FÍSICA COMPUTACIONAL 1

GRÁFICOS E VISUALIZAÇÃO

Pacote pylab

Contém diversas funções que podem ser usadas para gerar gráficos. Lembre de importar as funções antes de usá-las. Por exemplo, você pode usar: from pylab import *

plot, show

"Plot" é usado para gerar gráficos do tipo *x versus y*. Você deve fornecer duas listas (ou vetores) como argumento. **Ex:**

```
from pylab import plot,show
x = [0.5, 1.0, 2.0, 4.0, 7.0, 10.0]
y = [1.0, 2.4, 1.7, 0.3, 0.6, 1.8]
plot(x,y)
show()
```

Repare que apenas vimos o gráfico depois que usamos o comando show(). Como veremos, esta é uma característica útil, que nos permite modificar o gráfico inicial (inserindo legendas ou uma função adicional ao gráfico, por exemplo)

Linspace

Esta é uma função do pacote numpy, que é útil para gerar gráficos de funções. A função "linspace" gera um vetor contendo diversos pontos dentro de um intervalo. Sua sintaxe é "linspace(a,b,num)", onde *a* é o valor inicial e *b* é o valor final do intervalo, e *num* é o número de pontos que o vetor resultante deve conter. Este comando é usando no código que segue para gerar 100 valores entre 0 e 10. Estes pontos então são usados para gerar um gráfico da função seno. Repare que usamos a função seno do numpy para calcular o seno de todos elementos do vetor em uma linha.

```
from numpy import sin,linspace
from pylab import plot,show
x = linspace(0.0,10.0,100)
y = sin(x)
plot(x,y)
show()
```

Exercício 2.1: Gráficos de outras funções básicas

Repita o procedimento acima para obter gráficos das funções cosseno e raiz quadrada, com $0 < x < 10$.

Exercício 2.2: Plotando dados de um arquivo

Lembre que na aula passada vimos o comando loadtxt do pacote numpy. Use este comando para importar dados do arquivo "sunspots.txt" que está no SIGAA. Em seguida use o fatiamento para gerar um vetor só com os valores de x e outro só com valores de y. Por exemplo, se você importou os dados experimentais para um vetor de nome "dados", x= dados[:,0] gera um vetor com os dados da primeira coluna. Obtenha então um gráfico contendo os dados experimentais. Consulte o problema 3.1 para mais informações sobre estes dados.

xlim,ylim,xlabel,ylabel

Muitas vezes queremos rotular os eixos ou mudar os limites dos gráficos. Nestes casos, estas quatro funções são úteis. No exemplo abaixo, usamos a função "ylim" para mudar os limites do eixo y e usamos as funções "xlabel" e "ylabel" para adicionar rótulos.

```
from numpy import sin,linspace
from pylab import plot,show,ylim,xlabel,ylabel
x = linspace(0.0,10.0,100)
y = sin(x)
plot(x,y)
ylim (-1.1,1.1)
xlabel("eixo x")
ylabel("y = sen(x)")
show()
```

Exercício 2.3: Alterando a aparência da curva

Gere novamente os dados para plotar a função seno, mas desta vez use o comando "plot" com as seguintes modificações:

a) plot (x,y,"g- -")

b) plot (x,y,"ko")

Nos exemplos acima, a primeira letra indica a cor da curva. As letras permitidas são r, g, b, c, m, y, k, w, que correspondem as cores vermelha, verde, azul, ciano, magenta, amarelo, preto e branco. As opções mais comuns para a segunda letra são "-" para uma curva contínua, "- -" para uma curva tracejada, "o" para usar círculos para marcar os pontos e "s" para usar quadrados para marcar os pontos.

Exercício 2.4: Plotando dois gráficos ao mesmo tempo

Use o linspace para gerar uma sequência de dados, e em seguida use o comando "plot" para gerar um gráfico do seno e um segundo "plot" para gerar um gráfico do cosseno. Só então use o comando show(). Repare que as duas funções se encontram no mesmo gráfico. Tente usar as opções da função "plot" discutidas no exercício anterior para diferenciar as duas curvas.

scatter

"scatter" é usado para gerar gráficos de dispersão. Assim como a função "plot", ele recebe duas listas (ou vetores) como argumento. Também é necessário usar a função "show" para mostrar o resultado.

Exercício 2.5: Diagrama HR

Baixe do SIGAA o arquivo 'stars.txt'. A primeira coluna deste arquivo contém valores de temperatura e a segunda de magnitude. Estes dados podem ser usados para plotar um diagrama HR, que é bastante usado em estudos de evolução estelar.

a) Importe o arquivo usando o loadtxt, use fatiamento para obter separar as colunas x e y e faça um gráfico dos dados obtidos usando o scatter.

b) Mude o limite da temperature e da magnitude. Use valores de T entre 0 e 130000 e valores de H entre -5 e 20. Também adicione legendas para os dois eixos. Faça um novo gráfico com o scatter.

c) Diagramas HR são tipicamente plotados "invertidos", com valores altos de T e H perto da origem e valores menores distantes. Para obter um gráfico deste tipo, inverta os limites de T e H usados anteriormente. Por exemplo, para a coordenada x use agora "xlim(13000,0)".

imshow

"imshow" é usado para gerar gráficos de densidade. Por exemplo, se medirmos como a temperatura varia ao longo da superfície de um objeto, obteremos para cada ponto de coordenada (x,y) um valor de temperatura. Poderíamos então indicar o valor da temperatura local em um gráfico 2D usando cores: vermelho poderia indicar regiões quentes e azul regiões frias, por exemplo. O resultado seria um gráfico de densidade. O input da função "imshow" é tipicamente uma grade 2D de dados. Vamos ver como usar esta função no exemplo abaixo.

Exercício 2.6: Gráfico de densidade

Baixe do SIGAA o arquivo 'circular.txt' e importe os dados usando o loadtxt. Observe que obtivemos uma matriz de dados. Cada valor da matriz indica o valor local da grandeza que se quer plotar (por exemplo, temperatura), e os índices i e j da matriz indicam a posição do ponto. O código abaixo gera um gráfico de densidade com este conjunto de dados.

```
from numpy import loadtxt
from pylab import imshow,show
dados = loadtxt("circular.txt")
imshow(dados)
show()
```

Note que a origem do sistema está no canto superior esquerdo. Para entender porque isto ocorre, lembre que em uma matriz esta é a posição em que se encontra o termo com $i = j = 0$. Da mesma forma, observe que a posição x de um ponto é dada pelo índice j, uma vez que pontos mais a direita em uma matriz possuem valores de j maiores. Similarmente, a posição y de um ponto é dada pelo índice i.

a) Altere a posição da origem, de modo que ela se localize no canto inferior esquerdo. Para isto, utilize o comando "imshow(dados,origin="lower")".

b) Modifique o padrão de cor do gráfico. Para fazer isto, digite uma das seguintes palavras depois de usar o comando "imshow()" e antes de usar o comando "show()': jet(), gray(), hot(), spectral(), bone(), hsv().

c) Até o momento, as posições x e y correspondem aos índices i e j de um ponto na matriz de dados. Para alterar isto, use o comando "imshow" com a seguinte modificação "imshow(dados,origin="lower",extent=[0,10,0,5])"

Pacote vpython

Contém diversas funções que podem ser usadas para gerar gráficos 3D e até animações. Para instalar este pacote no linux, digite 'pip install vpython' no terminal. No windows, digite este comando no Anaconda prompt. Se você está acompanhando o livro do Mark Newman, note que o pacote 'visual' que ele menciona foi substituído pelo 'vpython', e que a sintaxe de alguns comandos foi modificada.

sphere()

Como o próprio nome da função indica, ela é usada para desenhar esferas. No exemplo abaixo, esta função é usado para desenhar uma esfera verde de raio definido em uma dada posição:

```
from vpython import sphere,color,vector
sphere(radius=0.5,pos=vector(1.0,-0.2,0.0),color=color.green)
```

Como o nome indica, a opção 'radius' indica o raio da esfera desenhada. Similarmente, a opção 'pos' indica a posição da esfera. Note que devemos usar a função "vector" para definir as posições que são passadas ao vpython; esta funcão também deve ser importada do vpython antes de ser usada. Finalmente, a opção 'color' define a cor. Cores válidas incluem red, green, blue, magenta, yellow, cyan, black, white. Se quisermos mudar o raio, a posição ou a cor depois da definição inicial, basta modificarmos o código da seguinte maneira:

```
from vpython import sphere,color,vector
s=sphere(radius=0.5,pos=vector(1.0,-0.2,0.0),color=color.green)
s.color=color.white
s.radius=0.7
s.pos=vector(0.0,0.0,0.0)
```

canvas()

Permite definir parâmetros da janela em que são mostrados os objetos 3D desenhados pelo vpython.

```
from vpython import sphere,canvas,color
scene2 = canvas(width = 1000, height = 1000,background=color.white)
sphere(color=color.black)
```

As opções width e height definem a área da janela em que serão desenhados os objetos 3D. A cor de fundo é definida usando a opção background.

Exemplo: Visualizando uma rede atômica

O código abaixo desenha uma rede cúbica simples composta por um único tipo de átomo. O parâmetro L é usado para definir o número de repetições e o parâmetro R para definir o raio das esferas.

```
from vpython import sphere,vector,canvas, color
scene2 = canvas(title='Rede cúbica simples', width = 1000, height = 1000,
background = color.white)
L = 3
R = 0.4
for i in range(-L,L+1):
    for j in range(-L,L+1):
        for k in range(-L,L+1):
            s=sphere(pos=vector(i,j,k),radius=R, color = color.blue)
```

Exercício 2.7: Rede do cloreto de sódio

Um cristal de cloreto de sódio tem átomos de sódio e cloro arranjados em uma rede cúbica, só que os átomos de sódio e cloro se alternam na rede, de modo que cada sódio está cercado por seis cloros e cada cloro está cercado por seis sódios. Crie uma visualização da rede de cloreto de sódio usando duas cores diferentes para representar os dois tipos de átomos. Este exercício corresponde a letra (a) do problema 3.4.

rate()

A função "rate" é bastante útil quando se quer fazer animações usando o python. Vimos anteriormente que é possível mudar a posição de uma esfera já definida. Se esta mudança de posição for feita sequencialmente dentro de um laço, podemos obter uma animação. O porém é que o computador muda a posição das esferas muito rapidamente, de forma que não conseguimos observar a animação. A função rate resolve este problema. A sua sintaxe é "rate(x)". Ao usarmos este comando, a função "rate" vai fazer com que o programa espere $1/x$ segundos antes de passar para a linha seguinte. No exemplo abaixo, este comando é usado para gerar a animação de uma esfera que percorre um círculo.

Exemplo: Uma esfera móvel

```
from vpython import sphere,rate,vector
from math import cos,sin,pi
from numpy import arange
s = sphere(pos=vector(1,0,0),radius=0.1)
for theta in arange(0,10*pi,0.1):
    rate(30)
    x = cos(theta)
    y = sin(theta)
    s.pos = vector(x,y,0.0)
```