Numeriske metoder (SO6: matematik-programmering)

Jonas Camillus Jeppesen, jcje@sde.dk

24. januar 2019

Matematiske beregninger i Python

Du har sikkert allerede prøvet at bruge f.eks. `sqrt()` funktionen fra Pythons `math` modul/bibliotek. I projektet her får du imidlertid brug for andre andre matematiske beregninger end kvadratroden. Dokumentationen for `math` modulet fortæller om dets forskellige matematiske funktioner: `https://docs.python.org/3/library/math.html`.

```python
from math import sqrt
print(sqrt(4))
```

Kildekode 1: Udregning af kvadratroden i Python.

Øvelse 1. *Undersøg i dokumentationen hvordan Python kan runde tal op (engelsk: ceiling) og runde tal ned (engelsk: flooring). Skriv et program runder følgende tal både op og ned: 5.7, 1.3, -5.7, -1.3.*

Øvelse 2. *Undersøg i dokumentationen hvordan Python kan udregne e^x, Eulers tal opløftet i x'te potens (engelsk: power x). Skriv et program som udregner: e^0, e^1 og e^{10}.*

Øvelse 3. *Fra matematik ved vi, at $\log_{10}(100) = 2$, $\ln(e^2) = 2$ og $\log_2(8) = 3$. $\log_{10}$, $\ln$ og $\log_2$ er logaritmefunktioner med forskellige baser (base 10, e og 2). Undersøg i dokumentationen hvordan Python kan udregne logaritmefunktionen med forskellige baser. Skrive et program som verificerer hvad der påstås i starten af øvelsen og som udregner: $\log_{10}(1000)$, $\log_2(256)$, $\ln(54.598)$.*

Øvelse 4. *Undersøg i dokumentationen hvordan Python kan beregner sinus (engelsk: sine), cosinus (engelsk: cosine), sinusinvers (engelsk: arc sine) og cosinusinvers (engelsk: arc cosine). Find ud af om funktionerne forventer input i radianer eller grader. Skriv et program som udregner højden af en rampe som danner vinkel $\alpha = 27.5°$ med vandret og som er $L = 7\,\mathrm{m}$ lang.*

Funktioner og lister Python

Du får i projektet brug for en solid viden om funktioner og lister i Python. Husk, at en funktion i Python defineres og kaldes på følgende måde:

```python
def myfun(a,b):
    # Do something useful
    # result =
    return result

r =  myfun(2,4)
print(r)
```

Kildekode 2: Definition og kald af en funktion i Python.

> **Øvelse 5.** *Skriv en funktion som tager to argumenter* `h` *og* `r`*, og returnerer volumenet af cylinderen med højde h og radius r.*

> **Øvelse 6.** *Skriv en funktion som tager argumenter* `x`, `a`, `b` *og* `c` *og returnerer $f(x) = ax^2 + bx + c$ (et enkelt reelt tal).*

Husk, at liste i Python kan bruges på følgende måder:

```python
# Opret liste
xs = list(range(0,4))

# Manuel oprettelse
# xs = [0,1,2,3]

# Indsæt elementer i liste
xs.append(4)
xs.insert(2,1.5)

# Tilgå elementer i en liste
print(xs[3])

# Gennemløb en liste
for x in xs:
    print(2*x)
```

Kildekode 3: Forskellige operationer på en listeoperationer i Python.

> **Øvelse 7.** *Skriv et program som konstruere en liste bestående af tallene x^2 for $x = 1, 2, 3, \ldots, 19, 20$.*

Python operatoren % bruges til at udregne resten der er tilbage efter heltalsdivision. Hvis man eksempelvis dividerer 6 med 4 er der en rest på 2 fordi $4 \cdot 1 + 2 = 6$. Dividerer man 9 med 2 er der en rest på 1 da $4 \cdot 2 + 1 = 9$.

```
r = 9%2

# Prints 1
print(r)
```

Kildekode 4: Eksempel på modulus operation i Python.

Øvelse 8. *Skriv et program som konstruerer en liste bestående af kun ulige heltal. Hint: brug* % *operatoren.*

Øvelse 9. *Skriv et program som konstruerer en liste bestående af kun lige heltal. Hint: brug* % *operatoren.*

Nu hvor du er blevet mindet om hvad man kan med lister så er det tid til at se på funktioner som tager lister som argumenter.

Øvelse 10. *Skriv en funktion som tager en liste af tal* `xs` *som argument og returnere en liste der indeholder værdierne* $f(x) = 2x^2 + 2$ *for hvert* `x` *i argumentlisten* `xs`*.*

Øvelse 11. *Skriv en funktion som udregner funktionen* $f(x) = p_0 + p_1 x + p_2 x^2 + \ldots + p_{n-1} x^{n-1} + p_n x^n$ *for hvert* `x` *i en liste* `xs`*.* p_0*,* p_1 *osv. er koefficienter funktionen også skal acceptere som input. Hint: Brug to lister som argumenter og brug to for-loop.*

Plot i Python med matplotlib

Nu hvor du kan bruge Python til at udregne matematiske funktionsværdier skal du lære plotte punkter og tegne diagrammer i Python.

Modulet matplotlib kan bruges til at plotte data og visualisere data i Python. Det installeres fra kommandolinjen vha. pip-værktøjet:

```
$ pip install matplotlib
Collecting matplotlib
[output removed to save space]
Installing collected packages: kiwisolver, numpy, pyparsing, six, cycler, python-dateutil, matplotlib
Successfully installed cycler-0.10.0 kiwisolver-1.0.1 matplotlib-3.0.2 numpy-1.16.0 pyparsing-2.3.1 python-dateutil-2.7.5 six-1.12.0
```

Kommandolinje 1: Installation af matplotlib vha. pip. Hvis du bruger macOS skal du bruge `pip3` i stedet.

```python
import matplotlib.pyplot as plt

xs = [1, 2, 3, 4, 5, 6]
ys = [1, 4, 9, 16, 25, 36]

plt.plot(xs,ys, 'b*')
plt.xlabel("x")
plt.ylabel("y=x^2")
plt.title("Plot of y=x^2")
plt.show()
```

Kildekode 5: Et meget simpelt plot vha. matplotlib.

Øvelse 12. *Prøv koden fra Kildekode 5.*

Øvelse 13. *I stedet for* `'b*'` *i Kildekode 5 prøv* `'r^'`, `'r^--'` *og* `'go-'`.

Øvelse 14. *Skriv et program som plotter funktionen* $f(x) = 2x^3 - 3x^2 + 5$ *i intervallet* $x \in [-10; 10]$. *Hint: Brug det du har lært i afsnittet om lister og funktioner.*

Øvelse 15. *Skriv en funktion som tager tre argumenter; reele tal* a *og* b, *samt et positivt heltal* n. *Funktionen skal returnere en liste af* n *tal, ligeligt fordelt i intervallet* $[a; b]$. *Kald funktionen for* `linspace(a,b,n)`, *den er nyttig når man skal plotte funktioner.*

Øvelse 16. *Skriv et program som plotter* $f(x) = 2x + 5$ *og* $g(x) = x^2 - 2$ *i samme koordinatsystem i to forskellige farver. Hint: Kald* `plt.plot()` *to gange, en til hver funktion.*

Hvis en figur indeholder mere end en graf eller et plot så skal den have en signaturforklaring [1] for at være læselig. Kildekode 6 viser hvordan en signaturforklaring tilføjes en figur.

[1] engelsk: legend

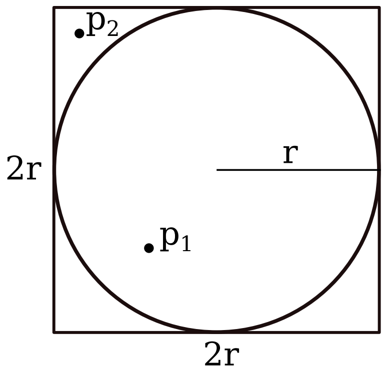
Kildekode 6: Figur med signaturforklaring.

```python
import matplotlib.pyplot as plt

xs = [1, 2, 3, 4, 5, 6]
ys = [1, 4, 9, 16, 25, 36]

plt.plot(xs,ys, 'go-', label="(x,y)")
plt.plot(ys,xs, 'r*--', label="(y,x)")

plt.xlabel("x")
plt.ylabel("y")
plt.title("Good figures have legends")
plt.legend()
plt.show()
```

Øvelse 17. *Tilføj en signaturforklaring til figuren fra Øvelse 16.*

Numeriske metoder

I dette afsnit skal vi se på nogle numeriske metoder til at løse matematiske problemer der ikke har eksakte løsninger.



Figur 1: En cirkel med radius r indskrevet i et kvadrat med sidelængder $L = 2r$.

Estimering af π

π kan estimeres på en computer vha. en såkaldt Monte Carlo metode. Betragt kvadratet og cirklen på figur 1. Forholdet cirklens areal og kvadratets areal er:

$$\frac{A_{circ}}{A_{sqr}} = \frac{\pi r^2}{2r \cdot 2r} = \frac{\pi r^2}{4r^2} = \frac{\pi}{4} \tag{1}$$

Hvis en computer kunne finde disse to arealer, uden at bruge π til beregning af cirklens areal, kan man altså få estimeret π som:

$$\pi = \frac{4A_{circ}}{A_{sqr}} \tag{2}$$

En måde at estimere de to arealer er ved generere tilfældige punter som ligger inden for kvadratet, som $p_1 = (x_1, y_1)$ og $p_2 = (x_2, y_2)$ på figur 1.

Nogle af disse punkter vil ligge inden for cirklen og andre udenfor (men indenfor kvadratet). Arealforholdet kan nu estimeres som forholdet mellem antal punkter inden for cirklen og antal punkter genereret i alt. Dermed kan π estimeres som

$$\pi = \frac{4A_{circ}}{A_{sqr}} \approx \frac{4n_{in}}{n_{total}} \tag{3}$$

hvor n_{in} er antal punkter inden i cirklen og n_{total} er antal punkter generet i alt. Algoritme 2 viser algoritmen i pseudokode.

```
function EstimatePi(n)
    i ← 0
    n_in ← 0
    while i < n do
        i ← i + 1
        x ← random real number from [−r, r]
        y ← random real number from [−r, r]
        if (x, y) is inside circle then
            n_in ← n_in + 1
    return 4n_in / n
```

Algoritme 1: Algoritme til at estimere π vha. en Monte Carlo metode

Øvelse 18. *Skriv et Python program som estimerer π vha. Algoritme 2. Programmet skal til sidst også plotte de genererede punkter vha. matplotlib, således punkterne inden for cirklen er røde og uden for cirklen er blå. Hint: Brug* `import random`.

Ligningsløsning med halveringsmetoden (binærsøgning)

At løse ligninger af typen $f(x) = 0$ kaldes også at finde rødder for funktionen $f(x)$ hvis $f(x)$ er et polynomium, ellers kaldes det at finde nulpunkter. Stort set alle ligninger kan omformuleres til at være et rod- eller nulpunktsproblem:

$$f(x) = 5 \implies f(x) - 5 = 0 \tag{4}$$

$$f(x) = g(x) \implies f(x) - g(x) = 0 \tag{5}$$

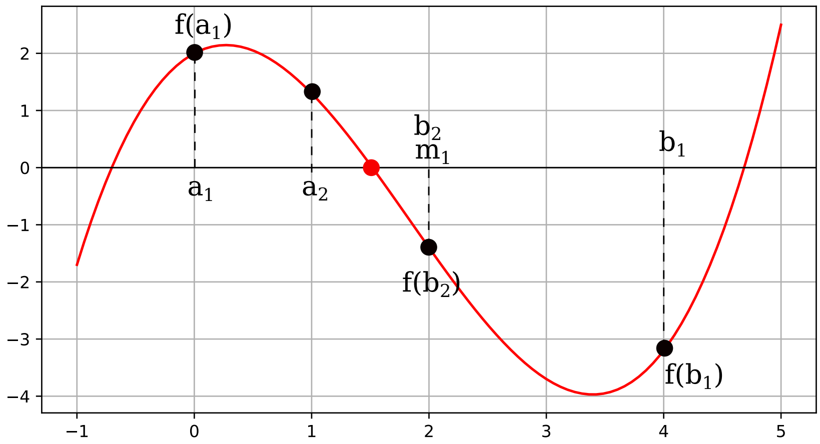
En ligning som $\sin(x) = x$ kan ikke løses analytisk, men den kan løses numerisk ved at finde nulpunkter for $f(x) = \sin(x) - x$.

Halveringsmetoden[2] er en numerisk metode til at finde nulpunkter.

[2] engelsk: bisection method

Figur 2 illustrerer hvordan halveringsmetoden virker. Først vælges et interval $[a_1; b_1]$ således funktionens nulpunkt ligger i intervallet. Herefter udregnes intervallets midtpunkt $m_1 = \frac{a+b}{2}$. m er nu vores første bud på nulpunktet for $f(x)$. Er vi ikke tilfredse med vores bud så indsnævrer vi intervallet med et nyt interval som er mindre; enten $[a_1, m_1]$ eller $[m_1, b_1]$. Et af de to intervaller indeholder med sikkerhed vores nulpunkt.

Øvelse 19. *Find (det matematiske) kriterier for hvilket af de to intervaller $[a_1, m_1]$ og $[m_1, b_1]$ nulpunktet befinder sig i*



Figur 2: Halveringsmetoden illustreret for funtionen $f(x) = 0.4x^3 - 2.2x^2 + 1.1x + 2$

I problemet på figuren befinder nulpunktet sig i det nye interval $[a_1; m_1]$. Vi kan nu finde midtpunktet $m2 = \frac{a_1+m_1}{2}$ af det nye interval og det giver os et nyt bedste bud på nulpunktet. Algoritme **??** beskriver algoritmen i pseudokode.

function BisectionMethod(f,a,b,nmax,tol)
    $i \leftarrow 0$
    $n_{in} \leftarrow 0$
    while $i < n$ **do**
        $i \leftarrow i + 1$
        $x \leftarrow$ random real number from $[-r, r]$
        $y \leftarrow$ random real number from $[-r, r]$
        if (x, y) is inside circle **then**
            $n_{in} \leftarrow n_{in} + 1$
    return $\frac{4n_{in}}{n}$

Algoritme 2: Algoritme til at estimere π vha. en Monte Carlo metode

Øvelse 20. *Skriv et Python program som kan udregne et nulpunkt vha. halveringsmetoden.*

Ekstraopgave 1. Skriv en funktion, der tager en liste af tal som argument, og returnerer det index i listen, hvor den største værdi findes. Hvis den største værdi optræder flere gange i listen, skal du returnere det første index.

Skriv en tilsvarende funktion, der returnerer det mindste tal.

Ekstraopgave 2. Skriv et python-program der

- genererer en liste med n ligeligt fordelte x-værdier mellem 0 og 100. (F.eks. med `linspace` fra Øvelse 15)
- genererer en liste med n tilfældige y-værdier.
- bruger `matplotlib` til at tegne et punktdiagram med de genererede punkter. Punktet med den største y-værdi skal være en anden farve end resten.

Ekstraopgave 3. Skriv et python-program der tager en liste med tal som argument, og returnerer gennemsnittet af tallene.

Ekstraopgave 4. Undersøg hvad koden herunder gør. Forklar indholdet af liste y.

```python
def f(x):
  return 3*x+4

x = range(0,100)
y = [f(v) for v in x]
```

Brug ovenstående kode til at generere en graf for funktionen `f`.