STK-IN1050 - Statistikk for informatikere

15. Bootstrap

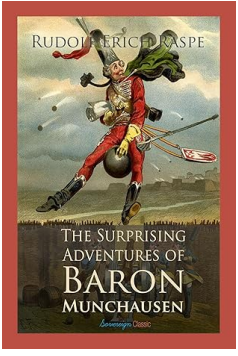
Matematisk institutt

2026-03-25

Bootstrap-metoden

Bootstrap-metoden er en data-dreven måte å måle usikkerhet på, uten å anta en bestemt sann fordeling (f.eks. normalfordeling).

«The Baron had fallen to the bottom of a deep lake. Just when it looked like all was lost, he thought to pick himself up by his own bootstraps.»



Barnebok fra 1781

Kilde: amazon.com

Vi starter med et enkelt eksempel[1]

Anta at vi ønsker å investere et fast beløp i to forskjellige fond som henholdsvis gir avkastningene X og Y, der X og Y er stokastiske variabler.

Vi investerer en andel α av beløpet i X, og den resterende andelen $1 - \alpha$ i Y.

Vi ønsker å velge α slik at den totale risikoen, eller variansen, av investeringen blir minst mulig. Med andre ord vil vi minimere

$$\text{Var}(\alpha X + (1 - \alpha)Y).$$

[1] https://web.stanford.edu/~hastie/MOOC-Slides/cv_boot.pdf

Eksempel, forts.: minimal risiko

Det kan vises at verdien som minimerer risikoen er gitt ved

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}},$$

der $\sigma_X^2 = \text{Var}(X)$, $\sigma_Y^2 = \text{Var}(Y)$ og $\sigma_{XY} = \text{Cov}(X, Y)$.

Men verdiene σ_X^2, σ_Y^2 og σ_{XY} er ukjente.

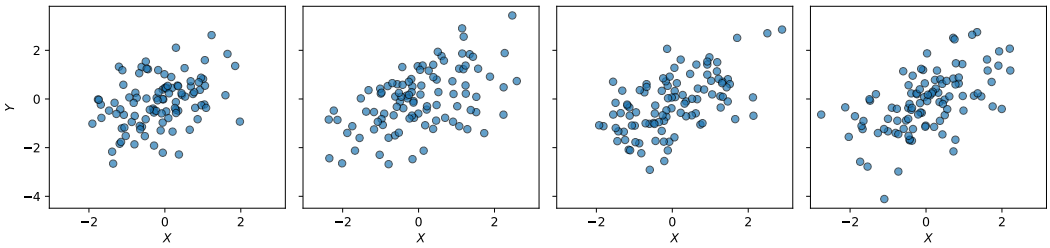
Eksempel, forts.: minimal risiko

Vi kan beregne estimater, $\hat{\sigma}_X^2$, $\hat{\sigma}_Y^2$ og $\hat{\sigma}_{XY}$, ved hjelp av et datasett som inneholder observasjoner av X og Y.

Deretter kan vi estimere verdien av α som minimerer variansen til investeringen ved

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}.$$

Eksempel, forts.: simulerte data



Hver figur viser 100 simulerte avkastninger for fondene X og Y.

De resulterende estimatene for α er: 0.629, 0.576, 0.621, 0.703.

Eksempel, forts.: sampling via simulering

For å estimere standardavviket til $\hat{\alpha}$ gjentar vi prosessen med å simulere 100 parvise observasjoner av X og Y, og beregne $\hat{\alpha}$ 1000 ganger.

På den måten får vi 1000 estimater av α, som vi kaller

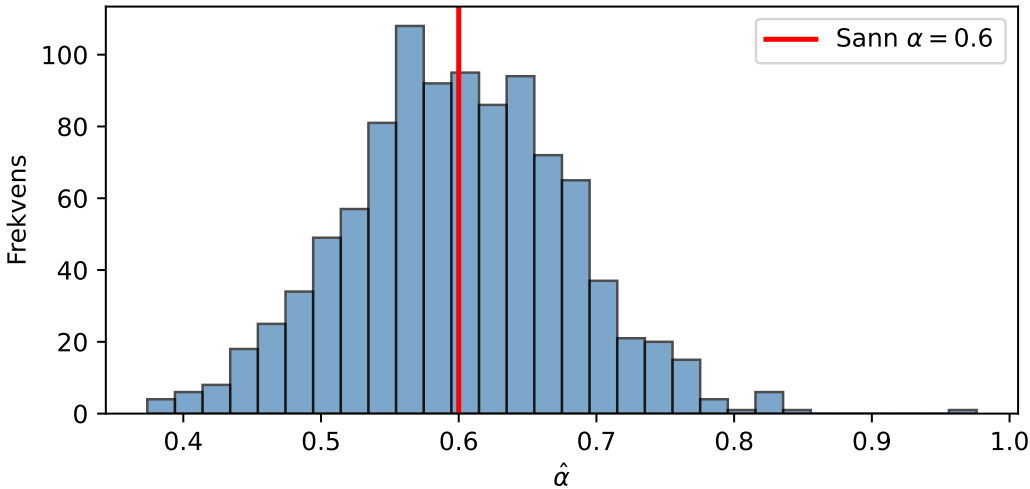
$$\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_{1000}.$$

I disse simuleringene er parametrene satt til

$$\sigma_X^2 = 1, \quad \sigma_Y^2 = 1.25, \quad \sigma_{XY} = 0.5,$$

og dermed vet vi at den sanne verdien er $\alpha = 0.6$.

Resultater fra sampling via simulering



Resultater fra sampling via simulering

Gjennomsnittet av de 1000 estimatene er

$$\bar{\alpha} = \frac{1}{1000} \sum_{r=1}^{1000} \hat{\alpha}_r = 0.5998,$$

som er ganske nær den sanne verdien $\alpha = 0.6$.

Standardavviket til estimatene er

$$\sqrt{\frac{1}{1000 - 1} \sum_{r=1}^{1000} (\hat{\alpha}_r - \bar{\alpha})^2} = 0.080.$$

Resultater fra sampling via simulering

Dette gir oss et godt inntrykk av nøyaktigheten til $\hat{\alpha}$:

$$\text{se}(\hat{\alpha}) \approx 0.080.$$

Grovt sett forventer vi altså at $\hat{\alpha}$ vil avvike fra α med rundt 0.08 i gjennomsnitt, for et tilfeldig utvalg fra populasjonen.

Tilbake til den virkelige verden

Prosedyren med å generere helt nye utvalg fra populasjonen kan ikke brukes i praksis, fordi vi for reelle data ikke kan hente ut uendelig mange nye, uavhengige datasett fra den sanne fordelingen.

Tilbake til den virkelige verden

Bootstrap-metoden lar oss imitere prosessen med å skaffe nye datasett, slik at vi kan anslå variasjonen til estimatet uten å generere flere virkelige utvalg.

I stedet for å hente gjentatte, uavhengige datasett fra populasjonen, konstruerer vi nye datasett ved gjentatt trekking av observasjoner fra det opprinnelige datasettet med tilbakelegging.

Hvert slikt **«bootstrap-datasett»** lages ved trekking med tilbakelegging og har samme størrelse som originaldatasettet. Som følge av dette kan noen observasjoner forekomme flere ganger i ett bootstrap-datasett, mens andre ikke forekommer i det hele tatt.

Eksempel med tre observasjoner

Opprinnelig datasett Z:

Obs	X	Y
1	4.3	2.4
2	2.1	1.1
3	5.3	2.8

Tre mulige bootstrap-datasett (Z^{*1}, Z^{*2},Z^{*3}):

Obs	X	Y
1	4.3	2.4
2	2.1	1.1
1	4.3	2.4

Obs	X	Y
2	2.1	1.1
2	2.1	1.1
1	4.3	2.4

Obs	X	Y
2	2.1	1.1
3	5.3	2.8
1	4.3	2.4

Bootstrap-prosedyre for α

La det første bootstrap-datasettet være Z^{*1}. Vi bruker Z^{*1} til å beregne et nytt bootstrap-estimat av α, som vi kaller $\hat{\alpha}^{*1}$.

Denne prosedyren gjentas B ganger (for en stor verdi av B, for eksempel $B = 100$ eller $B = 1000$), og gir B ulike bootstrap-datasett

$$Z^{*1}, Z^{*2}, \ldots, Z^{*B},$$

og B tilsvarende estimater

$$\hat{\alpha}^{*1}, \hat{\alpha}^{*2}, \ldots, \hat{\alpha}^{*B}.$$

Bootstrap-prosedyre for α

Vi estimerer standardfeilen til disse bootstrap-estimatene ved

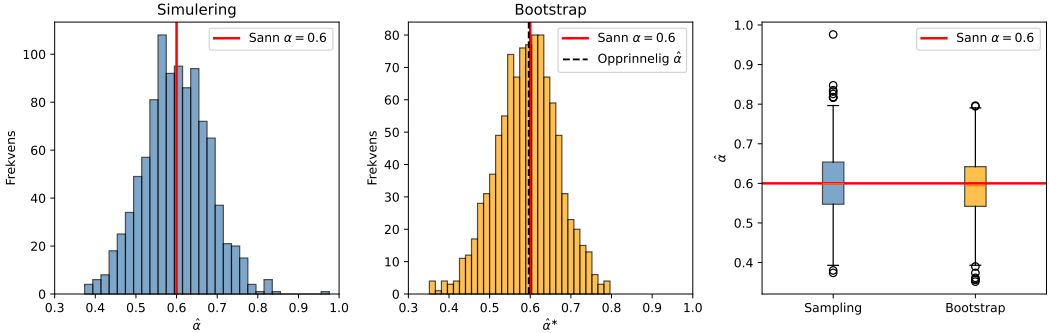
$$\widehat{\mathrm{se}}_B(\hat{\alpha}) = \sqrt{\frac{1}{B-1}\sum_{r=1}^{B}\left(\hat{\alpha}^{*r} - \bar{\hat{\alpha}}^{*}\right)^2},$$

der

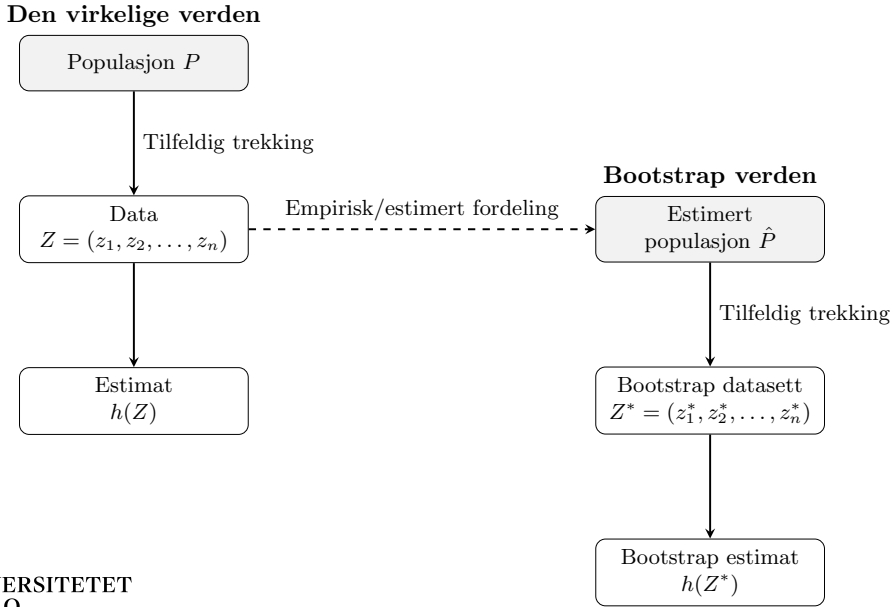
$$\bar{\hat{\alpha}}^{*} = \frac{1}{B}\sum_{r=1}^{B}\hat{\alpha}^{*r}.$$

Dette fungerer som et estimat for standardfeilen til $\hat{\alpha}$ beregnet fra det opprinnelige datasettet. For vårt eksempel er $B = 1000$ og $\widehat{\mathrm{se}}_B(\hat{\alpha}) = 0.075$.

Sammenligning av simulering og bootstrap



Vi ser at bootstrap-fordelingen ligner fordelingen vi hadde fått dersom vi kunne trekke mange nye utvalg fra populasjonen.


Den virkelige verden
Populasjon P
Tilfeldig trekking
Data
$Z = (z_1, z_2, \ldots, z_n)$
Empirisk/estimert fordeling
Bootstrap verden
Estimert
populasjon $\hat{P}$
Tilfeldig trekking
Estimat
$h(Z)$
Bootstrap datasett
$Z^* = (z_1^*, z_2^*, \ldots, z_n^*)$
Bootstrap estimat
$h(Z^*)$

Gruppediskusjon

- Beskriv med egne ord hva bootstrap-metoden gjør
- Hva er de underliggende antakelsene for at bootstrap skal fungere?
- Hvordan kan avhengighet i f.eks. finansdata som i dette eksemplet bryte disse antakelsene?
- Foreslå to konkrete scenarioer der bootstrap kan gi misvisende resultater for $\hat{\alpha}$

Noen punkter til diskusjonen

- Datasettet bør være et representativt, (omtrent) tilfeldig utvalg fra populasjonen
- Observasjonene antas i praksis å være uavhengige og fra samme fordeling
- Utvalget bør være stort nok til at fordelingen i data er rimelig godt fanget
- Estimatoren bør være relativ glatt som funksjon av data
- I liknende simulerte settinger bør bootstrap-fordelingen ligne den ekte sampling-fordelingen

Noen punkter til diskusjonen

- Hvis dataene har en komplex struktur, kan det kreve litt ettertanke å finne en passende måte å konstruere bootstrap-utvalg på
- For eksempel: hvis dataene er en tidsserie med tidsavhengighet, kan vi ikke bare trekke enkeltobservasjoner med tilbakelegging – da bryter vi tidsavhengigheten i dataene som er viktig å bevare
- I stedet kan vi lage blokker av påfølgende observasjoner, og trekke hele blokker med tilbakelegging. Deretter «limer» vi disse blokkene sammen for å få et bootstrap-datasett som ligner mer på en realistisk tidsserie

Mer generelt, så har vi to forskjellige tilnærminger til bootstrap-metoden:

Ikke-parametrisk bootstrap

- Trekker med tilbakelegging fra de observerte dataene
- Trenger ingen eksplisitt modell for fordelingen
- Basert på den empiriske fordelingen til dataene

Parametrisk bootstrap

- Antar en parameterisert modell for populasjonen
- Estimerer parametre
- Simulerer nytt datasett fra denne modellen

Parametrisk bootstrap og standardfeil

La Y_i, $i = 1, \ldots, n$, være levetidene til n komponenter hvor vi har observert $n = 10$ levetider[2]:

```
[41.53, 18.73, 2.99, 30.34, 12.33, 117.52, 73.02, 223.63, 4.0, 26.78]
```

En standard modell for levetider er eksponensiellfordelingen med tetthet

$$f(y; \theta) = \theta e^{-\theta y}, \quad y \geq 0.$$

Her er $E(Y_i) = \mu = 1/\theta$ og dermed $E(\overline{Y}) = \mu$; naturlige estimatorer er

$$\hat{\mu} = \overline{Y} = 55.087, \quad \hat{\theta} = 1/\overline{Y} = 0.0185.$$

[2] Devore, Jay L, Kenneth N Berk, Matthew A Carlton, et al. 2012. Modern Mathematical Statistics with Applications. Vol. 285. Springer.

Estimatoren $\hat{\theta}$ og standardfeil

Vi er interessert i **standardfeilen** til $\hat{\theta}$, dvs. hvor mye vil $\hat{\theta}$ variere fra utvalg til utvalg?

Hvis vi kunne gjenta eksperimentet mange ganger i virkeligheten:

- Vi kunne beregne $\hat{\theta}_1, \ldots, \hat{\theta}_B$.
- Variansen og standardfeilen til $\hat{\theta}$ kunne estimeres empirisk:

$$\widehat{\mathrm{Var}}(\hat{\theta}) = \frac{1}{B-1}\sum_{b=1}^{B}\left(\hat{\theta}_b - \bar{\theta}\right)^2, \quad \widehat{\mathrm{se}}(\hat{\theta}) = \sqrt{\widehat{\mathrm{Var}}(\hat{\theta})}.$$

Simuleringsbasert sampling

Vi er interessert i **standardfeilen** til $\hat{\theta}$, dvs. hvor mye vil $\hat{\theta}$ variere fra utvalg til utvalg?

I praksis kan vi ikke gjenta eksperimentene, men vi kan **simulere**:

- Anta at eksponentialmodellen er korrekt, og at $\theta = 0.0185$ er den sanne verdien
- Vi genererer $B = 1000$ datasett fra **den (antatt) sanne modellen**
- For hvert datasett beregner vi $\hat{\theta}^{(b)} = 1/\overline{Y}^{(b)}$
- Standardfeilen estimeres som standardavviket til $\hat{\theta}^{(1)}, \ldots, \hat{\theta}^{(B)}$

Simuleringsbasert sampling

```
import numpy as np
y = [41.53,18.73,2.99,30.34,12.33,117.52,73.02,223.63,4.00,26.78]
def boot_par_theta_se(y,B):
    n = len(y)
    theta_hat = 1/np.mean(y)
    theta_sim = np.zeros(B)
    for b in range(B):
        ystar = np.random.exponential(1/theta_hat, n)
        theta_sim[b] = 1/np.mean(ystar)
    theta_se = np.std(theta_sim, ddof=1)
    return(theta_se)

print(f"Estimert SE = {boot_par_theta_se(y,1000):.6f}")
```

```
Estimert SE = 0.006856
```

Parametrisk bootstrap: valg av B

La oss kjøre funksjonen flere ganger:

```
se_sim = np.zeros(5)
for k in range(5):
    se_sim[k] = boot_par_theta_se(y, 1000)
se_sim
```

```
array([0.00695299, 0.00763227, 0.00685596, 0.00720537, 0.00690205])
```

Øker vi *B* til 10 000:

```
se_sim = np.zeros(5)
for k in range(5):
    se_sim[k] = boot_par_theta_se(y, 10000)
se_sim
```

```
array([0.00730103, 0.00714035, 0.00702593, 0.00712386, 0.0072376 ])
```

Parametrisk bootstrap: valg av B

Vi har to kilder til usikkerhet:

1. Usikkerhet i de opprinnelige observasjonene $y_1, \ldots, y_n$.
2. Usikkerhet fra simuleringen (valg av B).

I praksis er punkt 1 mye viktigere enn punkt 2, så $B \approx 1000$ er ofte nok.

Ikke-parametrisk bootstrap

Parametrisk bootstrap forutsetter at modellen $f(y;\theta)$ er riktig, eller i hvert fall rimelig

Hvis vi er usikre på om eksponentialmodellen er god nok, kan vi bruke **ikke-parametrisk bootstrap** i stedet:

- **Få antakelser** om den sanne fordelingen *F*
- Bruker den **empiriske fordelingen** til dataene

Empirisk fordeling og $\widehat{F}$

La $F(x) = P(X \leq x)$ være den sanne kumulative fordelingsfunksjonen.

En naturlig estimator er den **empiriske fordelingsfunksjonen**:

$$\widehat{F}(x) = \frac{1}{n}\big(\#\{\, i : y_i \leq x \,\}\big).$$

Under $\widehat{F}$ har vi:

$$P(X = y_i) = \frac{1}{n}, \quad i = 1, \ldots, n.$$

Dvs. å **trekke fra** $\widehat{F}$ er det samme som å trekke med tilbakelegging fra $y_1, \ldots, y_n$.

Ikke-parametrisk bootstrap: sampling

```
def boot_nonpar_theta_se(y, B):
    n = len(y)
    theta_hat = 1/np.mean(y)
    theta_sim = np.zeros(B)
    for b in range(B):
        ystar = np.random.choice(y, size=n, replace=True)
        theta_sim[b] = 1/np.mean(ystar)
    se_np_sim = np.std(theta_sim, ddof=1)
    return se_np_sim

print(f"Estimert SE = {boot_nonpar_theta_se(y,1000):.6f}")
```

```
Estimert SE = 0.010052
```

Parametrisk vs ikke-parametrisk: standardfeil

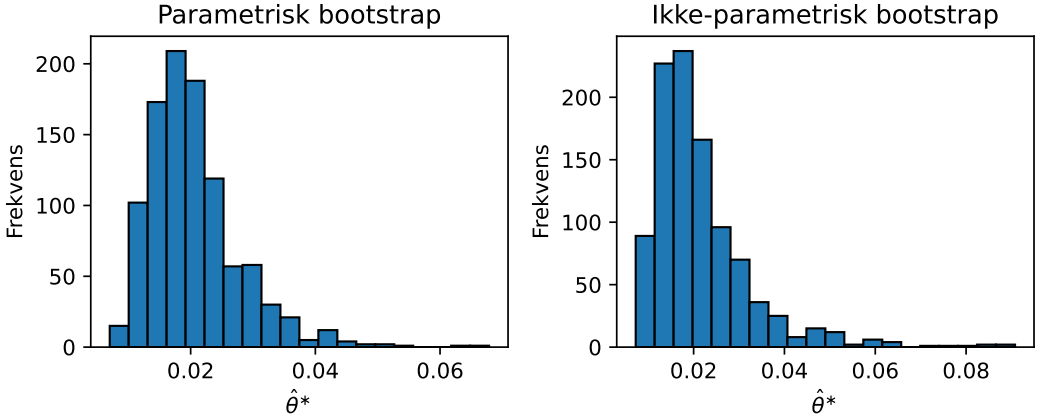
```
se_par_list = np.zeros(5)
se_nonpar_list = np.zeros(5)
for k in range(5):
    se_par_list[k] = boot_par_theta_se(y, 1000)
    se_nonpar_list[k] = boot_nonpar_theta_se(y, 1000)
se_par_list, se_nonpar_list
```

```
(array([0.00727662, 0.0072574 , 0.00675549, 0.00692455, 0.00689853]),
 array([0.01030632, 0.00934366, 0.01065344, 0.00960008, 0.01030294]))
```

Den ikke-parametriske standardfeilen er noe større:

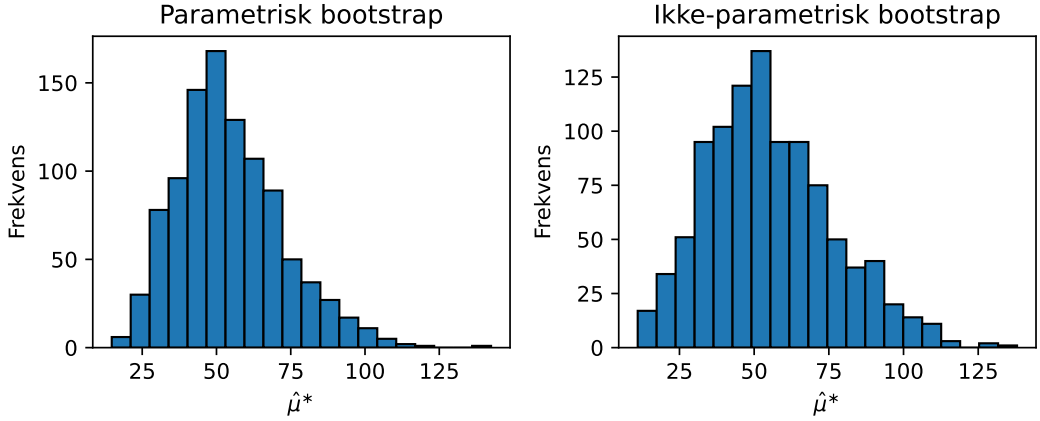
- kan tyde på at eksponensiellmodellen ikke passer perfekt;
- hvis modellen var helt korrekt, burde parametrisk og ikke-parametrisk bootstrap gi ganske like resultater.

Bootstrap og fordeling til estimatoren



Begge fordelinger er tydelig skjeve: n er lavt ($n = 10$) og $\hat{\theta}$ er ikke et gjennomsnitt.

Alternativ: Bootstrap for $\hat{\mu} = \bar{Y}$ i stedet for $\hat{\theta}$



Nå er fordelingene nærmere normalfordelingen, selv om noe skjevhet gjenstår (lite *n*, og underliggende data er skjeve).

Bootstrap og forventningsskjevhet

Vi vil nå bruke bootstrap til å si noe om forventningsskjevhet i $\hat{\theta}$:

$$\text{Bias}(\hat{\theta}) = \mathbb{E}[\hat{\theta}] - \theta.$$

I praksis bruker vi bootstrap-simuleringer og estimerer:

$$\widehat{\text{Bias}}(\hat{\theta}) = \overline{\theta}^* - \hat{\theta},$$

der $\overline{\theta}^*$ er gjennomsnittet av de bootstrap-simulerte $\hat{\theta}^*$-ene.

Resultater

	Estimat	SE	Bias
Parametrisk	0.018153	0.007548	0.001991
Ikke-parametrisk	0.018153	0.009630	0.002938

- Estimatet $\hat{\theta}$ er det samme i begge tilfeller (det er basert på de observerte dataene).
- Standardfeilen er litt større i ikke-parametrisk bootstrap.
- Bias ser ut til å være relativt liten i begge tilfeller.

MSE: varians + bias^2

Husk at den gjennomsnittlige kvadratfeilen (MSE) kan skrives som

$$\text{MSE}(\hat{\theta}) = \mathbb{E}[(\hat{\theta} - \theta)^2] = \underbrace{\mathbb{E}[(\hat{\theta} - \mathbb{E}[\hat{\theta}])^2]}_{\text{varians}} + \underbrace{(\mathbb{E}[\hat{\theta}] - \theta)^2}_{\text{bias}^2}.$$

Vi kan approksimere MSE ved:

```
mse_par = se_par**2 + bias_par**2
mse_nonpar = se_nonpar**2 + bias_nonpar**2
print(f"MSE = {mse_par:.6f}, bias² = {bias_par**2:.6f} (parametrisk)")
print(f"MSE = {mse_nonpar:.6f}, bias² = {bias_nonpar**2:.6f}  (ikke-parametrisk)")
```

```
MSE = 0.000061, bias² = 0.000004 (parametrisk)
MSE = 0.000101, bias² = 0.000009  (ikke-parametrisk)
```

Konklusjon: Variansdelen dominerer og forventningskjevheten er neglisjerbar

Oppsummering: parametrisk vs ikke-parametrisk bootstrap

- **Parametrisk bootstrap**
  - Antar en spesifikk modell $F(x;\theta)$ (her eksponentialfordeling).
  - Estimerer $\hat{\theta}$ fra data og simulerer datasett fra $F(\cdot;\hat{\theta})$.
  - Mer effektiv hvis modellen er noenlunde korrekt.

- **Ikke-parametrisk bootstrap**
  - Antar lite om fordelingen.
  - Bruker empirisk fordeling $\widehat{F}$; trekker med tilbakelegging fra de observerte y_i.
  - Robust mot modellfeil; standardfeil kan bli noe større.

- Begge metoder prøver å etterligne den "ekte" sampling-fordelingen til estimatoren, slik vi så for $\hat{\alpha}$ i investerings-eksempelet.

Bootstrap-konfidensintervaller

Den enkleste måten er å bruke Wald-metoden:

$$\hat{\theta} \approx N\big(\theta, [\widehat{\text{se}}(\hat{\theta})]^2\big).$$

Et $(1-\alpha)100\%$ konfidensintervall blir da

$$\hat{\theta} \pm z_{\alpha/2}\, \widehat{\text{se}}(\hat{\theta}),$$

der $\widehat{\text{se}}(\hat{\theta})$ er bootstrap-standardfeilen, og $z_{\alpha/2}$ er $(1-\alpha/2)$ kvantil i standard normalfordeling.

Men: hvis fordelingen til $\hat{\theta}$ er skjev, kan normalapproksimasjonen være dårlig.

Wald-metoden i levetid-eksempelet

For levetids-eksempelet så vi at bootstrap-fordelingen til $\hat{\theta}$ er sterkt skjev.

Derimot var bootstrap-fordelingen til $\hat{\mu} = \overline{Y}$ mye nærmere en normalfordeling.

Idé: Bruk Wald-metoden for μ:

$$\hat{\mu} \pm z_{\alpha/2}\, \widehat{\mathrm{se}}(\hat{\mu}),$$

og bruk sammenhengen $\theta = 1/\mu$ for å få intervall for θ:

$$\left[\frac{1}{\hat{\mu} + z_{\alpha/2}\widehat{\mathrm{se}}(\hat{\mu})}, \frac{1}{\hat{\mu} - z_{\alpha/2}\widehat{\mathrm{se}}(\hat{\mu})} \right].$$

Wald-metoden i levetid-eksempelet

```
from scipy.stats import norm
import numpy as np
import pandas as pd

z = norm.ppf(0.975)          # 95%-intervall

# 1) Normalintervall direkte for theta
int1 = [theta_hat - z*se_nonpar, theta_hat + z*se_nonpar]
# 2) Normalintervall for mu og så transformere tilbake
mu_hat = 1/theta_hat
se_mu_nonpar = np.std(mu_sim_nonpar, ddof=1)
int_mu = [mu_hat - z*se_mu_nonpar, mu_hat + z*se_mu_nonpar]
int2 = np.array([1/int_mu[1], 1/int_mu[0]])

data = {
    "Nedre 2.5%": [int1[0], int2[0]],
    "Øvre 2.5%": [int1[1], int2[1]],
}
df = pd.DataFrame(data, index=["Normal approx", "Normal approx + trans"])
df
```

Wald-metoden i levetid-eksempelet

	Nedre 2.5%	Øvre 2.5%
Normal approx	-0.000721	0.037027
Normal approx + trans	0.010325	0.075083

Her har vi brukt resultatene fra ikke-parametrisk bootstrap.

De to 95%-intervallene blir ganske ulike.

Negative verdier bør være umulig siden $\theta > 0$, men ikke klart heller om den andre er mye bedre.

Percentile-intervall

Et lternativ er **percentile-intervallet**:

$$[\hat{q}_{\alpha/2}, \hat{q}_{1-\alpha/2}],$$

altså direkte kvantiler av de simulerte verdiene $\hat{\theta}^{*b}$.

Denne metoden bygger på noen antakelser om formen på fordelingen (symmetri-type antakelser).

	Nedre 2.5%	Øvre 2.5%
Parametrisk perc	0.010433	0.038318
Ikke-parametrisk perc	0.010340	0.045039

Fordel: intervallet holder seg automatisk innenfor "lovlig område" (her kan nedre grense ikke bli negativ hvis alle $\hat{\theta}^{*b} > 0$).

Innebygde rutiner: `scipy.stats.bootstrap`

I mange programmer finnes det ferdige bootstrap-funksjoner.

Typisk:

- Du definerer en funksjon som beregner $\hat{\theta}$ fra data.
- Rutinene gjør **ikke-parametrisk** bootstrap (lettest generelt).

I `scipy.stats` finnes funksjonen `bootstrap`, som støtter:

- `"basic"` – basic bootstrap-intervall (se kapittel 9 i bok)
- `"percentile"` – percentile-intervall
- `"bca"` – BCa (bias-corrected and accelerated), mer avansert metode

Innebygde rutiner: `scipy.stats.bootstrap`

```
from scipy.stats import bootstrap

def thetahatF(sample):
    return 1 / np.mean(sample)

int_perc = bootstrap((y,), thetahatF,
                     confidence_level=0.95,
                     n_resamples=1000,
                     method="percentile",
                     vectorized=False).confidence_interval
```

Innebygde rutiner: resultater

	Lower 2.5%	Upper 2.5%
Basic	-0.014035	0.026583
Percentile	0.010232	0.046559
BCa	0.008614	0.037914

Vi ser at:

- percentile-intervallene ligner det vi fikk med egne implementasjoner.
- BCa-intervallene er her svært like percentile-intervallene:
  - lav forventningsskjevhet i $\hat{\theta}$,
  - øvrige forutsetninger for percentile-metoden ser rimelig ut.

Simulasjonsstudie: dekning

Vi avslutter med en simulering for å undersøke treffsikkerheten (dekningen) til ulike bootstrap-intervall.

Oppsett:

- Sett $\theta = 0.0185$ («sann» verdi).
- Simuler M datasett fra eksponentialfordeling med denne θ.
- For hvert datasett:
  - beregn ulike bootstrap-intervall for θ,
  - sjekk om intervallet inneholder den sanne parameteren.
- Godt kalibrert intervall:
  - For $\alpha = 0.05$ ønsker vi dekning ≈ 0.95.

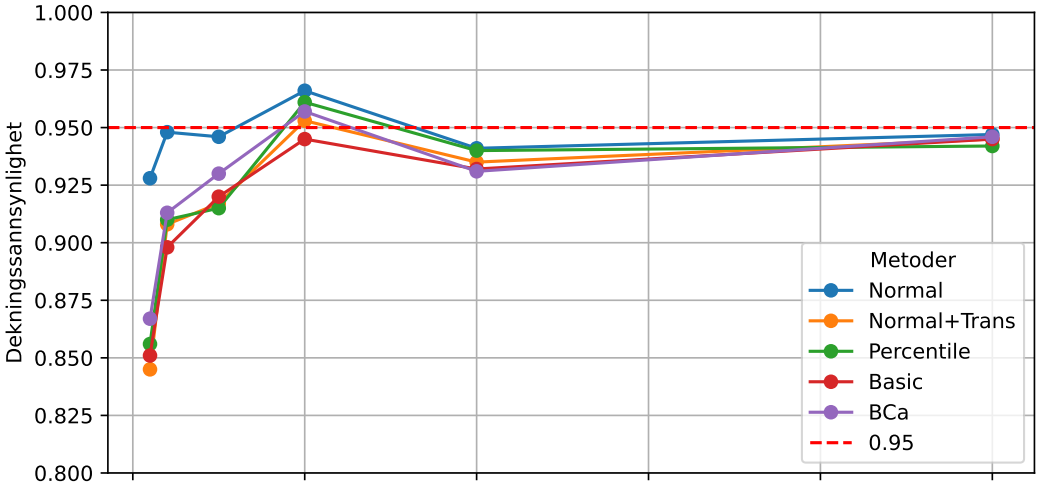
Første test: $n = 10$

```
Normal          : 0.9500
Normal+Trans    : 0.9100
Percentile      : 0.9200
Basic           : 0.8500
BCa             : 0.9100
```

De fleste metodene gjør det ganske bra allerede ved $n = 10$.

Alle metodene bygger på asymptotisk teori: forventes å fungere best når n er stor.

Dekning som funksjon av *n*



1

Hva er bootstrap først og fremst en metode for?

(a) Å estimere usikkerhet ved å resample data
(b) Å finne årsakssammenhenger i data
(c) Å øke antall observasjoner i datasettet
(d) Å fjerne outliers automatisk

2

Hva betyr ikke-parametrisk bootstrap?

(a) Vi endrer datasettet manuelt før analyse
(b) Vi bruker bare teoretiske formler, ikke data
(c) Vi antar at dataene følger en normalfordeling
(d) Vi resampler direkte fra de observerte dataene

3

Hva er en viktig fordel med bootstrap?

(a) Den gir alltid et eksakt svar
(b) Den krever ofte færre antakelser om fordelingen
(c) Den fungerer bare for store datasett
(d) Den erstatter behovet for statistisk usikkerhet

4

Når kan bootstrap være spesielt nyttig?

(a) Når den teoretiske fordelingen er vanskelig å bruke
(b) Når vi bare vil sortere data alfabetisk
(c) Når vi vil unngå å regne ut noe statistikk
(d) Når datasettet ikke inneholder noen observasjoner