IDL

AGA503

19 de maio de 2014

IDL

- **IDL**: **I***nterative* **D***ata* **L***anguage*
- Desenvolvido pea *Research Inteligent System (RSI)*

Links interessantes:

- Tutorial em português:
  - `http://www.sulsoft.com.br/idl/guia/`
- IDL Astronomy User's Library
  - `http://idlastro.gsfc.nasa.gov/homepage.html`
  - `http://idlastro.gsfc.nasa.gov/ftp/astron.tar.gz`
- Coyote's Guide to IDL Programming
  - `http://www.dfanning.com`
  - `ftp://ftp.dfanning.com/pub/dfanning/outgoing/coyote2nd/`

Bibliotecas

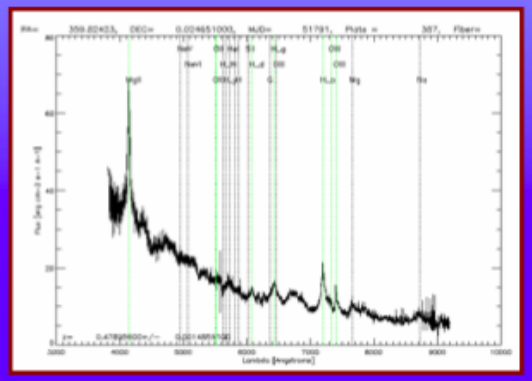
- FITS (fotometria)
- Manipulação de imagens (STSDAS, etc.)
- Matemática e Estatística
- Estatística robusta
- Elaboração de gráficos
- Banco de dados
- Entre outros...

✓IDL e a Astronomia

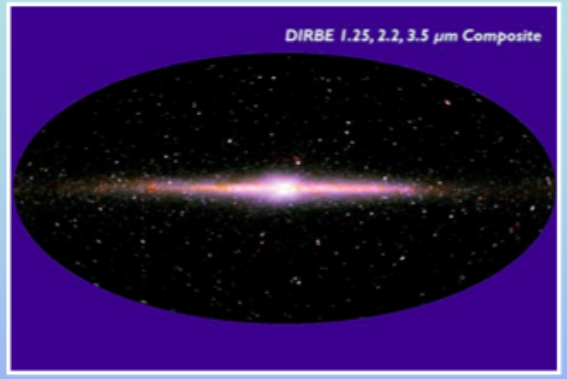
- Primeiros pacotes (GSFC) analisar espectros IUE, imagens HST

- Outros pacotes escritos em IDL: Chandra, ROSAT, COBE, GRO, HST/NICMOS, HST/ACS, SOHO, ISO, Spitzer, Herschel-Planck, SDSS, entre outros ....

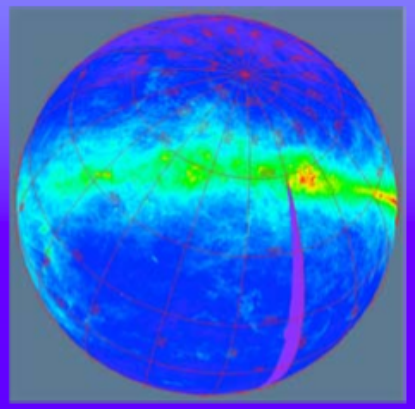
Espectro (SDSS)



DIRBE/COBE



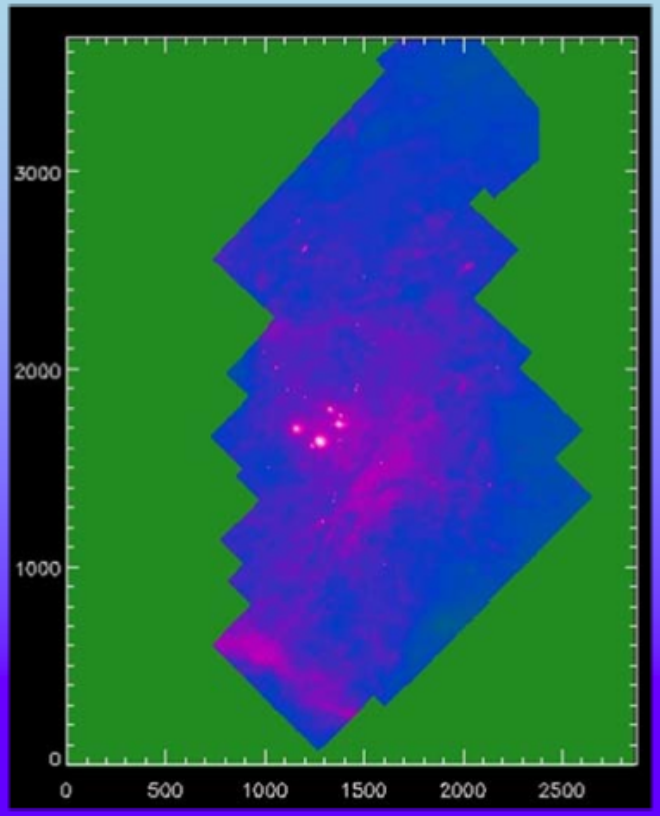
HEALPix (Herschel-Planck) mapa de extinção de Dobashi et al. (2005)

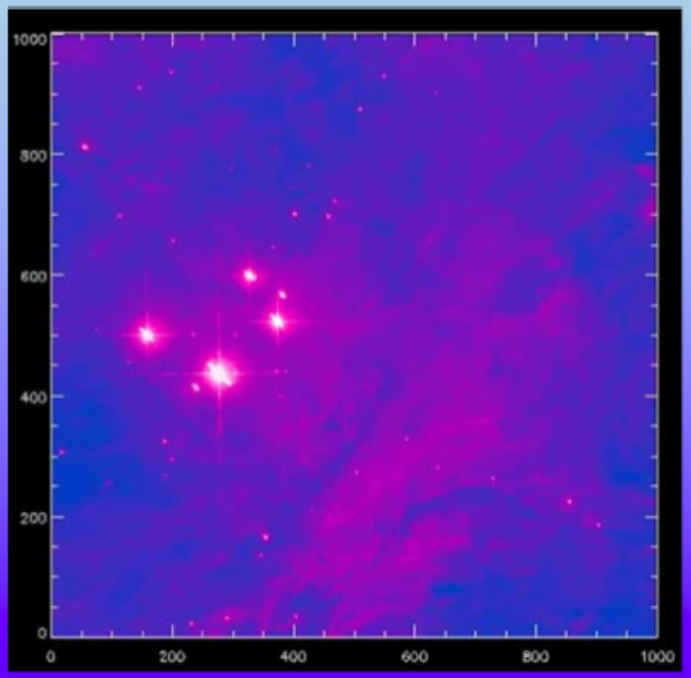


Curso de IDL no IAG (2009).

✓ Imagens HST

http://casa.colorado.edu/~bally/ HST/HST/master/



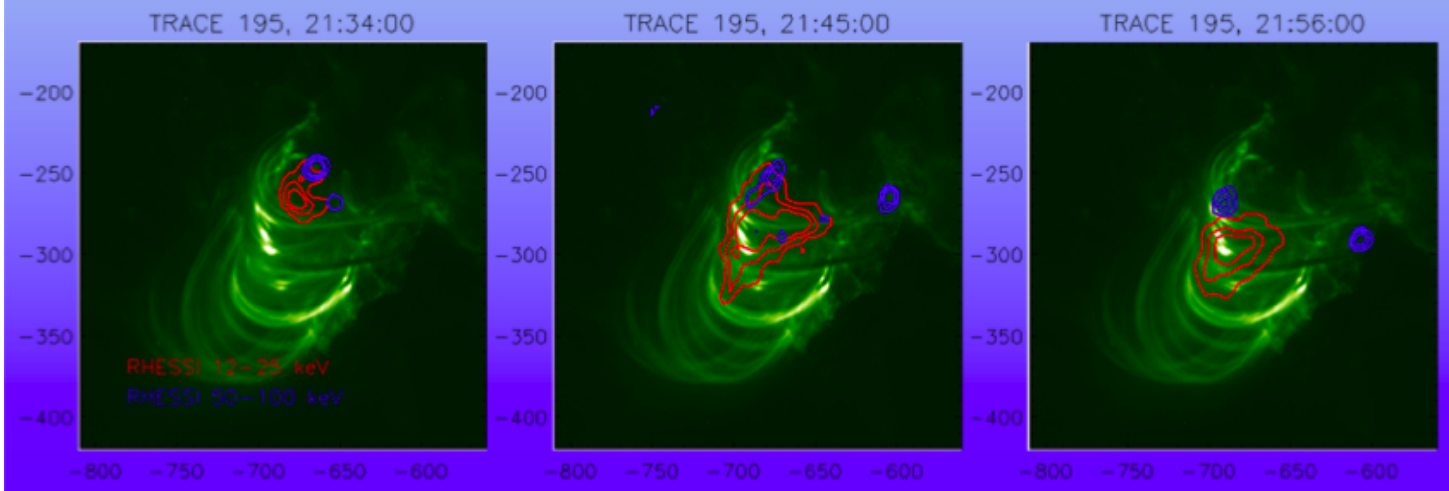


Curso de IDL no IAG (2009).

✓ Mapas do sol

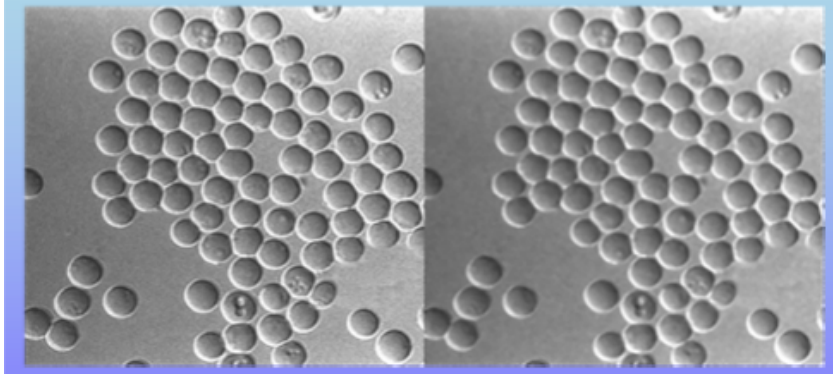
http://orpheus.nascom.nasa.gov/~zarro/idl/maps.html

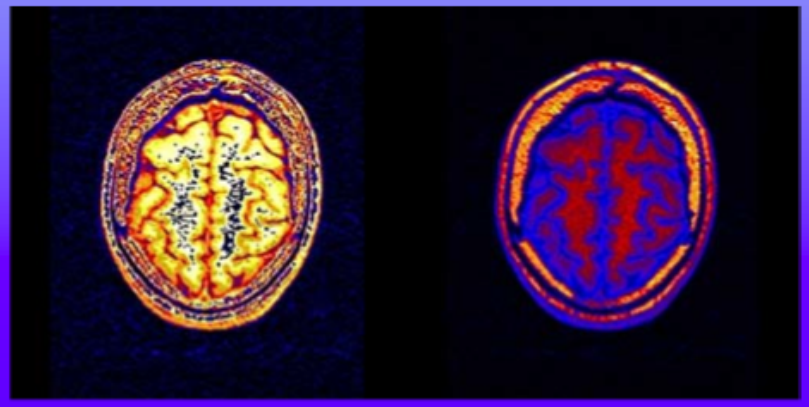
http://hesperia.gsfc.nasa.gov/~ptg/trace-align/

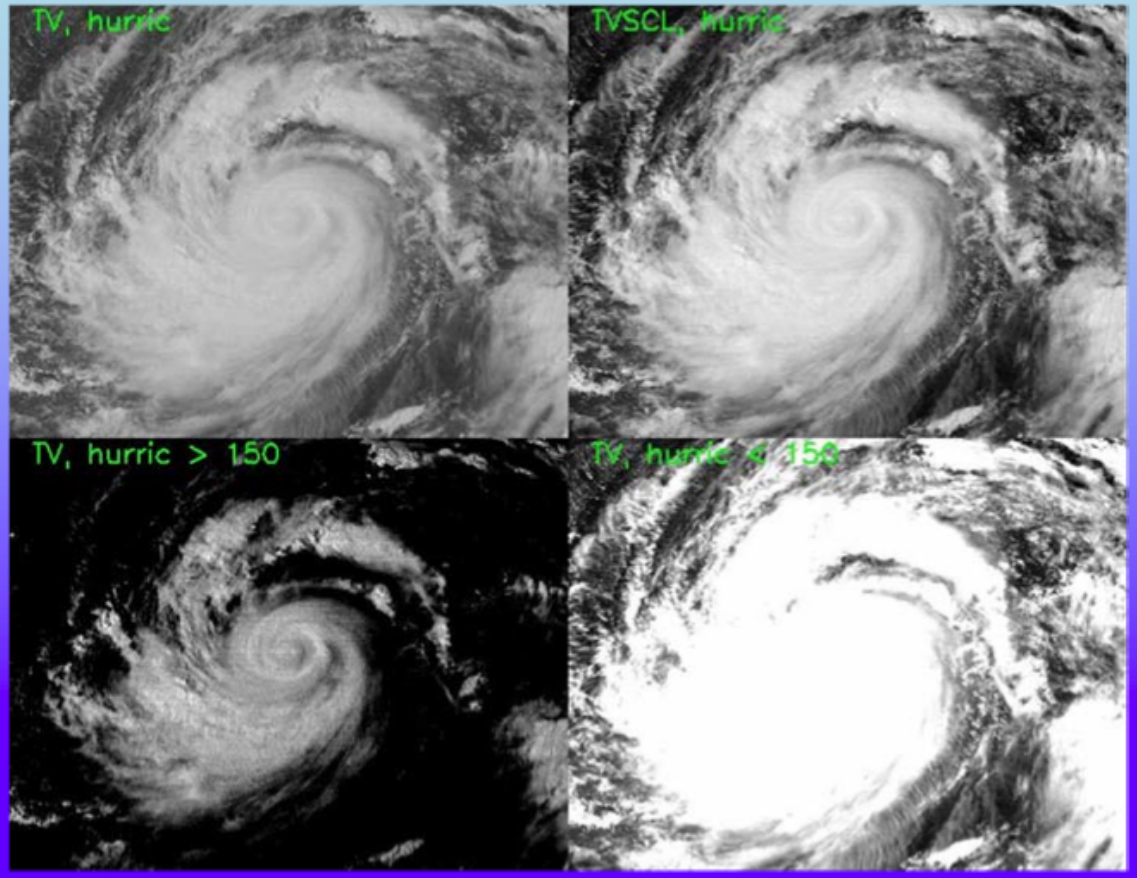


Curso de IDL no IAG (2009).

✓ Processamento de imagens






TV, hurric
TVSCL, hurric
TV, hurric > 150
TV, hurric < 150

Executando o IDL

- **IDL**: Linhas de comando
- **IDLDE**: Ambiente de desenvolvimento

Alguns comandos:

- **.compile**: compila programas ou funções
- **.run** ou **.r**: compila e roda
- **$comando**: roda um comando unix
- **;**: comentário no código ou na linha de comando
- **?**: ajuda

✓ IDLDE (Development Enviroment)

Curso de IDL no IAG (2009).

Exemplo:

- crie dois vetores (x,y) com 100 elementos cada (n);
- atribua valores para o vetor x = [0,2π] com passos de 2π/100;
- faça y = cos(x)

Usando C

```
#define N 101
int n;
float pi=3.1415926;
float x[N], y[N];
dx = 2*pi/N;
for(n=0;n<N;n++) {
x[n] = n*dx;
y[n] = cos(x[n]);
}
```

Usando IDL (loop)

```
n = 101
dx = 2*!pi/n
for i=0,n-1 do begin
    x[n] = n*dx
    y[n] = cos(x[n])
endfor
end
```

Usando IDL (vetores)

```
n = 101
x = findgen(n)*2*!pi/(n-1)
y = cos(x)
```

Sintaxe

Separação por vírgula:

- PLOT,x,y
- help,x,y
- PRINT,x,y

Tipos de variáveis:

- **INTEGER**: Ex.: a = 1
- **FLOAT**: Ex.: a = 1. / b = 1.e3
- **DOUBLE**: Ex.: a = 1.d3
- **STRING**: Ex.: a = "bla"

Operações Matemáticas

Divisão:

- **IDL> x = 8/3** (resulta em 2)
- **IDL> x = 8./3** (resulta em 2.667)
- **IDL> x = 8/3.** (resulta em 2.667)

Exponenciação:

- **IDL> x = 1024∧2** (resulta em 0. Porque?)
- **IDL> x = 1024.∧2** (resulta em 1.04858e+06)
- **IDL> x = 1024∧2.** (resulta em 1.04858e+06)

Operações Matemáticas

Logaritmo:

- **IDL> x = ALOG(100)** (retorna ln(100))
- **IDL> x = ALOG10(100)** (retorna log10(100))

Mais alguns comandos:

READ e PROMPT:

- READ, x, PROMPT='Digite o valor de x:'

Operadores:

- ( ): para chamar uma função, ex: x = fltarr(10)
- [ ]: indice do vetor/matriz, ex: print, x[2]
- eq: igual a
- ne: diferente de
- lt: menor que
- le: igual ou menor que
- gt: maior que
- ge: igual ou maior que
- &&: E lógico
- ||: OU lógico

Programas e Funções:

PROGRAMA

```
PRO nome
        ... comandos...
END

FUNCTION, nome, par1, par2, ...
        ... comandos...
RETURN, valor
END
```

Exercício: fazer um programa que peça a frequência da função coseno e faça um gráfico desta função

Controles de Fluxo

IF ... THEN ... ELSE ...

```
IF (condicao) THEN BEGIN
        ... comandos ...
ENDIF ELSE BEGIN
        ... comandos ...
ENDELSE
```

CASE ... OF ... ENDCASE ...

```
CASE x OF
        1: PRINT, 'One'
        2: PRINT, 'Two'
        3: PRINT, 'Three'
        4: PRINT, 'Four'
ENDCASE
```

Exercício: modificar o programa anterior para que ele apresente mensagem de erro caso a frequência dada seja negativa.

Controles de Fluxo

FOR ... DO ...

FOR i = 0 , n-1, step DO ...comandos...

WHILE ... DO ...

WHILE condicao DO ...comandos...

FOR ... DO ...

```
FOR i = 0 , n-1, step DO BEGIN
        ...comandos...
ENDFOR
```

WHILE ... DO ...

```
WHILE condicao DO BEGIN
        ...comandos...
ENDWHILE
```

Exercício: modificar o programa anterior para que ele sobreplote 3 harmônicos do cosseno.

Controles de Fluxo

BREAK

Define um ponto para saída imediata

```
FOR i=0, 99 DO BEGIN
        IF (i GT 50) THEN BREAK
        PRINT, i
ENDFOR
```

GOTO

Define um ponto para redirecionamento

```
FOR i=0, 5 DO BEGIN
        PRINT, 'i = ', i
        IF (i GT 2) THEN GOTO,
saida
ENDFOR
        saida: PRINT, 'i maior que 2'
```

Funções:

FUNCTION

```
FUNCTION, nome, par1, par2, ...
        ... comandos...
RETURN, expressão
END
```

Exemplo: função que retorne o valor médio de um vetor qualquer

```
FUNCTION media, vetor
m = TOTAL(vetor)/N_ELEMENTS(vetor)
RETURN, m
END
```

```
IDL> .compile media
IDL> x = findgen(100)
IDL> PRINT, media(x)
```

Exercício 1:

Faça uma função que, dados valores de temperatura T e comprimento de onda λ, retorne a função de Planck normalizada:

$$\frac{B_\lambda}{\int_0^\infty B_\lambda d\lambda} = \frac{1}{\sigma T^4} \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k T}} - 1}$$

Escrevendo de outra forma:

$$\frac{c_1}{\sigma T^4 \lambda^5} \frac{1}{e^{\frac{c_2}{\lambda T}} - 1}$$

onde

$\sigma = 5.67051 \times 10^{-5}$ (cgs)
$c_1 = 3.7417749 \times 10^{-5}$ (cgs)
$c_2 = 1.4387687$ (cgs)

Vetores e Matrizes

Tipo	Função de Criação	Índice que gera a Função
Byte	BYTARR	BINDGEN
Integer	INTARR	INDGEN
Float	FLTARR	FINDGEN
Double	DBLARR	DINDGEN
Complex	COMPLEXARR	CINDGEN
Double complex	DCOMPLEXARR	DCINDGEN
String	STRARR	SINDGEN

Curso de IDL no IAG (2009).

Vetores

Criando vetores:

- x = FLTARR(5): vetor com 5 elementos todos com valor 0.
- x = [3,5,8,9,45]
- x = FINDGEN(5): vetor com 5 elementos com valores variando de 0 a 4 (n-1).

Algumas funções úteis:

- MIN(x): valor mínimo do vetor
- MAX(x): valor máximo do vetor
- WHERE(x EQ 5): índice do vetor correspondente ao elemento 5
- Ordenar valores em ordem crescente:

```
y = SORT(x)
PRINT, x[y]
```

- Inverter a ordem de um array:

```
z = REVERSE(x)
PRINT, x[z]
```

- N_ELEMENTS(x): tamanho do vetor

Exercício: brincar com a funcionalidade de WHERE

Matrizes

Criando matrizes:

- x = FLTARR(3,3): matriz 3x3
- x = [[0,1,2],[3,4,5],[6,7,8]]
- x = FINDGEN(3,3): matriz 3x3

No IDL: m[coluna,linha]

Algumas funções úteis:

- Soma: m1 + m2
- Subtração: m1 - m2
- Multiplicação: m1 # m2
- Transposta: TRANSPOSE(m)
- Inversa: INVERT(m)
- Determinante: DETERM(m)
- Achando o mínimo da matriz e seu indice:

  minimo = MIN(matriz,location)
  index = ARRAY_INDICES(matriz, location)

Exercício: construir uma matriz 3x3 e calcular seu determinante.

Leitura de Arquivos

Rotinas	Finalidade
OPENR	Abre um arquivo para leitura.
OPENW	Abre um arquivo para escrita.
OPENU	Abre um arquivo para atualização (leitura e/ou escrita).
CLOSE	Fecha um arquivo ou uma lista de arquivos.
FREE_LUN	Fecha arquivos e limpa as unidades de arquivo.

Curso de IDL no IAG (2009).

- LUN: número de unidade lógica, é um número simples associado a um arquivo do IDL.
- GET_LUN: deixa que o IDL especifique um LUN na escala de 100 – 128, para prevenir conflitos com outros LUNs já em uso.
- READF: lê os dados do arquivo.
- PRINTF: escreve dados no arquivo.

Exemplo:

```
OPENR, lun, 'dados.txt',/GET_LUN
READF, lun, <nome do array>
CLOSE, lun

FREE_LUN, lun
```

Exercício 2:

Faça um programa que leia o arquivo com os dados do EP3.5 e faça um gráfico com os dados.

Dica:

- Para saber o número de linhas do arquivo, existe uma função chamada FILE_LINES, ex:

```
n=FILE_LINES('dados.txt')
```

Gráficos

Comando PLOT

IDL> ?PLOT

Exemplo:

```
n = 101
x = FINDGEN(n)*2.*!pi/(n-1)
y = COS(x)

PLOT, x, y
PLOT, x, y, XTITLE = 'Eixo X', YTITLE = 'Eixo Y', TITLE = 'Grafico'
```

Ex: coloque títulos no gráfico do exercício anterior

Gráficos

Comando PLOT

PLOT, x, y, $
XTITLE = 'Eixo X', YTITLE = 'Eixo Y', TITLE = 'Grafico', $
PSYM = 1

Valor	Símbolo
1	+
2	*
3	Período (.)
4	Diamante
5	Triângulo
6	Quadrado
7	X
10	Histograma

Ex: coloque símbolos no gráfico do exercício anterior. Dica: para ter tanto símbolos quando a linha, faça PSYM = -1

Gráficos

Comando PLOT

```
PLOT, x, y, $
    XTITLE = 'Eixo X', YTITLE = 'Eixo Y', TITLE = 'Grafico', $
    LINESTYLE = 2
```

Valor	Estilo de linha
0	Sólida
1	Pontilhada
2	Tracejada
3	Pontilhada - tracejada
4	Tracejada – pontilhada (2X)
5	Tracejado longo

Gráficos

Múltiplas Janelas Gráficas

```
WINDOW, 0, XSIZE=800, YSIZE=800
PLOT, x, y, $
    XTITLE = 'Eixo X', YTITLE = 'Eixo Y', TITLE = 'Grafico', $
    PSYM = 1

WINDOW, 1, XSIZE=800, YSIZE=800
PLOT, x, y*y, $
    XTITLE = 'Eixo X', YTITLE = 'Eixo Y', TITLE = 'Grafico', $
    PSYM = 2
```

Ex: gere gráficos em 2 janelas diferentes.

Gráficos

Comandos PLOT e OPLOT

```
PLOT, x, y, $
    XTITLE = 'Eixo X', YTITLE = 'Eixo Y', TITLE = 'Grafico', $
    LINESTYLE = 2

OPLOT, x, y*y, $
    LINESTYLE = 3, $
    COLOR = FSC_Color('red')
```

OBS: o FSC_Color é uma rotina muito útil disponível no site Coyote. Ex: Baixá-la para o seu computador e fazer um gráfico em cores.

Gráficos

Comandos XYOUTS e LEGEND

XYOUTS, px, py, 'Texto'

LEGEND, ['legenda1', 'legenda2'], PSYM = [1,2], $
POS = [px,py], COLOR = [cor1, cor2], $
TEXTCOLORS = [cor1,cor2]

Ex: Colocar uma legenda no gráfico do exercício 2.

Gráficos

Comando !P.MULTI: vários gráficos na mesma janela.

!P.MULTI = [x, y, z]

- **x**: separa o gráfico em páginas, mas só será possível visualizar o último gráfico;
- **y**: coloca em colunas diferentes;
- **z**: coloca em linhas diferentes.

Exemplo:

```
!P.MULTI = [0,1,2]

PLOT, x, y, $
    XTITLE = 'Eixo X', YTITLE = 'Eixo Y', TITLE = 'Grafico' $
    LINESTYLE = 2

PLOT, x, y*y, $
    LINESTYLE = 3 $
    COLOR = FSC_Color('red')
```

Gráficos

Comando ERRPLOT: barras de erro

PLOT, x, y

ERRPLOT, x, y-sigma, y+sigma

Gráficos

Comando SET_PLOT e DEVICE: salva os gráficos em .ps e .eps

```
SET_PLOT, 'ps'
DEVICE, FILENAME = 'grafico.eps', /ENCAPS
    PLOT, x, y $
    outros comandos desejados ...
    DEVICE, /CLOSE
SET_PLOT, 'X'
```

Ex: Gerar um eps do gráfico do exercício 2.

Gráficos

Comando IPLOT: cria uma ferramenta inteligente para manipular o gráfico.

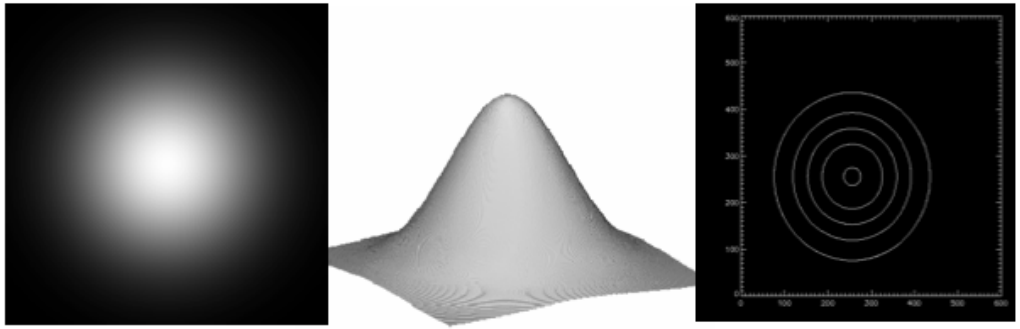
```
IPLOT, x, y, $
    XTITLE = 'Eixo X', YTITLE = 'Eixo Y', TITLE = 'Grafico' $
    LINESTYLE = 2
```

Gráficos 3D

Comandos TV (TVIMAGE) , SHADE_SURF (SURFACE) e CONTOUR.

3d surfaces

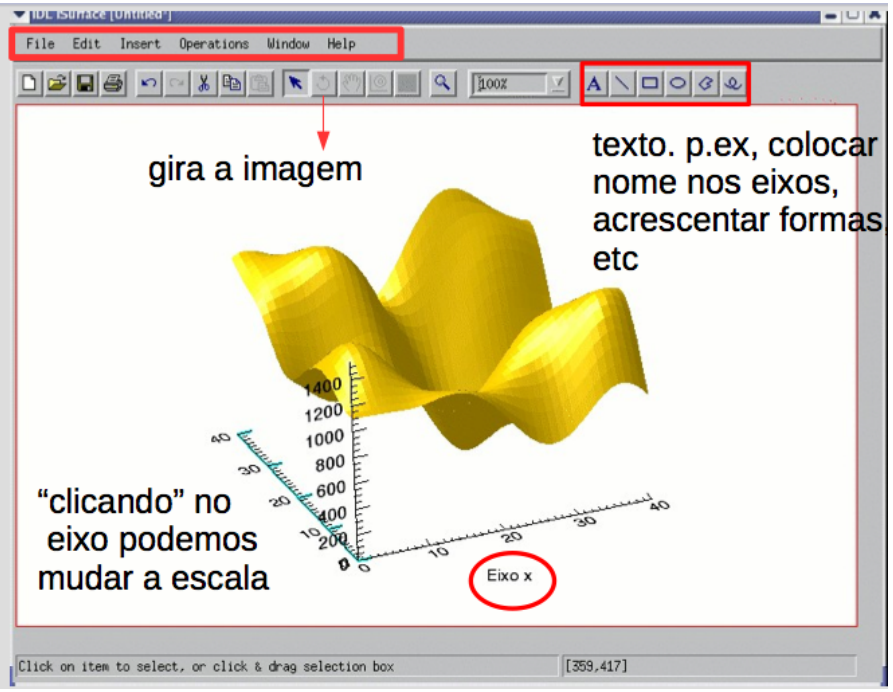
```
window,xsize=512,ysize=512 ; create a window
x = findgen(512)-256
y = x
xy = exp(-x*x/20000.0) # exp(-y*y/20000.0)
img = byte(255.0*xy)
tv, img
shade_surf,img,xstyle=4,ystyle=4,zstyle=4
contour,img
```



Gráficos 3D

Comandos ISURFACE, IIMAGE, IVOLUME, SURFACE e ICONTOUR.

- Bem mais convivial por ser interativo
- Disponível a partir da versõa 6.0!
- Depois das alterações salva diretamente a imagem



Curso de IDL no IAG (2009).

EP

- Ler a tabela de dados;
- Gerar dois vetores (para as duas temperaturas T1 e T2), com valores igualmente espaçados, entre um valor mínimo e um máximo dados pelo usuário;
- Gerar um modelo, definido com a suma de duas funções de Planck normalizadas;
- Calcular o χ^2 para cada combinação entre T1 e T2;
- Gerar o mapa de χ^2 e descobrir o mínimo;
- Plotar dois gráficos: χ^2 versus T1 (fixando T2 no melhor ajuste) e χ^2 versus T2 (fixando T1 no melhor ajuste);
- Plotar o gráfico com os pontos, ajuste e resíduos;
- Por fim, imprimir na tela o $\chi^2_{reduzido}$ e as duas temperaturas correspondentes ao melhor ajuste.

DICA: Para plotar o mapa, usar a função TVIMAGE:

```
PLOT, T1, T2
TVIMAGE, 255.-( matriz / MAX( matriz )*255. ), /NOINTERPOLATION, /OVERPLOT
```