1 Monte Carlo Metoden

I dette afsnit vil opgaven komme ind på...

Monte Carlo Metoden har sit navn efter Stanislaw Ulams onkel, som ofte spillede på Casino Monte Carlo i Monaco. Metoden referer herved til de sandsynligheder og de tilfældige udkom der kan finde sted i sådanne spil. Ulam var en af de forskere der arbejdede på Monte Carlo Metoden under Manhatten Projektet, som blev iværksat i 1942 [**MSD**]. Metoden blev udviklet for at kunne simulere neutroners diffusion. Det var nødvendigt at simulere på denne måde, da problemet var for komplekst til at kunne blive afledt algebragisk. Metoden blev først anvendt på gamle analoge computere, hvilket begrænsede kompleksiteten af simulationen [**AHF**].

Ydermere er Monte Carlo Metoden også en algoritme, det betyder at det er en process som er udført i nogle logiske trin,trin som beskriver udførelse af metoden. En algoritme kunne f.eks. være trinnene involveret i at sortere et sæt kort. Lad kortspillet være et sæt kort og et element værende ét enkelt spillekort, kortspillet kan således sorteres på følgende måde.

1. Vælg et vilkårligt element e_i fra sættet.
2. Placere elementet bagest i sættet.
3. Sammenlign nu e_i med elementet lige før e_{i-1} hvis $e_i > e_{i-1}$ bytter de plads. Dette gentages indtil $e_i < e_{i-1}$ eller $i = 0$.
4. Gentag denne process indtil kortspillet er sorteret

Denne sorterings algoritme er også kaldet boblesortering og dets navn kommer af måden de elementer i sættet bobler til toppen. [**BS**]

1.1 Plat eller Krone

Et eksempel på udførelse af Monte Carlo Simulering kunne være spillet Plat eller Krone, spillets regler er som følgende.

1. Hver spiller satser på en side af mønten.
2. Mønten kastes op i luften.
3. Vinderen er den spiller hvis sats er den side af mønten der vender op af.

Man kan også beskrive Plat og Krone med et symmetrisk sandsynlighedsfelt (U, P), hvor udfaldsrummet U ville være $U = \{Plat, Krone\}$ og sandsynlighedsfunktionen P hvor følgende er sandt da det er et symmetrisk sandsynlighedsfelt.

$$P(Plat) = P(Krone) = \frac{1}{2} \tag{1.1}$$

Vi kan desuden beskrive mønten med en stokastiske variable X. Den forventede værdi af X kan ligedes beskrives som værende $\mathbb{E}[X] = \frac{1+2}{2} = 1.5$ hvis $Plat = 1$ og $Krone = 2$ altså den gennemsnitlig værdi af udfaldsrummet.

Vi kan finde frem til en approximering af den forventede værdi af X ved brug af Monte Carlo Simulering. I den sammenhæng lad $\mathbb{E}[X]$ værende den forventede værdi af X. Vi kan derved opskrive følgende udsagn hvor x_i er en konkret simulering af X.

$$\mathbb{E}[X] = \sum_{i=1}^{n} \frac{x_i}{n} \tag{1.2}$$

Vi kan udfører en simulering der simulere 1000 eksperimenter, hvor X er uafhængig.

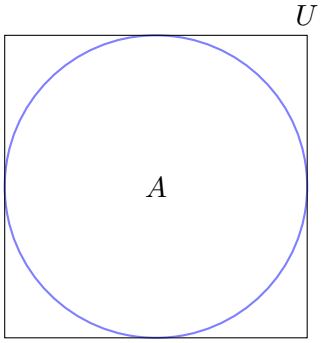
```
import random

random.seed(42)

print(sum([random.randint(1,2) for _ in range(1000)]) / 1000.000)
```

Dette vil give os $\mathbb{E}[X] = 1.519$, hvis vi udførte dette eksperiment 100 gange ville vi finde at vores resultat varierer. Dette skyldes at X er tilfældigt bestemt for hver simulering og er uafhængig af forgående simuleringer,altså hvad der ville betragtes som normal opførelse for en mønt [**NM**].

1.2 Numerisk integration

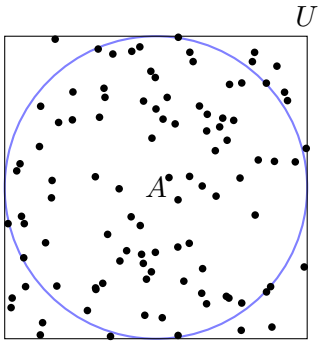
Det overstående eksempel med Plat eller Krone virker måske lidt dumt da vi med nemhed kan udlede det algebraisk, altså tage summen af udfaldsrummet og dele med antallet af mulige udfald. Men et andet tilfælde hvor man kan anvende Monte Carlo Metoden er numerisk integration. I nogle tilfælde er det let at udlede integralet algebragisk, men der findes også tilfæde hvor det er umuligt. Disse funktioners integraler kan udledes ved hjælp af numeriske metoder, her i blandt Monte Carlo Metoden også kaldet Hit-or-Miss. [**SBM**]

Lad os betragte enheds cirkel der befinder sig inden for en kvadrat af med dimensionerne 2×2, denne figurs integral kan let udledes men er valgt for nemheds skyld.



Vi kan starte med at bestemme udfaldsrummet af den overstående figur, dette er U og er alle de punkter der findes i kvadratets plan. Det integral vi ønsker at finde er cirklen, lad

cirklen være delmængde A. Desuden er sandsynligheden P for alle udfald i udfaldsrummet U lige, altså for to given udfald i U er $P(U_1) = P(U_2)$. Altså dette betyder at der er tale om et symmetrisk sandsynlighedsfelt, med et udfaldsrum der er bestående af punkter der findes på kvadratets plan. Den stokastiske variable X vil lige ledes repræsentere et muligt punkt på kvadratets plan. Dette felt er i teorien et kontinuert sandsynlighedsfelt, hvilket også ville gøre vores stokastiske variable til en kontinuer variable. Hvis vi har n stokastiske variabler X og n' betegner de stokastiske variabler der ligger inden for cirklens areal, altså således at hvis $X = (0.5; 0.5)$ vil det være et udfald i delmængden A. Dette betyder også alle elementer i delmængden A vil opfylde $a_i x^2 + a_i y^2 \leq 1^2$, hvor a_i er et konkret tilfælde og $a_i x$ er dets x koordinat samt $a_i y$ er dets y koordinat. Nedenstående figur viser et eksempel på n antal konkrete stokastiske variabler x_i i udfaldsrummet U hvor n' er de konkrete x_i som befinder sig i A.



Vi kan herved udregne en approximering af sandsynligheden for $P(A)$.

$$P(A) = \frac{n'}{n} \tag{1.3}$$

Hvis vi ønsker af finde arealet af cirklen skal vi gange med arealet af vores udfaldsrum, da dette var en kvadrat med dimensionerne 2×2 får vi følgende udtryk.

$$Areal(A) = \frac{n'}{n} \times 4 \tag{1.4}$$

En simulering af dette kan udtrykkes i følgende Python kode, hvor vi har 100000 konkrete stokastiske variabler

```
import random

random.seed(42)

kvadrattet = [(random.uniform(-1,1),
               random.uniform(-1,1)) for _ in range(100000)]
cirklen = [p for p in kvadrattet
           if p[0]*p[0] + p[1]*p[1] <= 1]

nk = len(kvadrattet) # lad dette vaere n
nc = len(cirklen)    # lad dette vaere n'
a = 2 * 2            # lad dette vaere arealet af kvadrattet

print("areal = {:d}/{:d}*{:d}={:f}".format(nc, nk, a, nc/nk*a))
```

Vi vil finde at arealet af cirklen ville være 3.140280 og derved vil $\pi = 3.140280$, da fejlen ved udregningen af et integral på denne måde er $\sqrt{\frac{1}{n}}$ ville vi finde at vi skulle øge antallet af punkter med en faktor 100 for at reducere fejlen med en faktor 10 [**SBM**].