Observatório do Valongo / UFRJ
Métodos Computacionais da Astronomia 2020/1-PLE

Professor: Pedro da Silveira Ferreira

Estudo dirigido / Lista 3 (Parte 1)

Entrega: 14/01/2021

---

1. [Numpy – Índices e estruturas] Resolva os itens **b)** até **g)** utilizando as propriedades da indexação dos arrays. **a)** Leia o arquivo array.dat usando o comando np.loadtxt() e armazene o mesmo como "array_original" **b)** Defina um novo array como sendo igual a 3ª coluna do array_original, e um outro array como sendo a 5ª linha do array_original. **c)** Defina um novo array com com as colunas pares do array_ogirinal **d)** Defina um novo array com as linhas impares do array_original **e)** Defina um novo array com os números múltiplos de 5 do array_original. **f)** Defina um array com apenas os números positivos do array_original. **g)** Defina um novo array como sendo a transposta do array_original **h)** Defina um novo array como sendo um array achatado (apenas uma linha) do array_original usando reshape, ravel ou flatten (olhe na documentação do numpy imbutida nos links anteriores) **i)** Utilizando reshape sobre o array do item anterior, gere um array final com 12 linhas e 2 colunas. (1 ponto)

2. [Numpy – Funções] Resolva os itens a seguir utilizando as funções do numpy **a)** Calcule a média, desvio padrão, quadrado, raiz quadrada, seno, soma (dos itens de cada array) e soma acumulada (dos itens de cada array) dos arrays gerados na questão 1. **b)** usando dois arrays aleatórios criados com np.random.random(size=50), faça as operações de soma, divisão, multiplicação e produto escalar entre os arrays. (0.5 ponto)

3. [For com list comprehension] Resolva utilizando as funções do numpy e/ou math. Gere uma lista de valores da função seno variando de -2π até 2π em passos de $\pi/100$. Faça duas versões, uma usando a função np.arange e outra np.linspace para definir o range do for e o pi do np.pi ou math.pi, utilizando list comprehension. (0.5 ponto)

Aplicações na astronomia

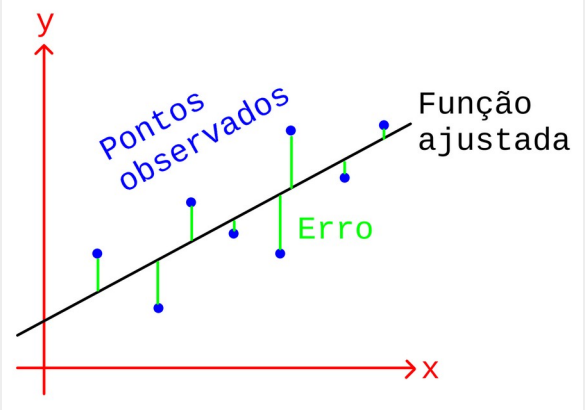
4. [Funções + Pandas + Algoritmo iterativo + Numpy] – **Medindo a constante de Hubble**
a) Utilizando o pandas, leia a tabela diagrama_de_Hubble.csv, que contém as velocidades de recessão e distâncias de diversas galáxias **b)** leia a mesma tabela com o numpy usando a função np.genfromtxt com o delimitador correto (geralmente ";","," , "\t" ou '\s+' em arquivos csv, veja a documentação para mais detalhes), perceba que para tabelas simples como essa o numpy já basta. **c)** Faça um plot dos pontos da tabela utilizando o matplotlib. **d)** Utilizando o método dos mínimos quadrados* (MMQ) obtenha a constante de Hubble de expansão do universo ajustando uma reta aos pontos (veja a referência do método em referência). Utilize MMQ via força bruta (testando diversos valores) com um loop for executado em um intervalo razoável (após inspeção do plot do item anterior) para os valores de a e b da reta, esse loop deve ser feito de forma que primeiro ocorra em passos mais largos (primeira iteração), encontre 3 pontos do intervalo com o menor erro e novamente subdivida esse intervalo em passos menores (próxima iteração), isso deve ocorrer o número de vezes o suficiente para que o a e b variem menos que 0.2% entre uma iteração e outra (múltiplas iterações). Crie a função que aplica o mmq a ser utilizada nos loops. **e)** Faça também o MMQ na forma analítica (utilizando as fórmulas analíticas fornecidas nos comentários, as linhas sobre os símbolos representam a média daquele valor), utilize as funções praticadas no exercício 2. No caso da linha reta o MMQ possui uma solução analítica exata, porém para casos mais complexos não, é necessário um código que utilize múltiplas iterações como feito no item d). Você encontrou um valor próximo ao valor encontrado na literatura para a constante de Hubble? (Referência para os valores) (1.5 pontos)

5. [Argparse + Astropy] **– Posição de astros em tempo real e mudança de coordenadas**
a) Utilizando o argparse e o astropy, faça um código em python que peça as coordenadas do observador (latitude, longitude e altitude) e o nome do objeto e retorne (print) as coordenadas do objeto no tempo atual (é livre a escolha da biblioteca para obter o tempo), em coordenas altazimutais e equatoriais. Veja este tutorial e este tutorial

para auxiliar com o astropy. **b)(extra)** Utilize o comando watch no terminal para que as coordenadas sejam atualizadas de 2 em 2 segundos, experimente com as coordenadas da lua. (1.5+0.25 pontos)

***Comentários:**

O MMQ é baseado na minimização dos quadrados dos erro (diferença entre os pontos observados e a reta). Na figura ao lado os erros são representados em verde. A equação abaixo é a forma "força bruta" do MMQ para a reta (que será a função ajustada).



$$f(x) = \sum_{pontos\ obs} \left(y^{observador} - y^{ajustado}\right)^2$$

$$f(x) = \sum_{pontos\ obs} \left(y^{observador} - a\,x - b\right)^2$$

As equações abaixo representam a forma analítica, exata, para o caso da reta:

$$a = \bar{y} - b\bar{x}$$

$$b = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

Esse é o método mais simples de ajuste de uma reta.

Observatório do Valongo / UFRJ
Métodos Computacionais da Astronomia 2020/1-PLE
Professor: Pedro da Silveira Ferreira

Estudo dirigido / Lista 3 (Parte 2)

Entrega: 19/01/2021

---

6. [Algorítimos recursivos vs Iterativos] Faça uma função recursiva para calcular todos os números primos até o numero n dado como input da função. (1 ponto)

Aplicações na astronomia

7. [Funções + Algoritmo iterativo + Matplotlib plot] – **Verificando o período de binárias eclipsantes**
a) Usando os arquivos "binaria_eclipsante_1.dat" e "binaria_eclipsante_2.dat" vamos medir o período de dois pares de binárias eclipsantes perfeitamente alinhadas e de mesmo brilho (situação idealizada). Nos arquivos as colunas são tempo em dias, brilho relativo e erro do brilho relativo, sendo o brilho relativo 0 o brilho relativo médio (renormalizado para 0), -1 quando apenas uma estrela é visível e 1 quando ambas estrelas estão visíveis (lado a lado). Veja este pequeno vídeo ilustrando um sistema binário eclipsante. Em ambos os casos, utilizando o método de minimização do χ^2 * (veja mais sobre o método nos comentários), um método um pouco mais sofisticado que o MMQ, e que é utilizado comumente na astronomia (em artigos). Tente ajustar uma curva do tipo $sen(\omega t)$ para encontrar o período do eclipse, onde $\omega = 2\pi/T$, sendo T o período em dias. **b)** Faça um plot com os pontos originais as barras de erro e a curva ajustada, utilize cores e marcadores diferentes para cada sistema binário, coloque os devidos títulos nos eixos do plot e inclua linhas verticais (gridlines) em cada ponto que marca a repetição do ciclo (período completo). Coloque os plots dos dois sistemas um sobre o outro formando uma única figura, as escalas nos eixos x precisam ser diferentes para uma boa visualização. Essas "sirenes" cósmicas, quando bem medidas, são muito úteis para a astronomia para medição de distância e outros parâmetros. **c)** Inclua uma fase na função seno ajustada e nos pontos ($t = t + fase$), utilizando o ipywidgets inclua um slider para ajustar a fase de cada plot. **Atenção: Crie seu próprio código para minimizar o χ^2, será parecido com o do MMQ da lista anterior.** (4 pontos)

***Comentários:**
1- Na questão 1 use a função $sen(\omega t)$ na equação do χ^2. Minimize o χ^2 variando ω.

Sobre o método do χ^2: Ele difere do MMQ pois leva em consideração o erro da medida (σ), nós dividimos pelo erro de forma que ele serve como um peso para qualidade da medida daquele ponto, dessa forma a grandeza a ser minimizada no loops será:

$$\chi^2 = \sum_{pontos\ obs} \left(\frac{y^{observador} - y^{ajustado}}{\sigma} \right)^2$$
