Half the difference was who lives there

Nord's crude attack rate is 7.94 per 1,000 and Sud's is 5.51. Standardised to a common population they are 7.25 and 5.95, so the gap falls from 2.43 to 1.30 — and the missing half was age structure.

Pierre Robentz Cassion

Lesson 6 of 8

Public Health and Applied Epidemiology — Comparing places

What this lesson covers

- Two districts are not comparable as they stand
- Direct standardisation
- What the standard population is, and why it matters
- Indirect standardisation, and when you need it
- Age is not the only confounder
- Report the pair
- What comes next

Two districts are not comparable as they stand — In Python

```
import pandas as pd

population = pd.read_csv("district-population-2024.v1.csv")

structure = (
    population.pivot(index="district", columns="age_band", values="population")
)
shares = structure.div(structure.sum(axis=1), axis=0)
print((shares * 100).round(1))
```

Two districts are not comparable as they stand — In R

```
library(dplyr)

population |>
  mutate(share = population / sum(population), .by = district) |>
  tidyr::pivot_wider(id_cols = district, names_from = age_band, values_from = share)
```

Two districts are not comparable as they stand

District	0-4	5-14	15-44	45+
Nord	**22%**	30%	35%	13%
Centre	15%	25%	42%	18%
Sud	**11%**	20%	44%	25%

Two districts are not comparable as they stand

- Nord has twice the share of under-fives that Sud has — and lesson 4 established that under-fives have three to four…

Direct standardisation — In Python (cont.)

```
cases = pd.read_csv("cholera-line-list-2024.v1.csv")

observed = (
    cases.groupby(["district", "age_band"]).size().rename("cases").to_frame()
    .join(population.set_index(["district", "age_band"])["population"])
)
observed["rate"] = observed["cases"] / observed["population"]

standard = population.groupby("age_band")["population"].sum()
standard_share = standard / standard.sum()

standardised = (
    observed["rate"].unstack()          # district x age_band
    .mul(standard_share, axis=1).sum(axis=1)
)
crude = (
```

Direct standardisation — In Python (cont.)

```
    cases.groupby("district").size()
    / population.groupby("district")["population"].sum()
)

comparison = pd.DataFrame({
    "crude_per_1000": 1000 * crude,
    "standardised_per_1000": 1000 * standardised,
})
comparison["difference"] = (
    comparison["standardised_per_1000"] - comparison["crude_per_1000"]
)
print(comparison.round(2))
```

Direct standardisation — In R

```r
observed <- cases |> count(district, age_band, name = "cases") |>
  left_join(population, by = c("district", "age_band")) |>
  mutate(rate = cases / population)

standard <- population |> summarise(pop = sum(population), .by = age_band) |>
  mutate(share = pop / sum(pop))

observed |>
  left_join(standard, by = "age_band") |>
  summarise(standardised = sum(rate * share), .by = district)
```

Direct standardisation

District	Crude	Standardised	Change
Nord	7.94	**7.25**	−0.69
Centre	6.47	6.60	+0.13
Sud	5.51	**5.95**	+0.44

Direct standardisation

- Nord falls, Sud rises, and the gap between them narrows from 2.43 to 1.30 per 1,000 — About half the apparent...

What the standard population is, and why it matters

- **The combined study population**, as above. Simple, defensible, and internal — the standardised rates are comparable...
- **A national population.** Lets you compare against other districts standardised the same way.
- **A published world standard** — the WHO or Segi world standard population. Lets you compare internationally, and...
- State the standard — A standardised rate is only interpretable against others standardised to the same population, and...

What the standard population is, and why it matters — In Python

```
print("Standard: combined population of the three districts, 145,000")
```

What the standard population is, and why it matters — In R

```
# Say it in the table caption, every time.
```

Indirect standardisation, and when you need it — In Python

```
standard_rates = observed.groupby("age_band").apply(
    lambda g: g["cases"].sum() / g["population"].sum()
)

expected = (
    population.assign(rate=population["age_band"].map(standard_rates))
    .assign(expected=lambda d: d["population"] * d["rate"])
    .groupby("district")["expected"].sum()
)
smr = cases.groupby("district").size() / expected
print(smr.round(3))
```

Indirect standardisation, and when you need it — In R

```
standard_rates <- observed |>
  summarise(rate = sum(cases) / sum(population), .by = age_band)

population |>
  left_join(standard_rates, by = "age_band") |>
  summarise(expected = sum(population * rate), .by = district) |>
  left_join(count(cases, district, name = "observed"), by = "district") |>
  mutate(smr = observed / expected)
```

Age is not the only confounder

- Standardising on age and then claiming the remaining difference is programme performance is the error this lesson. . .

Report the pair — Example

Cholera attack rate by district, weeks 1-16

District	Crude	Age-standardised	Population
Nord	7.94	7.25	48,000
Centre	6.47	6.60	62,000
Sud	5.51	5.95	35,000

Standardised directly to the combined population of the three districts. About half the crude Nord-Sud difference is age structure: Nord has 22% of its population under five against Sud's 11%, and under-fives have three to four times the attack rate of adults aged 15-44.

What comes next

- Standardisation removed one alternative explanation.

Read the full lesson, with runnable code

https://data-analysis.cassion.dev/courses/public-health-epidemiology/epi-06-age-standardisation