# Introdução à Física Computacional (4300218)

Profa. Kaline Coutinho
kaline@if.usp.br
Sala 2056 – Edifício Principal

Aula 4

Programação em Pythom para físicos:
Gráficos 2D com funções da biblioteca
`matplotlib.pylab`

# Pacote pylab

- Este pacote faz parte da biblioteca `matplotlib` (documentação on-line `matplotlib.org`).
- Foi inspirado no programa Matlab e tem ferramentas que produzem os mesmos tipos de gráficos.
- Nesta aula vamos apresentar os comandos para gerar gráficos de linha e símbolos e densidades com diferentes tons de cores. Mas outros tipos de gráfico são possíveis como: de contorno, polar, pizza, histograma, etc. (ver na documentação on-line).

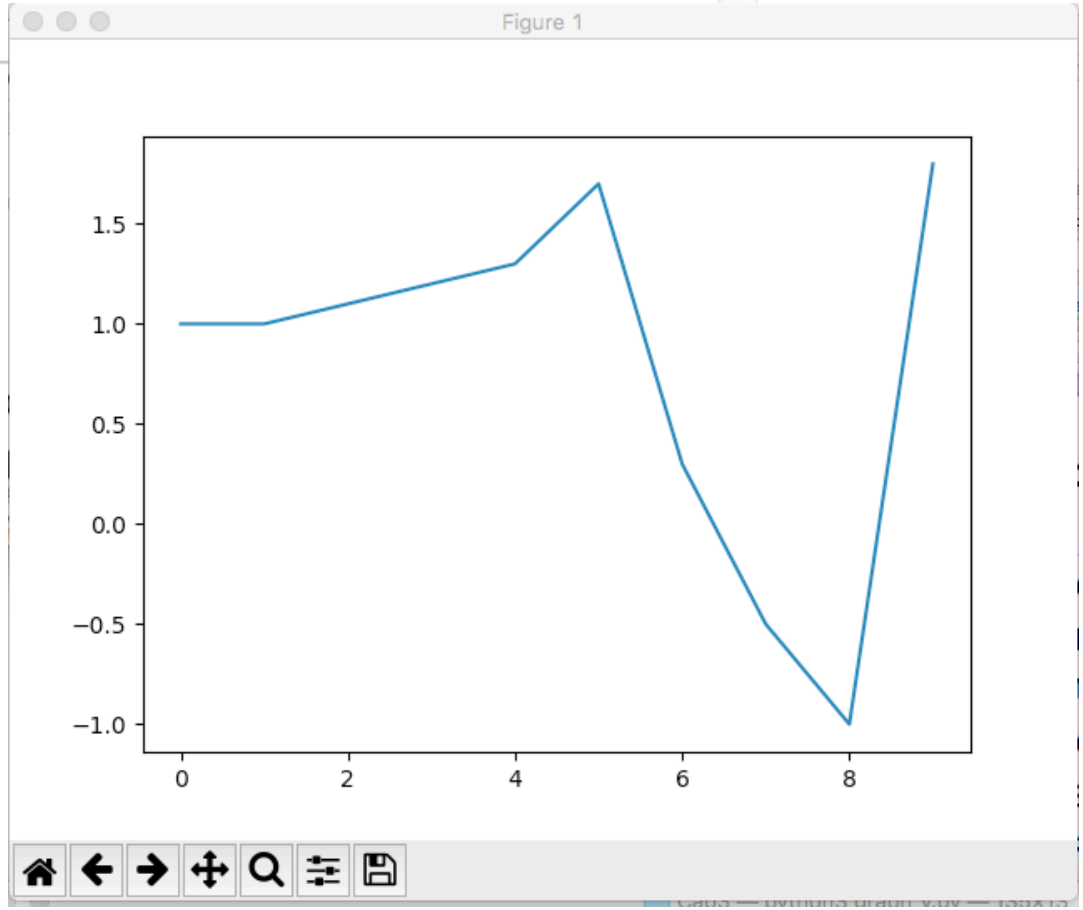
# Funções: `plot()` e `show()`

- A função `plot(y)` ou `plot(x,y)` cria um gráfico de valores especificados na memória.
- A função `show()` mostra o gráfico numa janela e bloqueia a execução dos comandos seguintes até a janela ser fechada.
- Ambas funções devem ser importadas

# Exemplo 1: `plot(y)`

graph_y.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplinas/FisicaC...

```
from pylab import plot, show
y = [ 1.0, 1.0, 1.1, 1.2, 1.3, 1.7, 0.3, -0.5, -1.0, 1.8]
plot(y)
show()
```




Neste programa chamado graph_y.py, dez valores reais são atribuidos ao vetor y e em seguida, estes valores são graficados no eixo y versus o índice do vetor [0,1,2,3,4,5,6,7,8,9] apresentado no eixo x.

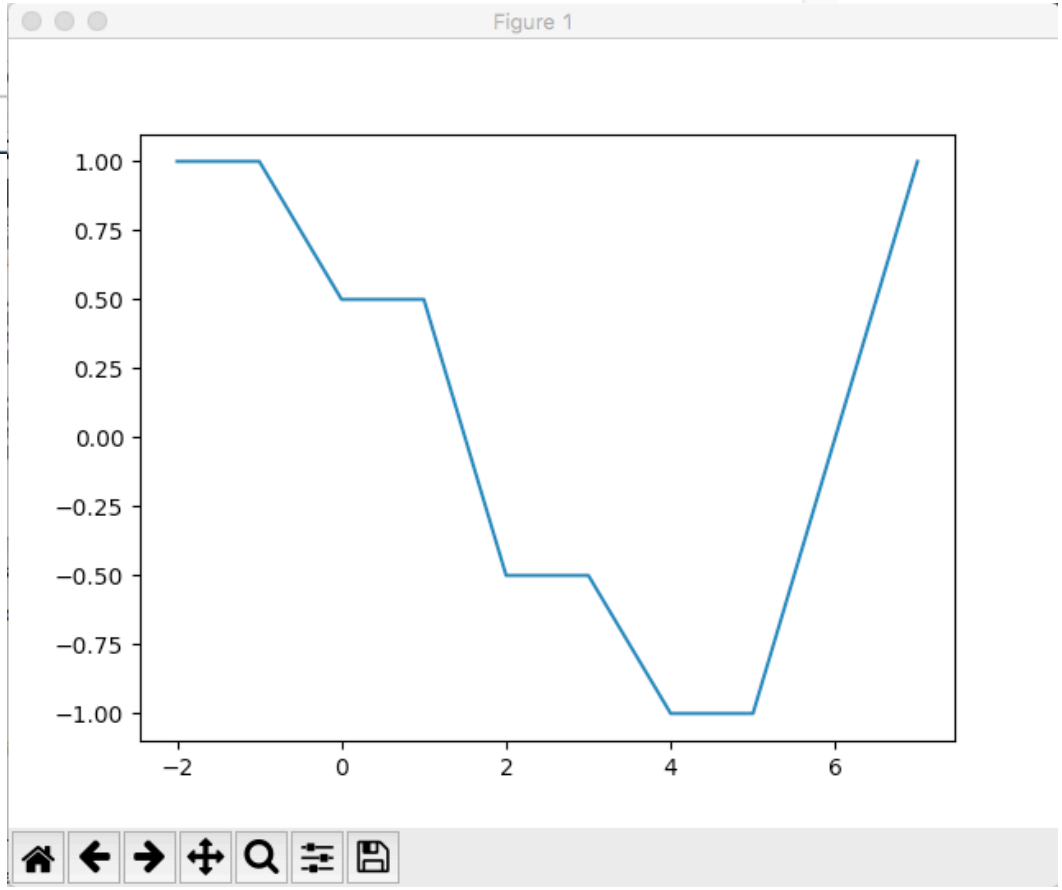
Note os botões na parte inferior da janela que apresentam várias funcionalidades como salvar a imagem, por exemplo.

# Exemplo 2: `plot(x,y)`

graph_xy.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplinas/FisicaComput...

```
from pylab import plot, show
x = [ -2, -1, 0, 1, 2, 3, 4, 5, 6, 7]
y = [ 1.0, 1.0, 0.5, 0.5, -0.5, -0.5, -1.0, -1.0, 0.0, 1.0]
plot(x,y)
show()
```

Neste programa chamado graph_xy.py, dez valores são atribuidos aos vetores x e y e em seguida, estes valores são graficados um versus o outro.



# Atribuição de valores aos vetores

- Os valores podem ser atribuidos aos vetores x e y das seguintes formas:

(i) todos os valores de uma única vez (já mostrados nos exemplos 1 e 2);

(ii) um valor de cada vez; ou

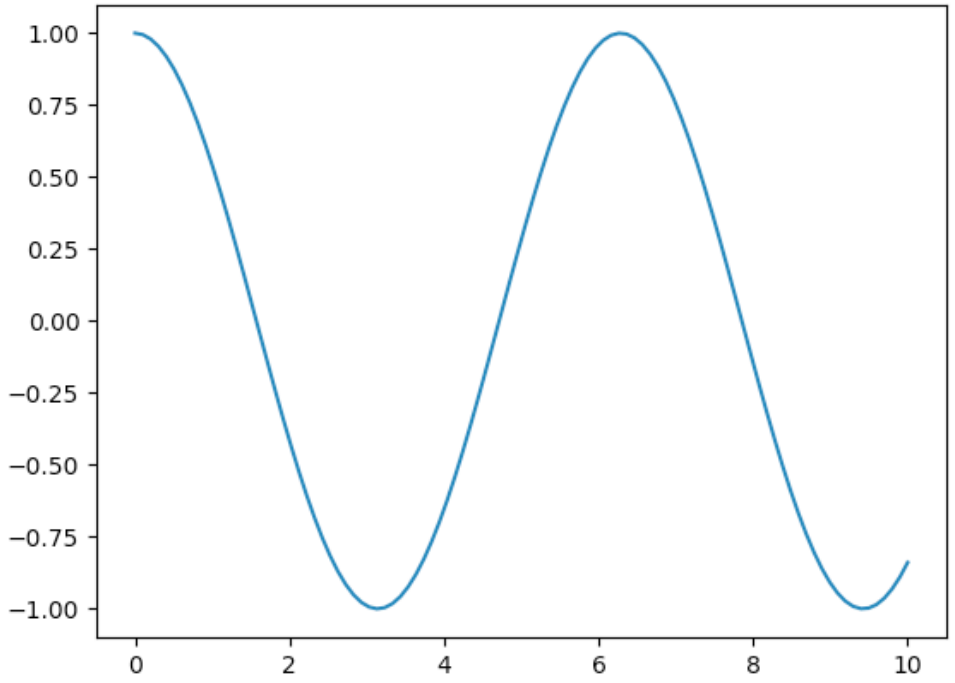
(iii) através da leitura em um arquivo.

# Exemplo 3: `plot(x,y)`

*graph_xy_cos.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplin...

```
from pylab import plot, show
from numpy import linspace, cos
x = linspace(0,10,100)
y = cos(x)
plot(x,y)
show()
```

Figure 1




Atribuição de valores aos vetores:

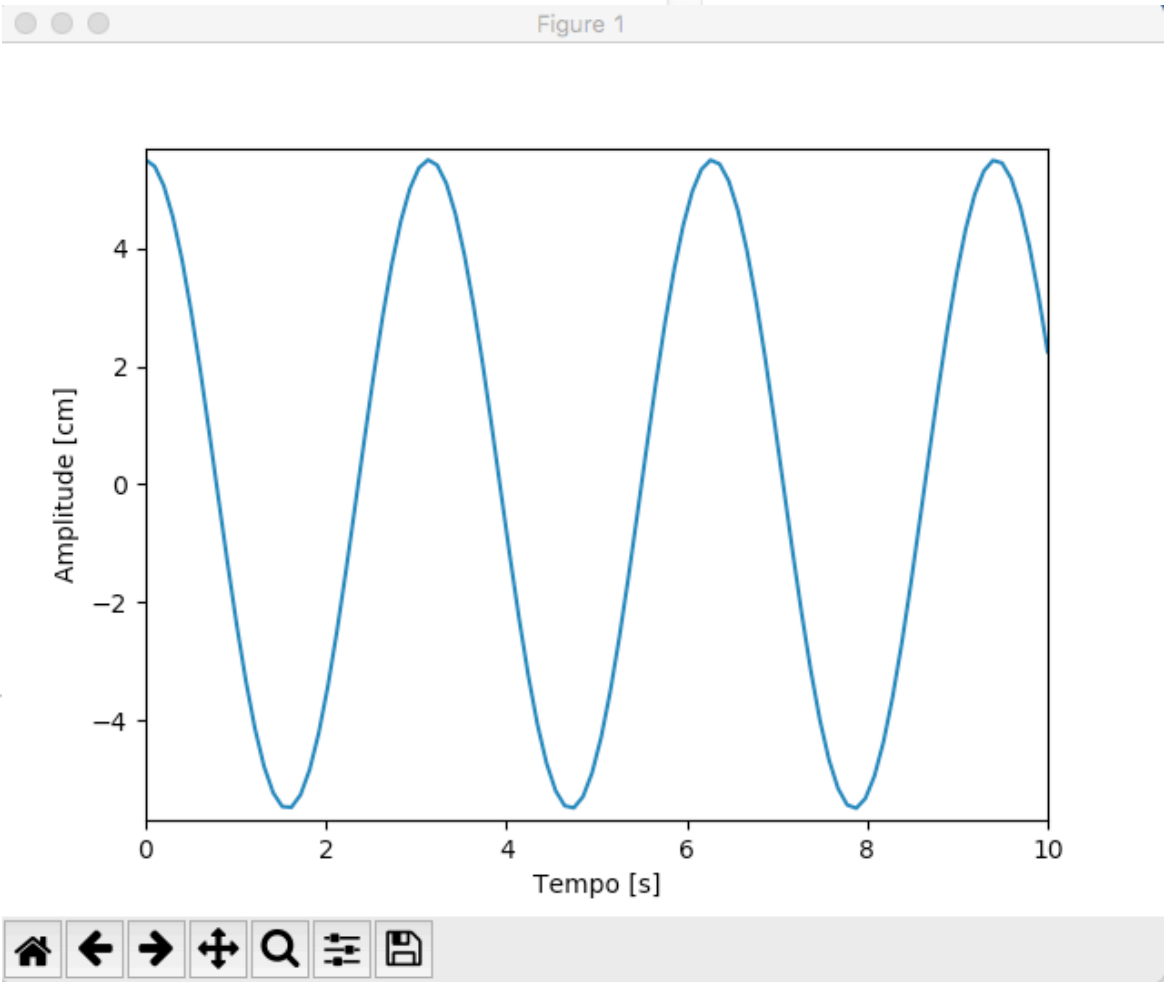
(i) todos os valores de uma única vez.

Note que a função `cos()` da biblioteca `numpy` recebe um vetor com argumento e não apenas um número como a mesma função da biblioteca `math`.

# Exemplo 4: `plot(x,y)`

graph_xy_oscilacao.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplinas...

```
from pylab import plot, show, xlim, ylim, xlabel, ylabel
from numpy import linspace
from math import cos

xt = []
yt = []
for t in linspace(0.0,10.0,100):
    xt.append(t)
    yt.append(5.5*cos(2.0*t))

plot(xt,yt)
xlabel('Tempo [s]')
ylabel('Amplitude [cm]')
xlim(0.0,10.0)
ylim(-5.7,5.7)
show()
```




Atribuição de valores aos vetores:

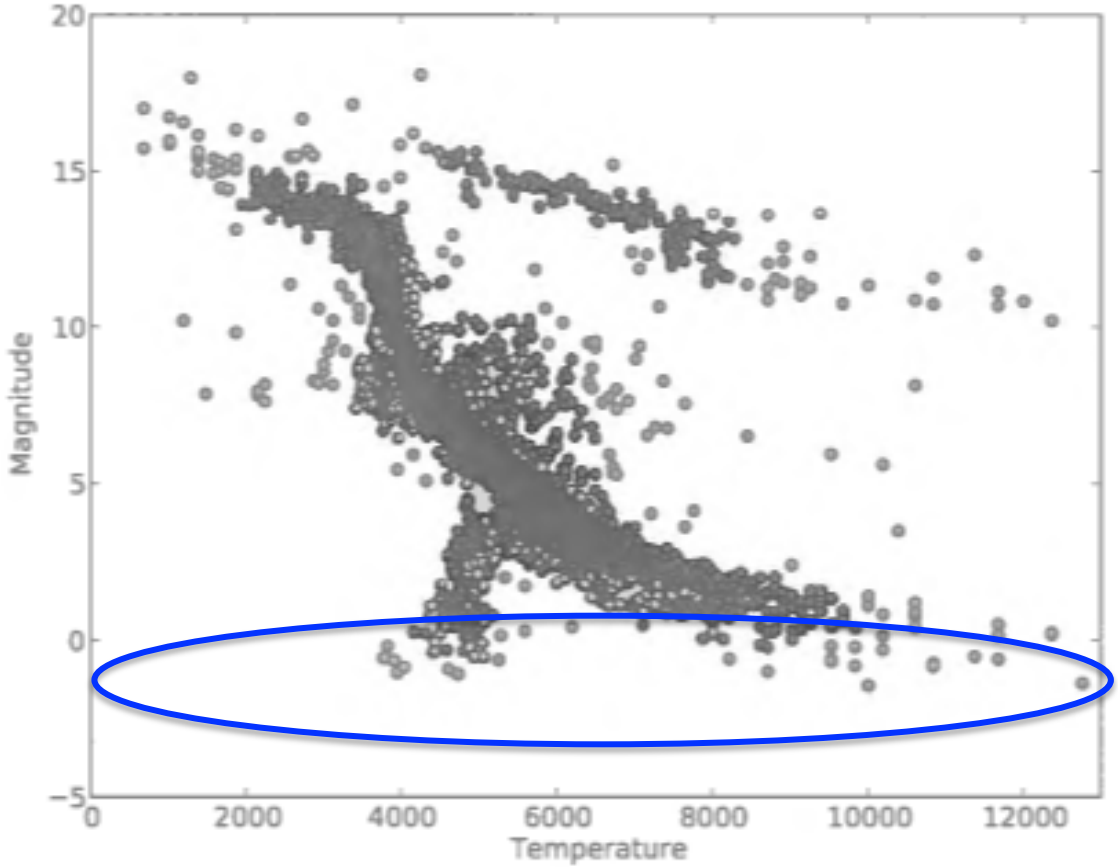
(ii) um valor de cada vez.

Opções de visualização: rótulos e escala dos eixos.

# Opções: `plot(x,y,"g--")`

- A letra representa a cor: r (red), g (green), b (blue), c (cyan), m (magenta), y (yellow), k (black) e w (white).
- Em seguida vem o tipo da linha: - (solid), -- (dashed), . (dot), o (circle), s (square).

# Dados experimentais




Os dados experimentais devem ser lidos de arquivos.

Não podemos gerar gráficos como este com linhas, pois não há significado físico em ligar uma medida de luminosidade de uma estrela com de outra.

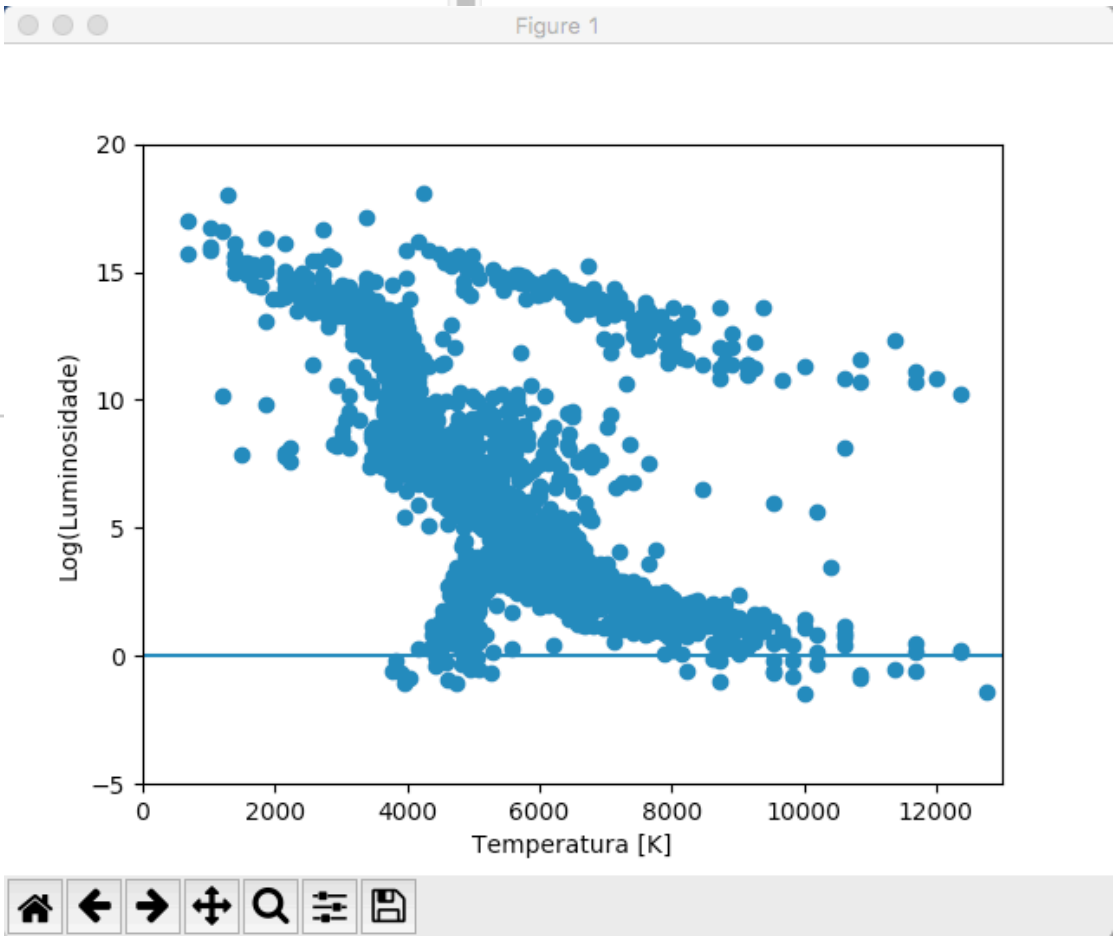
Existe luminosidade negativa?

**Figure 3.4: The Hertzsprung–Russell diagram.** A scatter plot of the magnitude (i.e., brightness) of stars against their approximate surface temperature (which is estimated from the color of the light they emit). Each dot on the plot represents one star out of a catalog of 7860 stars that are close to our solar system.

# Exemplo 5: `scatter(x,y)`

hrdiagram.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplinas...

```
from pylab import plot, scatter,xlabel,ylabel,xlim,ylim,sho
from numpy import loadtxt

data = loadtxt("stars.txt",float)
sun = loadtxt("sol.txt", float)
scatter(data[:,0],data[:,1])
plot(sun[:,0],sun[:,1])
xlabel("Temperatura [K]")
ylabel("Log(Luminosidade)")
xlim(0,13000)
ylim(-5,20)
```




Atribuição de valores aos vetores:
(iii) através da leitura em um arquivo.

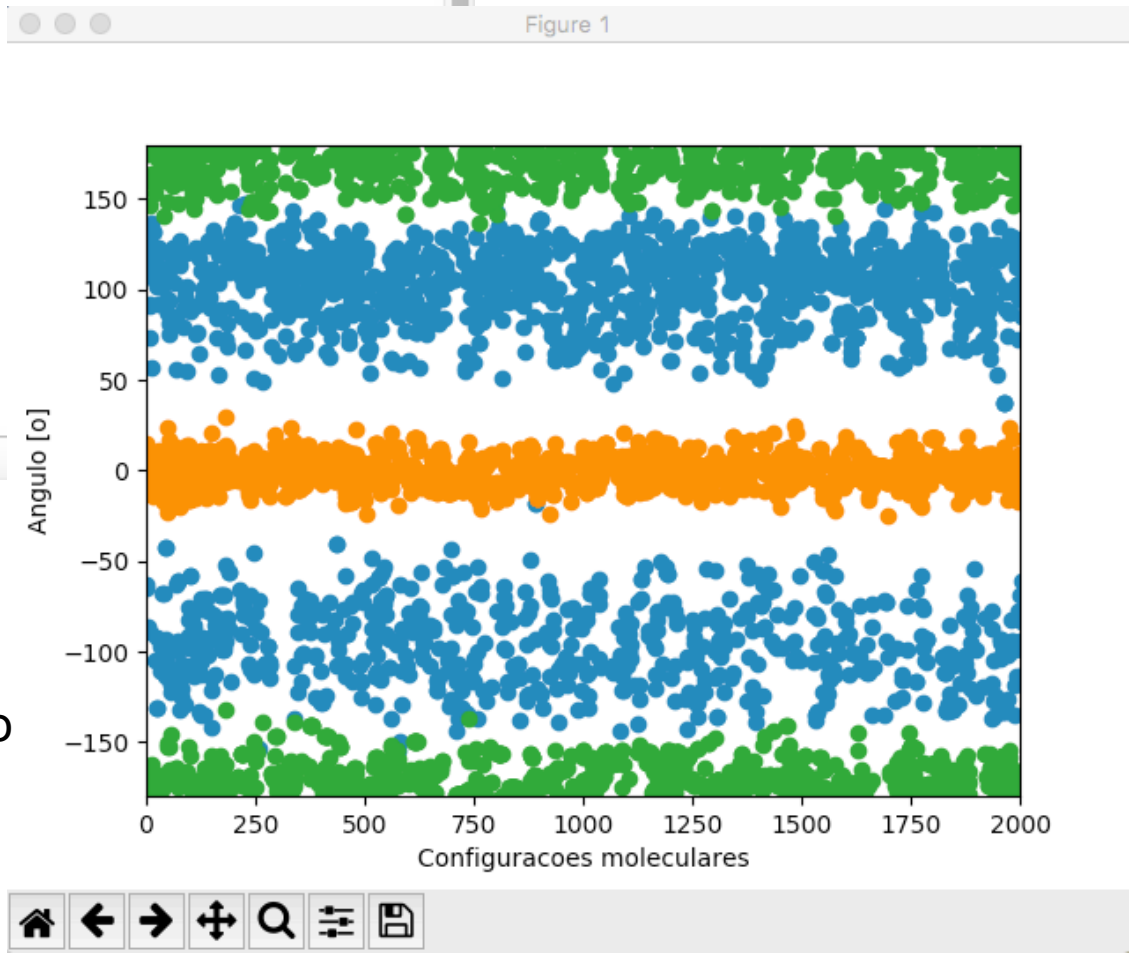
Devemos incluir unidades nos rótulos dos eixos.

# Exemplo 6: `scatter(x,y)`

graph_xy_file-line.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplinas...

```
from pylab import scatter, show, xlim, ylim, xlabel, ylabel
from numpy import loadtxt

data = loadtxt("multi_y.txt",float)
scatter(data[:,0],data[:,1])
scatter(data[:,0],data[:,9])
scatter(data[:,0],data[:,10])
xlabel('Configuracoes moleculares')
ylabel('Angulo [o]')
xlim(0,2000)
ylim(-180,180)

show()
```

Se a função `plot()` ou `scatter()` for chamada várias vezes, multiplos dados serão apresentados no mesmo gráfico.



# Exemplo 7: `hist(x)`

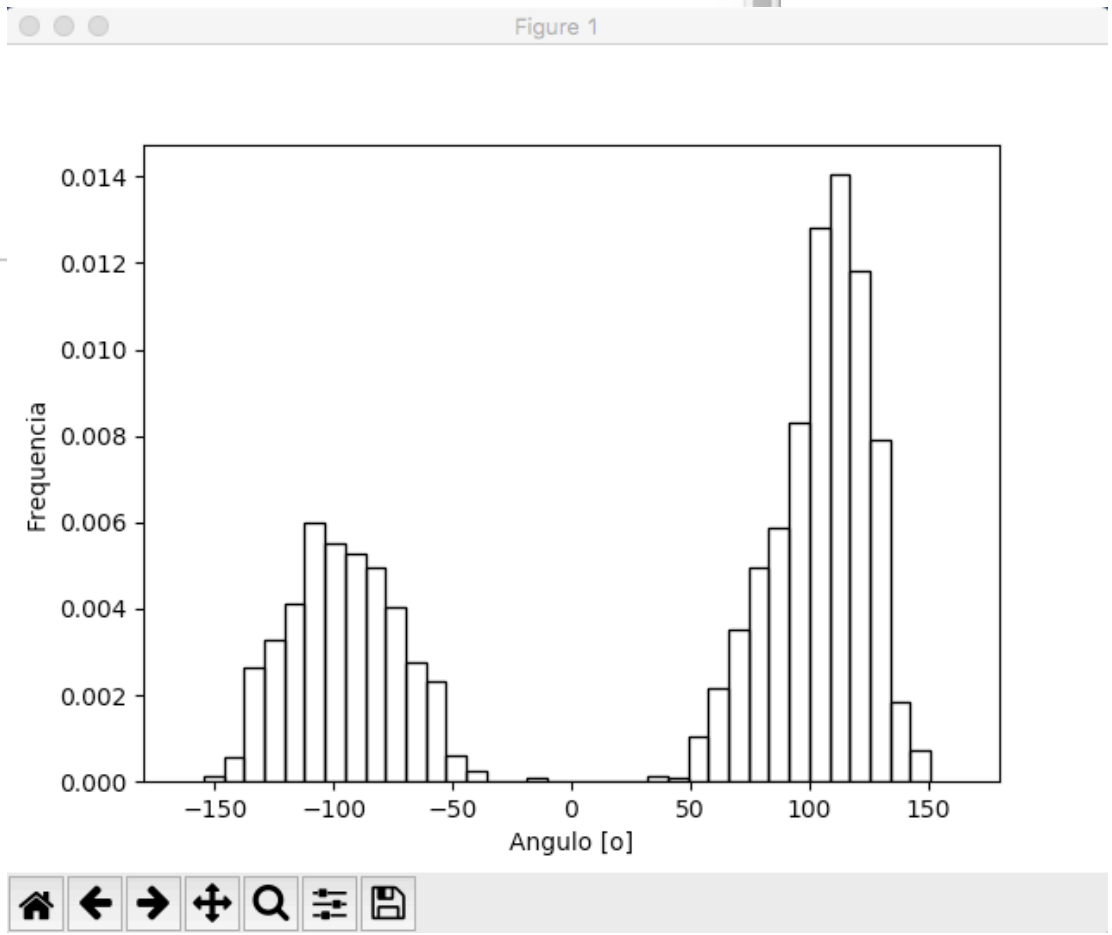
graph_xy_file-hist.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplinas/FisicaCo...

```
from pylab import hist, show, xlim, ylim, xlabel, ylabel
from numpy import loadtxt

data = loadtxt("multi_y.txt",float)
hist(data[:,1],36,density=1,histtype='bar',fill=False)
xlabel('Angulo [o]')
ylabel('Frequencia')
xlim(-180,180)

show()
```

A função `hist()` gera o histograma dos dados. O número 36 representa o número de intervalos (bin) do histograma.

`histtype =` bar ou step

`fill =` True ou False



# Gráfico de densidade 2D

- Este tipo de gráfico é muito usado na Física para analisar numa superfície: variações de temperatura ou cargas, deposição de átomos, intensidades de ondas, etc., onde as cores ou tons de cinza representam diferentes intensidades da grandeza analisada.
- A função `imshow(x,y)` apresenta cores dos dados no formato: `data[i,j]` seguindo uma matriz onde `i` representa colunas e `j` representa linhas com a origem no canto superior esquerdo:

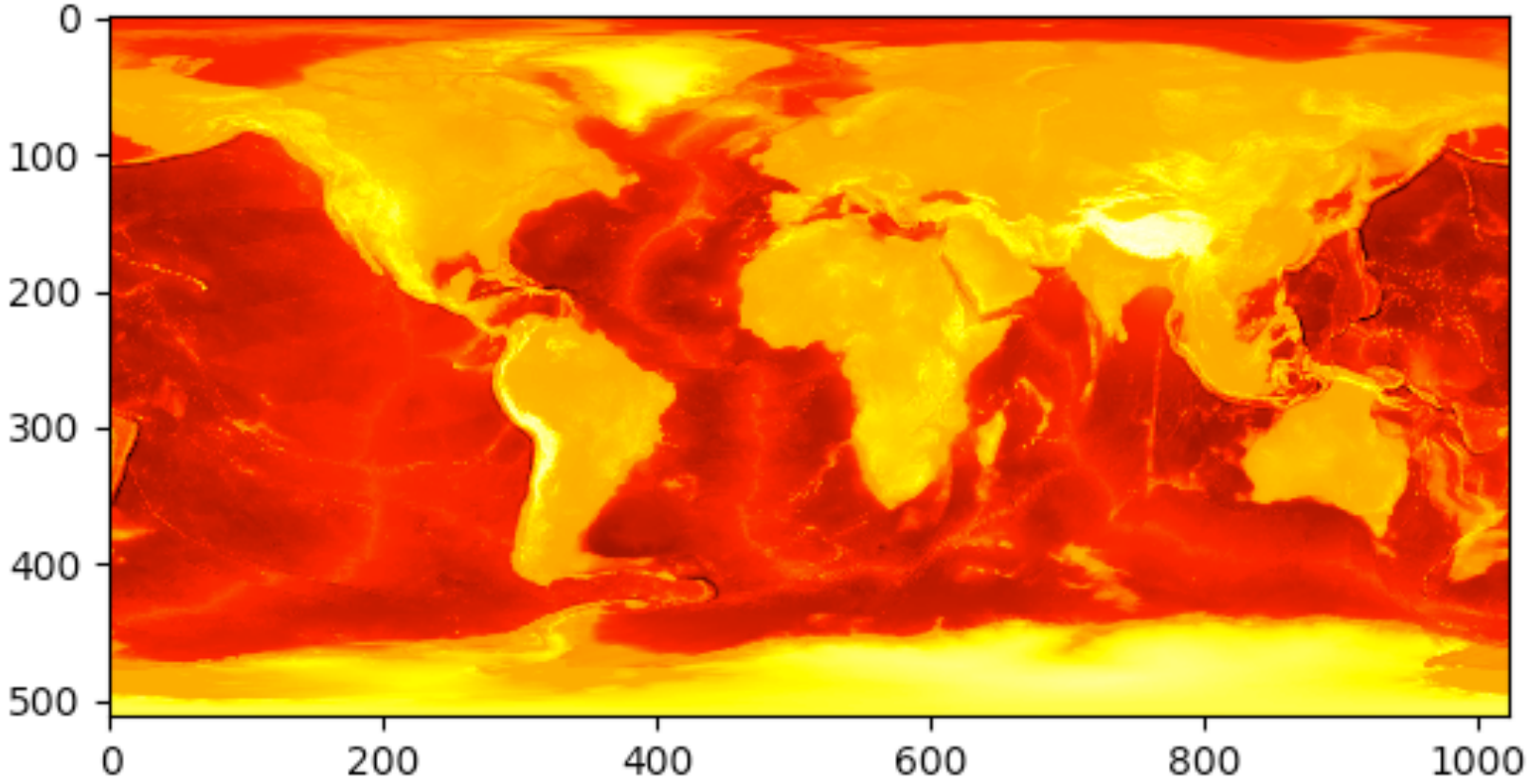
$$\begin{bmatrix} 0,0 & 1,0 & 2,0 & \cdots & N,0 \\ 0,1 & 1,1 & 2,1 & \cdots & N,1 \\ 0,2 & 1,2 & 2,2 & \cdots & N,2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0,M & 1,M & 2,M & \cdots & N,M \end{bmatrix}$$

# Exemplo 8: `imshow(x,y)`

altitude.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplinas/Fis...

```
from pylab import imshow, show, hot
from numpy import loadtxt

data = loadtxt("altitude.txt",float)
imshow(data)
hot()
show()
```

Nos eixos apresentam-se as quantidades de dados horizontais e verticais (N=1024 e M=512).
Existem vários conjuntos de cores predefinidos: jet, gray, hot, spectral, bone, hsv, inversegray, redwhiteblue, etc.



# Exemplo 9: `imshow(x,y)`

circular.py - /Users/kaline/Dropbox/Doc-Kaline/IFUSP/Disciplinas/Fis...

```
from pylab import imshow, show, gray
from numpy import loadtxt

data = loadtxt("circular.txt",float)
imshow(data,origin="lower")
gray()
show()
```

A opção `origin="lower"` é usada para colocar a origem no canto inferior esquerdo.

Figure 1

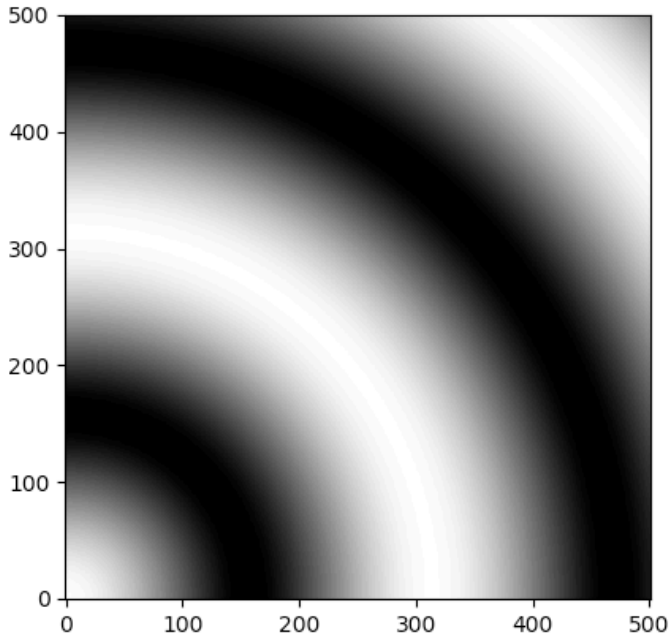
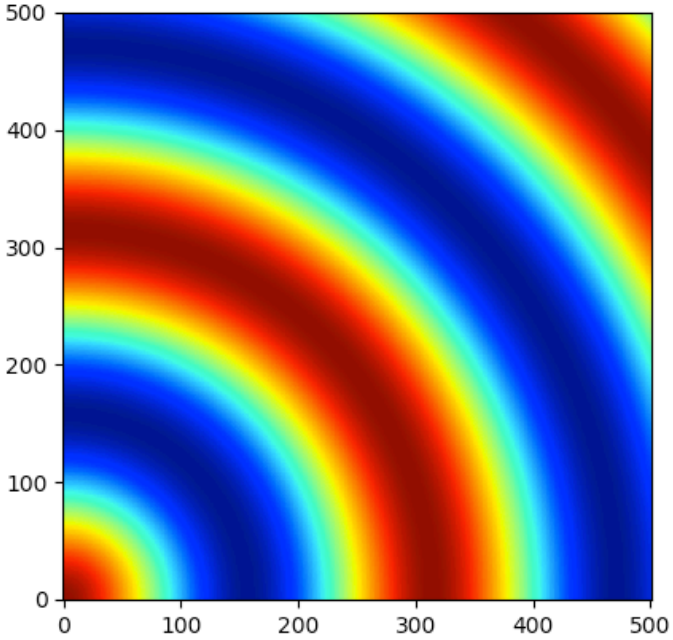



Figure 1



# Exercício:

- Fazer o download do arquivo Dados.zip (do moodle), selecionar 4 arquivos de dados e realizar 4 gráficos de linha, símbolo, histograma e densidade com programas em python usando as funções `plot()`, `scatter()`, `hist()` e `imshow()`, respectivamente.