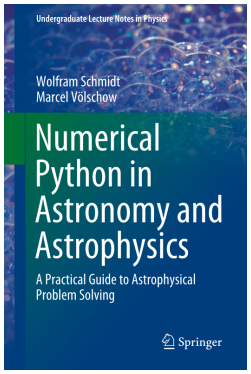
Resolução de Problemas do Livro

Numerical Python in stronomy and Astrophysics: A practical guide to astrophysical problem solving (Schmidt, W.; Völschow, M)



por

Igo da Costa Andrade

Referência

SCHMIDT, W.; VÖLSCHOW, M. **Numerical Python in stronomy and Astrophysics**: A practical guide to astrophysical problem solving. Switzerland, Springer, 2021.

Capítulo 2: Computação e exibição de dados[1]

Exercícios

2.1 Compute a declinação do Sol para os equinócios e solstícios usando apenas funções trigonométricas do módulo `math` em um loop `for` explícito. Imprima os resultados e cheque se eles concordam com os valores computados usando NumPy nesta seção. Este exercício ajuda a compreender o que está por trás de um loop impícito.

Solução:

Variação anual da declinação do sol:

$$\delta_\odot = -\arcsin\left[\sin\epsilon_0 \cos\left(\frac{360^\circ}{365.24}\right)(N+10)\right]$$

```
import math
from datetime import datetime, timedelta

year = 2024
jan1st = datetime(year,1,1)
eventos = [
  {
    "nome": "Equinócio de outono",
    "data": datetime(year,3,20)
  },
  {
    "nome": "Solstício de inverno",
    "data": datetime(year,6,20)
  },
  {
```

[1] Título original: *Computing and Displaying Data*.

```
    "nome": "Equinócio de primavera",
    "data": datetime(year,9,22)
  },
  {
    "nome": "Solstício de verão",
    "data": datetime(year,12,21)
  }
]


omega = 2*math.pi/365.24
ecl = math.radians(23.44)
for evento in eventos:
  diff = evento['data'] - jan1st
  N = diff.days
  delta = -math.asin(math.sin(ecl) * math.cos(omega * (N+10)))
  print(f"{evento['nome']} ({evento['data'].strftime('%d/%m/%Y')})")
  print(f"Declinação = {math.degrees(delta):.2f} deg.")
```

```
## Equinócio de outono (20/03/2024)
## Declinação = -0.91 deg.
## Solstício de inverno (20/06/2024)
## Declinação = 23.43 deg.
## Equinócio de primavera (22/09/2024)
## Declinação = -0.42 deg.
## Solstício de verão (21/12/2024)
## Declinação = -23.44 deg.
```

■

2.2 O contador do dia N na Eq. (2.1) pode ser calculado para uma data com a ajuda do módulo `datetime`. Por exemplo, o dia de equinócio vernal mp ano de 2020 é dado por

```
import datetime
vernal_equinox = datetime.date(2020, 3, 20) - \
                 datetime.date(2020, 1,  1)
```

Então `vernal_equinox.days` resulta em 79. Defina um array `N` (equinócios e solstícios) usando datetime.

Solução:

```
import datetime
import numpy as np

N = np.array([
  (datetime.date(2024, m, d)-datetime.date(2024,1,1)).days for m, d \
    in zip([3, 6, 9, 12],[20, 20, 22, 21])
])
print(N)
```

```
## [ 79 171 265 355]
```

2.3 Uma fórmula mais precisa para a declinação do Sol leva em consideração a excentricidade $e = 0.0167$ da órbita da Terra:

$$\delta_\odot = -\arcsin\left[\sin(\epsilon_0)\cos\left(\frac{360^\circ}{365.24}(N+10) + e\frac{360^\circ}{\pi}\sin\left(\frac{360^\circ}{365.24}(N-2)\right)\right)\right]$$

Calcule a declinação assumindo uma órbita circular (Eq. 2.1), a declinação resultante da fórmula acima, a diferença entre esses valores, e o desvio relativo da aproximação circular em % para equinócios e solstícios e liste seus resultados em uma tabela. Certifique-se de que um número adequado de dígitos seja exibido para comparar as fórmulas.

Solução:

```
# Bibliotecas necessárias
import math
import numpy as np
import datetime

# Função auxiliar
def sun_declination(N, ecl, omega, e=0.0):
  delta = -np.arcsin(
    math.sin(ecl) * np.cos(omega*(N+10) + 2*e*np.sin(omega*(N-2)))
  )
  return delta

# Dados do problema
ecl = math.radians(23.44)
omega = 2*math.pi/365.24
e=0.0167

year = 2024
jan1st = datetime.date(year,1,1)
eventos = [
  {"nome": "Equinócio de outono", "data": datetime.date(year,3,20)},
  {"nome": "Solstício de inverno", "data": datetime.date(year,6,20)},
  {"nome": "Equinócio de primavera", "data": datetime.date(year,9,22)},
  {"nome": "Solstício de verão", "data": datetime.date(year,12,21)}
]
tbl_header = f"{f'Evento ({year})':<30s} {'Decl. (Órb. Circular)':<20s} {'Decl. (Órb. Elíptica)':<2

tbl = "="*len(tbl_header) + "\n"
tbl += tbl_header + "\n"
tbl += "-"*len(tbl_header) + "\n"

for evento in eventos:
  N = (evento['data'] - jan1st).days
  delta_circ = sun_declination(N=N, ecl=ecl, omega=omega, e=0.0)
  delta_elipse = sun_declination(N=N, ecl=ecl, omega=omega, e=e)
  erro = (delta_circ-delta_elipse)/delta_elipse
  row = f"""{f"{evento['nome']} ({evento['data'].strftime('%d/%m')})":<30s} """ + f"{math.degrees(de
```

```
        f"{math.degrees(delta_elipse):>20.2f}° " + f"{erro:>10.2%}"
  tbl += row + "\n"
tbl += "="*len(tbl_header)


print(tbl)
```

```
## ================================================================================================
## Evento (2024)                  Decl. (Órb. Circular) Decl. (Órb. Elíptica)   Erro (%)
## ------------------------------------------------------------------------------------------------
## Equinócio de outono (20/03)                  -0.91°                -0.17°    440.87%
## Solstício de inverno (20/06)                 23.43°                23.43°     -0.02%
## Equinócio de primavera (22/09)               -0.42°                 0.33°   -227.79%
## Solstício de verão (21/12)                  -23.44°               -23.44°      0.01%
## ================================================================================================
```