
$\text{factor}(x^{11} + x + 1) = (x^2 + x + 1)(x^9 - x^8 + x^6 - x^5 + x^3 - x^2 + 1)$
$\text{factor}(x^{12} + x + 1) = x^{12} + x + 1$
$\text{factor}(x^{13} + x + 1) = x^{13} + x + 1$
$\text{factor}(x^{14} + x + 1) = (x^2 + x + 1)(x^{12} - x^{11} + x^9 - x^8 + x^6 - x^5 + x^3 - x^2 + 1)$
$\text{factor}(x^{15} + x + 1) = x^{15} + x + 1$
$\text{factor}(x^{16} + x + 1) = x^{16} + x + 1$
$\text{factor}(x^{17}+x+1) = (x^2+x+1)(x^{15}-x^{14}+x^{12}-x^{11}+x^9-x^8+x^6-x^5+x^3-x^2+1)$
$\text{factor}(x^{18} + x + 1) = x^{18} + x + 1$
$\text{factor}(x^{19} + x + 1) = x^{19} + x + 1$
$\text{factor}(x^{20} + x + 1) = (x^2 + x + 1)(x^{18} - x^{17} + x^{15} - x^{14} + x^{12} - x^{11} + x^9 - x^8 + x^6 - x^5 + x^3 - x^2 + 1)$
(%o13) *done*

Exemplo 7.8 Dadas duas listas L1 e L2 de mesmo comprimento, substituir $L1[k]$, o k-ésimo elemento de L1, por $\frac{L1[k]+L2[k]}{2}$, a média aritmética entre o k-ésimo elemento de L1 e o k-ésimo elemento de L2. Depois do cálculo de cada média, atribuir o valor 0 a cada elemento de L2.

```
(%i14) L1: [5, 7, 1, 1, -3, 4, 10]$
       L2: [1, 2, 0, 8, 1, 2, 6]$
(%i15) for k from 1 thru length(L1) do (
           x: L1[k], y: L2[k],
           z: (x + y)/2,
           L1[k]: z, L2[k]: 0);
```

```
(%o15)  done

(%i16)  display(L1, L2);
```
$L1 = [3, \frac{9}{2}, \frac{1}{2}, \frac{9}{2}, -1, 3, 8]$
$L2 = [0, 0, 0, 0, 0, 0, 0]$
```
(%o16)  done
```

Exemplo 7.9 O **googol** é definido como sendo o número 10^{100}, ou seja, é o algarismo 1 seguido de cem zeros no sistema decimal. Determine qual é o menor valor do inteiro positivo n cujo fatorial é maior do que 1 googol.

```
(%i14)  googol: 10^100$ n: 1$
(%i15)  while n! < googol do n: n+ 1;
(%o15)  done

(%i16)  n;
(%o16)  70

(%i17)  is(70! > googol);
(%o17)  true

(%i18)  is(69! > googol);
(%o18)  false
```

Concluímos dessa forma que o menor inteiro n para o qual $n! > 10^{100}$ é $n = 70$.

Exemplo 7.10 Dada uma lista L de números, calcular a soma dos quadrados dos elementos de L. Para calcular qualquer somatório, podemos iniciar sempre com uma `soma = 0`.

```
(%i19)  L: [4, 5, 0, -1, -2, 3, 4, 5, -7]$
(%i20)  soma: 0$
(%i21)  for x in L do soma: soma + x^2$
(%i22)  soma;
(%o22)  145
```

Exemplo 7.11 Sabe-se que

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k} \left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right).$$

Calcule a soma de todos os termos desse somatório que são maiores do que $\varepsilon = 10^{-12}$.

```
(%i23) f(k) := 1/16^k*(4/(8*k+1)-2/(8*k+4)-1/(8*k+5)-1/(8*k+6));
```

(%o23) $$f(k) := \frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} + \frac{-1}{8k+5} + \frac{-1}{8k+6}\right)$$

```
(%i24) soma: 0$ epsilon: 10^-12$
(%i25) for k from 0 while abs(f(k)) > epsilon do soma: soma + f(k);
(%o25) done
(%i26) soma;
```

(%o26) $$\frac{1633115836009679979817751 2637}{5198369158853231585211187200}$$

```
(%i27) float(%);
(%o27) 3.141592653588972
```

Se quiséssemos ver quais os termos que foram somados, bastaria trocar o `soma: soma + f(k);` anterior por `(soma: soma + f(k), print(f(k)));`

Exemplo 7.12 Os comandos for podem ser encaixados, ou seja, podemos ter um for "dentro" de outro. Isso normalmente é necessário quando temos vários índices envolvidos i, j, k, ... Neste exemplo, calculamos o valor do somatório $p = \sum_{i=1}^{5}\sum_{j=1}^{i} ix^j$. Note que j variar de 1 até i é o mesmo que variar de i até 1.

```
(%i28) p: 0$
(%i29) for i from 1 thru 5 do
           for j from i step -1 thru 1 do (
               p: p + i*x^j,
               print("i = ", i, "j = ", j, "p = ", p));
```

$i = 1\ j = 1\ p = x$
$i = 2\ j = 2\ p = 2x^2 + x$
$i = 2\ j = 1\ p = 2x^2 + 3x$
$i = 3\ j = 3\ p = 3x^3 + 2x^2 + 3x$
$i = 3\ j = 2\ p = 3x^3 + 5x^2 + 3x$
$i = 3\ j = 1\ p = 3x^3 + 5x^2 + 6x$
$i = 4\ j = 4\ p = 4x^4 + 3x^3 + 5x^2 + 6x$
$i = 4\ j = 3\ p = 4x^4 + 7x^3 + 5x^2 + 6x$
$i = 4\ j = 2\ p = 4x^4 + 7x^3 + 9x^2 + 6x$
$i = 4\ j = 1\ p = 4x^4 + 7x^3 + 9x^2 + 10x$

$i = 5\ j = 5\ p = 5x^5 + 4x^4 + 7x^3 + 9x^2 + 10x$
$i = 5\ j = 4\ p = 5x^5 + 9x^4 + 7x^3 + 9x^2 + 10x$
$i = 5\ j = 3\ p = 5x^5 + 9x^4 + 12x^3 + 9x^2 + 10x$
$i = 5\ j = 2\ p = 5x^5 + 9x^4 + 12x^3 + 14x^2 + 10x$
$i = 5\ j = 1\ p = 5x^5 + 9x^4 + 12x^3 + 14x^2 + 15x$
(%o29) *done*

```
(%i30) p;
```
(%o30) $5x^5 + 9x^4 + 12x^3 + 14x^2 + 15x$

Portanto, a soma desses termos é $5x^5 + 9x^4 + 12x^3 + 14x^2 + 15x$.

O *Maxima* tem implementado um comando `sum(...)` para calcular somatórios. O somatório desse exemplo poderia ser calculado facilmente com um comando `sum(sum(i*x^j, j, 1, i), i, 1, 5);`

Exemplo 7.13 Dado uma função de x em forma de expressão algébrica e um inteiro positivo n, determinar o polinômio de Taylor de grau n em torno de $x = 0$.

```
(%i31) fx: atan(x)$ n: 20$
(%i32) p: subst(x = 0, fx)$
(%i33) for k from 1 while k <= n do
         p: p + subst(x = 0, fx: diff(fx, x)/k)*x^k;
```
(%o33) *done*

```
(%i34) p;
```
(%o34) $$-\frac{x^{19}}{19} + \frac{x^{17}}{17} - \frac{x^{15}}{15} + \frac{x^{13}}{13} - \frac{x^{11}}{11} + \frac{x^9}{9} - \frac{x^7}{7} + \frac{x^5}{5} - \frac{x^3}{3} + x$$

Exemplo 7.14 Dado um valor $a > 0$, a sequência em que x_1 é próximo de $\sqrt{a}$ e $x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right)$, $n = 1, 2, 3, \ldots$ converge para $\sqrt{a}$. Sendo $a = 2$ e $x_1 = 1$, calcular vários termos dessa sequência x_2, x_3, ... e encerrar os cálculos quando $\Delta = |x_{n+1} - x_n| < \varepsilon = 10^{-8}$. Dessa forma, o último valor calculado é uma excelente aproximação de $\sqrt{a}$, obtido com poucas operações aritméticas.

```
(%i35) a: 3$ epsilon: 10^-8$ delta: 1$ x1: 1.0$
(%i39) while delta > epsilon do (
           x : 1/2*(x1 + a/x1),
           delta: abs(x - x1),
           x1: x )$
(%i40) x;
```

```
(%o40)  1.732050807568877

(%i41)  sqrt(3.0);
(%o41)  1.732050807568877
```

Comandos equivalentes

Um comando **while condição do ...** é considerado equivalente a **unless not condição do ...**. Pelo mesmo motivo, um comando **unless condição do ...** é equivalente a **while not condição do ...**. Isso significa que `while` e `unless` podem ser trocados um pelo outro, desde que seja feita a negação da condição. Por exemplo,

- `for n from 10 while n < 16 do print(n)`
- `for n from 10 unless n >= 16 do print(n)`

são equivalentes porque ambos mostram a mesma lista de inteiros: 10, 11, 12, 13, 14, 15. Note que `n >= 16` é a negação de `n < 16`.

Incremento variável

Para que a variável assuma os valores seguindo uma lei de formação de acordo com uma função fornecida, deve-se usar um **next** `função(variável)` na linha de comando do comando for. Por exemplo,

- **for n from 1 thru 100 next 2*n do ...** faz n variar a partir de 1, dobrando o seu valor a cada passo; dessa forma, n assume os valores 1, 2, 4, 8, 16, 32 e 64, todos pertencentes ao intervalo [1, 100].
- **for n from 1 thru 100 next 4*n do ...** faz a variável n assumir os valores 1, 4, 16 e 64 do intervalo [1, 100].
- **for n from 1 thru 100 next n+1 do ...** é equivalente a **for n from 1 thru 100 do ...** e faz o valor de n variar de 1 a 100, de 1 em 1.

Omissão de parte da linha de comando

O **for** `variável` **from** `valor_inicial` **...** também pode ser usado na forma **for** `variável:valor_inicial` **...**, ou seja, pode-se usar dois pontos no lugar do `from`.

Se o valor inicial da variável for igual a 1, então esse valor pode ser omitido. Assim, **for variável from 1 ...** é equivalente a **for variável ...**

Diversas partes da linha de comando podem ser omitidas. Por exemplo, comandos como

- `thru valor_final do comando`,
- `while condição do comando`,
- `unless condição do comando`

fazem sentido para o *Maxima*.

As linhas de comando a seguir são consideradas equivalentes e todas elas listam os inteiros de 1 a 6:

- `for n from 1 thru 6 step 1 do print(n);`
- `for n from 1 step 1 thru 6 do print(n);`
- `for n from 1 thru 6 do print(n);`
- `for n:  1 thru 6 do print(n);`
- `for n thru 6 do print(n);`
- `n:  1; thru 6 do (print(n:  n+1);`
- `n:  1; while n <= 6 do (print(n:  n+1);`
- `n:  1; unless n > 6 do (print(n:  n+1);`

Quando o `for` é omitido, um comando do tipo `n:  n+1` precisa ser acrescentado para poder incrementar o valor de n. O comando `print(n:  n+1)` é o mesmo que `(print(n), n:  n+1)`.

7.5 O comando if

O comando **if** pode ser usado sempre que um comando for executado condicionalmente, ou seja, somente quando determinadas condições forem satisfeitas. Sua sintaxe é

if `condição` **then** $\texttt{comando}_1$ **else** $\texttt{comando}_2$

e funciona da seguinte forma:

- A `condição` é avaliada, ou seja, é calculado um valor verdadeiro ou falso para ela.
- Se a condição for verdadeira, então o $\texttt{comando}_1$ será executado e $\texttt{comando}_2$ será ignorado.

- Se a condição for falsa, então o `comando`$_2$ será executado e `comando`$_1$ será ignorado.

O `else comando`$_2$ pode ser omitido de modo que o if pode ser usado na forma simplificada

if `condição` **then** `comando`$_1$

Nesse caso, o `comando`$_1$ será executado somente se a `condição` for verdadeira.

Um comando if pode ser usado "dentro" de outro if, ou seja, é permitido os if's serem encaixados: **if ... then if ... then ...**. No caso de if's encaixados, o **else if** pode ser usado na forma **elseif**, como se fosse um novo comando.

Exemplo 7.15 Considere os seguintes comandos de atribuição de valores às variáveis x e a. Se x for menor do que 5, então é atribuido o valor 3 à variável a; caso contrário, ou seja se $x \geq 5$, então é atribuído o valor 7 a a. Como $x = 10 > 5$, então é realizada a atribuição de valor $a = 7$ que pode ser comprovada com o `display(a)` no final.

```
(%i42)  x: 10 $
(%i43)  if x < 5 then a: 3 else a: 7 $
(%i44)  display(a)$
```
$a = 7$

Exemplo 7.16 Dados valores de x e y, interpretando esses valores como coordenadas de um ponto P no plano cartesiano, identificar a qual região do plano (quadrante) esse ponto pertence.

```
(%i45)  x: 2$  y: -3$
(%i46)  if x = 0 or y = 0 then print("P pertence a um dos eixos")
        elseif x > 0 then
            if y > 0 then print("P pertence ao primeiro quadrante")
            else print("P pertence ao quarto quadrante")
        else
            if y > 0 then print("P pertence ao segundo quadrante")
            else print("P pertence ao terceiro quadrante")$
```
P pertence ao quarto quadrante

Exemplo 7.17 Definir a seguinte função f:

$$f(x) = \begin{cases} 3x - 5 & \text{, se} \quad x < -4 \text{ ou } x > 6 \\ 4 - x & \text{, se} \quad -4 \leq x < 0 \\ 1 - 2x & \text{, se} \quad 0 \leq x < 2 \\ 4x + 1 & \text{, se} \quad 2 \leq x \leq 6 \end{cases}$$

```
(%i47)  f(x) := if x < -4 or x > 6 then 3*x - 5
                elseif -4 <= x and x < 0 then 4 - x
                elseif 0 <= x and x < 2 then 1 - 2*x
                elseif 2 <= x and x <= 6 then 4*x + 1 $
```

Exemplo 7.18 Listar todos os números primos do intervalo [1000000, 1000200]. Para isso, utilizamos a função primep(x) que é verdadeira (true) somente se x for primo.

```
(%i48)  for p from 1000000 thru 1000200 do
            if primep(p) then print(p, "é primo");
```

```
1000003 é primo
1000033 é primo
1000037 é primo
1000039 é primo
1000081 é primo
1000099 é primo
1000117 é primo
1000121 é primo
1000133 é primo
1000151 é primo
1000159 é primo
1000171 é primo
1000183 é primo
1000187 é primo
1000193 é primo
1000199 é primo
```

Exemplo 7.19 Conhecida uma lista L, contar quantos elementos da lista pertencem a um intervalo $[a, b]$. Para isso, usamos um `for x in L ...` para percorrer a lista e um `if a <= x and x <= b then ...` para testar se $x \in [a, b]$. Para cada x nesse intervalo, incrementamos de uma unidade uma variável auxiliar `cont` cujo valor antes de iniciar a contagem é 0.

```
(%i49)  L: [-3, 1, 0, 12, 100, 1000, 25, 17, -4, -6, 2, 6, 16, 11, 9, -9, 8]$
(%i50)  cont: 0$ a: 0$ b: 20$
(%i51)  for x in L do if a <= x and x <= b then cont: cont+1$
(%i52)  cont;
```

```
(%o52)  10
```

Observamos, no final, que a lista L tem 10 elementos no intervalo $[0, 20]$.

Exemplo 7.20 Dada uma lista L, calcular o valor máximo de seus elementos. Inicialmente, supomos que o maior elemento max é o primeiro elemento da lista, ou seja, que max = L[1]. Percorremos toda a lista L com um `for k ...` e comparamos max com o elemento L[k]. Sempre que for encontrado um L[k] maior do que max, é feita uma redefinição de max ao qual é atribuído o valor L[k]. No final, max será o maior valor da lista.

```
(%i53)  L: [-5, -3, 1, 4, 7, 19, 200, -100, 12, 17, -4, 8]$
(%i54)  max: L[1]$
(%i55)  for k from 1 thru length(L) do
            if max < L[k] then max: L[k]$
(%i56)  max;
(%o56)  200
```

Exemplo 7.21 Dada uma lista L, calcular o valor mínimo de seus elementos. Inicialmente, supomos que o menor elemento min é o primeiro elemento da lista, ou seja, que min = L[1]. Percorremos toda a lista L com um `for k ...` e comparamos min com o elemento L[k]. Sempre que for encontrado um L[k] menor do que min, é feita uma redefinição de min ao qual é atribuído o valor L[k]. No final, min será o menor valor da lista.

```
(%i57)  L: [-5, -3, 1, 4, 7, 19, 200, -100, 12, 17, -4, 8]$
(%i58)  min: L[1]$
(%i59)  for k from 1 thru length(L) do
            if min > L[k] then min: L[k] $
(%i60)  min;
(%o60)  −100
```

7.6 Bloco de comandos

Diversos comandos podem ser agrupados em um bloco e, depois, podem ser executados se for chamado o nome do bloco ao qual ele é atribuído. Os comandos precisam ser "envolvidos" por um block(...) da seguinte forma:

block([variável$_1$, variável$_2$, ...**],** comando$_1$, comando$_2$, ...**)**

O bloco de comandos pode ter variáveis locais que só são conhecidas dentro do bloco se seus nomes estiverem entre colchetes no início do bloco. O nome do bloco deve ter parâmetros (variáveis) e, na sua definição, deve ser usado um operador ":=", semelhante à definição de uma função. Por exemplo:
`BlocoF(x, y) := block([z, w], z: log(x), w: log(y), z+w);`
define um bloco com nome "BlocoF" que admite dois parâmetros x e y e que possui duas variáveis locais z e w. Quando esse bloco é chamado, é calculado o valor de log(x) e atribuído a z, é calculado log(y) e atribuído a w e, no final, é calculada a soma de z+w que é o valor que esse bloco retorna para o lugar de onde ele foi chamado. Ainda com relação a esse exemplo, se for usado um comando M : BlocoF(2, 3), então o valor de M passará a ser log(2)+log(3) depois da execução do bloco.

O valor que um bloco retorna é sempre o último valor calculado, a não ser que tenha um comando `return(valor)` no bloco. Nesse caso, o valor retornado é o que estiver especificado no `return`.

Exemplo 7.22 Definir um comando `CalculaSoma()` que solicite do usuário dois valores *a* e *b* e calcule a sua soma. Neste caso, *a* e *b* podem ser números inteiros, reais ou complexos, polinômios ou expressões algébricas, listas de mesmo comprimento etc.

```
(%i61)  CalculaSoma() := block(
            [a, b, s],
            a: read("Valor de a = "),
            b: read("Valor de b = "),
            s: a + b,
            print("A soma dos valores é ", s))$
```

Utilizando agora o comando assim definido:

```
(%i62)  CalculaSoma();
Valor de a =  3 + 5*%i;
Valor de b =  4 + 6*%i;
A soma dos valores é 11*%i + 7
(%o62)  11 * %i + 7
```

Exemplo 7.23 Definir um comando `tabela(f, a, b)` que construa uma tabela de valores de $f(x)$, com x variando de a a b.

```
(%i61)  tabela(f, a, b) := block([x],
            if a > b then return("ERRO: a deve ser menor ou igual a b"),
            fpprintprec: 6,
            print("x F(x) "),
            print("=========="),
            for x from a thru b do print(x, float(f(x))),
            print("=========="),
            fpprintprec: 0,
            return(true))$
```

Depois de definido, esse comando pode ser utilizado. Para isso, basta usar o nome de uma função conhecida como parâmetro, bem como as extremidades *a* e *b* de um intervalo. Para reduzir o número de casas decimais impressas, ajustamos a variável global `fpprintprec` que controla a quantidade de algarismos significativos mostrados. No final, o bloco de comandos retorna um valor `true` se ele tiver sido completamente executado com sucesso.

```
(%i62)  f(x) := sqrt(x)/(1 + x^2);
(%i63)  tabela(f, 16, 10);
(%o63)  ERRO: a deve ser menor ou igual a b

(%i64)  tabela(f, 10, 16);
x  F(x)
=========
10  0.03131
11  0.02719
12  0.02389
13  0.02121
14  0.01899
15  0.01714
16  0.01556
=========
(%o64)  true

(%i65)  tabela(sin, 4, 10);
x  F(x)
=========
4  -0.7568
5  -0.9589
6  -0.2794
7  0.657
8  0.9894
```

```
9   0.4121
10   -0.544
=========
(%o65)   true
```

Exemplo 7.24 Definir um comando medias(x, y, z) em formato de bloco que calcule as médias aritmética, geométrica e harmônica dos parâmetros x, y e z.

```
(%i66)   medias(x, y, z) := block(
             [m, g, h],
             if x <= 0 or y <= 0 or z <= 0 then
                 return("ERRO: x, y, z devem ser positivos"),
             m: (x + y + z)/3,
             g: (x*y*z)^(1/3),
             h: 1/((1/x + 1/y + 1/z)/3),
             print("A média aritmética de x, y, z é ", m),
             print("A média geométrica de x, y, z é ", g),
             print("A média harmônica de x, y, z é ", h))$
```

Depois de definido, o comando medias(x, y, z) pode ser utilizado.

```
(%i67)   medias(1, 4, -16);
(%o67)   ERRO: x, y, z devem ser positivos

(%i68)   medias(1, 4, 16);
A média aritmética de x, y, z é 7
A média geométrica de x, y, z é 4
A média harmônica de x, y, z é 16/7

(%o68)   16/7

(%i69)   m;
(%o69)   m

(%i70)   g;
(%o71)   g

(%i72)   h;
(%o72)   h
```

Os valores m, g, h que são definidos como variáveis locais permanecem in-

definidos fora do bloco.

7.7 Exemplos diversos

Exemplo 7.25 Entrar com 3 listas de comprimento 3 e, interpretando essas listas como vetores do $\mathbb{R}^3$, verificar se esses vetores são linearmente dependentes ou independentes. Para isso, definir uma matriz M cujas linhas são as listas fornecidas, calcular o determinante de M e verificar se esse determinante é nulo ou não.

```
(%i73)  Base() := block(
            [v1, v2, v3, det, M],
            /* Entrada dos dados no formato de lista [a, b, c] */
            v1: read("vetor 1 (lista) = "),
            v2: read("vetor 2 (lista) = "),
            v3: read("vetor 3 (lista) = "),
            M: matrix(v1, v2, v3),
            det: determinant(M),
            /* Saída dos resultados */
            print("Matriz M = ", M, "det(M) = ", det),
            if det = 0 then
                print("Os vetores são LD")
            else
                print("Os vetores são LI"))$
```

Testando o comando Base() assim definido:

```
(%i74)  Base();
vetor 1 (lista) =  [1, 2, 3];
vetor 1 (lista) =  [5, 4, 2];
vetor 1 (lista) =  [1, 1, 1];
```

$$\text{Matriz M} = \begin{bmatrix} 1 & 2 & 3 \\ 5 & 4 & 2 \\ 1 & 1 & 1 \end{bmatrix} \qquad \det(M) = -1$$

```
Os vetores são LI
(%o74)  Os vetores são LI
```

Exemplo 7.26 Definir um comando `eq2grau(a, b, c)` que forneça as raízes

de uma equação do segundo grau $ax^2 + bx + c = 0$ e outro comando `ResolveEquacao()` que solicite do usuário os coeficientes da equação e repasse a tarefa de calcular as raízes para o comando `eq2grau(...)`.

```
(%i75)  eq2grau(a, b, c) := block(
            /* variáveis locais: */ [Delta, x1, x2],
            /* Resolução de uma equação do 2o. grau */
            if a = 0 then return("ERRO: a deve ser diferente de zero."),
            Delta : b^2 - 4*a*c,
            x1: (-b + sqrt(Delta))/(2*a),
            x2: (-b - sqrt(Delta))/(2*a),
            if Delta = 0 then
                print("Raízes iguais x1 = x2 = ", x1)
            elseif Delta > 0 then
                print("Raízes reais: x1 = ", x1, ", x2 = ", x2)
            else
                print("Raízes complexas: x1 = ", x1, ", x2 = ", x2),
            return("fim"))$

(%i76)  ResolveEquacao() := block(
            print("Resolvendo uma equação do segundo grau"),
            a: read("Forneça o valor de a: "),
            b: read("Forneça o valor de b: "),
            c: read("Forneça o valor de c: "),
            eq2grau(a, b, c))$
```

Testando o comando eq2grau(a, b, c):

```
(%i77)  eq2grau(1, 5, 6);
Raízes reais: x1 = −2, x2 = −3
(%o77)  fim

(%i78)  eq2grau(0, 5, 6);
(%o78)  ERRO: a deve ser diferente de zero.
```

Testando agora o ResolveEquacao():

```
(%i79)  ResolveEquacao();
Resolvendo uma equação do segundo grau
Forneça o valor de a:  1;
```

```
Forneça o valor de b:  5;
Forneça o valor de c:  -2;
Raízes reais: x1 = (√33−5)/2,  x2 = (−√33−5)/2
(%o79)  fim
```

Exemplo 7.27 O método de Newton para determinação de uma raiz da equação $f(x) = 0$ consiste em se determinar o limite da sequência definida por $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$, $n = 0, 1, 2, 3, \ldots$, sendo x_0 uma aproximação inicial da raiz. Dados uma função em forma de expressão algébrica na variável x, uma aproximação inicial da raiz x_0 e $\varepsilon > 0$, calcular os termos da sequência até que $\Delta = |x_{n+1} - x_n| \leq \varepsilon$, ou seja, enquanto $\Delta = |x_{n+1} - x_n| > \varepsilon$. O último termo da sequência calculado é considerado uma aproximação para o limite da sequência e também uma raiz aproximada da equação.

```
(%i73)  newton(exp, x0, epsilon) := block(
            [x, x1, delta: 1, f, df],
            define(f(x), expr),
            define(df(x), diff(expr, x)),
            while (delta > epsilon) do (
                x1: x0 - f(x0)/df(x0),
                delta: abs(x1 - x0),
                x0: float(x1)),
            return(float(x1)))$
```

Podemos testar o bloco de comandos assim definido para determinar uma raiz de cada uma das equações $2^x - x^2 = 0$ e $\cos x - x^2 + 4 = 0$, usando $x_0 = -1$, ε = 1e-10 $= 10^{-10}$ na primeira, e $x_0 = 2$, ε = 1e-8 $= 10^{-8}$ na segunda.

```
(%i74)  newton(2^x - x^2, -1, 1e-10);
(%o74)  −0.76666469596212
```

```
(%i75)  newton(cos(x) - x^2 + 4, 2, 1e-8);
(%o75)  1.914020619025985
```

Concluímos assim que uma raiz aproximada da equação $2^x - x^2 = 0$ é -0.76666469596212 e da equação $\cos x - x^2 + 4 = 0$ é 1.914020619025985.

Exemplo 7.28 Dada uma expressão algébrica do segundo grau $ax^2 + bx$, "com-

pletar o quadrado" dessa expressão significa determinar qual valor deve ser adicionado para transformá-la em um quadrado de uma soma ou de uma diferença. Se $a > 0$ e o valor adicionado for igual a $s = \frac{b^2}{4a}$, então temos $ax^2 + bx = ax^2 + bx + s - s = (\sqrt{a}x + \sqrt{(}s))^2 - s$, se $b > 0$, e $ax^2 + bx = (\sqrt{a}x - \sqrt{s})^2 - s$, se $b < 0$. Se $a < 0$, então usamos que $ax^2 + bx = -(-ax^2 - bx)$ e completamos o quadrado de $-ax^2 - bx$. Vamos fornecer também um terceiro parâmetro m que se for diferente de 0, então mostramos alguns detalhes dos cálculos e se $m = 0$ então mostramos só o resultado final.

```
(%i76)  completaquadrado(a, b, m) := block(
            [s, expr, x],
            if a = 0 then return("ERRO: a não deve ser nulo"),
            if a < 0 then return( (-1)*completaquadrado(-a, -b, m)),
            if m # 0 then print("Completando o quadrado de ax^2 + bx"),
            s: b^2/(4*a),
            if m # 0 then print("Basta somar e subtrair o valor ", s),
            if m # 0 then print("A expressão dada equivale a "),
            if b > then expr: (sqrt(a)*x + sqrt(s))^2 - s
            else expr: (sqrt(a)*x - sqrt(s))^2 - s,
            return(expr));
```

Uma vez definido, vamos exemplificar a execução desse bloco de comandos completando os quadrados de $x^2 + 5x$, de $-x^2 - 5x$ e de $3x^2 - 11x$.

```
(%i77)  completaquadrado(1, 5, 0);
```
$$(\%o77)\quad \left(x + \frac{5}{2}\right)^2 - \frac{25}{4}$$
```
(%i78)  completaquadrado(-1, -5, 0);
```
$$(\%o78)\quad \frac{25}{4} - \left(x + \frac{5}{2}\right)^2$$
```
(%i79)  completaquadrado(3, -11, 1);
Completando o quadrado de ax^2 + bx
Basta somar e subtrair o valor 121/12
A expressão dada equivale a
```
$$(\%o79)\quad \left(\sqrt{3}x - \frac{11}{2\sqrt{3}}\right)^2 - \frac{121}{12}$$

Exemplo 7.29 Dados cinco pontos (x_i, y_i) do plano entre os quais não existem três colineares, a equação da cônica que passa por esses cinco pontos é dada

por

$$\begin{vmatrix} x^2 & y^2 & xy & x & y & 1 \\ x_1^2 & y_1^2 & x_1y_1 & x_1 & y_1 & 1 \\ x_2^2 & y_2^2 & x_2y_2 & x_2 & y_2 & 1 \\ x_3^2 & y_3^2 & x_3y_3 & x_3 & y_3 & 1 \\ x_4^2 & y_4^2 & x_4y_4 & x_4 & y_4 & 1 \\ x_5^2 & y_5^2 & x_5y_5 & x_5 & y_5 & 1 \end{vmatrix} = 0.$$

Interpretando cinco pontos do plano como sendo uma lista de listas no formato

$$L = [[x_1, y_1], [x_2, y_2], [x_3, y_3], [x_4, y_4], [x_5, y_5]],$$

construir um comando Conica5pontos(L, a, b) que determine a equação da cônica que passa por esses pontos e desenhe o seu gráfico com um comando `draw2d(..., implicit(equação, x, -a, a, y, -b, b))` do pacote `draw`.

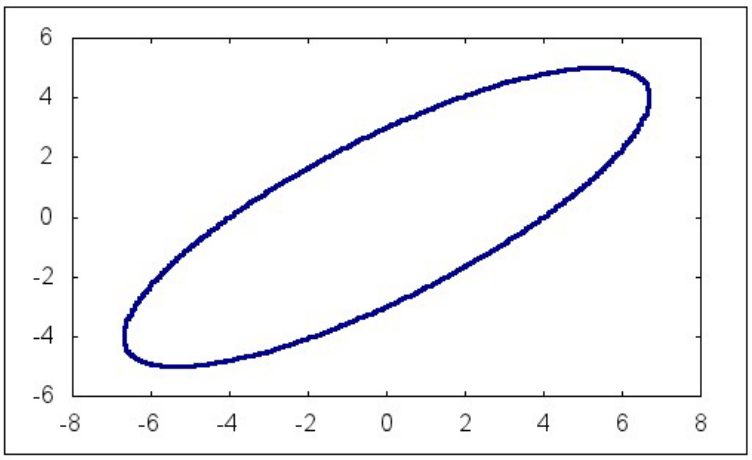
```
(%i80)  Conica5pontos(P, a, b) := block(
            [det, eq, graf, x, y],
            det: matrix([x^2, y^2, x*y, x, y, 1],
            [P[1][1]^2, P[1][2]^2, P[1][1]*P[1][2], P[1][1], P[1][2], 1],
            [P[2][1]^2, P[2][2]^2, P[2][1]*P[2][2], P[2][1], P[2][2], 1],
            [P[3][1]^2, P[3][2]^2, P[3][1]*P[3][2], P[3][1], P[3][2], 1],
            [P[4][1]^2, P[4][2]^2, P[4][1]*P[4][2], P[4][1], P[4][2], 1],
            [P[5][1]^2, P[5][2]^2, P[5][1]*P[5][2], P[5][1], P[5][2], 1],
            eq: expand(determinant(det) = 0),
            print("Equação da cônica: ", eq),
            graf: implicit(eq, x, -a, a, y, -b, b),
            wxdraw2d(color = dark-blue, line_width = 3, graf) )$
```

Depois de definido, podemos agora exemplificar o uso desse comando construindo uma cônica que passa em uma lista de pontos E, e, depois, uma cônica que passa pelos pontos de H.

```
(%i81)  load(draw)$
(%i82)  E: [[-4,0], [4,0], [0,3], [0,-3], [5, 5]]$
(%i83)  Conica5pontos(E, 8, 6);
```

Equação da cônica: $19200y^2 - 23088xy + 10800x^2 - 172800 = 0$

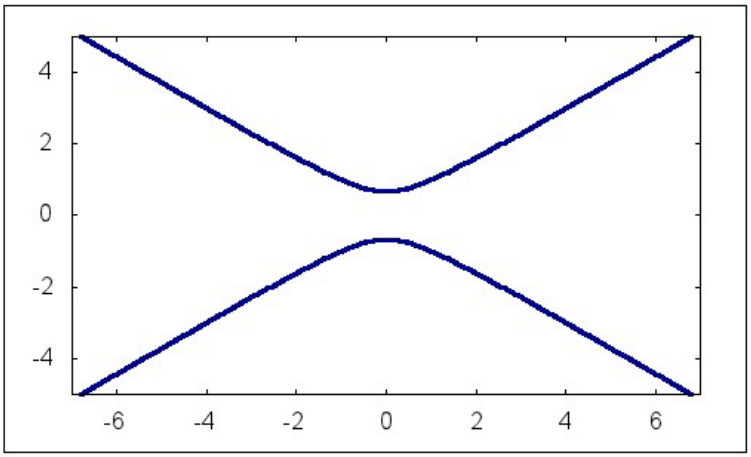
(%t83)




```
(%i84)  H: [[-4, 3], [-4, -3], [4, 3], [4, -3], [-1, -1]]$
(%i85)  Conica5pontos(H, 7, 5);
```

Equação da cônica: $34560y^2 - 18432x^2 - 16128 = 0$

(%t85)



Exemplo 7.30 Toda fração positiva $r = \frac{a}{b}$ e menor do que 1 pode ser escrita como uma soma de frações positivas distintas, com numeradores iguais a 1, denominadas *frações unitárias*. Um algoritmo que sempre funciona nesse caso é o seguinte:

(1) Calculamos q, o inverso multiplicativo do menor inteiro maior do que $\frac{1}{r}$, ou seja, $q = 1/\text{ceiling}(\frac{1}{r})$. Essa fração q é uma das parcelas da soma procurada e guardamos o valor de q como coordenada de uma lista através do comando `cons(q, lista)`.

(2) Se $r - q = 0$, encerramos, senão redefinimos r como sendo $r - q$ e retornamos ao item (1)

No final, obtemos uma lista do tipo $[\frac{1}{s_1}, \frac{1}{s_2}, \ldots, \frac{1}{s_n}]$. Aplicamos a essa lista o operador "+"para obter o somatório $r = \frac{1}{s_1} + \frac{1}{s_2} + \cdots + \frac{1}{s_n}$. É preciso que a variável global `simp` esteja ajustada como `false` para evitar que o somatório

seja efetuado e simplificado.

```
(%i86)  unitaria(r) := block( [lista: [ ], q],
                        if ratnump(r) and r > 0 and r < 1 then
                            while r # 0 do (
                                q: 1/ceiling(1/r),
                                lista: cons(q, lista),
                                r: r - q),
                        return(reverse(lista)))$

(%i87)  decomp(p) := block( [lista: unitaria(p)],
                        simp: false,
                        print(p, "= ", apply("+", lista)),
                        simp: true)$
```

A função `ratnump(r)` resulta `true` se r for um número racional e `false` em caso contrário. Ao se executar o bloco com um parâmetro que não seja racional, resulta uma lista vazia.

```
(%i88)  unitaria(6/7);
```

$$(\%o88)\quad [\frac{1}{2}, \frac{1}{3}, \frac{1}{42}]$$

```
(%i89)  decomp(6/7)$
```

$$\frac{6}{7} = \frac{1}{2} + \frac{1}{3} + \frac{1}{42}$$

```
(%i90)  unitaria(50/101);
```

$$(\%o90)\quad [\frac{1}{3}, \frac{1}{7}, \frac{1}{54}, \frac{1}{2937}, \frac{1}{37376262}]$$

```
(%i91)  decomp(50/101)$
```

$$\frac{50}{101} = \frac{1}{3} + \frac{1}{7} + \frac{1}{54} + \frac{1}{2937} + \frac{1}{37376262}$$

```
(%i92)  decomp(118/247)$
```

$$\frac{118}{247} = \frac{1}{3}+\frac{1}{7}+\frac{1}{649}+\frac{1}{673273}+\frac{1}{1133240658050}+\frac{1}{2568468778114060818946950}$$

Apêndice A

Inequações e números complexos

A.1 Operadores relacionais e lógicos

O *Maxima* possui vários operadores relacionais e lógicos que podem ser usados nas mais diversas situações. Quando é avaliada uma expressão que envolve esses operadores, resulta em um valor true (verdadeiro) ou false (falso).

- **=** Igualdade. Exemplos: $3 = 3$ resulta true, enquanto que $1 = 2$ resulta false.
- **#** Desigualdade. Exemplos: $3 \# 3$ resulta false, enquanto que $1 \# 2$ resulta true.
- $<$ Menor do que ... Exemplos: $3 < 4$ resulta true, enquanto que $5 < 5$ resulta false.
- $<=$ Menor do que ou igual a ... : Exemplos: $5 <= 5$ resulta true, enquanto que $4 <= 3$ resulta false.
- $>$ Maior do que ... Exemplos: $9 > 7$ resulta true, enquanto que $7 > 7$ resulta false.
- $>=$ Maior do que ou igual a ... Exemplos: $7 >= 7$ resulta true, enquanto que $7 > 9$ resulta false.
- **and** Conectivo E. Uma expressão composta do tipo expressão1 and expressão2 é considerada true somente quando expressão1 e expressão2 forem ambas true. Exemplos: $(4 \# 5$ and $1 < 3)$ resulta true, enquanto que $(2 = 2$ and $3 <= 0))$ resulta false.
- **or** Conectivo OU. Uma expressão composta do tipo expressão1 ou expressão2 é considerada false somente quando expressão1 e expressão2 forem ambas false. Exemplos: $(3 = 4$ or $5 = 5)$ resulta true, enquanto que $(4 > 9$ or $2 \# 2)$ resulta false.

- **not** Negação. A negação de true é false e a negação de false é true. Exemplos: not $(3 = 5)$ resulta true, enquanto not $(1 = 1)$ resulta false.

Assim como ocorre com a grande maioria dos nomes de comandos, variáveis ou constantes, os operadores lógicos and, or e not devem ser digitados em letras minúsculas.

Um comando **is(expressão)** pode ser usado para avaliar determinada expressão, resultando sempre em um valor true, false ou unknown (desconhecido).

Um comando **assume(hipótese)** pode ser usado para acrescentar determinada hipótese a respeito do valor de uma expressão, um comando **forget(hipótese)** pode ser usado para desfazer (esquecer) determinada hipótese e um **facts()** para mostrar as hipóteses acrescentadas.

Exemplo A.1 Avaliar se as seguintes expressões são verdadeiras ou falsas:

- $\frac{1}{6} < \frac{1}{2} - \frac{1}{3}$
- $10! \geq 10^6$
- $3 \neq 5$ or $4 < 1$
- $3 \neq 5$ and $4 < 1$
- not $(1 > 3)$ or not $(10 > 9)$
- not $(1 > 3)$ and not $(10 > 9)$

```
(%i1)  is ( 1/6 < 1/2 - 1/3 );
(%o1)  false

(%i2)  is ( 10! >= 10^6 );
(%o2)  true

(%i3)  is (3 # 5 or 4 < 1);
(%o3)  true

(%i4)  is (3 # 5 and 4 < 1);
(%o4)  false

(%i5)  is (not (1 > 3 ) or not (10 > 9));
(%o5)  true

(%i6)  is (not (1 > 3 ) and not (10 > 9));
(%o6)  false
```

Exemplo A.2 Supondo $x > 5$, avaliar se as sentenças $x > 1$, $x > 4$, $x > 9$, $x < 0$ e $x < 10$ são verdadeiras ou falsas.

```
(%i7)  assume(x > 5);
(%o7)  [x > 5]

(%i8)  is (x > 1);
(%o8)  true

(%i9)  is (x > 4);
(%o9)  true

(%i10)  is (x > 9);
(%o10)  unknown

(%i11)  is (x < 0);
(%o11)  false

(%i12)  is (x < 10);
(%o12)  unknown

(%i13)  facts();
(%o13)  [x > 5]

(%i14)  forget(x > 5);
(%o14)  [x > 5]

(%i15)  facts();
(%o15)  []
```

Exemplo A.3 Supondo $1 < y < 5$, avaliar se são verdadeiras ou falsas as seguintes sentenças: $0 < y < 6$, $2 < y < 4$ e $[y < 0$ ou $y > 10]$. Para o *Maxima*, uma dupla desigualdade do tipo $a < y < b$ é equivalente a duas desigualdades $[a < y$ e $y < b]$.

```
(%i16)  assume(1 < y and y < 5);
(%o16)  [y > 1, y < 5]

(%i17)  is (y > 0 and y < 6);
(%o17)  true

(%i18)  is (y < 0 or y > 10);
(%o18)  false

(%i19)  is ( y > 2 and y < 4);
```

```
(%o19)  unknown
```

A.2 Inequações

O *Maxima* possui dois comandos para resolver inequações:

- **solve_rat_ineq(inequação)** do pacote solve_rat_ineq que resolve apenas inequações polinomiais ou racionais (quocientes de polinômios) com uma única variável;
- **fourier_elim([inequação, ...], [variável, ...])** do pacote fourier_elim que admite inequações polinomiais, racionais, modulares ou sistemas de inequações, com uma ou várias variáveis.

O símbolo $\leq$ deve ser digitado na linha de comando do *Maxima* na forma <=, e $\geq$ na forma >=.

Exemplo A.4 Resolver as inequações $x^2 - 5x + 6 > 0$, $x^2 - 5x + 6 \geq 0$, $x^2 - 5x + 6 < 0$ e $x^2 - 5x + 6 \leq 0$. Convém criar um apelido para o nome do comando
`solve_rat_ineq` porque ele é um pouco longo.

```
(%i20)  load(solve_rat_ineq)$
(%i21)  alias(ineq, solve_rat_ineq);
(%o21)  [ineq]

(%i22)  ineq(x^2 - 5*x + 6 > 0);
(%o22)  [[x < 2], [x > 3]]

(%i23)  ineq(x^2 - 5*x + 6 >= 0);
(%o23)  [[x <= 2], [x >= 3]]

(%i24)  ineq(x^2 - 5*x + 6 < 0);
(%o24)  [[x > 2, x < 3]]

(%i25)  ineq(x^2 - 5*x + 6 <= 0);
(%o25)  [[x >= 2, x <= 3]]
```

As soluções apresentadas pelo solve_rat_eq são em forma de lista de listas. No caso deste exemplo, a soluções de cada inequação deve ser interpretada da seguinte forma:

- [[$x < 2$], [$x > 3$]] significa $x < 2$ ou $x > 3$
- [[$x <= 2$], [$x >= 3$]] significa $x \leq 2$ ou $x \geq 3$
- [[$x > 2, x < 3$]] significa $x > 2$ e $x < 3$ que equivale a $2 < x < 3$
- [[$x >= 2, x <= 3$]] significa $x \geq 2$ e $x \leq 3$ que equivale a $2 \leq x \leq 3$

Exemplo A.5 Resolver a inequação $\frac{6x^2+12x+17}{-2x^2+7x-5} \geq -1$.

```
(%i26)  load(solve_rat_ineq)$
(%i27)  inequacao: (6*x^2 + 12*x + 17)/(-2*x^2 + 7*x - 5) >= -1;
```

(%i27) $\frac{6x^2+12x+17}{-2x^2+7x-5} \geq -1$

```
(%i28)  solve_rat_ineq(inequacao);
```

(%i28) $[[x >= -4, x <= -\frac{3}{4}], [x > 1, x < \frac{5}{2}]]$

A solução mostrada significa $x \geq -4$ e $x \leq -\frac{3}{4}$ ou $x > 1$ e $x < \frac{5}{2}$ que é o mesmo que $-4 \leq x \leq -\frac{3}{4}$ ou $1 < x < \frac{5}{2}$.

Exemplo A.6 Determinar a solução de $x^4+2x^3-13x^2-14x+24 > 0$ e interpretar graficamente, destacando no gráfico e no eixo dos x os pontos que são soluções.

```
(%i29)  load(solve_rat_ineq)$
(%i30)  f(x) := x^4 + 2*x^3 -13*x^2 - 14*x + 24;
```

(%o30) $f(x) := x^4 + 2x^3 + (-13)x^2 + (-14)x + 24$

```
(%i31)  solve_rat_ineq(f(x) > 0);
```

(%o31) $[[x < -4], [x > -2, x < 1], [x > 3]]$

```
(%i32)  r(x) := if f(x) > 0 then f(x);
```

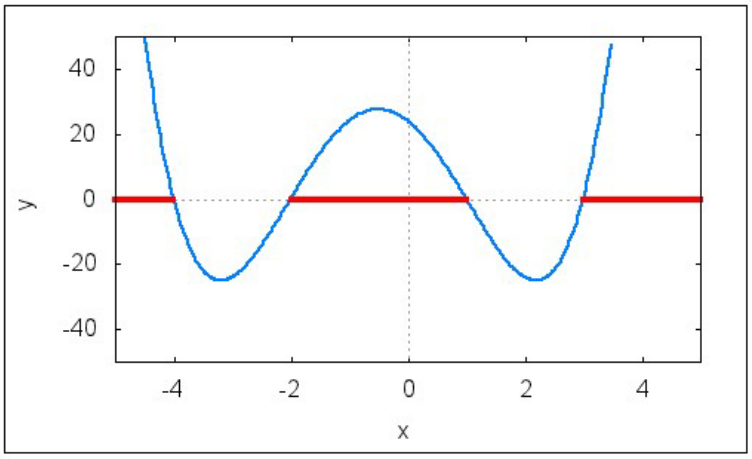
(%o32) r(x):=if f(x) > 0 then f(x)

```
(%i33)  s(x) := if f(x) > 0 then 0;
```

(%o33) s(x):=if f(x) > 0 then 0

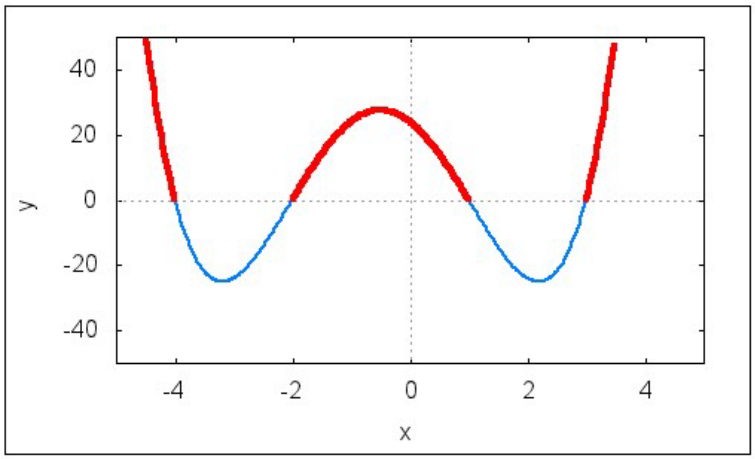
```
(%i34)  wxplot2d([f(x), s(x)], [x, -5, 5], [y, -50, 50], [style, [lines, 2], [lines, 4]], [legend, false]);
```

(%t34)



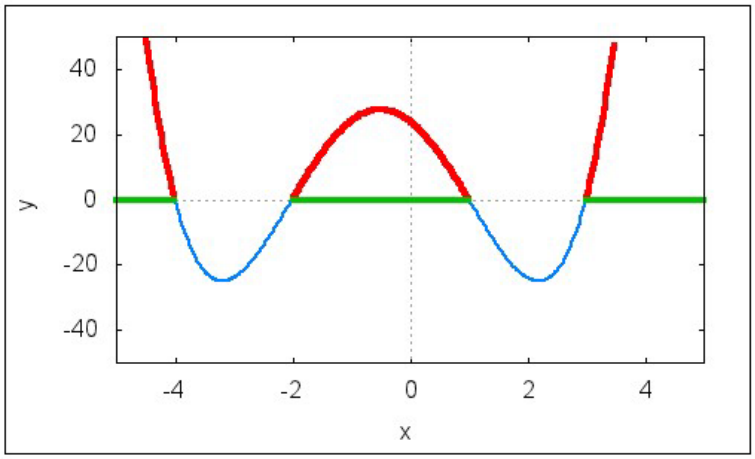

```
(%i35)  wxplot2d([f(x), r(x)], [x, -5, 5], [y, -50, 50], [style, [lines, 2], [lines, 4]], [legend, false]);
```

(%t35)




```
(%i36)  wxplot2d([f(x), r(x), s(x)], [x, -5, 5], [y, -50, 50], [style, [lines, 2], [lines, 4], [lines, 4]], [legend, false]);
```

(%t36)

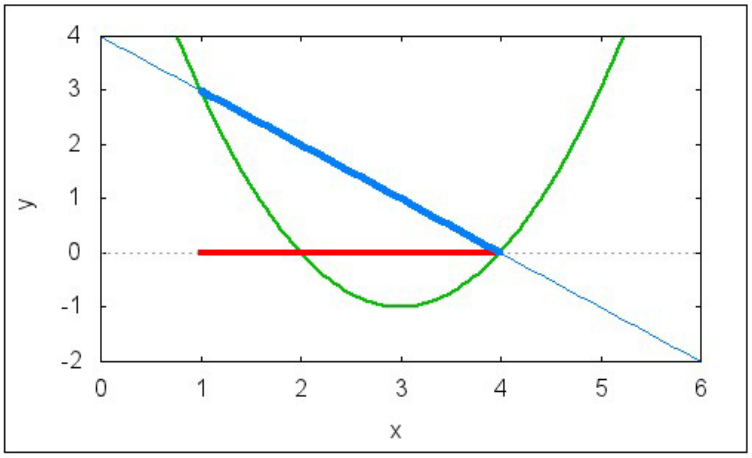


Exemplo A.7 Resolver $x^2 - 6x + 8 < 4 - x$ e interpretar graficamente.

```
(%i37)  f(x) := x^2 - 6*x + 8$
(%i38)  g(x) := 4-x$
```

```
(%i39)  solve_rat_ineq(f(x) < g(x));
(%o39)  [[x > 1, x < 4]]

(%i40)  r(x) := if f(x) <= g(x) then 0$
(%i41)  s(x) := if f(x) < g(x) then g(x)$
(%i42)  wxplot2d([f(x), g(x), r(x), s(x)], [x,0,6], [y,-2,4],
          [style, [lines,2,3], [lines,1,1], [lines,4,2], [lines,4,1]], [legend,false]);
```

(%t42)

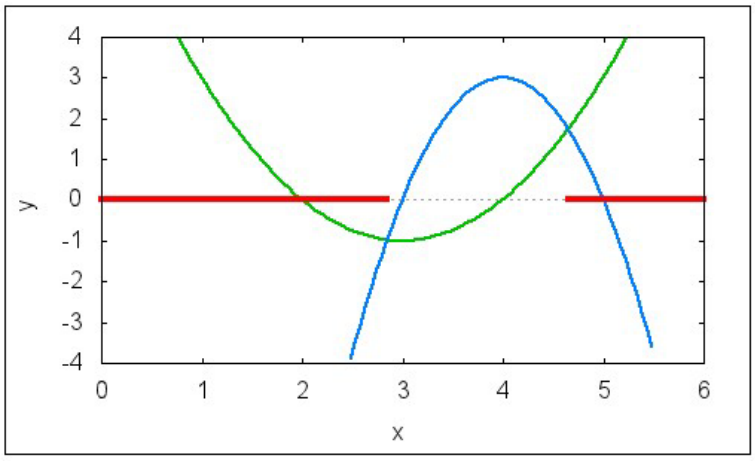


Estão destacados no gráfico os pontos da reta que estão acima da parábola, com suas respectivas abscissas no eixo dos x. Esses pontos correspondem aos que são soluções da inequação dada.

Exemplo A.8 Resolver $x^2-6x+8 \geq -3x^2+24x-45$ e interpretar graficamente.

```
(%i43)  alias(ineq, solve_rat_ineq)$
(%i44)  load(ineq)$
(%i45)  f(x) := x^2 - 6*x + 8$
(%i46)  g(x) := -3*x^2 + 24*x - 45$
(%i47)  ineq(f(x) >= g(x));
(%o47)  [[x <= -(sqrt(13)-15)/4], [x >= (sqrt(13)+15)/4]]

(%i48)  s(x) := if f(x) >= g(x) then 0$
(%i49)  wxplot2d([f(x), g(x), s(x)], [x,0,6], [y,-4,4],
                [style, [lines, 2,3], [lines, 2,1], [lines, 4,2]], [legend,false]);
```

(%t49)



Estão destacados no eixo dos x os pontos que correspondem às soluções.

Exemplo A.9 Resolver as inequações $|4x - 3| > 1$, $|x + |x|| > 7$ e $|x + 1| < -5 - x^2$.

Como essas inequações são modulares, então a maneira de resolvê-las com o *Maxima* é usando o comando fourier_elim do pacote de mesmo nome.

```
(%i50)  load(fourier_elim)$
(%i51)  fourier_elim([ abs(4*x - 3) > 1], [x]);
(%o51)  [x < 1/2] ∨ [1 < x]
(%i52)  fourier_elim([ abs(x + abs(x)) > 7], [x]);
(%o52)  [7/2 < x]
(%i53)  fourier_elim( [abs(x + 1) < -5 - x^2], [x]);
(%o53)  emptyset
```

A.3 Números complexos

O *Maxima* possui várias funções básicas para efetuar cálculos com números complexos $a + bi$, onde a e b são reais e $i = \sqrt{-1}$.

- **realpart(z)** Parte real de z. Exemplo: realpart$(a + bi) = a$.
- **imagpart(z)** Parte imaginária de z. Exemplo: imagpart$(a + bi) = b$.
- **rectform(z)** Forma retangular de z. Resulta em $a + bi$ equivalente a z.
- **polarform(z)** Forma polar de z. Resulta em $re^{i\theta}$ equivalente a z, onde r e θ são reais.

- **conjugate(z)** Conjugado de z. Exemplo: conjugate$(a+bi) = a-bi$.
- **carg(z)** Argumento de z que é um ângulo $\theta \in]-\pi, \pi]$ tal que $re^{i\theta} = z$. Exemplo: carg$(a+bi) =$ arctg $\frac{b}{a}$.
- **cabs(z)** Valor absoluto de z. Exemplo: cabs$(a+bi) = \sqrt{a^2+b^2}$.
- **exponentialize(expressão)** Escreve as funções trigonométricas e hiperbólicas que aparecerem na expressão no formato de exponencial complexa.

Se for atribuído o valor true à variável global **demoivre**, então a exponencial complexa e^{a+bi} é transformada em $e^a(\cos b + i \operatorname{sen} b)$. O valor padrão (default) da variável demoivre é false. Se for atribuído um valor false à variável global **%piargs**, então o cosseno e o seno de alguns ângulos como $\frac{\pi}{6}$, $\frac{\pi}{4}$, $\frac{\pi}{3}$, $\frac{\pi}{2}$, ... não são calculados.

Exemplo A.10 Determinar a forma retangular dos seguintes números: $z_1 = \left(\frac{1+i}{\sqrt{2}}\right)^{2015}$, $z_2 = \frac{-2+11i}{5+3i}$, $z_3 = \left(\frac{1+i}{1-i}\right)^{16} + \left(\frac{1-i}{1+i}\right)^{8}$ e $z_4 = \sqrt{1+\sqrt{i}}$.

```
(%i54)  z1: ((1 + %i)/sqrt(2))^2015;
```
(%o54) $\frac{(\%i+1)^{2015}}{2^{\frac{2015}{2}}}$

```
(%i55)  rectform(z1);
```
(%o56) $\frac{1}{\sqrt{2}} - \frac{\%i}{\sqrt{2}}$

```
(%i57)  z2: (-2 + 11*%i)/(5 + 3*%i);
```
(%o57) $\frac{11\%i-2}{3\%i+5}$

```
(%i58)  rectform(z2);
```
(%o58) $\frac{61\%i}{34} + \frac{23}{34}$

```
(%i59)  z3: ((1+%i)/(1-%i))^16 + ((1-%i)/(1+%i))^8;
```
(%o59) $\frac{(\%i+1)^{16}}{(1-\%i)^{16}} + \frac{(1-\%i)^8}{(\%i+1)^8}$

```
(%i60)  rectform(z3);
```
(%o61) 2

```
(%i62)  z4: sqrt(1 + sqrt(%i));
```
(%o62) $\sqrt{(-1)^{1/4}+1}$

```
(%i63)  rectform(z4);
```
(%o63) $\frac{\sqrt{\sqrt{(\frac{1}{\sqrt{2}}+1)^2+\frac{1}{2}}-\frac{1}{\sqrt{2}}-1}\,\%i}{\sqrt{2}} + \frac{\sqrt{\sqrt{(\frac{1}{\sqrt{2}}+1)^2+\frac{1}{2}}+\frac{1}{\sqrt{2}}+1}}{\sqrt{2}}$

```
(%i64)  radcan(%);
```

(%o64) $\frac{2^{1/4}\sqrt{\sqrt{2}\sqrt{\sqrt{2}+2}-\sqrt{2}-1}\,\%i+2^{1/4}\sqrt{\sqrt{2}\sqrt{\sqrt{2}+2}+\sqrt{2}+1}}{2}$

Exemplo A.11 Determinar as partes real e imaginárias dos números $w_1 = (3-7i)^5$ e $w_2 = i^i$.

```
(%i65) w1: (3 - 7*%i)^5 $ w2: %i^%i $
(%i66) realpart(w1);
(%o66) 23028

(%i67) imagpart(w1);
(%o67) 11228

(%i68) realpart(w2);
(%o68) %e^-%pi/2

(%i69) imagpart(w2);
(%o69) 0
```

Exemplo A.12 Determinar as formas polares de $z_1 = -\frac{1}{2} + i\frac{\sqrt{3}}{2}$ e $z_2 = 4-7i$, usando a variável demoivre igual a false e, depois, igual a true.

```
(%i70) z1: -1/2 + %i*sqrt(3)/2; z2: 4 - 7*%i;
(%o71) sqrt(3)%i/2 - 1/2

(%o72) 4 - 7%i

(%i73) demoivre: false$
(%i74) polarform(z1);
(%o74) %e^(2%i%pi/3)

(%i75) polarform(z2);
(%o75) sqrt(65) %e^(-%i atan(7/4))

(%i76) demoivre: true$
(%i77) polarform(z1);
(%o77) sqrt(3)%i/2 - 1/2

(%i78) polarform(z2);
(%o78) sqrt(65)(4/sqrt(65) - 7%i/sqrt(65))
```

Exemplo A.13 Atribuindo um valor false à variável global %piargs e um valor

true à variável demoivre, determine as formas polares dos números $1+i$, $-1+i$, $-\sqrt{5}$ e $10i$.

```
(%i79)  %piargs: false$ demoivre: true$
(%i80)  polarform( 1 + %i);
(%o801)  √2(%i sin(π/4) + cos(π/4))
(%i81)  polarform(-1 + %i);
(%o81)  √2(%i sin((3π/4) + cos(3π/4))
(%i82)  polarform(-sqrt(5));
(%o82)  √5(%i sin(π) + cos(π))
(%i83)  polarform(10*%i);
(%o83)  10(%i sin(π/2) + cos(π/2))
```

Exemplo A.14 Determinar $z = x + iy$ que seja solução de $z^2 = -16 + 30i$.

```
(%i84)  z: x + %i*y$   a: -16 + %i*30$
(%i85)  eq1: realpart(expand(z^2)) = realpart(a);
        eq2: imagpart(expand(z^2)) = imagpart(a);
(%o85)  x^2 − y^2 = −16
(%o86)  2xy = 30
(%i87)  solve([eq1, eq2], [x, y]);
(%o87)  [[x = −5%i, y = 3%i], [x = 5%i, y = −3%i], [x = −3, y = −5], [x = 3, y = 5]]
```

Pela resposta do programa, percebe-se que a solução do problema é $-3-5i$ ou $3+5i$.

Uma solução mais simples desse problema pode ser obtida da seguinte forma:

```
(%i88)  sqrt(-16 + 30*%i);
(%o88)  √(30%i − 16)
(%i89)  rectform(%);
(%o89)  5%i + 3
```

Exemplo A.15 Se z_1 e z_2 são dois números complexos quaisquer, então mostre que

$$|z_1 - z_2|^2 + |z_1 + z_2|^2 = 2(|z_1|^2 + |z_2|^2).$$

```
(%i90)  z1: a1 + %i*b1;   z2: a2 + %i*b2;
```
(%o90) $\%ib1 + a1$

(%o91) $\%ib2 + a2$

```
(%i92)  expr1: cabs(z1 - z2)^2 + cabs(z1 + z2)^2;
```
(%o92) $(b2 + b1)^2 + (b1 - b2)^2 + (a2 + a1)^2 + (a1 - a2)^2$

```
(%i93)  expr2: 2*(cabs(z1)^2 + cabs(z2)^2);
```
(%o93) $2(b2^2 + b1^2 + a2^2 + a1^2)$

```
(%i94)  expr1: expand(expr1);
```
(%o94) $2b2^2 + 2b1^2 + 2a2^2 + 2a1^2$

```
(%i95)  expr2: expand(expr2);
```
(%o95) $2b2^2 + 2b1^2 + 2a2^2 + 2a1^2$

```
(%i96)  expr1 - expr2;
```
(%o96) 0

Exemplo A.16 Se z_1 e z_2 são números complexos, então mostrar que é válida a seguinte igualdade:

$$|1 - z_1\bar{z}_2|^2 - |z_1 - z_2|^2 = (1 - |z_1|^2)(1 - |z_2|^2).$$

```
(%i97)  z1: a + b*%i$   z2: c + d*%i$
(%i98)  expr1: cabs(1 - z1*conjugate(z2))^2 - cabs(z1 - z2)^2;
```
(%o99) $(-bd - ac + 1)^2 + (ad - bc)^2 - (b - d)^2 - (a - c)^2$

```
(%i100)  expr2: (1 - cabs(z1)^2)*(1 - cabs(z2)^2);
```
(%o100) $(-b^2 - a^2 + 1)(-d^2 - c^2 + 1)$

```
(%i101)  expr1: expand(expr1);
```
(%o101) $b^2d^2 + a^2d^2 - d^2 + b^2c^2 + a^2c^2 - c^2 - b^2 - a^2 + 1$

```
(%i102)  expr2: expand(expr2);
```
(%o102) $b^2d^2 + a^2d^2 - d^2 + b^2c^2 + a^2c^2 - c^2 - b^2 - a^2 + 1$

```
(%i103)  expr1 - expr2;
```
(%o103) 0

Exemplo A.17 Sendo $z = \rho \cdot (\cos\theta + i \cdot \operatorname{sen}\theta)$, mostrar que $\dfrac{z}{z^2 + \rho^2}$ é real e que $\dfrac{\rho - i \cdot z}{\rho + i \cdot z}$ é imaginário puro.

```
(%i104) z: %rho*(cos(%theta) + %i*sin(%theta));
(%o104) ρ(%i sin(θ) + cos(θ))

(%i105) w1: z/(z^2 + %rho^2);
(%o105) ρ(%i sin(θ)+cos(θ)) / (ρ²(%i sin(θ)+cos(θ))²+ρ²)

(%i106) imagpart(w1);
(%o106) ρ(sin(θ)(ρ²(cos(θ)²−sin(θ)²)+ρ²)−2ρ² cos(θ)² sin(θ)) / ((ρ²(cos(θ)²−sin(θ)²)+ρ²)²+4ρ⁴ cos(θ)² sin(θ)²)

(%i107) trigsimp(%);
(%o107) 0

(%i108) w2: (%rho - %i*z)/(%rho + %i*z);
(%o108) (ρ−%iρ(%i sin(θ)+cos(θ))) / (%iρ(%i sin(θ)+cos(θ))+ρ)

(%i109) realpart(w2);
(%o109) ((ρ−ρ sin(θ))(ρ sin(θ)+ρ)−ρ² cos(θ)²) / ((ρ−ρ sin(θ))²+ρ² cos(θ)²)

(%i110) trigsimp(%);
(%o110) 0
```

Fica mostrado dessa forma que a parte imaginária de $\frac{z}{z^2+\rho^2}$ e a parte real de $\frac{\rho - i\cdot z}{\rho + i\cdot z}$ são nulas.

Exemplo A.18 Escrever na forma de exponencial complexa os números $\cos\frac{\pi}{7}$, $\cos 2i$, $\operatorname{sen}\beta$, $\operatorname{senh} 3$, $\operatorname{tgh} 5$ e $\cos\alpha + i\operatorname{sen}\alpha$.

```
(%i111) exponentialize( cos(%pi/7) );
(%o111) (%e^(%iπ/7)+%e^−(%iπ/7)) / 2

(%i112) exponentialize( cos(%i*2) );
(%o112) (%e²+%e^−2) / 2

(%i113) exponentialize( sin(beta) );
(%o113) −%i(%e^(%iβ)−%e^−(%iβ)) / 2

(%i114) exponentialize( sinh(3) );
(%o114) (%e³−%e^−3) / 2

(%i115) exponentialize( tanh(5) );
(%o115) (%e⁵−%e^−5) / (%e⁵+%e^−5)

(%i116) exponentialize( cos(alpha) + %i*sin(alpha) );
(%o116) (%e^(%iα)+%e^−(%iα)) / 2 + (%e^(%iα)−%e^−(%iα)) / 2
```

```
(%i117)  expand(%);
```
(%o117) $\%e^{\%i\alpha}$

Exemplo A.19 Dado um número complexo a, o lugar geométrico dos números complexos z que satisfazem a equação $|z-a|\,|z+a| = |a|^2$ é denominado *lemniscata*. Desenhar o gráfico da lemniscata em que $a = 2$.

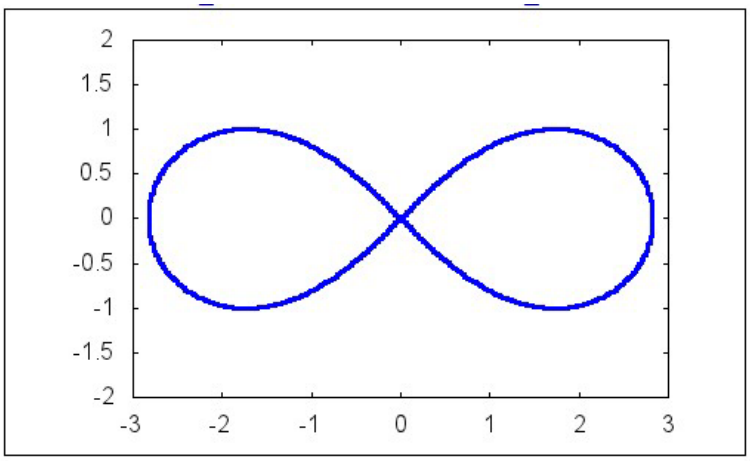
```
(%i118)  z: x + %i*y;
```
(%o118) $\%iy + x$

```
(%i119)  load(draw)$
(%i119)  a: 2$ graf: implicit(cabs(z-a)*cabs(z+a)=cabs(a)^2, x,-3,3, y,-2,2);
```

(%o119) $implicit(\sqrt{y^2+(x-2)^2}\,\sqrt{y^2+(x+2)^2} = 4, x, -3, 3, y, -2, 2)$

```
(%i120)  wxdraw2d(line_width=3, proportional_axes=xy, graf);
```

(%t120)



Apêndice B

Funções e operadores

B.1 Definição de uma função

Uma função pode ser definida com um := ou com um define(...):

- **função(variável) := expressão** Exemplo: f(x) := x^2 + 1;
- **função(variável 1, variável 2, ...) := expressão**
  Exemplo: f(x, y, z) := sqrt(x^2 + y^2 + z^2);
- **define(função(variável), expressão)**
  Exemplo: define(g(x), 3*x^2 - 5*x + 7);
- **define(função(variável 1, variável 2, ...), expressão)**
  Exemplo: define(g(x, y, z), x/2 + y/3 + z/4);

O operador ":=" não deve ser confundido com ":" , nem com "=" . Uma linha de comando como f(x) = x^2 + 1 ou f(x) : x^2 + 1 **não define** uma função f cujo valor em x é x^2+1.

Uma função definida por várias sentenças pode ser definida utilizando-se um comando do tipo:
função(variável) := if condição then expressão 1 else expressão 2;

Exemplo B.1 Definir uma função $f(x) = x^2 - 5x + 6$ e calcular $f(1)$, $f(2)$, $f(a)$ e $\frac{f(a+h)-f(a)}{h}$.

```
(%i1)  f(x) := x^2 - 5*x + 6;
(%o1)  f(x) := x^2 - 5x + 6

(%i2)  f(1);
(%o2)  2

(%i3)  f(2);
(%o3)  0
```

```
(%i4)  f(a);
```
(%o4) $a^2 - 5a + 6$

```
(%i5)  (f(a + h) - f(a))/h;
```
(%o5) $\frac{(h+a)^2-5(h+a)-a^2+5a}{h}$

Exemplo B.2 Escrever os 15 primeiros termos da progressão aritmética cujo termo geral é dado por $a_n = 11n - 27$.

```
(%i6)  a(n) := 11*n - 27;
```
(%o6) $a(n) := 11n - 27$

```
(%i7)  makelist(a(n), n, 1, 15);
```
(%o7) $[-16, -5, 6, 17, 28, 39, 50, 61, 72, 83, 94, 105, 116, 127, 138]$

Exemplo B.3 Definir as seguintes funções $g(x) = \frac{d^4}{dx^4}(\text{tg}\, x)$, $h(x) = \int cos^5 x dx$ e calcular g(1) e h(1).

```
(%i8)  define(g(x), diff(tan(x), x, 4));
```
(%o8) $g(x) := 8\sec(x)^2\tan(x)^3 + 16\sec(x)^4\tan(x)$

```
(%i9)  g(1);
```
(%o9) $8\sec(1)^2\tan(1)^3 + 16\sec(1)^4\tan(1)$

```
(%i10)  define(h(x), integrate(cos(x)^5, x));
```
(%o10) $h(x) := \frac{\sin(x)^5}{5} - \frac{2\sin(x)^3}{3} + \sin(x)$

```
(%i11)  h(1);
```
(%o11) $\frac{\sin(1)^5}{5} - \frac{2\sin(1)^3}{3} + \sin(1)$

As funções deste exemplo não podem ser definidas usando-se o operador :=. Por exemplo, se g(x) fosse definida por g(x) := diff(tan(x), x, 4), então teríamos g(1) = diff(tan(1), 1, 4) que não pode ser calculado.

Exemplo B.4 Definir uma função $F(\rho, \theta, \phi) := \rho^4 \cos(\theta)^2 \sin(\phi)$ e calcular $F(2, \frac{\pi}{4}, \frac{\pi}{3})$.

```
(%i12)  F(rho, theta, phi) := rho^4*cos(theta)^2*sin(phi);
```
(%o12) $F(\rho, \theta, \phi) := \rho^4 \cos(\theta)^2 \sin(\phi)$

```
(%i13)  F(2, %pi/4, %pi/3);
```

(%o13) $4\sqrt{3}$

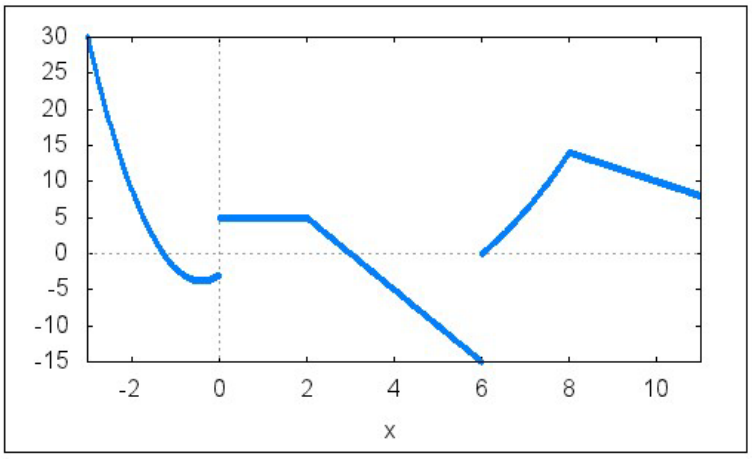
Exemplo B.5 Definir a função

$$f(x) = \begin{cases} 5x^2 + 4x - 3 & \text{, se} \quad x < 0 \\ 5 & \text{, se} \quad 0 \leq x < 2 \\ 15 - 5x & \text{, se} \quad 2 \leq x < 6 \\ x^2 - 7x + 6 & \text{, se} \quad 6 \leq x < 8 \\ -2x + 30 & \text{, se} \quad x \geq 8 \end{cases},$$

construir uma lista de valores $[n, f(n)]$ com n variando de -2 a 10 e o gráfico no intervalo $[-3, 11]$.

```
(%i14) f(x) := if x < 0 then 5*x^2 + 4*x -3
                  else if 0 <= x and x < 2 then 5
                  else if 2 <= x and x < 6 then 15 - 5*x
                  else if 6 <= x and x < 8 then x^2 - 7*x + 6

                  else -2*x +30 $
(%i15) makelist([n, f(n)], n, -2, 10);
(%o15) [[−2, 9], [−1, −2], [0, 5], [1, 5], [2, 5], [3, 0], [4, −5], [5, −10], [6, 0],
                    [7, 6], [8, 14], [9, 12], [10, 10]]
(%i16) wxplot2d(f(x), [x, -3, 11], [style, [points, 1, 1, 1]], [nticks, 200]);
```

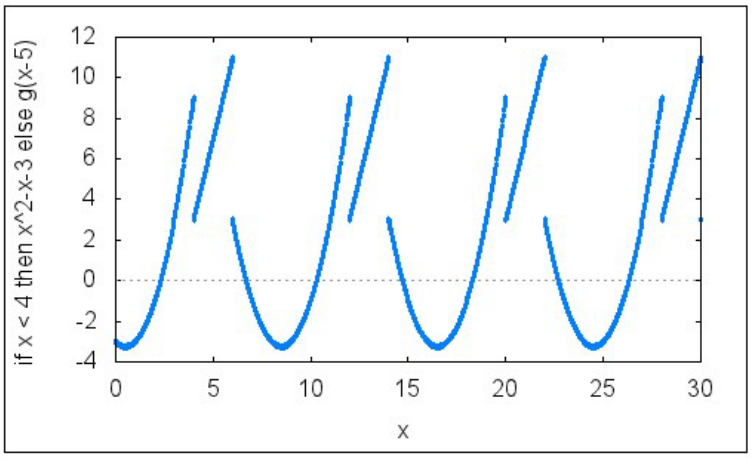
(%t16)



Exemplo B.6 Definir as funções $f(x) = \begin{cases} x^2 - x - 3 & \text{, se} \quad x < 4 \\ g(x-5) & \text{, se} \quad x \geq 4 \end{cases}$ e $g(x) = \begin{cases} 4x + 7 & \text{, se} \quad x < 1 \\ f(x-3) & \text{, se} \quad x \geq 1 \end{cases}$, listar os valores de $f(n)$ e $g(n)$ com n variando nos inteiros de -8 a 8 e construir o gráfico de $f(x)$ com $0 \leq x \leq 30$.

```
(%i17) f(x) := if x < 4 then x^2 - x - 3 else g(x - 5)$
(%i18) g(x) := if x < 1 then 4*x + 7 else f(x - 3)$
(%i19) makelist(f(n), n, -8, 8);
(%o19) [69, 53, 39, 27, 17, 9, 3, -1, -3, -3, -1, 3, 3, 7, 3, -1, -3]

(%i20) makelist(g(n), n, -8, 8);
(%o21) [-25, -21, -17, -13, -9, -5, -1, 3, 7, 3, -1, -3, -3, -1, 3, 3, 7]

(%i22) wxplot2d(f(x), [x, 0, 30], [style, [points, 1, 1, 1]], [nticks, 200]);
```

(%t22)



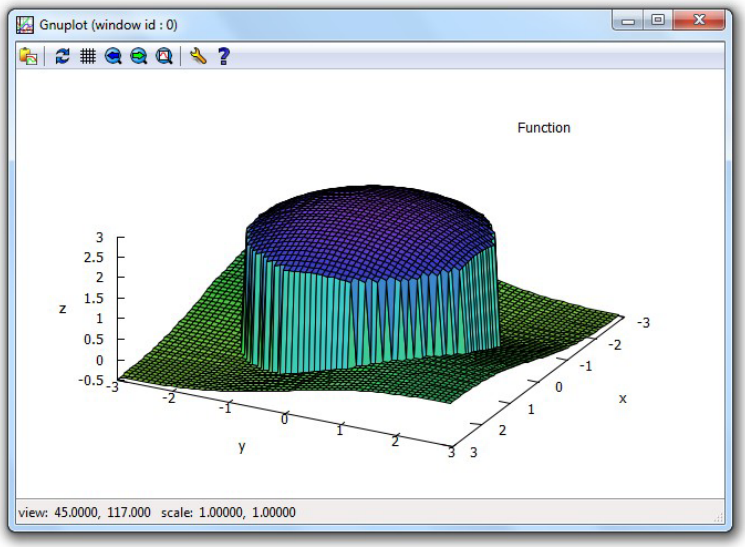
Exemplo B.7 Definir a função

$$g(x,y) = \begin{cases} \sqrt{9-x^2-y^2} & \text{, se} \quad x^2+y^2 \leq 4 \\ \dfrac{xy}{1+x^2+y^2} & \text{, se} \quad x^2+y^2 > 4 \end{cases}$$

e construir seu gráfico no domínio $-3 \leq x \leq 3, \ -3 \leq y \leq 3$.

```
(%i23) f(x, y) := if x^2 + y^2 <= 4 then sqrt(9 - x^2 - y^2)
                  else x*y/(1 + x^2 + y^2);
(%o23) f(x,y) := if x^2+y^2 <= 4 then sqrt(9-x^2-y^2) else xy/(1+x^2+y^2)

(%i24) plot3d(g(x, y), [x, -3, 3], [y, -3, 3], [grid, 60, 60]);
```

(%t24)



Para verificar a lista de funções definidas, deve-se usar um comando **dispfun(all)** ou consultar o valor da variável global **functions**. Para verificar a definição de alguma função, é só digitar **fundef(função)**.

```
(%i25) functions;
(%o26) [f(x), g(x), h(x,y,z)]

(%i27) fundef(h);
(%o28) h(x,y,z) := x+2y+4z+1
```

B.2 Definição de um operador

Um operador é uma função cujo nome pode ser formado por um ou vários caracteres simbólicos. Pode ser do tipo prefixado, infixado ou pósfixado, dependendo dele ser usado antes, no meio ou depois da lista de parâmetros (variáveis). Por exemplo, a negação "=" é prefixado porque é usado antes das variáveis ou valores tais como -x, -3, etc. Já a adição "+"é infixado pois é usado no meio (como a+b, 3+5, etc.) e o fatorial "!" é pósfixado pois é usado depois do valor (como n!, 5!, etc.). Além desses tipos, tem também o tipo delimitado (matchfix) que usa a lista de variáveis entre delimitadores do tipo "abre e fecha".

Para definir um operador, deve-se, antes de tudo, definir o seu tipo com um comando

- **prefix("operador")**,
- **infix("operador")**,
- **postfix("operador")** ou

- **matchfix("delimator esquerdo", "delimitador direito")**,

no caso do operador ser prefixado, infixado, pósfixado ou do tipo delimitado, respectivamente. Deve-se ter o cuidado de se colocar o nome do operador ou dos delimitadores entre aspas.

Depois disso, define-se como o operador atua em um ponto usando-se um comando do tipo

"operador"(lista de variáveis) := valor

ou

define("operador"(lista de variáveis), valor),

semelhante à definição de uma função. No caso do operador ser do tipo delimitado, deve-se usar o delimitador esquerdo como sendo o seu nome na definição do seu valor em um ponto.

Exemplo B.8 Definir o operador "+x"infixado e tal para todo a e todo b satisfaça a seguinte equação: $a +x b = a+b+ab$.

```
(%i29)  infix("+x");
(%o29)  +x

(%i30)  "+x"(a, b) := a + b + a*b;
(%o30)  a +x b := a + b + a * b

(%i31)  3 +x 4;
(%o31)  19

(%i32)  x +x x;
(%o33)  x^2 + 2x

(%i34)  a +x b +x c +x d;
(%o34)  ((ab + b + a)c + c + ab + b + a)d + d + (ab + b + a)c + c + ab + b + a
```

Exemplo B.9 Definir os seguintes operadores "<<"e ">>"infixados:

$$a << b \Leftrightarrow b - a > 100$$

e

$$a >> b \Leftrightarrow b << a.$$

```
(%i35)  infix("<<");
(%o35)  <<

(%i36)  "<<"(a, b) := is(b - a > 100);
```

```
(%o36)  a << b := is(b - a > 100)

(%i37)  infix("<<");
(%o37)  <<

(%i38)  ">>"(a, b) := b << a;
(%o38)  a >> b := b << a

(%i39)  3 << 4
(%o39)  false

(%i40)  3 << 150
(%o41)  true

(%i42)  200 >> 199
(%o42)  false

(%i43)  200 >> 4
(%o43)  true
```

Exemplo B.10 Definir o seguinte operador pósfixado "?": $n? = n(n-1)$.

```
(%i44)  postfix("?");
(%o44)  ?

(%i45)  "?"(n) := n*(n-1);
(%o45)  ?(n) := n(n - 1)

(%i46)  10?;
(%o46)  90

(%i47)  10??;
(%o48)  8010

(%i49)  x???;
(%o49)  (x - 1)x((x - 1)x - 1)((x - 1)x((x - 1)x - 1) - 1)
```

Exemplo B.11 Definir um operador delimitado por "|"e "|"e que satisfaça o seguinte: $|x|$ = valor absoluto de x.

```
(%i50)  matchfix("|", "|");
(%o51)  |

(%i52)  "|"(x) := abs(x);
```

```
(%o52)  (|x|) := abs(x)

(%i53)  | - 11|;
(%o53)  11

(%i54)  |1/4 - sqrt(3)|;
(%o54)  sqrt(3) - 1/4
```

;

Uma definição um pouco mais sofisticada do operador valor absoluto poderia ser: "|"(x) := if imagpart(x) = 0 then abs(x) else cabs(x); Com essa definição, é possivel calcular valor absoluto de reais e também de complexos.

Exemplo B.12 Sendo x e y duas listas de números reais de comprimento 3, definir o seguinte operador delimitado por "[{"e"}]":

$$[\{x, y\}] = \sqrt{(x[1] - y[1])^2 + (x[2] - y[2])^2 + (x[3] - y[3])^2}$$

Neste caso, x e y podem ser interpretados como coordenadas de pontos no espaço e [{ x, y }] como a distância entre eles.

```
(%i55)  matchfix("[{", "}]");
(%o55)  [{

(%i56)  "[{"(x, y) := sqrt((x[1]-y[1])^2 + (x[2]-y[2])^2 + (x[3]-y[3])^2);
(%o56)  [{x, y}] := sqrt((x[1] - y[1])^2 + (x[2] - y[2])^2 + (x[3] - y[3])^2)

(%i57)  [{ [3, 4, 5], [1, 0, -3] }];
(%o57)  2*sqrt(21)

(%i58)  [{ [a, b, c], [0, 0, -2] }];
(%o58)  sqrt((c + 2)^2 + b^2 + a^2)
```

Para listar os operadores definidos, basta usar um **dispfun(all)**:

```
(%i59)  dispfun(all);
(%t60)  a  +x  b := a + b + a*b

(%t61)  a <<  b := is(b - a > 100)

(%t62)  a >>  b := b << a

(%t63)  ?(n) := n*(n - 1)

(%t64)  (|x|) := abs(x)

(%t65)  [{x, y}] := sqrt((x[1] - y[1])^2 + (x[2] - y[2])^2 + (x[3] - y[3])^2)
```

(%o66) [%t60, %t61, %t62, %t63, %t64, %t65]

Para apagar a definição de um operador, pode-se usar um comando **remfunction("operador")** ou **kill("operador")**.

B.3 Funções anônimas

Um comando de um dos tipos mostrados a seguir:

- **lambda([variável], expressão)**
- **lambda([variável 1, variável 2, ...], expressão)**

pode ser usado para criar uma função anônima.

Por exemplo, lambda([x], x^2) define uma funçao $x \longmapsto x^2$, ou seja, uma função que associa a cada x o seu quadrado x^2. Outro exemplo: lambda([x, y, z], x^2 + y^2 + z^2) é a função $(x, y, z) \longmapsto x^2 + y^2 + z^2$.

Uma função anônima pode ser usada como valor atribuído a uma variável e pode aparecer em qualquer lugar que uma função comum poderia aparecer. Sendo assim, uma atribuição do tipo

nome: lambda([variável], expressão)

é considerado equivalente a uma definição de função do tipo

nome(variável) := expressão.

Exemplo B.13 Aplicar a função anônima $x \longmapsto \sqrt{x+1}$ nos pontos 1, 2 e k.

```
(%i67) lambda([x], sqrt(x + 1))(1);
(%o67) sqrt(2)

(%i68) lambda([x], sqrt(x + 1))(2);
(%o68) sqrt(3)

(%i69) lambda([x], sqrt(x + 1))(k);
(%o69) sqrt(k + 1)
```

Exemplo B.14 Definir a função anônima $(x, y) \longmapsto \dfrac{(x+y-2)^2 + x + 3y - 2}{2}$, atribuí-la a uma variável H e calcular $H(2, 3)$, $H(4, 7)$ e $H(a, b)$.

```
(%i70) H: lambda([x, y], ((x + y - 2)^2 + x + 3*y -2)/2);
```
(%o71) $lambda\left([x, y], \frac{(x+y-2)^2+x+3y-2}{2}\right)$

```
(%i72) H(2, 3);
```
(%o72) 9

```
(%i73) H(4, 7);
```
(%o73) 52

```
(%i74) H(a, b);
```
(%o74) $\frac{(b+a-2)^2+3b+a-2}{2}$

Exemplo B.15 Sendo L a lista $[1, 2, 3, 4, 5, 6]$, mapear as funções anônimas $x \longmapsto 1/x$, $x \longmapsto x^x$ e $x \longmapsto 10 - 3x$ nessa lista.

```
(%i75) L: [1, 2, 3, 4, 5, 6]$
(%i76) map( lambda([x], 1/x), L);
```
(%o76) $[1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}]$

```
(%i77) map( lambda([x], x^x), L);
```
(%o78) $[1, 4, 27, 256, 3125, 46656]$

```
(%i79) map( lambda([x], 10-3*x), L);
```
(%o79) $[7, 4, 1, -2, -5, -8]$

B.4 Função com número variável de argumentos

Se o último ou único argumento de uma função definida com um define for uma lista com um único elemento, então a função admite um número variável de argumentos.

Exemplo B.16 Definir uma função S que resulte no somatório dos seus argumentos.

```
(%i80) define(S([X]), '(apply("+", X)));
```
(%o80) $S([X]) := apply(+, X)$

```
(%i81) S(1, 2, 3);
```
(%o81) 6

```
(%i82) S(a, b, c, d);
```

```
(%o82)  d + c + b + a

(%i83)  ordergreat(a, b, c, d, e, f, g)$
(%i84)  S(a, b, c, d, e, f);
(%o84)  a + b + c + d + e + f

(%i85)  S(a, a, b, a, g, f, f, c, d, b, b);
(%o85)  3a + 3b + c + d + 2f + g
```

É importante usar o apóstrofo antes do apply para retardar a execução desse comando. Se não tivesse o apóstrofo, o comando seria executado antes do momento certo.

Exemplo B.17 Definir uma função R que admita pelo menos três argumentos e que seja definida pela raiz quadrada da soma dos quadrados dos três primeiros argumentos, mais a soma das quartas potências dos demais argumentos (se existirem), ou seja, $R(a, b, c, x_1, \ldots, x_n) = \sqrt{a^2 + b^2 + c^2} + x_1^4 + \ \ldots \ + x_n^4$.

```
(%i86)  define(R(a, b, c, [x]), sqrt(a^2+b^2+c^2) + '(apply("+", x^4)));
(%o87)  R(a, b, c, [x]) := sqrt(c^2 + b^2 + a^2) + apply(+, x^4)

(%i88)  R(r, s, p);
(%o88)  sqrt(s^2 + r^2 + p^2)

(%i89)  R(r, s, p, u);
(%o89)  u^4 + sqrt(s^2 + r^2 + p^2)

(%i90)  R(r, s, p, u, v, w, x, y, z);
(%o90)  z^4 + y^4 + x^4 + w^4 + v^4 + u^4 + sqrt(s^2 + r^2 + p^2)
```

Exemplo B.18 Definir uma função M que resulte na média aritmética dos seus argumentos: $M(a_1, a_2, \ldots, a_n) = \dfrac{a_1 + a_2 + \cdots + a_n}{n}$.

```
(%i91)  define(M([A]), '(apply("+", a)/length(A)));
(%o91)  M([A]) := apply(+,A)/length(A)

(%i92)  M(3, 6);
(%o92)  9/2

(%i93)  M(a, b, c, d);
(%o93)  (d+c+b+a)/4
```

```
(%i94)  M(M(r, s, t), M(u, v, w, x, y, z));
```

$$(\%o94)\quad \frac{\frac{z+y+x+w+v+u}{6}+\frac{t+s+r}{3}}{2}$$

Exemplo B.19 Definir uma função G que resulte na média geométrica dos seus argumentos: $G(a_1, a_2, \ldots, a_n) = \sqrt[n]{a_1 \cdot a_2 \cdot \quad \ldots \quad \cdot a_n}$.

```
(%i95)  define(G([X]), '(apply("*", X)^(1/length(X))));
```

$$(\%o95)\quad G([X]) := apply(*, X)^{\frac{1}{\mathrm{length}(X)}}$$

```
(%i96)  G(3, 7);
```

$$(\%o96)\quad \sqrt{21}$$

```
(%i97)  G(a, b, c);
```

$$(\%o97)\quad a^{1/3}b^{1/3}c^{1/3}$$

```
(%i98)  G(r, r, w, w, w, x, y, z, z);
```

$$(\%o98)\quad r^{2/9}w^{1/3}x^{1/9}y^{1/9}z^{2/9}$$

B.5 Propriedades de funções e operadores

As funções, operadores, variáveis, constantes e todo objeto que faz parte do *Maxima* possuem propriedades relacionadas com eles. Para verificar que propriedades são essas, basta usar um comando **properties(objeto)**:

```
(%i99)  properties(fpprec);
(%o99)  [system value, assign property]

(%i100)  properties(%pi);
(%o100)  [database info, numer]

(%i101)  properties("+");
(%o101)  [mirror symmetry, nary, rule, operator]
```

Uma propriedade ou uma lista de propriedades pode ser atribuída a um objeto através do comando **declare(objeto, propriedade)** ou **declare(objeto, [lista de propriedades])**. Por exemplo, declare(f, linear) atribui a propriedade "linear" ao objeto "f".

As principais propriedades reconhecidas pelo *Maxima* são:

- **additive** Se uma função f de uma variável for declarada aditiva (additive),

então f(x + y) será simplificado para f(x) + f(y). Se a função f possuir várias variáveis, então a proriedade additive afeta apenas a primeira coordenada: f(a + b, y) será simplificado para f(a, y) + f(b, y).

- **multiplicative** Se uma função f de uma variável for declarada multiplicativa (multiplicative), então f(x*y) será simplificado para f(x)*f(y). Se a função f possuir várias variáveis, então a proriedade multiplicative afeta apenas a primeira coordenada: f(a*b, y) será simplificado para f(a, y) * f(b, y).
- **outative** Se uma função f de uma variável for declarada outative, então f(a*x) será simplificado para a*f(x), onde a é uma constante. Se a função f possuir várias variáveis, então a proriedade outative afeta apenas a primeira coordenada: f(x, y) será simplificado para x * f(1, y).
- **linear** Equivale às propriedades additive e outative.
- **symmetric** Se uma função f de várias variáveis for declarada simétrica (symmetric), então a ordem dos parâmetros pode ser trocada que o valor da função não se altera: f(z, x, y) = f(x, y, z) = f(z, y, x) etc.
- **antisymmetric** Se uma função f de várias variáveis for declarada anti-simétrica (antisymmetric), então a cada troca de dois parâmetros o valor da função troca de sinal: f(z, x, y) = -f(x, z, y), f(x, z, y) = -f(x, y, z) etc.
- **commutative** É o mesmo que symmetric.
- **evenfun** Uma função f de uma variável declarada evenfun é uma função par: f(-x) = f(x).
- **oddfun** Uma função f de uma variável declarada oddfun é uma função ímpar: f(-x) = -f(x).
- **lassociative** Associatividade à esquerda: f(a, b, c) = f(f(a, b), c)
- **rassociative** Associatividade à direita: f(a, b, c) = f(a, f(b, c))
- **nary** Aplicar várias vezes uma função nary é o mesmo que aplicar apenas uma única vez: f(f(x)) = f(x), f(f(a, b), f(c, d)) = f(a, b, c, d) etc.

Exemplo B.20 Declarar uma função f como aditiva e calcular seu valor nos pontos $a + b + c$ e $x + 4$.

```
(%i102)  f(a + b + c);
(%o102)  f(c + b + a)

(%i103)  declare(f, additive);
(%o103)  done
```

```
(%i104)  f(a + b + c);
(%o104)  f(c) + f(b) + f(a)

(%i105)  f(4 + x)
(%o105)  f(x) + f(4)

(%i106)  properties(f);
(%o106)  [additive]
```

Exemplo B.21 Declarar uma função g como multiplicativa e calcular seu valor em alguns pontos como abc e $2y$.

```
(%i107)  g(a*b*c);
(%o107)  g(abc)

(%i108)  declare(g, multiplicative);
(%o108)  done

(%i109)  g(a*b*c);
(%o109)  g(a)g(b)g(c)

(%i110)  g(2*y)
(%o110)  g(2)g(y)

(%i111)  properties(g);
(%o111)  [multiplicative]
```

Exemplo B.22 Declarar uma função h como antisimétrica e calcular seu valor nos pontos (y, x, z), (x, z, y), (y, z, x), (x, x, z) e (x, y, y).

```
(%i112)  h(y, x, z);
(%o112)  h(y, x, z)

(%i113)  declare(h, antisymmetric);
(%o113)  done

(%i114)  h(y, x, z);
(%o114)  -h(x, y, z)

(%i115)  h(x, z, y);
(%o115)  -h(x, y, z)

(%i116)  h(x, x, z);
(%o116)  0
```

```
(%i117)  h(x, y, y);
(%o117)  0
```

Referências Bibliográficas

[1] Neble, M. V. R., Galván, J. R. R. (2005), *Introducción a Maxima*, Universidad de Cádiz, disponível na internet em PDF.

[2] J. R. R. Galván (2007), *Maxima con wxMaxima: software libre en el aula de matemáticas*, Oficina de Software Libre de la Universidad de Cádiz, disponível na internet em PDF.

[3] W. Haager (2011), *Graphics with Maxima*, disponível em `www.austromath.at/daten/maxima/zusatz`

[4] R. Ipanaqué (2012), *Breve Manual de Maxima*, Segunda Edición, disponível em `www.eumed.net/libros/2010c/728`

[5] Macsyma Inc. (1996), *Macsyma Mathematics and System Reference Manual*, 16^{th} ed., disponível em `www.cs.berkeley.edu/~fateman/macsyma/docs/`

[6] Macsyma Inc. (1996), *Macsyma User's Guide,* 2^{nd} ed., disponível em `www.cs.berkeley.edu/~fateman/macsyma/docs/`

[7] Macsyma Inc. (1998), *Macsyma Scientific Graphics Reference Manual,* disponível em `www.cs.berkeley.edu/~fateman/macsyma/docs/`

[8] M. R. Riotorto (2008), *Primeros pasos en Maxima*, disponível em `www.telefonica.net/web2/biomates`

[9] *Maxima Manual* (2000), manual original do Maxima, 1070 páginas, disponível em `maxima.sourceforge.net/docs/manual/en/maxima.pdf`

[10] J. E. Villate (2007), *Introdução aos Sistemas Dinâmicos – Uma abordagem prática com Maxima*, disponível em `fisica.fe.up.pt/maxima/book/sistdinam-1_2.pdf`

Índice Remissivo

Lenimar Nunes de Andrade nasceu em janeiro de 1962 em Patu, uma pequena cidade do alto sertão do Rio Grande do Norte. É Bacharel em Matemática pela Universidade Federal da Paraíba (1982), Mestre em Matemática pela Universidade Federal de Pernambuco (1987), Doutor em Engenharia Elétrica pela Universidade Estadual de Campinas (1998) e Professor Titular da Universidade Federal da Paraíba. Desde 1988 vem se interessando por linguagens de programação de computadores.