CASSION · FORMATION

Analyse d'enquêtes, échantillonnage et pondération

Analyser correctement une enquête en grappes à deux degrés — pondérations, strates, effet de plan et intervalles de confiance — selon les conventions déjà imposées par les EDS, les MICS et la méthodologie SMART.

Formateur	Pierre Robentz Cassion
Niveau	Intermédiaire
Heures apprenant	22 h
Enseignée en	Python · R
Mise à jour	28 juillet 2026
Secteurs	Santé publique · Nutrition · Sécurité alimentaire
Normes et méthodologies	Enquête SMART · Enquête démographique et de santé (EDS) · Enquête par grappes à indicateurs multiples (MICS) · Objectifs de développement durable (ODD)

Consulter et exécuter la formation en ligne sur data-analysis.cassion.dev/fr/cours/survey-analysis-sampling

Ce que vous saurez faire

— Reconstituer les pondérations d'un plan à partir d'une base de sondage, ajustement pour non-réponse compris, et prouver qu'elles totalisent la population

— Dire pourquoi une moyenne non pondérée issue d'une enquête en grappes est un chiffre faux, et de combien elle l'est sur une enquête donnée

— Calculer une taille d'échantillon pour estimer une prévalence, et énoncer la précision que le budget a réellement permis d'atteindre

— Estimer une proportion avec un intervalle de confiance ajusté du plan, en R comme en Python, et lire l'effet de plan dans le résultat

— Traiter explicitement la non-réponse et les grappes de remplacement plutôt que de les ignorer

— Refuser de désagréger au-delà de ce que l'échantillon permet, et dire dans un rapport où se situe cette limite

Prérequis

Aucun. Cette formation ne suppose aucune expérience préalable de la programmation.

Sommaire

À propos de cette formation

Une estimation d'enquête n'est pas un nombre. C'est un nombre, un intervalle et un plan de sondage qui a produit les deux — et laisser tomber l'un des deux derniers est la manière la plus courante dont les résultats d'enquête sont mal rapportés dans ce secteur.

C'est le cours le plus lourd du programme à ce jour, et le premier qui exige réellement des statistiques. Ce poids se justifie, car l'alternative coûte cher — une enquête ménage mobilise des véhicules, des enquêteurs et six semaines, et l'analyser comme un tableur d'observations indépendantes jette l'essentiel de ce qui a été acheté.

Presque tout s'exécute sur un jeu de données bâti pour ce cours — 996 entretiens de ménage tirés d'une base de 470 zones par sondage stratifié à probabilité proportionnelle à la taille. **Il livre la base et non les pondérations**, comme l'enquête SMART livre des mesures brutes plutôt que des scores z, parce que reconstituer les pondérations est l'exercice.

Trois constats de cette enquête donnent sa forme au cours. L'insécurité alimentaire est de 32,8 % non pondérée et de 29,1 % pondérée, car la plus petite strate est la plus suréchantillonnée et la plus touchée. L'effet de plan de cette estimation vaut 1,60, si bien que les 996 entretiens de l'enquête en valent environ 623 indépendants. Et l'effet de plan de l'accès à une source améliorée vaut **0,91** — inférieur à un, car le gain de la stratification l'emporte sur la perte due aux grappes — ce qui empêche de retenir « effet de plan » comme synonyme de « pénalité ».

Python et R, tout du long. Le paquet `survey` de R est l'implémentation de référence employée par ce secteur et le cours l'enseigne ; les exemples Python calculent les mêmes quantités directement, si bien que l'arithmétique reste visible au lieu d'être déléguée.

Vous devriez avoir suivi *Conception d'indicateurs et cadre logique*, ou avoir au moins pris l'habitude d'écrire un dénominateur avant tout calcul. Ce cours ajoute l'intervalle au nombre que l'autre vous a appris à définir.

Première partie

L'échantillon n'est pas la population

CHAPITRE 1

La moyenne de votre échantillon n'est la moyenne de rien

Leçon 1 sur 8 · 120 min

32,8 % contre 29,1 % sur les mêmes 996 entretiens, et 62,8 % contre 69,8 % sur les mêmes ménages. L'écart, c'est le plan de sondage, et ce n'est pas du bruit.

1.1 Deux nombres tirés d'un seul fichier

Prenez l'enquête ménage, comptez les ménages classés en insécurité alimentaire, divisez par le nombre de ménages. Cela fait 32,8 %.

Pondérez maintenant chaque ménage par le nombre de ménages qu'il représente dans la population, et le même fichier donne 29,1 %.

```python
import pandas as pd

survey = pd.read_csv("household-survey-2025.v1.csv")
frame = pd.read_csv("household-survey-frame-2025.v1.csv")

unweighted = (survey["food_insecure"] == "true").mean()
print(f"unweighted: {unweighted:.1%}")
```

```r
library(dplyr)
library(readr)

survey <- read_csv("household-survey-2025.v1.csv")
frame  <- read_csv("household-survey-frame-2025.v1.csv")

mean(survey$food_insecure == "true")
```

Les deux nombres sont calculés correctement. L'un est une estimation de la population et l'autre une description de qui a été interrogé, et **un seul des deux est ce que le rapport prétend rapporter.**

1.2 D'où vient l'écart

Rien à voir avec l'arithmétique. Tout à voir avec la façon dont l'échantillon a été tiré.

Strate	Ménages dans la base	Zones tirées	Ménages enquêtés	Insécurité alimentaire
Urbain	21 270	25	316	14,6 %
Rural accessible	28 958	25	337	36,2 %
Rural reculé	6 200	25	343	46,4 %

Lisez ensemble la première et la dernière colonne. **Le rural reculé représente 11 % de la population et 34 % de l'échantillon**, et c'est de loin la strate la plus touchée.

Une moyenne non pondérée des trois lignes surreprésente donc d'un facteur trois la strate la plus en difficulté. Les 32,8 % ne sont pas l'insécurité alimentaire du district ; c'est celle

d'une population dont un tiers des ménages sont ruraux reculés, et cette population n'existe pas.

```python
by_stratum = (
    survey.assign(insecure=survey["food_insecure"] == "true")
    .groupby("stratum")
    .agg(interviews=("insecure", "size"), rate=("insecure", "mean"))
)
frame_totals = frame.groupby("stratum")["households"].sum()

comparison = by_stratum.join(frame_totals.rename("frame_households"))
comparison["sample_share"] = comparison["interviews"] / comparison["interviews"].sum()
comparison["population_share"] = (
    comparison["frame_households"] / comparison["frame_households"].sum()
)
print(comparison.round(3))
```

```r
survey |>
  summarise(interviews = n(), rate = mean(food_insecure == "true"), .by = stratum) |>
  left_join(summarise(frame, frame_households = sum(households), .by = stratum),
            by = "stratum") |>
  mutate(sample_share     = interviews / sum(interviews),
         population_share = frame_households / sum(frame_households))
```

Strate	Part de l'échantillon	Part de la population
Urbain	31,7 %	37,7 %
Rural accessible	33,8 %	51,3 %
Rural reculé	34,4 %	11,0 %

Lorsque ces deux colonnes diffèrent, une estimation non pondérée est biaisée, et l'ampleur comme le sens du biais se déduisent entièrement du tableau.

1.3 Pourquoi concevoir le plan ainsi

La réaction immédiate est que l'échantillon a été mal tiré. Ce n'est pas le cas, et comprendre pourquoi fait l'essentiel de cette leçon.

Une allocation égale entre strates inégales est un choix délibéré et classique, et il achète quelque chose de précis — **une estimation exploitable pour chaque strate séparément.** Le rural reculé compte 11 % des ménages, si bien qu'un échantillon proportionnel à la population y aurait mis environ 110 entretiens : assez pour un chiffre national, très loin d'assez pour dire quoi que ce soit de la strate elle-même.

L'arbitrage est explicite.

— **L'allocation proportionnelle** donne la meilleure estimation nationale et des estimations infranationales faibles.

— **L'allocation égale** donne des estimations infranationales comparables et exige une pondération pour tout chiffre national.

Les EDS, les MICS et la plupart des évaluations humanitaires choisissent la seconde, car l'objet même de l'enquête est en général de comparer des lieux. **Les pondérations ne corrigent pas une erreur. Elles sont le prix du plan.**

1.4 Les estimations par strate n'ont pas besoin de pondérations

Le plan a une conséquence utile qu'il vaut la peine de voir tôt.

```
print(by_stratum["rate"].round(3))
```

```
survey |> summarise(rate = mean(food_insecure == "true"), .by = stratum)
```

À l'intérieur d'une strate, ce plan est **autopondéré** — chaque ménage avait la même probabilité de sélection — si bien que le taux non pondéré de la strate est déjà l'estimation. Les pondérations n'interviennent qu'au moment de combiner les strates.

C'est pourquoi un rapport peut légitimement présenter des chiffres non pondérés par strate à côté d'un total pondéré, et pourquoi le faire sans préciser lequel est lequel égare tout le monde.

1.5 Le sens n'est pas toujours le même

L'insécurité alimentaire baisse quand on pondère. L'accès à une source améliorée monte, et davantage.

```
for outcome in ["food_insecure", "improved_water_source"]:
    print(f"{outcome:24} unweighted {(survey[outcome] == 'true').mean():.1%}")
```

```
survey |>
  summarise(across(c(food_insecure, improved_water_source), ~ mean(.x == "true")))
```

Résultat	Non pondéré	Pondéré
Insécurité alimentaire	32,8 %	29,1 %
Source d'eau améliorée	62,8 %	69,8 %

Sept points sur le second. Il n'existe aucune règle générale voulant que les estimations non pondérées soient trop hautes ou trop basses : le biais va dans le sens où la strate suréchantillonnée diffère, et il diffère d'une ampleur différente pour chaque résultat.

On ne peut donc pas corriger un chiffre non pondéré à vue de nez, et un rapport qui en présente un avec la mention « non pondéré, le vrai chiffre est donc un peu plus bas » devine.

1.6 Que faire lorsqu'il n'y a pas de base de sondage

Il arrive d'hériter d'un jeu de données sans base et sans pondérations. Trois positions honnêtes, et aucune ne consiste à rapporter la moyenne non pondérée comme une estimation.

— **Rapporter par strate seulement**, si les strates sont enregistrées. Les chiffres infranationaux sont d'ailleurs souvent ce que le public voulait.

— **Reconstituer des pondérations approximatives** à partir d'une source de population externe, en documentant qu'elles sont approximatives.

— **Dire que l'enquête ne permet pas d'estimation de population.** C'est une vraie réponse, meilleure qu'un nombre qui sera cité pendant trois ans.

1.7 La suite

Les pondérations sont calculables, à partir de la base que ce jeu de données livre. La leçon suivante les construit depuis le début — probabilité de sélection à chaque degré, pondération de base, ajustement pour non-réponse — puis les prouve, en vérifiant qu'elles totalisent le nombre de ménages que la base déclare.

CHAPITRE 2

Construire les pondérations à partir de la base

Leçon 2 sur 8 · 120 min

Probabilité de sélection à chaque degré, la pondération de base qu'elle implique, l'ajustement pour non-réponse, et le contrôle qui prouve l'ensemble — des pondérations totalisant 56 428, exactement ce que la base déclare.

2.1 Une pondération est un nombre avec un seul sens

Une pondération de sondage, c'est le nombre d'unités de la population que représente l'unité tirée.

Cette phrase est toute la leçon. Le reste n'est que l'arithmétique pour y parvenir, et chaque contrôle est une façon de demander si les pondérations veulent toujours dire cela.

La pondération est l'inverse de la probabilité de sélection. Un ménage ayant une chance sur 60 d'être retenu représente 60 ménages ; un ménage ayant une chance sur 18 en représente 18. Rien de plus mystérieux.

2.2 Premier degré — probabilité proportionnelle à la taille

Vingt-cinq zones de dénombrement ont été tirées dans chaque strate, avec une probabilité proportionnelle aux ménages que chaque zone contient.

```
P(area i selected) = n_h * M_i / M_h
```

où `n_h` est le nombre de zones tirées dans la strate h (25), `M_i` les ménages de la zone i, et `M_h` les ménages de toute la strate.

```python
import pandas as pd

frame = pd.read_csv("household-survey-frame-2025.v1.csv")

stratum_households = frame.groupby("stratum")["households"].sum()
selected = frame[frame["selected"] == True].copy()

selected["p_stage1"] = (
    25 * selected["households"] / selected["stratum"].map(stratum_households)
)
print(selected[["ea_id", "stratum", "households", "p_stage1"]].head())
```

```r
library(dplyr)

stratum_households <- frame |> summarise(M_h = sum(households), .by = stratum)

selected <- frame |>
  filter(selected) |>
  left_join(stratum_households, by = "stratum") |>
  mutate(p_stage1 = 25 * households / M_h)
```

Une zone plus grande a plus de chances d'être retenue. C'est tout l'objet du tirage proportionnel à la taille — il place les entretiens là où sont les gens, ce qui évite qu'une enquête dépense un tiers de son budget dans des hameaux.

2.3 Second degré — un tirage fixe

Quatorze ménages ont été tirés dans chaque zone retenue, à partir de la liste de cette zone.

```
P(household selected | area selected) = m / M_i
```

avec $m = 14$.

```
selected["p_stage2"] = 14 / selected["households"]
selected["p_overall"] = selected["p_stage1"] * selected["p_stage2"]
print(selected.groupby("stratum")["p_overall"].describe()[["min", "max"]])
```

```
selected <- selected |>
  mutate(p_stage2 = 14 / households,
         p_overall = p_stage1 * p_stage2)

selected |> summarise(min = min(p_overall), max = max(p_overall), .by = stratum)
```

2.4 Les deux probabilités se compensent

Développez le produit et `M_i` disparaît.

```
P(household) = (n_h * M_i / M_h) * (m / M_i) = n_h * m / M_h
```

Tout ménage d'une strate a la même probabilité globale, quelle que soit la taille de sa zone. **Le plan est autopondéré à l'intérieur d'une strate.**

Exécutez le code et le minimum et le maximum de `p_overall` sont identiques dans chaque strate, ce qui est l'arithmétique qui le confirme. Ce n'est ni une coïncidence ni une commodité — c'est précisément pourquoi le tirage proportionnel à la taille va de pair avec un tirage fixe, dans les EDS, les MICS et la méthodologie SMART. La mesure de taille qui décide des zones visitées est annulée par celle qui décide du nombre de ménages tirés.

Deux conséquences à retenir. Une pondération qui varie à l'intérieur d'une strate dans un plan proportionnel à la taille signale un problème — en général une base dont la mesure de taille est périmée. Et la pondération de base est une propriété de la strate, non du ménage, si bien qu'elle se calcule en trois nombres.

```
base_weight = stratum_households / (25 * 14)
print(base_weight.round(1))
```

```
stratum_households |> mutate(base_weight = M_h / (25 * 14))
```

Strate	Ménages dans la base	Pondération de base
Urbain	21 270	60,8
Rural accessible	28 958	82,7
Rural reculé	6 200	17,7

Un ménage rural reculé en représente dix-huit ; un ménage rural accessible en représente quatre-vingt-trois. Ce rapport est toute la différence entre 32,8 % et 29,1 %.

2.5 L'ajustement pour non-réponse

Quatorze ménages ont été tirés par zone. Moins ont été enquêtés — 996 des 1 050 tirés. Les ménages enquêtés doivent porter ceux qui ne l'ont pas été.

```python
survey = pd.read_csv("household-survey-2025.v1.csv")

interviewed = selected.set_index("ea_id")["households_interviewed"]

survey["base_weight"] = survey["stratum"].map(base_weight)
survey["nr_adjustment"] = 14 / survey["ea_id"].map(interviewed)
survey["weight"] = survey["base_weight"] * survey["nr_adjustment"]

print(survey["weight"].describe()[["min", "max"]].round(1))
```

```r
survey <- survey |>
  left_join(select(selected, ea_id, households_interviewed), by = "ea_id") |>
  left_join(mutate(stratum_households, base_weight = M_h / (25 * 14)), by = "stratum") |>
  mutate(weight = base_weight * 14 / households_interviewed)

range(survey$weight)
```

Les pondérations vont désormais de 17,7 à 96,5.

Ajustez à l'intérieur de la zone, pas de la strate. Les ménages qui n'ont pas répondu dans une zone donnée ressemblent bien plus aux ménages ayant répondu dans cette zone qu'à la moyenne de la strate. Ajuster au niveau de la strate est plus simple et jette précisément l'information qui rend l'ajustement utile.

L'hypothèse est énoncée et elle n'est pas gratuite — **on suppose que les non-répondants ressemblent aux répondants de la même zone.** Là où vous avez des raisons d'en douter — des concessions fermées dans un quartier, un incident de sécurité un jour donné — dites-le dans les limites, car aucune pondération n'y remédie.

2.6 Le contrôle qui prouve toute la construction

```python
total = survey["weight"].sum()
frame_total = frame["households"].sum()
print(f"weights sum to {total:,.0f}; frame holds {frame_total:,}")
assert abs(total - frame_total) < 1
```

```r
c(weights = sum(survey$weight), frame = sum(frame$households))
```

56 428 et 56 428. Les pondérations totalisent exactement le nombre de ménages que la base déclare, ce que « combien d'unités celle-ci représente » doit signifier pour signifier quelque chose.

Faites ce contrôle à chaque fois. Il attrape une strate dont la pondération de base a employé le mauvais dénominateur, un ajustement pour non-réponse appliqué deux fois, et une jointure qui a perdu des zones — trois erreurs autrement invisibles, puisque l'estimation obtenue ressemble toujours à un pourcentage.

2.7 Pondérations normalisées, et quand s'en servir

Certains logiciels veulent des pondérations de moyenne un plutôt que totalisant la population.

```python
survey["weight_norm"] = survey["weight"] / survey["weight"].mean()
```

```r
survey <- survey |> mutate(weight_norm = weight / mean(weight))
```

Proportions et moyennes sont identiques dans les deux cas — la mise à l'échelle s'annule. **Les totaux, non.** Une pondération normalisée ne peut pas estimer « combien de ménages sont en insécurité alimentaire », seulement « quelle part ». Gardez la pondération à l'échelle de la population comme référence et dérivez la normalisée là où un outil l'exige.

2.8 Pondérations par personne et de sous-échantillon

Deux autres pondérations découlent de la même logique, et les deux sont régulièrement fausses.

Une pondération par personne est la pondération du ménage multipliée par sa taille, car un ménage de huit représente huit fois plus de personnes que de ménages.

```python
people = survey[survey["household_size"].notna()].copy()
people["person_weight"] = people["weight"] * people["household_size"]
print(f"estimated population {people['person_weight'].sum():,.0f}")
```

```r
people <- survey |>
  filter(!is.na(household_size)) |>
  mutate(person_weight = weight * household_size)

sum(people$person_weight)
```

Cela donne 333 336 contre 328 127 dans la liste de base — 1,6 % d'écart. Vingt ménages n'ont pas de taille enregistrée et sortent, et une liste de base est elle-même une estimation faite des mois plus tôt. **Un écart de cet ordre est normal et mérite d'être énoncé ; un écart de 20 % signale un problème.**

Une pondération de sous-échantillon s'applique quand une unité est choisie parmi plusieurs. Un enfant de moins de cinq ans a été mesuré par ménage, si bien qu'un enfant mesuré dans un ménage comptant trois enfants éligibles en représente trois.

```python
children = survey[survey["child_muac_mm"].notna()].copy()
children["child_weight"] = children["weight"] * children["children_under5"]
```

```r
children <- survey |>
  filter(!is.na(child_muac_mm)) |>
  mutate(child_weight = weight * children_under5)
```

Sautez cette multiplication et les enfants des grands ménages sont sous-représentés — or les grands ménages sont systématiquement plus pauvres, donc le biais va dans un sens prévisible. Sur cette enquête, la malnutrition aiguë par PB est de 13,3 % parmi les enfants mesurés et de **11,4 %** une fois les pondérations enfant appliquées.

2.9 Enregistrez les pondérations à côté des données

```
survey[["household_id", "ea_id", "stratum", "base_weight",
        "nr_adjustment", "weight"]].to_csv("outputs/weights.csv", index=False)
```

```
survey |>
  select(household_id, ea_id, stratum, base_weight, households_interviewed, weight) |>
  readr::write_csv(here::here("outputs", "weights.csv"))
```

Conservez les **composantes**, pas seulement la pondération finale. Quand quelqu'un demandera dans un an pourquoi un ménage compte pour 96 et un autre pour 18, la réponse est dans les colonnes et non dans votre souvenir d'un script.

2.10 La suite

Vous savez désormais produire une estimation de population. L'unité suivante demande ce qu'elle a coûté — combien de ménages exige une précision donnée, et pourquoi la formule de taille d'échantillon des manuels donne à peu près la moitié de ce qu'une enquête en grappes réclame réellement.

Deuxième partie

Ce que le plan achète

CHAPITRE 3

Quelle précision le budget a-t-il achetée ?

Leçon 3 sur 8 · 120 min

La formule de taille d'échantillon, les quatre nombres qu'elle exige, et pourquoi la réponse pour une enquête en grappes vaut environ le double de celle des manuels. Plus sa lecture à l'envers, celle dont on a le plus souvent besoin.

3.1 La question est toujours posée trop tard

La taille d'échantillon se décide dans une proposition, des mois avant toute analyse, et généralement par la personne qui rédige le budget. L'analyste en hérite.

Cette leçon fonctionne donc dans les deux sens. À l'endroit, pour dimensionner une enquête que vous concevez. **À l'envers, pour déterminer quelle précision permet l'enquête que vous avez déjà** — la version dont vous aurez le plus souvent besoin, et celle qui décide de ce que vous avez le droit de publier.

3.2 La formule, et les quatre nombres qu'elle contient

Pour une proportion :

```
n = z^2 * p * (1 - p) / d^2 * deff
```

Quatre entrées, et chacune est une décision que quelqu'un doit prendre.

— `z` — le niveau de confiance. 1,96 pour 95 %. Presque jamais modifié.

— `p` — la proportion attendue. Vous ne la connaissez pas, c'est l'objet de l'enquête. Prenez le tour précédent, une zone comparable, ou 0,5 si vous n'avez vraiment rien, car 0,5 maximise l'échantillon requis et constitue donc le pari prudent.

— `d` — la précision voulue, en marge absolue. C'est le nombre réellement négocié, et celui que personne n'écrit.

— `deff` — l'effet de plan. La leçon suivante lui est entièrement consacrée ; pour dimensionner, prenez la valeur du tour précédent ou 2,0 comme provisoire.

```python
import math

def sample_size(p, d, deff=1.0, z=1.96, response=1.0):
    n = z**2 * p * (1 - p) / d**2 * deff
    return math.ceil(n / response)

print(sample_size(0.30, 0.05))            # simple random
print(sample_size(0.30, 0.05, deff=2.4))  # cluster survey
print(sample_size(0.30, 0.03, deff=2.4))  # tighter precision
```

```r
sample_size <- function(p, d, deff = 1, z = 1.96, response = 1) {
  ceiling(z^2 * p * (1 - p) / d^2 * deff / response)
}
```

```
c(sample_size(0.30, 0.05),
  sample_size(0.30, 0.05, deff = 2.4),
  sample_size(0.30, 0.03, deff = 2.4))
```

p attendu	Précision	deff	Ménages nécessaires
30 %	±5 points	1,0	323
30 %	±5 points	2,4	775
30 %	±3 points	2,4	2 152

Trois enseignements de ce tableau.

La mise en grappes plus que double l'exigence. 323 devient 775 pour la même précision. Toute taille d'échantillon qui ne mentionne pas d'effet de plan dimensionne un sondage aléatoire simple, et personne n'en réalise dans ce secteur.

La précision coûte très cher. Passer de ±5 à ±3 points triple presque l'échantillon. La précision coûte selon le *carré* de l'amélioration, et c'est le fait le plus utile de cette leçon quand on vous demande un chiffre plus serré.

La non-réponse est un diviseur, pas un détail. Attendre 90 % de réponse signifie diviser par 0,9 — et cela s'applique aux ménages *tirés*, non aux ménages enquêtés, même distinction que dans la leçon sur la pondération.

3.3 La lecture à l'envers

Vous avez 996 entretiens et un effet de plan de 1,60. Quelle précision cela achète-t-il ?

```
def precision(n, p, deff=1.0, z=1.96):
    return z * math.sqrt(p * (1 - p) / n * deff)

print(f"+/- {precision(996, 0.291, deff=1.60):.1%}")
print(f"+/- {precision(343, 0.464, deff=1.60):.1%}")   # rural remote alone
```

```
precision <- function(n, p, deff = 1, z = 1.96) z * sqrt(p * (1 - p) / n * deff)

c(overall = precision(996, 0.291, deff = 1.60),
  remote  = precision(343, 0.464, deff = 1.60))
```

Environ ±3,6 points au total, et environ ±5,3 points pour le rural reculé seul.

Faites ce calcul avant de promettre quoi que ce soit. Il dit quelles comparaisons l'enquête peut trancher et lesquelles non, et c'est le calcul qui sous-tend la dernière leçon de ce cours.

3.4 Le nombre qui se négocie réellement

C'est sur `d` que porte le débat, et il se mène d'ordinaire dans la mauvaise monnaie — on discute de la taille d'échantillon alors que le désaccord porte sur la précision nécessaire de la réponse.

Recadrez. La précision n'a de sens que face à une décision.

— **Un seuil à franchir.** Si la question est de savoir si la MAG dépasse 15 % et que l'estimation est proche de 14 %, il faut un intervalle assez étroit pour tenir d'un seul côté de la ligne. C'est une exigence de précision chiffrée.

— **Un changement à détecter.** Détecter une amélioration de cinq points entre deux tours exige environ *quatre fois* l'échantillon nécessaire à estimer un niveau à ±5 points, car deux estimations portent chacune leur erreur.

— **Une comparaison entre groupes.** Même problème — deux intervalles, tous deux à resserrer.

```python
# Detecting a difference between two groups of equal size
def sample_per_group(p1, p2, deff=1.0, power_z=0.84, z=1.96):
    pbar = (p1 + p2) / 2
    n = (z + power_z) ** 2 * 2 * pbar * (1 - pbar) / (p1 - p2) ** 2
    return math.ceil(n * deff)

print(sample_per_group(0.30, 0.25, deff=2.4))
```

```r
sample_per_group <- function(p1, p2, deff = 1) {
  pbar <- (p1 + p2) / 2
  ceiling((1.96 + 0.84)^2 * 2 * pbar * (1 - pbar) / (p1 - p2)^2 * deff)
}

sample_per_group(0.30, 0.25, deff = 2.4)
```

Exécutez-le et le nombre est inconfortable. **Énoncez-le tôt et à voix haute**, car une enquête dimensionnée pour estimer un niveau puis utilisée pour affirmer un changement est de loin l'excès le plus fréquent du rapportage de ce secteur.

3.5 Grappes contre ménages par grappe

À budget fixe, vous arbitrez entre plus de grappes et plus de ménages dans chacune, et les deux ne se valent pas.

Plus de grappes vaut presque toujours mieux, car l'effet de plan croît avec le nombre de ménages *par grappe*. Trente grappes de dix valent bien mieux que dix grappes de trente pour les mêmes 300 entretiens.

La contre-pression est le coût — une grappe coûte un véhicule et une journée, un ménage coûte une heure. La convention SMART d'environ 30 grappes se situe exactement à cet arbitrage, et c'est un défaut raisonnable faute de mieux.

```python
for m in [8, 14, 20, 30]:
    deff = 1 + (m - 1) * 0.11
    clusters = math.ceil(775 * (deff / 2.4) / m)
    print(f"{m:>3} households x {clusters:>3} clusters -> deff {deff:.2f}")
```

```r
for (m in c(8, 14, 20, 30)) {
  deff <- 1 + (m - 1) * 0.11
  cat(sprintf("%3d households, deff %.2f\n", m, deff))
}
```

À une corrélation intra-grappe de 0,11 — la valeur mesurée sur cette enquête — quatorze ménages par grappe donnent un effet de plan dû aux grappes d'environ 2,4, et trente en donneraient 4,2. **Ce quatorze est une décision de conception, pas un hasard.**

3.6 Écrivez le calcul

Une taille d'échantillon est une argumentation, et elle a sa place dans le protocole d'enquête avec ses entrées visibles.

```
Indicator        Household food insecurity prevalence
Expected p       0.30 (2023 round, same instrument)
Precision        +/- 5 percentage points, 95% confidence
Design effect    2.0 (2023 round measured 1.9; rounded up)
Expected response 92%
Households       323 * 2.0 / 0.92 = 703, rounded to 25 clusters x 14 = 350 per
                 stratum, 1,050 in total across three strata
Rationale        Equal allocation, because each stratum must support its own
                 estimate. National figures require weighting.
```

Chaque nombre de ce bloc est contestable, et c'est l'intérêt. Un protocole disant « 1 050 ménages ont été enquêtés » appelle la question et ne peut pas y répondre.

3.7 La suite

Deux des quatre entrées étaient l'effet de plan, resté jusqu'ici un provisoire. La leçon suivante le mesure — d'où il vient, comment le calculer sur vos propres données, et pourquoi un résultat de cette enquête a un effet de plan inférieur à un.

CHAPITRE 4

L'effet de plan, et celui qui est sorti sous un

Leçon 4 sur 8 · 120 min

La mise en grappes coûte de la précision, la stratification en rachète, et l'effet de plan est le solde. 1,60 pour l'insécurité alimentaire et 0,91 pour l'accès à l'eau, sur les mêmes 996 entretiens.

4.1 Ce que c'est

L'effet de plan est un rapport.

```
deff = variance under this design / variance under simple random sampling
```

Un effet de plan de 2 signifie que votre estimation est aussi précise que l'aurait été un sondage aléatoire simple de moitié moindre. Il convertit un échantillon de 996 en **taille d'échantillon effective** — le nombre d'observations indépendantes que votre enquête vaut réellement.

```
effective n = n / deff
```

C'est cette seconde quantité qu'il faut mettre dans un rapport. « 996 ménages, échantillon effectif 623 » en dit plus que chacun des deux nombres, et c'est la réponse honnête à « quelle était la taille de votre enquête ».

4.2 Pourquoi les grappes vous coûtent

Les ménages d'une même zone de dénombrement se ressemblent. Ils partagent un point d'eau, un marché, une route, une récolte. Le deuxième ménage interrogé dans une zone vous apprend donc moins que le premier, et le quatorzième très peu.

La mesure de cette ressemblance est la **corrélation intra-grappe**, et elle se traduit directement en pénalité.

```
deff_clustering = 1 + (m - 1) * ICC
```

avec `m` le nombre d'entretiens par grappe.

```
import pandas as pd

by_area = (
    survey.assign(insecure=survey["food_insecure"] == "true")
    .groupby("ea_id")["insecure"]
    .agg(["mean", "size"])
)

p = (survey["food_insecure"] == "true").mean()
m = by_area["size"].mean()
between = by_area["mean"].var(ddof=0)
icc = (between - p * (1 - p) / m) / (p * (1 - p))
```

```
print(f"mean cluster size {m:.1f}, ICC {icc:.3f}, "
      f"clustering deff {1 + (m - 1) * icc:.2f}")
```

```
by_area <- survey |>
  summarise(rate = mean(food_insecure == "true"), n = n(), .by = ea_id)

p <- mean(survey$food_insecure == "true")
m <- mean(by_area$n)
icc <- (var(by_area$rate) * (nrow(by_area) - 1) / nrow(by_area) -
        p * (1 - p) / m) / (p * (1 - p))

c(m = m, icc = icc, deff = 1 + (m - 1) * icc)
```

Sur cette enquête — 13,3 entretiens par zone, ICC de 0,11, effet de plan dû aux grappes d'environ **2,4**.

Deux choses à retenir sur l'ICC.

— **C'est une propriété du résultat, pas de l'enquête.** La source d'eau est très regroupée — un village a un forage ou n'en a pas. La taille du ménage l'est à peine. Un effet de plan par résultat, toujours.

— **Les valeurs publiées sont la meilleure estimation disponible.** Les rapports EDS et SMART publient des effets de plan, et employer celui du tour précédent pour le même indicateur au même endroit vaut infiniment mieux que supposer 1.

4.3 Pourquoi la stratification en rachète

La stratification fait l'inverse. En forçant l'échantillon à couvrir chaque strate, elle supprime la possibilité d'un échantillon tombé surtout dans une seule — et cette possibilité fait partie de la variance d'un sondage aléatoire simple.

Le gain est grand exactement quand les strates diffèrent beaucoup sur le résultat. L'accès à une source améliorée va de 88 % en urbain à 41 % en rural reculé, et la stratification garantit que ces strates sont représentées en proportions fixes plutôt qu'au hasard.

4.4 Le solde, mesuré

Les deux forces tirent en sens inverse, donc **calculez l'effet de plan sur les données plutôt que de raisonner dessus.** Estimez la variance sous le plan réel et divisez par ce qu'aurait donné un sondage aléatoire simple.

```
import numpy as np

def design_effect(df, outcome, weight="weight"):
    p = np.average(df[outcome] == "true", weights=df[weight])
    total_w = df[weight].sum()

    variance = 0.0
    for _, stratum in df.groupby("stratum"):
        areas = stratum.groupby("ea_id").apply(
            lambda g: (g[weight] * ((g[outcome] == "true") - p)).sum()
        )
        n = len(areas)
        if n < 2:
            continue
        variance += n / (n - 1) * ((areas - areas.mean()) ** 2).sum()
```

```
    variance /= total_w**2

    srs = p * (1 - p) / len(df)
    return p, np.sqrt(variance), variance / srs

for outcome in ["food_insecure", "improved_water_source"]:
    p, se, deff = design_effect(survey, outcome)
    print(f"{outcome:24} p={p:.1%} se={se:.4f} deff={deff:.2f} "
          f"n_eff={len(survey) / deff:.0f}")
```

```
library(survey)

design <- svydesign(ids = ~ea_id, strata = ~stratum, weights = ~weight,
                    data = survey, nest = TRUE)

svymean(~I(food_insecure == "true"), design, deff = TRUE)
svymean(~I(improved_water_source == "true"), design, deff = TRUE)
```

Résultat	Estimation	Effet de plan	n effectif
Insécurité alimentaire	29,1 %	**1,60**	623
Source d'eau améliorée	69,8 %	**0,91**	1 098

4.5 Celui qui est sous un

Relisez la deuxième ligne. **Un effet de plan de 0,91 signifie que l'enquête est plus précise qu'un sondage aléatoire simple de même taille.**

Ce n'est pas une erreur, et c'est le résultat le plus utile de cette leçon. L'accès à l'eau diffère énormément entre strates — 88 %, 62 %, 41 % — et la stratification garantit que les trois entrent dans l'estimation en proportions fixes. La variance qu'un sondage aléatoire simple porte du fait de *ne pas* savoir combien de ménages urbains il attrapera est entièrement supprimée, et ce gain l'emporte sur la perte due aux grappes.

L'insécurité alimentaire diffère aussi entre strates, mais sa corrélation intra-zone est plus forte, si bien que les grappes l'emportent et le solde vaut 1,60.

« Effet de plan » n'est donc pas synonyme de « pénalité ». C'est le solde de deux forces opposées, et laquelle l'emporte dépend du résultat. Un rapport d'enquête citant un effet de plan unique pour toute l'enquête a moyenné ce qui ne doit pas l'être.

Remarquez aussi que le chiffre calculé plus haut pour les seules grappes valait 2,4, et que l'effet de plan complet du même résultat vaut 1,60. **La différence est ce qu'a acheté la stratification**, et cela mérite d'être rapporté quand vous défendez un plan de sondage.

4.6 D'où viennent les effets de plan quand vous n'en avez pas

Il vous en faut un avant que l'enquête existe, et il n'y a pas encore de données. Par ordre de préférence.

— **Le tour précédent de la même enquête**, même indicateur, même zone. Presque toujours disponible et presque toujours ignoré.

— **Un rapport d'enquête publié** pour le même indicateur dans un contexte comparable. Les rapports EDS tabulent les effets de plan ; les rapports de plausibilité SMART les énoncent.

— **Une convention.** Les enquêtes SMART supposent couramment 1,5 pour l'anthropométrie ; les évaluations humanitaires emploient souvent 2,0 pour les résultats au niveau ménage.

— **1,0** n'est jamais une hypothèse défendable pour une enquête en grappes, et un protocole qui l'emploie a sous-dimensionné l'enquête de moitié.

4.7 Rapportez-le

```python
summary = pd.DataFrame({
    "outcome": ["Food insecure", "Improved water source"],
    "estimate": ["29.1%", "69.8%"],
    "ci_95": ["25.5–32.7%", "67.0–72.5%"],
    "deff": [1.60, 0.91],
    "n": [996, 996],
    "n_effective": [623, 1098],
})
```

```r
tibble::tribble(
  ~outcome,                ~estimate, ~deff, ~n_effective,
  "Food insecure",           "29.1%",  1.60,          623,
  "Improved water source",   "69.8%",  0.91,         1098
)
```

Cinq colonnes, une ligne par indicateur, et cela a sa place en annexe de tout rapport d'enquête. Cela permet à un lecteur de juger l'estimation, de la comparer à une autre enquête et de dimensionner le tour suivant — trois choses impossibles à partir du seul pourcentage.

4.8 La suite

Vous avez un effet de plan, donc une erreur type, donc de quoi cesser de calculer tout cela à la main. La leçon suivante déclare le plan une fois au logiciel et laisse chaque estimation en hériter — `survey` sous R, et l'arithmétique équivalente en Python.

Troisième partie

Estimer correctement

CHAPITRE 5

Déclarez le plan une seule fois

Leçon 5 sur 8 · 120 min

Un objet de plan, et toute estimation ultérieure hérite des pondérations, des strates et des grappes. Le paquet survey de R, l'arithmétique équivalente en Python, et les quatre erreurs qui produisent en silence une simple moyenne.

5.1 Cessez de porter le plan à la main

Tout ce qui précède a été calculé explicitement, et c'était voulu — il faut savoir ce qu'est l'arithmétique. À partir d'ici, le plan doit être déclaré une fois et hérité, car l'alternative consiste à penser à appliquer pondérations, strates et grappes à chaque estimation d'un script de cinquante lignes, et l'estimation que vous oublierez est celle qui ira dans le résumé.

5.2 L'objet de plan sous R

```
library(survey)

design <- svydesign(
  ids     = ~ea_id,      # the primary sampling unit
  strata  = ~stratum,    # the stratification
  weights = ~weight,     # the weight built in lesson 2
  data    = survey,
  nest    = TRUE         # cluster ids are nested within strata
)

design
```

Quatre arguments et chacun porte.

— **`ids`** est l'unité primaire de sondage — la zone de dénombrement, non le ménage. Passer `~household_id` ici est l'erreur la plus fréquente de ce cours et produit silencieusement une variance de sondage aléatoire simple.

— **`strata`** retire la variance inter-strates, qui est le gain mesuré à la leçon précédente.

— **`weights`** rend l'estimation ponctuelle sans biais.

— **`nest = TRUE`** indique au paquet que les identifiants de zone ne sont uniques qu'à l'intérieur d'une strate. Omettez-le là où ils sont réutilisés entre strates et les zones seront fusionnées.

Ensuite, chaque estimation sort du plan.

```
svymean(~I(food_insecure == "true"), design, deff = TRUE)
svytotal(~I(food_insecure == "true"), design)
svyby(~I(food_insecure == "true"), ~stratum, design, svymean, vartype = c("se", "ci"))
svyciprop(~I(food_insecure == "true"), design, method = "logit")
```

`svyby` est le cheval de trait — n'importe quelle estimation, par n'importe quel regroupement, avec le plan propagé. Et `svyciprop` est ce qu'il faut pour une proportion : la leçon suivante explique pourquoi l'intervalle de `svymean` a la mauvaise forme près de 0 et de 1.

5.3 La même chose en Python

Python n'a pas d'équivalent de `survey` sur lequel ce secteur se soit fixé, si bien que les exemples calculent l'estimateur directement. C'est un coût et aussi un avantage — l'arithmétique reste visible.

```python
import numpy as np
import pandas as pd

class SurveyDesign:
    """Stratified two-stage design, ultimate-cluster variance."""

    def __init__(self, data, ids, strata, weights):
        self.data, self.ids, self.strata, self.weights = data, ids, strata, weights

    def mean(self, indicator):
        d = self.data
        y = indicator(d).astype(float)
        w = d[self.weights]
        p = np.average(y, weights=w)

        variance, total_w = 0.0, w.sum()
        for _, stratum in d.groupby(self.strata):
            residual = stratum[self.weights] * (indicator(stratum).astype(float) - p)
            totals = residual.groupby(stratum[self.ids]).sum()
            n = len(totals)
            if n < 2:
                continue
            variance += n / (n - 1) * ((totals - totals.mean()) ** 2).sum()
        se = np.sqrt(variance) / total_w

        srs = np.sqrt(p * (1 - p) / len(d))
        return {"estimate": p, "se": se, "deff": (se / srs) ** 2,
                "ci_low": p - 1.96 * se, "ci_high": p + 1.96 * se}

design = SurveyDesign(survey, ids="ea_id", strata="stratum", weights="weight")
print(design.mean(lambda d: d["food_insecure"] == "true"))
```

```r
# The R equivalent of the whole class above:
svymean(~I(food_insecure == "true"), design, deff = TRUE)
```

La classe Python fait trente lignes et l'appel R une seule. Cette asymétrie est réelle et c'est pourquoi l'analyse d'enquêtes se fait d'ordinaire en R dans ce secteur. **Employez R pour l'estimation d'enquête quand vous avez le choix** ; la voie Python est pour les équipes qui ne l'ont pas, et pour comprendre ce que fait l'appel R.

5.4 L'approximation par grappe ultime

Les deux implémentations emploient le même raccourci, et il vaut la peine de le nommer car il ressemble à une simplification sans en être une.

La variance se calcule à partir des **totaux des unités primaires de sondage**, en ignorant entièrement le second degré. C'est exact lorsque le premier degré est tiré avec remise, et légèrement conservateur sinon — cela surestime très légèrement la variance en ignorant la correction de population finie au second degré.

Être légèrement conservateur est le bon sens d'erreur, et c'est pourquoi tous les grands paquets font cela par défaut. `survey` l'appelle l'approximation par grappe ultime, et tous les

rapports EDS aussi.

5.5 Les quatre façons d'obtenir une simple moyenne par accident

Chacune produit un nombre d'allure correcte qui est une estimation de sondage aléatoire simple.

Passer le ménage comme grappe. `ids = ~household_id` dit que chaque ménage est sa propre unité primaire, donc qu'il n'y a pas de mise en grappes à prendre en compte, et l'erreur type s'effondre.

Oublier les pondérations. `svydesign(..., weights = NULL)` donne à chaque ménage le même poids, ce qui est le 32,8 % de la leçon 1.

Filtrer le tableau au lieu de sous-ensembler le plan. Celle-ci est subtile et fréquente.

```r
# WRONG: the design object no longer knows about the dropped strata
rural <- svydesign(ids = ~ea_id, strata = ~stratum, weights = ~weight,
                   data = filter(survey, stratum != "urban"), nest = TRUE)

# RIGHT
rural <- subset(design, stratum != "urban")
```

```python
# Same rule: subset the design, keeping the full strata structure in view
rural = SurveyDesign(survey[survey["stratum"] != "urban"],
                     ids="ea_id", strata="stratum", weights="weight")
```

Sous-ensembler un plan préserve la structure des strates et des grappes pour l'estimation de variance. Le reconstruire depuis un tableau filtré jette les zones sans ménage restant, ce qui change les degrés de liberté et peut produire en silence une strate à une seule grappe — où la contribution à la variance est indéfinie et discrètement écartée.

Calculer sur un tableau joint. Joignez l'enquête à quoi que ce soit en un-à-plusieurs et les pondérations deviennent fausses, car un ménage apparaît plusieurs fois. Agrégez au ménage avant d'estimer : c'est la leçon sur la granularité du cours de jointures qui revient dans un autre décor.

5.6 Des estimations autres que des proportions

```r
svymean(~household_size, design, na.rm = TRUE)
svytotal(~I(food_insecure == "true"), design)
svyquantile(~minutes_to_water, design, quantiles = c(0.5, 0.9), na.rm = TRUE)
svyratio(~I(food_insecure == "true"), ~I(children_under5 > 0), design)
```

```python
print(design.mean(lambda d: d["household_size"] > 7))
```

Deux remarques. **Un total pondéré est l'estimation que l'on veut le plus souvent et qu'on ne peut le plus souvent pas obtenir**, car elle exige des pondérations à l'échelle de la population — les pondérations normalisées de la leçon 2 donneraient un total de 996. Et `na.rm = TRUE` sur une moyenne d'enquête est une décision et non une commodité : cela suppose que les ménages manquants ressemblent aux ménages observés de leur strate, hypothèse identique à celle de l'ajustement pour non-réponse, et à énoncer une fois pour les deux.

5.7 Le plan du sous-échantillon

Les enfants mesurés forment un sous-échantillon avec sa propre pondération, et ils exigent leur propre objet de plan — non un filtre sur celui des ménages.

```
children <- subset(design, !is.na(child_muac_mm))
children <- update(children, child_weight = weight * children_under5)

child_design <- svydesign(ids = ~ea_id, strata = ~stratum,
                          weights = ~child_weight,
                          data = subset(survey, !is.na(child_muac_mm)), nest = TRUE)

svyciprop(~I(child_muac_mm < 125), child_design, method = "logit")
```

```
measured = survey[survey["child_muac_mm"].notna()].copy()
measured["child_weight"] = measured["weight"] * measured["children_under5"]

child_design = SurveyDesign(measured, ids="ea_id", strata="stratum",
                            weights="child_weight")
print(child_design.mean(lambda d: d["child_muac_mm"] < 125))
```

Les grappes sont les mêmes zones, donc la mise en grappes s'applique toujours. Seuls la pondération et la population changent.

5.8 Écrivez le plan à côté des estimations

```
Design           Stratified two-stage cluster
Strata           urban, rural-accessible, rural-remote
PSU              enumeration area (75 sampled, 25 per stratum)
Stage 1          PPS, size measure = households at frame listing
Stage 2          SRS, 14 households per area
Weights          base = M_h / (25 * 14), non-response adjusted within area
Variance         ultimate cluster, Taylor linearisation
Software         R survey 4.x / equivalent direct computation in Python
```

Sept lignes. Tout lecteur peut désormais reproduire vos erreurs types, et tout relecteur peut voir en dix secondes si vous avez fait les choses correctement — ce qu'on ne peut pas dire de la plupart des annexes d'enquête.

5.9 La suite

L'objet de plan suppose que tous les ménages tirés ont été enquêtés et que toutes les zones tirées ont été visitées. Ni l'un ni l'autre n'est vrai ici. La leçon suivante traite les 54 ménages non enquêtés et les deux zones qu'il a fallu remplacer.

CHAPITRE 6

Les ménages que vous n'avez pas eus

Leçon 6 sur 8 · 120 min

Cinquante-quatre entretiens manquants concentrés dans la strate la moins semblable aux autres, deux grappes de remplacement qui changent une probabilité que personne ne recalcule, et l'hypothèse sur laquelle repose tout ajustement.

6.1 La non-réponse n'est pas une donnée manquante

Une valeur manquante est une question sans réponse. La non-réponse est un **ménage tiré et jamais enquêté**, et la différence compte car la non-réponse ne laisse aucune ligne — le cas le plus difficile du cours de nettoyage, dans un décor où la base de sondage dit exactement combien de lignes devraient s'y trouver.

```python
selected = frame[frame["selected"] == True]

response = selected.groupby("stratum").agg(
    selected=("households_selected", "sum"),
    interviewed=("households_interviewed", "sum"),
)
response["rate"] = response["interviewed"] / response["selected"]
print(response)
```

```r
frame |>
  filter(selected) |>
  summarise(selected = sum(households_selected),
            interviewed = sum(households_interviewed), .by = stratum) |>
  mutate(rate = interviewed / selected)
```

Strate	Tirés	Enquêtés	Réponse
Urbain	350	316	90,3 %
Rural accessible	350	337	96,3 %
Rural reculé	350	343	98,0 %

996 sur 1 050. Un taux de réponse global de 95 % se lit comme excellent, et le fait utile n'est pas la moyenne — c'est que **le déficit se concentre en zone urbaine**, qui est aussi la moins touchée par l'insécurité alimentaire, de vingt points.

Une analyse non ajustée sous-représente donc exactement la strate qui tirerait l'estimation vers le bas. Le sens du biais se déduit de ce seul tableau, avant toute modélisation.

6.2 L'ajustement, et l'hypothèse qui le porte

L'ajustement de la leçon 2 gonfle la pondération de chaque ménage répondant pour qu'il porte les non-répondants de sa propre zone.

```python
survey["nr_adjustment"] = 14 / survey["ea_id"].map(
    selected.set_index("ea_id")["households_interviewed"]
```

```
)
print(survey.groupby("stratum")["nr_adjustment"].mean().round(3))
```

```
survey |> summarise(mean_adjustment = mean(14 / households_interviewed), .by = stratum)
```

L'hypothèse est que les non-répondants ressemblent aux répondants de la même zone. Elle n'est pas gratuite et n'est pas testable depuis l'enquête, donc elle a sa place dans la section des limites, en toutes lettres.

> La non-réponse a été ajustée à l'intérieur de chaque zone de dénombrement. La réponse urbaine était de 90,3 % contre 96 à 98 % en rural. Si les non-répondants urbains sont systématiquement mieux lotis que les répondants urbains — les concessions fermées et l'absence en journée le suggèrent — l'estimation ajustée reste légèrement trop élevée.

Remarquez ce que fait ce paragraphe. Il nomme le sens du biais résiduel après ajustement, ce qui est le maximum qu'une enquête puisse honnêtement offrir.

6.3 Ajustez au bon niveau

Trois niveaux sont possibles et ils ne se valent pas.

— **Dans la zone.** Ce que fait cette enquête. Le meilleur quand la réponse varie entre zones, ce qui est toujours le cas.

— **Dans la strate.** Plus grossier, et cela jette l'information selon laquelle une zone a eu quatre refus et sa voisine aucun.

— **Dans une classe de réponse.** Regroupez les zones selon un facteur prédictif de la réponse — densité urbaine, jour de marché, sécurité — et ajustez dans le groupe. Meilleur que la strate quand les zones sont petites, car une zone à trois entretiens donne un facteur d'ajustement instable.

Ce dernier point compte ici — une zone à très peu d'entretiens aboutis reçoit un gros ajustement issu d'un petit dénominateur, et une seule zone de ce type peut déplacer l'estimation d'une strate. Vérifiez.

```
weights_by_area = survey.groupby("ea_id")["weight"].first().sort_values()
print(weights_by_area.tail(3))
print(f"largest weight / smallest: {weights_by_area.max() / weights_by_area.min():.1f}x")
```

```
survey |>
  summarise(weight = first(weight), n = n(), .by = ea_id) |>
  arrange(desc(weight)) |>
  head(3)
```

Écrêtez ou signalez toute pondération très éloignée des autres. Une pondération valant trois fois la base de sa strate est une zone faisant le travail de trois, et elle gonfle la variance sans ajouter d'information. L'écrêtage est un jugement assorti d'un coût — il introduit un léger biais pour retirer une grande variance — et comme tout jugement de ce cours il s'écrit avec son seuil.

6.4 Les grappes de remplacement

Deux zones du rural reculé n'ont pas pu être atteintes et ont été remplacées. La base l'enregistre.

```python
replacements = frame[frame["replacement_for"].notna()]
print(replacements[["ea_id", "stratum", "households", "replacement_for"]])
```

```r
frame |> filter(!is.na(replacement_for)) |>
  select(ea_id, stratum, households, replacement_for)
```

C'est là qu'un plan cesse discrètement d'être celui qui a été documenté, et il existe deux traitements défendables.

Traiter le remplacement comme l'original. Le choix pragmatique, et ce que fait presque tout le monde. Cela suppose que la zone de remplacement est échangeable avec celle qu'elle remplace — raisonnable si le remplacement était aléatoire dans la strate, ce que le protocole devrait dire et ne dit souvent pas.

Traiter l'original comme une non-réponse au niveau de la grappe. Plus honnête et plus difficile : la strate compte en réalité 23 grappes répondantes et non 25, et ses pondérations se gonflent en conséquence.

```python
remote_areas = (frame["stratum"] == "rural-remote") & (frame["selected"] == True)
responding = int(remote_areas.sum()) - len(replacements)
print(f"25 selected, {responding} reached, cluster-level adjustment "
      f"{25 / responding:.3f}")
```

```r
c(selected = 25, reached = 23, adjustment = 25 / 23)
```

L'écart sur cette enquête est faible. **La raison de le faire explicitement n'est pas l'ampleur de la correction, c'est que les zones inaccessibles ne forment jamais un sous-ensemble aléatoire.** Une zone qu'on ne peut pas atteindre en février est reculée, ou peu sûre, ou inondée, et ce sont précisément les conditions associées aux résultats mesurés. La remplacer par une zone accessible retire systématiquement les pires cas.

> Signalez le remplacement même si vous le traitez comme échangeable. « Deux des 25 grappes du rural reculé ont été remplacées pour cause d'inaccessibilité ; les deux remplaçantes étaient accessibles, de sorte que l'estimation du rural reculé est probablement conservatrice » est une phrase dont le lecteur a besoin et qu'il ne peut pas reconstituer.

6.5 Les taux de réponse ont leur place dans le rapport

```python
report = response.reset_index()
report["non_response_bias_risk"] = ["high", "low", "low"]
report["adjustment"] = "within enumeration area"
print(report)
```

```r
tibble::tribble(
  ~stratum,            ~selected, ~interviewed, ~rate,  ~risk,
  "urban",                   350,          316, 0.903, "high",
  "rural-accessible",        350,          337, 0.963, "low",
```

```
  "rural-remote",           350,         343, 0.980, "low"
)
```

Quatre colonnes et une appréciation du risque. Un rapport d'enquête qui donne un taux de réponse global unique a caché la seule chose de la non-réponse qui affecte l'estimation — l'endroit où elle est tombée.

6.6 Quand la réponse est assez mauvaise pour s'arrêter

Il n'existe pas de seuil universel, et la recommandation honnête est comparative plutôt qu'absolue.

— **Au-dessus de 90 %** — ajustez et rapportez. Le biais résiduel est en général faible devant l'erreur d'échantillonnage.

— **De 80 à 90 %** — ajustez, rapportez, et décrivez le sens probable du biais résiduel. Ne comparez pas à un tour précédent au taux de réponse très différent sans le dire.

— **En dessous de 80 %** — l'ajustement porte trop de charge. Rapportez l'estimation avec une réserve bien visible, ou rapportez par strate là où la réponse était suffisante.

— **En dessous de 60 % dans une strate** — l'estimation de cette strate n'est pas utilisable, et le dire est le constat correct.

La comparaison la plus importante est celle avec le **tour précédent**. Une estimation qui a bougé de cinq points entre deux tours, alors que la réponse a bougé de quinze, a une explication alternative évidente qu'il faut traiter avant toute explication programmatique.

6.7 La suite

Vous avez une estimation, un plan et un compte rendu honnête de ce qui lui manque. La leçon suivante en fait ce qui se publie — un intervalle, de la bonne forme, avec le bon nombre de décimales, et la phrase qui l'entoure.

Quatrième partie

Ce que vous avez le droit de publier

CHAPITRE 7

L'intervalle, et la phrase qui l'entoure

Leçon 7 sur 8 · 120 min

Pourquoi l'intervalle des manuels donne 0,88 % pour une proportion qui ne peut pas descendre sous zéro, l'intervalle logit qui corrige cela, les 72 degrés de liberté dont vous disposez, et pourquoi deux intervalles qui se chevauchent ne signifient pas l'absence de différence.

7.1 À quoi sert l'intervalle

Une estimation d'enquête est une conjecture sur une population, faite à partir d'un échantillon. L'intervalle de confiance est la largeur honnête de cette conjecture, et publier l'estimation ponctuelle sans lui est le mésusage le plus répandu des données d'enquête dans ce secteur.

La lecture formelle — « 95 % des intervalles construits ainsi contiendraient la vraie valeur » — est correcte et inutile en réunion. C'est la lecture opérationnelle qu'il vous faut : **l'intervalle est l'ensemble des valeurs de population qui ne seraient pas surprenantes au vu de cet échantillon.** Si un seuil se situe à l'intérieur de votre intervalle, votre enquête ne peut pas dire de quel côté se trouve la population.

7.2 L'intervalle des manuels, et où il échoue

```
p +/- z * se
```

Correct au milieu de la plage et faux aux extrémités. Prenez la malnutrition aiguë sévère parmi les enfants mesurés.

```
p, se = 0.0217, 0.0066
print(f"Wald: {p - 1.96 * se:.2%} to {p + 1.96 * se:.2%}")
```

```
p <- 0.0217; se <- 0.0066
c(p - 1.96 * se, p + 1.96 * se)
```

2,17 %, intervalle **0,88 % à 3,46 %**. Symétrique, et la symétrie est le problème : une proportion ne peut pas être négative, et avec une estimation légèrement plus faible cet intervalle descendrait sous zéro et serait publié quand même.

7.3 L'intervalle logit

Passez à l'échelle des log-cotes, construisez l'intervalle là, revenez.

```
import math

logit = math.log(p / (1 - p))
se_logit = se / (p * (1 - p))
low, high = (logit - 1.96 * se_logit, logit + 1.96 * se_logit)
```

```
print(f"logit: {1 / (1 + math.exp(-low)):.2%} to {1 / (1 + math.exp(-high)):.2%}")
```

```
library(survey)
svyciprop(~I(child_muac_mm < 115), child_design, method = "logit")
```

1,19 % à 3,92 %. Asymétrique — plus étendu au-dessus de l'estimation qu'en dessous — ce qui est la forme correcte pour une petite proportion, et il ne peut pas sortir de l'intervalle 0-1 quelle que soit l'estimation.

Employez l'intervalle logit pour toute proportion sous 10 % environ ou au-dessus de 90 %. Au milieu, les deux concordent à la décimale près et cela n'a pas d'importance. `svyciprop(..., method = "logit")` sous R le fait directement ; les exemples Python de ce cours le calculent comme ci-dessus.

7.4 Les degrés de liberté dont vous disposez réellement

Le 1,96 suppose une loi normale, ce qui suppose beaucoup d'observations indépendantes. Dans une enquête en grappes, les unités indépendantes sont les **grappes**, non les ménages.

```
df = number of PSUs - number of strata
```

```
clusters = survey["ea_id"].nunique()
strata = survey["stratum"].nunique()
print(f"{clusters} clusters, {strata} strata, df = {clusters - strata}")
```

```
c(clusters = n_distinct(survey$ea_id), strata = n_distinct(survey$stratum))
degf(design)
```

Soixante-quinze grappes, trois strates, **72 degrés de liberté**, et t(0,975 ; 72) vaut 1,993 plutôt que 1,96. Une différence de 2 % sur la largeur de l'intervalle, ici sans importance, et c'est exactement le genre de chose qui cesse vite d'être sans importance — une enquête à douze grappes a neuf degrés de liberté et un multiplicateur de 2,26, soit 15 % de plus que l'approximation normale.

Employez le multiplicateur t, et rapportez les degrés de liberté. Cela ne coûte rien et cela dit au lecteur combien de grappes vous aviez sans qu'il ait à le demander.

7.5 Des intervalles qui se chevauchent ne constituent pas un test

Le contresens le plus lourd de conséquences de cette leçon.

```
for stratum in ["urban", "rural-accessible", "rural-remote"]:
    print(stratum)   # displaced households
```

Strate	Déplacés	Intervalle 95 %
Urbain	15,6 %	12,0 – 19,2 %
Rural accessible	8,0 %	4,7 – 11,3 %
Rural reculé	15,7 %	11,0 – 20,4 %

Les intervalles urbain et rural accessible ne se chevauchent pas, donc ces deux-là diffèrent. Urbain et rural reculé se chevauchent presque entièrement, donc ces deux-là ne diffèrent pas de façon détectable.

Vient maintenant le piège — **deux intervalles qui se chevauchent légèrement peuvent malgré tout différer significativement.** Comparer des intervalles est un test plus grossier que comparer la différence, car la différence a sa propre erreur type, plus petite que la somme des deux.

```
svyttest(I(displacement_status == "displaced") ~ I(stratum == "urban"),
         subset(design, stratum != "rural-remote"))
```

```
# The difference of two estimates, with its own standard error
diff = p_urban - p_rural
se_diff = math.sqrt(se_urban**2 + se_rural**2)
print(f"difference {diff:.1%}, 95% CI {diff - 1.99 * se_diff:.1%} to "
      f"{diff + 1.99 * se_diff:.1%}")
```

La règle — **pour comparer deux estimations, estimez la différence et mettez un intervalle dessus.** Lire deux intervalles est une étape de tri, pas une conclusion.

Remarquez aussi que la ligne Python ci-dessus suppose les deux estimations indépendantes, ce qui est vrai pour des strates séparées et faux pour deux sous-groupes d'une même strate — dans ce cas le terme de covariance compte, et `svyttest` ou `svycontrast` le traite correctement.

7.6 Arrondis et fausse précision

Un intervalle de ±3,6 points ne soutient pas trois décimales.

```
print(f"{p:.1%} (95% CI {low:.1%} to {high:.1%})")
```

```
sprintf("%.1f%% (95%% CI %.1f-%.1f)", 100 * p, 100 * low, 100 * high)
```

— **Une décimale** pour un pourcentage issu d'une enquête de cette taille. Deux laissent entendre une précision de 0,01 point, cent fois plus étroite que votre intervalle.

— **Arrondissez l'intervalle vers l'extérieur**, jamais vers l'intérieur. 25,47-32,71 devient 25,4-32,8.

— **N'arrondissez jamais l'estimation ponctuelle jusqu'à un seuil.** Une estimation de 14,96 % rapportée « 15 % » à côté d'un seuil d'urgence de 15 % a pris une décision de classification par arrondi.

7.7 La phrase

Le nombre, l'intervalle, le dénominateur, le plan. Une phrase, et c'est le livrable de tout ce cours.

> L'insécurité alimentaire des ménages est estimée à **29,1 %** (IC 95 % 25,5-32,7), sur 996 ménages répartis dans 75 grappes, pondérés selon la base de sondage et ajustés pour la non-réponse ; effet de plan 1,60, échantillon effectif 623.

Comparez avec ce qui se publie d'ordinaire.

> L'insécurité alimentaire touche 32,8 % des ménages.

La seconde phrase est non pondérée, sans intervalle, sans dénominateur et sans plan. Elle est aussi 3,7 points plus haute, et c'est elle qui sera citée.

7.8 Lire un intervalle face à un seuil

La question que ce secteur pose le plus souvent, et celle pour laquelle l'intervalle existe.

```
THRESHOLD = 0.15

if high < THRESHOLD:
    verdict = "below the threshold"
elif low > THRESHOLD:
    verdict = "above the threshold"
else:
    verdict = "cannot be distinguished from the threshold"
print(verdict)
```

```
dplyr::case_when(
  high < 0.15 ~ "below the threshold",
  low  > 0.15 ~ "above the threshold",
  TRUE        ~ "cannot be distinguished from the threshold"
)
```

La troisième branche est celle qu'on supprime, et c'est la réponse honnête la plus fréquente. **Une estimation de 14,2 % assortie d'un intervalle de 11,0-18,1 ne dit pas que la situation est sous le seuil d'urgence** ; elle dit que l'enquête ne peut pas trancher. Cette phrase est mal accueillie, exacte, et considérablement moins chère qu'une réponse dimensionnée sur un chiffre incapable de porter la décision.

7.9 La suite

L'intervalle dit ce que votre échantillon entier permet. Découpez-le en districts, groupes d'âge et sexes et chaque morceau permet bien moins. La dernière leçon dit où cesser de découper, et comment l'énoncer dans un tableau.

CHAPITRE 8

Où cesser de découper

Leçon 8 sur 8 · 120 min

Trois strates soutiennent une estimation. Strate par statut de déplacement donne neuf cellules dont la plus petite compte dix-huit ménages. Une règle fixée avant de regarder, et le tableau qui montre au lecteur où l'enquête s'est épuisée.

8.1 La demande qui arrive toujours

Le rapport d'enquête sort avec trois estimations par strate. Dans la semaine, quelqu'un le redemande par statut de déplacement. Puis par sexe du chef de ménage. Puis pour les ménages déplacés dirigés par une femme en zone rurale reculée, parce que c'est le groupe de la prochaine proposition.

Chaque demande est raisonnable et la dernière est sans réponse, et la difficulté est que **rien dans le logiciel ne refuse.** Chaque découpage produit un pourcentage.

8.2 Ce que fait le découpage

```python
for keys in [["stratum"],
             ["stratum", "displacement_status"],
             ["stratum", "main_livelihood"]]:
    cells = survey.groupby(keys).size()
    print(f"{' x '.join(keys):40} {len(cells):>3} cells, "
          f"smallest {cells.min():>3}, {(cells < 50).sum()} below 50")
```

```r
survey |> count(stratum) |> summarise(cells = n(), smallest = min(n))
survey |> count(stratum, displacement_status) |> summarise(cells = n(), smallest = min(n))
survey |> count(stratum, main_livelihood) |> summarise(cells = n(), smallest = min(n))
```

Désagrégation	Cellules	Plus petite	Sous 50
Strate	3	316	0
Strate × déplacement	9	18	5
Strate × moyen d'existence	18	6	9

Une variable supplémentaire fait passer la plus petite cellule de 316 ménages à

1. Une seconde la fait tomber à 6.

Et le décompte de ménages est la vue *optimiste*, car ce sont des observations groupées. Dix-huit ménages dans une cellule peuvent venir de quatre zones de dénombrement, soit quatre unités indépendantes et environ neuf degrés de liberté — pas dix-huit.

8.3 Comptez les grappes, pas seulement les ménages

```python
cells = survey.groupby(["stratum", "displacement_status"]).agg(
    households=("household_id", "size"),
    clusters=("ea_id", "nunique"),
)
print(cells.sort_values("clusters").head())
```

```r
survey |>
  summarise(households = n(), clusters = n_distinct(ea_id),
            .by = c(stratum, displacement_status)) |>
  arrange(clusters)
```

Une cellule issue de moins d'une dizaine de grappes ne peut pas soutenir une estimation de variance que vous voudriez défendre, quel que soit son nombre de ménages. C'est la version propre aux enquêtes de l'effectif minimal introduit par le cours sur les indicateurs, et elle est plus stricte, car la mise en grappes rend l'échantillon effectif plus petit que le décompte.

Deux cellules de cette enquête proviennent d'une seule grappe. Une estimation de variance à partir d'une grappe est indéfinie, et la plupart des logiciels écarteront la cellule en silence ou renverront zéro — une erreur type nulle sur une estimation de sous-groupe est toujours un défaut, et c'est toujours celui-là.

8.4 Fixez la règle avant de regarder

Trois seuils, décidés à l'avance et inscrits au plan d'analyse.

```python
MIN_HOUSEHOLDS = 50
MIN_CLUSTERS = 10
MAX_CI_WIDTH = 0.20      # 20 percentage points

def reportable(cell):
    return (cell["households"] >= MIN_HOUSEHOLDS
            and cell["clusters"] >= MIN_CLUSTERS
            and cell["ci_width"] <= MAX_CI_WIDTH)
```

```r
MIN_HOUSEHOLDS <- 50; MIN_CLUSTERS <- 10; MAX_CI_WIDTH <- 0.20

reportable <- function(d) {
  d$households >= MIN_HOUSEHOLDS & d$clusters >= MIN_CLUSTERS &
    d$ci_width <= MAX_CI_WIDTH
}
```

La troisième condition fait le vrai travail, et elle vaut mieux que les deux premières parce qu'elle porte sur la réponse plutôt que sur l'entrée. **Un intervalle plus large que 20 points ne peut pas distinguer « un problème grave » de « pas de problème »**, si bien que publier le point milieu appelle une décision que le nombre ne peut pas porter.

Là où un seuil du secteur est ce qui compte, fixez la largeur face à ce seuil : un intervalle devant tenir d'un seul côté de 15 % doit être plus étroit que la distance entre l'estimation et 15 %.

8.5 Supprimez, mais montrez la cellule

```python
by_cell = svyby_equivalent(survey, ["stratum", "displacement_status"], "food_insecure")
```

```python
by_cell["reported"] = by_cell.apply(reportable, axis=1)
by_cell.loc[~by_cell["reported"], ["estimate", "ci_low", "ci_high"]] = None
by_cell["note"] = by_cell["reported"].map(
    {False: "too few clusters to estimate", True: ""}
)
print(by_cell)
```

```r
by_cell <- svyby(~I(food_insecure == "true"), ~stratum + displacement_status,
                 design, svymean, vartype = "ci") |>
  mutate(reported = reportable(cur_data()),
         note = if_else(reported, "", "too few clusters to estimate"))
```

Ne supprimez jamais la ligne. Une cellule masquée qui affiche encore son effectif de ménages et un motif dit au lecteur que le groupe a été enquêté et que l'enquête n'a pas pu répondre pour lui. Une ligne supprimée se lit comme si le groupe n'existait pas, et la personne suivante reposera la question.

C'est la règle que le cours d'évaluation appliquait aux dimensions non mesurables et celui sur les indicateurs aux petits effectifs. Elle revient parce que c'est la même discipline sous-jacente — l'absence de preuve doit avoir une autre allure que l'absence.

8.6 Estimation par domaine, et ce qui ressemble à un filtre

Un point technique qui change la réponse.

```r
# WRONG: rebuilding the design from a filtered frame
displaced <- svydesign(ids = ~ea_id, strata = ~stratum, weights = ~weight,
                       data = filter(survey, displacement_status == "displaced"),
                       nest = TRUE)

# RIGHT: a domain estimate keeps the full design in view
svyby(~I(food_insecure == "true"), ~displacement_status, design, svymean)
```

```python
# The same rule: the variance calculation must still see every cluster,
# including those with no displaced households in them.
```

Un sous-groupe qui ne couvre pas toutes les grappes est un **domaine**, non une sous-population avec son propre plan. L'estimer depuis un tableau filtré perd les grappes qui n'en contiennent aucun élément, et ces grappes vides portent une information réelle sur la variance de la taille du sous-groupe. Le paquet `survey` de R traite cela correctement via `svyby` sur le plan complet ; une reconstruction filtrée non.

L'effet est en général modeste et toujours dans le même sens — **la version filtrée sous-estime l'erreur type**, c'est-à-dire le sens qui rend trop confiant.

8.7 Planifiez la désagrégation avant l'enquête

Le remède honnête à tout cela est en amont. Le cours sur les indicateurs le disait en général ; ici cela porte un chiffre.

Si un sous-groupe représente 12 % de la population et qu'il vous faut ±10 points dessus, la formule de taille d'échantillon appliquée à ce sous-groupe donne le nécessaire — et c'est en général bien plus que ce pour quoi l'enquête a été dimensionnée. C'est une conversation de conception, et les options sont réelles.

— **Suréchantillonner le sous-groupe**, ce à quoi sert la stratification. Cette enquête a suréchantillonné le rural reculé précisément pour pouvoir en rendre compte.

— **Accepter un intervalle plus large** pour ce sous-groupe, en le disant à l'avance.

— **Abandonner l'exigence**, en écrivant dans le protocole que l'enquête n'en rendra pas compte.

Ce qu'on ne peut pas faire, c'est décider après. Le jour où les données existent, les cellules sont ce qu'elles sont, et un tableau montrant huit cellules masquées sur dix-huit est la forme visible d'une conversation de conception qui n'a pas eu lieu.

8.8 Le tableau à publier

```python
final = by_cell[["stratum", "displacement_status", "households", "clusters",
                 "estimate", "ci_low", "ci_high", "note"]]
final.to_csv("outputs/tables/food_insecurity_by_stratum_displacement.csv", index=False)
```

```r
readr::write_csv(by_cell,
  here::here("outputs", "tables", "food_insecurity_by_stratum_displacement.csv"))
```

Strate	Statut	Ménages	Grappes	Estimation	IC 95 %	Note
Urbain	Résident	246	25	13,9 %	10,2-18,7	
Urbain	Déplacé	49	18	—	—	trop peu de ménages
Rural reculé	Résident	279	25	46,1 %	40,0-52,3	
Rural reculé	Retourné	24	12	—	—	trop peu de ménages

Sept colonnes, et les deux vides sont aussi instructives que les quatre remplies. Un lecteur voit exactement où l'enquête s'est épuisée, ce qui fait la différence entre un tableau qui répond aux questions et un tableau qui en engendre.

8.9 Ce que ce cours vous laisse

Vous savez reconstituer des pondérations à partir d'une base et prouver qu'elles totalisent la population, dimensionner une enquête face à une exigence de précision et lire cette exigence à l'envers sur une enquête dont vous héritez, mesurer un effet de plan au lieu de le supposer, estimer avec le plan propagé, traiter explicitement la non-réponse et les remplacements, publier un intervalle de la bonne forme, et dire où l'échantillon cesse de soutenir un découpage.

C'est tout ce sur quoi un analyste d'enquête est jugé, et c'est l'essentiel du module 3. Le dernier cours du module, *Données de routine et DHIS2*, revient aux systèmes de rapportage agrégé auxquels ce programme emprunte depuis le module 2 — éléments de données, combinaisons de catégories, hiérarchie des unités d'organisation — et répond à la question que provoque toujours un taux de couverture : qu'a-t-on exactement compté, et par qui ?

À propos de cette édition

Ce PDF est produit à partir de la même source que la version web de la formation ; les deux ne peuvent donc pas diverger. L'édition en ligne comporte le code exécutable, les jeux de données téléchargeables et les corrections publiées depuis la production de ce fichier.

> Tous les jeux de données cités dans cette formation sont synthétiques ou entièrement anonymisés. Aucun enregistrement ne renvoie à une personne réelle.

Cette formation est publiée en anglais et en français. L'édition française est une adaptation professionnelle reprenant la terminologie employée par l'UNICEF, l'OMS, le PAM, OCHA et les ministères nationaux — et non une traduction littérale.

Consulter et exécuter la formation en ligne sur data-analysis.cassion.dev/fr/cours/survey-analysis-sampling

Généré le 28 juillet 2026.