
# Monte Carlon menetelmä ja numeerinen integrointi Pythonilla

***Anne-Maria Ernvall-Hytönen***
Helsingin yliopisto

Monte Carlon menetelmä on oikeastaan menetelmäperhe. Ideana on toistaa satunnaista koetta ja tämän satunnaisen kokeen perusteella tehdä johtopäätöksiä yleisestä tilanteesta. Tässä tekstissä Monte Carlon menetelmää on tarkoitus valottaa numeerisen integroinnin näkökulmasta, eli pyritään käytännössä laskemaan erilaisia pinta-aloja Monte Carlon simuloinnilla.

## Tämän kevään 2021 ylioppilaskokeet ja Monte Carlo

Tämän kevään ylioppilaskokeissa tehtävän 8 tehtävänanto oli seuraava:

Tasojoukon $A$ pisteet $(x, y)$ määräytyvät epäyhtälöistä $0 \leq x \leq 2$, $0 \leq y \leq 4$ ja $y \geq x^2$. Tässä tehtävässä on tarkoitus arvioida joukon $A$ pinta-alaa simulaation avulla käyttämällä sitä tietoa, että todennäköisyys on suoraan verrannollinen pinta-alaan. Arvotaan pisteitä $(x, y)$ suorakulmiosta $B$, jonka määräävät epäyhtälöt $0 \leq x \leq 2$ ja $0 \leq y \leq 4$.

1. Tee sopivalla ohjelmistolla koodi, joka arpoo 1000 pistettä suorakulmiosta $B$ ja tulostaa vastauksena niiden pisteiden lukumäärän, jotka kuuluvat joukkoon $A$. Kerro sanallisesti ja sopivien kuvakaappausten avulla, miten toteutit koodisi. (Vihje: Voit käyttää esimerkiksi taulukkolaskennan satunnaislukugeneraattoria.) (6 p.)
2. Hille ajoi kohdassa 1 tekemänsä koodin 10 kertaa ja sai alla olevat luvut. Laske tulosten keskiarvo ja arvioi tämän perusteella joukon $A$ pinta-alaa. (6 p.)

   Hillen koodin tulosteet: 673, 664, 672, 679, 667, 650, 640, 678, 660, 667.

Tehtävän ensimmäisessä osassa on siis tarkoitus tuottaa algoritmi, joka arpoo pisteitä ja kertoo miten moni niistä kuuluu tiettyyn alueeseen. Toisessa kohdassa taas ajatellaan kuvitteellista tilannetta, jossa Hilleniminen henkilö on tehnyt koodin ja ajanut sen kymmenen kertaa ja saanut kasan tulosteita. Näiden tulosteiden perusteella olisi tarkoitus arvioida alueen pinta-alaa lähtien liikkeelle oletuksesta, joka voidaan muotoilla esimerkiksi seuraavasti: jos kysytty ala on $x$ % koko pinta-alasta, niin noin $x$ % arvotuista pisteistä osuu kyseiseen alaan.

Koko tehtävä ratkeaa sujuvasti ilman Pythonin käyttöä, esimerkiksi ihan tavallisella taulukkolaskentaohjelmalla, mutta koska tämän tekstin ideana on antaa esimerkkejä nimenomaan Pythonin käytöstä, keskitytään siihen.

YTL:n hyvän vastauksen piirteissä on tarjottu oheinen Pythonilla ohjelmoitu koodi yhdeksi mahdolliseksi ratkaisuksi:

```
import random
n=1
k=0
while n<1001:
    x=random.uniform(0,2)
    y=random.uniform(0,4)
    if x*x<=y:
        k=k+1
    n=n+1
print(k)
```

Yksinkertaisuudessaan siis koodi arpoo 1000 kertaa luvut $x$ ja $y$ annetulta väliltä ja testaa joka kerta, onko $x^2 \leq y$ vai ei. Jos $x^2 \leq y$, niin kasvatetaan laskuria, joka määrittää, kuinka moni pari toteuttaa tämän ehdon. (Eli pidetään kirjaa siitä, kuinka moni lukupari toteuttaa annetun ehdon.) Lopuksi tulostetaan niiden lukuparien $(x, y)$ lukumäärä, jotka toteuttavat annetun ehdon.

Jälkimmäisessä kohdassa ajatus on se, että annettujen lukujen keskiarvo antaa jonkinlaisen estimaatin siitä, kuinka monta pistettä voisi osua annettuun alueeseen. Keskiarvo on 665, joten $\frac{665}{1000}$ osuus lukupareista osuu annettuun alueeseen. Koska pisteitä arvotaan suorakulmiosta, jonka mitat ovat $2 \times 4$, eli ala on 8, on kysytyn alueen pinta-ala arviolta

$$\frac{665}{1000} \cdot 8 \approx 5{,}3.$$

Integroiden pinta-alan tarkaksi arvoksi saadaan

$$\int_0^2 (4 - x^2)dx = \left[4x - \frac{1}{3}x^3\right]_0^2 = 8 - \frac{8}{3} = \frac{16}{3} \approx 5{,}3.$$

Yhden desimaalin tarkkuudella tulokset siis ovat samat.

## Piin likiarvo simuloimalla

Arvioidaan seuraavaksi luvun $\pi$ likiarvo samalla idealla. Piirretään yksikköympyrä $2 \times 2$-neliöön ja selvitetään, kuinka suuri osa neliöstä kuuluu ympyrään. Yksikköympyrän ala on $\pi$, joten periaatteessa pisteistä suurin piirtein $\frac{\pi}{4}$-osan pitäisi osua siihen. Koodi näyttää tältä:

```
import random
n=1
k=0
while n<1001:
    x=random.uniform(-1,1)
    y=random.uniform(-1,1)
    if x*x+y*y<=1:
        k=k+1
    n=n+1
print(k)
```

Ajoin koodin 11 kertaa, jokaisella kerralla 1000 lukuparia, ja sain tulosteiksi: 790, 765, 778, 797, 748, 757, 802, 786, 789, 782, 768. Näiden lukujen keskiarvo on $\frac{8562}{11}$. Nyt siis voidaan arvioida, että

$$\frac{8562}{11 \cdot 1000} \approx \frac{\pi}{4},$$

eli

$$\pi \approx 4 \cdot \frac{8562}{11 \cdot 1000} \approx 3{,}11.$$

Ensimmäinen desimaali on siis oikein, toinen ei. Muokkasin ohjelmaa hiukan: tehdäänkin $10^6$ toistoa, mutta ajetaan ohjelma vain kerran. Tuloste on nyt 786444, eli tällä kerralla saadaan

$$\pi \approx 4 \cdot \frac{786444}{10^6} \approx 3{,}146.$$

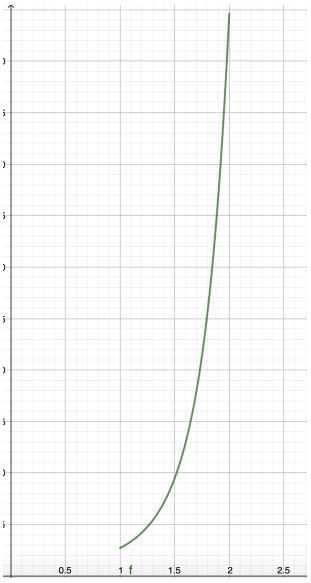
Jos koodia ajettaisiin hyvin monta kertaa ja laskettaisiin keskiarvoja tai otettaisiin hyvin iso otos, saataisiin parempia ja parempia likiarvoja luvulle $\pi$.

## Numeerinen integraali ja hankala integraali

Arvioidaan viimeiseksi integraalia

$$\int_1^2 e^{x^2} dx.$$

Funktio $x \to e^{x^2}$ ei ole integroitavissa alkeisfunktioiden avulla. Jos syötän Wolfram Alphaan komennon `integrate e^(x^2)`, niin Wolfram Alphan vastaus on $\frac{1}{2}\sqrt{\pi}\text{erfi}(x)$ + vakio. Funktiota erfi ei Wolfram Alpha linkkaa mihinkään sen syvempään, mutta Wolfram Alpha osaa liittää tämän funktion Dawsonin integraaliin tai funktioon ja kertoa, että ne liittyvät puolestaan lämmönjohtumiseen ja että niillä on muitakin sovelluksia. Toteutetaan nyt ohjelma, jolla saamme arvioitua integraalia. Huomaamme ensinnäkin, että kun $x = 1$, on funktion $e^{x^2}$ arvo $e$. Kun $x = 2$, on funktion arvo $e^4$. Joudumme siis operoimaan suorakulmiossa, joka saa $x$-akselilla arvot $[1, 2]$ ja $y$-akselille arvot $[0, e^4]$. Kuva tilanteesta näyttää tältä:




Tavoite on siis määrittää käyrän ja $x$-akselin väliin jäävän alueen pinta-ala. Kirjoitetaan koodi samalla periaatteella kuin aiemminkin. Nyt arvotaan $10^4$ pisteparia $(x, y)$:

```
import math
import random
k=0
n=1
while n<10001:
    x=random.uniform(1,2)
    y=random.uniform(0,math.exp(4))
    if y<=math.exp(x**2):
        k=k+1
    n=n+1
print(k)
```

Ajoin koodin kertaalleen ja sain tulokseksi 2723. Nyt integraalin arvio olisi siis

$$\int_1^2 e^{x^2}\,dx \approx \frac{2723}{10^4} \cdot 1 \cdot e^4 \approx 14{,}87.$$

Integraalille Wolfram Alpha antaa puolestaan arvoksi

$$\int_1^2 e^{x^2}\,dx \approx 14{,}99.$$

Virhe on tässäkin tapauksessa suhteellisen pieni ja tietysti tarkemman arvion saisi tälläkin kertaa joko lisäämällä toistojen määrää tai ajamalla ohjelman useampia kertoja.

## Lopuksi

Monte Carlon menetelmä soveltuu monen integraalin arviointiin sekä esimerkiksi hypoteesien testaamiseen ja moneen muuhun tarkoitukseen. Se soveltuu myös erinomaisen hyvin todennäköisyyslaskentaan. Monia todennäköisyyksiä on helppo laskea hiukan väärin, mutta testaamalla vastaavalla simuloinnilla voi vähintäänkin testata onko kertaluokka täysin pielessä.

Alkuun Pythonin kanssa pääsee esimerkiksi osoitteessa

`https://ohjelmointi-21.mooc.fi/`

olevalla oppaalla. Pythonille on useita kääntäjiä verkossa. Esimerkiksi tämän tekstin koodit on ajettu osoitteessa

`https://www.programiz.com/python-programming/online-compiler/`

olevalla kääntäjällä.