Pythonmatte

Programmering i matematikk programfag på vgs

Torodd F. Ottestad

Innhald

Om boka

Denne boka inneheld ulike måtar ein kan nytta programmering på i matematikk (programfag på vgs).

Eg nyttar Python som programmeringsspråk gjennom heile boka .

> **OBS**
>
> Boka er under utvikling og vert oppdatert med ujamne mellomrom.

For dei spesielt interesserte er boka laga med Quarto. For å lære meir om Quarto-bøker kan ein kikka her.

Logo: Programmer icons created by juicy_fish - Flaticon

Part I

Sannsyn og simulering S1/S2

I det følgjande kapittelet skal me sjå på korleis me kan simulera ulike stokastiske forsøk i Python.

...

1 Terningar og intro til simulering

Ein fin stad å starta med simulering er med terningar Her er sannsynet *uniformt* (det er like sannsynleg å få 2 som 5), og dei ulike utfalla er heiltal.

Det første som må gjerast er å gjera i stand "trekkaren" vår. Eg bruker her ein tilfeldighetsgenerator frå `NumPy` (dokumentasjon her).

```
import numpy as np
rng = np.random.default_rng()
```

Når me no har klargjort generatoren kan me bruka den innebygde `integers`-funksjonen for å trilla ein terning.

1.1 Ein terning

```
terning = rng.integers(1, 7)
print(terning)
```

```
6
```

> **!** Merk
>
> Her er verdien `terning` eit tal **større eller lik** 1 og **mindre enn** 7. Sidan det er heiltal me trekk er dermed
>
> $$\text{terning} \in \{1, 2, 3, 4, 5, 6\}$$

For å trilla fleire terningar kan me anten bruka løkker:

```
for i in range(10):
    print(rng.integers(1, 7))
```

```
3
4
1
6
5
2
1
5
1
5
```

eller så kan me legga inn eit argument `size` i `integers`. Då blir output ein array (ein form for liste) med `size` terningar:

```
terningar = rng.integers(1, 7, size=10)
print(terningar)
```

```
[2 3 4 6 1 1 1 1 6 6]
```

No har me det me treng for å kunna simulera eit stokastisk forsøk og estimera sannsyn ut frå simuleringa. Til dømes kan me prøva å finna ut av kor sannunleg det er å trilla 5 eller 6 på ein terning:

```
N = 1000000 # tal simuleringar

terningar = rng.integers(1, 7, size=N)

gunstige = sum(terningar >= 5)

sannsyn = gunstige / N

print(f"Sannsynet for 5 eller 6 er {sannsyn:.4f}")
```

```
Sannsynet for 5 eller 6 er 0.3336
```

> Forklaring: `gunstige = sum(terningar >= 5)`
>
> For å forstå denne ser me på eit døme:

```python
array = np.array([1, 2, 3, 4, 5, 6])

større_enn_3 = array > 3

print(array)
print(større_enn_3)
print(sum(større_enn_3))
```

```
[1 2 3 4 5 6]
[False False False  True  True  True]
3
```

Altså gjer me om verdiar til `True` eller `False`. Python reknar `True` som 1 og `False` som 0. Når me då summerer alle elementa i `array` får me antall `True` i arrayen.

1.2 Fleire terningar

Spørsmål som "Kva er sannsynet for at produktet av to terningar er 8 eller mindre" er fint å finna svar på ved hjelp av simulering:

```python
N = 1000000

terning1 = rng.integers(1, 7, size=N)
terning2 = rng.integers(1, 7, size=N)

produkt = terning1 * terning2
gunstige = sum(produkt <= 8)
sannsyn = gunstige / N

print(f"Sannsynet er {sannsyn:.4f}")
```

```
Sannsynet er 0.4452
```

Forklaring: `produkt = terning1 * terning2`

Kodelikna finn produktet av element på samme plass i dei to arrayane. Sjekk dømet:

```
a = np.array([1, 2, 3, 4, 5])
b = np.array([6, 7, 8, 9, 10])

c = a * b

print(c)
```

```
[ 6 14 24 36 50]
```

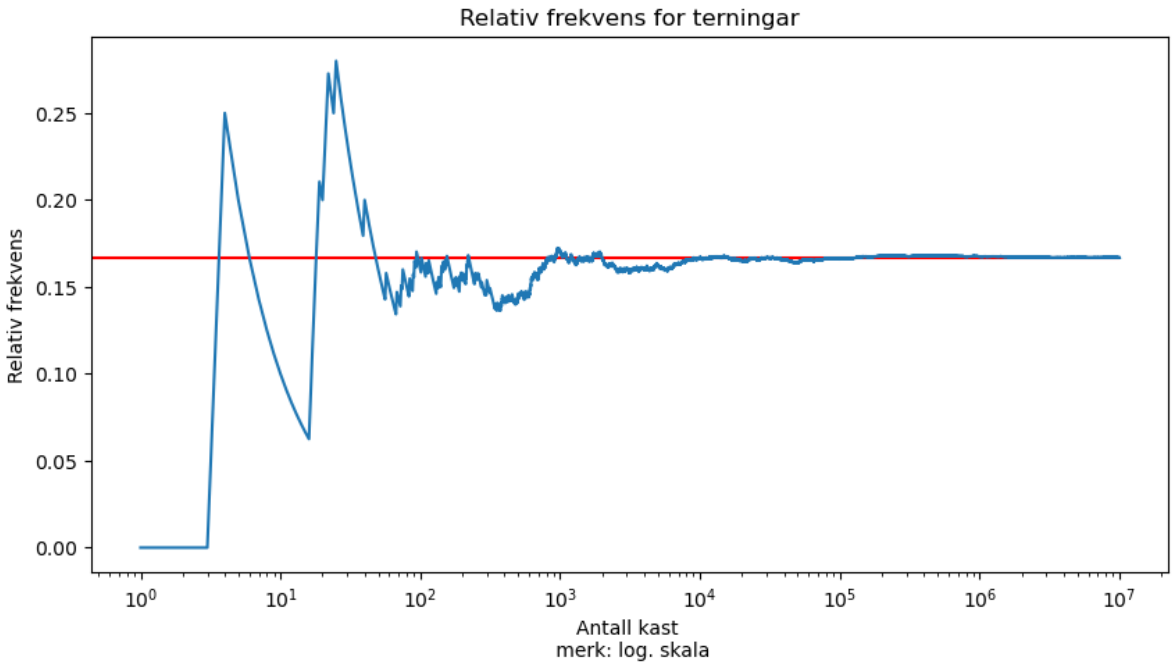
$1 \cdot 6 = 6$ og $2 \cdot 7 = 14$...

1.3 Nøyaktighet

Sjekkar kva som skjer når me triller fleire og fleire terningar (eller ein terning fleire gongar). For å visa samanhengen plottar me resultatet. I dømet ser me på sannsynet for å trilla 4 på ein terning.

```
import matplotlib.pyplot as plt

# antall kast
N = 10000000

# triller terningar
terningar = rng.integers(1, 7, size=N)

# finn den kumulative summen av terningar som er lik 4
kumulativ_sum = np.cumsum(terningar == 4)

# lager "x-akse" frå 1 til N
x = np.arange(1, N + 1)

# finn relativ frekvens
rel_frekvens = kumulativ_sum / x

plt.figure(figsize=(10, 5))                    # lagar ein figur med 10x5 mål
plt.hlines(1/6, 0, N, color="red")             # teiknar ein linje med farge "red" for d
plt.plot(x, rel_frekvens)                      # plottar x-akse og y-akse
plt.xscale("log")                              # logaritmisk x-akse
plt.xlabel("Antall kast \n merk: log. skala")  # namn på x-aksen
```

```
23  plt.ylabel("Relativ frekvens")                    # namn på y-aksen
24  plt.title("Relativ frekvens for terningar")       # tittel på figur
25  plt.show()
```



Her ser me at di fleire kast me gjennomfører, di nærare kjem den relative frekvensen den teoretiske verdien for å trilla ein firar på vanleg terning.

$$P(\text{firar}) = \frac{1}{6} \approx 0.167$$

2 Summary

In summary, this book has no content whatsoever.