A household is classified by its worst strategy

Ten strategies, three severity phases, and the rule is the maximum rather than the sum. 18.2% of these households used an emergency strategy, and one district's answer to "not applicable" makes its prevalence figures wrong while leaving its classification intact.

Pierre Robentz Cassion

Lesson 3 of 8

Food Security Analysis and IPC — Coping is a sequence, not a score

What this lesson covers

- The module, and why it is not another index
- Not applicable is not no
- The district where the distinction was lost
- The partial module
- Why coping is measured separately from consumption
- What comes next

The module, and why it is not another index

Phase	Strategies	What they have in common
Stress	Sold household assets, spent savings, borrowed money, sold more animals than usual	Reduce the ability to deal with future shocks; reversible
Crisis	Sold productive assets, withdrew children from school, cut health spending	Reduce future productivity; hard to reverse
Emergency	Sold house or land, begged, sold last female breeding animals	Reduce future productivity **and** are close to irreversible

The module, and why it is not another index

- The classification is a maximum, not a sum, and not a count — A household that sold its last breeding animals once is. . .

The module, and why it is not another index — In Python (cont.)

```python
import pandas as pd

coping = pd.read_csv("livelihood-coping-2024.v1.csv")

PHASE = {
    "sold_household_assets": "stress",
    "spent_savings": "stress",
    "borrowed_money": "stress",
    "sold_more_animals_than_usual": "stress",
    "sold_productive_assets": "crisis",
    "withdrew_children_from_school": "crisis",
    "reduced_health_expenditure": "crisis",
    "sold_house_or_land": "emergency",
    "begged": "emergency",
    "sold_last_female_animals": "emergency",
}
```

The module, and why it is not another index — In Python (cont.)

```
RANK = {"none": 0, "stress": 1, "crisis": 2, "emergency": 3}

used = coping[list(PHASE)].eq("yes")
severity = pd.Series("none", index=coping.index)
for strategy, phase in PHASE.items():
    higher = used[strategy] & (severity.map(RANK) < RANK[phase])
    severity[higher] = phase

print((severity.value_counts(normalize=True) * 100).round(1))
```

The module, and why it is not another index — In R

```r
library(dplyr)

phase <- c(sold_household_assets = "stress", spent_savings = "stress",
           borrowed_money = "stress", sold_more_animals_than_usual = "stress",
           sold_productive_assets = "crisis",
           withdrew_children_from_school = "crisis",
           reduced_health_expenditure = "crisis",
           sold_house_or_land = "emergency", begged = "emergency",
           sold_last_female_animals = "emergency")
rank <- c(none = 0, stress = 1, crisis = 2, emergency = 3)
```

The module, and why it is not another index

Classification	Households	Share
None	339	16.0%
Stress	754	35.5%
Crisis	642	30.3%
Emergency	386	**18.2%**

The module, and why it is not another index

- 48.5% used a crisis or emergency strategy — That single figure is what the IPC evidence table takes from this module,. . .

Not applicable is not no — In Python

```
answers = coping["sold_last_female_animals"].value_counts()
print(answers)

applicable = coping["sold_last_female_animals"].isin(["yes", "no"])
print(f"prevalence among applicable: "
      f"{coping.loc[applicable, 'sold_last_female_animals'].eq('yes').mean():.1%}")
print(f"prevalence if n/a counted as no: "
      f"{coping['sold_last_female_animals'].eq('yes').mean():.1%}")
```

Not applicable is not no — In R

```r
coping |> count(sold_last_female_animals)

coping |>
  filter(sold_last_female_animals %in% c("yes", "no")) |>
  summarise(prevalence = mean(sold_last_female_animals == "yes"), n = n())
```

Not applicable is not no

Strategy	Among households it applies to	If n/a counted as no
Sold more animals than usual	**37.4%**	27.5%
Sold productive assets	19.8%	17.3%
Withdrew children from school	19.7%	16.5%
Sold last female animals	8.6%	5.8%

Not applicable is not no

- Ten points of difference on the first row — The denominator for a per-strategy prevalence is the households the...
- The household-level classification survives this and the per-strategy prevalence does not — because a maximum over...

The district where the distinction was lost — In Python

```
by_district = coping.groupby("district")["sold_last_female_animals"].agg(
    yes=lambda s: (s == "yes").sum(),
    applicable=lambda s: s.isin(["yes", "no"]).sum(),
    not_applicable=lambda s: (s == "not-applicable").sum(),
)
by_district["correct"] = by_district["yes"] / by_district["applicable"]
by_district["naive"] = by_district["yes"] / coping.groupby("district").size()
print((by_district[["correct", "naive"]] * 100).round(1))
```

The district where the distinction was lost — In R

```
coping |>
  summarise(yes = sum(sold_last_female_animals == "yes"),
            applicable = sum(sold_last_female_animals %in% c("yes", "no")),
            n = n(), .by = district) |>
  mutate(correct = yes / applicable, naive = yes / n)
```

The district where the distinction was lost

District	Correct denominator	Every household
Nord-Ouest	**12.3%**	6.8%
Artibonite	9.7%	5.3%
Sud	7.5%	4.5%
Centre	6.5%	6.4%

The district where the distinction was lost — In Python

```
print(coping.groupby("district")[list(PHASE)]
      .apply(lambda block: (block == "not-applicable").sum().sum()))
```

The district where the distinction was lost — In R

```
coping |> summarise(across(all_of(names(phase)),
                    ~ sum(.x == "not-applicable")), .by = district)
```

The district where the distinction was lost

- On the correct denominator Nord-Ouest is nearly twice Centre. On the wrong one they are the same — A district. . .

The partial module — In Python

```
emergency = [s for s, phase in PHASE.items() if phase == "emergency"]
partial = coping[emergency].eq("").all(axis=1) | coping[emergency].isna().all(axis=1)
print(f"{partial.sum()} households cannot be classified above crisis")
```

The partial module — In R

```
coping |> filter(if_all(c(sold_house_or_land, begged, sold_last_female_animals),
                        ~ is.na(.x))) |> nrow()
```

Why coping is measured separately from consumption — In Python

```
survey = pd.read_csv("food-security-survey-2024.v1.csv")
merged = survey.merge(coping, on="household_id", how="left", indicator=True)
print(merged["_merge"].value_counts())
```

Why coping is measured separately from consumption — In R

```
survey |> left_join(coping, by = "household_id") |>
  summarise(matched = sum(!is.na(district.y)), n = n())
```

Why coping is measured separately from consumption

- That is the argument for measuring both, and the next lesson is what happens when you do

What comes next

- Four instruments now exist on these households: consumption, hunger, coping behaviour and livelihood strategies.

Read the full lesson, with runnable code

https://data-analysis.cassion.dev/courses/food-security-ipc/fs-03-livelihood-coping-strategies