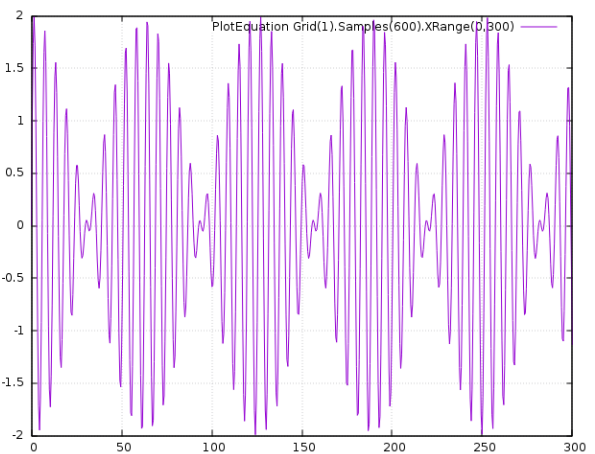
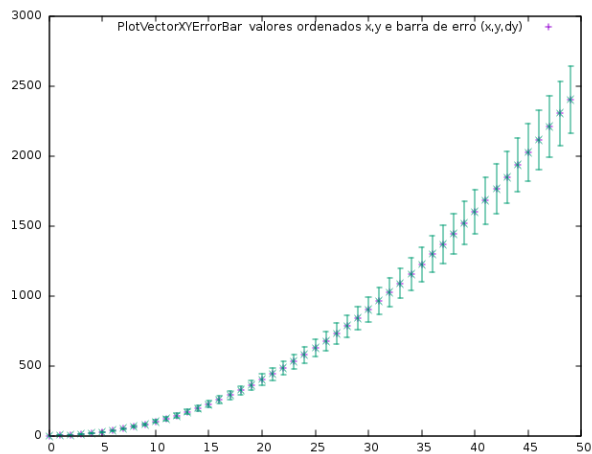
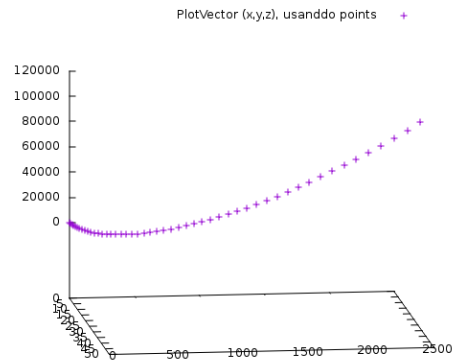
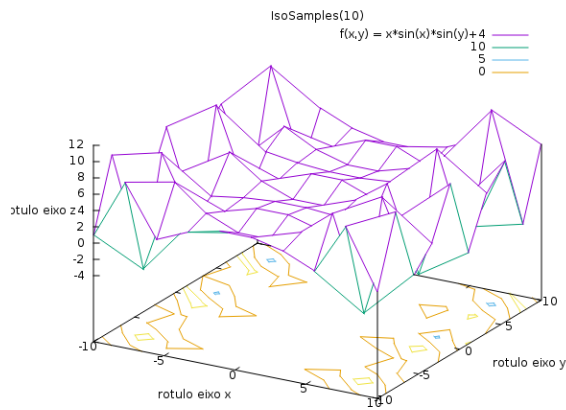


# Como Elaborar Gráficos Profissionais com Gnuplot



André Duarte Bueno,  
UENF-CCT-LENep-LDSC  
email: [bueno@lenep.uenf.br](mailto:bueno@lenep.uenf.br)

January 30, 2021

*Copyright(C) André Duarte Bueno.*

*Todos os direitos reservados e protegidos pela Lei 5.988 de 14/12/1973. É proibida a reprodução desta obra, mesmo parcial, por qualquer processo, sem prévia autorização, por escrito, do autor.*

*Editor: ANDRÉ DUARTE BUENO*

*Revisão técnica:*

*xxx*

*ISBN: XX-XXXX-XXX-X*

Este livro foi desenvolvido na UENF/CCT/LENEP/LDSC

Laboratório de Desenvolvimento de Software Científico - LDSC

<http://www.lenep.uenf.br/~ldsc>

do Laboratório de Engenharia e Exploração de Petróleo - LENEP

<http://www.lenep.uenf.br>

do Centro de Ciências e Tecnologia - CCT

<http://www.cct.uenf.br>

da Universidade Estadual do Norte Fluminense - Darcy Ribeiro - UENF

<http://www.uenf.br>

# Chapter 1

## Como Elaborar Gráficos Profissionais com Gnuplot

Apresenta-se neste capítulo o que é, como instalar e usar o software `gnuplot`.

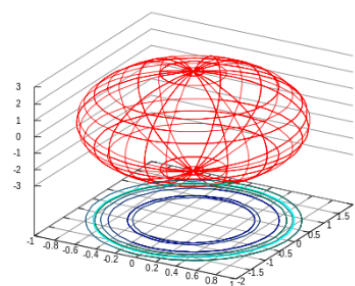
### 1.1 O que é o `gnuplot`?

- é um dos programas livres mais utilizados para se fazer gráficos profissionais no mundo do software livre.
- é um programa compatível com diversas plataformas, como GNU/LINUX, UNIX, IBM OS/2, MS Windows, DOS, Apple Macintosh, VMS, Atari, etc.
- é um programa de linha de comando, altamente interativo, com uma série de funções e comandos internos que possibilitam a construção de gráficos avançados de forma rápida e produtiva.
- permite configurar todos os detalhes do gráfico; a construção de múltiplos gráficos; salvar em vários formatos; plotar dados de arquivos de disco.
- permite criar scripts para automatizar as atividades.
- maiores detalhes no site <http://www.gnuplot.info/>.

**gnuplot** é um programa de linha de comando que pode plotar os gráficos de funções matemáticas em duas ou três dimensões, e outros conjuntos de dados. O programa pode ser executado na grande maioria dos computadores e sistemas operacionais (Linux, UNIX, Windows, Mac OS X...). Ele é um programa com uma *fairly* longa história, datando de antes de 1986. Este software não é distribuído sob a licença GPL.

`gnuplot` pode gerar saídas diretamente na tela, ou em muitos formatos de arquivos gráficos, incluindo PNG, EPS, SVG, JPEG e muitos outros. Ele também é capaz de produzir código LaTeX que possa ser incluído diretamente nos documentos LaTeX, fazendo uso de fontes LaTeX e poderosas habilidades com fórmulas. O programa pode ser usado tanto interativamente quanto através de scripts em lote (*batch mode*). Para um script de exemplo e sua saída, veja [esta espiral logarítmica](#). O programa é bem suportado e documentado. Ajuda extensiva pode ser encontrada na [internet](#)

### gnuplot



Renderização 3D de um elipsóide feito pelo `gnuplot`

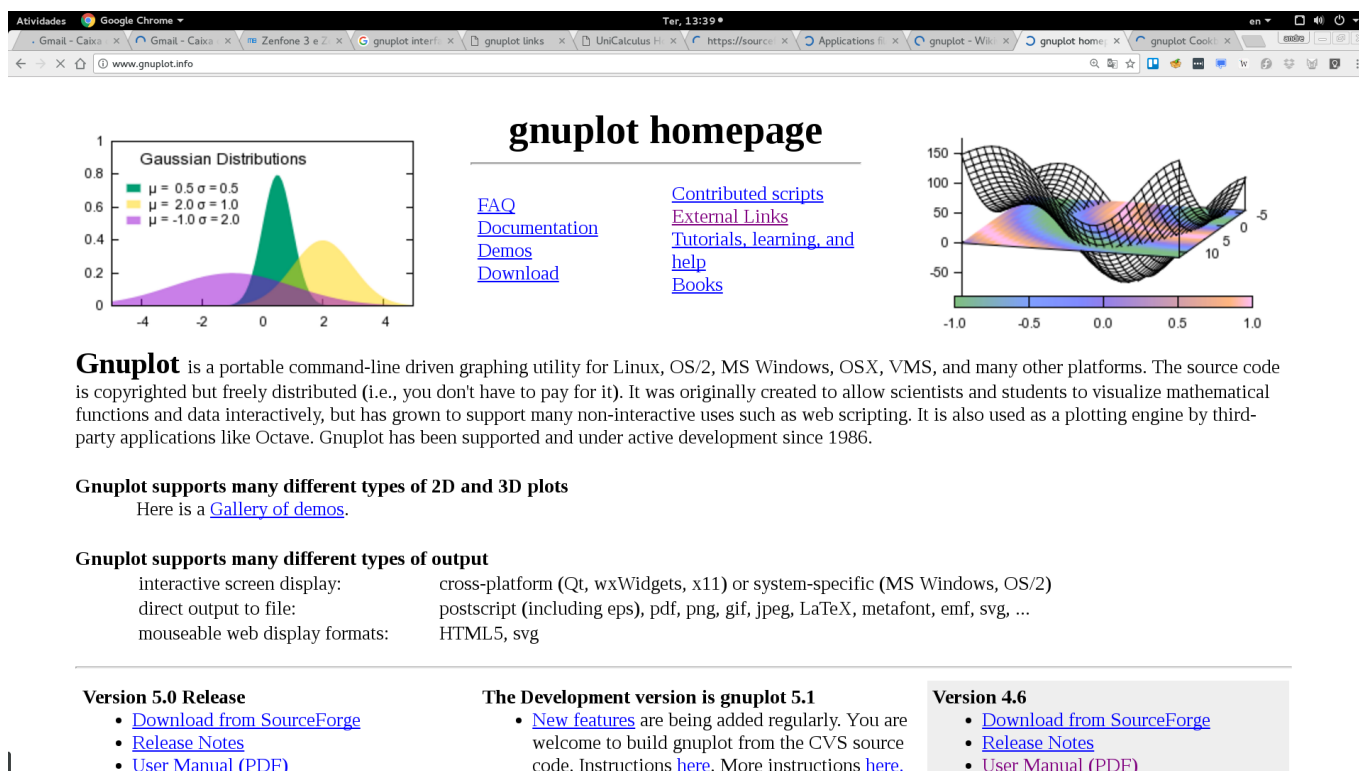
#### Lançamento ?

**Sistema operacional** multiplataforma

**Licença** licença própria

**Página oficial** <http://www.gnuplot.info/>

Fonte: Wikipedia



The screenshot shows the gnuplot homepage with a header "gnuplot homepage". On the left, there's a plot titled "Gaussian Distributions" showing three bell curves with different parameters:  $\mu = 0.5, \sigma = 0.5$  (green),  $\mu = 2.0, \sigma = 1.0$  (yellow), and  $\mu = -1.0, \sigma = 2.0$  (purple). In the center, there are links for "FAQ", "Documentation", "Demos", "Download", "Contributed scripts", "External Links", "Tutorials, learning, and help", and "Books". On the right, there's a 3D surface plot. Below the plots, a paragraph describes gnuplot as a portable command-line driven graphing utility. Further down, it lists supported 2D and 3D plot types and output formats. At the bottom, there are sections for "Version 5.0 Release" (with links to download, release notes, and user manual), "The Development version is gnuplot 5.1" (with links to new features and instructions), and "Version 4.6" (with links to download, release notes, and user manual).

**Gnuplot** is a portable command-line driven graphing utility for Linux, OS/2, MS Windows, OSX, VMS, and many other platforms. The source code is copyrighted but freely distributed (i.e., you don't have to pay for it). It was originally created to allow scientists and students to visualize mathematical functions and data interactively, but has grown to support many non-interactive uses such as web scripting. It is also used as a plotting engine by third-party applications like Octave. Gnuplot has been supported and under active development since 1986.

**Gnuplot supports many different types of 2D and 3D plots**  
Here is a [Gallery of demos](#).

**Gnuplot supports many different types of output**

interactive screen display:	cross-platform (Qt, wxWidgets, x11) or system-specific (MS Windows, OS/2)
direct output to file:	postscript (including eps), pdf, png, gif, jpeg, LaTeX, metafont, emf, svg, ...
mouseable web display formats:	HTML5, svg

**Version 5.0 Release**

- [Download from SourceForge](#)
- [Release Notes](#)
- [User Manual \(PDF\)](#)

**The Development version is gnuplot 5.1**

- [New features](#) are being added regularly. You are welcome to build gnuplot from the CVS source code. Instructions [here](#). More instructions [here](#).

**Version 4.6**

- [Download from SourceForge](#)
- [Release Notes](#)
- [User Manual \(PDF\)](#)

Homepage do gnuplot



## **Gnuplot in Action** **Second Edition**

by Philipp K. Janert

*Updated for gnuplot*  
**5**

Manning Publications (2016)  
ISBN: 1633430189  
ISBN-13: 9781633430181



## **gnuplot Cookbook**

by Lee Phillips

Packt Publishing (2012)  
ISBN : 184951724X  
ISBN-13 : 9781849517249

Livros do gnuplot

### 1.1.1 Novidades do gnuplot 4.0

- Eventos do mouse e teclas de atalho (consulte "help bind").
- Novo terminal 'epslatex', possibilita inserção de figuras eps em documentos L<sup>A</sup>T<sub>E</sub>X.
- O comando 'splot' é capaz de plotar mapas 2D e superfícies 3D em tons de cinza ou coloridas . Veja "help pm3d", "help palette", "help cbrange", 'set view map', 'set colorbox' and palette.
- Novo comando 'set datafile' permite setar informações de formato sobre o arquivo de dados.
- Outras novidades incluem: 'boxes', 'boxerrorbars', 'boxxyerrorbars', 'candlesticks', 'set style fill', 'frequency', 'unique', 'labels', 'history'.

### 1.1.2 Novidades do gnuplot 5.0

- Veja [http://gnuplot.info/ReleaseNotes\\_5\\_0.html](http://gnuplot.info/ReleaseNotes_5_0.html).
-

### 1.1.3 Instalando o gnuplot

- A opção mais simples é ir no site do gnuplot, <http://www.gnuplot.info>, e seguir as instruções para baixar e instalar e versão executável.
- Se você usa GNU/Linux pode usar o sistema de instalação de pacotes, no Fedora.

Exemplo:

```
dnf install gnuplot
dnf install gnuplot-doc
dnf install gnuplot-devel
```

- A opção para quem quer ter um programa mais rápido, mais eficiente, é baixar e instalar o **gnuplot** a partir do código fonte. Neste caso o **gnuplot** vai levar em conta todas as características de seu computador, como a versão específica do processador. O primeiro passo é obter uma cópia dos arquivos com o código fonte do **gnuplot**.
- Veja a seguir a sequência para fazer o download via ftp<sup>1</sup> no servidor <ftp.gnuplot.info>. Note que depois de estabelecida a conexão com o servidor, envio comandos para o servidor (ex: `cd pub/gnuplot`).

Exemplo:

```
ftp> open ftp.gnuplot.info
Connected to ftp.gnuplot.info (128.173.8.161).
220 ftp.gnuplot.info NcFTPd Server (unregistered copy) ready.
Name (ftp.gnuplot.info:andre): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password: bueno@lenep.uenf.br
230-You are user #1 of 150 simultaneous users allowed.
230-Welcome to ftp.gnuplot.info!
230 Logged in anonymously.
Remote system type is UNIX.
Using binary mode to transfer files.

ftp> cd pub/gnuplot
ftp> get gnuplot-5.6.0.tar.gz
ftp> quit
221 Goodbye.
```

- Uma alternativa é usar o **wget**, como em:

```
wget ftp.gnuplot.info/pub/gnuplot/gnuplot-5.6.0.tar.gz
```

- A seguir basta descompactar o arquivo:

```
tar -xvzf gnuplot-5.6.0.tar.gz
```

- e executar a sequência

```
./configure # Configura para minha máquina
make # Compila, gerando o executável
make install # Instala o gnuplot (como usuário root).
```

## 1.2 Comandos Básicos

Veremos alguns comandos básicos e a seguir como fazer gráficos 2D e depois como modificar as propriedades do gráfico.

---

<sup>1</sup>ftp é um programa que é usado para fazer a transferência de arquivos (*File Transfer Protocol*). É necessário um programa cliente ftp na sua máquina e um servidor ftp na máquina que será acessada.

### 1.2.1 Entrando e saindo do gnuplot

- Para entrar e sair do gnuplot é muito fácil, abra um terminal e execute o comando *gnuplot*. Note que aparece "gnuplot>"

Exemplo:

```
$gnuplot
G N U P L O T
Version 5.2 patchlevel 8      last modified 2019-12-01
Copyright (C) 1986-1993, 1998, 2004, 2007-2019
Thomas Williams, Colin Kelley and many others
gnuplot home:      http://www.gnuplot.info
faq, bugs, etc:    type "help FAQ"
immediate help:    type "help" (plot window: hit 'h')
Terminal type is now 'qt'
```

- Para plotar um gráfico

```
gnuplot> plot sin(x)
```

- Para executar um comando de terminal de dentro do gnuplot digite !comando:

**Protótipo:**

```
gnuplot> !comandoDeTerminal
```

Exemplo:

```
gnuplot> !pwd
gnuplot> !ls
```

- Para sair digite quit:

```
gnuplot> quit
```

### 1.2.2 Pedindo ajuda

- Depois de entrar no gnuplot, você pode pedir ajuda:

```
help
help nome_do_comando
help plot
```

- Para ver as configurações atuais (variáveis e funções atuais)

```
gnuplot> show all
```

- Para obter uma lista completa dos comandos do gnuplot consulte o manual do usuário, veja os exemplos apresentados no diretório demo ou consulte o grupo de discussão comp.graphics.apps.gnuplot.

### 1.2.3 Uso do mouse

- Gráficos 2D

- Ao mover o mouse sobre o gráfico, as coordenadas são apresentadas no canto esquerdo da tela.
- Um click com o botão do meio do mouse coloca as coordenadas no gráfico.
- Clicar e arrastar o mouse com o botão direito permite selecionar uma região para zoom.
- Exemplo: Execute o comando abaixo e teste o uso do mouse.

```
gnuplot> plot sin(x)
```

- Gráficos 3D

- Use o botão esquerdo do mouse para rotacionar o gráfico.
- O movimento na vertical do botão do meio do mouse altera a altura do gráfico.
- O movimento na horizontal muda o tamanho do gráfico.
- Exemplo: Execute o comando abaixo e teste o uso do mouse.

```
gnuplot> splot sin(x)*cos(x)
```

- Como mudar a visualização

- A partir da versão 4.0 você pode alterar os angulos de visualização diretamente com o mouse (clicar com botão esquerdo e mover o mouse).
- Nas versões anteriores ou nos scripts pode-se mudar a visualização com o parâmetro view.

Exemplo:

```
set view angulo_horizontal, angulo_vertical, zoom
```

## 1.2.4 Uso das teclas de atalho

- Para ver as teclas de atalho consulte “help bind”, pressione ‘h’ para um help interativo.

m	ativa/desativa uso mouse
g	grid
l	escala logarítmica
e	replot
r	mostra eixos na posição do cursor (2D)
5	para cordenadas polares
a	para autoscale
labels	botão do meio
space	muda para linha de comando

## 1.3 Gráficos 2D

### 1.3.1 Como plotar gráficos 2D

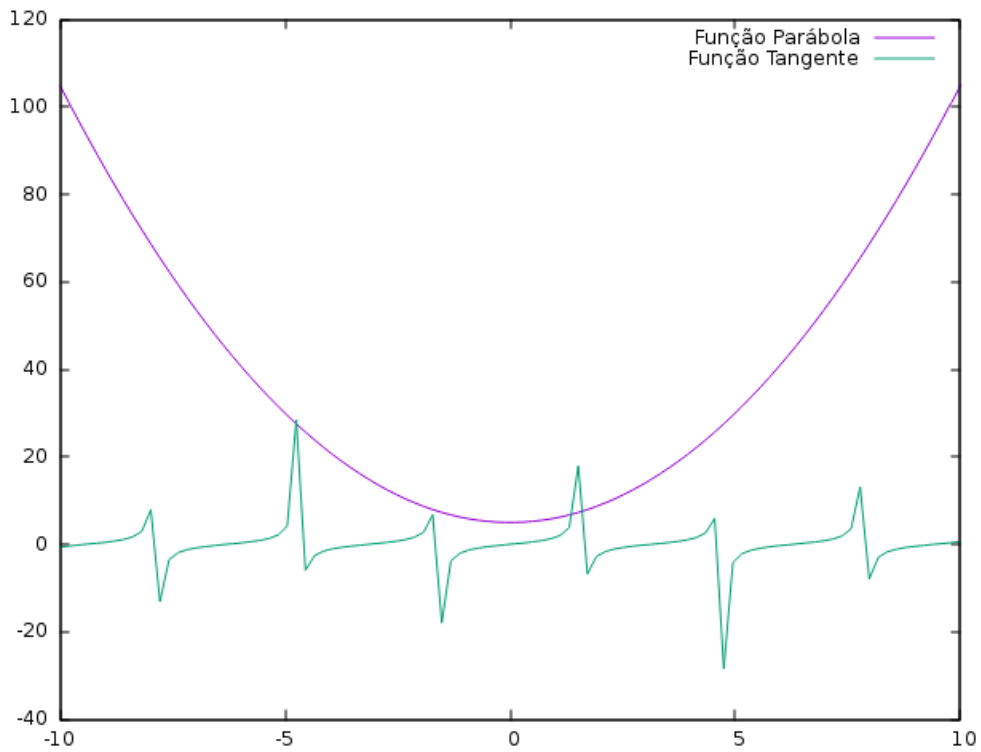
- O comando plot é o comando básico para gerar gráficos 2D.

**Syntaxe:**

```
plot {[intervalo]} {[função] | {"[arquivoDeDados]" {modificadores-arquivo}}}  
{axes [eixos] } { title "título" } {with [estilos] } {, {definitions,} [função] ...}
```

Exemplo:

```
plot x*sin(x)
plot 5 + 2*x + x**2 + x**3
# Adicionando título
plot 5 + 2*x + x**2 + x**3 title "Função Parábola"
# Plotando duas funções
plot 5 + x**2 title "Função Parábola" with lines, tan(x) title "Função Tangente"
# Modificando vários parâmetros
plot [-2:2] cos(x) w l lt 1 lw 5 t  
"[2:2] Intervalo, Função, With Lines, Linha tipo 1 (lt 1), largura 5 (lw 5) e titulo"
```



## 1.4 Como definir as propriedades do gráfico

- Você pode customizar absolutamente tudo em um gráfico do gnuplot.
- Para a lista completa de itens que podem ser customizados, veja o manual.

Exemplo:

```
plot x**2 title "Parabola x^2"
set xrange [-2:2]
set title "titulo do gráfico"
replot
```

### 1.4.1 Títulos e mensagens

- Para definir o título

```
set title "titulo do gráfico"
```

- Para definir o título do eixo x

```
set xlabel "titulo do eixo x (unidade)"
```

- Para definir o título do eixo y

```
set ylabel "titulo do eixo y (unidade)"
```

- Para inserir um texto/mensagem numa posição específica do gráfico

```
set label "mensagem" at posição_x, posição_y
```

- Para remover todas as mensagens

```
unset label
```



### 1.4.2 Eixos

- Para definir o intervalo do eixo x

```
set xrange [x_min:x_máx]
```

- Para definir o intervalo do eixo y

```
set yrange [y_min:y_máx]
```

- Deixando o gnuplot definir o intervalo automaticamente

```
set autoscale
```

- Para incluir o eixo x0

```
set xzeroaxis  
unset xzeroaxis
```

- Para incluir o eixo y0

```
set yzeroaxis  
unset yzeroaxis
```

### 1.4.3 Escalas

- Para usar escala logarítmica

```
set logscale
```

- Para definir apenas o eixo y como sendo logarítmico

```
unset logscale; set logscale y
```

- Para usar escalas automaticas

```
set autoscale  
unset autoscale  
show autoscale
```

### 1.4.4 Origem

- Para definir a origem

```
set origin 0.5 , 0.5
```

### 1.4.5 Marcadores (tics)

- Para alterar o número de marcadores

```
set xtics (1, 2, 3, 4)  
set xtics ("jan" 1, "fev" 2, "mar" 3,...)
```

- Para retornar o número de marcadores para o default

```
unset xtics;  
set xtics
```

### 1.4.6 Bordas

- Para definir as bordas

```
set border  
unset border  
show border
```

### 1.4.7 Legenda

- Opções de posição da legenda:
  - inside/outside, left/center/right, top/center/bottom
- Opções de borda na legenda:
  - nobox/box
- Opção sem legenda:
  - nokey

Exemplo:

```
plot sin(x)
set key inside left top
replot
set key inside left top box
replot
set nokey
replot
set key at 25., 50. # posição específica
set key at graph 0.3, 0.7 # posição específica relativa a área gráfico
```

### 1.4.8 Estilos de linha

- Para definir o estilo da linha

Protótipo:

plot função with `estilo_da_linha`

Exemplo:

```
plot x**2 with points
plot x**2 with lines
plot x**2 with linespoints
plot x**2 with dots
plot x**2 with impulses
plot x**2 with steps
```

Outros estilos de linha incluem:

- `filledcurves`, `fsteps`, `histeps`, `errorbar`, `boxes`, `boxerrorbar`, `vector`, `financebars`, `candlesticks`, `xerrorbar`,
- `xyerrorlines`, `errorlines`, `xerrorlines`, `yerrorbars`, `labels`, `xyerrorbars`, `yerrorlines`, `surface`, `vectors`, `parallelaxes`.
- Também é possível definir o tipo de linha com `lt n`, sendo `n` um número de 0-20.

```
plot x**2 lt 5
```

- É possível definir o tipo de cor com `lc n`

```
plot x**2 lc 5
```

- E a largura da linha com `lw n`. Veja o exemplo abaixo:

```
plot x**2 lc 3 lw 7
plot "Fluid-A.dat" using 1:3 title "C#REF-P#L-F#A-S#NR" with linespoints lt 2 lc 2,
"Fluid-A.dat" using 1:5 title "C#REF-P#W-F#A-S#NR" with linespoints lt 3 lc 4
```

- Outro exemplo:

```
plot [-2:2] cos(x) w l lt 1 lw 5 t \
"[2:2] Intervalo, Função, With Lines, Linha tipo 1 (lt 1), largura 5 (lw 5) e titulo"
```

### 1.4.9 Posição do cursor

- Para mover o cursor

```
set key 0.01,100
```

- Nome da função

```
set key top left
```

- Para eliminar o cursor

```
unset key
```

### 1.4.10 Ângulos

- Para definir o formato dos ângulos...

```
set angles {degrees | radians}  
show angles
```

### 1.4.11 Codificação de caracteres

- Define codificação de caractere

```
set encoding iso_8859_1
```

### 1.4.12 Localidade

- Para definir a localidade (país)

```
set locale "pt_BR"
```

### 1.4.13 Para plotar gráficos com barras de erros

- Para plotar gráficos com barras de erros

```
plot "arq.dat" using 1:2:3:4 with errorbars
```

### 1.4.14 Para incluir data e hora no gráfico

- Para incluir a data e hora no canto esquerdo

```
set time
```

### 1.4.15 Pausa

- Para realizar uma pausa a espera de comandos do usuário

```
pause <tempoSegundos> ['mensagem']
```

### 1.4.16 Bordas

- Para incluir bordas

```
set border  
unset border
```

- Para definir as margens

```
set lmargin 2  
set rmargin 2  
set bmargin 3  
set tmargin 3  
show margin
```

### 1.4.17 Tics

- Para incluir/eliminar tics

```
set tics / unset tics
set xtics / unset xtics
set ytics / unset ytics
```

### 1.4.18 Como definir o mapeamento e o tipo de coordenada

- Para definir as coordenadas de mapeamento

```
set mapping {cartesian | spherical | cylindrical}
```

- Para definir as coordenadas polar/cartesiana

```
set polar / unset polar
```

- Para definir o formato dos ângulos

```
set angles degrees
set angles radians
```

## 1.5 Funções

### 1.5.1 Funções internas do gnuplot

- Como você deve ter observado (uso da função seno), o gnuplot tem um conjunto de funções internas, que podem ser diretamente acessadas.
- As expressões matemáticas utilizadas nas linguagens C, FORTRAN, Pascal, e BASIC são aceitas.
- Os operadores são os mesmos da linguagem C (exceto exponenciação, que é realizada com \*\*, como em fortran).
- A precedência dos operadores obedece a ordem da linguagem C.

Função	Retorno
<code>abs(x)</code>	valor absoluto de x, $ x $ (x pode ser complexo)
<code>acos(x)</code>	arco-coseno de x
<code>asin(x)</code>	arco-seno de x
<code>atan(x)</code>	arco-tangente de x
<code>cos(x)</code>	coseno de x, x é um radiano
<code>cosh(x)</code>	coseno hiperbólico de x, x é um radiano
<code>erf(x)</code>	função erro de x
<code>exp(x)</code>	função exponencial de x, base e
<code>inverf(x)</code>	função erro invertida de x
<code>invnorm(x)</code>	distribuição normal invertida de x
<code>log(x)</code>	log de x, base e
<code>log10(x)</code>	log de x, base 10
<code>norm(x)</code>	função distribuição normal (ou Gaussiana)
<code>rand(x)</code>	gerador de números pseudo-randômicos
<code>sgn(x)</code>	1 se $x > 0$ , -1 se $x < 0$ , 0 se $x=0$
<code>sin(x)</code>	seno de x, x é um radiano
<code>sinh(x)</code>	seno hiperbólico de x, x é um radiano

`sqrt(x)` raiz quadrada de x

`tan(x)` tangente de x, x é um radiano

`tanh(x)` tangente hiperbólica de x, x é um radiano

- Outros exemplos incluem as funções Bessel, gamma, ibeta, igamma, lgamma e os operadores binários e unários (consulte o manual do gnuplot para obter uma lista completa e atualizada das funções internas).

### 1.5.2 Como definir e usar suas funções

- Os exemplos a seguir esclarecem como você deve proceder para criar suas funções.
- Para definir uma variável (obs: emissividade cerâmica=0.93)

```
boltzmann=5.6697e-8
ec=0.93
```

- Para criar uma função  $EmissaoRadiacao(T) = ec * boltzman * T^4$ ,

```
EmissaoRadiacao(T)=ec*boltzmann*T**4
plot EmissaoRadiacao(x)
```

- Para criar uma função  $f3D(x, y) = x^2 + y^2$ ,

```
f3D(x,y) = x**2 + y**2
splot f3D(x,y)
```

## 1.6 Recursos avançados do gnuplot

Esta seção apresenta alguns usos avançados do gnuplot.

### 1.6.1 Como plotar vários gráficos em um terminal

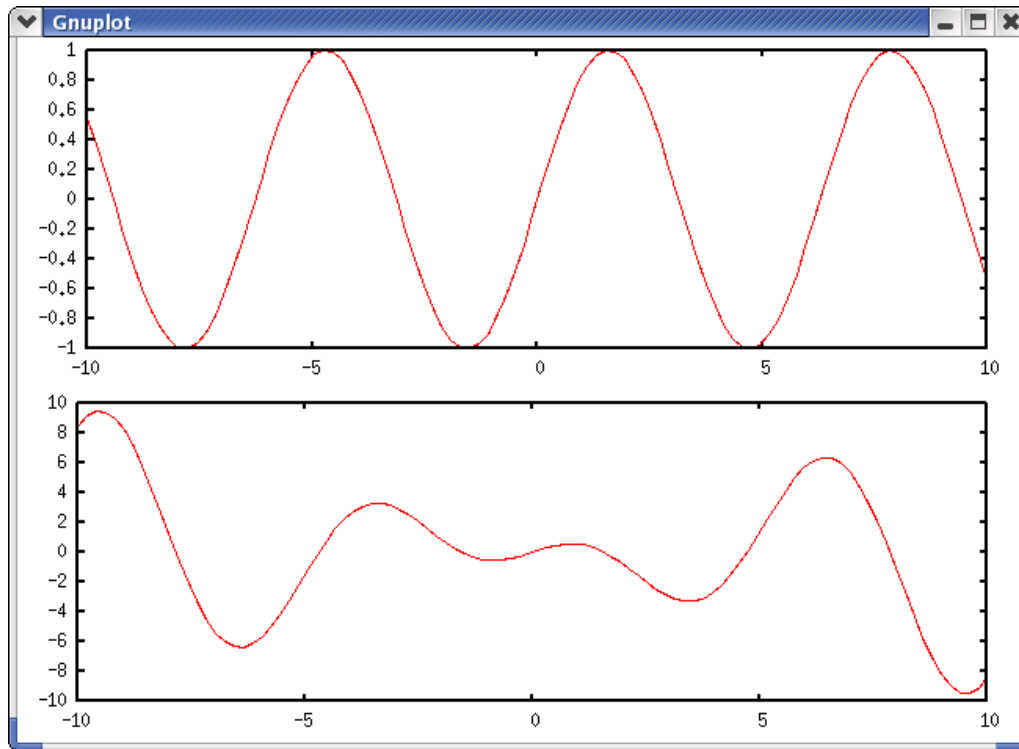
- Você pode colocar mais de um gráfico em um mesmo terminal, para tal, basta setar o atributo `multiplot`.

**Protótipo:**

```
gnuplot> set multiplot;
```

Exemplo:

```
multiplot> set multiplot
multiplot> set size 1,0.5 # x=100% y=50%
multiplot> set origin 0.0,0.5 # parte superior
multiplot> plot sin(x)
multiplot> set origin 0.0,0.0 # parte inferior
multiplot> plot cos(x)*x
multiplot> unset multiplot
```



### 1.6.2 Como gerar gráficos 2D a partir de arquivos de dados armazenados em disco

- Embora o gnuplot possibilite o uso de suas funções internas e a criação de funções do usuário, muitas vezes é necessário fazer um gráfico com dados armazenados em um arquivo de disco.
- Veja a seguir um arquivo com dados de temperatura média anual, e os comandos do gnuplot para fazer o gráfico.
- Arquivo de dados (temperatura.dat)

```
2000 26
2001 25
2002 29
2003 31
2004 29
```

- Comandos do gnuplot

Exemplo:

```
set xlabel "Ano"
set ylabel "Temperaturas (Graus Celsius)"
set yrange [20:40]
set xrange [1999:2005]
plot 'AnoXTemperatura.dat' using 1:2 title "Temp. médias anuais" with linespoint
```

### 1.6.3 Como exportar os gráficos em outros formatos

- Embora a saída padrão do gnuplot seja uma tela do X-Window, o gnuplot possibilita que os gráficos sejam diretamente gerados nos mais diferentes formatos.
- Os formatos de saída mais utilizados são:
  - set term canvas #HTML Canvas object
  - set term cgm #Computer Graphics Metafile
  - set term corel #EPS format for CorelDRAW
  - set term emtex #LaTeX picture environment with emTeX specials
  - set term epscairo #eps terminal based on cairo
  - set term epslatex #LaTeX picture environment using graphicx package
  - set term fig #FIG graphics language for XFIG graphics editor

- set term gif #GIF images using libgd and TrueType fonts
  - set term jpeg #JPEG images using libgd and TrueType fonts
  - **set term latex #LaTeX picture environment**
  - set term pdfcairo #pdf terminal based on cairo
  - **set term png #PNG images using libgd and TrueType fonts**
  - set term pngcairo #png terminal based on cairo
  - **set term postscript #PostScript graphics, including EPSF embedded files (\*.eps)**
  - set term pslatex #LaTeX picture environment with PostScript \specials
  - set term pstex #plain TeX with PostScript \specials
  - set term pstricks #LaTeX picture environment with PSTricks macros
  - **set term qt #Qt terminal**
  - set term texdraw #LaTeX texdraw environment
  - set term x11 X11 #Window System
  - set term xlib X11 #Window System (gnulib\_x11 dump)
- O exemplo a seguir ilustra como proceder para carregar um gráfico de um script e a seguir enviar a saída para um arquivo no formato postscript (extensão ps).

Abre o gnuplot e gera o gráfico

```
$gnuplot
```

```
load 'temperatura.gnuplot'
```

Define o nome do arquivo de saída

```
set out 'temperatura.ps'
set size 1, 0.5
```

Define o terminal (formato da saída)

```
set term postscript portrait enhanced mono lw 2 "Helvetica" 14
```

Refaz o gráfico enviando-o para o arquivo de disco

```
replot
!display temperatura.ps
```

Você pode enviar o gráfico direto para a impressora padrão

```
gnuplot> set out "|lpr"
gnuplot> replot
```

Ou para impressora específica

```
gnuplot> set out "|lpr -PnomeImpressora"
```

Deixa o terminal como sendo o X11

```
gnuplot> set terminal X11
gnuplot> set size 1,1
```

Deixa o terminal como sendo o qt (padrão)

```
gnuplot> set terminal qt
```

### 1.6.4 Plotando gráficos com diferentes números de amostragens - tables

O exemplo a seguir mostra o uso de tables. Note que vai gerar arquivos de disco com a saída de dados, samples3.dat, samples6.dat, samples11.dat, e depois plotar todos eles simultaneamente.

- Para gerar o gráfico no gnuplot com diferentes amostragens:

```
set samples 3
set table 'samples3.dat'
plot [-2:2] 2 - 3*x + 4*x*x - 1*x*x*x with impulses
plot [-2:2] 'samples3.dat' with lines
set samples 6
set table 'samples6.dat'
```

```

plot [-2:2] 2 - 3*x + 4*x*x - 1*x*x*x with impulses
plot [-2:2] 'samples6.dat' with lines
set samples 11
set table 'samples11.dat'
plot [-2:2] 2 - 3*x + 4*x*x - 1*x*x*x with impulses
plot [-2:2] 'samples11.dat' with lines
unset table
set samples 101
plot [-2:2] 'samples3.dat' with lines title "2 - 3*x + 4*x*x - 1*x*x*x (amostragem 2)", 'samples6.dat'
0 exemplo

```

Veja Figura 1.1 e 1.2.

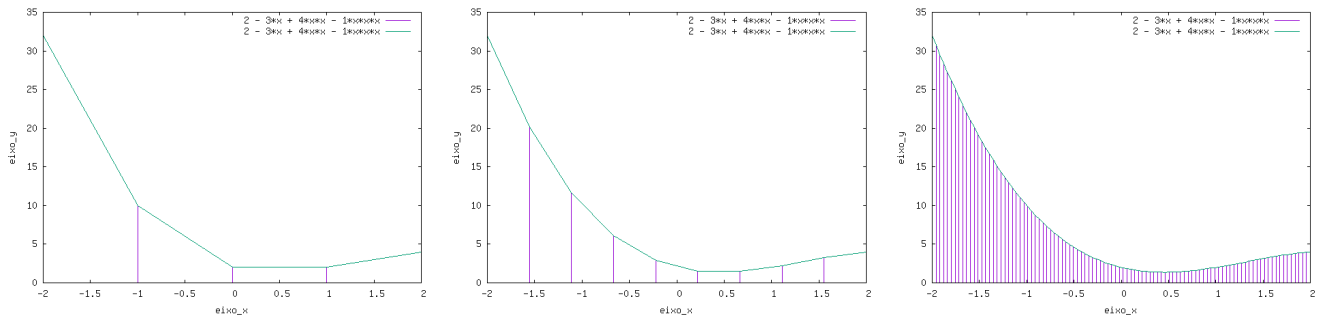


Figure 1.1: Curva de terceiro grau com diferentes amostragens

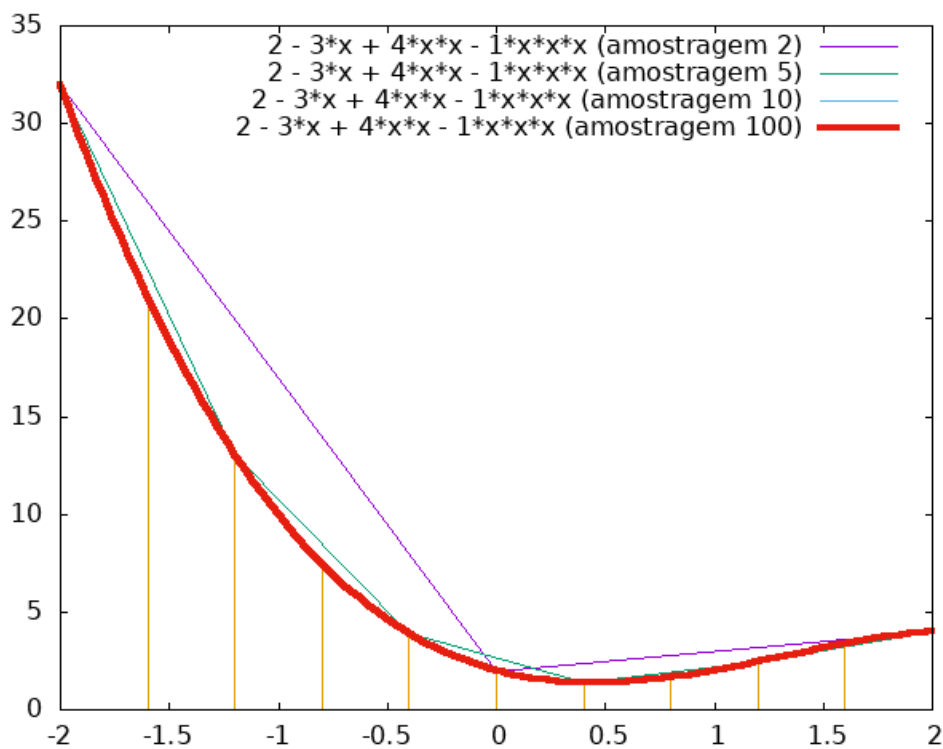


Figure 1.2: Curva de terceiro grau com diferentes amostragens - comparação direta

## 1.7 Scripts

O uso de scripts torna o uso do gnuplot muito mais poderoso, pois poderemos criar arquivos com os comandos de plotagens. Estes arquivos podem ser lidos e executados pelo gnuplot. Também é possível tornar os mesmos executáveis.



### 1.7.1 Como criar e usar scripts (macros)

- Com certeza o uso interativo do gnuplot é extremamente útil, mas você pode automatizar tarefas repetitivas utilizando o conceito de scripts.
- Um script nada mais é do que uma sequência de comandos para o gnuplot armazenadas em um arquivo de texto.
- Exemplo, abra um editor de texto, digite o texto a seguir e salve como `temperatura.gnuplot`:

```
# !gnuplot
# Script do gnuplot para plotar os dados de
# temperatura do arquivo "temperatura.gnuplot"

set title "Temperaturas médias anuais"
set xlabel "Ano"
set ylabel "Temperaturas (Graus Celsius)"

# set key 0.01,100

set label "Máxima" at 2003,32
set arrow from 2003,35 to 2003,33
set xr [2000:2005]
set yr [20:40]
plot "AnoXTemperatura.dat" using 1:2 title "temp" with linespoints
```

Veja na listagem 1.1 como ficou o arquivo salvo.

Listing 1.1: Exemplo de script do gnuplot

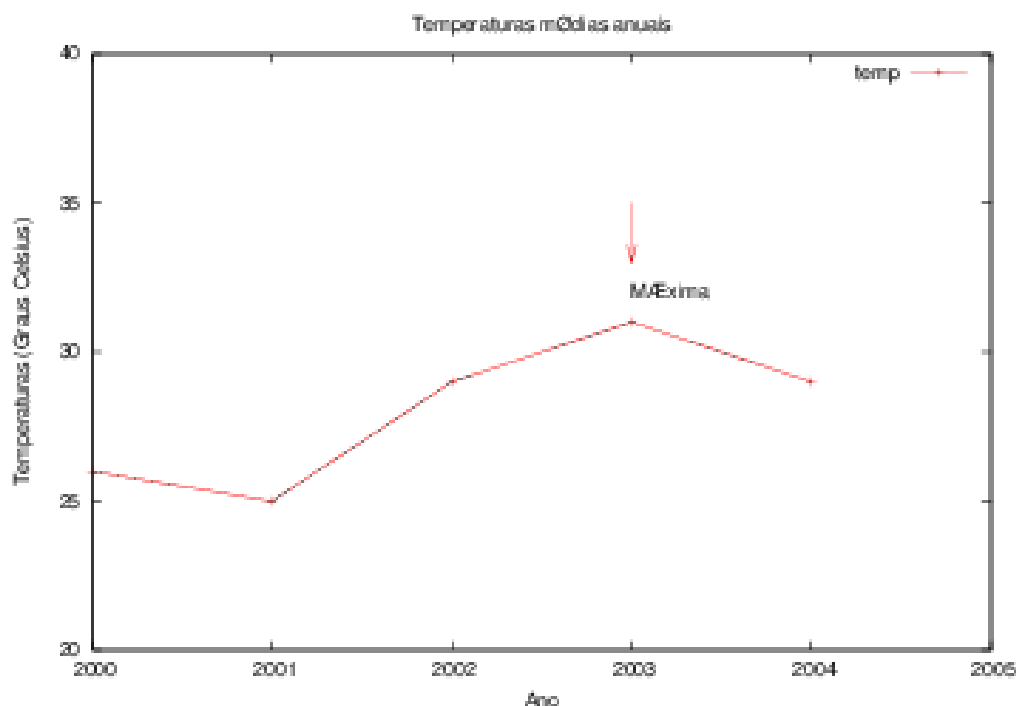
```
1# !gnuplot
2# Script do gnuplot para plotar os dados de
3# temperatura do arquivo "temperatura.gnuplot"
4set title "Temperaturas Médias Anuais"
5set xlabel "Ano"
6set ylabel "Temperaturas (Graus Celsius)"
7# set key 0.01,100
8set label "Máxima" at 2003,32
9set arrow from 2003,35 to 2003,33
10set xr [2000:2005]
11set yr [20:40]
12plot "AnoXTemperatura.dat" using 1:2 title 'temp' with linespoints
```

- Para executar o script basta abrir o gnuplot

```
$ gnuplot
```

- e a seguir carregar o script:

```
gnuplot> load 'temperatura.gnuplot'
```



- Outra forma de executar o script é digitar diretamente no terminal o comando para abrir o gnuplot e já executar o script:

```
gnuplot AnoXTemperatura.gnuplot
```

- Note que digitamos o script diretamente no editor de texto, uma outra forma de montar o script é executar o gnuplot, executar a sequência de comandos necessários para gerar o gráfico e a seguir salvar o script em disco.
- Para salvar todos os dados atuais em um arquivo de script

```
gnuplot> save 'AnoXTemperatura.gnuplot'
```

Salva o script com o nome AnoXTemperatura-salvo.gnuplot.

- Para salvar as funções ou as variáveis ou ainda as definições, use:

```
save functions 'funções.gnuplot'
save var 'variáveis.gnuplot'
save set 'definições.gnuplot'
```

- Outra forma de gerar o script é fazer o gráfico no gnuplot e depois salvar o mesmo usando o comando save.

## 1.7.2 Exemplo

Veja na listagem 1.2 exemplo de arquivo de dados gerado por um software que simula as curvas de histerese em processos de embebição/drenagem de rochas reservatório de petróleo.

Listing 1.2: Exemplo de arquivo de dados

13 Passo	Raio	Fluido_B	Fluido_A	F_A_isolado	Fluido_B_Tot	Fluido_A_Tot	Tempo(ms)
14 1	0	0	98.0698	1.93024	0	100	11824
15 2	1	0.22168	97.7962	1.98217	0.22168	99.7783	21061
16 3	2	11.4342	79.3704	9.19541	11.4342	88.5658	29554
17 4	3	37.7157	16.8426	45.4417	37.7157	62.2843	37527
18 5	4	40.5001	6.59595	52.904	40.5001	59.4999	46460
19 6	5	41.9434	2.9703	55.0863	41.9434	58.0566	55486
20 7	6	42.7891	1.81277	55.3981	42.7891	57.2109	64237
21 8	7	43.4637	0.762024	55.7743	43.4637	56.5363	72212
22 9	8	43.4637	0.762024	55.7743	43.4637	56.5363	80387
23 10	9	43.4637	0.762024	55.7743	43.4637	56.5363	89378
24 11	10	43.4637	0.762024	55.7743	43.4637	56.5363	98033
25 12	11	43.4637	0.762024	55.7743	43.4637	56.5363	106129
26 13	12	43.4637	0.762024	55.7743	43.4637	56.5363	114513
27 14	13	43.4637	0.762024	55.7743	43.4637	56.5363	123285
28 15	13	43.1335	56.8593	0.00722413	43.1407	56.8593	138140
29 16	12	43.1272	56.8656	0.00722413	43.1344	56.8656	152219
30 17	11	43.1142	56.8786	0.00722413	43.1214	56.8786	166340
31 18	10	43.095	56.8978	0.00725903	43.1022	56.8978	179096
32 19	9	43.0465	56.9457	0.00781742	43.0543	56.9457	191055
33 20	8	42.9775	57.0147	0.00781742	42.9853	57.0147	201329
34 21	7	42.8491	57.1427	0.00823621	42.8573	57.1427	211888
35 22	6	42.2632	57.7279	0.00882949	42.2721	57.7279	222391
36 23	5	41.5083	58.4791	0.0126335	41.5209	58.4791	232069
37 24	4	40.0853	59.8808	0.0338871	40.1192	59.8808	241352
38 25	3	37.307	62.4632	0.229811	37.5368	62.4632	250827
39 26	2	12.9776	74.9533	12.0691	25.0467	74.9533	260495
40 27	1	0.818351	82.1077	17.0739	17.8923	82.1077	270181
41 28	0	0	82.9261	17.0739	17.0739	82.9261	279830
42 29	0	17.0739	79.2418	3.68431	17.0739	82.9261	291105
43 30	1	17.2954	78.9684	3.73624	17.2954	82.7046	299637
44 31	2	23.5939	62.4641	13.9419	23.5939	76.4061	308836
45 32	3	36.7926	16.6856	46.5218	36.7926	63.2074	316610
46 33	4	39.4357	6.58031	53.984	39.4357	60.5643	325848
47 34	5	40.8652	2.96842	56.1663	40.8652	59.1348	334127
48 35	6	41.7094	1.81242	56.4782	41.7094	58.2906	343325
49 36	7	42.3837	0.762024	56.8543	42.3837	57.6163	351699
50 37	8	42.3837	0.762024	56.8543	42.3837	57.6163	361229
51 38	9	42.3837	0.762024	56.8543	42.3837	57.6163	369587
52 39	10	42.3837	0.762024	56.8543	42.3837	57.6163	378597
53 40	11	42.3837	0.762024	56.8543	42.3837	57.6163	386915
54 41	12	42.3837	0.762024	56.8543	42.3837	57.6163	396053
55 42	13	42.3837	0.762024	56.8543	42.3837	57.6163	405158
56 43	13	41.5233	57.9393	0.537378	42.0607	57.9393	420688
57 44	12	41.517	57.9456	0.537378	42.0544	57.9456	434526
58 45	11	41.504	57.9586	0.537378	42.0414	57.9586	448123
59 46	10	41.4848	57.9778	0.537413	42.0222	57.9778	459879

60	47	9	41.4363	58.0257	0.537971	41.9743	58.0257	470656
61	48	8	41.3673	58.0947	0.537971	41.9053	58.0947	481339
62	49	7	41.2389	58.2227	0.53839	41.7773	58.2227	490995
63	50	6	40.6531	58.808	0.538983	41.192	58.808	500400
64	51	5	39.8981	59.5591	0.542787	40.4409	59.5591	509638
65	52	4	38.4751	60.9609	0.564041	39.0391	60.9609	518466
66	53	3	35.6835	63.5432	0.773261	36.4568	63.5432	527241
67	54	2	11.2958	76.0298	12.6744	23.9702	76.0298	536738
68	55	1	0.818351	82.063	17.1186	17.937	82.063	544921
69	56	0	0	82.8814	17.1186	17.1186	82.8814	553841

Veja na listagem 1.3 exemplo de script do gnuplot.

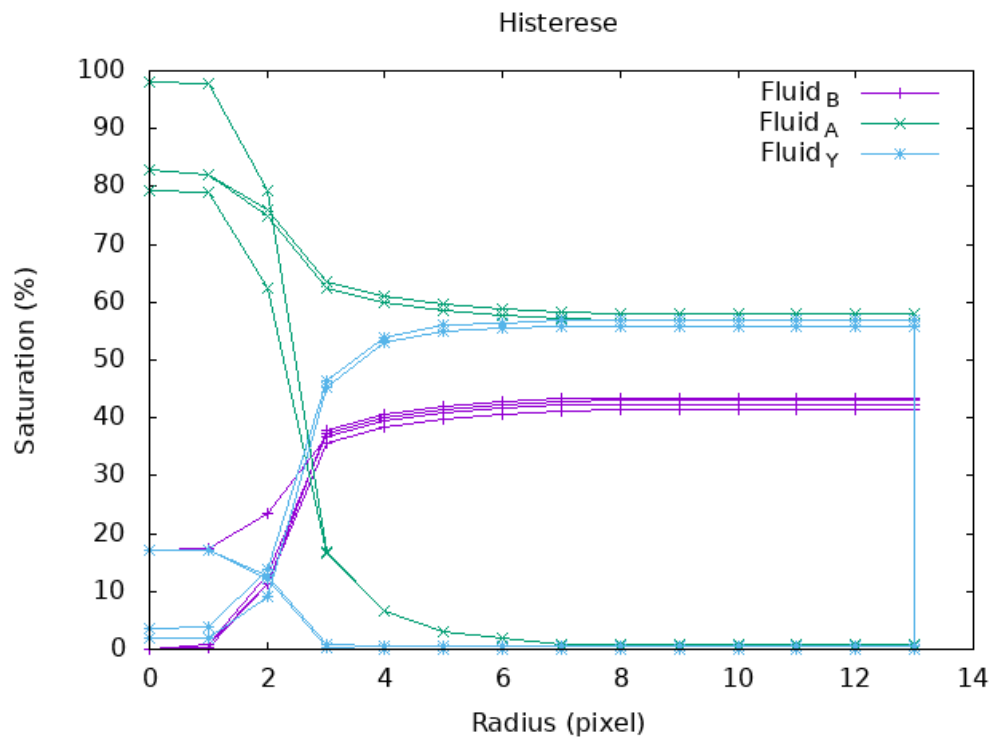
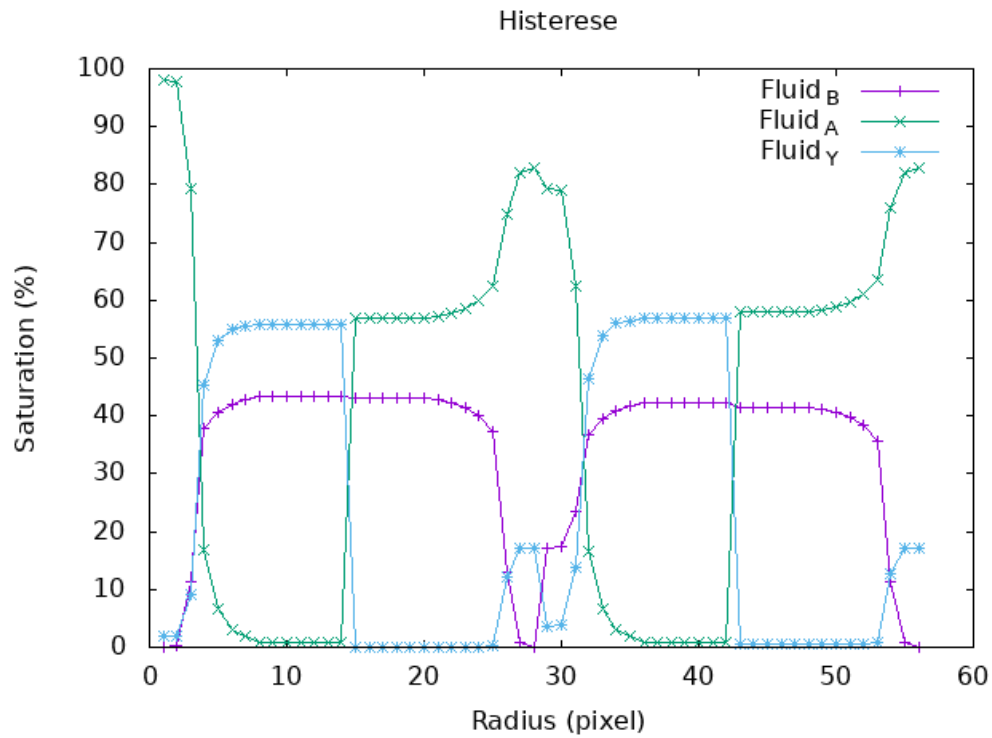
Listing 1.3: Exemplo de arquivo script do gnuplot

```

70#!gnuplot
71# Linhas de comentarios comecam com #
72# Para gerar o grafico
73# gnuplot histereseC-png.gnuplot
74# load 'histereseC-png.gnuplot'
75
76set title "Histerese"
77set xlabel "Radius (pixel)"
78set ylabel "Saturation (%)"
79
80set size 1,1
81# set xrange [0:56]
82# set yrange [0:100]
83set label left
84
85# Define tipo arquivo saída
86# set terminal x11
87# set terminal qt
88# set terminal postscript landscape eps colour
89set terminal png
90
91# Define nome arquivo
92!rm histereseC.png
93
94# Plota o gráfico no arquivo histereseC.png
95# Função do raio
96set out "histereseC-raio.png"
97plot "TISCImbibition.dat" using 2:3 title 'Fluid_B' with linespoints , \
98      "TISCImbibition.dat" using 2:4 title 'Fluid_A' with linespoints , \
99      "TISCImbibition.dat" using 2:5 title 'Fluid_Y' with linespoints
100# Mostra o gráfico usando programa display
101!display histereseC-raio.png &
102
103# Função do passo
104set out "histereseC-passo.png"
105plot "TISCImbibition.dat" using 1:3 title 'Fluid_B' with linespoints , \
106      "TISCImbibition.dat" using 1:4 title 'Fluid_A' with linespoints , \
107      "TISCImbibition.dat" using 1:5 title 'Fluid_Y' with linespoints
108# Mostra o gráfico usando programa display
109!display histereseC-passo.png &
110
111
112# Repete a plotagem, agora numa janela
113set size 1,1
114set terminal qt
115replot

```

Veja a seguir o gráfico gerado.



- Você encontra no site <http://gnuplot.sourceforge.net/demo/> diversos exemplos de scripts. a dica é acessar o mesmo e ver qual script atende seu interesse.
- Se instalar o pacote gnuplot-doc terá acesso a diversos exemplos.

Exemplo:

```
dnf install gnuplot-doc
cd /usr/share/doc/gnuplot-doc/demo
gnuplot all.dem
```

## 1.8 Gráficos 3D

### 1.8.1 Como plotar gráficos 3D - superficies

- O comando `splot` é o comando básico para gerar gráficos 3D.

#### Syntaxe:

```
splot {<intervalo>} <função> | "<arquivoDeDados>" {modificadores-arquivo}}
{title "título"} {with <estilos>} {, {definitions,} <função> ...}
```

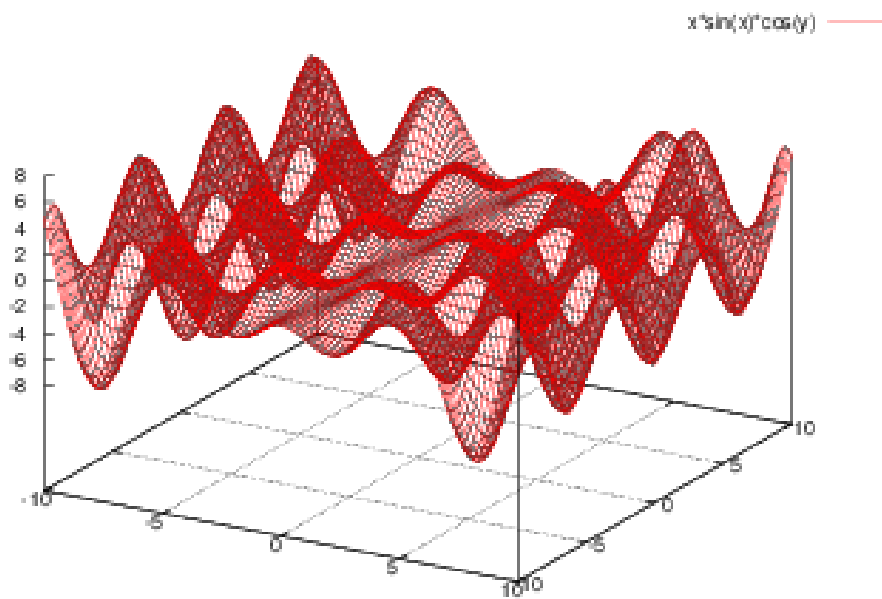
Exemplo:

# Para plotar um gráfico em 3 Dimensões

```
set grid
splot x*sin(x)*cos(y)
```

# Para melhorar a resolução

```
set isosamples 100,100
replot
```



### 1.8.2 Como fazer gráficos 3D avançados

- Veja a seguir alguns comandos avançados.

Exemplo:

```
splot x**2+y**2
```

- Para ocultar as linhas dos eixos

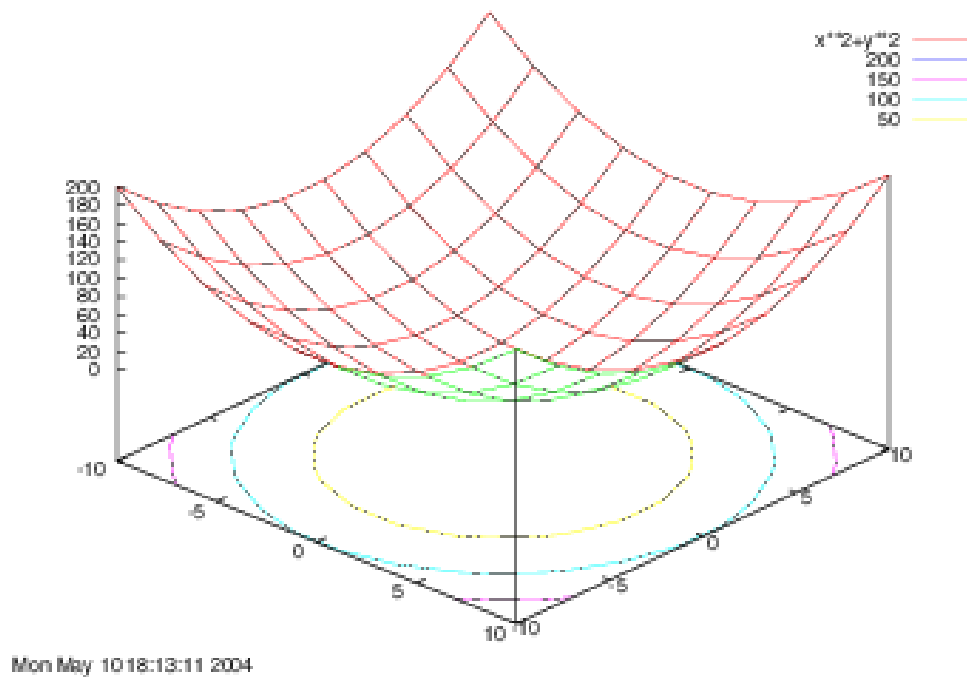
```
set hidden3d
unset hidden3d
```

- Para adicionar isolinhas nas superfícies

```
set contour surface
replot
```

- Para adicionar isolinhas na base

```
set contour base
set surface
show contour
replot
```



### 1.8.3 Como plotar gráficos 3D a partir de dados de um arquivo de disco

- Em muitos casos temos uma função do tipo  $z = f(x, y)$ , em que, para cada par ordenado  $x, y$  temos um valor de  $z$ . Isto gera uma superfície em 3 dimensões (como a ilustrada na figura acima).
- Veja a seguir os dados armazenados no arquivo "listagens/dadosxy.dat"

```
0  1  4  9
1  2  5 10
4  5  8 13
9 10 13 18
16 17 20 25
25 26 29 34
```

- Para plotar usando o `gnuplot` use o comando abaixo:

Exemplo:

```
$ cd listagens && gnuplot
splot "dadosxy.dat" matrix with lines
set xtics ("100" 0, "200" 1, "300" 2)
replot
```

- Referências:

- <http://lowrank.net/gnuplot/plot3d2-e.html>
- <http://lowrank.net/gnuplot/datafile-e.html>

### 1.8.4 Como plotar contornos

- Veja exemplo de como plotar contornos em: completar.....

### 1.8.5 Como plotar equações não paramétricas em 3D

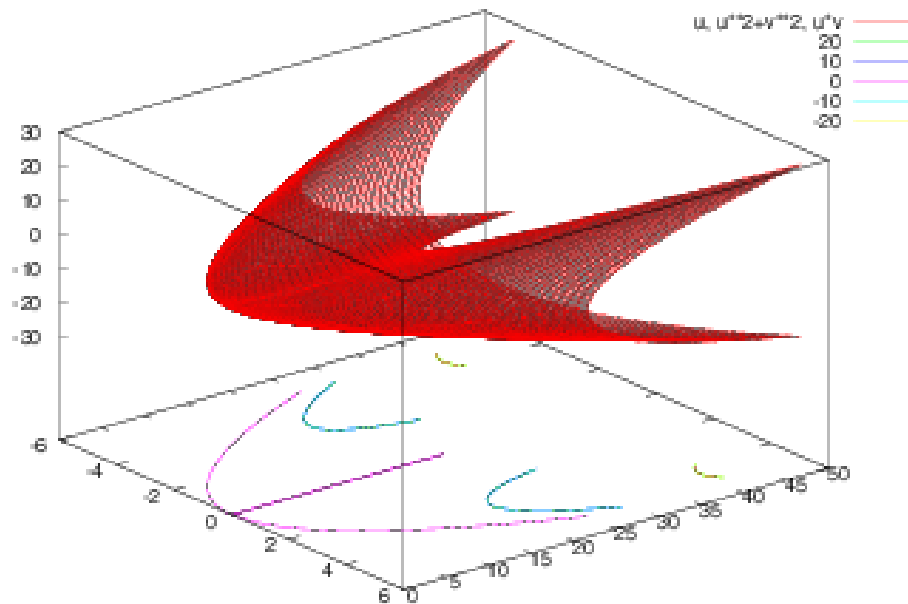
- Veja exemplo de como plotar superfícies não paramétricas em: completar.....

### 1.8.6 Como plotar equações paramétricas em 3D

- Veja exemplo de como plotar superfícies paramétricas em: completar.....

Exemplo

```
splot u, u**2+v**2, u*v
set grid
splot u, u**2+v**2, u*v
set isosamples 30,30
set parametric
splot u, u**2+v**2, u*v
```



Mon May 10 16:01:06 2004

## 1.9 Exemplos de demonstração (demos)

- Verifique se foram instalados os tutoriais e demos do gnuplot (diretório `/usr/share/doc/gnuplot-doc/demo` no Fedora). Vá para o diretório de demos, abra o gnuplot e então execute todos os demos usando a sequência abaixo<sup>2</sup>.

```
$cd /usr/share/doc/gnuplot-doc/demo
gnuplot> load "All.dem"
```

- No diretório que repassei tem uma cópia

```
$cd demos-do-gnuplot
$gnuplot
gnuplot> load "All.dem"
```

- Gnuplot 5.0 acesse o link abaixo:

– [http://gnuplot.sourceforge.net/demo\\_5.0/](http://gnuplot.sourceforge.net/demo_5.0/)

- Procure na pasta distribuída para você pelo diretório "demos-do-gnuplot".

## 1.10 Usando bibliotecas computacionais - A classe CGnuplot

- Primeiro vimos que posso usar o gnuplot para fazer gráficos simples de forma prática e direta.

```
plot sin(x)
```

<sup>2</sup>Você pode acessar os resultados diretamente através do site <http://gnuplot.sourceforge.net/demo/>.

- A seguir vimos que posso incrementar meu gráfico colocando propriedades como título do gráfico, dos eixos, mudar título da função, espessura e cor da linha, tipo de linha e dezenas de outras propriedades das curvas e do gráfico.

```
plot 5 + x**2 title "Função Parábola'" with lines, tan(x) title "Função Tangente"
plot cos(x) w l lt 1 lw 5 t "This is line type 1, thickness 5"
```

- Vimos que posso plotar arquivos externos, arquivos de dados. Note que os dados ou funções plotadas eram definidas dentro do terminal do `gnuplot`.

```
plot "Fluid-A.dat" using 1:3 title "C#REF-P#L-F#A-S#NR" with linespoints lt 2 lc 2, \
"Fluid-A.dat" using 1:5 title "C#REF-P#W-F#A-S#NR" with linespoints lt 3 lc 4.
```

- Finalmente vimos que posso fazer contas com as colunas do arquivo

```
plot 'arq.dat' using (3*$2):(sin($3+$1))
```

- Num outro momento aprendemos a criar scripts, um arquivo de texto que tem o passo a passo para criar meus gráficos. Vou usar scripts sempre que tiver de plotar muitos gráficos.

```
gnuplot
load "script"
```

- Vimos que o diretório "*demo*" tem vários scripts prontos, que posso ver ali qual tem a sequência que preciso e mudar a mesma para fazer o gráfico que quero.
- Mas note que os comandos eram digitados num editor de texto e interpretados pelo `gnuplot`. O programa `gnuplot` lê a linha do script e a executa, uma a uma.
- Em resumo, rodo um simulador que gera, no formato ASCII, o arquivo de dados/resultados; Depois, crio um script que o `gnuplot` vai executar para gerar o gráfico.

```
./Simulador dados1.dat
./Simulador dados2.dat
...
./Simulador dadosn.dat
gnuplot
load script1
load script2
```

- Note que existe uma evolução, a cada etapa estou usando novos recursos de forma a facilitar o meu dia a dia, minhas tarefas de rotina.
- Veremos agora que existem biblioteca computacionais escritas em C/Fortran/C++ que permitem que eu envie comandos do meu programa diretamente para o `gnuplot`. Ou seja, eu não necessariamente preciso criar um arquivo de dados, os dados(vetores) que estão na memória do programa que estou executando são enviados diretamente para o `gnuplot`. Também posso fazer tudo o que faria com o `gnuplot`, enviando comandos do meu programa para o `gnuplot`. Meu programa, mesmo sendo em modo terminal, vai fazer gráficos.

### 1.10.1 Exemplo de código em C++ que usa o gnuplot

- Na imagem a seguir a classe `Gnuplot` contém métodos/funções que podem ser chamadas pelo meu programa. A função

```
Gnuplot& Title (const std::string& title = "");
```

- retorna um gráfico, se chama `Title`, e recebe como parâmetro uma *string* com o título do gráfico.



```

class Gnuplot
{public:
    Gnuplot & Title (const std::string & title = "");
    Gnuplot & YLabel (const std::string & label = "y");
    Gnuplot & XLabel (const std::string & label = "x");
    Gnuplot & XRange (const int iFrom, const int iTo);
    Gnuplot & YRange (const int iFrom, const int iTo);
    Gnuplot & PlotFile (const std::string & filename,
        const int column = 1, const std::string & t = "");
    Gnuplot & PlotVector (const std::vector<double>&x,
        const std::string & title = "");
    Gnuplot & PlotVector (const std::vector<double>&x,
        const std::vector<double>&y,
        const std::string & title = "");
    Gnuplot & PlotSlope (const double a, const double b,
        const std::string & title = "");
    Gnuplot & PlotEquation (const std::string & eq,
        const std::string & title = "");
    Gnuplot & Replot ();
};

```

#### Gnuplot

```

+ Terminal(type : const std::string&)
+ Gnuplot(style : const std::string&)
+ Gnuplot(x : const std::vector< double >&, title : const std::string&, style : const std::string&, labelx : const std::string&, labely : const std::string&)
+ Cmd(cmdstr : const std::string&) : Gnuplot&
+ operator <<(cmdstr : const std::string&) : Gnuplot&
+ SaveTops(filename : const std::string&) : Gnuplot&
+ Style(stylestr : const std::string&) : Gnuplot&
+ Surface(_surface : int) : Gnuplot&
+ Legend(position : const std::string&) : Gnuplot&
+ Title(title : const std::string&) : Gnuplot&
+ XLabel(label : const std::string&) : Gnuplot&
+ XRange(iFrom : const int, iTo : const int) : Gnuplot&
+ XAutoscale() : Gnuplot&
+ YAutoscale() : Gnuplot&
+ XLogscale(base : const double) : Gnuplot&
+ PlotFile(filename : const std::string&, column : const int, title : const std::string&) : Gnuplot&
+ PlotVector(x : const std::vector< double >&, title : const std::string&) : Gnuplot&
+ PlotFile(filename : const std::string&, column_x : const int, column_y : const int, title : const std::string&) : Gnuplot&
+ PlotVector(x : const std::vector< double >&, y : const std::vector< double >&, title : const std::string&) : Gnuplot&
+ PlotFile(filename : const std::string&, column_x : const int, column_y : const int, column_z : const int, title : const std::string&) : Gnuplot&
+ PlotVector(x : const std::vector< double >&, y : const std::vector< double >&, z : const std::vector< double >&, title : const std::string&) : Gnuplot&
+ PlotSlope(a : const double, b : const double, title : const std::string&) : Gnuplot&
+ PlotEquation(equation : const std::string&, title : const std::string&) : Gnuplot&
+ PlotEquation3d(equation : const std::string&, title : const std::string&) : Gnuplot&
+ Replot() : Gnuplot&
+ ResetPlot() : Gnuplot&

```

Veja na listagem 1.5 exemplo de programa em C++ que usa a classe CGnuplot. Não se preocupe com a linguagem em si, apenas note como as instruções são passadas para o `gnuplot` de forma bastante direta.

Listing 1.4: Exemplo de código pequeno em C++ que usa a classe CGnuplot

```

117 // Programa de teste da classe CGnuplot.
118 #include <iostream>
119 #include "CGnuplot.h"
120
121 using namespace std;          // Usando espaco de nomes da std
122
123 void wait_for_key ();
124
125 int main(int argc, char* argv[]) {
126
127     cout << "\n=====
128         << "\n===== Programa de teste da LIB_LDSC =====
129         << "\n=====
130         << "\n=====
131         << "\nUSO: "
132         << "\n./cggnuplot.teste.min"
133         << "\n===== \n";
134
135     Gnuplot::Terminal("qt");    // Tipo de terminal gráfico
136
137     // ----- Graficos 2D -----

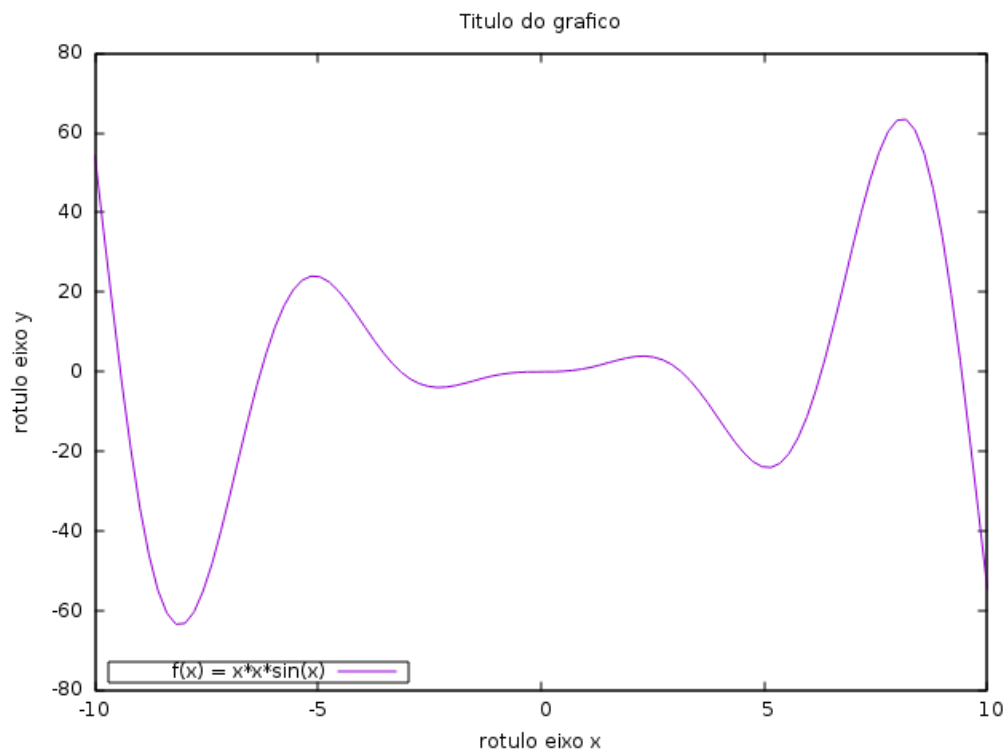
```

```

138 Gnuplot g2d ("lines"); // Construtor
139 g2d.Legend("inside").Legend("left").Legend("bottom").Legend("box");
140 g2d.Title("Titulo do grafico"); // Titulo do grafico
141 g2d.XLabel("rotulo_eixo_x"); // Rotulo eixo x
142 g2d.YLabel("rotulo_eixo_y"); // Rotulo eixo y
143 g2d.XRange(-10,10); // Seta intervalo do eixo x.
144 g2d.PlotEquation( "x*x*sin(x)");// Plota uma determinada equacao
145 wait_for_key();
146 // Usando os diferentes estilos de graficos
147
148 // Muda o estilo da funcao para linhas e replota
149 cout << "g2d.Style(\"points\")" << endl;
150 g2d.Style("points"); // Muda estilo linha
151 g2d.PlotEquation( "x*x*x*sin(x)");// Plota uma determinada equacao
152 wait_for_key();
153
154 // Muda o estilo da funcao para impulsos, muda titulo e plota nova equacao
155 cout << "Style(\"impulses\")" << endl;
156 g2d.Style("impulses").Title("Style(impulses)").PlotEquation( "x*x+5");
157 wait_for_key();
158
159 // Vou mudar o terminal, que é o dispositivo de saída
160 g2d << "set term png\n"; // seta o terminal para imagem no formato png
161 g2d << "set output \"imagemGrafico.png\""; // seta o nome do arquivo
162 g2d.replot();// plota o grafico no arquivo png
163 wait_for_key();
164
165 g2d.Reset(); // Reseta estado do grafico
166 cout << "\n***Fim do exemplo\n";
167 return 0;
168 }
169
170 void wait_for_key ()
171 {
172     cout << endl << "Pressione ENTER para continuar..." << endl;
173     std::cin.clear(); // Zera estado de cin
174     std::cin.ignore(std::cin.rdbuf()->in_avail()); // Ignora
175     std::cin.get(); // Espera o pressionamento do enter
176     return;
177 }

```

Veja a seguir o gráfico gerado.



Listing 1.5: Exemplo de código em C++ que usa várias funções da classe CGnuplot

```

178 //////////////////////////////////////
179 //                               Programa de teste da classe CGnuplot. //
180 //////////////////////////////////////

```

```

181// Esta interface usa pipes e nao ira funcionar em sistemas que nao suportam
182// o padrao POSIX pipe.
183//
184// O mesmo foi testado em sistemas Windows (MinGW e Visual C++) e GNU/Linux(GCC/G++)
185//
186// Este programa foi originalmente escrito por:
187// Historico de versoes:
188// 0. Interface para linguagem C
189//   por N. Devillard (27/01/03)
190// 1. Interface para C++: tradução direta da versao em C
191//   por Rajarshi Guha (07/03/03)
192// 2. Correcoes para compatibilidadde com Win32
193//   por V. Chyzhdenka (20/05/03)
194// 3. Novos métodos membros, correcoes para compatibilidade com Win32 e Linux
195//   por M. Burgis (10/03/08)
196// 4. Tradução para Portugues, documentacao e modificacoes na interface
197//   por Bueno.A.D. (30/07/08)
198//
199//
200//
201// Requisitos:
202// - O programa gnuplot deve estar instalado (veja http://www.gnuplot.info/download.html)
203// - No Windows: setar a Path do Gnuplot (i.e. C:/program files/gnuplot/bin)
204//   ou setar a path usando: Gnuplot::set_GNUPlotPath(const std::string &path);
205//   antes de criar qualquer objeto da classe.
206//
207//
208
209// ----- Inclusão de arquivos -----
210#include <iostream>
211
212/// A classe CGnuplot usa pipes no estilo POSIX para se comunicar com o gnuplot.
213/// POSIX-Pipe-communication.
214#include "CGnuplot.h"
215
216// Se estamos no windows
217#if defined(WIN32) || defined(_WIN32) || defined(__WIN32__) || defined(__TOS_WIN__)
218#include <conio.h> // Para acesso a getch(), necessario em wait_for_key()
219#include <windows.h> // Para acesso a Sleep()
220void sleep(int i) { Sleep(i*1000); }
221#endif
222
223// ----- Variaveis globais -----
224const int SLEEP_LGTH = 2; // Tempo de espera em segundos
225const int NPOINTS = 50; // Dimensao do array (vetor)
226#define SLEEP_LGTH 2
227#define NPOINTS 50
228
229using namespace std; // Usando espaco de nomes da std
230
231// ----- Funcoes globais -----
232/// @brief O programa para ate o pressionamento de uma tecla.
233void wait_for_key();
234
235// ----- Funcao Principal -----
236/// @brief Funcao principal
237int main(int argc, char* argv[])
238{
239    cout << "\n=====
240    << "\n===== _Programa_de_teste_da_LIB_LDSC_ =====
241    << "\n=====
242    << "\n=====
243    << "\nUSO:"
244    << "\n./cggnuplot.min"
245    << "\n===== \n";
246
247    // Se a variavel do gnuplot nao esta setada, faca isto antes de
248    // criar objetos da classe CGnuplot, usando o método publico e estatico:
249    // Gnuplot::set_GNUPlotPath("C:/program files/gnuplot/bin/");
250
251    // Seta o terminal padrao para visualizacao dos graficos (normalmente nao necessario),
252    // Usuarios de Mac devem usar a opcao "aqua", e nao x11.
253    // Gnuplot::set_terminal_std("x11");
254    cout << "-----\n"
255    << "----> _Plotando_graficos_do_gnuplot_usando_a_classe_CGnuplot_ ----\n"
256    << "----> _Exemplo_de_controle_do_gnuplot_usando_C++_ ----\n"
257    << "----> _Os_titulos_do_grafico_ilustram_a_operacao_realizada_ ----\n"
258    << "-----\n" << endl;
259

```

```

260 // Controla a ocorrencia de excessoes, usa a classe GnuplotException
261 try
262 {
263
264     // Teste geral
265     // Terminal padrao do gnuplot no fedora 9
266     // Se nao funcionar em seu sistema, comente a linha
267     Gnuplot::Terminal("qt");//Gnuplot::Terminal("wxt");
268
269     // ----- Graficos 2D -----
270     Gnuplot g2d = Gnuplot("lines");// Construtor
271     g2d.PointSize(0.8);           // Escala o tamanho do ponto usado na plotagem
272                                 // Legenda
273     g2d.Legend("inside").Legend("left").Legend("bottom").Legend("box");
274     g2d.Title("Titulo_do_grafico");// Titulo do grafico
275     g2d.XLabel("rotulo_eixo_x");  // Rotulo eixo x
276     g2d.YLabel("rotulo_eixo_y");  // Rotulo eixo y
277     g2d.XRange(-10,10);           // Seta intervalo do eixo x.
278     g2d.PlotEquation( "x*x*sin(x)");// Plota uma determinada equacao
279
280                                 // Usando os diferentes estilos de graficos
281     cout << "Style(\"lines\")" << endl;
282     g2d.Style("Style(_lines_)").Replot();
283     wait_for_key();
284     g2d.Reset();                 // Reseta estado do grafico
285
286     cout << "Style(\"points\")" << endl;
287     g2d.Style("points").Title("Style(_points_)").PlotEquation( "x");
288     wait_for_key();
289     g2d.Reset();
290
291     cout << "Style(\"linespoints\")" << endl;
292     g2d.Style("linespoints").Title("Style(_linespoints_)").PlotEquation( "x*x");
293     wait_for_key();
294     g2d.Reset();
295
296     cout << "Style(\"impulses\")" << endl;
297     g2d.Style("impulses").Title("Style(_impulses_)").PlotEquation( "x*x+_5");
298     wait_for_key();
299     g2d.Reset();
300
301     cout << "Style(\"dots\")" << endl;
302     g2d.Style("dots").Title("Style(_dots_)").PlotEquation( "x*x*x");
303     wait_for_key();
304     g2d.Reset();
305
306     cout << "Style(steps)" << endl;
307     g2d.Style("steps").Title("Style(_steps_)").PlotEquation( "x*x*x*x");
308     wait_for_key();
309     g2d.Reset();
310
311     cout << "Style(\"fsteps\")" << endl;
312     g2d.Style("fsteps").Title("Style(_fsteps_)").PlotEquation( "x*x*sin(x)");
313     wait_for_key();
314     g2d.Reset();
315
316     cout << "Style(\"histeps\")" << endl;
317     g2d.Style("histeps").Title("Style(_histeps_)").PlotEquation( "x*x*sin(x)");
318     wait_for_key();
319
320                                 // Legendas, posicoes possiveis
321     cout << "Legend(\"inside_left_top_nobox\")" << endl;
322     g2d.Legend("inside_left_top_nobox").Title("Legend(_inside_left_top_nobox_)").Replot();
323     wait_for_key();
324
325     cout << "Legend(\"inside_center_center_nobox\")" << endl;
326     g2d.Legend("inside_center_center_nobox").Title("Legend(_inside_center_center_nobox_)").Replot();
327     wait_for_key();
328
329     cout << "Legend(\"inside_right_bottom_box\")" << endl;
330     g2d.Legend("inside_right_bottom_box").Title("Legend(_inside_right_bottom_box_)").Replot();
331     wait_for_key();
332
333     cout << "Legend(\"outside_right_top_box\")" << endl;
334     g2d.Legend("outside_right_top_box").Title("Legend(_outside_right_top_box_)").Replot();
335     wait_for_key();
336
337     // ----- Graficos 3D -----

```

```

338 Gnuplot g3d = Gnuplot("lines"); // Construtor
339 g3d.Grid(1); // Ativa/Desativa o grid
340 g3d.Samples(50); // Seta taxa de amostragem
341 g3d.IsoSamples(50); // Seta densidade de isolinhas
342 g3d.Hidden3d(); // Ativa/Desativa remocao de linhas ocultas
343 g3d.Surface(); // Ativa/Desativa a visualizacao da superficie
344 g3d.Title("Titulo_do_grafico"); // Titulo do grafico
345 g3d.XLabel("rotulo_eixo_x"); // Rotulo eixo x
346 g3d.YLabel("rotulo_eixo_y"); // Rotulo eixo y
347 g3d.ZLabel("rotulo_eixo_z"); // Rotulo eixo z
348 g3d.XRange(-10,10); // Seta intervalo do eixo x.
349 g3d.ZAutoscale(); // Seta autoescala de z
350 g3d.PlotEquation3d( "x*sin(x)*sin(y)+4" );
351 wait_for_key();
352
353 // Suavizacao
354 cout << "Smooth(0)" << endl;
355 g3d.Smooth(0).Title("Smooth(0)").Replot(); // Desativa suavizacao
356 wait_for_key();
357 cout << "Smooth(1)" << endl;
358 g3d.Smooth(1).Title("Smooth(1)").Replot(); // Ativa suavizacao
359 wait_for_key();
360
361 // Ativar/desativar grid
362 cout << "Grid()" << endl;
363 g3d.Grid().Title("Grid()").Replot();
364 wait_for_key();
365 cout << "Grid(0)" << endl;
366 g3d.Grid(0).Title("Grid(0)").Replot();
367 wait_for_key();
368
369 // Ocultar linhas escondidas
370 cout << "Hidden3d(0)" << endl;
371 g3d.Hidden3d(0).Title("Hidden3d(0)").Replot();
372 wait_for_key();
373 cout << "Hidden3d()" << endl;
374 g3d.Hidden3d().Title("Hidden3d()").Replot();
375 wait_for_key();
376
377 // Taxa amostragem
378 cout << "Samples(10)" << endl;
379 g3d.Samples(10).Title("Samples(10)").Replot();
380 wait_for_key();
381 cout << "Samples(50)" << endl;
382 g3d.Samples(50).Title("Samples(50)").Replot();
383 wait_for_key();
384
385 // Densidade de isolinhas
386 cout << "IsoSamples(10)" << endl;
387 g3d.IsoSamples(10).Title("IsoSamples(10)").Replot();
388 wait_for_key();
389 cout << "IsoSamples(50)" << endl;
390 g3d.IsoSamples(50).Title("IsoSamples(50)").Replot();
391 wait_for_key();
392
393 // Contorno em superficies, base, surface, both.
394 cout << "Contour(\"base\")" << endl;
395 g3d.Contour("base").Title("Contour(base)").Replot();
396 wait_for_key();
397 cout << "Contour(\"surface\")" << endl;
398 g3d.Contour("surface").Title("Contour(surface)").Replot();
399 wait_for_key();
400 cout << "Contour(\"both\")" << endl;
401 g3d.Contour("both").Title("Contour(both)").Replot();
402 wait_for_key();
403
404
405 // ----- A seguir exemplos do codigo original -----
406 Gnuplot g1 = Gnuplot("lines");
407 cout << "***_Plota_uma_equacao_da_forma_y=ax+b;_com_a=1,_b=0" << endl;
408 g1.Title("PlotSlope_y=x"); // Seta o titulo.
409 g1.PlotSlope(1.0,0.0,"PlotSlope_y=x"); // Plota Reta
410 wait_for_key();
411
412 cout << "***_Plota_uma_equacao_da_forma_y=ax+b;_com_a=2,_b=0" << endl;
413 cout << "PlotSlope_y=2*x" << endl;
414 g1.PlotSlope(2.0,0.0,"y=2x");
415 wait_for_key();
416

```

```

417     cout << "***_Plota_uma_equacao_da_forma_y=ax+b;_com_a=-1,_b=0" << endl;
418     cout << "PlotSlope_y=-x" << endl;
419     g1.PlotSlope(-1.0,0.0,"y=-x");
420     wait_for_key();
421     g1.Title();
422
423                                     // Equacoes
424     g1.ResetPlot();                                     // Reseta o grafico
425     cout << endl << endl << "***_Plotando_Equacoes" << endl;
426
427     cout << "***_PlotEquation_y=sin(x)" << endl;
428     g1.PlotEquation("sin(x)", "PlotEquation_sine, sin(x)"); // Plota uma equacao
429     wait_for_key();
430
431     cout << "***_y=log(x)" << endl;
432     g1.Legend("box").Legend("left").PlotEquation("log(x)", "PlotEquation_logarithm, log(x)"); //
433     // Plota uma equacao
434     wait_for_key();
435
436     cout << "***_y=sin(x)*cos(2*x)" << endl;
437     g1.PlotEquation("sin(x)*cos(2*x)", "PlotEquation, sin(x)*cos(2*x)"); // Plota uma equacao
438     wait_for_key();
439                                     // Controlando estilos de graficos - styles
440     g1.ResetPlot();
441     cout << endl << endl << "***_Mostrando_estilos_-_styles" << endl;
442
443     cout << "***_sin(x)_usando_PointSize(0.8).Style(\"points\")" << endl;
444     g1.PointSize(0.8).Style("points");
445     g1.PlotEquation("sin(x)", "PlotEquation_sin(x), usando_points");
446     wait_for_key();
447
448     cout << "***_sine_usando_estilo_de_impulses" << endl;
449     g1.Style("impulses");
450     g1.PlotEquation("sin(x)", "PlotEquation_sin(x), usando_impulses");
451     wait_for_key();
452
453     cout << "***_sine_usando_estilo_de_steps" << endl;
454     g1.Style("steps");
455     g1.PlotEquation("sin(x)", "PlotEquation_sin(x), usando_steps");
456     wait_for_key();
457
458                                     // Salvando para arquivo postscript - ps
459     g1.ResetAll();                                     // Reseta todos os dados
460     cout << endl << endl << "***_Salvando_para_arquivo_postscript_-_ps" << endl;
461
462     cout << "y=sin(x)_salvo_no_arquivo_test_output.ps_no_diretorio_corrente" << endl;
463     g1.SaveTops("test_output");
464     g1.Style("lines").Samples(300).XRange(0,5);
465     g1.PlotEquation("sin(12*x)*exp(-x)").PlotEquation("exp(-x)");
466
467     cout << "***_Plotando_novamente_em_uma_janela" << endl;
468     g1.ShowOnScreen(); // Ativa janela de saida grafica
469
470                                     // Usando vetores do usuario (conjunto de dados)
471     cout << "***_Criando_vetores_x,y,y2,dy,_z_a_serem_plotados" << endl;
472     std::vector<double> x, y, y2, dy, z_a_serem_plotados;
473
474     for (int i = 0; i < NPOINTS; i++) // Preenche os vetores x, y, z
475     {
476         x.push_back((double)i); // x[i] = i
477         y.push_back((double)i * (double)i); // y[i] = i^2
478         z.push_back( x[i]*y[i] ); // z[i] = x[i]*y[i] = i^3
479         dy.push_back((double)i * (double)i / (double) 10); // dy[i] = i^2 / 10
480     }
481     y2.push_back(0.00); y2.push_back(0.78); y2.push_back(0.97); y2.push_back(0.43);
482     y2.push_back(-0.44); y2.push_back(-0.98); y2.push_back(-0.77); y2.push_back(0.02);
483
484     g1.ResetAll();
485     cout << endl << endl << "***_Plota_vetor_y_de_doubles" << endl;
486     g1.Style("impulses").PlotVector(y, "PlotVector_y, usando_impulses");
487     wait_for_key();
488
489     g1.ResetPlot();
490     cout << endl << endl << "***_Plota_vetores_x_e_y, pares_ordenados_(x,y)" << endl;
491     g1.Grid();
492     g1.Style("points").PlotVector(x,y, "PlotVector_x_e_y, pares_ordenados_(x,y), usando_points");
493     wait_for_key();
494

```



```

495     g1.ResetPlot();
496     cout << endl << endl << "***Plota vetores x,y,z, valores ordenados(x,y,z)" << endl;
497     g1.Grid(0);
498     g1.PlotVector(x,y,z,"PlotVector(x,y,z), usando points");
499     wait_for_key();
500
501     g1.ResetPlot();
502     cout << endl << endl << "***Plota vetores x,y,z, valores ordenados barra de erro(x,y,
503         dy)" << endl;
504     g1.PlotVectorXYErrorBar(x,y,dy,"PlotVectorXYErrorBar valores ordenados x,y,z barra de erro(x
505         ,y,dy)");
506     wait_for_key();
507
508     // Usando multiplas janelas de saida
509     cout << endl << endl;
510     cout << "***multiple output windows" << endl;
511
512     g1.ResetPlot();
513     g1.Style("lines");
514     cout << "window 1: sin(x)" << endl;
515     g1.Grid(1).Samples(600).XRange(0,300);
516     g1.PlotEquation("sin(x)+sin(x*1.1)", "PlotEquation Grid(1).Samples(600).XRange(0,300)");
517     wait_for_key();
518
519     g1.XAutoscale().Title("XAutoscale()").replot();
520     wait_for_key();
521
522     Gnuplot g2;
523     cout << "Janela 2: plotando vetores" << endl;
524     g2.PlotVector(y2,"Pontos de y2");
525     g2.Smooth().PlotVector(y2,"Smooth(cspline)");
526     g2.Smooth("bezier").PlotVector(y2,"Smooth(bezier)");
527     g2.Smooth();
528     wait_for_key();
529
530     cout << "Janela 3: plotando equacoes, log(x)/x" << endl;
531     Gnuplot g3("lines");
532     g3.Grid(1);
533     g3.PlotEquation("log(x)/x", "log(x)/x");
534     wait_for_key();
535
536     cout << "Janela 4: splot x*x+y*y" << endl;
537     Gnuplot g4("lines");
538     g4.ZRange(0,100);
539     g4.XLabel("x-axis").YLabel("y-axis").ZLabel("z-axis");
540     g4.PlotEquation3d("x*x+y*y");
541     wait_for_key();
542
543     cout << "Janela 5: splot usando Hidden3d" << endl;
544     Gnuplot g5("lines");
545     g5.IsoSamples(25).Hidden3d();
546     g5.PlotEquation3d("x*y*y");
547     wait_for_key();
548
549     Gnuplot g6("lines");
550     cout << "Janela 6: splot usando Contour" << endl;
551     g6.IsoSamples(60).Contour();
552     g6.Surface().PlotEquation3d("sin(x)*sin(y)+4");
553     wait_for_key();
554
555     g6.Surface().Replot();
556     wait_for_key();
557
558     Gnuplot g7("lines");
559     cout << "Janela 7: usando Samples" << endl;
560     g7.XRange(-30,20).Samples(40);
561     g7.PlotEquation("besj0(x)*0.12e1").PlotEquation("(x**besj0(x))-2.5");
562     wait_for_key();
563
564     g7.Samples(400).Replot();
565     wait_for_key();
566
567     Gnuplot g8("filledcurves");
568     cout << "Janela 8: filledcurves" << endl;
569     g8.Legend("outside right top").XRange(-5,5);
570     g8.PlotEquation("x*x").PlotEquation("-x*x+4");
571
572     // Plota uma imagem
573     Gnuplot g9;

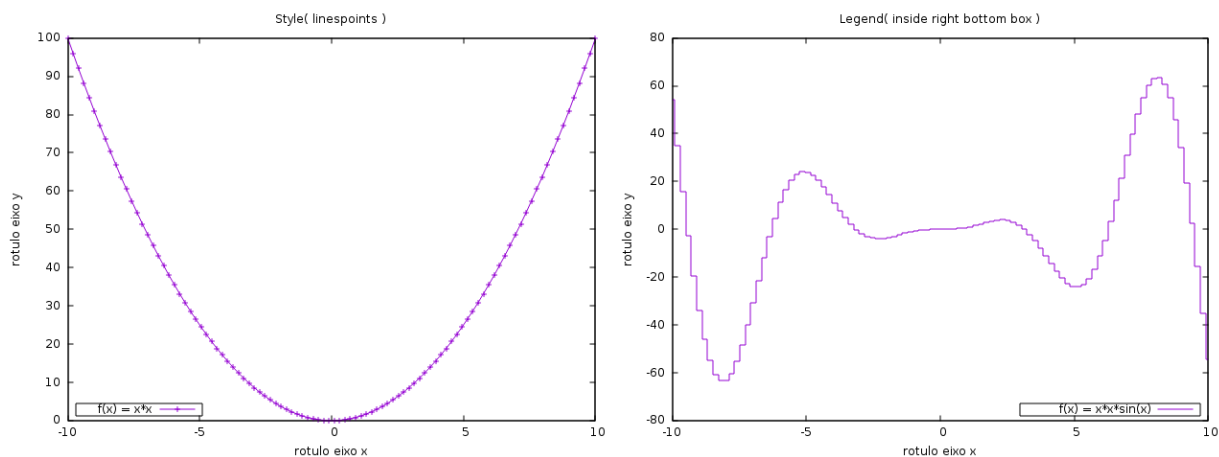
```

```

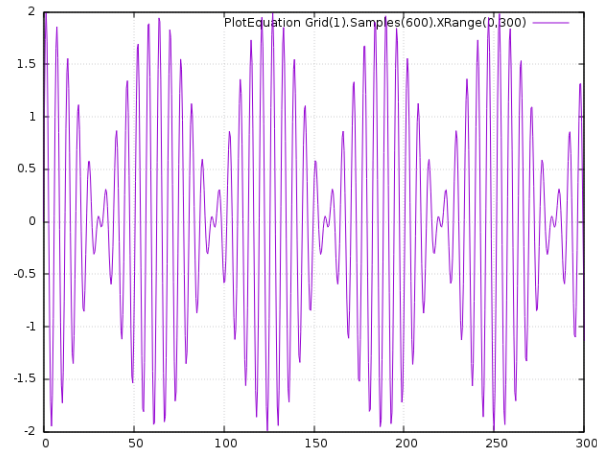
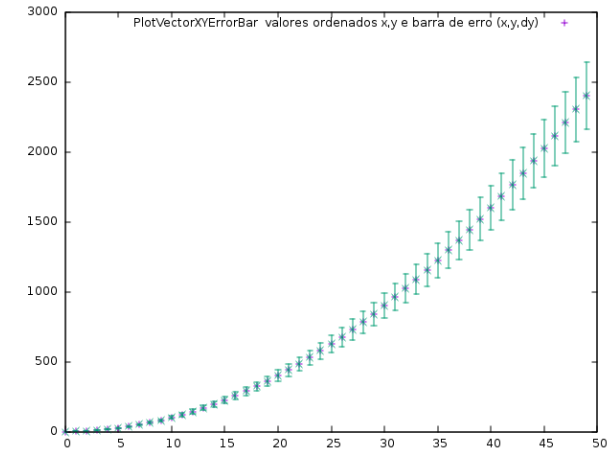
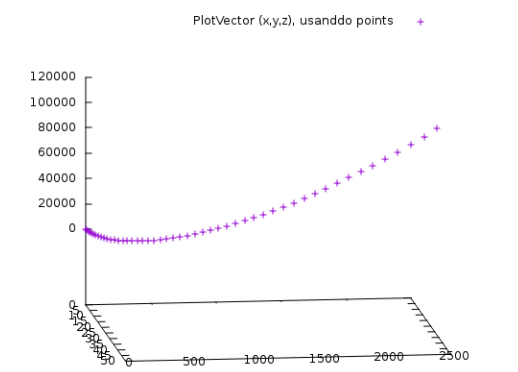
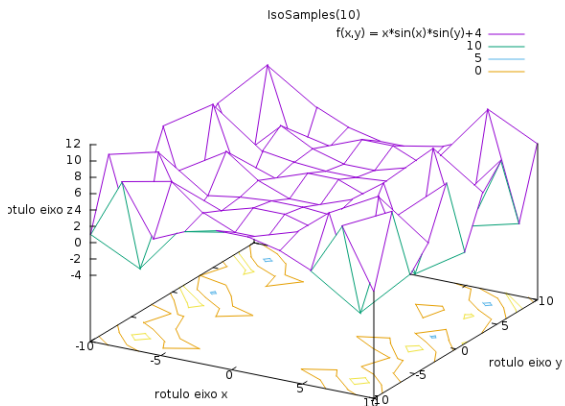
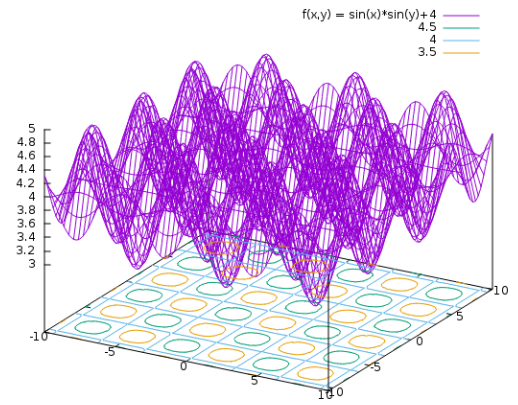
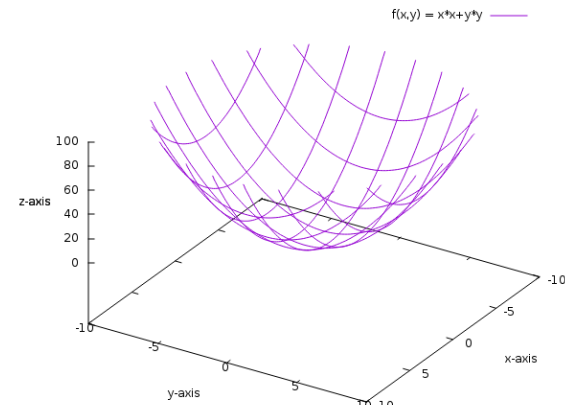
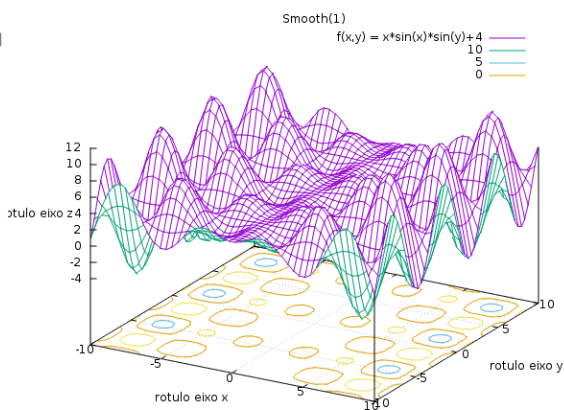
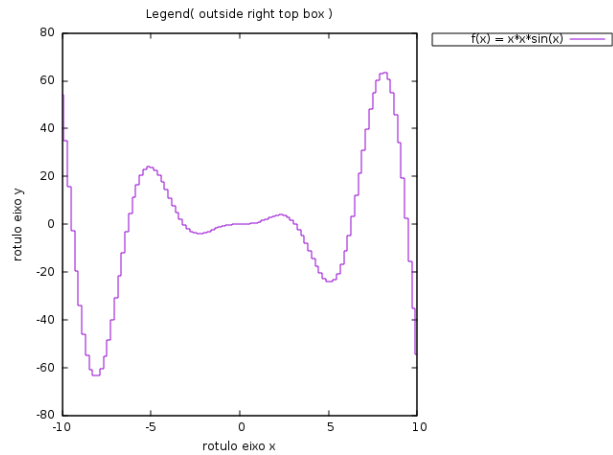
572     cout << "Janela_9: plot_image" << endl;
573     const int iWidth = 255;
574     const int iHeight = 255;
575     g9.XRange(0,iWidth).set_yrange(0,iHeight).CBRange(0,255);
576     g9.Command("set_palette_gray");
577     unsigned char ucPicBuf[iWidth*iHeight];
578                                     // Gera imagem em tons de cinza
579     for(int iIndex = 0; iIndex < iHeight*iWidth; iIndex++)
580     {
581         ucPicBuf[iIndex] = iIndex % 255;
582     }
583     g9.plot_image(ucPicBuf,iWidth,iHeight,"greyscale");
584     wait_for_key();
585
586     g9.PointSize(0.6).Legend(0).PlotSlope(0.8,20);
587     wait_for_key();
588
589                                     // Controle manual
590     Gnuplot g10;
591     cout << "Janela_10: controle_manual" << endl;
592     g10.Cmd("set_samples_400").Cmd("plot_abs(x)/2"); // Usando Cmd()
593     g10 << "replot_sqrt(x)" << "replot_sqrt(-x)"; // Usando operador <<
594     wait_for_key();
595
596 }
597 catch (GnuplotException ge)
598 {
599     cout << ge.what() << endl;
600 }
601
602 cout << endl << "***_Fim_do_exemplo_" << endl;
603
604 return 0;
605 }
606
607 void wait_for_key ()
608 {
609     // Todas as teclas serão consideradas, inclusive as setas
610     #if defined(WIN32) || defined(_WIN32) || defined(__WIN32__) || defined(__TOS_WIN__)
611         cout << endl << "Pressione qualquer tecla para continuar..." << endl;
612         FlushConsoleInputBuffer(GetStdHandle(STD_INPUT_HANDLE));
613         _getch();
614     #elif defined(unix) || defined(__unix) || defined(__unix__) || defined(__APPLE__)
615         cout << endl << "Pressione ENTER para continuar..." << endl;
616         std::cin.clear(); // Zera estado de cin
617         std::cin.ignore(std::cin.rdbuf()->in_avail()); // Ignora
618         std::cin.get(); // Espera o pressionamento do enter
619     #endif
620     return;
621 }

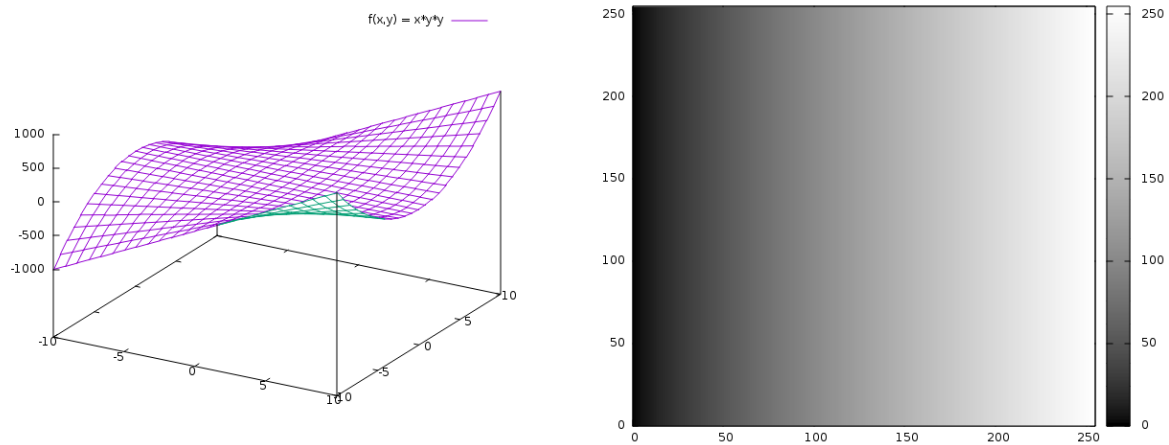
```

Veja a seguir alguns gráficos gerados.









### 1.10.2 Sentenças adicionais

- Um gráfico do gnuplot tem outros atributos, podendo-se citar: `arrow`, `border`, `clip`, `contour`, `grid`, `mapping`, `polar`, `surface`, `time`, `view`.
- Para obter uma lista completa dos comandos do gnuplot, consulte o manual do usuário, veja os exemplos apresentados no diretório `demo` ou ainda consulte o grupo de discussão `comp.graphics.apps.gnuplot`.
- O arquivo `~/.gnuplot`, é um arquivo oculto/escondido, que fica no seu diretório (`~`) e que contém configurações que são executadas automaticamente quando o gnuplot é inicializado. É possível modificar este arquivo para setar rapidamente as coisas que você usa com mais frequência.
- Para converter o formato das imagens use o programa `convert`

Exemplo:

```
convert imagemOriginal.pdf imagemFinal.jpg
```

```
convert -verbose -density 150 -trim imagemOriginal.pdf -quality 100 -flatten -sharpen 0x1.0 in
```

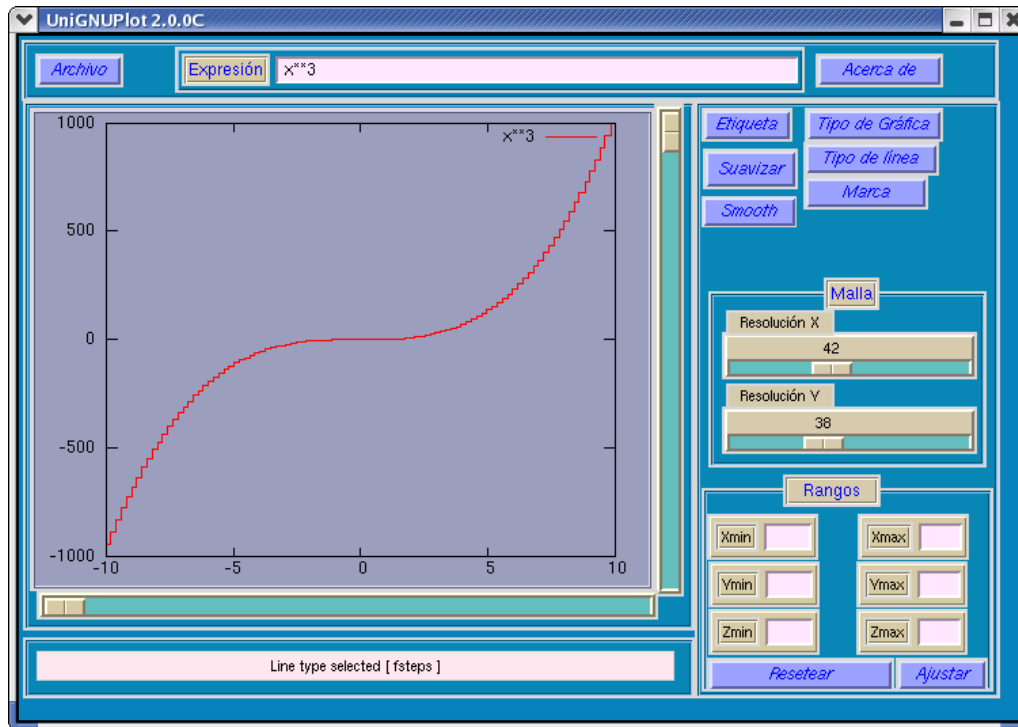
- É possível usar o programa `convert` para gerar vídeos a partir de um conjunto de gráficos. Por exemplo, um simulador gera dezenas de gráficos e quero gerar um vídeo mostrando a evolução dos gráficos. Para maiores detalhes veja o manual do programa `convert`.

Exemplo:

```
convert -delay 5 *.png -quality 100% -compress None -loop 0 video.mpeg
```

## 1.11 Interfaces gráficas para o gnuplot

- Você encontra no site do gnuplot informações atualizadas sobre interfaces gráficas para o gnuplot.
- Veja o `kile`, <http://kile.sourceforge.net>.
- Veja o `Unignuplot`, <http://unicalculus.sourceforge.net>.



## 1.12 Leituras adicionais

- **Iniciante:**
  - Site do gnuplot: <http://www.gnuplot.info>.
  - Tutorial iniciante: `file:gnuplot/tutorial1.ps`.
  - Guia de referência: `file:/gnuplot/Manual-lunardi/gpcard.ps`.
- **Intermediário:**
  - Tutorial intermediário: <http://www.cs.uni.edu/Help/gnuplot/TOC.html>.
  - FAQ: <http://www.gnuplot.info/faq.html>,  
`file:/gnuplot/faq/gnuplot-faq.html`.
- **Avançado:**
  - Plotando contornos: `file:gnuplot/Manual-lunardi/contours.ps`.
  - Plotando superfícies: `file:gnuplot/Manual-lunardi/surface1.ps`, `file:gnuplot/Manual-lunardi/surface2.ps`.
  - Plotando gráficos para o latex: `file:gnuplot/Manual-lunardi/tutorial.ps`.
  - Perl e gnuplot como ferramentas computacionais no ensino de ciências exatas: <http://www.revistadolinux.com.br/artigos/005,030,3,118,855.html>.
  - Manual: `file:gnuplot/Manual-lunardi/manual.ps`.
  - Manual oficial: `file:gnuplot/gnuplot-v3.7.pdf`.
  - Links adicionais: <http://www.gnuplot.info/links.html>.
  - Gnuplot em real time: <https://www.youtube.com/watch?v=GgO55NzBBgs>.