Niels Bohr Institutet, Københavns Universitet

Python Aflevering 3

Simulation - Central Limit Theorem

Mekanik og relativitetsteori lab-teamet 2022

Revideret af: Sejr Sebastian Bergman, Jeppe Grejs Petersen, Athene Demuth, Selma Peiter Færch, Julie Harder Gabrielsen, Jacob Thornfeldt Hansen, Katrine Rahbek, og Børge Svane Nielsen

Baggrund:

I er begyndt at stifte bekendtskab med det statistiske begreb, en sandsynlighedsfordeling, hovedsagligt normalfordelingen. Det er denne fordeling som I trækker tal fra når I laver en måling i laboratoriet. Når I udregner et gennemsnit og en standardafvigelse er det netop normalfordelingen, der ligger bag. Ligningen der beskriver normalfordelingen er givet ved

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}, \tag{1}$$

hvor σ er standardafvigelsen og μ er gennemsnittet.

I skal i denne aflevering se styrken ved Central Limit Theorem (CLT). Det er på grund af dette teorem at I kan antage at jeres data er normalfordelte og at I derfor kan bruge middelværdien som jeres måleresultat og standardafvigelsen til at finde usikkerhederne. Det er grundlaget for al den usikkerhedsanalyse I laver i MekRel.

CLT siger at hvis man summer mange fordelinger vil de i sidste ende blive normalfordelte (gaussiske). Relationen til laboratoriet er så at når I laver et forsøg, lægger I mange statistiske usikkerheder oven i hinanden hver gang I laver et forsøg. For mere teori kan I læse i kapitel 1.8 i statistik noten.

I kan forstille jer at i måler den præsis samme ting N gange, og så finder middelværdien og usikkerheden. I stedet for at måle i laboratoriet, simulerer vi det her i et Python program, og så kan vi antage at der kun er statistiske usikkerheder. Dette gøres så 10 000 gange. Dette skal i gøre for 4 forskellige $N = 1, 5, 20$ og 50. I skal simulere at jeres måledata kommer fra en uniform fordeling mellem -1 og 1. Der skal altså være lige stor sandsynelighed for alle tal i intervallet $[-1, 1)$.

I kan tænke på N som antal terninger, som slås 10000 gange, hvor der tages middelværdien af alle øjnene. Hvis N=1, hvordan forventer man at fordelling vil se ud? Hvad hvis N=2?.

Opgave:

I skal skrive en Jupyter Notebook, som laver en masse forsøg for jer. Jeres kode skal gøre følgende:

- Have en funktion som returnerer et array med 10000 middelværdier, af jeres forsøg med de N målinger, uden brug af løkker! Brug i stedet `numpy.random.uniform`.
- Tegne et histogram over hvert af de fire forsøg, udført 10 000 gange.
- Fitte en normalfordeling til jeres bins og counts [1]. Brug her ligning (1). Arealet under en normalfordelingen er 1, så i skal enten normalisere jeres histogram, eller gange ligning (1) med antalet af counts. For at fitte skal I gætte nogle startværdier for fit-parametrene. Hvad ved vi om gennemsnittet?
- Give en værdi på gennemsnitsværdien for alle forsøgene, med usikkerhed. *Hint:* Usikkerhden på middelværdien er givet ved $\Delta_\mu = \frac{\sigma}{\sqrt{N}}$, hvor σ er standardafvigelsen og N er antallet af tal i datasættet.

OBS: Det er meget vigtigt at I uploader en `.ipynb` med jeres kode. Afleveres der i andet format end `.ipynb`, vurderes opgaven ikke og der gives 0 point. Husk også at koden skal kunne køre på vores computer, så restart kernel and run all, så i er sikre på at den kan køre.

[1] **OBS:** Hver opmærksom på om i fitter til kanten eller midten af jeres bins.

Hvad kan `numpy.random.uniform`?

Den kan lave tilfældige tal. Den skal have en værdi a og b, som udgør et interval $[a, b)$. Derudover gives en tuple, f.eks. $(2, 2)$. Følgende kode i Python

```
numpy.random.uniform(1, 101, (4,50))
```

returnerer da en 4×50 matrice af tilfældige tal i intervallet $[1, 101)$, hvor det er lige sandsynligt at få hvert tal.

Mit fit virker ikke!

Hvis man ikke kan få fittet til at virke kan det være fordi at parametrene for modellen skal have et start gæt. Dette gøres ved at skrive følgende (hvis I har importeret `curve_fit` fra `scipy.optimize`):

```
curve_fit(funktion, x, y, p0=[a,b])
```

I dette eksempel er der to parametre (a og b) der skal bestemmes. For at give et rigtigt start gæt skal man vide noget om data. Hvad er et godt gæt på σ og μ i ligning (1)?

Hvor kan jeg finde hjælp?

I Python caféen om fredagen og i noten her: https://python-intro.nbi.ku.dk/