Elements of Sampling Survey

Longhai Li

2026-11-23

Table of contents

Table of Contents 3
Preface . . . 3

1 Introduction to Sampling Survey 4
1.1 A Bad Sampling Example . . . 4
1.2 Elements of Sampling Problems . . . 5
1.2.1 Basic definitions . . . 5
1.2.2 Target and sampled populations . . . 5
1.3 Selection Bias . . . 6
1.4 Measurement Error . . . 6
1.5 Sampling and Nonsampling Errors . . . 7
1.5.1 Why sample at all? . . . 7
1.6 Overview of Probability Sampling and Estimation Techniques . . . 7
1.7 A Framework for Probability Sampling . . . 10
1.7.1 The population and its samples . . . 11
1.7.2 Probability sampling . . . 11
1.7.3 The values y_i . . . 11
1.7.4 Inclusion probabilities and sampling weights . . . 12
1.7.5 Population quantities . . . 12
1.7.6 Statistics and sampling distributions . . . 13
1.7.7 Bias, variance and mean squared error . . . 14
1.7.8 Unbiased, precise and accurate . . . 15
1.7.9 Summary . . . 16

2 Data Analysis with R 17
2.1 Foundations . . . 17
2.1.1 Basic R objects . . . 17
2.2 Importing a dataset . . . 19
2.2.1 Displaying the full dataset . . . 19
2.2.2 Simple summaries . . . 20
2.3 The native pipe . . . 20
2.4 Data manipulation with dplyr . . . 21
2.5 Tables with gt . . . 23
2.5.1 Spanning headers grouped by statistic . . . 23
2.5.2 Spanning headers grouped by variable . . . 24
2.5.3 Automatic summary tables with gtsummary . . . 25
2.6 Graphics . . . 25
2.6.1 Base R graphics . . . 25
2.6.2 Saving and re-inserting an external plot . . . 27

2.6.3 Advanced visualization with ggplot2 . . . 28
2.7 Writing your own function . . . 29

3 Simple Random Sampling **31**
3.1 SRS and SRSWR . . . 31
3.1.1 Inclusion probabilities . . . 31
3.2 Estimators for Simple Random Sampling . . . 32
3.2.1 Estimating the population mean . . . 32
3.2.2 Estimating the population total . . . 33
3.2.3 Coefficient of variation . . . 33
3.2.4 Estimating a proportion . . . 33
3.3 Example: The Census of Agriculture . . . 34
3.3.1 Step-by-step calculation . . . 34
3.3.2 A function for repeated use . . . 36
3.3.3 Comparing with the true values . . . 38
3.4 Confidence Intervals . . . 39
3.4.1 Normality of the sample mean . . . 39
3.4.2 How large must n be for normality? . . . 39
3.5 Choosing the Sample Size . . . 40
3.5.1 Specifying the tolerable error . . . 40
3.5.2 Solving for n . . . 40
3.6 Randomization Theory for SRS . . . 42
3.6.1 Moments of Z_i . . . 42
3.6.2 Unbiasedness of $\bar{y}$ . . . 42
3.6.3 Variance of $\bar{y}$ . . . 42
3.6.4 Unbiasedness of s^2 . . . 43
3.7 A Simulation Demonstration of SRS Inference . . . 43
3.7.1 True values . . . 44
3.7.2 One SRS . . . 44
3.7.3 Repeating SRS 5000 times . . . 45
3.7.4 Empirical coverage rate of CIs . . . 48
3.8 Shinylive App for Illustrating SRS Theory . . . 49

4 Stratified Random Sampling **50**
4.1 What Is Stratified Sampling? . . . 50
4.2 Estimation with Stratified Samples . . . 51
4.2.1 Population quantities . . . 51
4.2.2 SRS estimators within each stratum . . . 52
4.2.3 Estimators for the population total and mean . . . 52
4.2.4 Variances and standard errors . . . 52
4.2.5 Confidence intervals . . . 53
4.3 Example: The Nelchina Caribou Herd . . . 53
4.4 Stratified Sampling for Proportions . . . 55
4.5 Alternative Formulas for $\overline{y}_{\text{str}}$ . . . 56
4.5.1 Weighting by population fractions instead of sample fractions . . . 57
4.5.2 Weighting each unit by its sampling weight . . . 57

4.6 Analyzing Stratified Data in R . . . 58
4.6.1 Example: Farm acreage by census region (agstrat.csv) . . . 58
4.7 Efficiency of Proportional Allocation . . . 63
4.7.1 Decomposition of the population variance . . . 63
4.7.2 The coefficient of determination R^2 . . . 64
4.7.3 Comparing the variances of proportional allocation and SRS . . . 65
4.8 Optimal Allocation . . . 66
4.8.1 The cost function . . . 66
4.8.2 Derivation . . . 66
4.8.3 The optimal allocation . . . 67
4.8.4 Example: Neyman allocation for the caribou survey . . . 67
4.8.5 Example: Seal holes (seals.csv) . . . 68
4.8.6 Example: Optimal allocation for polls . . . 69
4.8.7 Example: Dollar stratification . . . 69
4.9 Choosing Stratification Variables . . . 70
4.10 Simulation: Efficiency of Stratification and Allocation . . . 71
4.10.1 Stratifying by region . . . 71
4.10.2 Stratifying by acres82 . . . 76
4.11 Interactive Demonstration: Stratification by `acres82` or `region` . . . 79

5 Ratio and Regression Estimation **80**
5.1 Introduction . . . 80
5.2 Functions and packages for Analyzing Data . . . 80
5.2.1 General functions for ratio and regression estimation . . . 80
5.3 Ratio Estimation . . . 81
5.3.1 Theory . . . 81
5.3.2 Example: Population of France (by hand) . . . 84
5.3.3 Example: Farm Acreage, 1992 from 1987 (agsrs.csv) . . . 85
5.3.4 Example: Cherry Trees (cherry.csv) . . . 87
5.4 Regression Estimation . . . 93
5.4.1 Theory . . . 93
5.4.2 Example: Cherry Trees (cherry.csv) . . . 95
5.4.3 Example: Photo Counts of Dead Trees . . . 98
5.4.4 Shinylive App Illustrating Ratio Estimation and Regression Estimation . . . 102
5.5 Post-stratification . . . 103
5.5.1 Theory . . . 103
5.5.2 Example: A Nurses Survey (by hand) . . . 106
5.5.3 Example: agsrs_syn.csv . . . 107
5.5.4 Comparing the Two Post-stratifications . . . 112
5.5.5 Interactive Demonstration: Post-stratifying a Simple Random Sample . . . 113
5.6 Domain Estimation . . . 113
5.6.1 Theory . . . 113
5.6.2 Example: agsrs.csv . . . 115
5.6.3 Example: Teacher Aspirations (teacher data) . . . 116
5.7 Prediction with Machine-Learning Models . . . 117

5.8 Simulation study with agpop.csv . . . 117
5.8.1 Information of population data . . . 119
5.8.2 Simulation Studies . . . 120

6 Cluster Sampling **123**
6.1 What Is Cluster Sampling? . . . 123
6.1.1 Cluster sampling versus stratified sampling . . . 123
6.2 Notation for Cluster Sampling . . . 125
6.2.1 Population quantities . . . 125
6.2.2 Sample quantities . . . 126
6.2.3 Sampling weights . . . 127
6.3 One-Stage Cluster Sampling with Equal Cluster Sizes . . . 127
6.3.1 Example: GPA in a dormitory . . . 127
6.4 Comparing Cluster Sampling with SRS . . . 129
6.4.1 Intraclass correlation and adjusted R^2 . . . 129
6.5 One-Stage Cluster Sampling with Unequal Cluster Sizes . . . 130
6.5.1 The unbiased estimator . . . 130
6.5.2 The ratio estimator . . . 131
6.5.3 A toy example: why the unbiased estimator can be bad . . . 132
6.5.4 Example: algebra class test scores (algebra.csv) . . . 132
6.6 Two-Stage Cluster Sampling . . . 134
6.6.1 The unbiased estimator . . . 134
6.6.2 The ratio estimator . . . 135
6.6.3 Example: American coot egg volumes (coots.csv) . . . 135
6.6.4 Example: farm acreage, with states as clusters (agpop.csv) . . . 137
6.7 Interactive Demonstration: Clustering by State or at Random . . . 141

7 Unequal Probability Sampling **142**
7.1 Functions and Packages . . . 142
7.2 The Hansen–Hurwitz Estimator . . . 142
7.2.1 Theory . . . 142
7.2.2 Computation Example: Revenue from AI Token Usage . . . 144
7.2.3 Example: Physicians in U.S. Counties (`statepop.csv`) . . . 146
7.2.4 The HH Estimator of the Total in Terms of ψ_i . . . 152
7.3 Comparison of the HH, Ratio, and Average Estimators of the Total . . . 154
7.3.1 Theory . . . 154
7.3.2 Toy Example: Four Supermarkets . . . 157
7.3.3 Simulation Study using `agpop` Data . . . 158
7.3.4 Shinylive App Comparing SRS and UPSWR . . . 159
7.4 One-Stage Cluster Sampling with PPS . . . 159
7.4.1 Theory . . . 159
7.4.2 Example: Falls in Long-Term Care Homes . . . 163
7.5 Two-Stage Cluster Sampling with PPS . . . 165
7.5.1 Theory . . . 165
7.5.2 Example: Falls in Long-Term Care Homes, Two Stages . . . 166

7.6 UPS Without Replacement . . . 168
7.6.1 Review of UPSWR and the Horvitz–Thompson Estimator . . . 168
7.6.2 Selecting a Sample with Given Inclusion Probabilities . . . 170
7.6.3 Example: Acres of Farmland in Counties (agpop.csv) . . . 174
7.7 Summary and Discussion on UPS . . . 177
7.7.1 Summary of the Designs and Estimators . . . 177
7.7.2 When to Use Unequal Probability Sampling . . . 177
7.7.3 Practical Issues of Sampling Without Replacement . . . 178

Appendices **179**

A List of Important Formulas **179**
A.1 General Notation . . . 179
A.2 Simple Random Sampling . . . 179
A.3 Stratified Random Sampling . . . 180
A.4 Ratio and Regression Estimation . . . 181
A.5 Cluster Sampling . . . 182
A.6 Unequal Probability Sampling . . . 183

B List of Figures, Tables, and Apps **185**
B.1 Shinylive Apps . . . 185
B.2 List of Figures . . . 185
B.2.1 Introduction to Sampling Survey . . . 185
B.2.2 Data Analysis with R . . . 185
B.2.3 Simple Random Sampling . . . 185
B.2.4 strata . . . 186
B.2.5 ratio . . . 186
B.2.6 cluster . . . 186
B.2.7 ups . . . 186
B.3 List of Tables . . . 187
B.3.1 Data Analysis with R . . . 187
B.3.2 strata . . . 187
B.3.3 ratio . . . 187
B.3.4 cluster . . . 187
B.3.5 ups . . . 187

C Estimating Functions Source Code **188**

Table of Contents

Preface

Statistical inference in the real world almost always deals with bounded, finite populations—whether we are estimating agricultural yields, public health metrics, or economic indicators. This book explores the mathematical theory and practical mechanics of drawing reliable samples from those populations, examining how intelligent survey design directly dictates the precision and cost of our conclusions. The chapters cover a comprehensive suite of methodologies, including simple random sampling, stratified random sampling, cluster and systematic sampling, probability proportionate to size (PPS) sampling, and powerful auxiliary-variable strategies like difference, ratio, and regression estimation.

This book is part of the course materials for STAT 348 Sampling Techniques, which also has the slides, the Shinylive apps and the syllabus.

Key Features

- **Custom R Functions:** Rather than relying exclusively on pre-built packages, this text emphasizes writing custom functions to analyze survey data, helping students learn how R works under the hood.
- **Modern R Workflows:** Data manipulation and presentation leverage the modern R ecosystem, utilizing `dplyr` for clean wrangling, `ggplot2` for advanced visualization, and `gt` for publication-ready mathematical tables.
- **Simulation-Based Learning:** Extensive simulation studies demonstrate the practical differences between sampling schemes, with a core focus on how predictive auxiliary variables improve precision and reduce variance.

1 Introduction to Sampling Survey

A sample survey collects information about a population by measuring only part of it. This chapter introduces the main ideas. We start with a survey that went badly wrong, then define the basic terms, describe the errors that can spoil a survey, and sketch the main designs studied in the rest of the book. The second half sets up the framework of probability sampling that all later chapters use: samples and their probabilities, inclusion probabilities and sampling weights, population quantities, sampling distributions, and bias, variance and mean squared error. The material corresponds to Chapter 1 and Section 2.2 of Lohr, and Chapters 2 and 3.1–3.4 of Scheaffer et al.

1.1 A Bad Sampling Example

Shere Hite's book *Women and Love: A Cultural Revolution in Progress* (1987) reported a number of widely quoted results:

- 84% of women are "not satisfied emotionally with their relationships" (p. 804);
- 70% of all women "married five or more years are having sex outside of their marriages" (p. 856);
- 95% of women "report forms of emotional and psychological harassment from men with whom they are in love relationships" (p. 810);
- 84% of women report forms of condescension from the men in their love relationships (p. 809).

The book was widely criticized in newspaper and magazine articles throughout the United States. The *Time* magazine cover story "Back Off, Buddy" (October 12, 1987) called the conclusions of the study "dubious" and "of limited value". The problem was not the arithmetic but the way the data were collected:

- **The sample was self-selected.** The recipients of the questionnaire decided whether they would be in the sample. Hite mailed 100,000 questionnaires and only 4.5% were returned.
- **The questionnaires went to particular groups**, such as professional women's groups, counseling centers, church societies and senior citizens' centers, whose members may hold views different from those of other women in the United States.
- **The questionnaire was very long.** It had 127 essay questions, most with several parts. Who will tend to return such a survey? Probably women with strong feelings about the subject, who may differ systematically from those who did not reply.
- **Many questions were vague** (what counts as "love"?), which makes the percentages hard to interpret.
- **Many questions were leading**, suggesting to the respondent which answer to give.

A large number of responses (4,500) did not make up for any of these problems. The rest of this chapter names the problems precisely, and the rest of the book shows how a properly designed probability sample avoids the first two.

1.2 Elements of Sampling Problems

1.2.1 Basic definitions

- **Observation unit.** An object on which a measurement is taken; also called an **element**. In studies of human populations, the observation units are often individuals.
- **Target population.** The complete collection of observations we want to study. Defining the target population is often harder than it seems. Should a political poll target all adults eligible to vote, all registered voters, or everyone who voted in the last election?
- **Sample.** A subset of a population.
- **Sampled population.** The collection of all possible observation units that might have been chosen in the sample: the population from which the sample was actually taken.
- **Sampling unit.** A unit that can be selected for a sample. We may want to study individuals but have no list of them. Instead we select households, which are then the sampling units, while the individuals remain the observation units.
- **Sampling frame.** A list, map or other specification of the sampling units from which the sample may be selected, for example a list of telephone numbers, street addresses or farms.

1.2.2 Target and sampled populations

In an ideal survey the sampled population equals the target population. This ideal is rarely met (Figure 1.1). Some members of the target population are not in the sampling frame. Some who are in the frame cannot be reached, refuse to respond, or are not capable of responding. And the frame may contain units that are not eligible for the survey at all.

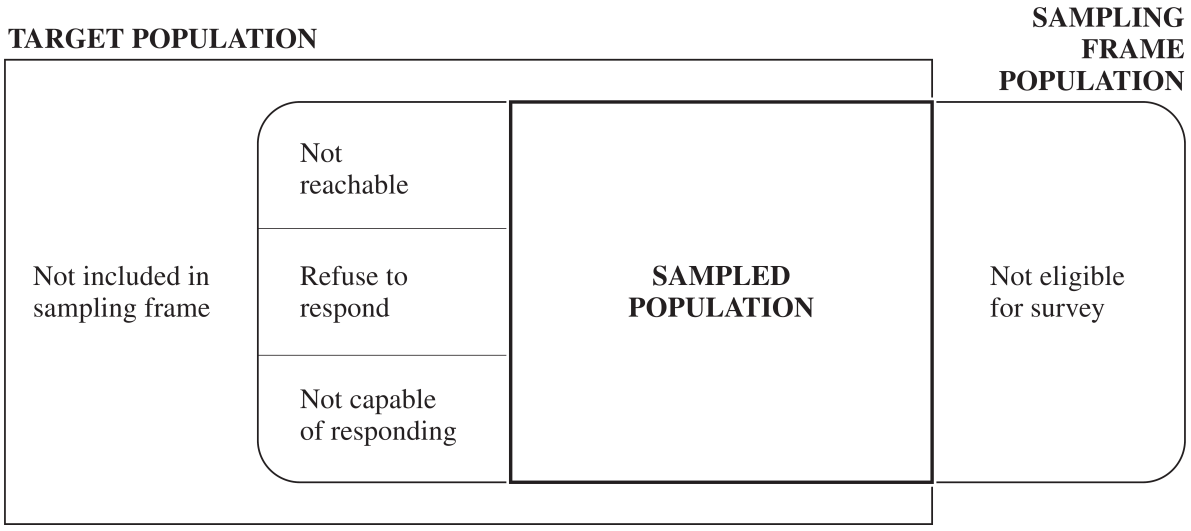

Figure 1.1: Target population, sampling frame population and sampled population. The sampled population is only the part of the target population that is in the frame and can and will respond.

In the Hite study the element was an individual woman and the target population was all adult U.S. women. The sampled population, however, was women who belonged to the organizations that received questionnaires *and* who would complete and return them.

1.3 Selection Bias

Selection bias occurs when part of the target population is not in the sampled population or, more generally, when some population units are sampled at a different rate than intended by the investigator. A **sample of convenience** is often biased, because the units that are easiest to select, or most likely to respond, are usually not representative of the harder-to-select or nonresponding units. For example:

- A convenience sample of adolescents used to study communication about AIDS probably overestimated how much parents and teenagers talk about it. Adolescents willing to talk to investigators are also more likely to talk to other authority figures, such as their parents.
- **Misspecifying the target population.** Polls before the 1994 Arizona gubernatorial primary predicted that Eddie Basha would trail by 9 or more points; he won with 37% of the vote. The polls targeted people who had voted in past primaries and missed the new rural voters who supported Basha.

Selection bias has several common sources.

Undercoverage is the failure to include all of the target population in the sampling frame. The U.S. Behavioral Risk Factor Surveillance System, a telephone survey, misses households without telephones and for many years missed households with only cell phones. Coverage varies by region and by income, so undercoverage can bias the estimates.

Overcoverage is the inclusion in the sampling frame of units that are not in the target population, for example when interviewers include respondents under 18 in a radio survey meant for adults only.

Nonresponse is the failure to obtain responses from all of the units chosen for the sample. It distorts the results even when the other sources of selection bias are minimized. Nonrespondents often differ in critical ways from respondents, but the extent of the difference is usually unknown. Some published surveys have response rates as low as 10%. It is difficult to generalize from such a survey when 90% of the sample could not be reached or refused to take part.

1.4 Measurement Error

When a response differs from the true value, **measurement error** has occurred. **Measurement bias** occurs when the responses tend to differ from the truth in one direction. Like selection bias, measurement bias must be considered and minimized at the design stage; no amount of data analysis can remove it afterwards. Some sources of measurement error:

- **People do not always tell the truth.** Farmers receiving food aid may underreport their crop yields, hoping to receive more aid.
- **People do not always understand the questions.** A 1993 Roper poll found that 25% of Americans did not believe the Holocaust had happened. The question used a confusing double negative. When it was reworded, only 1% thought it “possible” that the Holocaust never happened.

- **People forget.** In the U.S. National Crime Victimization Survey (NCVS), respondents asked about crimes in the last six months sometimes include events from further back. This is called **telescoping**.

Careful questionnaire design reduces measurement error:

- **Always test the questions** on a small group before the survey is carried out.
- **Keep the questions simple and clear.** Belson (1981) tested the question "What *proportion* of your evening viewing time do you spend watching news programmes?" on 53 people. Only 14 interpreted "proportion" correctly, as a percentage, part or fraction.
- **Use specific questions instead of general ones.** Instead of asking "Did anyone attack you in the last six months?", the NCVS asks a detailed series of questions: "With any weapon, for instance a gun or knife", "With anything like a baseball bat, frying pan, scissors, or stick", and so on.

1.5 Sampling and Nonsampling Errors

Sampling error results from taking one sample instead of examining the whole population. A different sample would likely give a different result. Sampling error is what a poll's "margin of error" describes, and because it comes from the random selection of the sample, it can be quantified in probabilistic terms. Most of this book is about measuring and reducing it.

Nonsampling errors, such as selection bias and measurement error, cannot be attributed to sample-to-sample variability. They are often far larger than the sampling error. A survey may proudly report a margin of error of 3% based on a 30% response rate, while ignoring the large selection bias that such a response rate may cause.

1.5.1 Why sample at all?

If samples have errors, why not always take a census of the whole population?

- **Cost.** A sample provides reliable information at far less cost than a census, and with a probability sample the sampling error can be quantified. Some measurements are destructive (for example, pulverizing a cookie to measure its fat content), so a census is not even possible.
- **Speed.** Data from a sample can be collected and processed more quickly, so the estimates are timely. An unemployment rate for 2005 published in 2015 would be useless.
- **Accuracy.** Estimates from a sample can be *more* accurate than those from a census. A census requires a large administrative effort with many data collectors, while a smaller operation allows the investigators to be more careful and to reduce nonsampling errors.

1.6 Overview of Probability Sampling and Estimation Techniques

In a **probability sample**, every unit in the population has a known, positive chance of being selected, and the sample is chosen by a random mechanism. This removes selection bias due to the investigator's choices, and it makes it possible to quantify the sampling error. This section previews the techniques studied in the rest of the book: three basic designs for drawing the sample (simple random, stratified and cluster sampling), estimators that use auxiliary

information (ratio and regression estimation), and designs with unequal selection probabilities. Each has its own chapter later in the book.

To compare the designs we use synthetic populations of $N = 1000$ units, with the y-values drawn from exponential distributions. The population for SRS and stratified sampling (Figure 1.2 and Figure 1.3) consists of four strata of sizes 400, 300, 200 and 100 whose means differ a lot: each is an exponential distribution with mean 1.5, shifted by 0, 4, 8 and 12 respectively. The population for cluster sampling (Figure 1.4) consists of ten clusters whose sizes range from 40 to 160 units, exponential with means from 2 to 8. Each design selects $n = 60$ units, shown in red; the unselected units are grey.

Simple random sampling. A **simple random sample (SRS)** is the simplest probability sample. An SRS of size n is taken when every possible subset of n units in the population has the same chance of being the sample. The design uses no information about the groups: in Figure 1.2 the 1000 units are listed in an arbitrary order, and 60 of them are chosen at random. By chance, a group may be over- or under-represented, and the sample values range over the whole spread of the population.

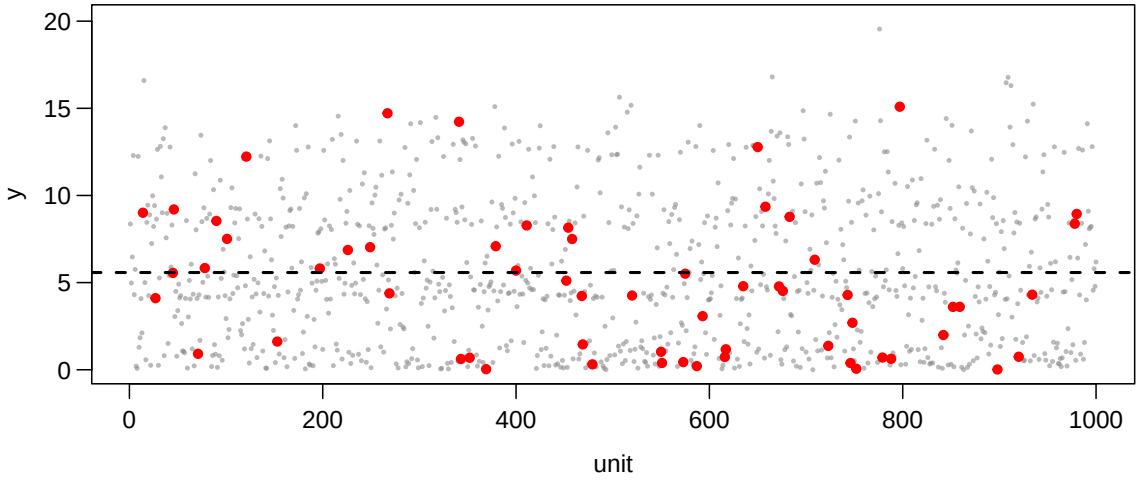

Figure 1.2: A simple random sample of $n = 60$ (red) from a population of $N = 1000$ units, listed in an arbitrary order. The dashed line is the population mean.

Stratified random sampling. In a **stratified random sample**, the population is divided into subgroups called **strata**, and an SRS is selected independently from each stratum. Strata are often subgroups of interest, such as provinces, terrain types or firm sizes. In Figure 1.3 the same 1000 units are arranged by stratum, with the strata sorted by their means (dashed lines). The sample is allocated in proportion to the stratum sizes, $60 \times N_h/1000$, so 24, 18, 12 and 6 units are drawn from the four strata. Every stratum is represented in the sample in a fixed proportion. Stratification often increases precision, because units within a stratum tend to be more similar than units selected at random from the whole population. Here the stratum means are far apart while the spread within each stratum is small, so the differences between the strata, which make up most of the variation in Figure 1.2, no longer contribute to the sampling error.

Cluster sampling. In a **cluster sample**, the observation units are grouped into larger sampling units called **clusters**. For example, to survey Lutheran church members in Minneapolis without a list of all members, we could take an SRS of churches (the clusters) and then subsample members within each selected church (the observation units). In Figure 1.4 the population consists of ten clusters of different sizes, from 40 to 160 units, sorted by their means. Three clusters are selected at random (shaded), and 20 units are subsampled in each. This is convenient, since only

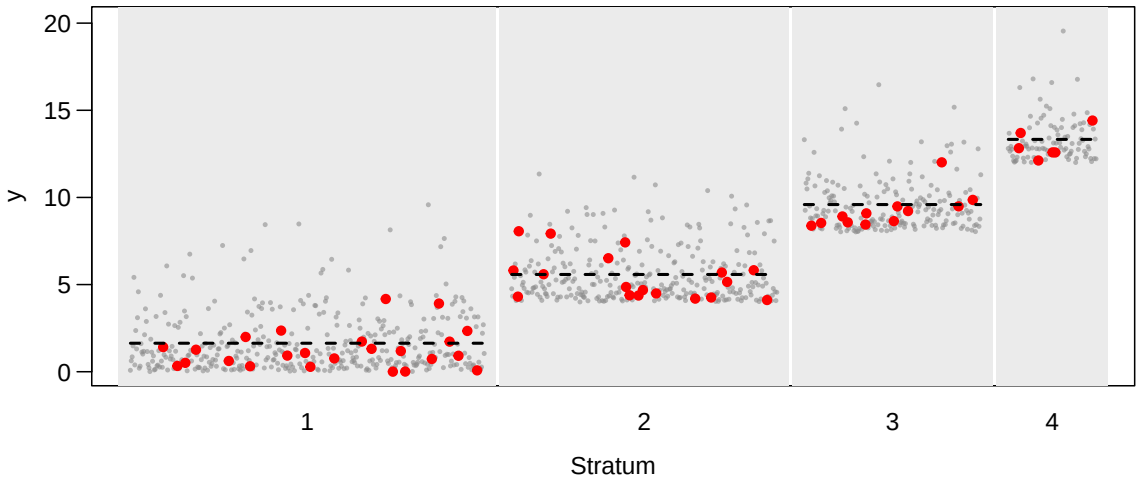

Figure 1.3: A stratified random sample from the same population: the four strata are sorted by their means (dashed lines), and SRSs of 24, 18, 12 and 6 units (red) are drawn from the strata of sizes 400, 300, 200 and 100.

three clusters need to be visited, but the seven clusters that are not selected contribute nothing to the sample. When the cluster means differ, the estimate depends heavily on *which* clusters happen to be chosen. Members of the same church may likewise be more similar to each other than a random sample of Lutherans would be. A cluster sample therefore often provides less information than an SRS of the same size, and the gain in convenience must be weighed against the loss in precision.

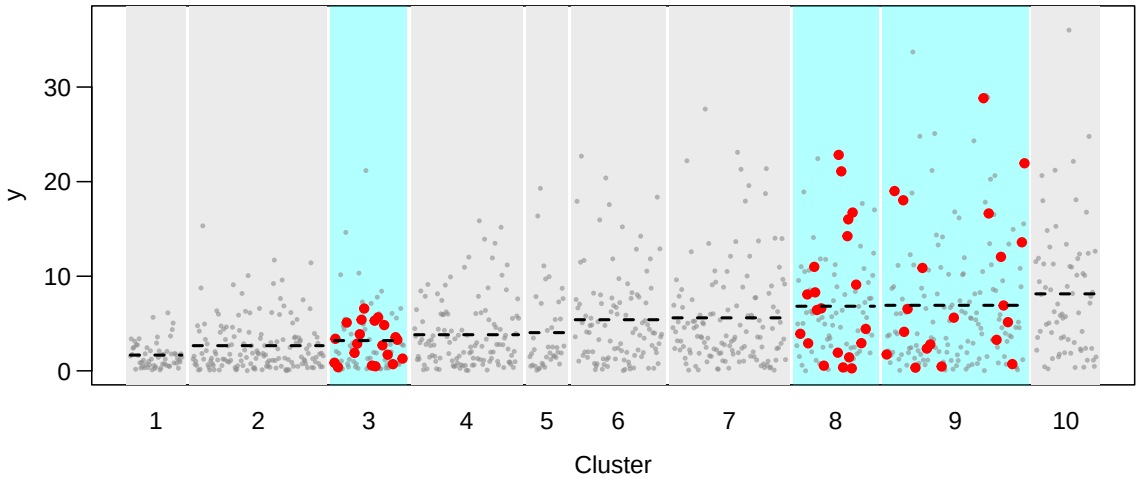

Figure 1.4: A cluster sample from a population of ten clusters of 40 to 160 units, sorted by their means (dashed lines). Three clusters are selected at random (shaded), and 20 units (red) are subsampled in each; the other clusters are not observed at all.

Using auxiliary information: ratio and regression estimation. Often we know, for every unit in the population, an **auxiliary variable** x_i that is related to y_i: last year's value of y, the size of a farm, the floor area of a store. Ratio and regression estimation take an ordinary sample but use the known x_i at the *estimation* stage. In Figure 1.5, y is roughly proportional to x. From an SRS (green circles) we estimate the slope $\hat{B} = \bar{y}/\bar{x}$ of the line $y = Bx$ (red), and use it to predict y for every unit of the population from its known x_i (red points). The **ratio estimate** $\hat{\bar{y}}_r = \hat{B}\,\bar{x}_U$

is the average of these predictions. In this sample the green units happen to have large x, so the sample mean $\bar{y}$ is too high. The ratio estimate corrects for this through the known $\bar{x}_U$ and lands close to $\bar{y}_U$. **Regression estimation** does the same with a fitted line $y = B_0 + B_1 x$ that need not pass through the origin. Both are studied in Section 5.1 and the sections that follow.

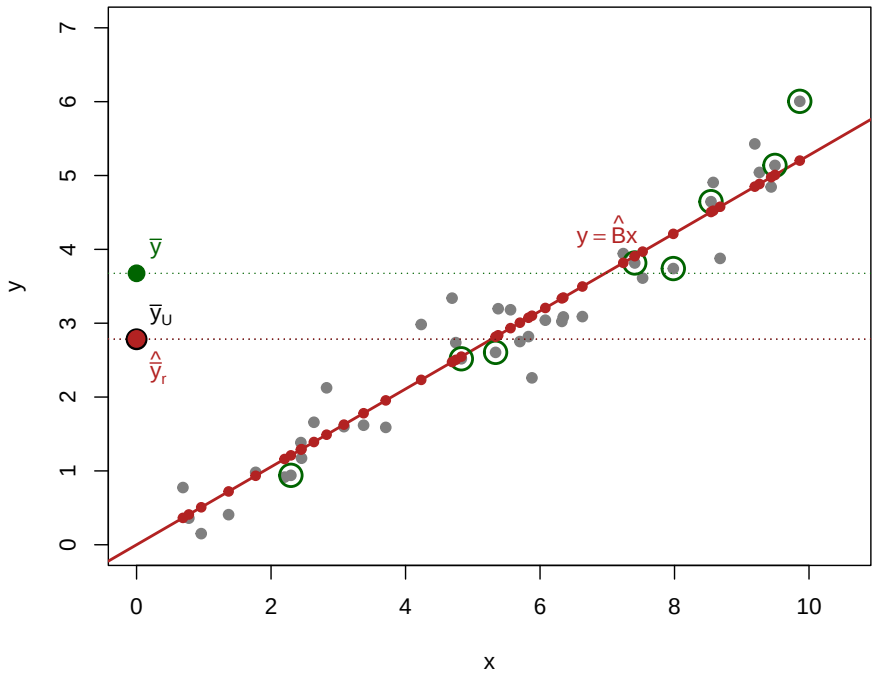

Figure 1.5: Ratio estimation. Grey: the population, whose x_i are all known. Green circles: an SRS. Red: the line $y = \hat{B}x$ fitted from the sample and the predicted value $\hat{B}x_i$ of every unit. Left margin: the population mean $\bar{y}_U$ (black), the sample mean $\bar{y}$ (green) and the ratio estimate $\hat{\bar{y}}_r$, the mean of the red points.

Unequal probability sampling. Auxiliary information can also be used at the *design* stage. In **unequal probability sampling (UPS)**, units are selected with different probabilities, usually proportional to a measure of size x_i that is expected to be roughly proportional to y_i. Large units, which contribute most to the population total, are then more likely to be in the sample. In Figure 1.6 the units of Figure 1.2 are drawn again, each with a size measure x_i equal to y_i multiplied by a random error, and the point sizes show the inclusion probabilities $\pi_i \propto x_i$. A sample of 60 units (red) is drawn with these probabilities. Most of the sampled units have large y, so the plain sample mean would be far too high. Each unit must instead be weighted by its sampling weight $w_i = 1/\pi_i$ (Section 1.7), which gives the many small units their due. When x_i is nearly proportional to y_i, the weighted estimator $\sum_{i \in \mathcal{S}} y_i/\pi_i$ of the total is very precise, because y_i/π_i is nearly the same for every unit. UPS is studied in Section 7.2 and the sections that follow.

1.7 A Framework for Probability Sampling

We now make the idea of a probability sample precise. The key concepts are the population, the sample, the probabilities of the samples, the inclusion probabilities, and the bias, variance and mean squared error of an estimator.

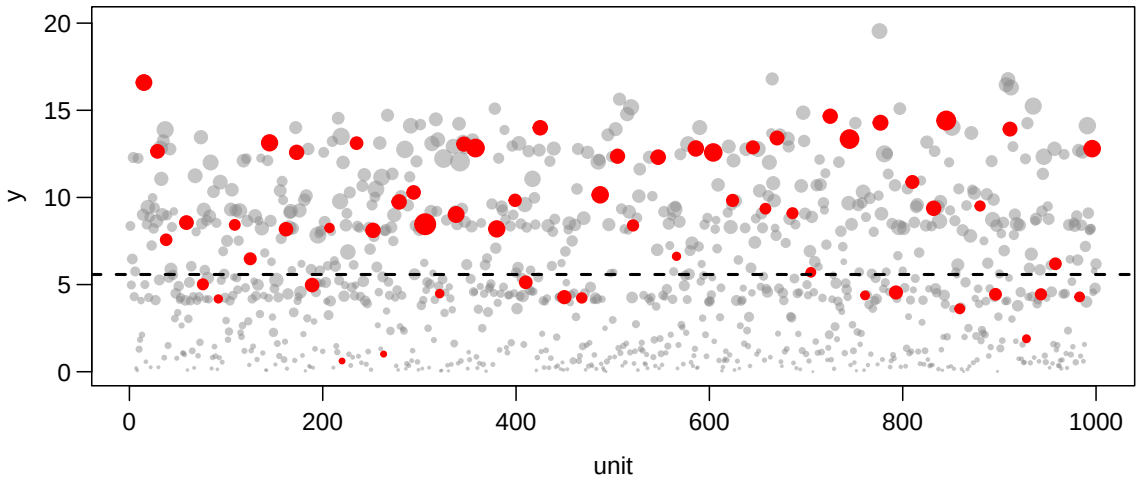

Figure 1.6: Unequal probability sampling from the population of Figure 1.2. Each unit's inclusion probability is proportional to a size measure x_i equal to y_i times a random error; point sizes show the probabilities. The $n = 60$ sampled units (red) are mostly large ones. The dashed line is the population mean.

1.7.1 The population and its samples

The finite **population**, or **universe**, of N units is denoted by the index set

$$\mathcal{U} = \{1, 2, \ldots, N\}. \tag{1.1}$$

Out of this population we can choose various **samples**, which are subsets of $\mathcal{U}$. The sample actually chosen is denoted $\mathcal{S}$; it consists of n of the units in $\mathcal{U}$.

For example, suppose $\mathcal{U} = \{1, 2, 3, 4\}$, so $N = 4$. There are $\binom{4}{2} = 6$ different samples of size $n = 2$:

$$\mathcal{S}_1 = \{1, 2\}, \quad \mathcal{S}_2 = \{1, 3\}, \quad \mathcal{S}_3 = \{1, 4\}, \quad \mathcal{S}_4 = \{2, 3\}, \quad \mathcal{S}_5 = \{2, 4\}, \quad \mathcal{S}_6 = \{3, 4\}. \tag{1.2}$$

1.7.2 Probability sampling

In **probability sampling**, each possible sample $\mathcal{S}$ has a known probability $P(\mathcal{S})$ of being chosen, and the probabilities of all possible samples sum to 1. The collection of these probabilities is the **sampling design**. One possible design for a sample of size 2 from $\mathcal{U} = \{1, 2, 3, 4\}$ is

$$P(\mathcal{S}_1) = \tfrac{1}{3}, \quad P(\mathcal{S}_4) = \tfrac{1}{6}, \quad P(\mathcal{S}_6) = \tfrac{1}{2}, \quad P(\mathcal{S}_2) = P(\mathcal{S}_3) = P(\mathcal{S}_5) = 0. \tag{1.3}$$

The probabilities are known **before** the sample is drawn. One way to carry out this design is to put six balls in a box, two labeled 1, one labeled 4 and three labeled 6, draw one ball at random, and take the sample $\mathcal{S}_i$ whose label is drawn. An SRS is the special design in which all six samples have probability $1/6$.

1.7.3 The values y_i

Let y_i be a characteristic of the ith unit in the population, such as the corn yield of farm i:

i	1	2	3	$\cdots$	N
y_i	y_1	y_2	y_3	$\cdots$	y_N

We treat each y_i as a **fixed** quantity. If farm 723 is in the sample, its corn yield y_{723} is known exactly; if it is not, y_{723} is unknown but still fixed. The only randomness is in *which* units are selected. This is the **design-based** (or randomization) approach, followed by Lohr and by this book. (Scheaffer et al. use Y_i for a random variable and y_i for its value.)

1.7.4 Inclusion probabilities and sampling weights

Once a sampling design is chosen, each unit in the population has a known probability of appearing in the sample, its **inclusion probability**

$$\pi_i = P(\text{unit } i \text{ is in the sample}) = \sum_{\mathcal{S} \ni i} P(\mathcal{S}), \tag{1.4}$$

the sum of the probabilities of all possible samples that contain unit i. In probability sampling all the π_i are known before the survey starts, and we assume $\pi_i > 0$ for every unit, so that every unit has a chance to be observed.

For the design above, unit 1 belongs only to $\mathcal{S}_1 = \{1, 2\}$ among the samples with positive probability, so

$$\pi_1 = P(\{1,2\}) = \tfrac{1}{3}, \qquad \pi_2 = P(\{1,2\}) + P(\{2,3\}) = \tfrac{1}{3} + \tfrac{1}{6} = \tfrac{1}{2}, \tag{1.5}$$

$$\pi_3 = P(\{2,3\}) + P(\{3,4\}) = \tfrac{1}{6} + \tfrac{1}{2} = \tfrac{2}{3}, \qquad \pi_4 = P(\{3,4\}) = \tfrac{1}{2}. \tag{1.6}$$

The inclusion probabilities do not sum to 1. They sum to 2, because every sample contains $n = 2$ units; in general $\sum_{i=1}^{N} \pi_i = n$ for a design with fixed sample size n.

The **sampling weight** of unit i is $w_i = 1/\pi_i$. It can be thought of as the number of population units that a sampled unit represents. For many sampling designs, the estimator of the population total can be written as

$$\hat{t} = \sum_{i \in \mathcal{S}} w_i \, y_i = \sum_{i \in \mathcal{S}} \frac{y_i}{\pi_i}. \tag{1.7}$$

For an SRS, $\pi_i = n/N$ and $w_i = N/n$ for every unit; for the other designs in this book, the weights differ between units.

1.7.5 Population quantities

The quantities we want to estimate are characteristics of the fixed values $y_1, \ldots, y_N$:

$$\textbf{total: } t = \sum_{i=1}^{N} y_i, \qquad \textbf{mean: } \bar{y}_U = \frac{t}{N}, \tag{1.8}$$

$$\textbf{variance: } S^2 = \frac{1}{N-1} \sum_{i=1}^{N} (y_i - \bar{y}_U)^2, \qquad \textbf{standard deviation: } S = \sqrt{S^2}. \tag{1.9}$$

Scheaffer et al. instead use $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}_U)^2$, dividing by N rather than $N-1$. The two differ only by the factor $(N-1)/N$.

Example: a population proportion. Let $y_i = 1$ if unit i has some attribute and $y_i = 0$ otherwise:

i	1	2	3	4	5
y_i	1	0	0	0	1

Then the total is the number of units with the attribute and the mean is the population proportion:

$$t = 1 + 0 + 0 + 0 + 1 = 2, \qquad \bar{y}_U = \frac{2}{5} = 0.4, \tag{1.10}$$

$$S^2 = \frac{1}{5-1}\Big((1-0.4)^2 \times 2 + (0-0.4)^2 \times 3\Big) = 0.3. \tag{1.11}$$

For a 0/1 variable, $S^2 = \frac{N}{N-1}\,p(1-p)$ with $p = \bar{y}_U$; here $\frac{5}{4}(0.4)(0.6) = 0.3$. As N grows, $S^2 \to p(1-p)$.

Example: number of devices per household. A population of $N = 4$ households has the following numbers of devices:

i	1	2	3	4
y_i	2	4	4	6

$$t = 2 + 4 + 4 + 6 = 16, \qquad \bar{y}_U = \frac{16}{4} = 4, \tag{1.12}$$

$$S^2 = \frac{(2-4)^2 + (4-4)^2 + (4-4)^2 + (6-4)^2}{4-1} = \frac{8}{3}. \tag{1.13}$$

1.7.6 Statistics and sampling distributions

A **statistic** is a function of the observations in the sample, used to estimate a population quantity. Since the sample is random, so is the statistic. Its **sampling distribution** is its distribution under repeated sampling: the values it takes over all possible samples, with the probabilities $P(\mathcal{S})$ of those samples.

For example, to estimate the total t we may use the sample mean scaled up by the population size,

$$\bar{y} = \frac{1}{n}\sum_{i \in \mathcal{S}} y_i, \qquad \hat{t} = N\bar{y}. \tag{1.14}$$

Take the devices population ($\bar{y}_U = 4$, $t = 16$) with the sampling design of the previous sections. For each of the six possible samples of size 2 we can compute $\bar{y}$:

$\mathcal{S}$	$\{1,2\}$	$\{1,3\}$	$\{1,4\}$	$\{2,3\}$	$\{2,4\}$	$\{3,4\}$
values	2, 4	2, 4	2, 6	4, 4	4, 6	4, 6
$P(\mathcal{S})$	$1/3$	0	0	$1/6$	0	$1/2$
$\bar{y}$	3	3	4	4	5	5

Collecting the samples that give the same value of $\bar{y}$, and adding their probabilities, gives the **sampling distribution of** $\bar{y}$ (Figure 1.7):

k	3	4	5
$P(\bar{y}=k)$	$1/3$	$1/6$	$1/2$

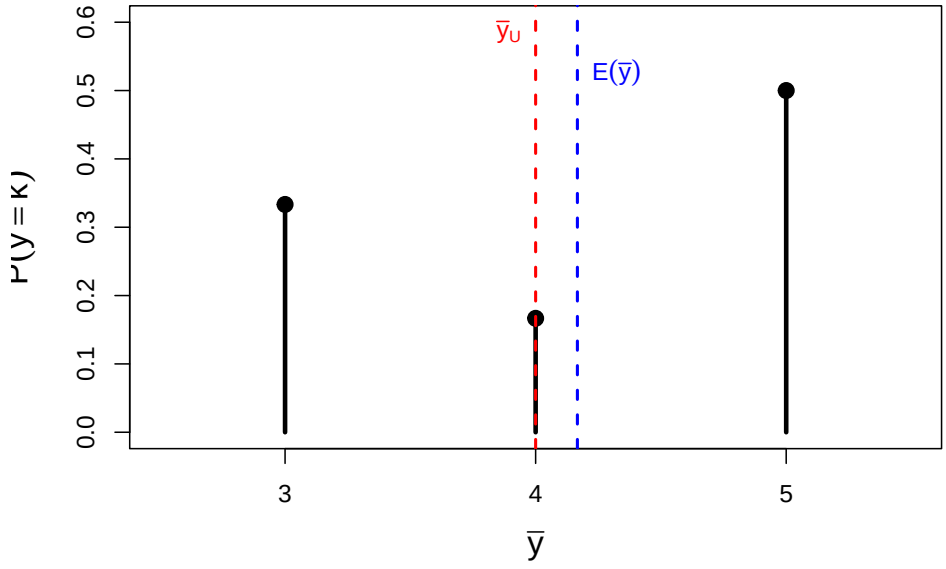

Figure 1.7: Sampling distribution of $\bar{y}$ for the devices population under the design $P(\{1,2\}) = 1/3$, $P(\{2,3\}) = 1/6$, $P(\{3,4\}) = 1/2$. The mean of the distribution, $E(\bar{y}) = 25/6$, is not the population mean $\bar{y}_U = 4$.

1.7.7 Bias, variance and mean squared error

The sampling distribution tells us how good an estimator $\hat{\theta}$ of a population quantity θ is. Three summaries are used:

$$\text{Bias}(\hat{\theta}) = E(\hat{\theta}) - \theta, \qquad V(\hat{\theta}) = E[(\hat{\theta} - E(\hat{\theta}))^2], \qquad \text{MSE}(\hat{\theta}) = E[(\hat{\theta} - \theta)^2]. \tag{1.15}$$

The variance measures the spread of the estimates around their own mean; the MSE measures their spread around the true value.

For $\bar{y}$ in the example,

$$E(\bar{y}) = 3 \times \tfrac{1}{3} + 4 \times \tfrac{1}{6} + 5 \times \tfrac{1}{2} = \frac{25}{6}, \qquad \text{Bias}(\bar{y}) = \frac{25}{6} - 4 = \frac{1}{6}, \tag{1.16}$$

$$V(\bar{y}) = \left(3 - \tfrac{25}{6}\right)^2 \tfrac{1}{3} + \left(4 - \tfrac{25}{6}\right)^2 \tfrac{1}{6} + \left(5 - \tfrac{25}{6}\right)^2 \tfrac{1}{2} = \frac{29}{36} \approx 0.8056, \tag{1.17}$$

$$\mathrm{MSE}(\bar{y}) = (3-4)^2 \cdot \tfrac{1}{3} + (4-4)^2 \cdot \tfrac{1}{6} + (5-4)^2 \cdot \tfrac{1}{2} = \frac{5}{6} \approx 0.8333. \tag{1.18}$$

The estimator is biased under this design. The design favours the sample $\{3, 4\}$, which contains the household with 6 devices, so $\bar{y}$ tends to be too large. Correspondingly $\hat{t} = 4\bar{y}$ has $E(\hat{t}) = 50/3 \approx 16.67$, a bias of $2/3$ for $t = 16$.

MSE = Bias² + Variance. Adding and subtracting $E(\hat{\theta})$,

$$\mathrm{MSE}(\hat{\theta}) = E\Big[(\hat{\theta} - E(\hat{\theta})) + (E(\hat{\theta}) - \theta)\Big]^2 = V(\hat{\theta}) + \big(\mathrm{Bias}(\hat{\theta})\big)^2, \tag{1.19}$$

because the cross term $2\,\mathrm{Bias}(\hat{\theta})\,E[\hat{\theta} - E(\hat{\theta})]$ is zero. In the example, $(1/6)^2 + 29/36 = 1/36 + 29/36 = 30/36 = 5/6$, which agrees with the direct calculation. In particular, when $\hat{\theta}$ is unbiased, $\mathrm{MSE}(\hat{\theta}) = V(\hat{\theta})$.

The same population under SRS. If instead all six samples have probability $1/6$ (an SRS), the six values of $\bar{y}$ are $3, 3, 4, 4, 5, 5$, so $E(\bar{y}) = 4 = \bar{y}_U$: the estimator is now unbiased. Its variance is $\frac{1}{6}(1+1+0+0+1+1) = \frac{2}{3}$, which agrees with the SRS formula $(1 - \frac{n}{N})\frac{S^2}{n} = \frac{1}{2} \cdot \frac{8/3}{2} = \frac{2}{3}$ of Section 3.2. The short R enumeration below checks both designs.

```
y <- c(2, 4, 4, 6)
samples <- combn(4, 2)                          # the 6 possible samples, one per column
ybar <- apply(samples, 2, function(s) mean(y[s]))
designs <- rbind(unequal = c(1/3, 0, 0, 1/6, 0, 1/2),
                 SRS     = rep(1/6, 6))
round(t(apply(designs, 1, function(P) {
  E <- sum(P * ybar)
  c(E = E, Bias = E - mean(y), V = sum(P * (ybar - E)^2), MSE = sum(P * (ybar - mean(y))^2))
})), 4)
```

```
            E   Bias      V    MSE
unequal 4.1667 0.1667 0.8056 0.8333
SRS     4.0000 0.0000 0.6667 0.6667
```

1.7.8 Unbiased, precise and accurate

An estimator $\hat{t}$ of t is

- **unbiased** if $E(\hat{t}) = t$;
- **precise** if $V(\hat{t}) = E\big[(\hat{t} - E[\hat{t}])^2\big]$ is small;
- **accurate** if $\mathrm{MSE}(\hat{t}) = E\big[(\hat{t} - t)^2\big]$ is small.

Precision measures how close the estimates from different samples are to *each other*; accuracy measures how close they are to the *truth*. A badly biased estimator may be very precise but will not be accurate. Figure 1.8 shows the three cases as archers aiming at the centre of a target: archer A is unbiased but imprecise, archer B is precise but biased, and archer C is both unbiased and precise, and therefore accurate. Selection bias and measurement bias make a survey like archer B: increasing the sample size makes the estimates more precise, but no closer to the truth.

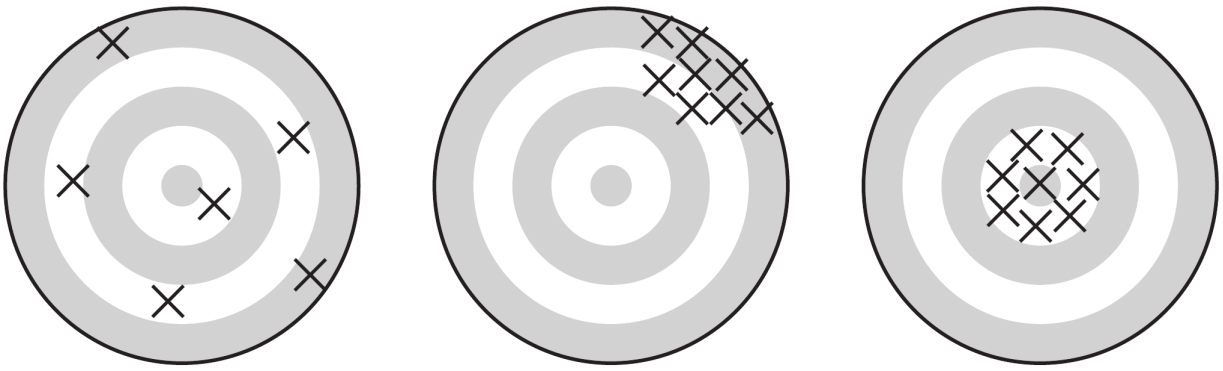

Figure 1.8: Unbiased but imprecise (A), precise but biased (B), and accurate (C).

1.7.9 Summary

The finite population $\mathcal{U}$ consists of the units $\{1, 2, \ldots, N\}$, with values $\{y_1, y_2, \ldots, y_N\}$. We select a sample $\mathcal{S}$ of n units from $\mathcal{U}$ using the selection probabilities that define the sampling design. The y_i are fixed but unknown quantities, unknown unless unit i happens to appear in the sample. All statements about the bias, variance and MSE of an estimator refer to its distribution over the possible samples. The following chapters work out these properties for each design, starting with simple random sampling in Section 3.1.

2 Data Analysis with R

Modern workflows with the native pipe, dplyr, gt, and ggplot2

```
library(dplyr)
library(gt)
library(gtsummary)
library(ggplot2)
```

All packages are loaded once in a hidden `setup` chunk, and `message`/`warning` suppression is set globally in the YAML header rather than repeated on every chunk.

2.1 Foundations

2.1.1 Basic R objects

```
# vectors
x <- seq(30, 3, by = -2)
a <- c(66.32, 69.87, 70.12, 90.37, 50.08, 61.20, 65.00, 57.65)

a[1]            # extract an element
```

```
[1] 66.32
```

```
a[1] <- 85.34   # replace an element
ma <- mean(a)
ma
```

```
[1] 68.70375
```

```
# read a vector of numbers from text without saving a file first
y <- scan(text = "7 8 9 10 11 12 13 13 14 17 17 45")
y
```

```
 [1]  7  8  9 10 11 12 13 13 14 17 17 45
```

```r
# matrices: matrix() fills by column unless byrow = TRUE
A <- matrix(1:8, nrow = 4, ncol = 2)
A
```

```
     [,1] [,2]
[1,]    1    5
[2,]    2    6
[3,]    3    7
[4,]    4    8
```

```r
D <- matrix(a, nrow = 4, ncol = 2, byrow = TRUE)
D
```

```
      [,1]  [,2]
[1,] 85.34 69.87
[2,] 70.12 90.37
[3,] 50.08 61.20
[4,] 65.00 57.65
```

```r
# a larger matrix, to illustrate element and row assignment
B <- matrix(1:5000, nrow = 100, ncol = 50)
B[2, 4] <- 45
B[1, ] <- 1:50
head(B[1, ])   # part of the first row
```

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

```r
head(B[, 1])   # part of the first column
```

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

```r
# lists
E <- list(newa = a, newA = A)
names(E)
```

```
[1] "newa" "newA"
```

```r
E$newA
```

```
     [,1] [,2]
[1,]    1    5
[2,]    2    6
[3,]    3    7
[4,]    4    8
```

```r
E$newa <- 10:17

# data frames
stat245_scores <- data.frame(
  student = c("Peter", "John", "Alice"),
  score   = c(30, 45, 50)
)
stat245_scores
```

```
  student score
1   Peter    30
2    John    45
3   Alice    50
```

The column of names is called `student` rather than `names`, so that the base function `names()` is not masked.

2.2 Importing a dataset

```
data_url <- "https://raw.githubusercontent.com/longhaiSK/sampling/main/sampling_book/data/"
agpop_file <- paste0(data_url, "agpop.csv")

# -99 is the missing-value code in this file, so convert it to NA on import
agpop <- read.csv(agpop_file, na.strings = "-99")

colnames(agpop)
```

```
 [1] "county"   "state"    "acres92"  "acres87"  "acres82"  "farms92"  "farms87"  "farms82"  "largef92"
[10] "largef87" "largef82" "smallf92" "smallf87" "smallf82" "region"
```

```
c(rows = nrow(agpop), cols = ncol(agpop))
```

```
rows cols
3078   15
```

Because `na.strings = "-99"` is applied at import, every `-99` is already `NA` in `agpop`. All later filtering therefore uses `is.na()`; comparisons such as `acres92 != -99` no longer identify the missing values.

2.2.1 Displaying the full dataset

Printing several thousand rows into an HTML document consumes vertical space and slows the browser. With `df-print: paged` (set under `format: html` in the YAML header), a printed data frame appears as a paged table: all rows are embedded as one lightweight data blob, and only the current page is drawn. Static targets such as Typst or PDF have no pager, so only the first rows are shown there.

```
head(agpop)
```

```
                county state acres92 acres87 acres82 farms92 farms87 farms82 largef92 largef87 largef82
1 ALEUTIAN ISLANDS AREA    AK  683533  726596  764514      26      27      28       14       16       20
2        ANCHORAGE AREA    AK   47146   59297  256709     217     245     223        9       10       11
3        FAIRBANKS AREA    AK  141338  154913  204568     168     175     170       25       28       21
4           JUNEAU AREA    AK     210     214     127       8       8      12        0        0        0
5  KENAI PENINSULA AREA    AK   50810   85712   98035      93     119     137        9       18       17
6        AUTAUGA COUNTY    AL  107259  116050  145044     322     388     453       25       32       32
  smallf92 smallf87 smallf82 region
1        6        4        1      W
2       41       52       38      W
3       12       18       25      W
4        5        4        8      W
5       12       18       19      W
6        8       19       17      S
```

2.2.2 Simple summaries

```
mean(agpop$acres92, na.rm = TRUE)
```

```
[1] 308582.4
```

```
sd(agpop$acres92, na.rm = TRUE)
```

```
[1] 425312.8
```

2.3 The native pipe

The native pipe `|>` takes the value on its left and passes it as the **first argument** to the function on its right, so `x |> f()` is another way of writing `f(x)`. Pipes chain, so the output of one step feeds the next.

```
x <- c(1, 2, 3, 4, 5, NA)

# without the pipe
sum(x, na.rm = TRUE)
```

```
[1] 15
```

```
log(sum(x, na.rm = TRUE), base = 10)
```

```
[1] 1.176091
```

```
round(exp(log(sum(x, na.rm = TRUE), base = 10)), digits = 2)
```

```
[1] 3.24
```

```
# the same computations, read left to right
x |> sum(na.rm = TRUE)
```

```
[1] 15
```

```
x |> sum(na.rm = TRUE) |> log(base = 10)
```

```
[1] 1.176091
```

```
x |> sum(na.rm = TRUE) |> log(base = 10) |> exp() |> round(digits = 2)
```

```
[1] 3.24
```

Extra arguments are supplied by name at each step; only the first argument comes from the pipe.

```
c(1, 4, 9, 16, NA) |>
  sum(na.rm = TRUE) |>
  sqrt() |>
  round(digits = 1)
```

```
[1] 5.5
```

```
c(5, 1, NA, 3, 9) |>
  sort(decreasing = TRUE, na.last = NA) |>
  mean(trim = 0.1)
```

```
[1] 4.5
```

When the piped value belongs in an argument other than the first, mark its position with the placeholder _. The argument holding _ must be named. This requires R 4.2.0 or later.

```
# mtcars is the `data` argument of lm(), not its first argument
lm(mpg ~ disp, data = mtcars)
```

```

Call:
lm(formula = mpg ~ disp, data = mtcars)

Coefficients:
(Intercept)         disp
   29.59985     -0.04122
```

```
mtcars |> lm(mpg ~ disp, data = _)
```

```

Call:
lm(formula = mpg ~ disp, data = mtcars)

Coefficients:
(Intercept)         disp
   29.59985     -0.04122
```

2.4 Data manipulation with dplyr

`dplyr` verbs take a data frame as their first argument, which makes them natural pipe targets.

```
# mutate() creates several columns at once, replacing repeated $ assignment
stat245_scores <- stat245_scores |>
  mutate(
    perc = score / 50 * 100,
    adj  = perc + 10
  )
stat245_scores
```

```
  student score perc adj
1   Peter    30   60  70
2    John    45   90 100
3   Alice    50  100 110
```

```
# filter() subsets rows
agpop_AK <- agpop |> filter(state == "AK")
agpop_W  <- agpop |> filter(region == "W")
agpop_lg <- agpop |> filter(largef92 > 10)

c(AK = nrow(agpop_AK), W = nrow(agpop_W), largefarm = nrow(agpop_lg))
```

```
      AK         W largefarm
       5       422      2184
```

```
# slice() subsets rows by position and select() subsets columns;
# together they replace agpop[1:20, c("acres92", "largef92")]
agpop_subset <- agpop |>
  slice(1:20) |>
  select(acres92, largef92)

head(agpop_subset)
```

```
  acres92 largef92
1  683533       14
2   47146        9
3  141338       25
4     210        0
5   50810        9
6  107259       25
```

```
vars_of_interest <- c("acres92", "acres87", "farms92", "largef92", "smallf92")

# group_by() + summarize() + across(): mean and variance of each variable by
# region, over counties that are complete on all five variables
ag_summary <- agpop |>
  filter(if_all(all_of(vars_of_interest), ~ !is.na(.x))) |>
  group_by(region) |>
  summarize(
    across(
      all_of(vars_of_interest),
      list(mean = ~ mean(.x), var = ~ var(.x)),
      .names = "{.col}_{.fn}"
    ),
    .groups = "drop"
  )
ag_summary
```

```
# A tibble: 4 x 11
  region acres92_mean acres92_var acres87_mean acres87_var farms92_mean farms92_var largef92_mean largef92_var
  <chr>         <dbl>       <dbl>        <dbl>       <dbl>        <dbl>       <dbl>         <dbl>        <dbl>
1 NC          327483.     7.35e10      334064.     7.33e10         740.     150770.          75.3         6490.
2 NE           94462.     6.21e 9      104387.     7.36e 9         529.     191801.           8.67         126.
3 S           200846.     5.97e10      203877.     5.89e10         563.     189097.          37.1         2241.
```

```
4 W          736031.     7.03e11      749812.     7.19e11         636.     608924.         99.3         9455.
# i 2 more variables: smallf92_mean <dbl>, smallf92_var <dbl>
```

Restricting to complete cases with `if_all(..., ~ !is.na(.x))` makes the means and variances comparable across variables, since each is computed on the same set of counties.

2.5 Tables with gt

`gt()` writes every row as styled HTML, even inside a scrolling container, so a full-size dataset enlarges the file and slows rendering. Reduce the data first (e.g. with `slice_head()` or a `summarize()` step); for browsing raw data, the paged display in the import section is the better tool.

2.5.1 Spanning headers grouped by statistic

`tab_spanner()` groups columns under a shared header. Here the means are grouped together and the variances are grouped together.

```
ag_summary |>
  gt() |>
  tab_header(
    title = "Regional Summary: Mean and Variance of Five Variables",
    subtitle = "acres92, acres87, farms92, largef92, smallf92"
  ) |>
  tab_spanner(label = md("**Mean**"),     columns = ends_with("_mean")) |>
  tab_spanner(label = md("**Variance**"), columns = ends_with("_var")) |>
  cols_label(
    region        = md("**Region**"),
    acres92_mean  = md("$\\bar{y}_{acres92}$"),
    acres87_mean  = md("$\\bar{y}_{acres87}$"),
    farms92_mean  = md("$\\bar{y}_{farms92}$"),
    largef92_mean = md("$\\bar{y}_{largef92}$"),
    smallf92_mean = md("$\\bar{y}_{smallf92}$"),
    acres92_var   = md("$s^2_{acres92}$"),
    acres87_var   = md("$s^2_{acres87}$"),
    farms92_var   = md("$s^2_{farms92}$"),
    largef92_var  = md("$s^2_{largef92}$"),
    smallf92_var  = md("$s^2_{smallf92}$")
  ) |>
  fmt_number(columns = ends_with("_mean"), decimals = 1) |>
  fmt_number(columns = ends_with("_var"), decimals = 0, sep_mark = ",") |>
  tab_options(table.width = pct(100))
```

The `$...$` labels are typeset by the HTML math engine after the table is written into the page; they are not processed by `gt` itself, and will appear as literal dollar signs in the Typst output. For a target-independent version, replace them with plain text or Unicode labels.

Table 2.1: Regional summary of five variables, grouped by statistic

Regional Summary: Mean and Variance of Five Variables
acres92, acres87, farms92, largef92, smallf92

Region	Mean					Variance				
	$\bar{y}_{acres92}$	$\bar{y}_{acres87}$	$\bar{y}_{farms92}$	$\bar{y}_{largef92}$	$\bar{y}_{smallf92}$	$s^2_{acres92}$	$s^2_{acres87}$	$s^2_{farms92}$	$s^2_{largef92}$	$s^2_{smallf92}$
NC	327,482.9	334,064.2	740.2	75.3	42.8	73,461,356,635	73,319,885,590	150,770	6,490	1,147
NE	94,462.2	104,387.4	529.3	8.7	49.4	6,205,925,458	7,358,098,722	191,801	126	3,728
S	200,845.8	203,876.7	563.2	37.1	44.0	59,700,811,734	58,908,068,196	189,097	2,241	4,157
W	736,031.4	749,811.7	636.0	99.3	121.4	702,941,261,694	718,684,588,595	608,924	9,455	87,941

2.5.2 Spanning headers grouped by variable

Reordering the columns with `select()` puts each variable's mean and variance side by side under one spanner.

```
ag_summary |>
  select(
    region,
    acres92_mean,  acres92_var,
    acres87_mean,  acres87_var,
    farms92_mean,  farms92_var,
    largef92_mean, largef92_var,
    smallf92_mean, smallf92_var
  ) |>
  gt() |>
  tab_header(
    title = "Regional Summary: Mean and Variance of Five Variables",
    subtitle = "Grouped by variable"
  ) |>
  tab_spanner(label = md("**acres92**"),  columns = c(acres92_mean, acres92_var)) |>
  tab_spanner(label = md("**acres87**"),  columns = c(acres87_mean, acres87_var)) |>
  tab_spanner(label = md("**farms92**"),  columns = c(farms92_mean, farms92_var)) |>
  tab_spanner(label = md("**largef92**"), columns = c(largef92_mean, largef92_var)) |>
  tab_spanner(label = md("**smallf92**"), columns = c(smallf92_mean, smallf92_var)) |>
  # the spanner already names the variable, so the labels can stay generic
  cols_label(
    region        = md("**Region**"),
    acres92_mean  = md("$\\bar y$"), acres92_var  = md("$s^2$"),
    acres87_mean  = md("$\\bar y$"), acres87_var  = md("$s^2$"),
    farms92_mean  = md("$\\bar y$"), farms92_var  = md("$s^2$"),
    largef92_mean = md("$\\bar y$"), largef92_var = md("$s^2$"),
    smallf92_mean = md("$\\bar y$"), smallf92_var = md("$s^2$")
  ) |>
  fmt_number(columns = ends_with("_mean"), decimals = 1) |>
  fmt_number(columns = ends_with("_var"), decimals = 0, sep_mark = ",") |>
  tab_options(table.width = pct(100))
```

Both tables report identical numbers; only the grouping differs. Grouping by variable eases comparison of a variable's mean against its variance, whereas grouping by statistic eases comparison of means across variables.

Table 2.2: Regional summary of five variables, grouped by variable

Regional Summary: Mean and Variance of Five Variables
Grouped by variable

	acres92		acres87		farms92		largef92		smallf92	
Region	$\bar{y}$	s^2	$\bar{y}$	s^2	$\bar{y}$	s^2	$\bar{y}$	s^2	$\bar{y}$	s^2
NC	327,482.9	73,461,356,635	334,064.2	73,319,885,590	740.2	150,770	75.3	6,490	42.8	1,147
NE	94,462.2	6,205,925,458	104,387.4	7,358,098,722	529.3	191,801	8.7	126	49.4	3,728
S	200,845.8	59,700,811,734	203,876.7	58,908,068,196	563.2	189,097	37.1	2,241	44.0	4,157
W	736,031.4	702,941,261,694	749,811.7	718,684,588,595	636.0	608,924	99.3	9,455	121.4	87,941

Table 2.3: Automatic descriptive summary of agpop variables by region

Characteristic	**NC** N = 1,054[1]	**NE** N = 220[1]	**S** N = 1,382[1]	**W** N = 422[1]
acres92	326,571 (271,188)	93,600 (78,906)	200,009 (244,132)	730,267 (836,614)
Unknown	2	7	6	4
acres87	333,754 (270,834)	102,458 (86,127)	204,202 (243,247)	746,210 (847,298)
Unknown	4	5	8	6
largef92	75 (81)	8 (11)	37 (47)	98 (97)
farms92	737 (390)	509 (441)	559 (435)	625 (777)

[1] Mean (SD)

2.5.3 Automatic summary tables with gtsummary

`gtsummary::tbl_summary()` produces a descriptive table directly from the raw data, without a `summarize()` step.

```
agpop |>
  select(region, acres92, acres87, largef92, farms92) |>
  tbl_summary(
    by = region,
    statistic = list(all_continuous() ~ "{mean} ({sd})")
  )
```

2.6 Graphics

2.6.1 Base R graphics

Base R is convenient for quick diagnostic plots.

```r
hist(agpop$acres92, main = "Histogram of 1992 Acres", xlab = "Acres")
```

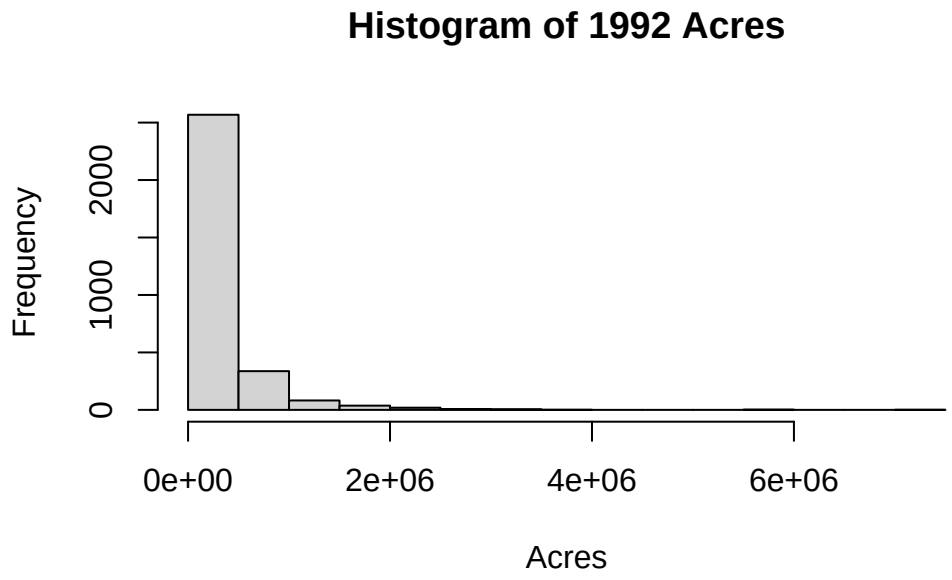

Figure 2.1: Histogram of 1992 farm acreage

```r
plot(agpop$acres87, agpop$acres92,
     main = "1992 vs 1987 Acreage",
     xlab = "Acres 1987", ylab = "Acres 1992")
abline(a = 0, b = 1, col = "red")
```

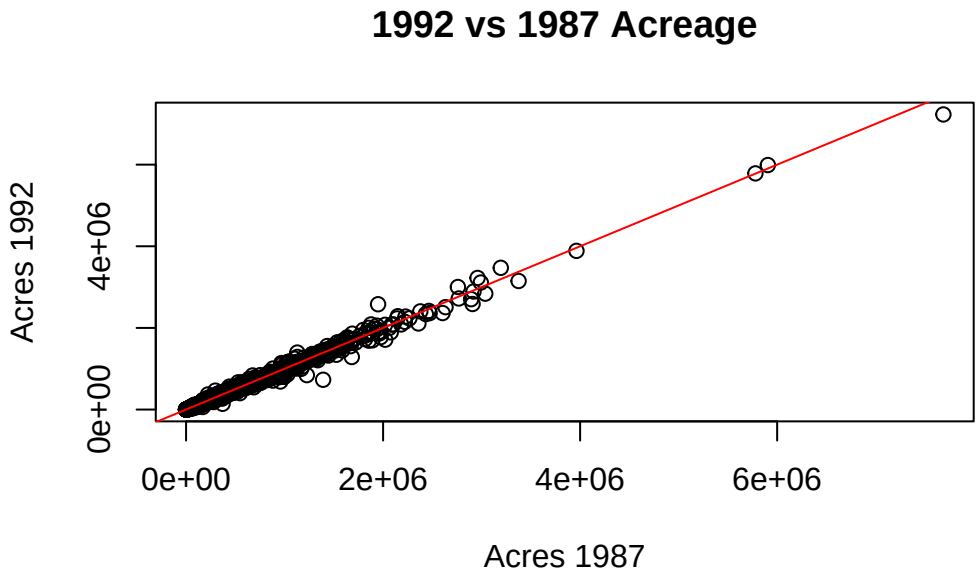

Figure 2.2: 1992 vs. 1987 farm acreage, with the 1:1 line

2.6.2 Saving and re-inserting an external plot

A high-resolution figure for a journal or a slide deck can be written straight to disk: open a graphics device (`png()`, `pdf()`, `jpeg()`), run the plotting code, then close the device with `dev.off()`. When `res` is raised, `width` and `height` must be given in inches so the figure does not shrink.

```
dir.create("figures", showWarnings = FALSE)

png("figures/agpop_scatter_highres.png",
    width = 6, height = 4, units = "in", res = 300)

plot(agpop$acres87, agpop$acres92,
     main = "1992 vs 1987 Acreage",
     xlab = "Acres 1987", ylab = "Acres 1992")
abline(a = 0, b = 1, col = "red")

dev.off()
```

The saved file can then be placed back into the document with `knitr::include_graphics()`.

```
knitr::include_graphics("figures/agpop_scatter_highres.png")
```

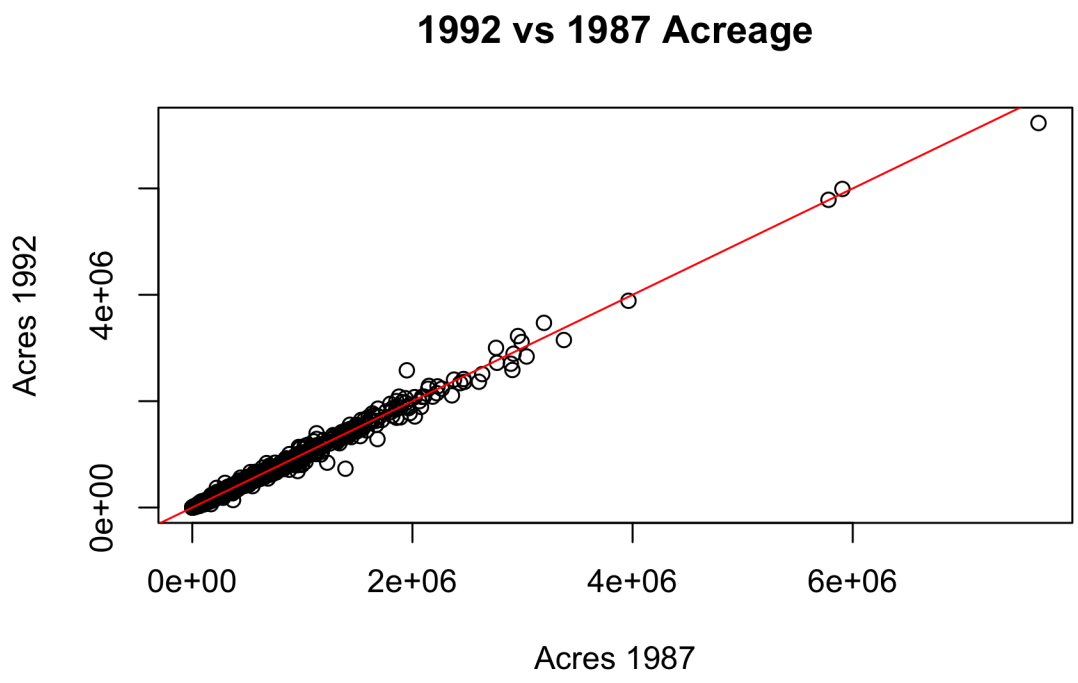

Figure 2.3: The saved high-resolution PNG, re-inserted

2.6.3 Advanced visualization with ggplot2

`ggplot2` provides a grammar of graphics for layered displays. Below, a histogram of log-transformed 1992 acreage carries both an empirical density curve and a fitted normal curve.

```
ag_clean <- agpop |>
  filter(!is.na(acres92), acres92 > 0) |>
  mutate(log_acres = log(acres92))

mean_log <- mean(ag_clean$log_acres)
sd_log   <- sd(ag_clean$log_acres)

ggplot(ag_clean, aes(x = log_acres)) +
  geom_histogram(aes(y = after_stat(density)), bins = 30,
                 fill = "steelblue", color = "white", alpha = 0.7) +
  geom_density(color = "darkred", linewidth = 1.2) +
  stat_function(
    fun = dnorm,
    args = list(mean = mean_log, sd = sd_log),
    color = "black", linetype = "dashed", linewidth = 1
  ) +
  labs(
    title = "Distribution of Log(Acres92)",
    subtitle = "Histogram with empirical density (red) and normal fit (black dashed)",
    x = "Log of 1992 farm acreage",
    y = "Density"
  ) +
  theme_minimal()
```

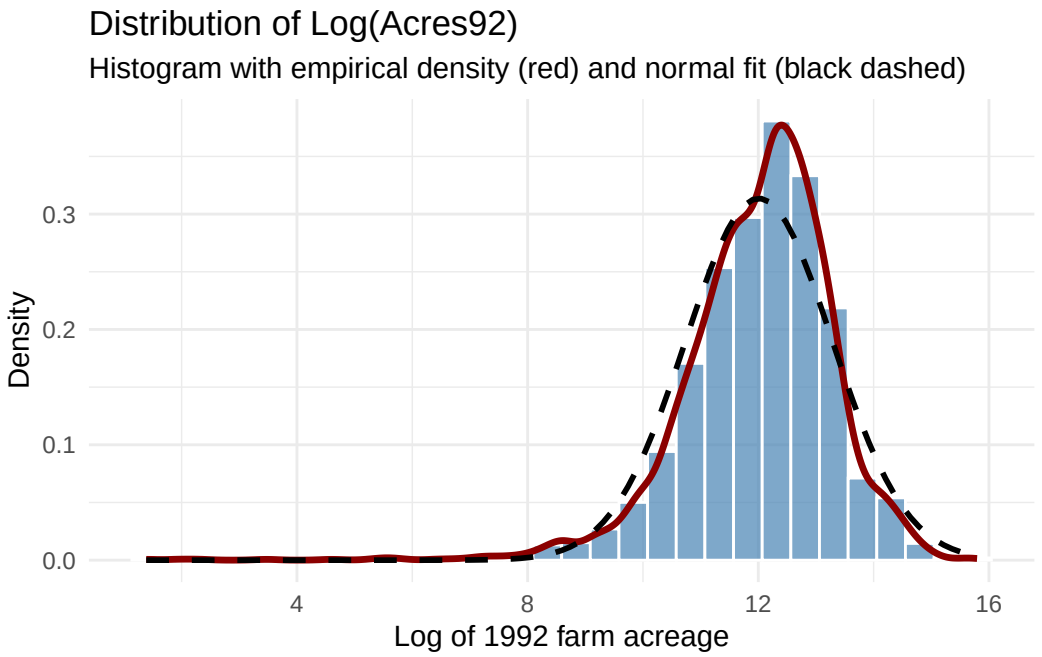

Figure 2.4: Distribution of log(acres92): histogram with empirical density and normal fit

Mapping the histogram to `after_stat(density)` puts it on the same vertical scale as the two curves, so the three layers are directly comparable.

2.7 Writing your own function

```
## data is a matrix or data frame; returns the mean of each column
means_col <- function(data) {
  n <- ncol(data)
  cmeans <- rep(NA_real_, n)
  for (j in seq_len(n)) {
    cmeans[j] <- mean(data[[j]], na.rm = TRUE)
  }
  names(cmeans) <- colnames(data)
  cmeans
}
```

Indexing with `data[[j]]` extracts column `j` as a vector for both data frames and tibbles, whereas `data[, j]` returns a one-column tibble in the tibble case. `seq_len(n)` is used in place of `1:n` so that a zero-column input yields an empty loop rather than iterating over `1, 0`.

```
agpop_numeric <- agpop |> select(where(is.numeric))

custom_means  <- means_col(agpop_numeric)
builtin_means <- colMeans(agpop_numeric, na.rm = TRUE)

data.frame(
  Variable        = colnames(agpop_numeric),
  Custom_Function = custom_means,
  Built_in        = builtin_means,
  row.names       = NULL
) |>
  gt() |>
  tab_header(
    title = "Comparing Mean Calculations",
    subtitle = "Custom function vs. base R colMeans()"
  ) |>
  cols_label(
    Variable        = md("**Variable**"),
    Custom_Function = md("**Custom `means_col()`**"),
    Built_in        = md("**Base `colMeans()`**")
  ) |>
  fmt_number(columns = c(Custom_Function, Built_in), decimals = 2)
```

```
all.equal(custom_means, builtin_means)
```

```
[1] TRUE
```

Table 2.4: Comparing column means: custom function vs. base R colMeans()

Comparing Mean Calculations
Custom function vs. base R colMeans()

Variable	Custom `means_col()`	Base `colMeans()`
acres92	308,582.41	308,582.41
acres87	315,373.71	315,373.71
acres82	321,972.52	321,972.52
farms92	625.50	625.50
farms87	678.28	678.28
farms82	728.06	728.06
largef92	56.18	56.18
largef87	54.86	54.86
largef82	52.62	52.62
smallf92	54.09	54.09
smallf87	59.54	59.54
smallf82	60.97	60.97

Selecting numeric columns with `where(is.numeric)` is preferable to the positional `select(3:13)`, which breaks silently if the column order of the source file changes. The `all.equal()` result confirms agreement with the optimized built-in.

3 Simple Random Sampling

Simple random sampling is the simplest probability sampling design, and the building block for all the designs in later chapters. A stratified sample is an SRS within each stratum; a cluster sample is an SRS of clusters. This chapter defines the design, derives its estimators and their standard errors, shows how to build confidence intervals and choose the sample size, and checks the theory by simulation. The material corresponds to Sections 2.3–2.6 and 2.8 of Lohr and Sections 4.1–4.5 of Scheaffer et al.

3.1 SRS and SRSWR

There are two versions of simple random sampling.

A **simple random sample with replacement (SRSWR)** of size n from a population of N units can be thought of as n independent draws of one unit, each unit having probability $1/N$ on every draw. The same unit can be drawn more than once.

A **simple random sample without replacement (SRS)** of size n is selected so that every possible subset of n distinct units in the population has the same probability of being the sample. There are $\binom{N}{n}$ such subsets, so

$$P(\mathcal{S}) = \frac{1}{\binom{N}{n}} = \frac{n!\,(N-n)!}{N!} \tag{3.1}$$

for every subset $\mathcal{S}$ of size n. Unless stated otherwise, "SRS" in this book means sampling without replacement. In practice an SRS is drawn with computer-generated random numbers; in R,

```
sample(1:N, size = n)
```

returns the labels of n distinct units chosen in this way.

Figure 3.1 illustrates an SRS of $n = 60$ from a synthetic population of $N = 1000$ units. The design uses no information about the units other than their labels, so the sampled units (red) are scattered at random over the population, and their values cover its whole range.

3.1.1 Inclusion probabilities

To study which units end up in the sample, define the sample-membership indicator

$$Z_i = \begin{cases} 1 & \text{if unit } i \text{ is in the sample,} \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

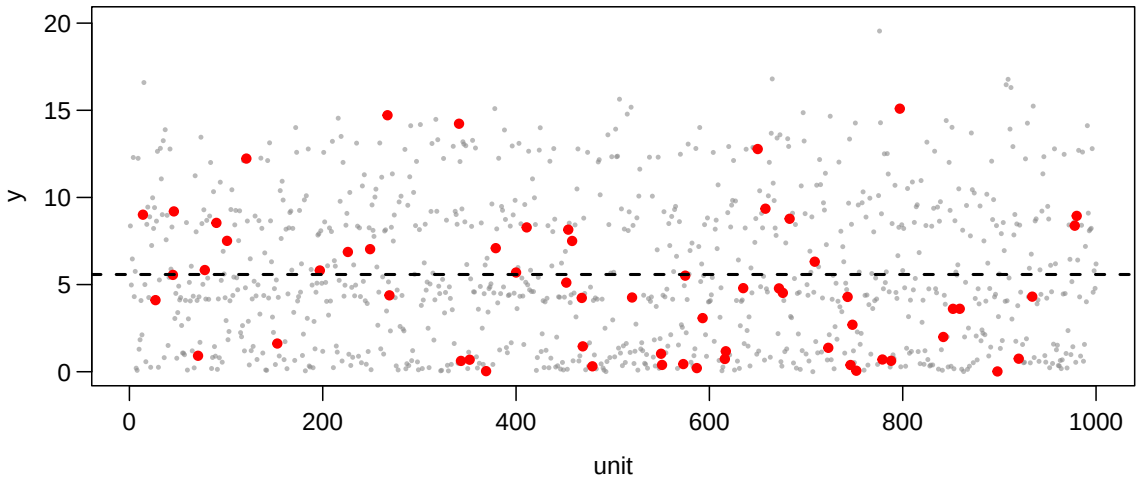

Figure 3.1: A simple random sample of $n = 60$ (red) from a synthetic population of $N = 1000$ units, listed in an arbitrary order. The dashed line is the population mean.

The probability $\pi_i = P(Z_i = 1)$ is the **inclusion probability** of unit i. If unit i is in the sample, the other $n-1$ sampled units must come from the remaining $N-1$ units, so

$$\pi_i = P(Z_i = 1) = \frac{\text{number of samples including unit } i}{\text{number of possible samples}} = \frac{\binom{N-1}{n-1}}{\binom{N}{n}} = \frac{n}{N}. \tag{3.3}$$

Every unit has the same chance n/N of being in the sample. The indicators $Z_1, \ldots, Z_N$ are the only random quantities in design-based inference; we use them in Section 3.6 to prove the properties of the estimators.

3.2 Estimators for Simple Random Sampling

Write y_i for the value of unit i in the population $U = \{1, \ldots, N\}$, and

$$\bar{y}_U = \frac{1}{N}\sum_{i=1}^{N} y_i, \qquad t = \sum_{i=1}^{N} y_i = N\bar{y}_U, \qquad S^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \bar{y}_U)^2 \tag{3.4}$$

for the population mean, total and variance. These are fixed, unknown numbers.

3.2.1 Estimating the population mean

The natural estimator of $\bar{y}_U$ is the sample mean, and the natural estimator of S^2 is the sample variance:

$$\bar{y} = \frac{1}{n}\sum_{i\in\mathcal{S}} y_i = \frac{1}{n}\sum_{i=1}^{N} Z_i y_i, \qquad s^2 = \frac{1}{n-1}\sum_{i\in\mathcal{S}}(y_i - \bar{y})^2. \tag{3.5}$$

The sample mean is unbiased for $\bar{y}_U$, and its variance is

$$V(\bar{y}) = \left(1 - \frac{n}{N}\right)\frac{S^2}{n} \tag{3.6}$$

(both are proved in Section 3.6). This is the familiar S^2/n of independent sampling, multiplied by the factor $1 - n/N$, the **finite population correction** (fpc). Sampling without replacement removes units from the pool, so the sample covers more of the population than independent draws would, and the variance is smaller. When the sampling fraction n/N is small, the fpc is close to 1 and can be ignored. When $n = N$, the sample is a census and $V(\bar{y}) = 0$.

Since s^2 is unbiased for S^2 (also proved in Section 3.6), replacing S^2 by s^2 gives an unbiased estimator of the variance. Its square root is the **standard error**:

$$\boxed{\hat{V}(\bar{y}) = \left(1 - \frac{n}{N}\right)\frac{s^2}{n}, \qquad \mathrm{SE}(\bar{y}) = \sqrt{\left(1 - \frac{n}{N}\right)\frac{s^2}{n}}.} \tag{3.7}$$

3.2.2 Estimating the population total

Since $t = N\bar{y}_U$, the total is estimated by scaling up the sample mean:

$$\boxed{\hat{t} = N\bar{y}, \qquad \hat{V}(\hat{t}) = N^2\left(1 - \frac{n}{N}\right)\frac{s^2}{n}, \qquad \mathrm{SE}(\hat{t}) = N\,\mathrm{SE}(\bar{y}).} \tag{3.8}$$

Sampling weights. The estimated total can also be written with **sampling weights**. The sampling weight of a unit is the reciprocal of its inclusion probability, $w_i = 1/\pi_i = N/n$, and

$$\hat{t} = \sum_{i \in \mathcal{S}} w_i y_i = \sum_{i \in \mathcal{S}} \frac{N}{n} y_i. \tag{3.9}$$

Each sampled unit "represents" N/n units of the population: itself and $N/n - 1$ unsampled ones. In an SRS all weights are equal; in later chapters (stratified, cluster and unequal-probability sampling) they are not, but the weighted-sum form of $\hat{t}$ carries over.

3.2.3 Coefficient of variation

The standard error is in the units of y. A scale-free measure of precision is the **coefficient of variation** (CV), defined when $\bar{y}_U \neq 0$:

$$\mathrm{CV}(\bar{y}) = \frac{\sqrt{V(\bar{y})}}{E(\bar{y})} = \sqrt{1 - \frac{n}{N}}\,\frac{S}{\sqrt{n}\,\bar{y}_U}, \qquad \widehat{\mathrm{CV}}(\bar{y}) = \frac{\mathrm{SE}(\bar{y})}{\bar{y}}. \tag{3.10}$$

Because $\hat{t} = N\bar{y}$, the factor N cancels and $\mathrm{CV}(\hat{t}) = \mathrm{CV}(\bar{y})$. A CV of 0.05 means the standard error is 5% of the estimate, whatever the units of measurement.

3.2.4 Estimating a proportion

Estimating a proportion is a special case of estimating a mean. Define $y_i = 1$ if unit i has the characteristic of interest and $y_i = 0$ otherwise. Then the population proportion and the sample proportion are

$$p = \frac{1}{N}\sum_{i=1}^{N} y_i = \bar{y}_U, \qquad \hat{p} = \frac{1}{n}\sum_{i\in\mathcal{S}} y_i = \bar{y}, \tag{3.11}$$

so $\hat{p}$ is unbiased for p. For a 0/1 variable $y_i^2 = y_i$, and the variances simplify:

$$S^2 = \frac{N}{N-1}\,p(1-p), \qquad s^2 = \frac{n}{n-1}\,\hat{p}(1-\hat{p}). \tag{3.12}$$

Substituting into the general formulas,

$$V(\hat{p}) = \Big(\frac{N-n}{N-1}\Big)\frac{p(1-p)}{n}, \qquad \boxed{\hat{V}(\hat{p}) = \Big(1-\frac{n}{N}\Big)\frac{\hat{p}(1-\hat{p})}{n-1}.} \tag{3.13}$$

The number of units with the characteristic, Np, is estimated by $N\hat{p}$ with standard error $N\,\mathrm{SE}(\hat{p})$.

3.3 Example: The Census of Agriculture

The U.S. government conducts a Census of Agriculture every five years. It collects data on all farms (any place producing \$1000 or more of agricultural products) in the 50 states, for each of the $N = 3078$ counties and county-equivalents. The file `agpop.csv` contains the 1982, 1987 and 1992 farm data for the whole population. An SRS of $n = 300$ counties was selected with computer-generated random numbers and saved as `agsrs.csv`. We use it to estimate the mean and total of `acres92`, the number of acres devoted to farms in 1992, and the proportion of counties with fewer than 200,000 acres in farms.

Throughout the agriculture examples in this book, county acreages are expressed in **thousands of acres**. The raw figures run into the millions, and every mean, variance and total reads far more easily on that scale. The helper `acres_in_thousands()` performs the conversion as the data are read.

3.3.1 Step-by-step calculation

We first compute the estimates directly from the formulas of Section 3.2.

```
## read survey data
agsrs <- read.csv ("data/agsrs.csv")
## work in thousands of acres, the unit used for every agriculture example here
agsrs[c("acres92", "acres87", "acres82")] <- agsrs[c("acres92", "acres87", "acres82")] / 1000
head(agsrs)
```

```
             county state acres92 acres87 acres82 farms92 farms87 farms82 largef92 largef87 largef82 smallf92
1     COFFEE COUNTY    AL 175.209 179.311 194.509     760     842     944       29       28       21       57
2    COLBERT COUNTY    AL 138.135 145.104 161.360     488     563     686       37       41       42       12
3      LAMAR COUNTY    AL  56.102  59.861  72.334     299     362     447        4        4        3       16
4    MARENGO COUNTY    AL 199.117 220.526 231.207     434     471     622       48       66       62       14
5     MARION COUNTY    AL  89.228 105.586 113.618     566     658     748        7        9        9       11
6 TUSCALOOSA COUNTY    AL  96.194 120.542 134.616     436     521     650       20       17       23       18
  smallf87 smallf82 region
1       47       66      S
2       44       47      S
3       20       30      S
4       11       28      S
5       23       27      S
6       32       29      S
```

```r
## extract the variable of interest
sdata <- agsrs$acres92
N <- 3078

## do calculation
n <- length (sdata)
ybar <- mean (sdata)
s <- sd (sdata)
se.ybar <- sqrt((1 - n / N)) * s / sqrt(n)
mem <- qt (0.975, df = n - 1) * se.ybar

## return estimate vector for pop mean
c (Est. = ybar, S.E. = se.ybar, ci.low = ybar - mem, ci.upp = ybar + mem)
```

```
     Est.      S.E.    ci.low    ci.upp
297.89705  18.89843 260.70626 335.08784
```

```r
## return estimate vector for pop total
c (Est. = ybar, S.E. = se.ybar, ci.low = ybar - mem, ci.upp = ybar + mem) * N
```

```
      Est.       S.E.     ci.low     ci.upp
 916927.11   58169.38  802453.86 1031400.36
```

In thousands of acres, the sample gives $\bar{y} = 297.897$ and $s = 344.55$, so

$$\mathrm{SE}(\bar{y}) = \sqrt{\left(1 - \frac{300}{3078}\right)\frac{344.55^2}{300}} = 18.898, \tag{3.14}$$

$$\hat{t} = N\bar{y} = 3078(297.897) = 916,927, \qquad \mathrm{SE}(\hat{t}) = 3078(18.898) = 58,169. \tag{3.15}$$

The average county has about 298 thousand acres in farms, and the U.S. total is estimated at about 917 million acres. The estimated coefficient of variation is

$$\widehat{\mathrm{CV}}(\hat{t}) = \widehat{\mathrm{CV}}(\bar{y}) = \frac{\mathrm{SE}(\bar{y})}{\bar{y}} = \frac{18.898}{297.897} = 0.0634, \tag{3.16}$$

so the standard error is about 6.3% of the estimate.

The data are highly skewed (Figure 3.2): most counties have fewer than 500 thousand acres in farms, but some have more than 1.5 million. For such data the mean is pulled up by the long right tail, and we should also report the **median**, which is 196.7 thousand acres here, well below the mean.

```
hist(sdata, breaks = 30, col = "lightblue", border = "white",
     xlab = "acres92 (thousands of acres)", main = "")
abline(v = c(ybar, median(sdata)), col = c("firebrick", "darkgreen"), lwd = 2, lty = c(1, 2))
legend("topright", bty = "n", lwd = 2, lty = c(1, 2), col = c("firebrick", "darkgreen"),
       legend = c("mean", "median"))
```

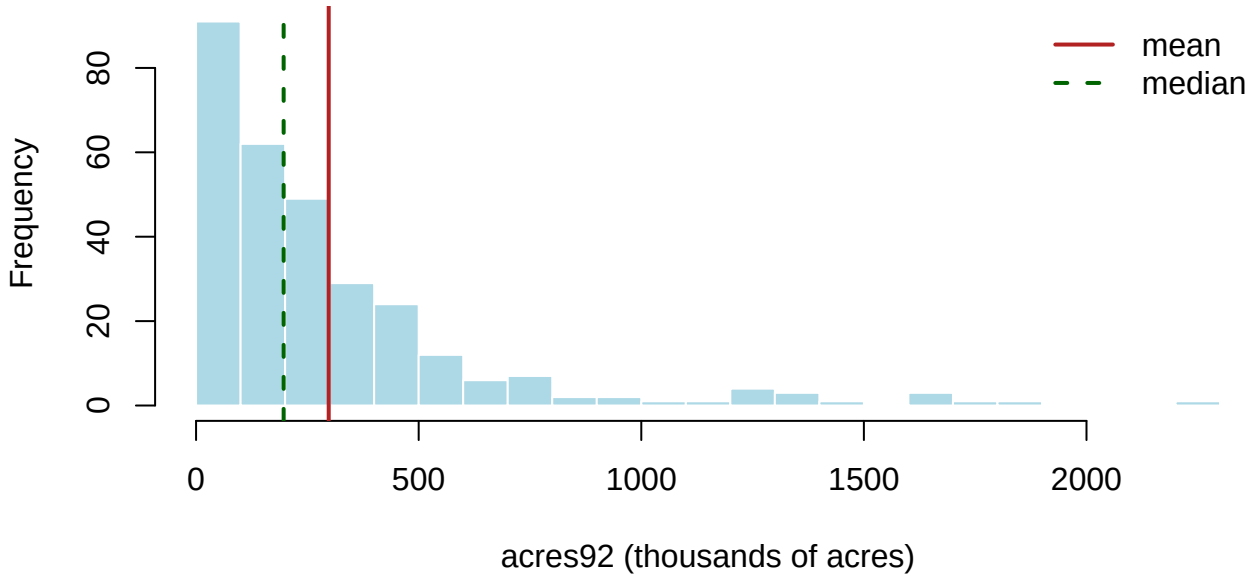

Figure 3.2: Farm acreage in 1992 (thousands of acres) for the SRS of 300 counties. The distribution is strongly skewed to the right.

A proportion. Let $y_i = 1$ if county i has fewer than 200 thousand acres in farms. In the sample, 153 of the 300 counties do, so $\hat{p} = 153/300 = 0.51$ and

$$\mathrm{SE}(\hat{p}) = \sqrt{\left(1 - \frac{300}{3078}\right)\frac{(0.51)(0.49)}{299}} = 0.0275. \tag{3.17}$$

3.3.2 A function for repeated use

The same calculation is needed again and again, so we wrap it in a function. To avoid repeating code in every chapter, all estimating functions (and their `gt` table formatting helpers) live in a single shared file, `samplingestimate.r`, sourced at the start of this chapter. The SRS function is `srs_est()`.

The full source of `samplingestimate.r` is listed in Appendix C.

```
agsrs <- acres_in_thousands (read.csv ("data/agsrs.csv"))
```

Estimating the mean of acres92. The function's `$table` shows the row-level working values (the first 6 rows and the Sum), and `$estimate` holds the point estimate, SE and CI, formatted with `format_est_gt()`.

```
res_mean <- srs_est(agsrs[,"acres92"], N = 3078, estimate = "mean")
res_mean$table
```

SRS Mean Estimation: Working Table
$n = 300$

i	y_i	$\hat{y}_i$	e_i	$(y_i - \hat{y}_i)^2$[1]
1	175.21	297.90	-122.69	15,052.36
2	138.13	297.90	-159.76	25,523.91
3	56.10	297.90	-241.80	58,464.84
4	199.12	297.90	-98.78	9,757.50
5	89.23	297.90	-208.67	43,542.77
6	96.19	297.90	-201.70	40,684.12
...	...	...	...	...
Sum	**89,369.11**	**297.90**	**0.00**	**35,496,086.45**

[1] $\hat{y}_i = \bar{y}$ for every unit under the SRS model; $s_y^2 = \sum_i (y_i - \hat{y}_i)^2/(n-1) = 1.19 \times 10^5$.

$$\bar{y} = \frac{\sum_i y_i}{n} = \frac{89,369.11}{300} = 297.90$$

$$\text{SE}(\bar{y}) = \sqrt{1 - \frac{n}{N}}\frac{s_y}{\sqrt{n}} = \sqrt{1 - \frac{300}{3078}}\frac{344.55}{\sqrt{300}} = 18.90$$

```
res_mean$estimate |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
297.90	18.90	260.71	335.09

Estimating the total of acres92. Setting `estimate = "total"` makes the function scale the mean and confidence interval by $N = 3078$.

```
res_total <- srs_est(agsrs[,"acres92"], N = 3078, estimate = "total", show.details = FALSE)
res_total$estimate |> format_est_gt("total")
```

$\hat{t}$	SE($\hat{t}$)	95% CI Lower	95% CI Upper
916,927.11	58,169.38	802,453.86	1,031,400.36

Estimating the proportion of counties with fewer than 200K acres. We create the 0/1 indicator and estimate its mean.

```r
acres92.is.fewer.200k <- as.numeric (agsrs[,"acres92"] < 200)   # 200 thousand acres
head(acres92.is.fewer.200k)
```

```
[1] 1 1 1 1 1 1
```

```r
res_prop <- srs_est(acres92.is.fewer.200k, N = 3078, estimate = "mean", show.details = FALSE)
res_prop$estimate |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
0.51	0.03	0.46	0.56

Estimating the number of such counties.

```r
res_prop_total <- srs_est(acres92.is.fewer.200k, N = 3078, estimate = "total", show.details =
↪  FALSE)
res_prop_total$estimate |> format_est_gt("total")
```

$\hat{t}$	SE($\hat{t}$)	95% CI Lower	95% CI Upper
1,569.78	84.54	1,403.42	1,736.14

3.3.3 Comparing with the true values

Because the census file `agpop.csv` contains the whole population, we can check how close the SRS estimates came to the true values. (In a real survey these are, of course, unknown.)

```r
agpop <- acres_in_thousands (read.csv ("data/agpop.csv"))
#true mean (thousands of acres)
mean (agpop[, "acres92"], na.rm = T)
```

```
[1] 308.5824
```

```r
# true total (thousands of acres)
sum (agpop[, "acres92"], na.rm = T)
```

```
[1] 943953.6
```

```r
# true proportion of counties with less than 200K acres for farming
mean (agpop[, "acres92"] < 200, na.rm = T)
```

```
[1] 0.5145472
```

```r
# true number of counties with less than 200K acres for farming
sum (agpop[, "acres92"] < 200, na.rm = T)
```

```
[1] 1574
```

All the true values lie inside the 95% confidence intervals computed above.

3.4 Confidence Intervals

A confidence interval (CI) indicates the accuracy of an estimate. A 95% CI is often explained heuristically: if we took samples from the population over and over again, and constructed a CI by our procedure for each possible sample, we would expect 95% of the resulting intervals to include the true value of the population quantity. In design-based inference this is literally what the probability refers to: the population is fixed and the randomness is over the $\binom{N}{n}$ possible samples.

3.4.1 Normality of the sample mean

In an SRS without replacement the observations are not independent, so the ordinary central limit theorem does not apply directly. A finite-population version is due to Hájek (1960): if certain technical conditions hold and n, N and $N - n$ are all "sufficiently large", then the sampling distribution of

$$\frac{\bar{y} - \bar{y}_U}{\sqrt{(1 - \frac{n}{N})}\,\frac{S}{\sqrt{n}}} \tag{3.18}$$

is approximately standard normal. Replacing S by s gives the standard error, and approximate $100(1-\alpha)\%$ CIs for the mean, total and proportion are

$$\bar{y} \pm t_{\alpha/2,\,n-1}\,\mathrm{SE}(\bar{y}), \qquad \hat{t} \pm t_{\alpha/2,\,n-1}\,\mathrm{SE}(\hat{t}), \qquad \hat{p} \pm t_{\alpha/2,\,n-1}\,\mathrm{SE}(\hat{p}). \tag{3.19}$$

The t quantile is customary; for the sample sizes used in surveys it is very close to the normal quantile $z_{\alpha/2}$.

Example (continued). For the agriculture sample, $t_{0.025,\,299} = 1.968$, and the 95% CIs are

$$\text{mean: } 297.9 \pm 1.968(18.9) = [260.7,\ 335.1] \text{ thousand acres}, \tag{3.20}$$

$$\text{total: } [802,454,\ 1,031,400] \text{ thousand acres}, \tag{3.21}$$

$$\text{proportion: } 0.51 \pm 1.968(0.0275) = [0.456,\ 0.564], \tag{3.22}$$

matching the output of `srs_est()` in Section 3.3.

3.4.2 How large must n be for normality?

The normal approximation needs a larger sample when the population is skewed, as acreage is. Sugden, Smith and Jones (2000) extend Cochran's rule and recommend the minimum sample size

$$n_{\min} = 28 + 25 \left(\frac{\sum_{i=1}^{N} (y_i - \bar{y}_U)^3}{N S^3} \right)^2, \tag{3.23}$$

where the quantity in parentheses is the **skewness** of the population. The more skewed the population, the larger the sample required.

```
g1 <- (sum((sdata - ybar)^3) / n) / s^3   # sample skewness
n_min <- 28 + 25 * g1^2
c(skewness = g1, n_min = n_min)
```

```
  skewness      n_min
  2.567866 192.848373
```

For the agriculture data we substitute the sample quantities: $s = 344.55$ and $\sum_{i \in \mathcal{S}} (y_i - \bar{y})^3 / n = 1.0504 \times 10^8$ (in thousands of acres), giving a skewness of 2.57 and

$$n_{\min} = 28 + 25(2.568)^2 \approx 193. \tag{3.24}$$

Our sample of $n = 300$ exceeds $n_{\min}$, so it appears large enough for the sampling distribution of $\bar{y}$ to be approximately normal. The simulation in Section 3.7 confirms this.

3.5 Choosing the Sample Size

Before a survey is carried out we must decide how large the sample should be. The answer depends on how precise the estimate must be.

3.5.1 Specifying the tolerable error

The desired precision is often expressed in absolute terms,

$$P(|\bar{y} - \bar{y}_U| \le e) = 1 - \alpha, \tag{3.25}$$

where e is the **margin of error**. For many surveys of people that measure a proportion, $e = 0.03$ and $\alpha = 0.05$. Alternatively, precision can be specified in relative terms, which controls the CV:

$$P\left(\left| \frac{\bar{y} - \bar{y}_U}{\bar{y}_U} \right| \le r \right) = 1 - \alpha, \qquad \text{that is,} \qquad e = r\,\bar{y}_U. \tag{3.26}$$

For example, if $\bar{y}_U \approx 180$ thousand and $r = 0.1$, then $e = 18$ thousand.

3.5.2 Solving for n

Using the normal approximation, the margin of error is $z_{\alpha/2}\,\mathrm{SD}(\bar{y})$, so we need n with

$$e = z_{\alpha/2}\sqrt{\left(1-\frac{n}{N}\right)\frac{S^2}{n}}. \tag{3.27}$$

Solving in two steps is convenient. First find the sample size n_0 we would need for an infinite population (no fpc):

$$n_0 = \left(\frac{z_{\alpha/2}\,S}{e}\right)^2. \tag{3.28}$$

Then adjust for the finite population:

$$\boxed{n = \frac{n_0}{1+n_0/N} = \frac{z_{\alpha/2}^2 S^2}{e^2 + z_{\alpha/2}^2 S^2/N}.} \tag{3.29}$$

When N is large relative to n_0, $n \approx n_0$; the population size matters only when n_0 is a substantial fraction of N. The formula needs a value for S, which is unknown. It can come from a pilot sample, a previous survey of a similar population, or a guess. For a proportion, $S^2 \approx p(1-p) \le 1/4$, so using $p = 1/2$ gives a conservative (largest) sample size.

```
srs_size <- function(S, e, N = Inf, alpha = 0.05) {
  n0 <- (qnorm(1 - alpha / 2) * S / e)^2
  c(n0 = n0, n = n0 / (1 + n0 / N))
}
```

Example: margin of error 0.03 for a proportion. We want to estimate the proportion of the $N = 1251$ recipes in a cookbook that do not involve animal products, with a 95% CI of margin of error 0.03. Using the conservative $p = 1/2$,

$$n_0 = \frac{(1.96)^2(\tfrac{1}{2})(1-\tfrac{1}{2})}{(0.03)^2} \approx 1067. \tag{3.30}$$

Since n_0 is large relative to $N = 1251$, the fpc adjustment matters:

$$n = \frac{n_0}{1+n_0/N} = \frac{1067}{1+1067/1251} \approx 576. \tag{3.31}$$

If N were very large we would use $n \approx n_0 = 1067$.

```
srs_size(S = 0.5, e = 0.03, N = 1251)
```

```
      n0       n
1067.0719 575.8695
```

Example: the agriculture survey. A pilot sample of 30 counties was taken (one county was missing `acres92`). The sample SD of the remaining 29 observations was 519.085 thousand acres. With a desired margin of error of 60 thousand acres,

$$n_0 = (1.96)^2 \frac{519.085^2}{60^2} \approx 288, \qquad n = \frac{288}{1 + 288/3078} \approx 263. \tag{3.32}$$

The investigators took a sample of 300, in case the pilot's estimate of the SD was too low.

```
srs_size(S = 519.085, e = 60, N = 3078)
```

```
      n0       n
287.5217 262.9583
```

3.6 Randomization Theory for SRS

We now prove the properties of $\bar{y}$ and s^2 used above. In the **randomization theory** (design-based) approach, the values $y_1, \dots, y_N$ are fixed but unknown numbers. The only random variables are the indicators $Z_1, \dots, Z_N$ that record which units are in the sample:

$$\bar{y} = \sum_{i \in \mathcal{S}} \frac{y_i}{n} = \sum_{i=1}^{N} Z_i \frac{y_i}{n}. \tag{3.33}$$

All the randomness in $\bar{y}$ comes from $Z_1, \dots, Z_N$.

3.6.1 Moments of Z_i

Each Z_i is a Bernoulli random variable with $P(Z_i = 1) = \pi_i = n/N$ (Section 3.1). Since $Z_i^2 = Z_i$,

$$E[Z_i] = E[Z_i^2] = \frac{n}{N}, \qquad V(Z_i) = \frac{n}{N} - \left(\frac{n}{N}\right)^2 = \frac{n}{N}\left(1 - \frac{n}{N}\right). \tag{3.34}$$

The Z_i are not independent, because the sample size is fixed: if unit i is in the sample, there is one less place for the other units. For $i \neq j$, given that unit i is sampled, the remaining $n-1$ units are an SRS from the other $N-1$, so

$$E[Z_i Z_j] = P(Z_j = 1 \mid Z_i = 1)\, P(Z_i = 1) = \frac{n-1}{N-1} \cdot \frac{n}{N}, \tag{3.35}$$

$$\mathrm{Cov}(Z_i, Z_j) = E[Z_i Z_j] - E[Z_i]E[Z_j] = \frac{n}{N}\left(\frac{n-1}{N-1} - \frac{n}{N}\right) = -\frac{1}{N-1}\left(1 - \frac{n}{N}\right)\frac{n}{N}. \tag{3.36}$$

This **negative** covariance is the source of the finite population correction.

3.6.2 Unbiasedness of $\bar{y}$

By linearity of expectation,

$$E[\bar{y}] = \sum_{i=1}^{N} E[Z_i]\frac{y_i}{n} = \sum_{i=1}^{N} \frac{n}{N} \cdot \frac{y_i}{n} = \frac{1}{N}\sum_{i=1}^{N} y_i = \bar{y}_U. \tag{3.37}$$

So $\bar{y}$ is an **unbiased** estimator of $\bar{y}_U$, and $\hat{t} = N\bar{y}$ is unbiased for t.

3.6.3 Variance of $\bar{y}$

Expanding the variance of a sum,

$$\begin{aligned} V(\bar{y}) &= \frac{1}{n^2}\left[\sum_{i=1}^{N} y_i^2\, V(Z_i) + \sum_{i=1}^{N}\sum_{j\neq i} y_i y_j \,\mathrm{Cov}(Z_i, Z_j)\right] \\ &= \frac{1}{n^2}\,\frac{n}{N}\Big(1-\frac{n}{N}\Big)\left[\sum_{i=1}^{N} y_i^2 - \frac{1}{N-1}\sum_{i=1}^{N}\sum_{j\neq i} y_i y_j\right]. \end{aligned} \tag{3.38}$$

Using $\sum_i \sum_{j\neq i} y_i y_j = \big(\sum_i y_i\big)^2 - \sum_i y_i^2$, the term in brackets is

$$\frac{N\sum_i y_i^2 - \big(\sum_i y_i\big)^2}{N-1} = \frac{N\sum_i (y_i - \bar{y}_U)^2}{N-1} = NS^2. \tag{3.39}$$

Therefore

$$V(\bar{y}) = \frac{1}{n^2}\,\frac{n}{N}\Big(1-\frac{n}{N}\Big)NS^2 = \Big(1-\frac{n}{N}\Big)\frac{S^2}{n}. \tag{3.40}$$

Without the negative covariances, the result would be $\frac{N-1}{N}\cdot\frac{S^2}{n}$ (the SRSWR variance). The covariance terms produce the factor $1-n/N$.

3.6.4 Unbiasedness of s^2

Adding and subtracting $\bar{y}_U$ inside the sum of squares gives the identity

$$\sum_{i\in\mathcal{S}} (y_i - \bar{y})^2 = \sum_{i\in\mathcal{S}} (y_i - \bar{y}_U)^2 - n(\bar{y} - \bar{y}_U)^2. \tag{3.41}$$

Take expectations. For the first term, $E[\sum_{i=1}^{N} Z_i (y_i - \bar{y}_U)^2] = \frac{n}{N}(N-1)S^2$; for the second, $E[n(\bar{y}-\bar{y}_U)^2] = nV(\bar{y}) = (1-\frac{n}{N})S^2$. Hence

$$E\left[\sum_{i\in\mathcal{S}} (y_i - \bar{y})^2\right] = \frac{n(N-1)}{N}S^2 - \frac{N-n}{N}S^2 = (n-1)S^2, \tag{3.42}$$

and therefore $E[s^2] = S^2$. It follows that $\widehat{V}(\bar{y}) = (1-\frac{n}{N})\frac{s^2}{n}$ is unbiased for $V(\bar{y})$. (The standard error, its square root, is slightly biased, as a square root always is, but this is ignored in practice.)

3.7 A Simulation Demonstration of SRS Inference

The theory says that $\bar{y}$ is unbiased, that its standard deviation is $\sqrt{1-n/N}\,S/\sqrt{n}$, that it is approximately normal for $n = 300$, and that the 95% CIs cover $\bar{y}_U$ about 95% of the time. Because `agpop.csv` contains the whole population, we can check all four claims by drawing many samples from it.

```
# read population data, with acreages in thousands of acres
agpop <- acres_in_thousands (read.csv ("data/agpop.csv"))
# remove those counties with na
agpop <- subset( agpop, !is.na (acres92))
```

3.7.1 True values

We fix the sample size and compute the true population mean and the true standard deviation of $\bar{y}$ from the population S:

$$\mathrm{SD}(\bar{y}) = \sqrt{1 - \frac{n}{N}} \left(\frac{S}{\sqrt{n}} \right). \tag{3.43}$$

```
# sample size
n <- 300
# population size
N <- nrow (agpop); N
```

```
[1] 3059
```

```
# true value of population mean
ybarU <- mean (agpop[,"acres92"]); ybarU
```

```
[1] 308.5824
```

```
# true value of deviation of sample mean
true.se.ybar <- sqrt (1- n/N) * sd (agpop[,"acres92"]) / sqrt (n); true.se.ybar
```

```
[1] 23.32029
```

3.7.2 One SRS

A single random sample of $n = 300$, analyzed with `srs_est()`, is one instance of survey inference:

```
# srs sampling
srs <- sample (1:N,n)
head(agpop [srs, ])
```

```
           county state acres92 acres87 acres82 farms92 farms87 farms82 largef92 largef87 largef82 smallf92
2151  MAYES COUNTY    OK 277.981 260.252 268.039    1227    1258    1309       35       30       34       50
866  BARTON COUNTY    KS 580.199 556.433 559.531     770     937     962      188      165      170       35
2386 POTTER COUNTY    SD 507.101 552.844 541.015     321     371     341      163      180      176       10
1953 ALBANY COUNTY    NY  57.889  67.754  82.788     391     460     510        2        3        6       36
114   LOGAN COUNTY    AR 186.829 187.992 187.551     940     995    1058       21       16       14       30
81   CHICOT COUNTY    AR 269.122 320.847 323.410     344     409     419       91      103      107        3
     smallf87 smallf82 region
2151       46       55      S
866        80       51     NC
2386       24       16     NC
1953       44       29     NE
114        30       42      S
81          6        7      S
```

```r
# get data of variable "acres92"
sdata <- agpop [srs, "acres92"]
# analysis
res_one <- srs_est(sdata, N, estimate = "mean", show.details = FALSE)
res_one$estimate |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
291.31	20.90	250.17	332.44

3.7.3 Repeating SRS 5000 times

We now draw 5000 independent samples and store the estimates from each. (We use `show.details = FALSE` and keep only `$estimate`, so the loop skips building 5000 working tables.)

```r
nres <- 5000 # number of repeated sampling
simulation.results <- matrix (0, nres, 4) # matrix recording repeated results
colnames(simulation.results) <- c( "Est.",   "S.E.",   "ci.low", "ci.upp")

for (i in 1:nres)
{
    srs <- sample (N, n)
   sdata <- agpop [srs, "acres92"]
   simulation.results [i,] <- srs_est (sdata, N, estimate = "mean", show.details =
 ↪  FALSE)$estimate
}

head(simulation.results)
```

```
        Est.     S.E.   ci.low   ci.upp
[1,] 279.7295 17.80446 244.6916 314.7674
[2,] 298.6389 21.71196 255.9113 341.3665
[3,] 297.2253 20.55610 256.7723 337.6782
[4,] 300.1009 22.33300 256.1511 344.0507
[5,] 320.6076 27.30365 266.8760 374.3393
[6,] 337.7321 31.58475 275.5755 399.8887
```

```r
# look at the distribution of sample mean
par (mfrow= c(2,2))
dens_pop <- density (agpop$acres92, from = 0)
hist (agpop$acres92, breaks = 50, freq = FALSE, col = "lightblue", border = "white",
      ylim = c(0, max (dens_pop$y)),
      main = "Population of acres92", xlab = "acres92")
lines (dens_pop, col = "blue", lwd = 2)
hist (simulation.results[,1], breaks = 40, freq = FALSE, col = "lightblue", border = "white",
      main = "Sample means of acres92", xlab = "sample mean")
lines (density (simulation.results[,1]), col = "blue", lwd = 2)
curve (dnorm (x, mean = ybarU, sd = true.se.ybar), add = TRUE, col = "red", lwd = 2, lty = 2)
abline (v = ybarU, col = "red")
```

```
legend ("topright", bty = "n", lwd = 2, lty = c(1, 2), col = c("blue", "red"), cex = 0.8,
        legend = c("kernel density", "normal (theory)"))
qqnorm (simulation.results[,1], main="QQ plot of Sample Mean"); qqline(simulation.results[,1])
boxplot (simulation.results[,1], main = "Boxplot of Sample Mean")
abline (h = ybarU, col = "red")

mean (simulation.results[,1])
```

```
[1] 308.758
```

```
ybarU
```

```
[1] 308.5824
```

```
sd (simulation.results [,1])
```

```
[1] 23.4235
```

```
true.se.ybar
```

```
[1] 23.32029
```

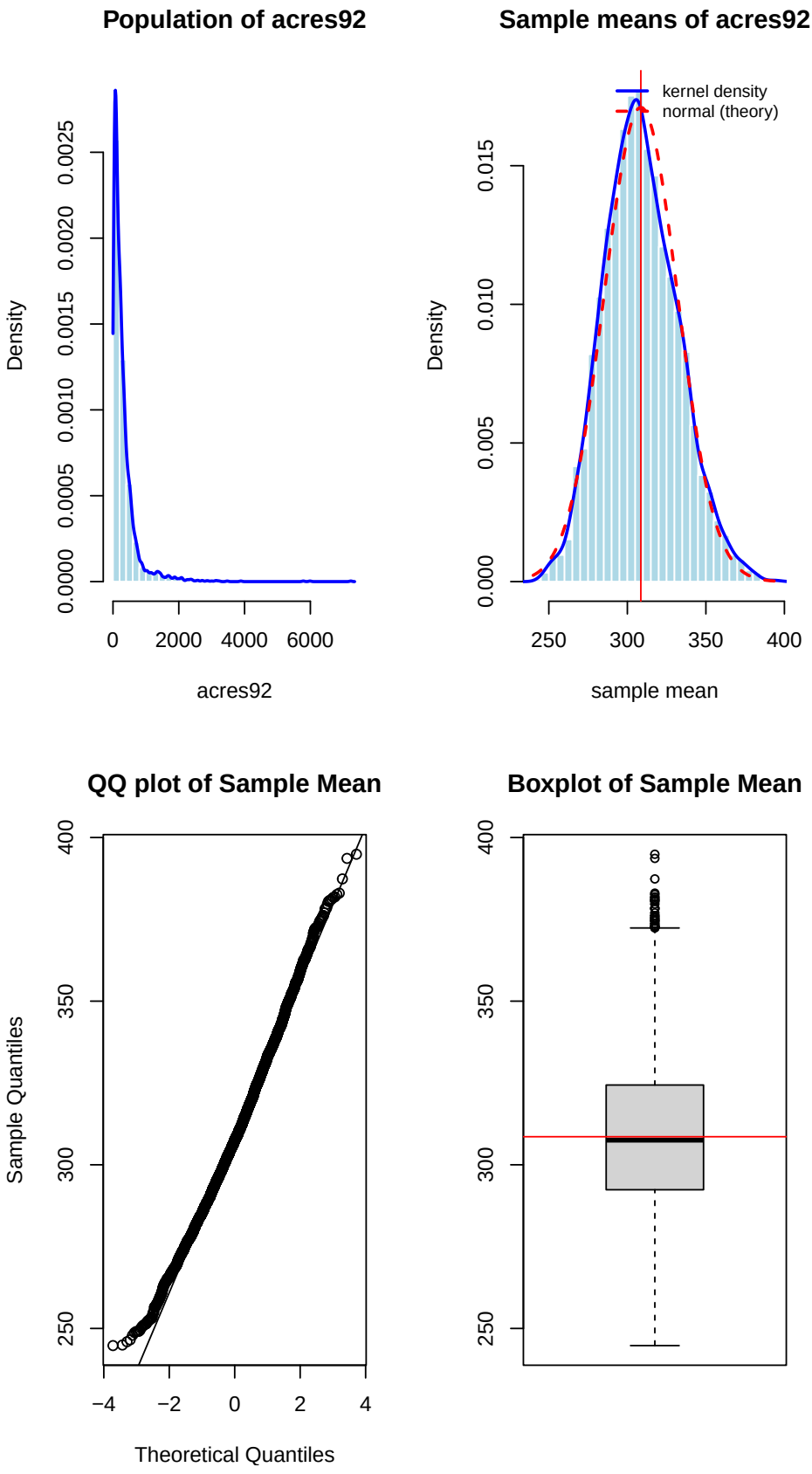

Figure 3.3: Population distribution and sampling distribution of the SRS sample mean over 5000 repeated samples (n = 300)

The blue curves are kernel density estimates. The population density (top left) is strongly skewed to the right, with a long tail of very large counties. The kernel density of the 5000 sample means (top right) is nearly symmetric and almost coincides with the normal density $N(\bar{y}_U,\ \mathrm{SD}(\bar{y})^2)$ predicted by the theory (red dashed curve); the QQ plot (bottom left) is close to a straight line, with only a slight bend in the upper tail left over from the skewness. So the sampling distribution of $\bar{y}$ is close to normal, as Hájek's theorem and the $n_{\min}$ check of Section 3.4 suggest. The average of the 5000 estimates, 308.76, is close to $\bar{y}_U = 308.58$ (unbiasedness), and their standard deviation, 23.42, is close to the theoretical value 23.32.

3.7.4 Empirical coverage rate of CIs

Finally we check whether the 95% CIs cover the true mean about 95% of the time.

```
library(plotrix)

# Convert to data frame (if it was a matrix) and use mutate to evaluate coverage
simulation.results <- as.data.frame(simulation.results) |>
  mutate(
    `Covered?` = as.numeric(ci.low < ybarU & ybarU < ci.upp)
  )

head(simulation.results)
```

```
      Est.     S.E.   ci.low   ci.upp Covered?
1 279.7295 17.80446 244.6916 314.7674        1
2 298.6389 21.71196 255.9113 341.3665        1
3 297.2253 20.55610 256.7723 337.6782        1
4 300.1009 22.33300 256.1511 344.0507        1
5 320.6076 27.30365 266.8760 374.3393        1
6 337.7321 31.58475 275.5755 399.8887        1
```

```
par(mfrow=c(1,1))
plotCI(
  x = 1:100,
  y = simulation.results$Est.[1:100],
  li = simulation.results$ci.low[1:100],
  ui = simulation.results$ci.upp[1:100],
  col = 2 - simulation.results$`Covered?`[1:100]
)
abline(h = ybarU, col = "blue")

# Empirical coverage rate
mean(simulation.results$`Covered?`)
```

```
[1] 0.9358
```

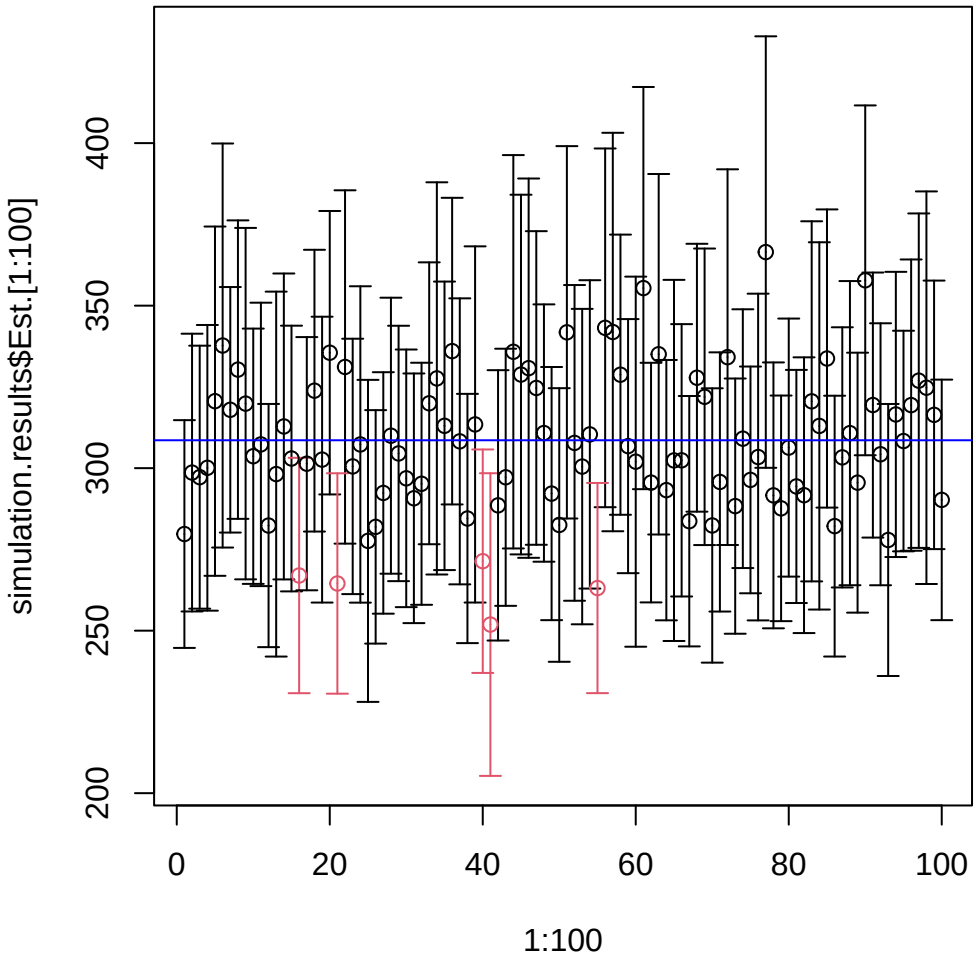

Figure 3.4: First 100 simulated 95% confidence intervals for the SRS sample mean, with coverage failures highlighted

The empirical coverage rate is 0.936, close to the nominal 0.95. (With a skewed population the coverage of the t interval tends to fall slightly short of nominal.) The intervals that miss (in red) are exactly the "5% of samples" in the heuristic interpretation of Section 3.4.

3.8 Shinylive App for Illustrating SRS Theory

The simulation for this section runs as a standalone app: Simple random sampling. It draws samples one at a time and builds up the sampling distribution and the coverage rate interactively, for any sample size and confidence level. It opens in a new tab, together with its documentation and the full list of apps.

4 Stratified Random Sampling

4.1 What Is Stratified Sampling?

Suppose the variable we are interested in has different means in different subpopulations. Then a stratified random sample can often give more precise estimates of population quantities than a simple random sample of the same size. The word *stratify* comes from Latin words meaning "to make layers". We divide the population of N units into H subpopulations, called **strata**. The strata do not overlap and together they make up the whole population, so each sampling unit belongs to exactly one stratum. We then draw an independent probability sample from each stratum and pool the information to estimate population quantities.

Figure 4.1 illustrates the design on a synthetic population of $N = 1000$ units in $H = 4$ strata of sizes $N_h = 400, 300, 200$ and 100, sorted by their means. The sample of $n = 60$ is allocated in proportion to the stratum sizes, $n_h = 60N_h/N = 24, 18, 12$ and 6, and an SRS of n_h units is drawn in each stratum, so every stratum is represented in the sample. Because the stratum means are far apart while the spread within each stratum is small, most of the variation in the population lies *between* the strata, and this part of the variation does not affect the stratified estimator (Section 4.2).

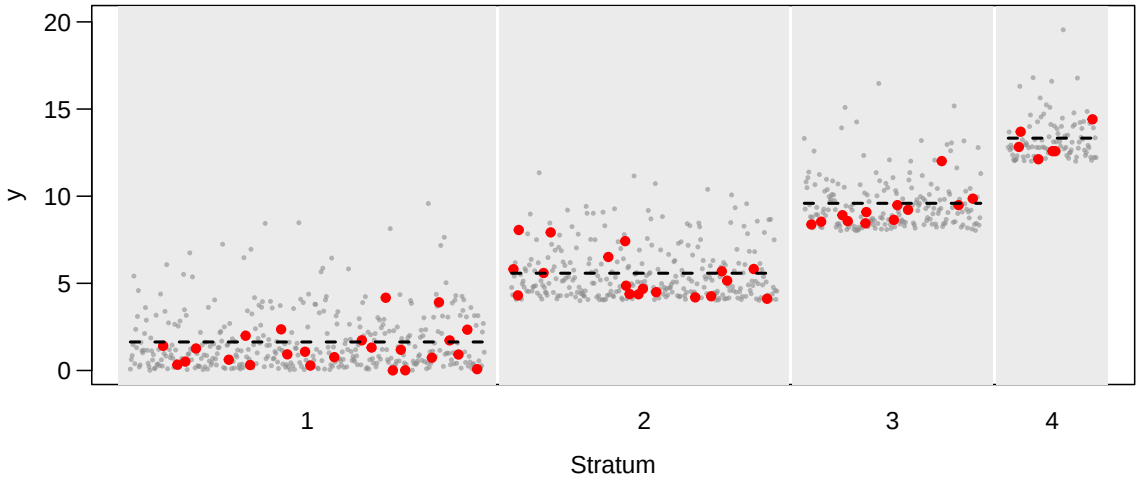

Figure 4.1: A stratified random sample from a synthetic population of $N = 1000$ units in $H = 4$ strata of sizes $N_h = 400, 300, 200, 100$, sorted by their means (dashed lines). An SRS of $n_h = 24, 18, 12, 6$ units (red) is drawn in each stratum (proportional allocation).

In this chapter the sample within each stratum is a simple random sample (SRS), drawn without replacement. The design is then called **stratified random sampling**. The material corresponds to Sections 3.1–3.5 of Lohr and Sections 5.1–5.9 of Scheaffer et al.

Example: stratifying U.S. counties by census region. In the SRS chapter we sampled about 10% of the 3078 U.S. counties to estimate the average farm acreage per county. Here we use the four U.S. census regions as strata. To compare stratified sampling directly with SRS, we again sample about 10% of the counties, this time 10% of the counties in *each* stratum:

Stratum	Counties in stratum (N_h)	Counties in sample (n_h)
Northeast	220	21
North Central	1054	103
South	1382	135
West	422	41
Total	**3078**	**300**

We select four separate, independent SRSs, one per stratum. Knowing which counties are in the sample from the Northeast tells us nothing about which counties are in the sample from the South. We analyze these data in Section 4.6.1.

4.2 Estimation with Stratified Samples

4.2.1 Population quantities

Let y_{hj} be the value of unit j in stratum h, for $j = 1, \dots, N_h$ and $h = 1, \dots, H$, so that $N = \sum_{h=1}^{H} N_h$. The **stratum weight**

$$\pi_h = \frac{N_h}{N} \tag{4.1}$$

is the proportion of population units in stratum h, so $\sum_{h=1}^{H} \pi_h = 1$. The population quantities of each stratum, and of the whole population, are

$$t_h = \sum_{j=1}^{N_h} y_{hj} \quad \text{(stratum total)}, \qquad t = \sum_{h=1}^{H} t_h \quad \text{(population total)}, \tag{4.2}$$

$$\overline{y}_{hU} = \frac{t_h}{N_h} \quad \text{(stratum mean)}, \qquad \overline{y}_U = \frac{t}{N} = \sum_{h=1}^{H} \pi_h \, \overline{y}_{hU} \quad \text{(overall mean)}, \tag{4.3}$$

$$S_h^2 = \frac{1}{N_h - 1} \sum_{j=1}^{N_h} (y_{hj} - \overline{y}_{hU})^2 \quad \text{(stratum variance)}. \tag{4.4}$$

The overall mean $\overline{y}_U$ is the average of the stratum means, weighted by the stratum weights.

4.2.2 SRS estimators within each stratum

Let $\mathcal{S}_h$ be the SRS of n_h units drawn from stratum h, with total sample size $n = \sum_{h=1}^{H} n_h$. Within stratum h we have an ordinary SRS, so we estimate the stratum quantities with the SRS formulas, applied separately to each stratum:

$$\overline{y}_h = \frac{1}{n_h} \sum_{j \in \mathcal{S}_h} y_{hj}, \qquad \hat{t}_h = \frac{N_h}{n_h} \sum_{j \in \mathcal{S}_h} y_{hj} = N_h \overline{y}_h, \qquad s_h^2 = \frac{1}{n_h - 1} \sum_{j \in \mathcal{S}_h} (y_{hj} - \overline{y}_h)^2. \tag{4.5}$$

4.2.3 Estimators for the population total and mean

Adding up the estimated stratum totals gives the stratified estimator of the population total. Dividing it by N gives the estimator of the population mean:

$$\boxed{\hat{t}_{\text{str}} = \sum_{h=1}^{H} \hat{t}_h = \sum_{h=1}^{H} N_h \overline{y}_h, \qquad \overline{y}_{\text{str}} = \frac{\hat{t}_{\text{str}}}{N} = \sum_{h=1}^{H} \pi_h \overline{y}_h.} \tag{4.6}$$

So $\overline{y}_{\text{str}}$ is a weighted average of the stratum sample means, with $\overline{y}_h$ weighted by $\pi_h = N_h/N$, the fraction of population units in stratum h. To use stratified sampling, the stratum sizes N_h must be known.

Unbiasedness. Since an SRS is taken in each stratum, $E[\overline{y}_h] = \overline{y}_{hU}$. Averaging these unbiased stratum estimates with the fixed weights N_h/N keeps the overall estimate unbiased:

$$E[\overline{y}_{\text{str}}] = \sum_{h=1}^{H} \frac{N_h}{N} E[\overline{y}_h] = \sum_{h=1}^{H} \frac{N_h}{N} \cdot \frac{1}{N_h} \sum_{j=1}^{N_h} y_{hj} = \frac{1}{N} \sum_{h=1}^{H} \sum_{j=1}^{N_h} y_{hj} = \overline{y}_U. \tag{4.7}$$

It follows that $E[\hat{t}_{\text{str}}] = N \overline{y}_U = t$.

4.2.4 Variances and standard errors

The strata are sampled independently, so the $\hat{t}_h$ are independent and their variances add. Each $\hat{t}_h$ has the SRS variance $N_h^2(1 - n_h/N_h)S_h^2/n_h$, so

$$V(\hat{t}_{\text{str}}) = \sum_{h=1}^{H} V(\hat{t}_h) = \sum_{h=1}^{H} \Big(1 - \frac{n_h}{N_h}\Big) N_h^2 \frac{S_h^2}{n_h}, \qquad V(\overline{y}_{\text{str}}) = \frac{V(\hat{t}_{\text{str}})}{N^2} = \sum_{h=1}^{H} \Big(1 - \frac{n_h}{N_h}\Big) \pi_h^2 \frac{S_h^2}{n_h}. \tag{4.8}$$

Only the variability *within* the strata enters these variances. The differences *between* the stratum means do not. This is the source of the gain from stratification: if the strata are internally homogeneous, every S_h^2 is small, even when the population as a whole is very variable.

Substituting the sample variances s_h^2 for S_h^2 gives unbiased variance estimators, and their square roots are the standard errors:

$$\boxed{\hat{V}(\overline{y}_{\text{str}}) = \sum_{h=1}^{H} v_h, \qquad v_h = \Big(1 - \frac{n_h}{N_h}\Big) \pi_h^2 \frac{s_h^2}{n_h}, \qquad \text{SE}(\overline{y}_{\text{str}}) = \sqrt{\hat{V}(\overline{y}_{\text{str}})},} \tag{4.9}$$

$$\hat{V}(\hat{t}_{\text{str}}) = N^2\,\hat{V}(\overline{y}_{\text{str}}) = \sum_{h=1}^{H}\Big(1-\frac{n_h}{N_h}\Big)N_h^2\frac{s_h^2}{n_h}, \qquad \text{SE}(\hat{t}_{\text{str}}) = N\,\text{SE}(\overline{y}_{\text{str}}). \tag{4.10}$$

Here v_h is the contribution of stratum h to the estimated variance. It appears as the last column of the working tables in this chapter. Each stratum needs at least two sampled units, $n_h \geq 2$, so that s_h^2 can be computed.

4.2.5 Confidence intervals

Suppose either the sample sizes within the strata are large, or the design has a large number of strata. Then $\overline{y}_{\text{str}}$ is approximately normal, and an approximate $100(1-\alpha)\%$ confidence interval for $\overline{y}_U$ is

$$\overline{y}_{\text{str}} \pm z_{\alpha/2}\,\text{SE}(\overline{y}_{\text{str}}). \tag{4.11}$$

When n is not large, it is common to replace the normal quantile by the quantile of a t distribution with $n-H$ degrees of freedom, for example $\hat{t}_{\text{str}} \pm t_{\alpha/2,\,n-H}\,\text{SE}(\hat{t}_{\text{str}})$ for the total. The functions used in this book use $z_{0.025} = 1.96$.

4.3 Example: The Nelchina Caribou Herd

Siniff and Skoog (1964) used stratified random sampling to estimate the size of the Nelchina herd of Alaska caribou in February 1962. Preliminary estimates of caribou density were used to divide the area of interest into six strata. Each stratum was divided into a grid of 4-mi^2 sampling units. Stratum A, for example, contained $N_1 = 400$ sampling units, of which $n_1 = 98$ were randomly selected for the survey. Figure 4.2 shows the layout.

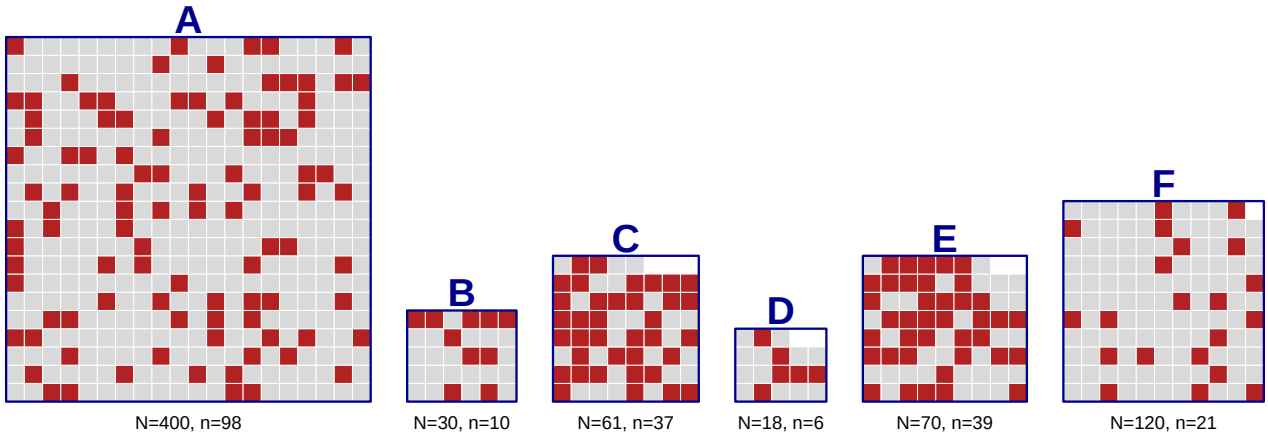

Figure 4.2: The six caribou strata. Each grid holds one stratum's N_h sampling units; the red cells are the n_h units selected at random in that stratum.

Spreadsheet calculation. The survey gives the sample mean $\overline{y}_h$ and the sample variance s_h^2 of the caribou counts in each stratum. With $N = 699$, the calculation fits in a spreadsheet. The column $\pi_h\overline{y}_h$ holds each stratum's contribution to $\overline{y}_{\text{str}} = \sum_h \pi_h\overline{y}_h$. The column $v_h = (1-\frac{n_h}{N_h})\pi_h^2\frac{s_h^2}{n_h}$ holds its contribution to $\hat{V}(\overline{y}_{\text{str}}) = \sum_h v_h$:

Stratum	N_h	n_h	$\pi_h = N_h/N$	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	v_h
A	400	98	0.5722	24.1	5,575	13.7911	14.0648
B	30	10	0.0429	25.6	4,064	1.0987	0.4991
C	61	37	0.0873	267.6	347,556	23.3528	28.1455
D	18	6	0.0258	179.0	22,798	4.6094	1.6798
E	70	39	0.1001	293.7	123,578	29.4120	14.0728
F	120	21	0.1717	33.2	9,795	5.6996	11.3409
Total	**699**	**211**	**1.0000**			**77.9636**	**69.8029**

Results. The estimated mean number of caribou per sampling unit, and its standard error, are

$$\bar{y}_{\text{str}} = 77.96, \qquad \text{SE}(\bar{y}_{\text{str}}) = \sqrt{69.8029} = 8.35. \tag{4.12}$$

Multiplying by $N = 699$ gives the estimated total number of caribou:

$$\hat{t}_{\text{str}} = 699(77.96) = 54{,}497, \qquad \text{SE}(\hat{t}_{\text{str}}) = 699(8.35) = 5{,}840, \tag{4.13}$$

and an approximate 95% confidence interval for the size of the herd is

$$54{,}497 \pm 1.96(5{,}840) = [43{,}051,\ 65{,}943]. \tag{4.14}$$

This interval reflects only sampling error. If the field procedure for counting caribou tends to miss animals, the whole interval will be biased low.

Strata C and E have far fewer units than stratum A, yet they contribute most of the estimate and much of its variance. These are the high-density strata, where the caribou counts are large and highly variable. We return to this point when we discuss optimal allocation in Section 4.8.

The same calculation in R. The function `str_est()` from the shared file `samplingestimate.r` (see Section 4.6) takes the stratum summaries $\bar{y}_h$, s_h, n_h and N_h and produces the same working table:

```
Nh <- c(A = 400, B = 30, C = 61, D = 18, E = 70, F = 120)
nh <- c(98, 10, 37, 6, 39, 21)
ybarh <- c(24.1, 25.6, 267.6, 179.0, 293.7, 33.2)
s2h <- c(5575, 4064, 347556, 22798, 123578, 9795)

res_caribou <- str_est(ybarh, sqrt(s2h), nh, Nh, estimate = "mean")
res_caribou$table
```

Stratified Mean Estimation: Working Table

Stratum	N_h	n_h	π_h [1]	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	v_h [2]
A	400	98	0.57	24.10	5,575.00	13.79	14.06
B	30	10	0.04	25.60	4,064.00	1.10	0.50
C	61	37	0.09	267.60	347,556.00	23.35	28.15
D	18	6	0.03	179.00	22,798.00	4.61	1.68
E	70	39	0.10	293.70	123,578.00	29.41	14.07
F	120	21	0.17	33.20	9,795.00	5.70	11.34
Total	**699**	**211**	**1.00**			**77.96**	**69.80**

[1] $\pi_h = N_h/N$ is the stratum weight.
[2] $v_h = (1 - n_h/N_h)\,\pi_h^2\, s_h^2/n_h$ is stratum h's contribution to $V(\bar{y}_{str}) = \sum_h v_h$.
$\bar{y}_{str} = \sum_h \pi_h \bar{y}_h = 77.96$
$\text{SE}(\bar{y}_{str}) = \sqrt{\sum_h v_h} = \sqrt{69.80} = 8.35$

```
str_est(ybarh, sqrt(s2h), nh, Nh, estimate = "total")$estimate |> format_est_gt("total")
```

$\hat{t}$	SE($\hat{t}$)	95% CI Lower	95% CI Upper
54,496.60	5,840.01	43,050.18	65,943.02

4.4 Stratified Sampling for Proportions

A proportion is the mean of a 0/1 variable. So we can apply the same formulas with $\bar{y}_h = \hat{p}_h$, the sample proportion in stratum h, and $s_h^2 = \frac{n_h}{n_h - 1}\hat{p}_h(1 - \hat{p}_h)$:

$$\hat{p}_{\text{str}} = \sum_{h=1}^{H} \frac{N_h}{N}\hat{p}_h, \qquad \hat{V}(\hat{p}_{\text{str}}) = \sum_{h=1}^{H} \left(1 - \frac{n_h}{N_h}\right)\left(\frac{N_h}{N}\right)^2 \frac{\hat{p}_h(1 - \hat{p}_h)}{n_h - 1}. \tag{4.15}$$

The total number of units with the attribute is estimated by $N\hat{p}_{\text{str}}$, with standard error $N\,\text{SE}(\hat{p}_{\text{str}})$.

Example: ACLS survey of scholars. The American Council of Learned Societies (ACLS) used a stratified random sample of the members of seven disciplinary societies. It studied publication patterns and the use of computers and libraries (Morton and Price, 1989). The membership figures N_h, the numbers of valid returns n_h, and the percentage of female respondents were:

Discipline	Membership (N_h)	Valid returns (n_h)	Female members (%)
Literature	9,100	636	38
Classics	1,950	451	27
Philosophy	5,500	481	18
History	10,850	611	19
Linguistics	2,100	493	36

Discipline	Membership (N_h)	Valid returns (n_h)	Female members (%)
Political Science	5,500	575	13
Sociology	9,000	588	26
Totals	**44,000**	**3,835**	

We estimate the proportion of these scholars who are female, ignoring nonresponse for now. With $N = 44{,}000$, the spreadsheet calculation is:

Discipline	N_h	n_h	$\pi_h = N_h/N$	$\hat{p}_h$	$\pi_h \hat{p}_h$	v_h ($\times 10^{-5}$)
Literature	9,100	636	0.2068	0.38	0.0786	1.476
Classics	1,950	451	0.0443	0.27	0.0120	0.066
Philosophy	5,500	481	0.1250	0.18	0.0225	0.439
History	10,850	611	0.2466	0.19	0.0469	1.448
Linguistics	2,100	493	0.0477	0.36	0.0172	0.082
Political Science	5,500	575	0.1250	0.13	0.0163	0.276
Sociology	9,000	588	0.2045	0.26	0.0532	1.282
Total	**44,000**	**3,835**	**1.0000**		**0.2465**	**5.068**

Here $v_h = (1 - \frac{n_h}{N_h})\pi_h^2 \frac{\hat{p}_h(1-\hat{p}_h)}{n_h - 1}$. Summing the columns gives

$$\hat{p}_{\text{str}} = \frac{9100}{44{,}000}(0.38) + \cdots + \frac{9000}{44{,}000}(0.26) = 0.2465, \qquad \text{SE}(\hat{p}_{\text{str}}) = \sqrt{5.068 \times 10^{-5}} = 0.0071. \tag{4.16}$$

About 24.7% of the members are estimated to be female. To estimate the total number of female scholars, multiply both numbers by $N = 44{,}000$. This gives about 10,846 women, with a standard error of about 313.

In R, we pass $s_h = \sqrt{n_h \hat{p}_h(1-\hat{p}_h)/(n_h - 1)}$ to `str_est()`:

```
Nh <- c(Literature = 9100, Classics = 1950, Philosophy = 5500, History = 10850,
        Linguistics = 2100, PoliSci = 5500, Sociology = 9000)
nh <- c(636, 451, 481, 611, 493, 575, 588)
ph <- c(0.38, 0.27, 0.18, 0.19, 0.36, 0.13, 0.26)

sh <- sqrt(nh / (nh - 1) * ph * (1 - ph))
round(str_est(ph, sh, nh, Nh, estimate = "mean", show.details = FALSE)$estimate, 4)
```

```
  Est.   S.E. ci.low ci.upp
0.2465 0.0071 0.2326 0.2605
```

4.5 Alternative Formulas for $\overline{y}_{\text{str}}$

The estimator $\overline{y}_{\text{str}} = \sum_h \pi_h \overline{y}_h$ can be written in two other ways. Each one shows something about the design. The first compares it with the sample mean of an SRS and shows why the weights must be N_h/N. The second writes it

as a weighted average over individual units, with each unit weighted by its sampling weight. This is the form used for survey data files.

4.5.1 Weighting by population fractions instead of sample fractions

Suppose instead that we had taken an SRS of size $n = n_1 + \cdots + n_H$ from the whole population. Grouping the sampled units by stratum, the sample mean is

$$\overline{y}_{\text{SRS}} = \frac{1}{n} \sum_{(h,j)\in\mathcal{S}} y_{hj} = \frac{n_1\overline{y}_1 + \cdots + n_H\overline{y}_H}{n} = \overline{y}_1 \frac{n_1}{n} + \cdots + \overline{y}_H \frac{n_H}{n}, \tag{4.17}$$

which we compare with the stratified estimator

$$\overline{y}_{\text{str}} = \overline{y}_1 \frac{N_1}{N} + \cdots + \overline{y}_H \frac{N_H}{N}. \tag{4.18}$$

For proportions, similarly, $\hat{p}_{\text{SRS}} = \sum_h \frac{n_h}{n}\hat{p}_h$ while $\hat{p}_{\text{str}} = \sum_h \frac{N_h}{N}\hat{p}_h$.

Both estimators average the stratum sample means. The difference is in the weights. In an SRS, the stratum sizes in the sample n_h are random, so $\overline{y}_{\text{SRS}}$ weights each stratum by its *realized* sample fraction n_h/n. That fraction varies from sample to sample and can be distorted by nonresponse rates that differ between strata. In contrast, $\overline{y}_{\text{str}}$ uses the *known, fixed* population fraction N_h/N. These fixed weights make it unbiased for $\overline{y}_U$ whatever the n_h are. When $n_h/n = N_h/N$ for every h (proportional allocation, Section 4.7), the two estimators coincide.

4.5.2 Weighting each unit by its sampling weight

Write the stratified sample as $\mathcal{S} = (\mathcal{S}_1, \ldots, \mathcal{S}_H)$, where $\mathcal{S}_h$ is the SRS drawn from stratum h. The strata are sampled independently, so the probability of selecting a particular sample is the product of the probabilities of its stratum samples:

$$P(\mathcal{S}) = P(\mathcal{S}_1) \times P(\mathcal{S}_2) \times \cdots \times P(\mathcal{S}_H) = \frac{1}{\binom{N_1}{n_1}} \cdot \frac{1}{\binom{N_2}{n_2}} \cdots \frac{1}{\binom{N_H}{n_H}}. \tag{4.19}$$

For SRS we defined the sampling weight of a unit as the reciprocal of its inclusion probability. A unit j in stratum h is included with probability n_h/N_h, the sampling fraction in stratum h. (Do not confuse this with the stratum weight $\pi_h = N_h/N$.) So its sampling weight is

$$w_{hj} = \frac{N_h}{n_h}. \tag{4.20}$$

Unless every stratum has the same sampling fraction, units in different strata have different inclusion probabilities. Stratified sampling is therefore an example of **unequal-probability sampling**.

The sampling weight can be thought of as the number of population units that a sampled unit represents. Within stratum h all n_h sampled units have the same weight N_h/n_h, so the weights add up to the stratum sizes and to the population size:

$$\sum_{j\in\mathcal{S}_h} w_{hj} = n_h \cdot \frac{N_h}{n_h} = N_h, \qquad \sum_{h=1}^{H} \sum_{j\in\mathcal{S}_h} w_{hj} = \sum_{h=1}^{H} N_h = N. \tag{4.21}$$

Estimators in terms of the weights. Since $N_h \overline{y}_h = \frac{N_h}{n_h} \sum_{j \in \mathcal{S}_h} y_{hj} = \sum_{j \in \mathcal{S}_h} w_{hj}\, y_{hj}$, the estimated total is a weighted sum over *all* sampled units, regardless of stratum. Since the weights sum to N, the estimated mean is a weighted average:

$$\hat{t}_{\text{str}} = \sum_{h=1}^{H} \sum_{j \in \mathcal{S}_h} w_{hj}\, y_{hj}, \qquad \overline{y}_{\text{str}} = \frac{\sum_{h=1}^{H} \sum_{j \in \mathcal{S}_h} w_{hj} y_{hj}}{\sum_{h=1}^{H} \sum_{j \in \mathcal{S}_h} w_{hj}}. \tag{4.22}$$

This is how survey data files are usually distributed: one row per sampled unit, with a column of sampling weights. The `weight` column of `agstrat.csv` in Section 4.6.1 is an example.

For the caribou survey the weights are:

Stratum	N_h	n_h	$w_{hj} = N_h/n_h$
A	400	98	4.08
B	30	10	3.00
C	61	37	1.65
D	18	6	3.00
E	70	39	1.79
F	120	21	5.71

The weights alone are enough for the point estimates, but not for the variances. Strata B and D have the same weight, so the weights do not identify which stratum a unit belongs to. The variance estimator needs s_h^2 computed separately within each stratum, so the data file must also record each unit's stratum.

4.6 Analyzing Stratified Data in R

The estimating functions used throughout this book live in a single shared file, `samplingestimate.r`, sourced at the start of this chapter. They include `str_est`, `str_est_data` and `srs_est`, which compute the stratified mean, its standard error and confidence intervals. `str_est` works from the stratum summaries ($\overline{y}_h$, s_h, n_h, N_h), as in the caribou example. `str_est_data` computes everything directly from a data set with stratum and weight columns.

The full source of `samplingestimate.r` is listed in Appendix C.

4.6.1 Example: Farm acreage by census region (agstrat.csv)

We now analyze the stratified sample of counties described in Section 4.1. The goal is to estimate the average 1992 farm acreage per county (`acres92`, in thousands of acres), with the census region (`region`) as the stratification variable.

```
agstrat <- acres_in_thousands(read.csv("data/agstrat.csv"))
head(agstrat) |> gt()
```

Each row is a sampled county. The column `region` is its stratum, and `weight` is its sampling weight N_h/n_h.

Table 4.1: Sample survey data for agstrat.csv, including stratum and weight

county	state	acres92	acres87	acres82	farms92	farms87	farms82	largef92	largef87	largef82	smallf92	smallf87	smallf82	region	rn	weight
PIERCE COUNTY	NE	297.326	332.862	319.619	725	857	865	54	54	42	58	67	48	NC	805	10.23301
JENNINGS COUNTY	IN	124.694	131.481	139.111	658	671	751	14	13	14	42	36	38	NC	241	10.23301
WAYNE COUNTY	OH	246.938	263.457	268.434	1582	1734	1866	20	19	16	175	186	184	NC	913	10.23301
VAN BUREN COUNTY	MI	206.781	190.251	197.055	1164	1278	1464	23	17	9	56	66	55	NC	478	10.23301
OZAUKEE COUNTY	WI	78.772	85.201	89.331	448	483	527	6	5	5	56	49	48	NC	1028	10.23301
CLEARWATER COUNTY	MN	210.897	229.537	213.105	583	699	693	34	32	23	8	19	13	NC	496	10.23301

Table 4.2: Sample summary statistics for acres92 by region stratum

	N_h	n_h	$\bar{y}_h$	s_h
NC	1054	103	300.50	172.10
NE	220	21	97.63	87.45
S	1382	135	211.32	231.49
W	422	41	662.30	629.43

Summarizing acres92 in each stratum

We first compute the sample size (n_h), standard deviation (s_h) and mean ($\bar{y}_h$) for each region. These are matched with the known numbers of counties in each region (N_h).

```
nh <- tapply (agstrat[, "acres92"], agstrat[,"region"], length)
sh <- tapply (agstrat[, "acres92"], agstrat[,"region"], sd)
ybarh <- tapply (agstrat[, "acres92"], agstrat[,"region"], mean)
# create a vector with external information (true population size per stratum)
Nh <- c(NC = 1054, NE = 220, S= 1382, W = 422)

data.frame(Nh, nh, ybarh, sh) |>
  gt(rownames_to_stub = TRUE) |>
  cols_label(
    Nh = md("$N_h$"),
    nh = md("$n_h$"),
    ybarh = md("$\\bar{y}_h$"),
    sh = md("$s_h$")
  ) |>
  fmt_number(columns = c(ybarh, sh), decimals = 2)
```

The standard deviation in the W region is much larger than in the other regions.

Estimates from the stratum summaries

The `$table` of `str_est()` is the spreadsheet working table: it shows N_h, n_h, π_h, $\bar{y}_h$, s_h^2, $\pi_h \bar{y}_h$ and v_h for each stratum, with a Total row.

```
res_str_mean <- str_est(ybarh, sh, nh, Nh, estimate = "mean")
res_str_mean$table
```

Stratified Mean Estimation: Working Table

Stratum	N_h	n_h	π_h [1]	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	v_h [2]
NC	1,054	103	0.34	300.50	29,618.18	102.90	30.42
NE	220	21	0.07	97.63	7,647.47	6.98	1.68
S	1,382	135	0.45	211.32	53,587.49	94.88	72.20
W	422	41	0.14	662.30	396,185.95	90.80	163.99
Total	**3,078**	**300**	**1.00**			**295.56**	**268.30**

[1] $\pi_h = N_h/N$ is the stratum weight.
[2] $v_h = (1 - n_h/N_h)\,\pi_h^2\,s_h^2/n_h$ is stratum h's contribution to $V(\bar{y}_{str}) = \sum_h v_h$.
$\bar{y}_{str} = \sum_h \pi_h \bar{y}_h = 295.56$
$\text{SE}(\bar{y}_{str}) = \sqrt{\sum_h v_h} = \sqrt{268.30} = 16.38$

The table shows how each stratum contributes to the mean and to its variance. The `$estimate` component gives the overall result:

```
res_str_mean$estimate |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
295.56	16.38	263.46	327.67

The estimated mean is 295.6 thousand acres per county. The 95% confidence interval is [263.5, 327.7] thousand acres.

To estimate the population total, the total farm acreage of all U.S. counties, set `estimate = "total"`:

```
## population total estimate
res_str_total <- str_est(ybarh, sh, nh, Nh, estimate = "total")
res_str_total$estimate |> format_est_gt("total")
```

$\hat{t}$	SE($\hat{t}$)	95% CI Lower	95% CI Upper
909,736.04	50,417.25	810,918.23	1,008,553.84

The estimated total is 909,736 thousand acres. Because the strata are sampled independently, its variance is the sum of the four estimated stratum variances $N_h^2(1 - n_h/N_h)s_h^2/n_h$.

Estimates directly from the data

Instead of pre-computing the stratum summaries, `str_est_data()` works directly from the data file, using the stratum and weight columns. Summing the weights within each region recovers the stratum sizes N_h:

```
tapply(agstrat$weight, agstrat$region, sum)
```

```
  NC   NE    S    W
1054  220 1382  422
```

```
## the dataset contains a variable "weight"
res_agstrat_acres92 <- str_est_data(agstrat, y="acres92", stratum="region", weight="weight",
 ↪  estimate = "mean")
res_agstrat_acres92$table
```

Stratified Mean Estimation: Working Table

Stratum	N_h	n_h	π_h [1]	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	v_h [2]
NC	1,054	103	0.34	300.50	29,618.18	102.90	30.42
NE	220	21	0.07	97.63	7,647.47	6.98	1.68
S	1,382	135	0.45	211.32	53,587.49	94.88	72.20
W	422	41	0.14	662.30	396,185.95	90.80	163.99
Total	**3,078**	**300**	**1.00**			**295.56**	**268.30**

[1] $\pi_h = N_h/N$ is the stratum weight.
[2] $v_h = (1 - n_h/N_h)\,\pi_h^2\, s_h^2/n_h$ is stratum h's contribution to $V(\bar{y}_{str}) = \sum_h v_h$.
$\bar{y}_{str} = \sum_h \pi_h \bar{y}_h = 295.56$
$\text{SE}(\bar{y}_{str}) = \sqrt{\sum_h v_h} = \sqrt{268.30} = 16.38$

```
res_agstrat_acres92$estimate |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
295.56	16.38	263.46	327.67

The result matches the calculation from the stratum summaries. Estimating a different variable only requires changing `y`. For example, the average number of small farms per county in 1992 is estimated as 56.9:

```
res_agstrat_smallf92 <- str_est_data(agstrat, y="smallf92", stratum="region", weight="weight",
 ↪  estimate = "mean")
res_agstrat_smallf92$table
```

Stratified Mean Estimation: Working Table

Stratum	N_h	n_h	π_h [1]	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	v_h [2]
NC	1,054	103	0.34	44.26	1,286.43	15.16	1.32
NE	220	21	0.07	47.24	2,364.79	3.38	0.52
S	1,382	135	0.45	47.39	6,205.45	21.28	8.36
W	422	41	0.14	124.39	100,640.94	17.05	41.66
Total	**3,078**	**300**	**1.00**			**56.86**	**51.86**

[1] $\pi_h = N_h / N$ is the stratum weight.
[2] $v_h = (1 - n_h/N_h)\, \pi_h^2\, s_h^2 / n_h$ is stratum h's contribution to $V(\bar{y}_{str}) = \sum_h v_h$.
$\bar{y}_{str} = \sum_h \pi_h \bar{y}_h = 56.86$
$\text{SE}(\bar{y}_{str}) = \sqrt{\sum_h v_h} = \sqrt{51.86} = 7.20$

```
res_agstrat_smallf92$estimate |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
56.86	7.20	42.75	70.98

Comparing with the SRS estimate

The SRS chapter analyzed an SRS of 300 counties from the same population (`agsrs.csv`). How much did stratification gain?

```
agsrs <- acres_in_thousands(read.csv("data/agsrs.csv"))
res_agsrs_srs <- srs_est(agsrs[, "acres92"], N=3078, estimate = "mean", show.details = FALSE)
res_agsrs_srs$estimate |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
297.90	18.90	260.71	335.09

The SRS standard error is 18.9 thousand acres, compared with 16.4 for the stratified sample. Only the West had a sample variance larger than the overall SRS sample variance. The sample variance in the Northeast was much smaller than that of the population as a whole. Observations within a stratum tend to be more homogeneous than observations in the population overall, and this is what stratification exploits. We can estimate the relative gain by the ratio of the estimated variances:

```
## ratio of estimated variances (stratified / SRS), both with n = 300
ratio_var <- (res_agstrat_acres92$estimate["S.E."] / res_agsrs_srs$estimate["S.E."])^2
unname(ratio_var)
```

```
[1] 0.7512239
```

The stratified variance is about 75.1% of the SRS variance, a reduction of 24.9% at no extra sampling cost. Suppose these were the population variances. Then a stratified sample of only $300 \times 0.75 \approx 225$ counties would match the precision of an SRS of 300. These are estimates from two particular samples. Section 4.10 repeats the comparison over thousands of samples.

4.7 Efficiency of Proportional Allocation

So far the stratum sample sizes n_h were given. How should a total sample of size n be divided among the strata? We first study a simple benchmark, **proportional allocation**, which uses the same sampling fraction in every stratum:

$$\frac{n_h}{N_h} = \frac{n}{N} \quad \text{for all } h, \qquad \text{that is,} \qquad n_h = n\,\frac{N_h}{N} = n\pi_h. \tag{4.23}$$

Then every unit has the same inclusion probability n/N and the same sampling weight N/n, so the sample is **self-weighting**, and $\overline{y}_{\text{str}}$ equals the plain sample mean. The county sample in Section 4.1, with about 10% of each region, is an example. Proportional allocation is not optimal in general (Section 4.8), but it lets us see clearly when, and by how much, stratification beats SRS.

4.7.1 Decomposition of the population variance

The answer depends on how the population variance splits into a part *between* the strata and a part *within* them. This is the decomposition of one-way analysis of variance:

Source	df	Sum of squares
Between strata	$H-1$	$\text{SSB} = \sum_{h=1}^{H} N_h(\overline{y}_{hU} - \overline{y}_U)^2$
Within strata	$N-H$	$\text{SSW} = \sum_{h=1}^{H}(N_h-1)S_h^2 = \sum_{h=1}^{H}\sum_{j=1}^{N_h}(y_{hj}-\overline{y}_{hU})^2$
Total, about $\overline{y}_U$	$N-1$	$\text{SSTO} = (N-1)S^2 = \sum_{h=1}^{H}\sum_{j=1}^{N_h}(y_{hj}-\overline{y}_U)^2$

$$\text{SSTO} = \text{SSB} + \text{SSW}. \tag{4.24}$$

One way to see the decomposition is to view each sum of squares as the squared length of a vector of prediction errors. Stack all N values into a vector y, and consider two ways of predicting them:

	stratum 1			$\cdots$	stratum H		
$y^\top$	y_{11}	$\cdots$	y_{1N_1}	$\cdots$	y_{H1}	$\cdots$	y_{HN_H}
$\widehat{y}_0^\top$	$\overline{y}_U$	$\cdots$	$\overline{y}_U$	$\cdots$	$\overline{y}_U$	$\cdots$	$\overline{y}_U$
$\widehat{y}_1^\top$	$\overline{y}_{1U}$	$\cdots$	$\overline{y}_{1U}$	$\cdots$	$\overline{y}_{HU}$	$\cdots$	$\overline{y}_{HU}$

(4.25)

The prediction $\hat{y}_0$ ignores the strata and predicts every unit by the overall mean. The prediction $\hat{y}_1$ predicts each unit by its stratum mean. Then

$$\text{SSTO} = \|y - \hat{y}_0\|^2, \qquad \text{SSW} = \|y - \hat{y}_1\|^2, \qquad \text{SSB} = \|\hat{y}_1 - \hat{y}_0\|^2. \tag{4.26}$$

Since $y - \hat{y}_0 = (y - \hat{y}_1) + (\hat{y}_1 - \hat{y}_0)$, and the two pieces are orthogonal, the Pythagorean theorem gives SSTO = SSW + SSB. The pieces are orthogonal because, within each stratum, the deviations $y_{hj} - \overline{y}_{hU}$ sum to zero while $\overline{y}_{hU} - \overline{y}_U$ is constant. Figure 4.3 illustrates the two predictions.

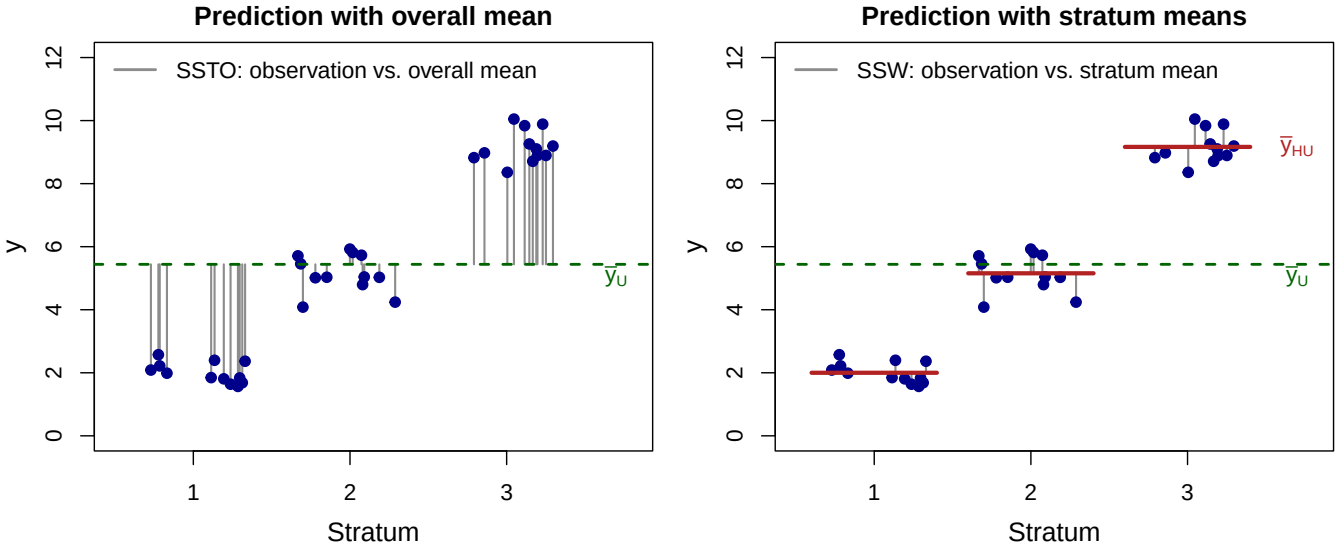

Figure 4.3: Left: predicting every unit by the overall mean (green dashed line); the squared errors (grey) sum to SSTO. Right: predicting each unit by its stratum mean (red segments); the squared errors sum to SSW. The difference SSTO − SSW is SSB.

4.7.2 The coefficient of determination R^2

The share of the total variation that is between strata is the **coefficient of determination**

$$R^2 = \frac{\text{SSB}}{\text{SSTO}} = 1 - \frac{\text{SSW}}{\text{SSTO}}. \tag{4.27}$$

R^2 generalizes the squared Pearson correlation coefficient. It measures how strongly a response y is related to one or more predictors. Here the predictor is "stratum membership", and R^2 is the proportion of the variance in y explained by the strata. It is the R^2 of a regression of y on the stratum indicators, as in the ANOVA fits of Section 4.10. When the stratum means differ widely relative to the spread within each stratum, R^2 is large (Figure 4.4, left). When the stratum means are nearly equal, R^2 is small (right).

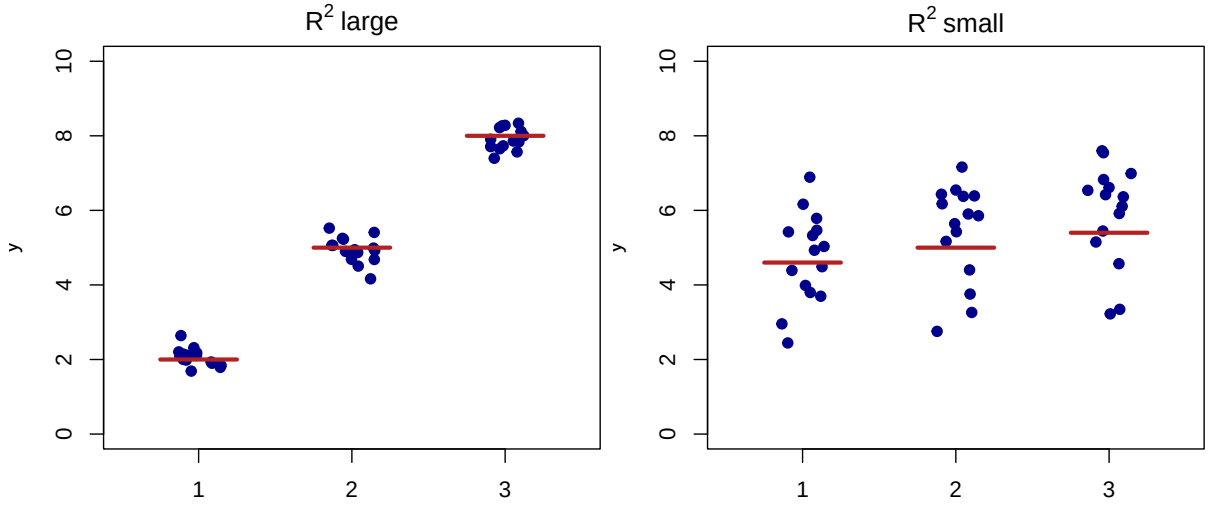

Figure 4.4: Strata with widely separated means (left, R^2 large) and with nearly equal means (right, R^2 small). Red segments are the stratum means.

4.7.3 Comparing the variances of proportional allocation and SRS

Under proportional allocation, $1-n_h/N_h = 1-n/N$ and $N_h^2/n_h = (N/n)N_h$ for every stratum. So the variance of $\hat{t}_{\text{str}}$ becomes

$$V_{\text{prop}}(\hat{t}_{\text{str}}) = \left(1-\frac{n}{N}\right)\frac{N}{n}\sum_{h=1}^{H} N_h S_h^2 = \left(1-\frac{n}{N}\right)\frac{N}{n}\left(\text{SSW} + \sum_{h=1}^{H} S_h^2\right), \tag{4.28}$$

using $\sum_h N_h S_h^2 = \sum_h (N_h-1)S_h^2 + \sum_h S_h^2$. Equivalently, for the mean,

$$\boxed{V_{\text{prop}}(\bar{y}_{\text{str}}) = \left(1-\frac{n}{N}\right)\frac{1}{n}\sum_{h=1}^{H}\pi_h S_h^2}, \tag{4.29}$$

a weighted average of the within-stratum variances in place of the overall S^2. For an SRS of the same size n, since $(N-1)S^2 = \text{SSTO} = \text{SSW} + \text{SSB}$,

$$V_{\text{SRS}}(\hat{t}) = \left(1-\frac{n}{N}\right)\frac{N^2 S^2}{n} = \left(1-\frac{n}{N}\right)\frac{N}{n}\cdot\frac{N}{N-1}\,(\text{SSW}+\text{SSB}). \tag{4.30}$$

When does stratification help? Comparing the two expressions, $V_{\text{prop}} < V_{\text{SRS}}$ exactly when $(N-1)(\text{SSW} + \sum_h S_h^2) < N(\text{SSW}+\text{SSB})$. Substituting $\text{SSW} = \sum_h (N_h-1)S_h^2$ and simplifying, the condition becomes

$$\text{SSB} > \sum_{h=1}^{H}\left(1-\frac{N_h}{N}\right)S_h^2. \tag{4.31}$$

This almost always holds. SSB is a sum of N squared deviations, one for each unit in the population. The right side has only H terms, typically far fewer. Proportional allocation can lose to SRS only when the stratum means are

almost identical, and even then the loss is negligible. In practice stratification with proportional allocation is never worse than SRS, up to terms of order $1/N_h$.

How much does it help? Dropping the small terms $\sum_h S_h^2$ and $N/(N-1)$, the ratio of the variances is

$$\frac{V_{\text{prop}}(\hat{t}_{\text{str}})}{V_{\text{SRS}}(\hat{t})} \approx \frac{\text{SSW} + \sum_h S_h^2}{\text{SSW} + \text{SSB}} \approx \frac{\text{SSW}}{\text{SSTO}} = 1 - R^2, \tag{4.32}$$

so that

$$\boxed{\text{percentage of variance reduction} \approx \frac{\text{SSB}}{\text{SSTO}} = R^2_{y|\text{strata}}}, \tag{4.33}$$

the proportion of the variance of y explained by stratum membership. The better the strata separate the values of y, the larger the gain. We verify this relation by simulation in Section 4.10.

4.8 Optimal Allocation

4.8.1 The cost function

The aim of optimal allocation is to gain the most information for the least cost. Let C be the total cost of the survey, c_0 the overhead cost, and c_h the cost of sampling one unit in stratum h:

$$C = c_0 + \sum_{h=1}^{H} c_h n_h. \tag{4.34}$$

We want to choose $n_1, \dots, n_H$ to minimize $V(\overline{y}_{\text{str}})$ for a fixed total cost C, or, equivalently, to minimize C for a fixed variance.

4.8.2 Derivation

Expanding the fpc, the variance separates into a part that depends on the n_h and a part that does not:

$$V(\overline{y}_{\text{str}}) = \sum_{h=1}^{H} \Big(1 - \frac{n_h}{N_h}\Big) \frac{N_h^2}{N^2} \frac{S_h^2}{n_h} = \sum_{h=1}^{H} \frac{N_h^2 S_h^2}{N^2} \cdot \frac{1}{n_h} \; - \; \underbrace{\sum_{h=1}^{H} \frac{N_h S_h^2}{N^2}}_{\text{free of } n_h}. \tag{4.35}$$

We minimize the first sum subject to $c_0 + \sum_h c_h n_h = C$, using the Lagrangian

$$L(n_1, \dots, n_H, \lambda) = \sum_{h=1}^{H} \frac{N_h^2 S_h^2}{N^2 n_h} + \lambda\Big(c_0 + \sum_{h=1}^{H} c_h n_h - C\Big). \tag{4.36}$$

Setting $\partial L / \partial n_h = 0$ gives

$$-\frac{N_h^2 S_h^2}{N^2 n_h^2} + \lambda c_h = 0 \quad \Longrightarrow \quad n_h = \frac{1}{N\sqrt{\lambda}} \cdot \frac{N_h S_h}{\sqrt{c_h}}. \tag{4.37}$$

Summing over h gives $n = \frac{1}{N\sqrt{\lambda}} \sum_l N_l S_l / \sqrt{c_l}$. Dividing n_h by n eliminates λ.

4.8.3 The optimal allocation

The optimal n_h is proportional to $N_h S_h / \sqrt{c_h}$:

$$\boxed{n_h = \left(\frac{N_h S_h / \sqrt{c_h}}{\sum_{l=1}^{H} N_l S_l / \sqrt{c_l}} \right) n, \qquad n_h \propto \frac{N_h S_h}{\sqrt{c_h}}.} \tag{4.38}$$

The total n itself follows from the budget, or from a target variance. The formula says to sample more heavily in a stratum when:

- the stratum is a large part of the population (N_h large);
- the stratum is more variable (S_h large); or
- sampling in the stratum is cheap (c_h small).

Two special cases are important. If the costs are equal, $c_1 = \cdots = c_H$, then $n_h \propto N_h S_h$. This is **Neyman allocation**:

$$\boxed{n_h = n \frac{N_h S_h}{\sum_{l=1}^{H} N_l S_l}}, \tag{4.39}$$

which gives the smallest possible variance for a total sample size n:

$$\boxed{V_{\text{opt}}(\overline{y}_{\text{str}}) = \frac{1}{n} \left(\sum_{h=1}^{H} \pi_h S_h \right)^2 - \frac{1}{N} \sum_{h=1}^{H} \pi_h S_h^2}. \tag{4.40}$$

If, in addition, the stratum standard deviations are equal, $S_1 = \cdots = S_H$, then $n_h \propto N_h$, which is proportional allocation. Neyman allocation improves on proportional allocation exactly when the S_h differ across strata.

In practice the S_h are unknown when the survey is designed. They are replaced by estimates from a previous survey or a pilot sample, or by rough guesses. The allocation is fairly robust to such guesses: moderately wrong values of S_h still give a variance close to the optimum.

4.8.4 Example: Neyman allocation for the caribou survey

To plan the caribou survey of Section 4.3, the investigators used rough estimates of the caribou counts as a proxy for S_h, with a total sample size of $n = 225$:

Stratum	N_h	s_h	$N_h s_h$	n_h (Neyman)	Sample size used
A	400	3,000	1,200,000	96.26	98
B	30	2,000	60,000	4.81	10
C	61	9,000	549,000	44.04	37
D	18	2,000	36,000	2.89	6
E	70	12,000	840,000	67.38	39
F	120	1,000	120,000	9.63	21
Total	**699**		**2,805,000**	**225**	**211**

For example, stratum E gets $225 \times 840{,}000/2{,}805{,}000 = 67.38$ units. It has only 10% of the population units, but it was expected to be the most variable. The investigators wanted the sampling fraction to be at least $1/3$ in the smaller strata, so they used the Neyman values only as a guideline. The small strata B, D and F received more units than Neyman allocation suggests, and stratum E received fewer.

4.8.5 Example: Seal holes (seals.csv)

This dataset comes from a survey of ringed seal breathing holes in three zones. The researchers used proportional allocation, sampling 20% of the areas in each zone.

```
## load data
seals <- read.csv("data/seals.csv")

# survey data summary in each stratum
nh <- as.vector(table(seals[,"zone"]))
sh <- tapply(seals[, "holes"], seals[,"zone"], sd)
ybarh <- tapply(seals[, "holes"], seals[,"zone"], mean)
Nh <- c(68, 84, 48)

res_seals <- str_est(ybarh, sh, nh, Nh, estimate = "mean")
res_seals$table
```

Stratified Mean Estimation: Working Table

Stratum	N_h	n_h	π_h [1]	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	v_h [2]
1	68	17	0.34	1.76	3.32	0.60	0.02
2	84	12	0.42	4.42	11.54	1.85	0.15
3	48	11	0.24	10.55	46.07	2.53	0.19
Total	**200**	**40**	**1.00**			**4.99**	**0.35**

[1] $\pi_h = N_h/N$ is the stratum weight.
[2] $v_h = (1 - n_h/N_h)\,\pi_h^2\, s_h^2/n_h$ is stratum h's contribution to $V(\bar{y}_{str}) = \sum_h v_h$.
$\bar{y}_{str} = \sum_h \pi_h \bar{y}_h = 4.99$
$\text{SE}(\bar{y}_{str}) = \sqrt{\sum_h v_h} = \sqrt{0.35} = 0.59$

Zone 3 has the largest standard deviation ($s_h \approx 6.8$), so seal density varies most there.

```
res_seals$estimate |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
4.99	0.59	3.83	6.14

The estimated mean number of seal holes per area is 5.

Table 4.3: Neyman-optimal allocation proportions for the seals survey zones

	C_h	N_h	S_h	$N_h S_h/\sqrt{C_h}$	L_h (Prop)
1	1	68	1.821	123.831	0.168
2	1	84	3.397	285.327	0.388
3	1	48	6.788	325.809	0.443

Zone 3 was the most variable, so sampling it more heavily would have lowered the variance. Treating the sample s_h as estimates of S_h, and assuming equal costs, we can compute the allocation proportions $L_h = n_h/n$ that a future survey should use:

```
## assuming the cost of surveying each zone is the same
Ch <- rep(1, 3)
Sh <- sh ## estimate population Sh with sample sh
NhShDCh <- Nh * Sh / sqrt(Ch)

## the optimal allocation proportion (Lh) for estimating the population total or mean:
Lh <- NhShDCh / sum(NhShDCh)

data.frame(Ch, Nh, Sh, NhShDCh, Lh) |>
  gt(rownames_to_stub = TRUE) |>
  cols_label(NhShDCh = md("$N_h S_h / \\sqrt{C_h}$"), Lh = md("$L_h$ (Prop)"), Ch=md("$C_h$"),
   ↪  Sh=md("$S_h$"), Nh=md("$N_h$")) |>
  fmt_number(columns = c(Sh, NhShDCh, Lh), decimals = 3)
```

Zone 2 has more areas ($N_h = 84$), but Zone 3's high variability means it should receive the largest share of the sampling effort, about 44%. Under proportional allocation it received only $48/200 = 24\%$.

4.8.6 Example: Optimal allocation for polls

For estimating a proportion, $S_h^2 \approx p_h(1-p_h)$. As a function of p, $p(1-p)$ is a quadratic maximized at $p = 1/2$ (Figure 4.5). Election regions where support is near 50%, the "swing states", contribute the most variance per unit sampled. Optimal allocation therefore gives them relatively larger samples. The curve is flat near its maximum: $p(1-p)$ is 0.21 at $p = 0.3$ and still 0.25 at $p = 0.5$. So Neyman allocation for proportions differs much from proportional allocation only when some p_h are close to 0 or 1.

4.8.7 Example: Dollar stratification

Dollar stratification is common in accounting audits. The recorded book amounts are used to stratify the population. For example, stratum 1 might be all loans over \$1 million, stratum 2 the loans between \$500,000 and \$999,999, and so on, down to loans under \$10,000. Optimal allocation is very efficient here. S_h is much larger in the strata with large loan amounts, so optimal allocation gives those strata a much higher sampling fraction. Often the top stratum is sampled completely ($n_h = N_h$). This makes sense: an error in a \$3,000,000 loan is likely to contribute far more to the total audited discrepancy than an error in a \$3,000 loan.

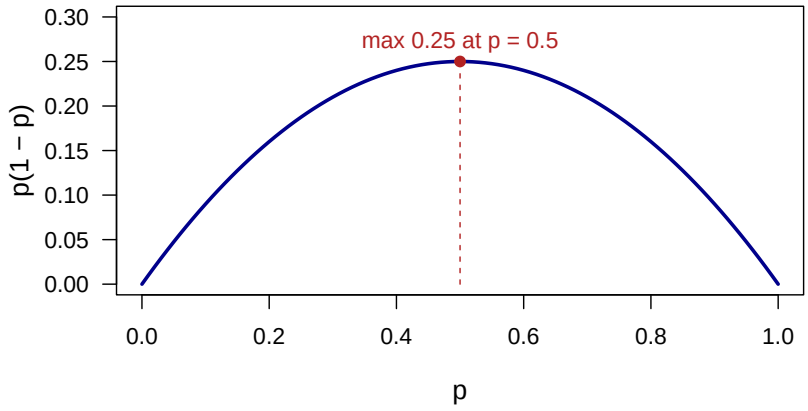

Figure 4.5: The variance of a 0/1 variable, $p(1-p)$, is largest at $p = 0.5$.

4.9 Choosing Stratification Variables

Stratification is most efficient when the stratum means differ widely. Then SSB is large and the variability within the strata is small. Ideally we would stratify by the values of y itself. But y is unknown before sampling, so we look instead for an **auxiliary variable** x that is known for every unit in the population and is highly correlated with y. Cutting the range of x into intervals then produces strata whose distributions of y differ (Figure 4.6).

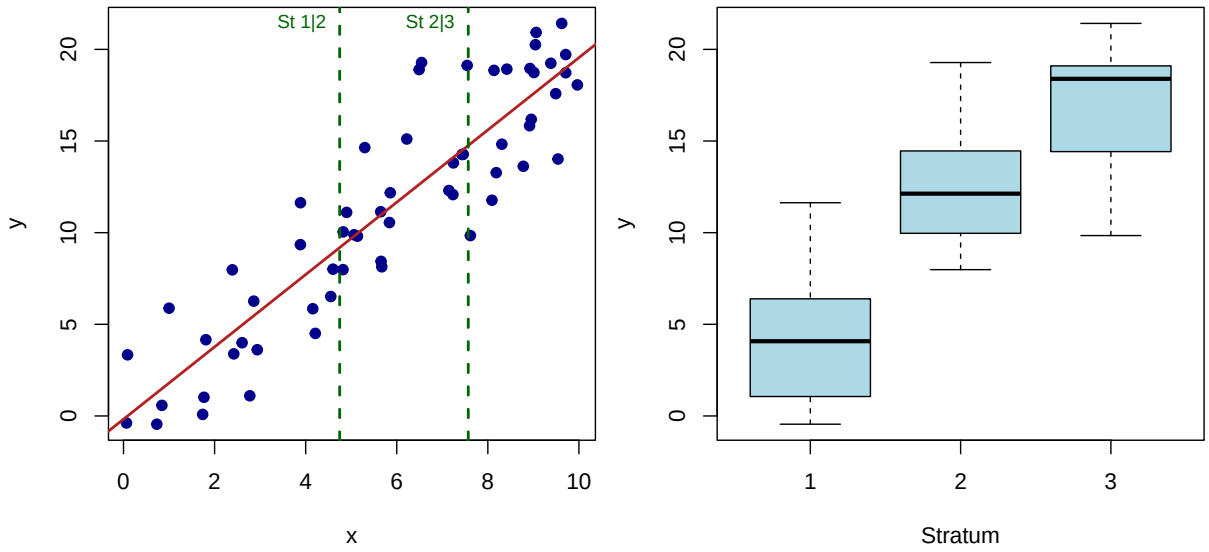

Figure 4.6: Cutting an auxiliary variable x at its terciles (left, dashed lines) produces strata with visibly different distributions of y (right).

Some examples:

- To estimate total business expenditure on advertising, we would like the businesses that spend the most on advertising in stratum 1, the next highest spenders in stratum 2, and so on. We might stratify by the number of employees or the size of the business, and by the type of product or service.

- For farm income, we might use farm size as the stratification variable, since we expect larger farms to have higher incomes.
- For the county farm data, the acreage in an earlier census (`acres82`) is known for every county and is much more strongly related to `acres92` than the region is. Section 4.10 shows how much this matters.

Stratified sampling also has limitations:

- **It requires information before sampling.** We need the stratum membership of every unit and the stratum sizes N_h. For optimal allocation we also need rough values of S_h.
- **One survey usually measures many variables.** A stratification variable that works well for one y may work poorly for another. A variable highly correlated with one outcome may be nearly uncorrelated with another outcome measured in the same survey. Proportional allocation is a safe compromise, since it never does materially worse than SRS for any variable. An allocation that is optimal for one variable can be inefficient for others.

4.10 Simulation: Efficiency of Stratification and Allocation

We now check the theory by simulation. The full `agpop.csv` dataset serves as the population. We draw 2000 repeated samples of $n = 300$ counties under three designs, SRS, proportional allocation (P2S) and Neyman allocation, and compare the spread of the resulting estimates of $\overline{y}_U$. We do this twice, with two different stratification variables.

4.10.1 Stratifying by region

```
agpop <- acres_in_thousands(read.csv("data/agpop.csv"))
agpop <- agpop[!is.na(agpop$acres92), ] ## remove those counties with na
N <- nrow(agpop)
```

```
agpop$stratum <- agpop$region
## reorder agpop for the ease of using strata of "sampling" (very important)
agpop <- agpop[order(agpop$stratum), ]
```

Section 4.7 showed that the variance reduction under proportional allocation is about R^2, the proportion of the variance of y explained by the strata. We compute it with a one-way ANOVA of `acres92` on the strata.

```
boxplot(acres92 ~ stratum, data = agpop)
```

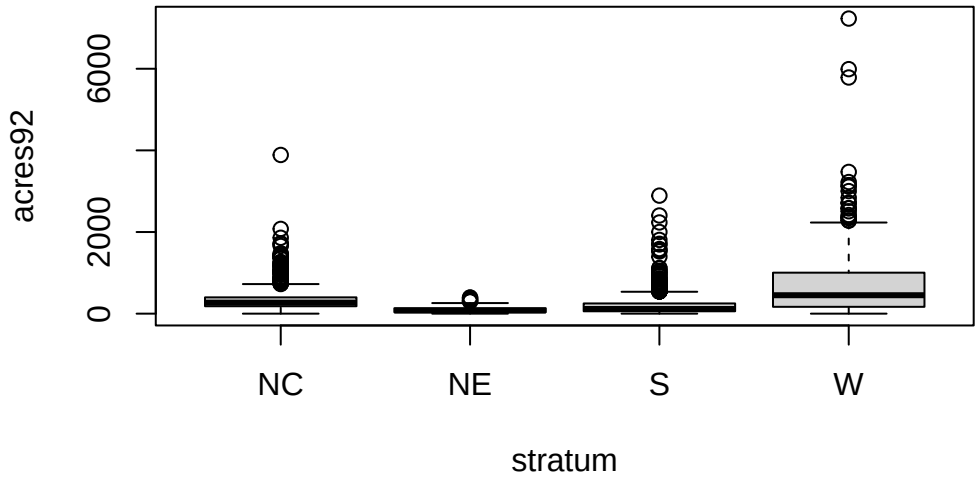

Figure 4.7: 1992 farm acreage by region stratum

```
lm_region <- lm(agpop$acres92 ~ agpop$stratum)
anova(lm_region)
```

```
Analysis of Variance Table

Response: agpop$acres92
               Df    Sum Sq  Mean Sq F value    Pr(>F)
agpop$stratum   3 100733006 33577669  226.73 < 2.2e-16 ***
Residuals    3055 452431724   148095
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
lm_region_gt <- format_lm_gt(lm_region)
lm_region_gt$fit
```

Model Fit Summary

R^2	Adj. R^2	$\hat{\sigma}$	F-statistic	p-value
0.1821	0.1813	384.8318	226.7299	0.0000

```
r_squared_region <- summary(lm_region)$r.squared
```

The R-squared is 0.182. Region explains only about 18.2% of the variance in acreage. That is helpful, but not strong.

A single stratified sample with proportional allocation

The function `strata()` from the **sampling** package draws an SRS of the given size in each stratum. The data must be sorted by stratum. The sampling weights are the reciprocals of the inclusion probabilities it returns.

```
# doing one stratified sampling
Nh <- tapply(1:nrow(agpop), agpop$stratum, length)
nh <- round(Nh/sum(Nh)*300)

## doing stratified sampling
strsample <- strata(agpop, "stratum", size = nh, method = "srswor")

# collecting data on sampled counties
agstrat <- agpop[strsample$ID_unit, ]
agstrat$weight <- 1/strsample$Prob

str_est_data(agstrat, "acres92", "stratum", "weight", estimate = "mean")$estimate |>
  format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
328.10	18.65	291.55	364.65

Repeating proportional allocation 2000 times

```
Nh <- tapply(1:nrow(agpop), agpop$stratum, length)
nh <- round(Nh/sum(Nh)*300) ## make sure the order matches Nh

nres <- 2000
str_p2s_simulated <- matrix(0, nres, 4)
for (i in 1:nres) {
    strsample <- strata(agpop, "stratum", size = nh, method = "srswor")
    agstrat <- agpop[strsample$ID_unit, ]
    agstrat$weight <- 1/strsample$Prob
    str_p2s_simulated[i,] <- str_est_data(agstrat, "acres92", "stratum", "weight", estimate =
↪   "mean", show.details = FALSE)$estimate
}
```

Repeating Neyman allocation 2000 times

In a simulation the population S_h are known, so we can use the exact Neyman allocation:

```
Nh <- tapply(agpop$acres92, agpop$stratum, length)
Sh <- tapply(agpop$acres92, agpop$stratum, sd)
nh_opt <- round((Nh*Sh)/sum(Nh*Sh) * 300)

data.frame(Nh, Sh, nh_opt) |>
  gt(rownames_to_stub = TRUE) |>
  fmt_number(columns = Sh, decimals = 2)
```

Table 4.4: Population sizes and Neyman-optimal sample sizes by region stratum

	Nh	Sh	nh_opt
NC	1052	271.19	87
NE	213	78.91	5
S	1376	244.13	102
W	418	836.61	106

The West, with the largest S_h, receives a disproportionately large sample under Neyman allocation.

```
nres <- 2000
str_neyman_simulated <- matrix(0, nres, 4)
for (i in 1:nres) {
    strsample <- strata(agpop, "stratum", size = nh_opt, method = "srswor")
    agstrat <- agpop[strsample$ID_unit, ]
    agstrat$weight <- 1/strsample$Prob
    str_neyman_simulated[i,] <- str_est_data(agstrat, "acres92", "stratum", "weight", estimate
 ↪  = "mean", show.details = FALSE)$estimate
}
```

Repeating SRS 2000 times

```
nres <- 2000
srs_simulated <- matrix(0, nres, 4)
for (i in 1:nres) {
    srs <- sample(sum(Nh), sum(nh))
    agsrs <- agpop[srs, ]
    srs_simulated[i,] <- srs_est(agsrs[, "acres92"], N = sum(Nh), estimate = "mean",
 ↪  show.details = FALSE)$estimate
}
```

Comparing the three designs

```
sim_results_str_region <- data.frame(
  "SRS" = srs_simulated[,1],
  "Prop2size" = str_p2s_simulated[,1],
  "Neyman" = str_neyman_simulated[,1]
)
boxplot(sim_results_str_region)
abline(h = mean(agpop$acres92), col = "red")
```

Table 4.5: Simulated mean, variance, and variance reduction by allocation method (region strata)

	Mean	Variance	Relative Variance	Percentage of Variance Reduction
SRS	308.9	583.4	1.000	0.00%
Prop2size	307.9	452.0	0.775	22.53%
Neyman	307.9	297.0	0.509	49.09%

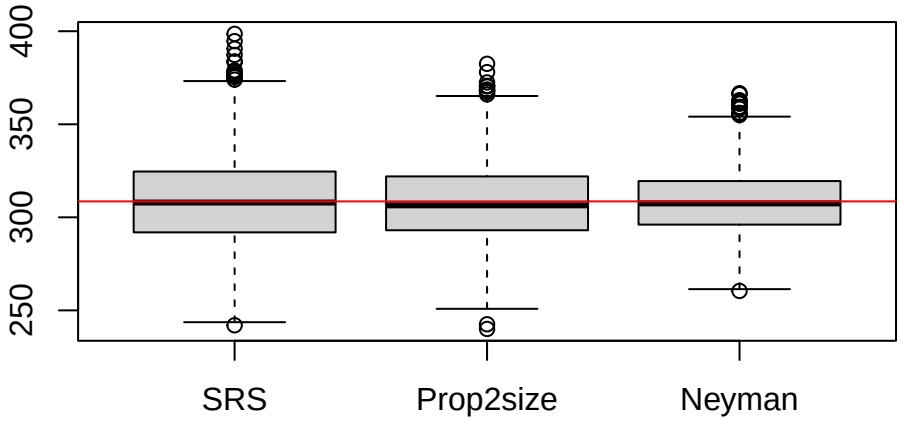

Figure 4.8: Sampling distribution of the mean under SRS, proportional, and Neyman allocation (region strata, 2000 simulations)

```
sim_means <- sapply(sim_results_str_region, mean)
sim_var <- sapply(sim_results_str_region, var)
sim_var_relative <- sim_var / sim_var[1]
sim_var_reduction <- 1 - sim_var_relative

data.frame(
  "Mean" = sim_means,
  "Variance" = sim_var,
  "Relative Variance" = sim_var_relative,
  "Percentage of Variance Reduction" = sim_var_reduction,
  check.names = FALSE
) |>
  gt(rownames_to_stub = TRUE) |>
  fmt_number(columns = c(Mean, Variance), decimals = 1) |>
  fmt_percent(columns = c(`Percentage of Variance Reduction`), decimals = 2) |>
  fmt_number(columns = c(`Relative Variance`), decimals = 3)
```

All three designs are centred on the true mean (the red line). Proportional allocation reduces the variance by 22.5% compared with SRS, close to $R^2 = 0.182$ as the theory predicts. Neyman allocation reduces it by 49.1%, because the regions differ greatly in their standard deviations S_h.

4.10.2 Stratifying by acres82

Now we stratify by a variable that is highly correlated with the target: the farm acreage of each county a decade earlier (`acres82`).

```
agpop <- acres_in_thousands(read.csv("data/agpop.csv"))
## acres82 is the stratifier here, so drop the counties missing either variable
agpop <- agpop[!is.na(agpop$acres92) & !is.na(agpop$acres82), ]
N <- nrow(agpop)
```

We convert the continuous `acres82` into four strata by cutting it at its 25th, 75th and 95th percentiles. The cut is deliberately uneven. The top 5% of counties, where the acreages run away, get a stratum of their own, as in dollar stratification. This is what makes the stratification so effective below.

```
agpop$stratum <- cut(agpop$acres82,
                     breaks = quantile(agpop$acres82, probs = c(0,0.25,0.75,0.95,1)),
                     include.lowest = T)
levels(agpop$stratum) <- paste0("acres82", c("Q1", "Q2", "Q3", "Q4"))
agpop <- agpop[order(agpop$stratum), ]
```

```
boxplot(acres92 ~ stratum, data = agpop)
```

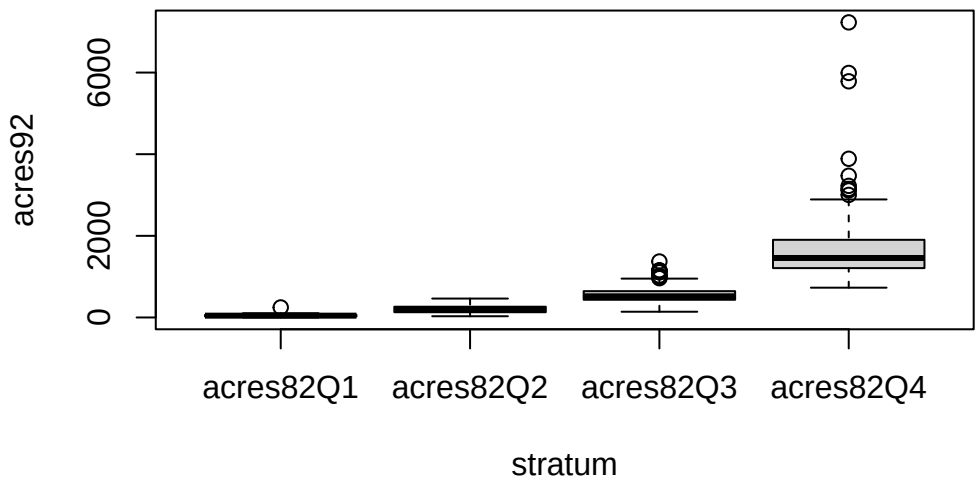

Figure 4.9: 1992 farm acreage by acres82 quantile stratum

```
lm_acres82 <- lm(agpop$acres92 ~ agpop$stratum)
lm_acres82_gt <- format_lm_gt(lm_acres82)
lm_acres82_gt$fit
```

Model Fit Summary

R^2	Adj. R^2	$\hat{\sigma}$	F-statistic	p-value
0.7402	0.7400	216.9580	2,895.9129	0.0000

```
r_squared_acres82 <- summary(lm_acres82)$r.squared
```

The R-squared is now 0.74: the earlier acreage explains 74% of the variance in 1992 acreage. This makes it a far better stratification variable than region.

Repeating proportional allocation 2000 times

```
Nh <- tapply(1:nrow(agpop), agpop$stratum, length)
nh <- round(Nh/sum(Nh)*300)

nres <- 2000
str_p2s_simulated <- matrix(0, nres, 4)
for (i in 1:nres) {
    strsample <- strata(agpop, "stratum", size = nh, method = "srswor")
    agstrat <- agpop[strsample$ID_unit, ]
    agstrat$weight <- 1/strsample$Prob
    str_p2s_simulated[i,] <- str_est_data(agstrat, "acres92", "stratum", "weight", estimate =
 ↪  "mean", show.details = FALSE)$estimate
}
```

Repeating Neyman allocation 2000 times

```
Nh <- tapply(1:nrow(agpop), agpop$stratum, length)
Sh <- tapply(agpop$acres92, agpop$stratum, sd)
nh_opt <- round((Nh*Sh)/sum(Nh*Sh) * 300)

nres <- 2000
str_neyman_simulated <- matrix(0, nres, 4)
for (i in 1:nres) {
    strsample <- strata(agpop, "stratum", size = nh_opt, method = "srswor")
    agstrat <- agpop[strsample$ID_unit, ]
    agstrat$weight <- 1/strsample$Prob
    str_neyman_simulated[i,] <- str_est_data(agstrat, "acres92", "stratum", "weight", estimate
 ↪  = "mean", show.details = FALSE)$estimate
}
```

Repeating SRS 2000 times

```
nres <- 2000
srs_simulated <- matrix(0, nres, 4)
for (i in 1:nres) {
    srs <- sample(sum(Nh), sum(nh))
    agsrs <- agpop[srs, ]
    srs_simulated[i,] <- srs_est(agsrs[, "acres92"], N = sum(Nh), estimate = "mean",
 ↪  show.details = FALSE)$estimate
}
```

Comparing the three designs

```
sim_results_str_acres82 <- data.frame(
  "SRS" = srs_simulated[,1],
  "Prop2size" = str_p2s_simulated[,1],
  "Neyman" = str_neyman_simulated[,1]
)
boxplot(sim_results_str_acres82)
abline(h = mean(agpop$acres92), col = "red")
```

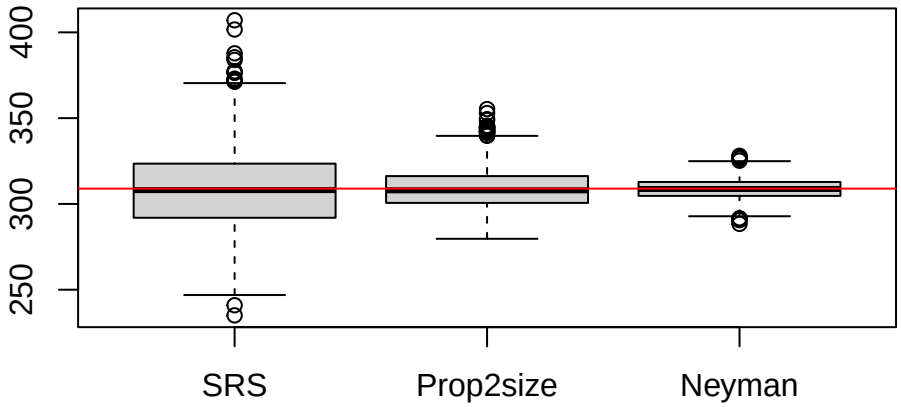

Figure 4.10: Sampling distribution of the mean under SRS, proportional, and Neyman allocation (acres82 strata, 2000 simulations)

```
sim_means <- sapply(sim_results_str_acres82, mean)
sim_var <- sapply(sim_results_str_acres82, var)
sim_var_relative <- sim_var / sim_var[1]
sim_var_reduction <- 1 - sim_var_relative

data.frame(
  "Mean" = sim_means,
  "Variance" = sim_var,
```

Table 4.6: Simulated mean, variance, and variance reduction by allocation method (acres82 strata)

	Mean	Variance	Relative Variance	Percentage of Variance Reduction
SRS	308.5	537.1	1.000	0.00%
Prop2size	309.2	138.3	0.257	74.26%
Neyman	308.7	34.9	0.065	93.50%

```
  "Relative Variance" = sim_var_relative,
  "Percentage of Variance Reduction" = sim_var_reduction,
  check.names = FALSE
) |>
  gt(rownames_to_stub = TRUE) |>
  fmt_number(columns = c(Mean, Variance), decimals = 1) |>
  fmt_percent(columns = c(`Percentage of Variance Reduction`), decimals = 2) |>
  fmt_number(columns = c(`Relative Variance`), decimals = 3)
```

The results are dramatic. Proportional allocation reduces the variance by 74.3%, again close to $R^2 = 0.74$. Neyman allocation reduces it by 93.5%. The top stratum is both small and highly variable, so Neyman allocation samples it heavily, just as dollar stratification samples the largest loans. Stratifying by a variable that is highly correlated with y is very powerful.

4.11 Interactive Demonstration: Stratification by `acres82` or `region`

The simulation for this section runs as a standalone app: Stratified Sampling. It opens in a new tab, together with its documentation and the full list of apps.

5 Ratio and Regression Estimation

5.1 Introduction

When auxiliary information (a variable x) is highly correlated with our target variable of interest (y), we can use **ratio** or **regression** estimation to improve precision without increasing the sample size. We can also use auxiliary population structures to adjust our estimates after the fact via **post-stratification**, or estimate means for specific sub-populations (**domains**).

Throughout, $(x_i, y_i), i = 1, \dots, n$, is an SRS of size n from a population of N units, and $\bar{x}_U$ (equivalently $t_x = N\bar{x}_U$) is known. Each topic below has the same layout: a *theory* part that states the estimator, explains where it comes from and gives its standard error, followed by worked examples with data.

The topics are less separate than they look. The ratio, regression and post-stratified estimators each fill in a prediction $\hat{y}_i$ for every unit of the population and average them; they differ only in the auxiliary information used to form $\hat{y}_i$, and a domain mean is itself a ratio estimator. The regression and post-stratification sections make this explicit, and the post-stratification section ends with a table that puts the estimators side by side.

5.2 Functions and packages for Analyzing Data

We begin by loading the necessary libraries, including `gt` for formatting mathematical tables, and defining our custom functions for Ratio, Regression, Domain, and Post-stratified estimation. We also define a versatile `gt` helper to render our estimation outputs cleanly.

5.2.1 General functions for ratio and regression estimation

The estimating functions used throughout this book (including `reg_est`, `ratio_est`, `domain_est`, `srs_est`, and the `format_est_gt` table-formatting helper used below) live in a single shared file, `samplingestimate.r`, which we source below.

For post-stratification, we reuse the shared `str_est()` stratified estimator, but we plug in the expected sample sizes $n_h = n \times \frac{N_h}{N}$ instead of the actual observed random n_h.

The full source of `samplingestimate.r` is listed in Appendix C.

5.3 Ratio Estimation

5.3.1 Theory

5.3.1.1 The ratio estimator

For ratio estimation to apply, two quantities y_i and x_i must be measured on each sample unit; x_i is often called an **auxiliary variable** or **subsidiary variable**. In the population of size N,

$$t_y = \sum_{i=1}^{N} y_i, \qquad t_x = \sum_{i=1}^{N} x_i, \tag{5.1}$$

and their ratio is

$$B = \frac{t_y}{t_x} = \frac{\bar{y}_U}{\bar{x}_U}, \qquad \text{so that} \qquad \bar{y}_U = B \cdot \bar{x}_U. \tag{5.2}$$

Ratio and regression estimation both take advantage of the correlation of x and y in the population; the higher the correlation, the better they work.

If an SRS is taken, natural estimators for B, t_y and $\bar{y}_U$ are

$$\boxed{\hat{B} = \frac{\bar{y}}{\bar{x}} = \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i} = \frac{\hat{t}_y}{\hat{t}_x}, \qquad \hat{t}_{yr} = \hat{B}\, t_x = N \hat{\bar{y}}_r, \qquad \hat{\bar{y}}_r = \hat{B}\, \bar{x}_U}, \tag{5.3}$$

where t_x and $\bar{x}_U$ are assumed **known**. This is the key requirement for ratio estimation: we need to know the population total or mean of the auxiliary variable x in advance. The ratio estimator is slightly biased, but the bias is negligible when n is large.

There is a useful way to read these formulas. Once $\hat{B}$ is estimated, form the **fitted value** $\hat{y}_i = \hat{B} x_i$ for every unit in the population, using its known x_i. Summing or averaging these over the population recovers the ratio estimators:

$$\hat{t}_{yr} = \sum_{i=1}^{N} \hat{y}_i = \hat{B}\, t_x, \qquad \hat{\bar{y}}_r = \frac{1}{N} \sum_{i=1}^{N} \hat{y}_i = \hat{B}\, \bar{x}_U. \tag{5.4}$$

Figure 5.1 shows this for a small population. The grey points are the population, whose x_i are known, and the green circles are an ordinary SRS. The red line $\hat{y} = \hat{B}x$ is fitted from the sample, and the red points are the fitted values for every unit.

```
plot_ratio_concept()
```

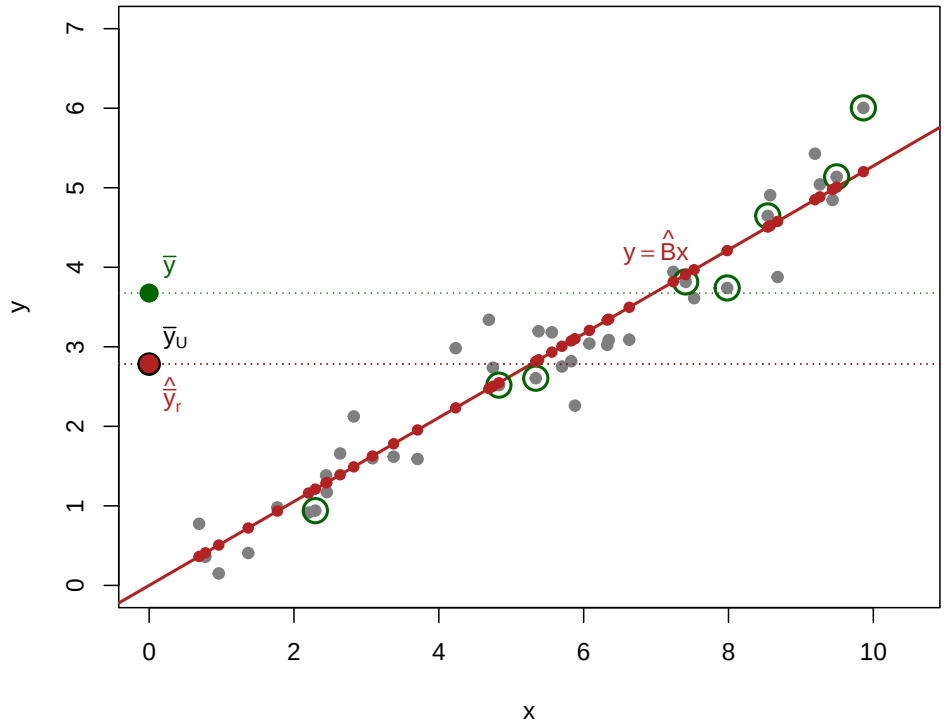

Figure 5.1: Illustration of the benefits of ratio estimation. Grey: the population. Green circles: an SRS. Red: the line fitted from the sample and the fitted values for every unit.

This sample happens to have $\bar{x} > \bar{x}_U$, so the naive $\bar{y}$ (green) lands well above $\bar{y}_U$. The ratio estimator corrects for that through the known $\bar{x}_U$: it is the height of the red line at $x = \bar{x}_U$, so $\hat{\bar{y}}_r$ (red) is close to $\bar{y}_U$ (black). Averaged over samples both are (essentially) unbiased, but $\hat{\bar{y}}_r$ varies far less.

5.3.1.2 Where ratio estimation is used

Ordinary SRS estimation of a mean or proportion is a special case of ratio estimation, obtained by setting $x_i = 1$ for every unit: then $\bar{x}_U = \bar{x} = 1$, $t_x = N$, $B = \bar{y}_U$ and $\hat{B} = \bar{y}$, so $\hat{\bar{y}}_r = \bar{y}$ (including $\hat{p}$) and $\hat{t}_{yr} = N\bar{y}$.

Genuine ratio estimation needs an x whose population total is known more easily than that of y. Three typical situations:

- **Counting people.** Let x_i be the number of registered births in commune i of France and y_i the number of residents. The total number of births is known from administrative records, so $\hat{B} = \hat{t}_y/\hat{t}_x$ estimates residents per birth, and $\hat{t}_{yr} = \hat{B}\,t_x$ estimates the population of France. For example, if a sample of communes has $\hat{t}_y = 2{,}037{,}615$ residents and $\hat{t}_x = 71{,}866$ births, then $\hat{B} = 2{,}037{,}615/71{,}866 = 28.35$ residents per birth, and with $t_x = 1$ million births in all of France, $\hat{t}_{yr} = 28.35 \times 1$ million $= 28.35$ million residents.
- **Field yield.** Let y_i be the bushels of grain harvested in field i and x_i its acreage. Then B is the average yield in bushels per acre, $\bar{y}_U$ the average yield per field and t_y the total yield. The total acreage t_x is usually easy to obtain from other records, even when t_y is not.

- **The same quantity at an earlier time.** When the same quantity is measured at different times, an earlier measurement often makes an excellent auxiliary variable. If the 1987 acreage of every county is known but 1992 acreage is observed only for a sample of counties, let y_i be the 1992 farm acreage and x_i the 1987 farm acreage. This is the example used in the simulation study at the end of this chapter.

5.3.1.3 Standard error of the ratio estimator

In large samples the bias of $\hat{\bar{y}}_r$ is negligible relative to its variance, so $\text{MSE}(\hat{\bar{y}}_r) \approx V(\hat{\bar{y}}_r)$. To understand the variance, look at the estimation error,

$$\hat{\bar{y}}_r - \bar{y}_U = \frac{\bar{y}}{\bar{x}}\bar{x}_U - B\bar{x}_U = \frac{\bar{x}_U}{\bar{x}}\left(\bar{y} - B\bar{x}\right), \tag{5.5}$$

where the second form comes from factoring out $\bar{x}_U/\bar{x}$. Define the **population residual** of each element as its deviation from the true population line,

$$d_i = y_i - Bx_i \qquad (B = t_y/t_x). \tag{5.6}$$

Then $\bar{y} - B\bar{x}$ is simply the sample mean $\bar{d}$ of these residuals, so the error of the estimator is the scaled sample mean of the residuals. Squaring and taking expectations,

$$\text{MSE}(\hat{\bar{y}}_r) \approx \left(\frac{\bar{x}_U}{\bar{x}}\right)^2 V(\bar{d}), \qquad V(\bar{d}) = \left(1 - \frac{n}{N}\right)\frac{S_d^2}{n}, \tag{5.7}$$

because the d_i are fixed population values, so $\bar{d}$ is an ordinary sample mean under SRS.

In practice B is unknown, so the d_i cannot be computed. We replace B by $\hat{B}$ and use the **sample residuals** from the fitted line,

$$e_i = y_i - \hat{B}x_i, \qquad s_e^2 = \frac{1}{n-1}\sum_{i=1}^{n} e_i^2. \tag{5.8}$$

Substituting s_e^2 for the unknown S_d^2 gives the variance estimators

$$\boxed{\begin{aligned} &\hat{V}(\hat{B}) = \frac{1}{\bar{x}^2}\left(1 - \frac{n}{N}\right)\frac{s_e^2}{n}, \\ &\hat{V}(\hat{\bar{y}}_r) = \bar{x}_U^2\,\hat{V}(\hat{B}) = \left(\frac{\bar{x}_U}{\bar{x}}\right)^2\left(1 - \frac{n}{N}\right)\frac{s_e^2}{n}, \\ &\hat{V}(\hat{t}_{yr}) = N^2\,\hat{V}(\hat{\bar{y}}_r) \end{aligned}} \tag{5.9}$$

For samples that are large enough (typically $n > 30$) the sampling distributions are approximately normal, and approximate 95% intervals are $\hat{B} \pm t_{0.975,\,n-1}\,\text{SE}(\hat{B})$ for the ratio, and the same form for the mean and the total.

The sample residual e_i is the vertical distance from the point (x_i, y_i) to the fitted line through the origin, $y = \hat{B}x$. No centring appears in s_e^2 because $\hat{B} = \bar{y}/\bar{x}$ forces $\sum_i e_i = \sum_i y_i - \hat{B}\sum_i x_i = 0$ exactly, so the residuals already average to zero; the divisor is $n - 1$ for the one parameter $\hat{B}$ estimated from the sample. Compare the variance with the SRS one, $(1 - \frac{n}{N})\,s_y^2/n$: ratio estimation replaces the scatter of y about $\bar{y}$ by the scatter about the line. That is the whole source of the gain, and the whole risk, since an x that predicts y poorly gives $s_e^2 > s_y^2$ and a *less* precise estimate than the plain sample mean.

5.3.1.4 When does ratio estimation help?

Let R be the population Pearson correlation of x and y,

$$R = \frac{\sum_{i=1}^{N}(x_i - \bar{x}_U)(y_i - \bar{y}_U)}{(N-1)S_x S_y}. \tag{5.10}$$

Comparing $\mathrm{MSE}(\hat{\bar{y}}_r)$ with $\mathrm{MSE}(\bar{y}) = (1 - \frac{n}{N})\, S_y^2/n$ shows that

$$\mathrm{MSE}(\hat{\bar{y}}_r) \le \mathrm{MSE}(\bar{y}) \quad \Longleftrightarrow \quad R \ge \frac{BS_x}{2S_y} = \frac{\mathrm{CV}(x)}{2\,\mathrm{CV}(y)}. \tag{5.11}$$

If the coefficients of variation of x and y are roughly equal, ratio estimation pays off whenever the correlation between x and y exceeds $1/2$.

The gain can be summarized, analogously to R^2 for stratification, by

$$\frac{\mathrm{MSE}(\hat{\bar{y}}_r)}{\mathrm{MSE}(\bar{y})} \approx \frac{S_d^2}{S_y^2} \qquad \Longrightarrow \qquad 1 - \frac{S_d^2}{S_y^2} \text{ is the proportion of variance removed.} \tag{5.12}$$

This is not exactly the R^2 of fitting the linear model $y = Bx$, because $\hat{B}$ is not a least-squares estimate of B, but it carries similar information about how much variance ratio estimation removes.

5.3.2 Example: Population of France (by hand)

This toy version of the France population example demonstrates the mechanics of ratio estimation, using a sample of $n = 5$ out of $N = 8$ communes. We observe x_i = registered births and y_i = residents, both in thousands:

Commune i	x_i (000s)	y_i (000s)	$\hat{y}_i = \hat{B}x_i$	$e_i = y_i - \hat{y}_i$	e_i^2
1	1	18	30	−12	144
2	2	78	60	18	324
3	3	62	90	−28	784
4	4	132	120	12	144
5	5	160	150	10	100
Sum	**15**	**450**	**450**	**0**	**1496**

Figure 5.2 shows the five communes, the fitted line $y = \hat{B}x$ through the origin, and the residuals e_i.

The ratio estimate of the slope and the sample residual variance are

$$\hat{B} = \frac{\sum y_i}{\sum x_i} = \frac{450}{15} = 30, \qquad s_e^2 = \frac{\sum e_i^2}{n-1} = \frac{1496}{4} = 374. \tag{5.13}$$

Assume the number of registered births is known from administrative records to be $t_x = 40$ (thousand) in all $N = 8$ communes. Then $\bar{x}_U = t_x/N = 5$ (thousand), while the sample mean is only $\bar{x} = 3$ (thousand): we sampled

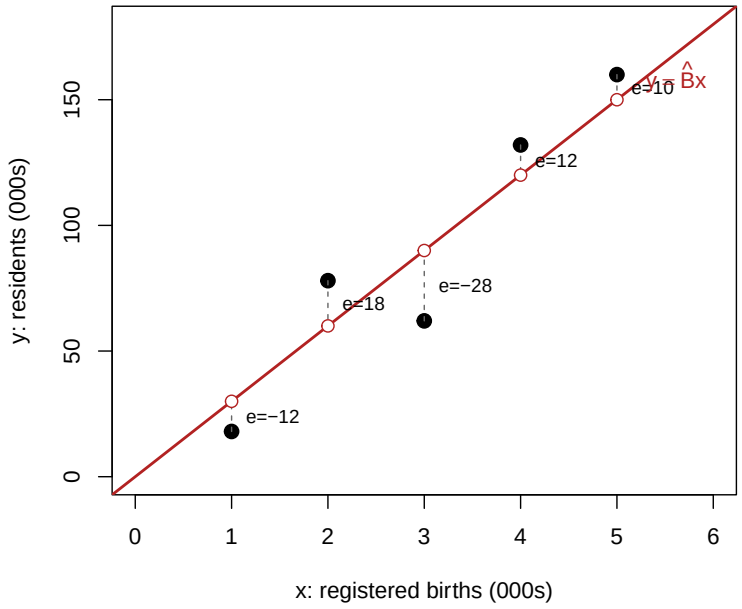

Figure 5.2: The toy France example: residents against registered births for the $n = 5$ sampled communes, with the ratio line $y = \hat{B}x$ (red) and the residuals e_i (dashed).

smaller-than-average communes. The estimates are

$$\hat{\bar{y}}_r = \hat{B}\bar{x}_U = 30(5) = 150 \text{ (thousand residents)}, \qquad \text{SE}(\hat{\bar{y}}_r) = \sqrt{\left(1 - \frac{5}{8}\right)\left(\frac{5}{3}\right)^2 \frac{374}{5}} = 8.827, \tag{5.14}$$

$$\hat{t}_{yr} = \hat{B}\, t_x = 30(40) = 1200 \text{ (thousand)}, \tag{5.15}$$

$$\text{SE}(\hat{t}_{yr}) = \sqrt{\left(1 - \frac{5}{8}\right)\left(\frac{40}{3}\right)^2 \frac{374}{5}} = 70.616 \text{ (thousand)}, \tag{5.16}$$

so a 95% confidence interval for the total population is $1200 \pm 1.96(70.616) = [1061.6,\ 1338.4]$ thousand, that is, **[1.062, 1.338] million** residents. For the slope, $\text{SE}(\hat{B}) = \sqrt{(1 - \frac{5}{8})\,\frac{374}{5 \cdot 3^2}} = 1.765$.

5.3.3 Example: Farm Acreage, 1992 from 1987 (agsrs.csv)

When the same quantity is measured at different times, an earlier measurement often makes an excellent auxiliary variable. The Census of Agriculture gives the 1987 farm acreage of every county, but suppose the 1992 acreage is known only for the SRS of $n = 300$ counties in `agsrs.csv`. Let

$$y_i = \text{farm acreage of county } i \text{ in 1992}, \qquad x_i = \text{farm acreage of county } i \text{ in 1987}. \tag{5.17}$$

In 1987 a total of $t_x = 964{,}470{,}625$ acres were devoted to farms, so $\bar{x}_U = 964{,}470{,}625/3078 = 313{,}343.3$ acres per county, or 313.3 thousand acres in the units used here.

```
agsrs <- acres_in_thousands(read.csv("data/agsrs.csv"))
tx <- 964470.625                          # thousands of acres, all 3078 counties in 1987
res_ag_ratio <- ratio_est(agsrs$acres92, agsrs$acres87, xbarU = tx / 3078, N = 3078,
                          estimate = "mean", show.details = FALSE)$estimate
res_ag_srs   <- srs_est(agsrs$acres92, N = 3078, estimate = "mean", show.details =
 ↪  FALSE)$estimate
Bhat_ag <- mean(agsrs$acres92) / mean(agsrs$acres87)
c(ybar = mean(agsrs$acres92), xbar = mean(agsrs$acres87), Bhat = Bhat_ag)
```

```
      ybar       xbar       Bhat
297.8970467 301.9537233   0.9865652
```

```
res_ag_ratio |> format_est_gt("ratio_mean")
```

$\bar{y}_r$	SE($\bar{y}_r$)	95% CI Lower	95% CI Upper
309.13	1.80	305.59	312.68

```
res_ag_srs |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
297.90	18.90	260.71	335.09

From the sample, $\bar{y} = 297.9$ and $\bar{x} = 302$ thousand acres, so

$$\hat{B} = \frac{\bar{y}}{\bar{x}} = 0.986565, \qquad \hat{\bar{y}}_r = \hat{B}\,\bar{x}_U = 309.1, \qquad \hat{t}_{yr} = \hat{B}\,t_x = 951,513 \text{ thousand acres.} \tag{5.18}$$

The plain SRS estimate of the total, $N\bar{y} = 916,927$ thousand acres, is quite different, because the sampled counties happened to have $\bar{x}$ a little below $\bar{x}_U$. The ratio estimator corrects for this, and its standard error, 1.8, is about a tenth of the SRS standard error, 18.9.

```
par(mar = c(4, 4, 0.5, 0.5))
plot(agsrs$acres87, agsrs$acres92, pch = 19, col = adjustcolor("grey30", 0.6), las = 1,
     xlab = "Farm acreage 1987 (thousands)", ylab = "Farm acreage 1992 (thousands)")
abline(0, Bhat_ag, col = "firebrick", lwd = 2)
```

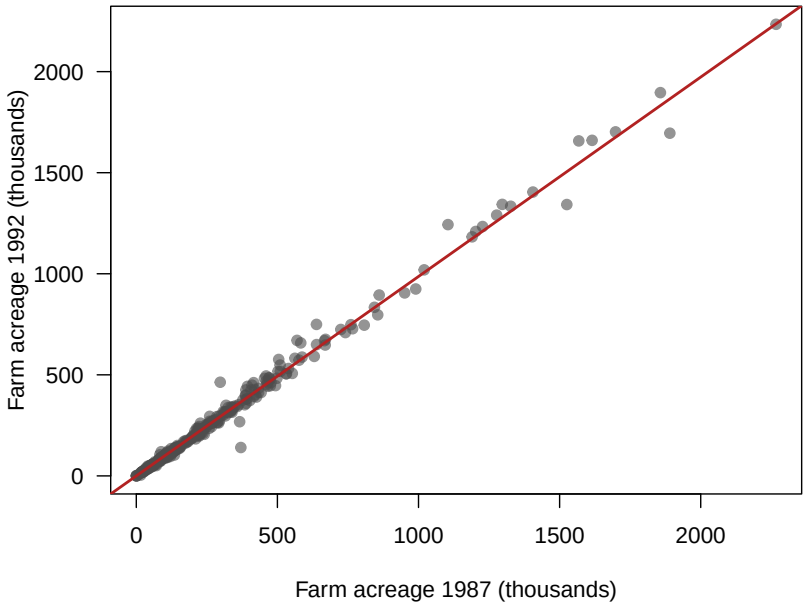

Figure 5.3: 1992 against 1987 farm acreage for the SRS of 300 counties, with the ratio line $y = \hat{B}x$ through the origin. The spread about the line grows with x.

The points lie close to a line through the origin, and the variability about the line **increases with** x. This pattern, with the variance of y roughly proportional to x, is exactly when ratio estimation works best.

5.3.4 Example: Cherry Trees (cherry.csv)

We want to estimate the volume of wood in cherry trees using their diameter as an auxiliary variable.

5.3.4.1 Importing data

First, let's load the data and look at the relationship between volume and diameter.

```
cherry <- read.csv("data/cherry.csv", header = T)
head(cherry) |> gt()
```

A scatter plot of volume against diameter shows whether the points lie close to a line through the origin.

```
plot(cherry$volume ~ cherry$diameter, main = "Volume vs Diameter")
```

Table 5.1: Sample data: cherry tree volume and diameter

diameter	height	volume
8.3	70	10.3
8.6	65	10.3
8.8	63	10.2
10.5	72	16.4
10.7	81	18.8
10.8	83	19.7

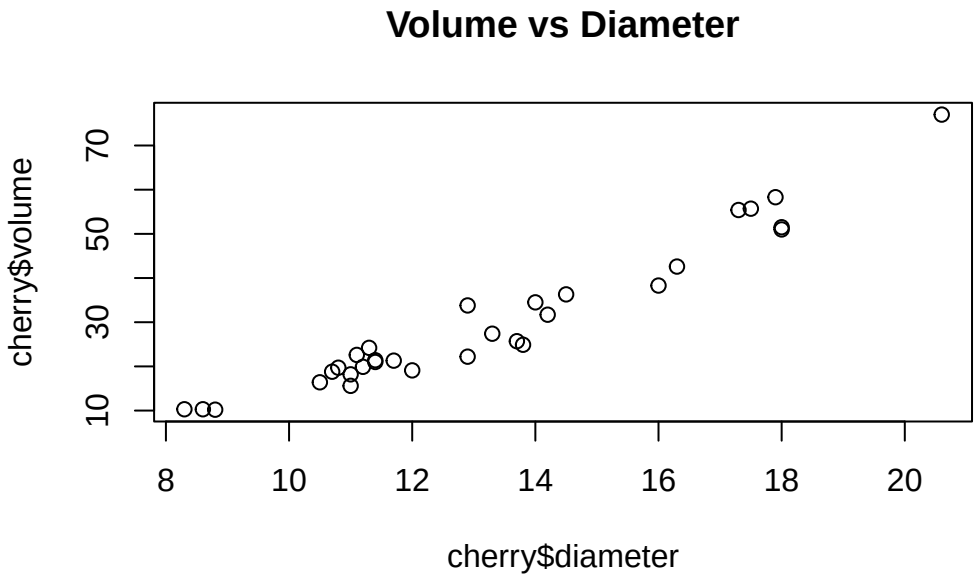

Figure 5.4: Cherry tree volume vs. diameter

Fitting the linear model with the intercept forced to zero, $y = Bx$, summarizes how strongly volume is related to diameter.

```
anova(lm(cherry$volume ~ 0 + cherry$diameter))
```

```
Analysis of Variance Table

Response: cherry$volume
               Df Sum Sq Mean Sq F value    Pr(>F)
cherry$diameter  1  33622   33622  373.12 < 2.2e-16 ***
Residuals       30   2703      90
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output Note: The plot and the linear model output (which forces a 0 intercept) confirm a very strong relationship, making this a prime candidate for Ratio estimation.

5.3.4.2 SRS estimate

To establish a baseline, we first estimate the mean and total volume using standard Simple Random Sampling (ignoring the auxiliary variable).

```
N <- 2967
## estimating the mean of volume
res_srs_vol <- srs_est(cherry$volume, N = N, estimate = "mean", show.details = FALSE)
srs_mean_vol_raw <- res_srs_vol$estimate
srs_mean_vol_raw |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
30.17	2.94	24.17	36.17

```
## estimating the total of volume
res_srs_total_vol <- srs_est(cherry$volume, N = N, estimate = "total", show.details = FALSE)
srs_total_vol_raw <- res_srs_total_vol$estimate
srs_total_vol_raw |> format_est_gt("total")
```

$\hat{t}$	SE($\hat{t}$)	95% CI Lower	95% CI Upper
89,517.26	8,713.67	71,721.58	107,312.94

Output Note: Under SRS, the standard error for the mean volume is 2.94.

5.3.4.3 Ratio Estimation Using the Function

We calculate the ratio $\hat{B} = \overline{y}/\overline{x}$; the function's `$table` shows the row-level working values x_i, y_i, fitted $\hat{y}_i = \hat{B}x_i$, residuals $e_i = y_i - \hat{y}_i$, and e_i^2 (with a Sum row and a footnote giving $\hat{B}$ and s_e^2), so we no longer need to build this table by hand.

```
ydata <- cherry$volume
xdata <- cherry$diameter
N <- 2967

res_ratio_model <- ratio_est(ydata, xdata, N = N, estimate = "model")
res_ratio_model$table
```

Ratio Estimation: Working Table
$n = 31$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	10.30	8.30	18.90	-8.60	73.99
2	10.30	8.60	19.59	-9.29	86.21
3	10.20	8.80	20.04	-9.84	96.84
4	16.40	10.50	23.91	-7.51	56.43
5	18.80	10.70	24.37	-5.57	31.00
6	19.70	10.80	24.60	-4.90	23.96
...	...	...	...	...	...
Sum	**935.30**	**410.70**	**935.30**	**0.00**	**2,821.59**

[1] $\hat{B} = \sum_i y_i / \sum_i x_i = 2.28$, $\hat{y}_i = \hat{B}\, x_i$, $s_e^2 = 94.05$.

$$\hat{B} = \frac{\sum_i y_i}{\sum_i x_i} = \frac{935.30}{410.70} = 2.28$$

$$\text{SE}(\hat{B}) = \frac{1}{\bar{x}} \sqrt{\left(1 - \frac{n}{N}\right) \frac{s_e^2}{n}} = \frac{1}{13.25} \sqrt{\left(1 - \frac{31}{2967}\right) \frac{94.05}{31}} = 0.13$$

We plot the data again, now with the fitted line $y = \hat{B}x$ through the origin.

```
plot(cherry$volume ~ cherry$diameter, main="Ratio Estimation Fit")
abline(a = 0, b = res_ratio_model$estimate["Est."], col="blue", lwd=2)
```

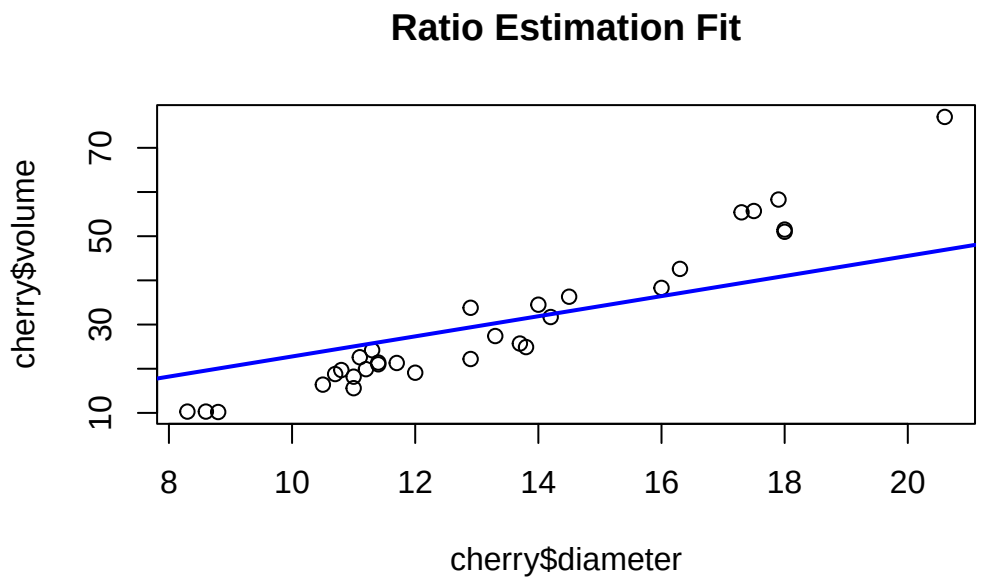

Figure 5.5: Cherry tree volume vs. diameter with the fitted ratio line

Output Note: The blue line represents our estimated ratio $\hat{B} \approx 2.28$. The working table above shows the individual residuals used to estimate s_e^2.

```
B_v2d <- res_ratio_model$estimate
B_v2d |> format_est_gt("ratio")
```

$\hat{B}$	SE($\hat{B}$)	95% CI Lower	95% CI Upper
2.28	0.13	2.01	2.54

Output Note: The ratio $\hat{B}$ is estimated at 2.277 with a standard error of 0.1308.

5.3.4.3.1 Estimating the mean volume of wood

If we know the true total of the population diameters is 41,835, we can derive the population mean diameter $\overline{x}_{\mathrm{U}}$ and pass it as `xbarU` with `estimate = "mean"`.

```
mean_diameters <- 41835 / N
res_ratio_mean <- ratio_est(ydata, xdata, xbarU = mean_diameters, N = N, estimate = "mean")
res_ratio_mean$table
```

Ratio Estimation: Working Table
$n = 31$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	10.30	8.30	18.90	-8.60	73.99
2	10.30	8.60	19.59	-9.29	86.21
3	10.20	8.80	20.04	-9.84	96.84
4	16.40	10.50	23.91	-7.51	56.43
5	18.80	10.70	24.37	-5.57	31.00
6	19.70	10.80	24.60	-4.90	23.96
...	...	...	...	...	...
Sum	**935.30**	**410.70**	**935.30**	**0.00**	**2,821.59**

[1] $\hat{B} = \sum_i y_i / \sum_i x_i = 2.28$, $\hat{y}_i = \hat{B}\, x_i$, $s_e^2 = 94.05$.
$\bar{y}_r = \hat{B}\, \bar{x}_U = 2.28 \times 14.10 = 32.11$
$\mathrm{SE}(\bar{y}_r) = \mathrm{SE}(\hat{B})\, \bar{x}_U = 0.13 \times 14.10 = 1.84$

```
ratio_mean_volume_raw <- res_ratio_mean$estimate
ratio_mean_volume_raw |> format_est_gt("ratio_mean")
```

$\bar{y}_r$	SE($\bar{y}_r$)	95% CI Lower	95% CI Upper
32.11	1.84	28.34	35.88

5.3.4.3.2 Estimating the total volume of wood

Using `estimate = "total"` with the known population total diameter gives the estimated population total volume directly.

```
t_diameters <- 41835

res_ratio_total <- ratio_est(ydata, xdata, xbarU = t_diameters / N, N = N, estimate = "total")
res_ratio_total$table
```

Ratio Estimation: Working Table
$n = 31$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	10.30	8.30	18.90	-8.60	73.99
2	10.30	8.60	19.59	-9.29	86.21
3	10.20	8.80	20.04	-9.84	96.84
4	16.40	10.50	23.91	-7.51	56.43
5	18.80	10.70	24.37	-5.57	31.00
6	19.70	10.80	24.60	-4.90	23.96
...	...	...	...	...	...
Sum	**935.30**	**410.70**	**935.30**	**0.00**	**2,821.59**

[1] $\hat{B} = \sum_i y_i / \sum_i x_i = 2.28$, $\hat{y}_i = \hat{B}\, x_i$, $s_e^2 = 94.05$.
$\hat{t} = \hat{B}\, \bar{x}_U\, N = 2.28 \times 14.10 \times 2,967 = 95,272.16$
$\text{SE}(\hat{t}) = \text{SE}(\hat{B})\, \bar{x}_U\, N = 0.13 \times 14.10 \times 2,967 = 5,471.43$

```
ratio_total_volume <- res_ratio_total$estimate
ratio_total_volume |> format_est_gt("total")
```

$\hat{t}$	SE($\hat{t}$)	95% CI Lower	95% CI Upper
95,272.16	5,471.43	84,098.00	106,446.32

5.3.4.3.3 Percentage of Variance Reduction

How much did ratio estimation help compared to ignoring the auxiliary variable?

```
variance_reduction <- 1 - (ratio_mean_volume_raw["S.E."] / srs_mean_vol_raw["S.E."])^2
unname(variance_reduction)
```

```
[1] 0.6057237
```

Output Note: Using the diameter as an auxiliary variable reduced the variance of our estimate by 60.6% compared to simple random sampling.

5.4 Regression Estimation

5.4.1 Theory

5.4.1.1 The linear model and the least-squares fit

Ratio estimation works best if the data are well fit by a straight line *through the origin*, with spread growing with x. Sometimes data instead appear evenly scattered about a straight line that does **not** go through the origin, so that the usual straight-line regression model applies,

$$y = B_0 + B_1 x. \tag{5.19}$$

In ratio estimation $y_i = Bx_i + d_i$ with residual spread growing with x; in regression estimation $y_i = B_0 + B_1 x_i + d_i$ with roughly constant spread.

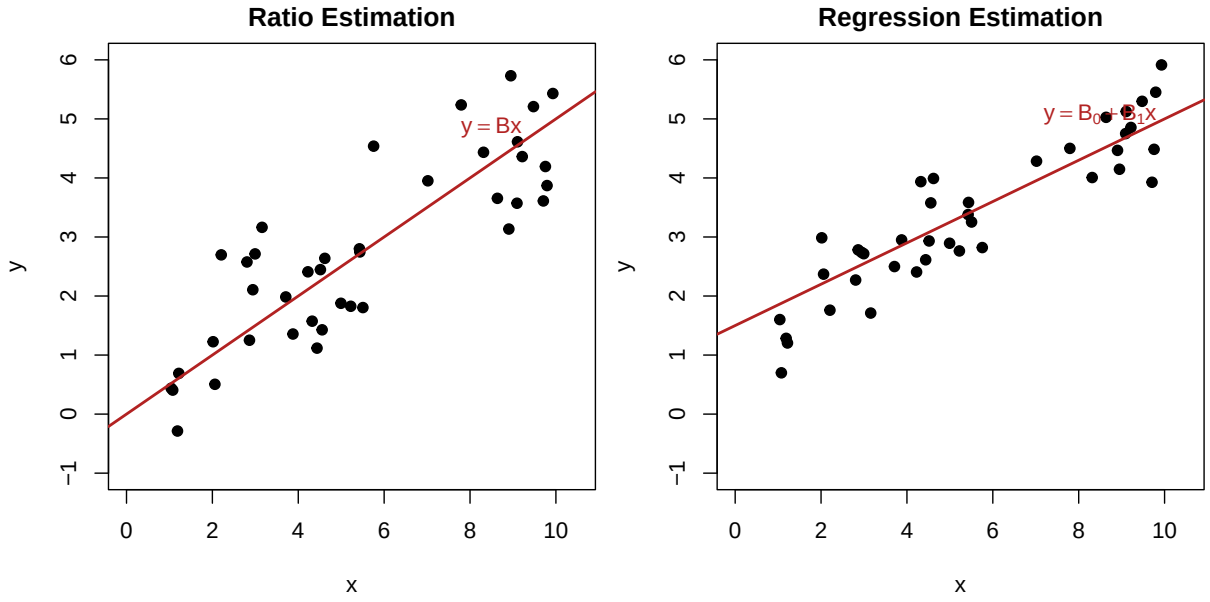

Figure 5.6: Data suited to ratio estimation (left: a line through the origin, with spread growing with x) and to regression estimation (right: a line with a non-zero intercept and constant spread).

The ordinary least squares slope and intercept, from the sample, are

$$\hat{B}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{r\, s_y}{s_x}, \qquad \hat{B}_0 = \bar{y} - \hat{B}_1 \bar{x}, \tag{5.20}$$

where r is the sample Pearson correlation coefficient of x and y.

5.4.1.2 The regression estimator

Assuming $\bar{x}_U$ (or t_x) is known, the regression estimator of the mean is the fitted line evaluated at $\bar{x}_U$:

$$\boxed{\hat{\bar{y}}_{reg} = \hat{B}_0 + \hat{B}_1 \bar{x}_U = \bar{y} + \hat{B}_1 \left(\bar{x}_U - \bar{x}\right), \qquad \hat{t}_{reg} = N\hat{\bar{y}}_{reg} = N\hat{B}_0 + \hat{B}_1 t_x}. \tag{5.21}$$

Equivalently, averaging the fitted value $\hat{y}_i = \hat{B}_0 + \hat{B}_1 x_i$ over *every* population unit gives the same estimator,

$$\hat{\bar{y}}_{reg} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{B}_0 + \hat{B}_1 x_i\right) = \hat{B}_0 + \hat{B}_1 \bar{x}_U. \tag{5.22}$$

This is a natural generalization of the ratio estimator $\hat{\bar{y}}_r = \frac{1}{N}\sum_{i=1}^{N} \hat{B} x_i$.

Figure 5.7 illustrates it. The grey points are the population and the green circles an ordinary SRS. The red line is fitted from the sample, $\hat{y}_i = \hat{B}_0 + \hat{B}_1 x_i$ for every unit, and the regression estimate is the average of the red points.

```
plot_reg_concept()
```

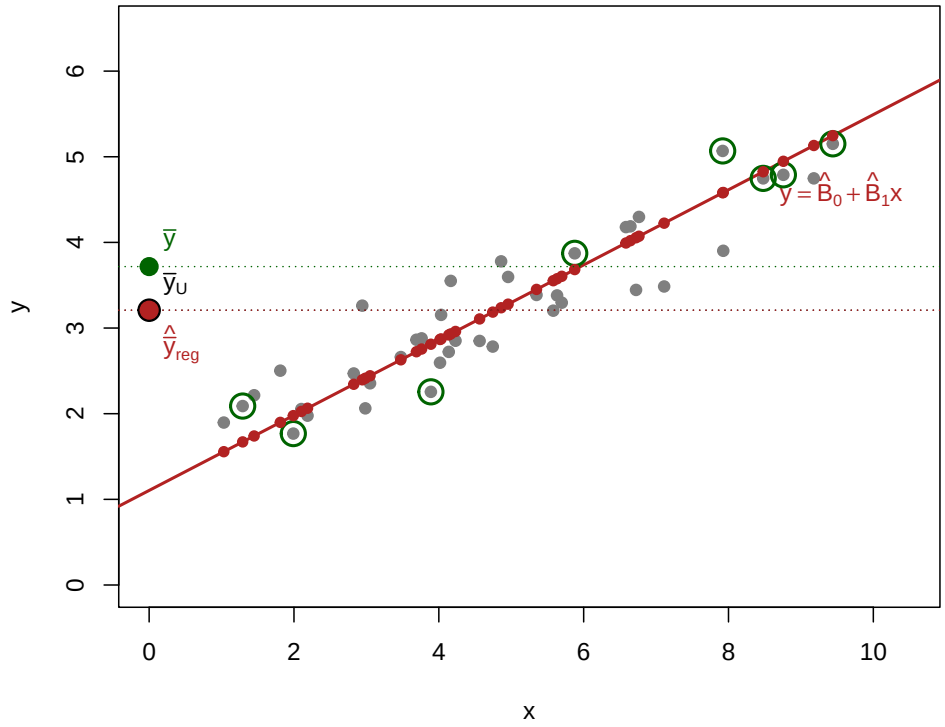

Figure 5.7: Illustration of the benefits of regression estimation. Grey: the population. Green circles: an SRS. Red: the line fitted from the sample and the fitted values for every unit.

The sampled x's average above $\bar{x}_U$, so the naive $\bar{y}$ (green) sits above $\bar{y}_U$; the regression estimator shifts the fit back to $\bar{x}_U$ and lands on target, $\hat{\bar{y}}_{reg}$ (red) $\approx \bar{y}_U$ (black). Both are essentially unbiased over repeated samples, but $\hat{\bar{y}}_{reg}$ is much less variable.

5.4.1.3 Standard error of the regression estimator

Let $e_i = y_i - (\hat{B}_0 + \hat{B}_1 x_i)$ be the regression residuals. The estimated variance uses the residual mean square of the regression,

$$\boxed{\hat{V}(\hat{\bar{y}}_{reg}) = \left(1 - \frac{n}{N}\right)\frac{s_e^2}{n}, \qquad s_e^2 = \frac{1}{n-2}\sum_{i=1}^{n}\left(y_i - \hat{B}_0 - \hat{B}_1 x_i\right)^2}, \tag{5.23}$$

with $\hat{V}(\hat{t}_{reg}) = N^2\hat{V}(\hat{\bar{y}}_{reg})$ and 95% intervals using $t_{0.975,\,n-2}$. Since $s_e^2 = s_y^2(1-r^2)(n-1)/(n-2)$, the standard error is approximately

$$\mathrm{SE}(\hat{\bar{y}}_{reg}) \approx \sqrt{\left(1-\frac{n}{N}\right)\frac{1}{n}\,s_y^2\,(1-r^2)}, \tag{5.24}$$

so the stronger the linear relationship, the smaller the SE.

The divisor is now $n-2$, since both $\hat{B}_0$ and $\hat{B}_1$ are estimated. For the same data $\sum_i e_i^2$ can only be smaller here than under the ratio fit (the ratio line is the special case $\hat{B}_0 = 0$), so the regression estimator is the safer of the two whenever the relationship misses the origin, at the cost of one degree of freedom. Note also that the factor $(\bar{x}_U/\bar{x})^2$ is absent: the fitted line is evaluated at $\bar{x}_U$ directly, so no correction for $\bar{x} \neq \bar{x}_U$ is needed.

5.4.2 Example: Cherry Trees (cherry.csv)

When the relationship between y and x doesn't pass exactly through the origin, we use Regression Estimation.

5.4.2.1 Importing data

We read the cherry data again. The known population total diameter, $t_x = 41{,}835$, over the $N = 2967$ trees gives the population mean diameter $\bar{x}_U$ needed by the regression estimator.

```
cherry <- read.csv("data/cherry.csv", header = T)
ydata <- cherry$volume
xdata <- cherry$diameter
t_diameters <- 41835
xbarU <- t_diameters / 2967
N <- 2967
```

5.4.2.2 Regression Estimation Using the Function

5.4.2.2.1 Fitting a linear regression model

First fit the ordinary least squares line of volume on diameter. This gives $\hat{B}_0$ and $\hat{B}_1$, together with the usual fit summary, and shows how well a straight line describes the data.

```
n <- length(ydata)
lmfit <- lm(ydata ~ xdata)
lmfit_gt <- format_lm_gt(lmfit)
lmfit_gt$coef
```

Coefficient Estimates

	Estimate	S.E.	t value	p value
(Intercept)	-36.9435	3.3651	-10.9783	0.0000
xdata	5.0659	0.2474	20.4783	0.0000

```
lmfit_gt$fit
```

Model Fit Summary

R^2	Adj. R^2	$\hat{\sigma}$	F-statistic	p-value
0.9353	0.9331	4.2520	419.3603	0.0000

The fitted line, which is not forced through the origin, is drawn over the data.

```
plot(xdata, ydata, main="Linear Regression: Volume vs Diameter")
abline(lmfit, col="green", lwd=2)
```

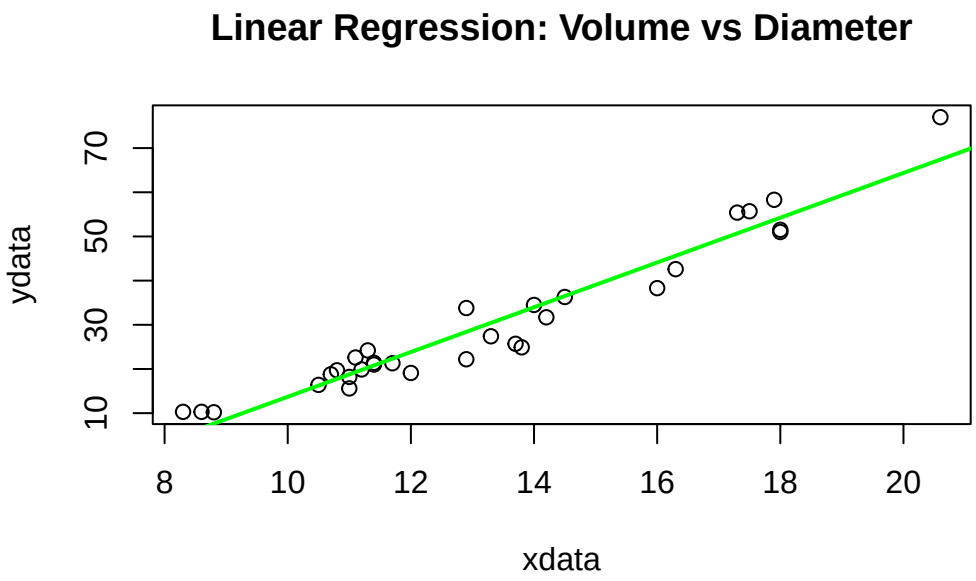

Figure 5.8: Cherry tree volume vs. diameter with the fitted regression line

The function's `$table` shows the row-level working values x_i, y_i, fitted $\hat{y}_i = \hat{B}_0 + \hat{B}_1 x_i$, residuals $e_i = y_i - \hat{y}_i$, and e_i^2 (with a Sum row and a footnote giving $\hat{B}_0$, $\hat{B}_1$, and s_e^2):

```
res_reg_model <- reg_est(ydata, xdata, N = N, estimate = "model")
res_reg_model$table
```

Regression Estimation: Working Table
$n = 31$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	10.30	8.30	5.10	5.20	27.01
2	10.30	8.60	6.62	3.68	13.52
3	10.20	8.80	7.64	2.56	6.57
4	16.40	10.50	16.25	0.15	0.02
5	18.80	10.70	17.26	1.54	2.37
6	19.70	10.80	17.77	1.93	3.73
...	...	...	...	...	...
Sum	**935.30**	**410.70**	**935.30**	**0.00**	**524.30**

[1] $\hat{B}_0 = -36.94$, $\hat{B}_1 = 5.07$, $\hat{y}_i = \hat{B}_0 + \hat{B}_1 x_i$, $s_e^2 = 18.08$.

$$\hat{B}_1 = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2} = \frac{1,496.64}{295.44} = 5.07$$

$$\text{SE}(\hat{B}_1) = \sqrt{\frac{s_e^2}{\sum_i (x_i - \bar{x})^2}} = \sqrt{\frac{18.08}{295.44}} = 0.25$$

5.4.2.2.2 Estimate the mean

We compute the regression estimate $\bar{y}_{reg}$ by passing `xbarU` with `estimate = "mean"`.

```
res_reg_mean <- reg_est(ydata, xdata, xbarU = xbarU, N = N, estimate = "mean")
res_reg_mean$table
```

Regression Estimation: Working Table
$n = 31$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	10.30	8.30	5.10	5.20	27.01
2	10.30	8.60	6.62	3.68	13.52
3	10.20	8.80	7.64	2.56	6.57
4	16.40	10.50	16.25	0.15	0.02
5	18.80	10.70	17.26	1.54	2.37
6	19.70	10.80	17.77	1.93	3.73
...	...	...	...	...	...
Sum	**935.30**	**410.70**	**935.30**	**0.00**	**524.30**

[1] $\hat{B}_0 = -36.94$, $\hat{B}_1 = 5.07$, $\hat{y}_i = \hat{B}_0 + \hat{B}_1 x_i$, $s_e^2 = 18.08$.

$$\bar{y}_{reg} = \hat{B}_0 + \hat{B}_1 \bar{x}_U = -36.94 + 5.07 \times 14.10 = 34.49$$

$$\text{SE}(\bar{y}_{reg}) = \sqrt{\left(1 - \frac{n}{N}\right)\frac{s_e^2}{n}} = \sqrt{\left(1 - \frac{31}{2967}\right)\frac{18.08}{31}} = 0.76$$

```
output_reg <- res_reg_mean$estimate
output_reg |> format_est_gt("reg_mean")
```

$\bar{y}_{reg}$	SE($\bar{y}_{reg}$)	95% CI Lower	95% CI Upper
34.49	0.76	32.93	36.04

Output Note: The Regression estimate for the mean volume is 34.486 with a standard error of 0.7597.

5.4.2.2.3 Estimate the total

The estimate of the total follows by passing `estimate = "total"`, which multiplies the estimated mean and its standard error by N.

```
reg_total_volume_raw <- reg_est(ydata, xdata, xbarU = xbarU, N = N, estimate =
 ↪  "total")$estimate
reg_total_volume_raw |> format_est_gt("reg_total")
```

$\hat{t}_{reg}$	SE($\hat{t}_{reg}$)	95% CI Lower	95% CI Upper
102,318.86	2,253.97	97,708.98	106,928.74

5.4.2.2.4 Percentage of Variance Reduction

As for the ratio estimator, we compare the standard error of the regression estimate with that of the SRS estimate:

```
reg_mean_volume_raw <- output_reg
var_reduction_reg <- 1 - (reg_mean_volume_raw["S.E."] / srs_mean_vol_raw["S.E."])^2
unname(var_reduction_reg)
```

```
[1] 0.9330895
```

Output Note: The variance reduction for the regression estimator is 93.3%, much larger than the 60.6% of the ratio estimator. The fitted intercept, $\hat{B}_0 = -36.94$, is far from zero: volume grows faster than in proportion to diameter, so a line through the origin fits poorly, while a straight line with an intercept fits well over the range of the data.

5.4.3 Example: Photo Counts of Dead Trees

Aerial photographs are a cheap way to survey a large area (for example a caribou herd), but they tend to **undercount** animals that are hidden from view overhead. A more expensive, intensive ground count on a subsample of plots gives accurate counts that can calibrate the photo counts. Let x_i be the photo count and y_i the field count on plot i. Fitting a ratio or a regression line on the subsample of plots that have both counts lets us correct the total photo count of the *entire* surveyed area, for example $\hat{t}_{yr} = \hat{B}\, t_x$, for the systematic undercounting bias.

To estimate the number of dead trees in an area, we divide the area into 100 square plots and count dead trees on photographs. We select an SRS of 25 plots for exact field counts. We know the true mean photo count is 11.3.

The photo counts and the field counts of the 25 sampled plots are entered below, and a straight line of field counts on photo counts is fitted.

```
photocounts <- c(10, 12, 7, 13, 13, 6, 17, 16, 15, 10, 14, 12, 10, 5, 12, 10,
                 10, 9, 6, 11, 7, 9, 11, 10, 10)

fieldcounts <- c(15, 14, 9, 14, 8, 5, 18, 15, 13, 15, 11, 15, 12, 8, 13,
                 9, 11, 12, 9, 12, 13, 11, 10, 9, 8)

lmfit <- lm(fieldcounts ~ photocounts)
lmfit_gt <- format_lm_gt(lmfit)
lmfit_gt$coef
```

Coefficient Estimates

	Estimate	S.E.	t value	p value
(Intercept)	5.0593	1.7635	2.8689	0.0087
photocounts	0.6133	0.1601	3.8316	0.0009

```
lmfit_gt$fit
```

Model Fit Summary

R^2	Adj. R^2	$\hat{\sigma}$	F-statistic	p-value
0.3896	0.3631	2.4062	14.6815	0.0009

The scatter plot shows the relationship between the two counts, with the fitted regression line.

```
plot(photocounts, fieldcounts, main="Field Counts vs Photo Counts")
abline(lmfit, col="purple", lwd=2)
```

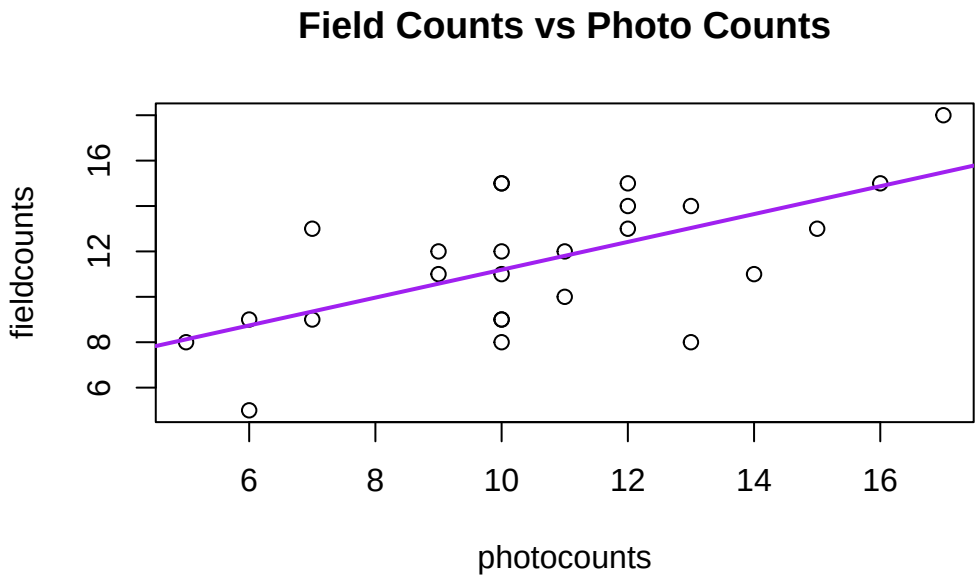

Figure 5.9: Field counts vs. photo counts of dead trees with the fitted regression line

Let's compare the three different estimation methods to see which provides the lowest Standard Error for the population mean.

```
# 1. Regression estimate for the mean
res_dt_reg <- reg_est(ydata = fieldcounts, xdata = photocounts, xbarU = 11.3, N = 100, estimate
 ↪  = "mean")
res_dt_reg$table
```

Regression Estimation: Working Table
$n = 25$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	15.00	10.00	11.19	3.81	14.50
2	14.00	12.00	12.42	1.58	2.50
3	9.00	7.00	9.35	-0.35	0.12
4	14.00	13.00	13.03	0.97	0.94
5	8.00	13.00	13.03	-5.03	25.32
6	5.00	6.00	8.74	-3.74	13.98
...	...	...	...	...	...
Sum	**289.00**	**265.00**	**289.00**	**0.00**	**133.16**

[1] $\hat{B}_0 = 5.06$, $\hat{B}_1 = 0.61$, $\hat{y}_i = \hat{B}_0 + \hat{B}_1 x_i$, $s_e^2 = 5.79$.

$\bar{y}_{reg} = \hat{B}_0 + \hat{B}_1 \bar{x}_U = 5.06 + 0.61 \times 11.30 = 11.99$

$$\text{SE}(\bar{y}_{reg}) = \sqrt{\left(1 - \frac{n}{N}\right)\frac{s_e^2}{n}} = \sqrt{\left(1 - \frac{25}{100}\right)\frac{5.79}{25}} = 0.42$$

```
dt_reg_raw <- res_dt_reg$estimate
dt_reg_raw |> format_est_gt("reg_mean")
```

$\bar{y}_{reg}$	SE($\bar{y}_{reg}$)	95% CI Lower	95% CI Upper
11.99	0.42	11.13	12.85

```
# (And estimating the total)
reg_est(ydata = fieldcounts, xdata = photocounts, xbarU = 11.3, N = 100, estimate =
 ↪  "total")$estimate |>
  format_est_gt("reg_total")
```

$\hat{t}_{reg}$	SE($\hat{t}_{reg}$)	95% CI Lower	95% CI Upper
1,198.93	41.68	1,112.72	1,285.14

```
# 2. Simple random sampling estimate for the mean (Ignoring photo counts)
res_dt_srs <- srs_est(sdata = fieldcounts, N = 100, estimate = "mean", show.details = FALSE)
dt_srs_raw <- res_dt_srs$estimate
dt_srs_raw |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
11.56	0.52	10.48	12.64

```
# 3. Ratio estimate for the mean
res_dt_ratio <- ratio_est(ydata = fieldcounts, xdata = photocounts, xbarU = 11.3, N = 100,
 ↪  estimate = "mean")
res_dt_ratio$table
```

Ratio Estimation: Working Table
$n = 25$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	15.00	10.00	10.91	4.09	16.76
2	14.00	12.00	13.09	0.91	0.83
3	9.00	7.00	7.63	1.37	1.87
4	14.00	13.00	14.18	-0.18	0.03
5	8.00	13.00	14.18	-6.18	38.16
6	5.00	6.00	6.54	-1.54	2.38
...	...	...	...	...	...
Sum	**289.00**	**265.00**	**289.00**	**0.00**	**184.64**

[1] $\hat{B} = \sum_i y_i / \sum_i x_i = 1.09$, $\hat{y}_i = \hat{B}\, x_i$, $s_e^2 = 7.69$.
$\bar{y}_r = \hat{B}\, \bar{x}_U = 1.09 \times 11.30 = 12.32$
$\mathrm{SE}(\bar{y}_r) = \mathrm{SE}(\hat{B})\, \bar{x}_U = 0.05 \times 11.30 = 0.51$

```
dt_ratio_raw <- res_dt_ratio$estimate
dt_ratio_raw |> format_est_gt("ratio_mean")
```

$\bar{y}_r$	$\mathrm{SE}(\bar{y}_r)$	95% CI Lower	95% CI Upper
12.32	0.51	11.27	13.38

Output Note:

- **SRS Mean:** SE = 0.522
- **Ratio Mean:** SE = 0.512
- **Regression Mean:** SE = 0.417

The sampled plots average $\bar{x} = 10.6$ photo-counted trees, below the population average $\bar{x}_U = 11.3$, so both the ratio and the regression estimates are above the plain sample mean $\bar{y} = 11.56$. The ratio estimator barely improves on SRS, since a line through the origin fits these data poorly. Because the intercept of the true line is non-zero, the **regression estimator** provides the most precise estimate in this scenario.

5.4.4 Shinylive App Illustrating Ratio Estimation and Regression Estimation

The simulation for this section runs as a standalone app: Ratio Estimation and Regression Estimation. It opens in a new tab, together with its documentation and the full list of apps.

5.5 Post-stratification

5.5.1 Theory

5.5.1.1 The post-stratified estimator

Sometimes a unit's stratum is unknown until *after* sampling, for instance a person's gender before the interview. We cannot then stratify in advance, but if the true population sizes N_h of the H groups are known from another source, we can still take an SRS, sort the units into groups afterwards, and weight each group mean by its *known* population share $\pi_h = N_h/N$, as in stratified sampling:

$$\boxed{\bar{y}_{post} = \sum_{h=1}^{H} \pi_h \, \bar{y}_h, \qquad \hat{t}_{post} = N \bar{y}_{post}}. \tag{5.25}$$

Compare this with the unweighted mean $\bar{y} = \sum_h (n_h/n)\, \bar{y}_h$, which weights the groups by the shares the sample happened to deliver. When the sample over- or under-represents a group, post-stratification corrects for it.

It also has a predictive reading, like the ratio and regression estimators. Every unit in group h is predicted by that group's sample mean, $\hat{y}_i = \bar{y}_h$ for $i \in \mathcal{U}_h$, so

$$\hat{t}_{post} = \sum_{i \in \mathcal{U}} \hat{y}_i = \sum_{h=1}^{H} N_h \, \bar{y}_h, \qquad \hat{\bar{y}}_{post} = \sum_{h=1}^{H} \pi_h \, \bar{y}_h. \tag{5.26}$$

Figure 5.10 illustrates the benefit. The grey points are the population, split into post-strata A to D (the width of each block is N_h). The green circles are an ordinary SRS of $n = 25$; the realized $n_h = (5, 5, 6, 9)$ do *not* match the population shares $\pi_h = (.36, .14, .21, .29)$, and they never do, since n_h is random. The red segments are the predictions $\hat{y}_i = \bar{y}_h$ in each stratum, whose weighted average is $\hat{\bar{y}}_{post}$.

```
plot_poststrat_concept()
```

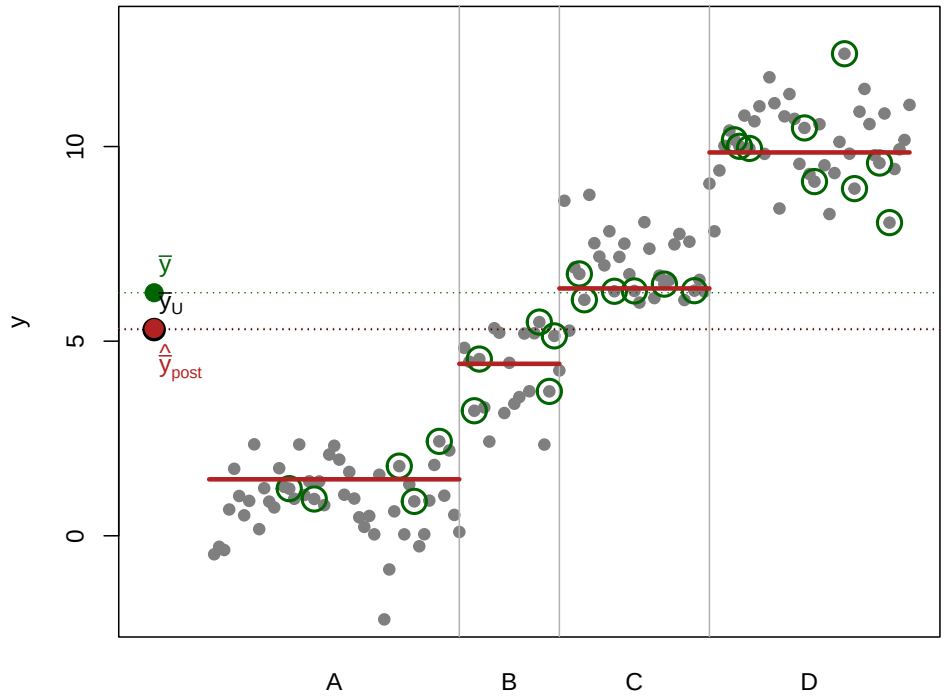

Figure 5.10: Illustration of the benefits of post-stratification.

This sample is light on A and heavy on D, so the naive $\bar{y}$ (green) is pulled up; reweighting by the known π_h undoes it, and $\hat{\bar{y}}_{post}$ (red) is close to $\bar{y}_U$ (black). It is this reweighting, repeated over samples, that cuts the variance.

5.5.1.2 Variance of the post-stratified mean

The observed group sizes n_h are random, so for the variance we substitute their expected values under SRS, the proportional sample sizes

$$n_h^{post} = n\,\frac{N_h}{N} = n\pi_h, \tag{5.27}$$

into the stratified variance formula. Every group then has sampling fraction $n_h^{post}/N_h = n/N$, so the finite population correction is the same in all groups and the variance simplifies:

$$\boxed{\hat{V}(\bar{y}_{post}) = \sum_{h=1}^{H} \pi_h^2 \left(1 - \frac{n_h^{post}}{N_h}\right) \frac{s_h^2}{n_h^{post}} = \left(1 - \frac{n}{N}\right) \frac{1}{n} \sum_{h=1}^{H} \pi_h s_h^2}, \tag{5.28}$$

the variance of proportional stratified sampling: the SRS variance with s^2 replaced by the weighted average of the within-group variances. Also $\hat{V}(\hat{t}_{post}) = N^2 \hat{V}(\bar{y}_{post})$. Here n_h, $\bar{y}_h$ and s_h^2 are the count, mean and variance of the units the sample happened to place in group h, so every group needs $n_h \geq 2$ for s_h^2 to exist. Substituting n_h^{post} treats the n_h as fixed at $n\pi_h$; carrying their randomness adds a second-order term $\frac{1}{n^2}\sum_h (1-\pi_h)s_h^2$, negligible unless some n_h are small.

5.5.1.3 Post-stratification as a regression estimator

Post-stratification is not a separate technique: it is the regression estimator of Section 5.4.1 with the **group indicators** as the auxiliary variables. Take the H dummies $x_{ih} = \mathbf{1}\{i \in \text{group } h\}$ and fit without an intercept. Then

- the fitted coefficients are the group means, $\hat{B}_h = \bar{y}_h$;
- the population means of those dummies are the known weights, $\bar{x}_{U,h} = N_h/N = \pi_h$;

so the regression estimator *is* the post-stratified one,

$$\hat{\bar{y}}_{reg} = \sum_h \hat{B}_h \, \bar{x}_{U,h} = \sum_h \pi_h \bar{y}_h = \hat{\bar{y}}_{post}. \tag{5.29}$$

Its residuals are each unit's deviation from its own group mean, $e_i = y_i - \bar{y}_{h(i)}$, and the regression variance $\left(1 - \frac{n}{N}\right) s_e^2/n$ applies with $p = H$ fitted parameters:

$$s_e^2 = \frac{1}{n-H} \sum_h \sum_{i \in h} (y_i - \bar{y}_h)^2 = \sum_h \frac{n_h - 1}{n-H} \, s_h^2. \tag{5.30}$$

Dropping the -1 and the $-H$ (small relative to n_h and n), and then using $n_h/n \approx \pi_h$, the regression variance reduces to the post-stratified one:

$$\begin{aligned} \hat{V}(\hat{\bar{y}}_{reg}) &= \left(1 - \frac{n}{N}\right) \frac{1}{n} \sum_{h=1}^{H} \frac{n_h - 1}{n-H} \, s_h^2 \\ &\approx \left(1 - \frac{n}{N}\right) \frac{1}{n} \sum_{h=1}^{H} \frac{n_h}{n} \, s_h^2 \\ &\approx \left(1 - \frac{n}{N}\right) \frac{1}{n} \sum_{h=1}^{H} \pi_h \, s_h^2 = \hat{V}(\hat{\bar{y}}_{post}). \end{aligned} \tag{5.31}$$

Both are weighted averages of the *same* s_h^2; only the weights differ,

$$\underbrace{\frac{n_h - 1}{n-H}}_{\text{sample proportions}} \qquad \text{versus} \qquad \underbrace{\pi_h = \frac{N_h}{N}}_{\text{population proportions}} \quad . \tag{5.32}$$

So the substitution $n_h \to n\pi_h$ above is simply: *pool the residuals with the weights we know, not with the ones the sample happened to deliver.* It is the same reasoning that puts $\bar{x}_U$, rather than $\bar{x}$, in the ratio estimator.

Seen this way, the estimators of this chapter are one estimator with different auxiliary variables, and $s_e^2 = \frac{1}{n-p} \sum_i e_i^2$ throughout, where p is the number of fitted parameters:

Estimator	Auxiliary variables	Residual e_i	p
Ratio	one x, line forced through the origin	$y_i - \hat{B}x_i$	1
Regression	intercept and x	$y_i - \hat{B}_0 - \hat{B}_1 x_i$	2
Post-stratification	H group indicators	$y_i - \bar{y}_h$	H

Domain estimation, treated next, stands slightly apart: there the divisor is the random n_d rather than the full n, because only the units falling in the domain carry information about it.

5.5.2 Example: A Nurses Survey (by hand)

A survey of $N = 10{,}000$ people, half women and half men, takes an SRS of $n = 400$ and asks whether they want to become a nurse. The sample has three times as many women as men, though the population is 50/50:

Stratum	N_h	n_h	Want nurse	$\hat{p}_h$
Female	5000	300	150	0.50
Male	5000	100	10	0.10

Ignoring this pulls the estimate $\hat{p}_{SRS} = \frac{160}{400} = (0.5)(0.75) + (0.1)(0.25) = 0.4$ toward the female (higher) rate. Weighting by the *known* population shares instead gives

$$\hat{p}_{post} = \hat{p}_f \frac{N_f}{N} + \hat{p}_m \frac{N_m}{N} = (0.5)(0.5) + (0.1)(0.5) = 0.3. \tag{5.33}$$

For the standard error, $\hat{p}_h$ plays the role of $\bar{y}_h$, and $s_h^2 = \frac{n_h}{n_h - 1}\hat{p}_h(1 - \hat{p}_h)$ from the realized counts. With the working sample sizes $n_h^{post} = n\pi_h = 200$ in each stratum (proportional to size allocation), every finite population correction is $1 - n/N = 0.96$:

Post-stratum	N_h	π_h	n_h^{post}	$\hat{p}_h$	s_h^2	$\pi_h \hat{p}_h$	$\hat{v}_h$
Female	5000	0.5	200	0.500	0.2508	0.2500	0.000301
Male	5000	0.5	200	0.100	0.0909	0.0500	0.000109
Total	**10000**	**1.0**	**400**			**0.3000**	**0.000410**

where $\hat{v}_h = \left(1 - n_h^{post}/N_h\right) \pi_h^2 \, s_h^2 / n_h^{post}$. The short-cut formula gives the same variance without the working sample sizes, using $\sum_h \pi_h s_h^2 = 0.5(0.2508) + 0.5(0.0909) = 0.1709$:

$$\hat{V}(\hat{p}_{post}) = \left(1 - \frac{n}{N}\right) \frac{1}{n} \sum_h \pi_h s_h^2 = \frac{0.96}{400}(0.1709) = 0.000410. \tag{5.34}$$

So $\text{SE}(\hat{p}_{post}) = \sqrt{0.000410} = 0.0203$ and an approximate 95% interval is $0.300 \pm 1.96(0.0203) = [0.260,\ 0.340]$. Treating the data as an unstratified SRS gives $\hat{p}_{SRS} = 0.4$ with $\text{SE} = \sqrt{\left(1 - \frac{400}{10000}\right)\frac{(0.4)(0.6)}{399}} \approx 0.0240$. Post-stratification is both unbiased and more precise, since it removes the between-sex variability from the standard error.

5.5.3 Example: agsrs_syn.csv

Any variable known for **every** unit of the population can be used to post-stratify an SRS. We use `agsrs_syn.csv`, a simple random sample of $n = 300$ counties drawn without replacement from the population in `agpop.csv` (the script `data/make_agsrs_syn.R` shows how), and compare two post-stratifiers: the census **region**, and **quantile groups of `acres87`** (the 1987 acreage, as in the post-stratification app). The population is the $N = 3041$ counties with no missing value in `acres92`, `acres87`, `acres82` or `farms92`.

5.5.3.1 By Region

If we know the true regional population sizes (N_h), we can post-stratify our SRS data.

The population is read from `agpop.csv`, keeping the counties with no missing value, and the sample from `agsrs_syn.csv`; acreages are converted to thousands of acres.

```
## the population, and the sample drawn from it (acres in thousands)
agpop_ps <- acres_in_thousands(read.csv("data/agpop.csv"))
agpop_ps <- agpop_ps[complete.cases(agpop_ps[, c("acres92", "acres87", "acres82", "farms92")]),
 ↪  ]
N <- nrow(agpop_ps)
agsrs_syn <- acres_in_thousands(read.csv("data/agsrs_syn.csv"))
```

From the sample we need the observed group sizes n_h, the group means $\bar{y}_h$ and the group standard deviations s_h. The regional population sizes N_h are the external information, taken from the population.

```
nh <- tapply(agsrs_syn[, "acres92"], agsrs_syn[,"region"], length)
n <- sum(nh)
sh <- tapply(agsrs_syn[, "acres92"], agsrs_syn[,"region"], sd)
ybarh <- tapply(agsrs_syn[, "acres92"], agsrs_syn[,"region"], mean)
# external information: the regional population sizes
Nh <- tapply(agpop_ps$acres92, agpop_ps$region, length)
```

Create $n_h^{\text{post-strat}} = n \times \frac{N_h}{N}$, instead of using the randomly observed n_h.

```
nh_post <- Nh / sum(Nh) * n

## regional summary for stratified sampling estimation
data.frame(
  ybarh = ybarh,
  sh = sh,
  nh = nh,
  pi_obs = nh/n,
  Nh = Nh,
  pi_pop = Nh/sum(Nh),
  nh_post = nh_post
) |>
  gt(rownames_to_stub = TRUE) |>
```

Table 5.2: Observed vs. population stratum proportions and post-stratified sample sizes

	$\bar{y}_h$	s_h	n_h^{obs}	π_h^{obs}	N_h	π_h^{pop}	n_h^{post}
NC	355.048	260.114	110	0.367	1049	0.345	103.486
NE	62.613	61.156	17	0.057	211	0.069	20.816
S	201.874	197.338	126	0.420	1368	0.450	134.956
W	613.793	691.154	47	0.157	413	0.136	40.743

```
  cols_label(
    ybarh = md("$\\bar{y}_h$"),
    sh = md("$s_h$"),
    nh = md("$n_h^{\\text{obs}}$"),
    pi_obs = md("$\\pi^{\\text{obs}}_h$"),
    Nh = md("$N_h$"),
    pi_pop = md("$\\pi^{\\text{pop}}_h$"),
    nh_post = md("$n_h^{\\text{post}}$")
  ) |>
  fmt_number(columns = c(ybarh, sh, pi_obs, pi_pop, nh_post), decimals = 3)
```

Output Note: The table above shows the difference between the sample proportion (π_h^{obs}) and the true population proportion (π_h^{pop}). Post-stratification forces the weights to reflect the true population structure.

The post-stratified estimate comes from `str_est()`, applied with the working sample sizes n_h^{post}. For comparison we also compute the unadjusted SRS estimate, and the true mean of the population.

```
## strata mean estimate
res_post_strat <- str_est(ybarh, sh, nh_post, Nh, estimate = "mean")
res_post_strat$table
```

Stratified Mean Estimation: Working Table

Stratum	N_h	n_h	π_h [1]	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	v_h [2]
NC	1,049	103	0.34	355.05	67,659.15	122.47	70.12
NE	211	21	0.07	62.61	3,740.00	4.34	0.78
S	1,368	135	0.45	201.87	38,942.48	90.81	52.63
W	413	41	0.14	613.79	477,694.47	83.36	194.92
Total	**3,041**	**300**	**1.00**			**300.99**	**318.46**

[1] $\pi_h = N_h/N$ is the stratum weight.
[2] $v_h = (1 - n_h/N_h)\,\pi_h^2\, s_h^2/n_h$ is stratum h's contribution to $V(\bar{y}_{str}) = \sum_h v_h$.
$\bar{y}_{str} = \sum_h \pi_h \bar{y}_h = 300.99$
$\text{SE}(\bar{y}_{str}) = \sqrt{\sum_h v_h} = \sqrt{318.46} = 17.85$

```
post_strat_est <- res_post_strat$estimate
post_strat_est |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
300.99	17.85	266.02	335.97

```
## compare with SRS estimate
res_srs_est <- srs_est(agsrs_syn$acres92, N = N, estimate = "mean", show.details = FALSE)
srs_est_raw <- res_srs_est$estimate
srs_est_raw |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
314.68	20.38	274.57	354.79

```
## true mean check
mean(agpop_ps$acres92)
```

```
[1] 309.9421
```

Output Note: The post-stratified estimate gives a Standard Error of 17.8, which is more precise than the unadjusted SRS standard error of 20.4.

The same standard error follows from the short-cut formula $\hat{V} = (1 - \frac{n}{N}) \frac{1}{n} \sum_h \pi_h s_h^2$ of Section 5.5.1, which needs only the population shares π_h and the group variances, not the working sample sizes. The working table lists $\bar{y}_h$, s_h^2 and π_h, and the two working columns $\pi_h \bar{y}_h$ and $\pi_h s_h^2$, whose totals are $\bar{y}_{post}$ and $\sum_h \pi_h s_h^2$:

```
## working table for the short-cut formula, given the group sizes and the group means and sds
shortcut_table <- function(Nh, ybarh, sh) {
  work <- data.frame(Nh = as.vector(Nh), pi_h = as.vector(Nh) / sum(Nh),
                     ybarh = as.vector(ybarh), s2h = as.vector(sh)^2,
                     row.names = names(Nh))
  work$pi_ybar <- work$pi_h * work$ybarh      # working column: pi_h * ybar_h
  work$pi_s2   <- work$pi_h * work$s2h        # working column: pi_h * s_h^2
  rbind(work, Total = c(sum(work$Nh), sum(work$pi_h), NA, NA,
                        sum(work$pi_ybar), sum(work$pi_s2)))
}

work_reg <- shortcut_table(Nh, ybarh, sh)
gt(work_reg, rownames_to_stub = TRUE) |>
  cols_label(Nh = md("$N_h$"), pi_h = md("$\\pi_h$"), ybarh = md("$\\bar{y}_h$"),
             s2h = md("$s_h^2$"), pi_ybar = md("$\\pi_h\\bar{y}_h$"),
             pi_s2 = md("$\\pi_h s_h^2$")) |>
  fmt_number(columns = c(pi_h, ybarh, pi_ybar), decimals = 3) |>
  fmt_number(columns = c(Nh, s2h, pi_s2), decimals = 0) |>
  sub_missing(missing_text = "") |>
```

Table 5.3: Short-cut formula working table: post-stratification by region

	N_h	π_h	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	$\pi_h s_h^2$
NC	1,049	0.345	355.048	67,659	122.475	23,339
NE	211	0.069	62.613	3,740	4.344	260
S	1,368	0.450	201.874	38,942	90.813	17,518
W	413	0.136	613.793	477,694	83.360	64,876
Total	**3,041**	**1.000**			**300.992**	**105,993**

```
  tab_style(style = cell_text(weight = "bold"),
            locations = cells_body(rows = "Total"))
```

The variance, the standard error and the estimate then follow from the totals of the table, and we compare the standard error with the one from `str_est()`:

```
sum_pi_s2 <- work_reg["Total", "pi_s2"]
V_short <- (1 - n / sum(Nh)) / n * sum_pi_s2
c(ybar_post = work_reg["Total", "pi_ybar"], SE_shortcut = sqrt(V_short), SE_str_est =
 ↪  post_strat_est[["S.E."]])
```

```
  ybar_post SE_shortcut  SE_str_est
  300.99224    17.84532    17.84532
```

5.5.3.2 By Quantile Groups of `acres87`

The 1987 acreage is recorded for every county, so the population can be cut into groups by its quantiles. As in the app, the cut points are the 25th, 75th and 95th percentiles of the population `acres87`, giving four groups $Q_1, \ldots, Q_4$ of very different sizes.

```
## population cut points at the 25%, 75% and 95% quantiles of acres87
brk <- quantile(agpop_ps$acres87, probs = c(0, 0.25, 0.75, 0.95, 1))
round(brk, 1)
```

```
    0%    25%    75%    95%   100%
   0.0   88.6  374.8 1015.3 7687.5
```

```
## attach the new grouping variable acres87Q to the population and to the sample
group_of <- function(x) cut(x, breaks = brk, include.lowest = TRUE, labels = paste0("Q", 1:4))
agpop_ps$acres87Q   <- group_of(agpop_ps$acres87)
agsrs_syn$acres87Q  <- group_of(agsrs_syn$acres87)

## known population group sizes, and the sample summaries by group
Nh_q    <- tapply(agpop_ps$acres92,  agpop_ps$acres87Q,  length)
nh_q    <- tapply(agsrs_syn$acres92, agsrs_syn$acres87Q, length)
sh_q    <- tapply(agsrs_syn$acres92, agsrs_syn$acres87Q, sd)
ybarh_q <- tapply(agsrs_syn$acres92, agsrs_syn$acres87Q, mean)
```

Table 5.4: Observed vs. population quantile-group proportions and post-stratified sample sizes (acres87 groups)

	$\bar{y}_h$	s_h	n_h^{obs}	π_h^{obs}	N_h	π_h^{pop}	n_h^{post}
Q1	47.292	25.882	76	0.253	761	0.250	75.074
Q2	206.733	83.565	141	0.470	1520	0.500	149.951
Q3	579.268	165.357	69	0.230	608	0.200	59.980
Q4	1,549.361	632.926	14	0.047	152	0.050	14.995

The table compares the observed group proportions with the population ones and gives the working sample sizes.

```
n <- sum(nh_q)
nh_post_q <- Nh_q / sum(Nh_q) * n

data.frame(
  ybarh = ybarh_q,
  sh = sh_q,
  nh = nh_q,
  pi_obs = nh_q / n,
  Nh = Nh_q,
  pi_pop = Nh_q / sum(Nh_q),
  nh_post = nh_post_q
) |>
  gt(rownames_to_stub = TRUE) |>
  cols_label(
    ybarh = md("$\\bar{y}_h$"),
    sh = md("$s_h$"),
    nh = md("$n_h^{\\text{obs}}$"),
    pi_obs = md("$\\pi^{\\text{obs}}_h$"),
    Nh = md("$N_h$"),
    pi_pop = md("$\\pi^{\\text{pop}}_h$"),
    nh_post = md("$n_h^{\\text{post}}$")
  ) |>
  fmt_number(columns = c(ybarh, sh, pi_obs, pi_pop, nh_post), decimals = 3)
```

Apply `str_est()` with the working sample sizes $n_h^{\text{post}} = n\,N_h/N$ in place of the observed n_h:

```
res_quantile <- str_est(ybarh_q, sh_q, nh_post_q, Nh_q, estimate = "mean")
quantile_est <- res_quantile$estimate
quantile_est |> format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
308.43	9.36	290.08	326.77

The short-cut working table for the quantile groups, with the same layout:

Table 5.5: Short-cut formula working table: post-stratification by acres87 quantile groups

	N_h	π_h	$\bar{y}_h$	s_h^2	$\pi_h \bar{y}_h$	$\pi_h s_h^2$
Q1	761	0.250	47.292	670	11.835	168
Q2	1,520	0.500	206.733	6,983	103.333	3,490
Q3	608	0.200	579.268	27,343	115.816	5,467
Q4	152	0.050	1,549.361	400,596	77.443	20,023
Total	**3,041**	**1.000**			**308.425**	**29,148**

```
work_q <- shortcut_table(Nh_q, ybarh_q, sh_q)
gt(work_q, rownames_to_stub = TRUE) |>
  cols_label(Nh = md("$N_h$"), pi_h = md("$\\pi_h$"), ybarh = md("$\\bar{y}_h$"),
             s2h = md("$s_h^2$"), pi_ybar = md("$\\pi_h\\bar{y}_h$"),
             pi_s2 = md("$\\pi_h s_h^2$")) |>
  fmt_number(columns = c(pi_h, ybarh, pi_ybar), decimals = 3) |>
  fmt_number(columns = c(Nh, s2h, pi_s2), decimals = 0) |>
  sub_missing(missing_text = "") |>
  tab_style(style = cell_text(weight = "bold"),
            locations = cells_body(rows = "Total"))
```

The variance, the standard error and the estimate follow from the totals, as before, and agree with `str_est()`:

```
sum_pi_s2_q <- work_q["Total", "pi_s2"]
V_short_q <- (1 - n / sum(Nh_q)) / n * sum_pi_s2_q
c(ybar_post = work_q["Total", "pi_ybar"], SE_shortcut = sqrt(V_short_q), SE_str_est =
 ↪  quantile_est[["S.E."]])
```

```
  ybar_post SE_shortcut  SE_str_est
 308.425474    9.358145    9.358145
```

5.5.4 Comparing the Two Post-stratifications

Finally we put the three estimates, the unadjusted SRS and the two post-stratified ones, side by side.

```
compare_est <- rbind(
  "SRS (no post-stratification)" = srs_est_raw[c("Est.", "S.E.")],
  "Post-stratified by region"    = post_strat_est[c("Est.", "S.E.")],
  "Post-stratified by acres87"   = quantile_est[c("Est.", "S.E.")]
)
compare_est |>
  as.data.frame() |>
  gt(rownames_to_stub = TRUE) |>
  cols_label(Est. = md("$\\bar{y}$"), S.E. = md("$\\mathrm{SE}$")) |>
  fmt_number(columns = everything(), decimals = 2)
```

Table 5.6: Estimates of the mean acres92 and their standard errors: SRS, post-stratified by region, and post-stratified by acres87 quantile groups

	$\bar{y}$	SE
SRS (no post-stratification)	314.68	20.38
Post-stratified by region	300.99	17.85
Post-stratified by acres87	308.43	9.36

Output Note: Both post-stratifications reduce the standard error of the SRS estimate, because the groups differ in mean acreage. The `acres87` groups reduce it much more than the regions, since the 1987 acreage is strongly related to the 1992 acreage, so the groups are far more homogeneous in `acres92`. A good post-stratifier is a variable that is known for the whole population and strongly related to y.

All three estimates are close to the population mean of `acres92`, 309.9, and their 95% intervals cover it.

5.5.5 Interactive Demonstration: Post-stratifying a Simple Random Sample

The simulation for this section runs as a standalone app: Post-stratification of a Simple Random Sample. It opens in a new tab, together with its documentation and the full list of apps.

5.6 Domain Estimation

5.6.1 Theory

5.6.1.1 Domains

Suppose there are D **domains**, subpopulations of interest determined only *after* sampling, such as men and women. Let $\mathcal{U}_d$ be the population units in domain d and $\mathcal{S}_d$ the sampled units in domain d, with $N_d = |\mathcal{U}_d|$ and $n_d = |\mathcal{S}_d|$. We might want the mean salary for men and for women separately, but we do not know a person's gender before selecting them, so we cannot stratify by domain in advance. Instead we estimate each domain's mean or total after sampling, by splitting the sample by domain ID.

The domain sample size n_d is a random variable: a different SRS would very likely give a different n_d. The domain mean is

$$\bar{y}_d = \frac{1}{n_d} \sum_{i \in \mathcal{S}_d} y_i, \tag{5.35}$$

which *looks* like an ordinary sample mean, but the randomness of n_d is the source of the extra uncertainty.

5.6.1.2 Domain means are ratio estimates

Define

$$x_i = \begin{cases} 1 & i \in \mathcal{U}_d \\ 0 & i \notin \mathcal{U}_d \end{cases} \qquad u_i = y_i x_i = \begin{cases} y_i & i \in \mathcal{U}_d \\ 0 & i \notin \mathcal{U}_d \end{cases} \tag{5.36}$$

Then $t_x = N_d$, $\bar{x}_U = N_d/N$, $\bar{y}_{U_d} = t_u/t_x = B$, $\bar{x} = n_d/n$, and

$$\bar{y}_d = \hat{B} = \frac{\bar{u}}{\bar{x}} = \frac{\hat{t}_u}{\hat{t}_x} = \frac{\sum_{i \in \mathcal{S}_d} y_i}{n_d}. \tag{5.37}$$

Domain mean estimation is therefore a special case of ratio estimation, with $\sum_{i=1}^{n} u_i / \sum_{i=1}^{n} x_i$ as the ratio. This is exactly why it is a ratio estimator and not a simple mean.

5.6.1.3 Standard error of a domain mean

Applying the formula for $\hat{V}(\hat{B})$ with x_i and u_i as defined above gives

$$\boxed{\hat{V}(\bar{y}_d) = \left(1 - \frac{n}{N}\right) \frac{s_d^2}{n_d}, \qquad s_d^2 = \frac{1}{n_d - 1} \sum_{i \in \mathcal{S}_d} (y_i - \bar{y}_d)^2}, \tag{5.38}$$

where S_d is the set of sampled units in domain d, n is the overall sample size, N is the population size, and s_d^2 is the sample variance of the y_i within domain d. (Exactly, the variance is $\left(1 - \frac{n}{N}\right) \frac{n}{n_d^2} \frac{(n_d - 1)s_d^2}{n-1}$, which is approximately the expression above.) Ninety-five percent intervals use $t_{0.975,\, n_d - 1}$.

This is **not** the same as naively applying SRS *within* domain d, as if n_d were fixed, which would use $\left(1 - \frac{n_d}{N_d}\right)$ in place of $\left(1 - \frac{n}{N}\right)$. Only the finite population correction differs: the **overall** rate (correct, since n_d is random) versus the **domain** rate (wrong). They agree only when $n_d/N_d \approx n/N$. The fpc uses the overall sampling fraction n/N also because the domain's population size N_d is usually unknown.

5.6.1.4 Estimating a domain total

- If N_d is **known**: $\hat{t}_{yd} = \bar{y}_d \times N_d$, with $\mathrm{SE}(\hat{t}_{yd}) = \mathrm{SE}(\bar{y}_d) \times N_d$.
- If N_d is **unknown**: estimate it first, $\hat{N}_d = N n_d/n$, and use $\hat{t}_{yd} = \bar{y}_d \times \hat{N}_d = N\bar{u}$, with

$$\mathrm{SE}(\hat{t}_{yd}) = N\,\mathrm{SE}(\bar{u}) = N\sqrt{\left(1 - \frac{n}{N}\right) \frac{s_u^2}{n}}. \tag{5.39}$$

5.6.2 Example: agsrs.csv

5.6.2.1 Estimating Domain Means

When estimating the mean for a specific subgroup (like a region), the sample size in that subgroup (n_d) is a random variable, which increases our uncertainty.

```
agsrs <- acres_in_thousands(read.csv("data/agsrs.csv"))
n <- nrow(agsrs)

## estimating domain means
region_agsrs_domain <- data.frame(rbind(
    NC = domain_est(agsrs$acres92[agsrs$region=="NC"], n=n, N = 3078),
    NE = domain_est(agsrs$acres92[agsrs$region=="NE"], n=n, N = 3078),
    S  = domain_est(agsrs$acres92[agsrs$region=="S"], n=n, N = 3078),
    W  = domain_est(agsrs$acres92[agsrs$region=="W"], n=n, N = 3078)
))

region_agsrs_domain |> format_est_gt("domain_mean")
```

	$\bar{y}_d$	SE($\bar{y}_d$)	95% CI Lower	95% CI Upper
NC	323.42	30.01	263.66	383.17
NE	106.55	28.96	45.45	167.64
S	246.19	23.50	199.77	292.61
W	518.98	70.30	377.21	660.75

If we incorrectly applied the SRS formula within each region, as if its n_d had been fixed in advance, the finite population correction would use the domain rate n_d/N_d instead of the overall rate n/N:

```
## naively applying SRS estimation to each domain (incorrect analysis)
region_agsrs_srs <- data.frame(rbind(
    NC = srs_est(agsrs$acres92[agsrs$region=="NC"], N = 1054, estimate = "mean", show.details
    ↪  = FALSE)$estimate,
    NE = srs_est(agsrs$acres92[agsrs$region=="NE"], N = 220,  estimate = "mean", show.details
    ↪  = FALSE)$estimate,
    S  = srs_est(agsrs$acres92[agsrs$region=="S"],  N = 1382, estimate = "mean", show.details
    ↪  = FALSE)$estimate,
    W  = srs_est(agsrs$acres92[agsrs$region=="W"],  N = 422,  estimate = "mean", show.details
    ↪  = FALSE)$estimate
))

region_agsrs_srs |> format_est_gt("mean")
```

	$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
NC	323.42	30.40	262.89	383.94
NE	106.55	29.21	44.93	168.17
S	246.19	23.26	200.24	292.13
W	518.98	70.03	377.74	660.21

Output Note: The two sets of standard errors differ only through the fpc. The naive SE is larger where the region happened to be **under**-sampled ($n_d/N_d < n/N$) and smaller where it was **over**-sampled ($n_d/N_d > n/N$). The correct formula ignores this sampling luck and always uses the overall rate n/N. With $n/N \approx 0.1$ here, the differences are small.

5.6.3 Example: Teacher Aspirations (teacher data)

5.6.3.1 Importing Data and Calculating Contingency Table

The teacher data record, for each student, the gender and whether they want to become a teacher. We tabulate the two variables and compute the proportion within each gender, the two domains.

```
teacher <- read.csv("data/college_teacher.csv")

table(teacher[, 2:1])
```

```
      teacher
gender   0   1
     1 156  84
     2 120  40
```

```
addmargins(table(teacher[, 2:1]), 2)
```

```
      teacher
gender   0   1 Sum
     1 156  84 240
     2 120  40 160
```

```
# Proportions of students who want to become a teacher in two domains (female and male)
prop.table(table(teacher[, 2:1]), margin = "gender")
```

```
      teacher
gender    0    1
     1 0.65 0.35
     2 0.75 0.25
```

5.6.3.2 Estimating Domain Means

Treating gender as a domain, with known sizes N_d of 3000 and 1000 out of $N = 4000$, `domain_est()` estimates each domain mean using the overall sampling fraction n/N in the variance.

```
Nh <- c(3000, 1000)
n <- nrow(teacher)
domain_teacher_est <- data.frame(rbind(
    female = domain_est(subset(teacher, gender==1)$teacher, n=n, N=4000),
    male   = domain_est(subset(teacher, gender==2)$teacher, n=n, N=4000)
))

domain_teacher_est |> format_est_gt("domain_mean")
```

	$\bar{y}_d$	SE($\bar{y}_d$)	95% CI Lower	95% CI Upper
female	0.35	0.03	0.29	0.41
male	0.25	0.03	0.19	0.31

5.7 Prediction with Machine-Learning Models

The ratio, regression and post-stratified estimators all fill in a prediction $\hat{y}_i$ for every unit of the population and average them. Each assumes a specific form for $E[y \mid x]$: a line through the origin, a straight line, or a constant within each group. More generally, we can fit **any** prediction model $\hat{m}(\cdot)$ to the sample (a smoother, a regression tree, a random forest, a neural network, …) and predict every population unit:

$$\hat{y}_i = \hat{m}(x_i), \quad i = 1, \dots, N, \qquad \boxed{\hat{\bar{y}}_{\text{pred}} = \frac{1}{N}\sum_{i=1}^{N} \hat{y}_i, \qquad \hat{t}_{y,\text{pred}} = \sum_{i=1}^{N} \hat{y}_i.} \tag{5.40}$$

The ratio and regression estimators are the special cases $\hat{m}(x) = \hat{B}x$ and $\hat{m}(x) = \hat{B}_0 + \hat{B}_1 x$.

In Figure 5.11 the true relationship is curved. A straight line (blue, dashed) underfits it, but a flexible loess curve (red) fitted to an SRS of 15 units tracks it closely, and the average of its predictions, $\hat{\bar{y}}_{\text{ML}}$, lands much closer to $\bar{y}_U$ than the sample mean.

Two practical points. Because $\hat{m}$ may be nonlinear, the estimator needs the value x_i of **every** population unit, not just $\bar{x}_U$. And there is usually no simple formula for its standard error, which is then estimated by the bootstrap or by cross-validation of the residuals.

5.8 Simulation study with agpop.csv

Let's test this practically by drawing 2000 random samples and seeing how SRS and Ratio estimators behave under repeated sampling.

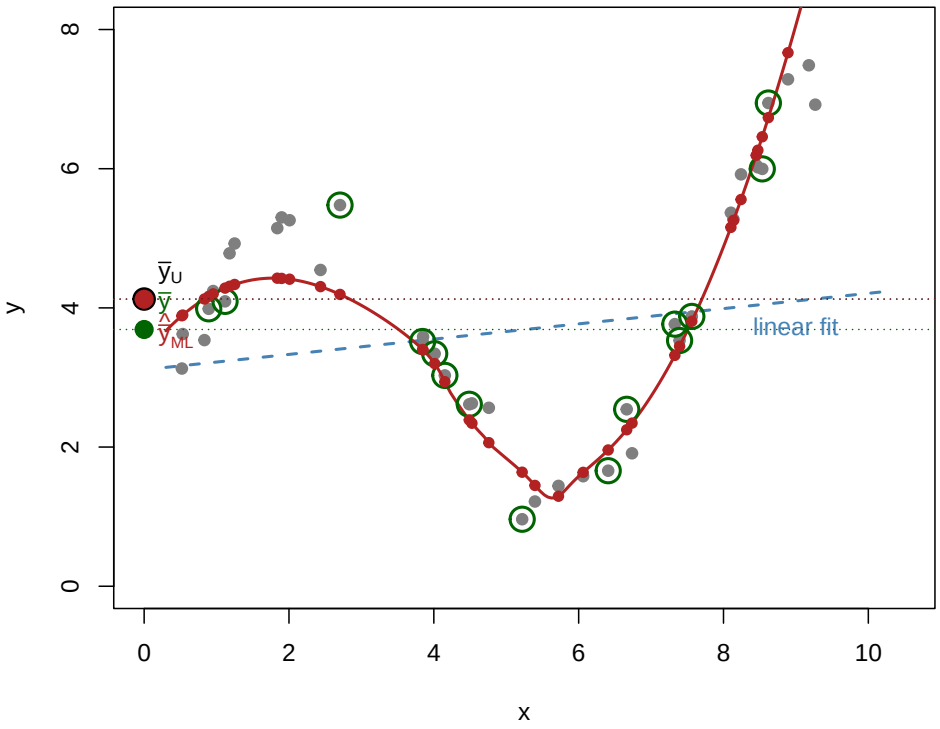

Figure 5.11: A prediction estimator with a flexible model. Grey: the population (x_i known). Green circles: an SRS. Red: a loess curve fitted to the sample and its predictions for every unit; blue dashed: a straight-line fit. Left margin: $\bar{y}_U$ (black), the sample mean (green) and the mean of the predictions (red).

5.8.1 Information of population data

We read the population, drop the counties missing `acres92` or `acres87`, and compute the true total t_y of 1992 acreage, the known total t_x of 1987 acreage and the true ratio $B = t_y/t_x$ (acreages in thousands).

```
agpop <- acres_in_thousands(read.csv("data/agpop.csv"))
## acres87 is the auxiliary variable, so drop counties missing either one
agpop <- agpop[!is.na(agpop$acres92) & !is.na(agpop$acres87), ]

# sample size
n <- 300
# population size
N <- nrow(agpop)

# true values that we want to estimate
tyU <- sum(agpop[, "acres92"])
# suppose known for ratio estimate
txU <- sum(agpop[, "acres87"])
B <- tyU / txU
```

The scatter plot of 1992 against 1987 acreage, with the line through the origin of slope B, shows how well the earlier acreage predicts the later one.

```
plot(agpop[, "acres87"], agpop[, "acres92"], main="Acres 1992 vs Acres 1987")
abline(a = 0, b = B, col="red")
```

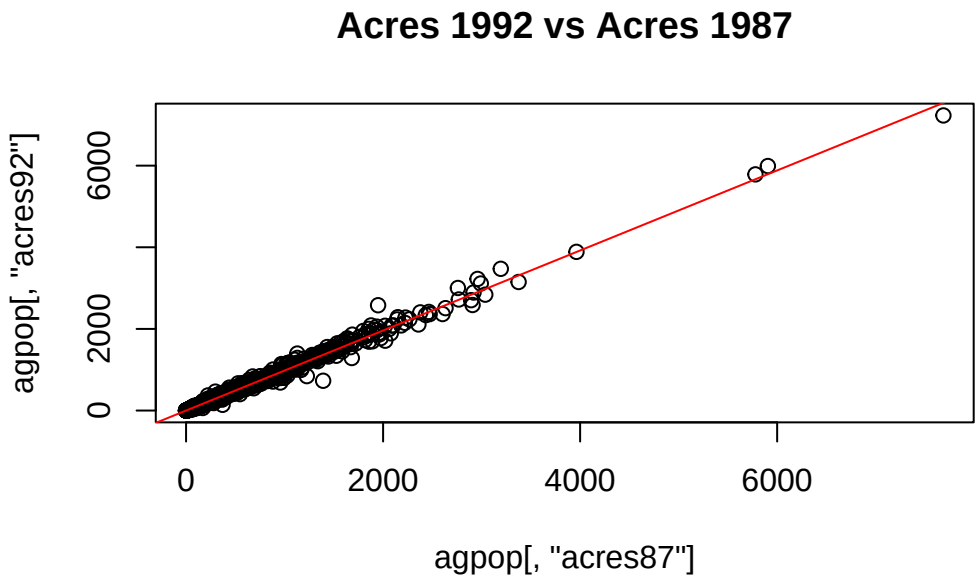

Figure 5.12: 1992 vs. 1987 county acreage with the true ratio line

We fit the population line and compute the expected variance reduction $1 - S_d^2/S_y^2$ from the population residuals $d_i = y_i - Bx_i$ of the theory section.

```
# linear model output
lm_gt <- format_lm_gt(lm(agpop$acres92 ~ agpop$acres87))
lm_gt$coef
```

Coefficient Estimates

	Estimate	S.E.	t value	p value
(Intercept)	-2.4509	0.8517	-2.8775	0.0040
agpop$acres87	0.9882	0.0016	618.4281	0.0000

```
lm_gt$fit
```

Model Fit Summary

R^2	Adj. R^2	$\hat{\sigma}$	F-statistic	p-value
0.9921	0.9921	37.8382	382,453.3759	0.0000

```
# expected reduction of variance of ratio estimate to srs
exp_var_red <- 1 - var(agpop$acres92 - B * agpop$acres87) / var(agpop$acres92)
exp_var_red
```

```
[1] 0.9920478
```

Output Note: Past acreage (1987) strongly predicts current acreage (1992). The theoretically expected variance reduction is roughly 99.2%.

5.8.2 Simulation Studies

Each of the 2000 simulations draws an SRS of $n = 300$ counties and computes two estimates of the total 1992 acreage: the SRS estimate $N\bar{y}$ and the ratio estimate $\hat{B}\,t_x$. The boxplot compares their sampling distributions around the true total (red line).

```
nsim <- 2000
sim_rat <- data.frame(simple = rep(0, nsim), ratio = rep(0, nsim), B = rep(0, nsim))

for (i in 1:nsim) {
  srs <- sample(N, n)
  y_srs <- agpop[srs, "acres92"]
  x_srs <- agpop[srs, "acres87"]

  # SRS estimate (Total)
  sim_rat$simple[i] <- mean(y_srs) * N
  # ratio estimate (Total)
```

```
  sim_rat$B[i] <- mean(y_srs) / mean(x_srs)
  sim_rat$ratio[i] <-  sim_rat$B[i] * txU
}

boxplot(sim_rat[, 1:2], main="Estimator Variance: SRS vs Ratio")
abline(h = tyU, col = "red", lwd=2)
```

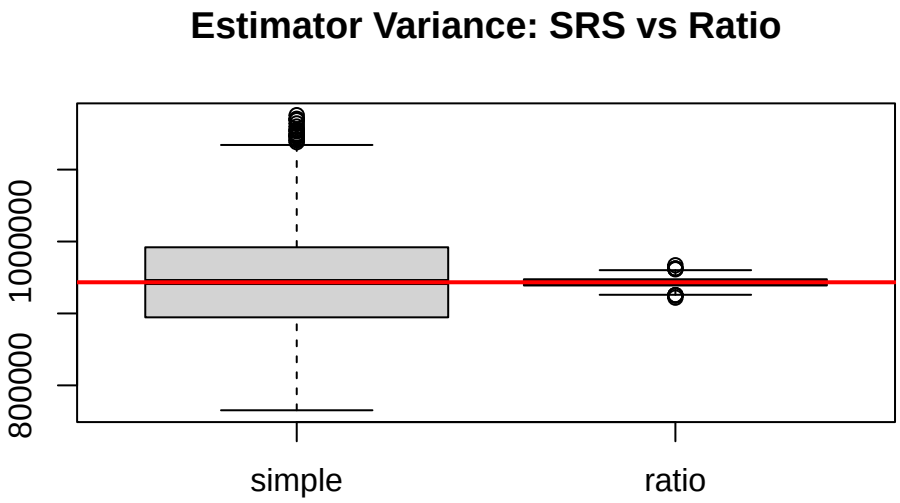

Figure 5.13: Estimator variance under repeated sampling: SRS vs. ratio estimator (2000 simulations)

We now calculate the actual variance across our 2000 simulations.

```
sim_var <- sapply(sim_rat[, 1:2], var)
ratio_sim_var <- sim_var / sim_var[1]
percentage_reduction <- 1 - ratio_sim_var

data.frame(sim_var, ratio_sim_var, percentage_reduction) |>
  gt(rownames_to_stub = TRUE) |>
  cols_label(
    sim_var = "Variance",
    ratio_sim_var = "Relative Variance",
    percentage_reduction = "Variance Reduction"
  ) |>
  fmt_number(columns = c(sim_var), decimals = 0) |>
  fmt_percent(columns = c(percentage_reduction), decimals = 2) |>
  fmt_number(columns = c(ratio_sim_var), decimals = 4)
```

Output Note: The boxplot dramatically illustrates the precision gain. The variance of the ratio estimator is only 0.8% of the SRS variance, confirming our theoretical expectations.

Table 5.7: Simulated variance and variance reduction: ratio estimator vs. SRS

	Variance	Relative Variance	Variance Reduction
simple	5,168,139,019	1.0000	0.00%
ratio	40,025,309	0.0077	99.23%

6 Cluster Sampling

6.1 What Is Cluster Sampling?

Suppose we want to find out how many bicycles are owned by the residents of a community of 10,000 households. We could take an SRS of 400 households. Alternatively, we could divide the community into blocks of about 20 households each, select 20 of the 500 blocks at random, and then interview every household (or a subsample of the households) in each selected block. The second plan is an example of **cluster sampling**. The blocks are the **primary sampling units (psus)**, or **clusters**. The households are the **secondary sampling units (ssus)**; often, as here, the ssus are the elements of the population.

Why would we sample clusters rather than elements?

- **A sampling frame of the observation units may be difficult, expensive or impossible to construct.** We cannot list all the honeybees in a region, or all the customers of a store. A list of the individuals in a city may be obtainable only through a list of housing units, which is time-consuming and expensive to build.
- **The population may be widely spread out geographically, or may occur in natural clusters** such as households or schools. It is then much cheaper to sample clusters than to take an SRS of individuals. To survey nursing-home residents, it is far cheaper to sample nursing homes and interview every resident in them than to take an SRS of residents, which might require travelling to a home to interview just one person.

The price is usually precision. Figure 6.1 illustrates a two-stage cluster sample from a synthetic population of $N = 10$ clusters of different sizes, from $M_i = 40$ to 160 units, sorted by their means. Three clusters are selected by SRS (shaded), and an SRS of $m_i = 20$ units is taken in each. Only the selected clusters are observed. When the cluster means differ, as here, the estimate depends heavily on *which* clusters happen to be chosen, and a cluster sample is usually less precise than an SRS of the same number of units.

The material in this chapter corresponds to Sections 5.1–5.3 of Lohr and Chapters 8 and 9 of Scheaffer et al.

6.1.1 Cluster sampling versus stratified sampling

Both designs partition the population into groups: every element belongs to exactly one stratum, or to exactly one cluster. The difference is in how the groups are used (Figure 6.2). In stratified sampling we take an SRS from *every* stratum. In one-stage cluster sampling we take an SRS of *clusters* and observe every element in the selected clusters; the other clusters are not observed at all.

The two designs therefore call for opposite kinds of groups. Stratification gains precision when each stratum is internally *homogeneous*, because only the variability within strata enters the variance. Cluster sampling loses precision when the clusters are internally homogeneous, because only a few clusters are seen, and the precision depends mainly on the variability *between* the cluster means. Ideally, each cluster would be a miniature of the whole population.

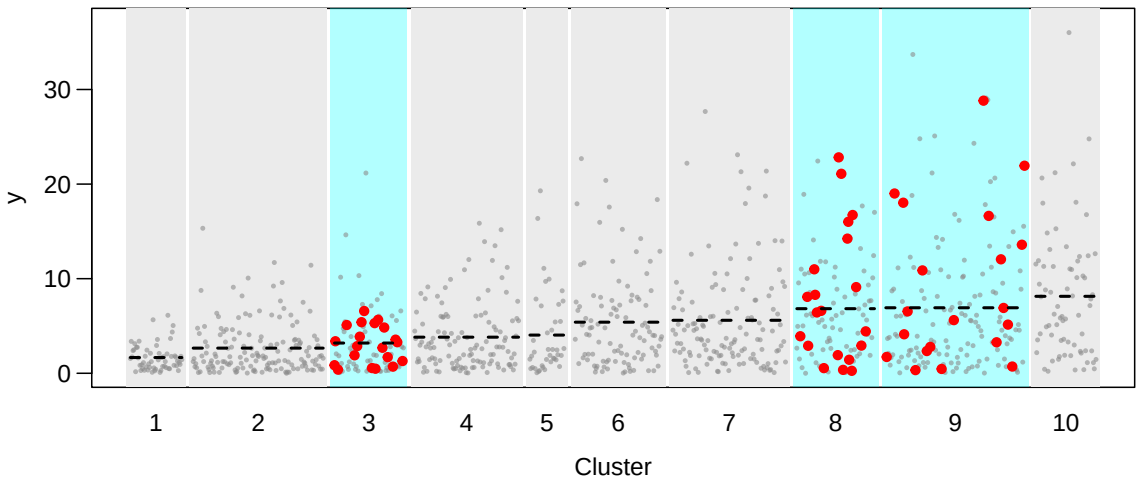

Figure 6.1: A two-stage cluster sample from a synthetic population of 10 clusters of 40 to 160 units, sorted by their means (dashed lines). Three clusters are selected at random (shaded), and 20 units (red) are subsampled in each; the other clusters are not observed at all.

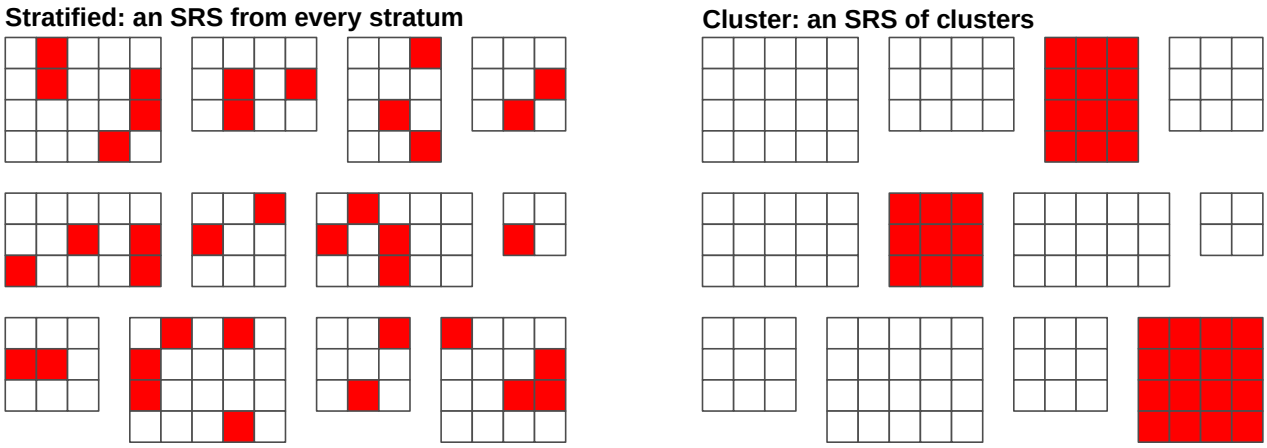

Figure 6.2: The same population of 12 groups of elements (cells), sampled two ways. Left: stratified sampling takes an SRS of elements (red) from every stratum. Right: one-stage cluster sampling takes an SRS of 3 clusters and observes all their elements.

6.2 Notation for Cluster Sampling

6.2.1 Population quantities

In cluster sampling the population $\mathcal{U}$ is the population of N psus. Note the change from earlier chapters: N **is now the number of psus, not the number of elements.** Psu i contains M_i ssus, and we write

$$y_{ij} = \text{the measurement on the } j\text{th ssu in the } i\text{th psu}, \qquad j = 1, \ldots, M_i, \quad i = 1, \ldots, N. \tag{6.1}$$

Figure 6.3 displays the population as a row of clusters. Each cluster is a column of cells, one cell per ssu, holding its value y_{ij}. The clusters have different numbers of cells: the size M_i of each cluster is written at its bottom, and its total t_i at its top.

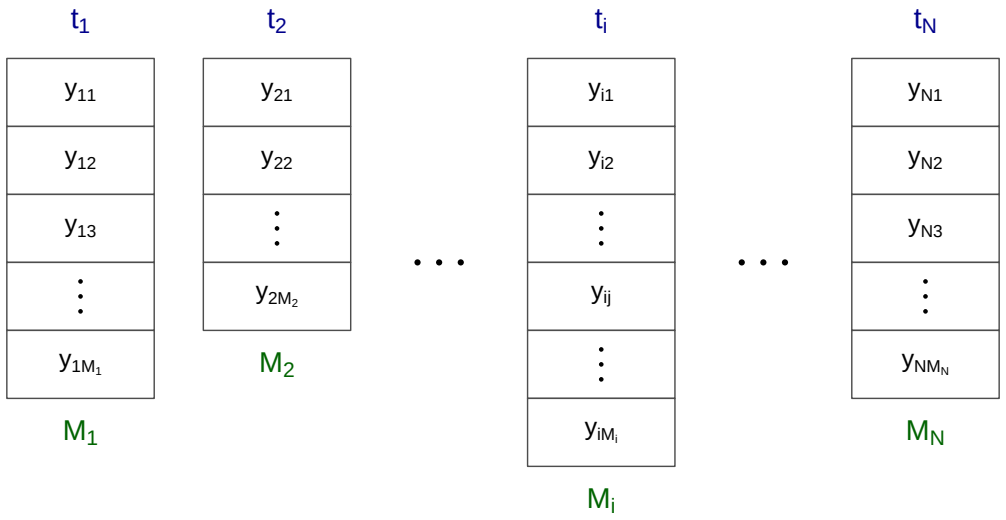

Figure 6.3: The population in cluster sampling. Cluster (psu) i is a column of M_i cells, one per ssu, holding the values $y_{i1}, \ldots, y_{iM_i}$. Its total $t_i = \sum_j y_{ij}$ is shown at the top and its size M_i at the bottom.

The **psu-level** quantities are the cluster totals and their spread:

$$t_i = \sum_{j=1}^{M_i} y_{ij} \quad \text{(total in psu } i\text{)}, \qquad t = \sum_{i=1}^{N} t_i \quad \text{(population total)}, \tag{6.2}$$

$$M_0 = \sum_{i=1}^{N} M_i \quad \text{(number of ssus)}, \tag{6.3}$$

$$S_t^2 = \frac{1}{N-1} \sum_{i=1}^{N} \left(t_i - \frac{t}{N} \right)^2 \quad \text{(population variance of the psu totals)}. \tag{6.4}$$

The **ssu-level** quantities describe the individual elements:

$$\bar{y}_U = \frac{1}{M_0} \sum_{i=1}^{N} \sum_{j=1}^{M_i} y_{ij} = \frac{t}{M_0} \quad \text{(population mean per ssu)}, \qquad \bar{y}_{iU} = \frac{t_i}{M_i} \quad \text{(mean in psu } i\text{)}, \tag{6.5}$$

$$S^2 = \frac{1}{M_0 - 1} \sum_{i=1}^{N} \sum_{j=1}^{M_i} (y_{ij} - \bar{y}_U)^2 \quad \text{(population variance)}, \tag{6.6}$$

$$S_i^2 = \frac{1}{M_i - 1} \sum_{j=1}^{M_i} (y_{ij} - \bar{y}_{iU})^2 \quad \text{(variance within psu } i\text{)}. \tag{6.7}$$

Here $\bar{y}_U$, the mean per element, is usually the quantity of interest, and $\bar{t}_U = t/N$ is the mean per psu.

6.2.2 Sample quantities

The sample is drawn in up to two stages (Figure 6.4). First an SRS $\mathcal{S}$ of n psus is drawn from the N psus. Then, in each sampled psu i, an SRS $\mathcal{S}_i$ of m_i of its M_i ssus is drawn. In **one-stage** cluster sampling every ssu of a sampled psu is observed, $m_i = M_i$; in **two-stage** cluster sampling $m_i < M_i$ for some psus.

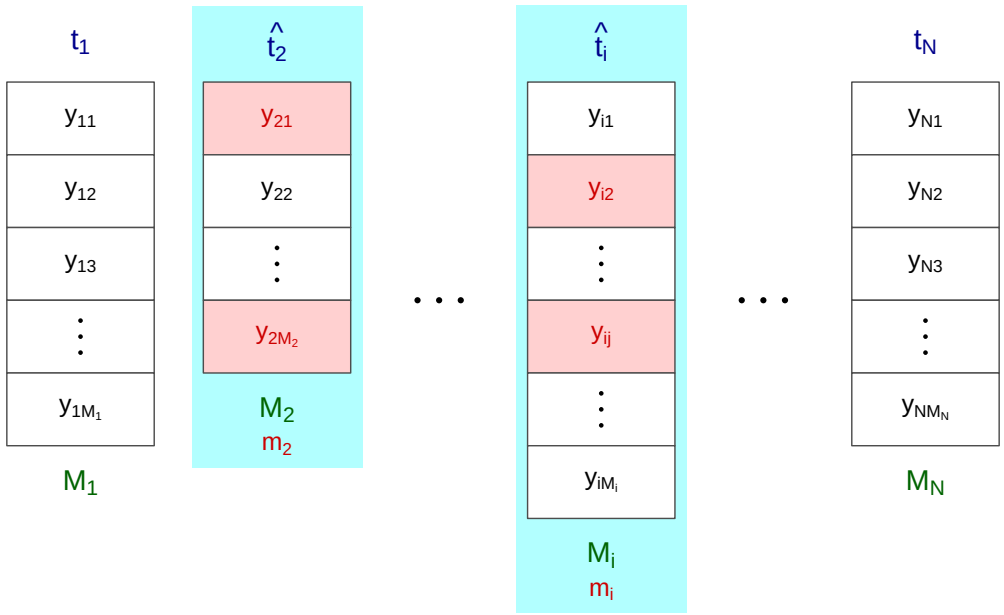

Figure 6.4: A two-stage cluster sample. The psus 2 and i are in the sample $\mathcal{S}$ (shaded); within each, m_i of the M_i ssus are subsampled (red). The psu totals are not observed and must be estimated, giving $\hat{t}_i$.

From the subsample in psu i we compute the sample mean and the estimated psu total,

$$\bar{y}_i = \frac{1}{m_i} \sum_{j \in \mathcal{S}_i} y_{ij}, \qquad \hat{t}_i = \sum_{j \in \mathcal{S}_i} \frac{M_i}{m_i} y_{ij} = M_i \bar{y}_i, \tag{6.8}$$

and from the n sampled psus the unbiased estimator of the population total and the sample variance of the estimated psu totals:

$$\hat{t}_{\text{unb}} = \frac{N}{n} \sum_{i \in \mathcal{S}} \hat{t}_i, \qquad \bar{t} = \frac{1}{n} \sum_{i \in \mathcal{S}} \hat{t}_i, \qquad s_t^2 = \frac{1}{n-1} \sum_{i \in \mathcal{S}} (\hat{t}_i - \bar{t})^2. \tag{6.9}$$

In one-stage sampling $\hat{t}_i = t_i$, the true psu total.

6.2.3 Sampling weights

Ssu j of psu i is in the sample if psu i is selected at the first stage and ssu j is selected at the second, so

$$P(\text{ssu } j \text{ of psu } i \text{ is selected}) = P(\text{psu } i \text{ selected}) \times P(\text{ssu } j \text{ selected} \mid \text{psu } i \text{ selected}) = \frac{n}{N} \cdot \frac{m_i}{M_i}. \tag{6.10}$$

The sampling weight is the reciprocal,

$$w_{ij} = \frac{N}{n} \cdot \frac{M_i}{m_i}, \qquad \hat{t}_{\text{unb}} = \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}_i} w_{ij}\, y_{ij}. \tag{6.11}$$

If the psus are blocks and the ssus are households, household j in block i represents $NM_i/(nm_i)$ households of the population.

6.3 One-Stage Cluster Sampling with Equal Cluster Sizes

Consider first the simplest case: every psu has the same number of ssus, $M_i = M$, and all of them are observed, $m_i = M$. This is uncommon in household surveys but does occur in agricultural and industrial sampling. Since every ssu in a sampled psu is observed, the psu totals t_i are known for the sampled psus, and we can ignore the individual elements. **The design is simply an SRS of n psu totals** $\{t_i, i \in \mathcal{S}\}$ from the N psu totals, and the SRS results of Section 3.2 apply directly:

$$\hat{t} = N\bar{t} = \frac{N}{n} \sum_{i \in \mathcal{S}} t_i, \qquad \text{SE}(\hat{t}) = N\sqrt{\left(1 - \frac{n}{N}\right)\frac{s_t^2}{n}}. \tag{6.12}$$

The population has NM elements, so the mean per element is estimated by

$$\hat{\bar{y}} = \frac{\hat{t}}{NM} = \frac{\bar{t}}{M}, \qquad \text{SE}(\hat{\bar{y}}) = \frac{1}{M}\sqrt{\left(1 - \frac{n}{N}\right)\frac{s_t^2}{n}} = \frac{\text{SE}(\bar{t})}{M}. \tag{6.13}$$

6.3.1 Example: GPA in a dormitory

A student wants to estimate the average GPA in a dormitory. Instead of listing all the students and taking an SRS, the student notices that the dorm consists of $N = 100$ suites of $M = 4$ students each. The student selects $n = 5$ suites at random and asks every person in them for their GPA:

```
gpa <- data.frame(
  suite = rep(1:5, each = 4),
  gpa   = c(3.08, 2.60, 3.44, 3.04,  2.36, 3.04, 3.28, 2.68,  2.00, 2.56, 2.52, 1.88,
            3.00, 2.88, 3.44, 3.64,  2.68, 1.92, 3.28, 3.20)
)
ti <- tapply(gpa$gpa, gpa$suite, sum)      # suite totals t_i
matrix(gpa$gpa, nrow = 5, byrow = TRUE,
       dimnames = list(paste("Suite", 1:5), paste("Person", 1:4))) |>
  cbind(`Total t_i` = ti)
```

```
        Person 1 Person 2 Person 3 Person 4 Total t_i
Suite 1     3.08     2.60     3.44     3.04     12.16
Suite 2     2.36     3.04     3.28     2.68     11.36
Suite 3     2.00     2.56     2.52     1.88      8.96
Suite 4     3.00     2.88     3.44     3.64     12.96
Suite 5     2.68     1.92     3.28     3.20     11.08
```

The psus are the suites. Treating the five suite totals as an SRS from the 100 suite totals,

```
N <- 100; n <- 5; M <- 4
t_hat  <- N / n * sum(ti)
s2_t   <- var(ti)
ybar   <- t_hat / (N * M)
se_ybar <- sqrt((1 - n / N) * s2_t / n) / M
c(t_hat = t_hat, tbar = mean(ti), s2_t = s2_t, ybar = ybar, SE = se_ybar)
```

```
      t_hat        tbar        s2_t        ybar          SE
1130.4000000  11.3040000   2.2556800   2.8260000   0.1636649
```

$$\hat{t} = \frac{100}{5}(12.16 + 11.36 + 8.96 + 12.96 + 11.08) = 1130.4, \qquad s_t^2 = 2.256, \tag{6.14}$$

$$\hat{\bar{y}} = \frac{1130.4}{400} = 2.826, \qquad \mathrm{SE}(\hat{\bar{y}}) = \sqrt{\left(1 - \frac{5}{100}\right)\frac{2.256}{(5)(4)^2}} = 0.164. \tag{6.15}$$

The same estimate is obtained by `srs_est()` applied to the suite totals, divided by M:

```
srs_est(ti / M, N = N, estimate = "mean", show.details = FALSE)$estimate |>
 ↪  format_est_gt("mean")
```

$\bar{y}$	SE($\bar{y}$)	95% CI Lower	95% CI Upper
2.83	0.16	2.37	3.28

(The confidence interval here uses $t_{0.025,\,4}$, since only $n = 5$ suites were sampled.)

Do not use the SRS formula for the 20 students. It is tempting to treat the $nM = 20$ students as an SRS from the 400 students in the dorm:

```
se_naive <- sqrt((1 - 20 / 400) * var(gpa$gpa) / 20)
c(SE_cluster = se_ybar, SE_naive_SRS = se_naive)
```

```
  SE_cluster SE_naive_SRS
   0.1636649    0.1121526
```

The naive standard error, 0.112, is much smaller than the correct 0.164. The 20 students are not an SRS of students: whole suites are selected together, and students in the same suite tend to have similar GPAs. Treating them as independent observations overstates the information in the sample. The next section makes this precise.

6.4 Comparing Cluster Sampling with SRS

How much precision do we lose by sampling clusters instead of elements? For equal-sized clusters the answer comes from the analysis-of-variance decomposition of the population, with the clusters playing the role of groups:

Source	df	Sum of squares
Between psus	$N-1$	$\text{SSB} = \sum_{i=1}^{N}\sum_{j=1}^{M}(\bar{y}_{iU}-\bar{y}_U)^2$
Within psus	$N(M-1)$	$\text{SSW} = \sum_{i=1}^{N}\sum_{j=1}^{M}(y_{ij}-\bar{y}_{iU})^2$
Total, about $\bar{y}_U$	$NM-1$	$\text{SSTO} = \sum_{i=1}^{N}\sum_{j=1}^{M}(y_{ij}-\bar{y}_U)^2 = (NM-1)S^2$

with mean squares $\text{MSB} = \text{SSB}/(N-1)$ and $\text{MSW} = \text{SSW}/(N(M-1))$.

The variance depends on the between-psu variability. Since every psu has size M, $t_i - \bar{t}_U = M(\bar{y}_{iU}-\bar{y}_U)$, so

$$S_t^2 = \frac{1}{N-1}\sum_{i=1}^{N}(t_i-\bar{t}_U)^2 = \frac{M}{N-1}\sum_{i=1}^{N}M(\bar{y}_{iU}-\bar{y}_U)^2 = M\cdot\text{MSB}, \tag{6.16}$$

and the variance of the estimated total from one-stage cluster sampling is

$$V(\hat{t}_{\text{cluster}}) = N^2\Big(1-\frac{n}{N}\Big)\frac{S_t^2}{n} = N^2\Big(1-\frac{n}{N}\Big)\frac{M\cdot\text{MSB}}{n}. \tag{6.17}$$

An SRS of the same number nM of elements, from the NM elements, would give

$$V(\hat{t}_{\text{SRS}}) = (NM)^2\Big(1-\frac{nM}{NM}\Big)\frac{S^2}{nM} = N^2\Big(1-\frac{n}{N}\Big)\frac{MS^2}{n}. \tag{6.18}$$

The ratio of the two variances is therefore

$$\frac{V(\hat{t}_{\text{cluster}})}{V(\hat{t}_{\text{SRS}})} = \frac{\text{MSB}}{S^2}. \tag{6.19}$$

If $\text{MSB} > S^2$, that is, if the cluster means vary more than would be expected from randomly formed groups, cluster sampling is **less efficient** than an SRS of the same number of elements.

6.4.1 Intraclass correlation and adjusted R^2

Two measures of how similar the elements within a cluster are, compared with elements from different clusters, are the **intraclass correlation coefficient**

$$\text{ICC} = 1 - \frac{M}{M-1}\cdot\frac{\text{SSW}}{\text{SSTO}}, \tag{6.20}$$

and the **adjusted** R^2, which also applies to clusters of unequal sizes,

$$R_a^2 = 1 - \frac{\text{MSW}}{S^2}. \tag{6.21}$$

Both are close to 0 when the clusters are like random groups of elements, and close to 1 when the elements within each cluster are nearly identical. For equal-sized clusters, the variance ratio can be written in terms of R_a^2:

$$\frac{V(\hat{t}_{\text{cluster}})}{V(\hat{t}_{\text{SRS}})} = \frac{\text{MSB}}{S^2} = 1 + \frac{N(M-1)}{N-1} R_a^2 \approx 1 + (M-1)R_a^2. \tag{6.22}$$

So $(M-1)R_a^2$ is approximately the proportional increase in variance from using cluster sampling instead of an SRS of the same size. Even a small R_a^2 matters when the clusters are large: with $M = 21$ elements per cluster and $R_a^2 = 0.05$, the variance doubles.

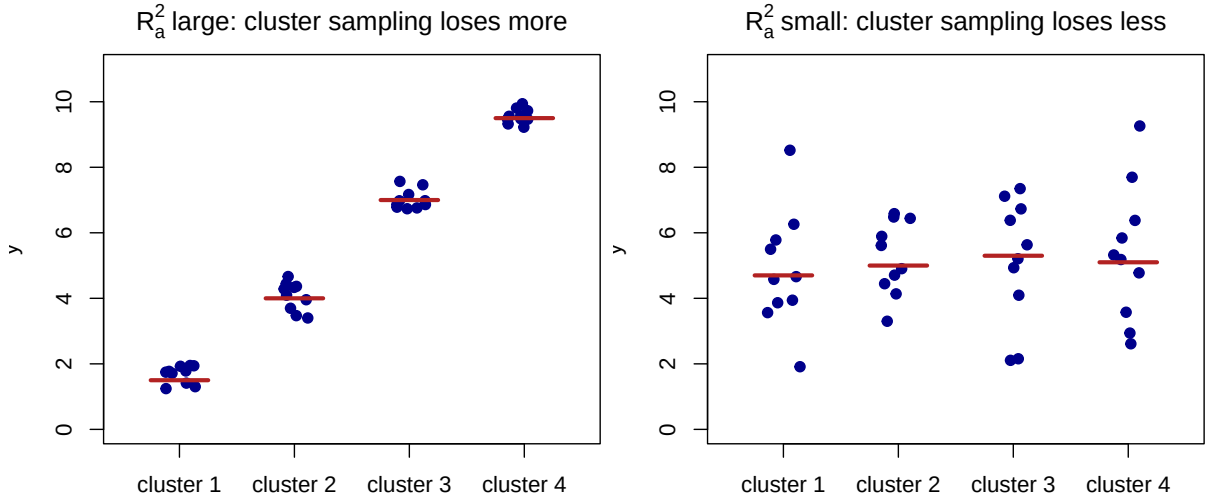

Figure 6.5: Left: the cluster means differ a lot relative to the spread within the clusters (R_a^2 large), so sampling a few clusters misses most of the population's variability and cluster sampling loses much precision. Right: the clusters look like random groups (R_a^2 small), and cluster sampling loses little.

This is the mirror image of stratification (Section 4.7). There, a large R^2 between strata *reduces* the variance, because every stratum is sampled. Here, a large R_a^2 between clusters *increases* the variance, because most clusters are not sampled.

6.5 One-Stage Cluster Sampling with Unequal Cluster Sizes

In most surveys the clusters have different sizes: classes have different numbers of students, and counties different numbers of farms.

6.5.1 The unbiased estimator

The unbiased estimator of the total is computed exactly as before,

$$\hat{t}_{\text{unb}} = \frac{N}{n} \sum_{i \in \mathcal{S}} t_i, \qquad \text{SE}(\hat{t}_{\text{unb}}) = N \sqrt{\left(1 - \frac{n}{N}\right) \frac{s_t^2}{n}}, \tag{6.23}$$

and $\hat{t}_{\text{unb}}/M_0$ is unbiased for $\bar{y}_U$ when M_0 is known. But now the cluster totals t_i vary for two reasons. A psu with many ssus tends to have a large total simply because it has more elements, quite apart from any differences in the y-values. When the cluster sizes differ a lot, this inflates s_t^2, and with it the variance of $\hat{t}_{\text{unb}}$.

6.5.2 The ratio estimator

Since $\bar{y}_U = t/M_0$, and t_i is usually roughly proportional to M_i, estimating $\bar{y}_U$ is a **ratio estimation** problem (see the ratio chapter) with y-variable t_i and auxiliary variable $x_i = M_i$:

$$\hat{\bar{y}}_r = \frac{\sum_{i\in\mathcal{S}} t_i}{\sum_{i\in\mathcal{S}} M_i} = \frac{\sum_{i\in\mathcal{S}} M_i \bar{y}_i}{\sum_{i\in\mathcal{S}} M_i}. \tag{6.24}$$

It is the total of the sampled elements divided by their number, a weighted average of the cluster means with weights M_i. Its standard error follows from the ratio-estimator formula with $x_i = M_i$:

$$\text{SE}(\hat{\bar{y}}_r) = \sqrt{\left(1 - \frac{n}{N}\right)\frac{s_e^2}{n\bar{M}^2}}, \qquad s_e^2 = \frac{1}{n-1}\sum_{i\in\mathcal{S}} e_i^2, \qquad e_i = t_i - \hat{\bar{y}}_r M_i, \tag{6.25}$$

where $\bar{M} = \sum_{i\in\mathcal{S}} M_i/n$ is the average size of the sampled clusters. The residuals e_i measure how far each cluster total lies from the line $t = \hat{\bar{y}}_r M$ through the origin (Figure 6.6). They are typically far less variable than the deviations $t_i - \bar{t}$ about the horizontal line $\bar{t}$, which determine s_t^2 and hence the standard error of the unbiased estimator, because the size-driven part of the variation has been removed. Note that the ratio estimator does not need M_0, the total number of elements in the population.

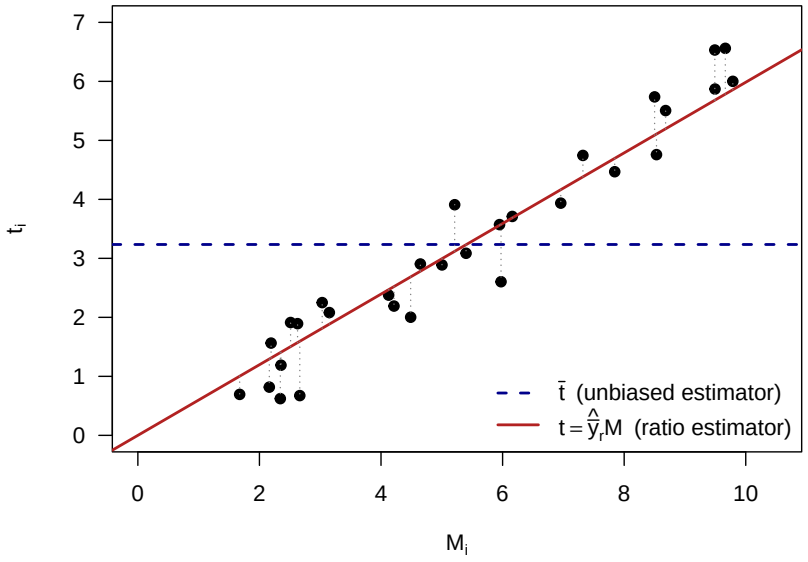

Figure 6.6: Cluster totals t_i against cluster sizes M_i. The unbiased estimator's standard error depends on the spread of the t_i about their mean $\bar{t}$ (blue dashed line), which is large because the sizes vary. The ratio estimator uses the line $t = \hat{\bar{y}}_r M$ (red), and its standard error depends only on the residuals about that line (dotted).

6.5.3 A toy example: why the unbiased estimator can be bad

Consider a whimsical population of just $N = 2$ clusters (kennels) of dogs: kennel A with $M_A = 30$ dogs and kennel B with $M_B = 10$ dogs. Every dog has exactly 4 legs, so the mean number of legs per dog is $\bar{y}_U = 4$, and $M_0 = 40$. We take a one-stage cluster sample of $n = 1$ kennel, each kennel being chosen with probability $1/2$.

The unbiased estimator. If kennel A is selected, $t_A = 30 \times 4 = 120$, $\hat{t}_{\text{unb}} = \frac{2}{1}(120) = 240$ and $\hat{\bar{y}}_{\text{unb}} = 240/40 = 6$. If kennel B is selected, $t_B = 40$, $\hat{t}_{\text{unb}} = 80$ and $\hat{\bar{y}}_{\text{unb}} = 80/40 = 2$. So

$$E[\hat{\bar{y}}_{\text{unb}}] = 6 \times \tfrac{1}{2} + 2 \times \tfrac{1}{2} = 4 = \bar{y}_U, \tag{6.26}$$

and the estimator is unbiased. But it is *never* equal to 4: it is always off by 2, even though every dog has exactly 4 legs. All of its variance comes from the different kennel sizes.

The ratio estimator. If kennel A is selected, $\hat{\bar{y}}_r = (30 \times 4)/30 = 4$; if kennel B is selected, $\hat{\bar{y}}_r = (10 \times 4)/10 = 4$. Both possible samples give exactly the right answer, and $V(\hat{\bar{y}}_r) = 0$. The ratio estimator removes the cluster-size variability that made $\hat{\bar{y}}_{\text{unb}}$ so unreliable. This is why ratio estimation, not unbiased estimation, is the standard method for clusters of unequal sizes.

6.5.4 Example: algebra class test scores (algebra.csv)

One-stage cluster samples are common in educational studies, since students are naturally clustered into classes and schools. Consider a population of 187 high school algebra classes in a city. An investigator takes an SRS of $n = 12$ classes and tests every student in them on their knowledge of functions. The classes are the psus, of different sizes M_i, and the students are the ssus.

```
algebra <- read.csv("data/algebra.csv")

book_table(algebra) |>
  tab_header(title = "Raw Data: Algebra Classes") |>
  cols_label(Mi = md("*M*~i~"))
```

The function `cluster_ratio()` (from `samplingestimate.r`) aggregates the data to one row per class and builds the ratio-estimation working table: the class size M_i, the class total t_i, the fitted total $\hat{\bar{y}}_r M_i$, and the residual e_i and its square.

```
alg_res <- cluster_ratio(data = algebra,
                         cname = "class",
                         csize = "Mi",
                         yvar = "score",
                         N = 187)
alg_res$table
```

Table 6.1: Raw data: algebra classes (sample)

Raw Data: Algebra Classes

class	M_i	score
23	20	57
23	20	90
23	20	56
23	20	57
23	20	46
23	20	55
23	20	62
23	20	66
23	20	78
23	20	76

Showing the first 10 of 299 rows; the full table is in the HTML version of the book.

Ratio Estimation: Working Table
$n = 12$

i	$\bar{y}_i$	$\hat{t}_i$ [1]	M_i	$\hat{B}M_i$ [2]	e_i	e_i^2
1	61.50	1,230.00	20.00	1,251.37	-21.37	456.73
2	64.23	1,670.00	26.00	1,626.78	43.22	1,867.74
3	58.42	1,402.00	24.00	1,501.65	-99.65	9,929.22
4	58.00	1,972.00	34.00	2,127.33	-155.33	24,127.75
5	58.00	1,508.00	26.00	1,626.78	-118.78	14,109.31
6	64.86	1,816.00	28.00	1,751.92	64.08	4,106.28
...	...	...	...	...	...	...
Sum	**...**	**18,708.00**	**299.00**	**18,708.00**	**0.00**	**194,827.04**

[1] $\hat{t}_i = \bar{y}_i\, M_i$.
[2] $\hat{B} = \sum_i \hat{t}_i / \sum_i M_i = 62.57$, $\hat{B}M_i = \hat{B}\, M_i$, $s_e^2 = 17,711.55$.

$$\bar{y}_r = \frac{\sum_i \hat{t}_i}{\sum_i M_i} = \frac{18,708.00}{299.00} = 62.57$$

$$\text{SE}(\bar{y}_r) = \frac{1}{\bar{x}}\sqrt{\left(1 - \frac{n}{N}\right)\frac{s_e^2}{n}} = \frac{1}{24.92}\sqrt{\left(1 - \frac{12}{187}\right)\frac{17,711.55}{12}} = 1.49$$

The Sum row gives $\sum M_i$, $\sum t_i$ and $\sum e_i^2$, from which

$$\hat{\bar{y}}_r = \frac{\sum t_i}{\sum M_i}, \qquad s_e^2 = \frac{\sum e_i^2}{n-1}, \qquad \bar{M} = \frac{\sum M_i}{12}, \qquad \text{SE}(\hat{\bar{y}}_r) = \sqrt{\left(1 - \frac{12}{187}\right)\frac{s_e^2}{12\,\bar{M}^2}}. \tag{6.27}$$

```
alg_res$estimate |> format_est_gt("ratio_mean")
```

$\bar{y}_r$	SE($\bar{y}_r$)	95% CI Lower	95% CI Upper
62.57	1.49	59.29	65.85

The estimated mean score of a student in the 187 classes is 62.57, with standard error 1.49 and 95% confidence interval [59.29, 65.85]. The interval uses $n - 1 = 11$ degrees of freedom: the classes, not the students, are the units that were randomized.

6.6 Two-Stage Cluster Sampling

Observing every ssu in a sampled psu can be wasteful: if the elements in a cluster are similar, measuring a few of them tells us almost as much as measuring all. In **two-stage cluster sampling** we

1. select an SRS $\mathcal{S}$ of n psus from the N psus, and
2. select an SRS $\mathcal{S}_i$ of m_i ssus from each selected psu i.

The psu totals are no longer observed and are estimated by $\hat{t}_i = M_i \bar{y}_i$ (Figure 6.4).

6.6.1 The unbiased estimator

The unbiased estimator of the total is

$$\hat{t}_{\text{unb}} = \frac{N}{n} \sum_{i \in \mathcal{S}} \hat{t}_i = \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}_i} w_{ij} y_{ij}, \qquad w_{ij} = \frac{N M_i}{n m_i}. \tag{6.28}$$

Its variance now has two sources: the variability *between* psus, from selecting only some of them, and the variability *within* psus, from subsampling:

$$V(\hat{t}_{\text{unb}}) = \underbrace{N^2 \Big(1 - \frac{n}{N}\Big) \frac{S_t^2}{n}}_{\text{between psus}} + \underbrace{\frac{N}{n} \sum_{i=1}^{N} \Big(1 - \frac{m_i}{M_i}\Big) M_i^2 \frac{S_i^2}{m_i}}_{\text{within psus}}. \tag{6.29}$$

The first term is exactly the one-stage variance. The second term is the extra cost of not observing every ssu in each sampled psu; it vanishes when $m_i = M_i$.

6.6.2 The ratio estimator

As in one-stage sampling, the mean per element is estimated by a ratio estimator, now with the estimated psu totals $\hat{t}_i = M_i \bar{y}_i$:

$$\boxed{\hat{\bar{y}}_r = \frac{\sum_{i\in\mathcal{S}} \hat{t}_i}{\sum_{i\in\mathcal{S}} M_i} = \frac{\sum_{i\in\mathcal{S}} M_i \bar{y}_i}{\sum_{i\in\mathcal{S}} M_i}.} \tag{6.30}$$

Its estimated variance is

$$\hat{V}(\hat{\bar{y}}_r) = \frac{1}{\bar{M}^2}\Big(1-\frac{n}{N}\Big)\frac{s_e^2}{n} + \frac{1}{nN\bar{M}^2}\sum_{i\in\mathcal{S}} M_i^2\Big(1-\frac{m_i}{M_i}\Big)\frac{s_i^2}{m_i}, \qquad e_i = \hat{t}_i - \hat{\bar{y}}_r M_i, \tag{6.31}$$

where s_i^2 is the sample variance of the m_i elements measured in psu i.

The two terms have a useful interpretation. Writing $M = \sum_{i\in\mathcal{S}} M_i = n\bar{M}$ for the number of elements in the sampled psus, the variance splits into the variance of a **ratio** estimate computed over the n sampled clusters, plus the cluster sampling fraction n/N times the variance of a **stratified** sample in which those same clusters are the strata:

$$\hat{V}(\hat{\bar{y}}_r) = \underbrace{\Big(1-\frac{n}{N}\Big)\frac{1}{n\bar{M}^2}\frac{\sum_{i\in\mathcal{S}}(\hat{t}_i - \hat{\bar{y}}_r M_i)^2}{n-1}}_{\hat{V}_{\text{ratio}}} + \underbrace{\frac{n}{N}\sum_{i\in\mathcal{S}}\Big(\frac{M_i}{M}\Big)^2\Big(1-\frac{m_i}{M_i}\Big)\frac{s_i^2}{m_i}}_{\hat{V}_{\text{str}}}. \tag{6.32}$$

$\hat{V}_{\text{ratio}}$ is exactly the ratio-estimate variance for $\sum \hat{t}_i / \sum M_i$, and $\hat{V}_{\text{str}}$ is exactly the stratified variance $\sum_h W_h^2(1 - n_h/N_h)s_h^2/n_h$ with the sampled clusters as strata: stratum sizes $N_h = M_i$, sample sizes $n_h = m_i$, and weights $W_h = M_i/M$. Each term

$$\hat{v}_i = \Big(\frac{M_i}{M}\Big)^2\Big(1-\frac{m_i}{M_i}\Big)\frac{s_i^2}{m_i} \tag{6.33}$$

is one stratum's contribution; the working tables below carry them in a $\hat{v}_i$ column whose Sum is $\hat{V}_{\text{str}}$.

Why is the second piece there at all? If we could see the true cluster totals t_i, the ratio piece alone would be the whole variance. Subsampling replaces t_i by the estimate $\hat{t}_i$, and the between-cluster sum of squares absorbs only the fraction $1 - n/N$ of the noise that this creates; the stratified piece pays the rest. Three special cases follow straight from the formula:

- **One-stage sampling** ($m_i = M_i$): every $\hat{v}_i = 0$, and only the ratio piece survives, as in Section 6.5.
- N **unknown or effectively infinite**: $n/N \to 0$, the stratified piece drops out, and s_e^2 alone does the whole job. In practice the second term is often negligible compared with the first.
- **A census of clusters** ($n = N$): the factor $1 - n/N$ kills the ratio piece and $n/N = 1$ leaves $\hat{V}(\hat{\bar{y}}_r) = \hat{V}_{\text{str}}$ exactly. Drawing every cluster *is* stratified sampling, and the formula says so.

6.6.3 Example: American coot egg volumes (coots.csv)

The data come from Arnold's (1991) study of the size and volume of American coot eggs in Minnedosa, Manitoba. The clutches (nests) are the psus and the eggs are the ssus. From each of $n = 184$ clutches, two eggs were measured; the clutch sizes M_i are known.

Table 6.2: Raw data: coot egg clutches (sample)

clutch	csize	length	breadth	volume	tmt
1	13	44.30	31.10	3.7957569	1
1	13	45.90	32.70	3.9328497	1
2	13	49.20	34.40	4.2156036	1
2	13	48.70	32.70	4.1727621	1
3	6	51.05	34.25	0.9317646	0
3	6	49.35	34.40	0.9007362	0
4	11	49.20	31.55	3.0182724	1
4	11	48.55	33.10	2.9783969	1
5	10	49.40	34.55	2.5045800	1
5	10	49.05	34.95	2.4868350	1

Showing the first 10 of 368 rows; the full table is in the HTML version of the book.

```
coots <- read.csv("data/coots.csv")
book_table(coots)
```

Figure 6.7 displays the data. The left panel plots the two egg volumes of each clutch against the clutch number. The right panel orders the clutches by their mean volume and joins the two eggs of each clutch by a vertical segment. The clutch means vary widely, but the two eggs *within* a clutch usually agree closely: eggs in the same clutch are much more similar than two eggs chosen at random from different clutches.

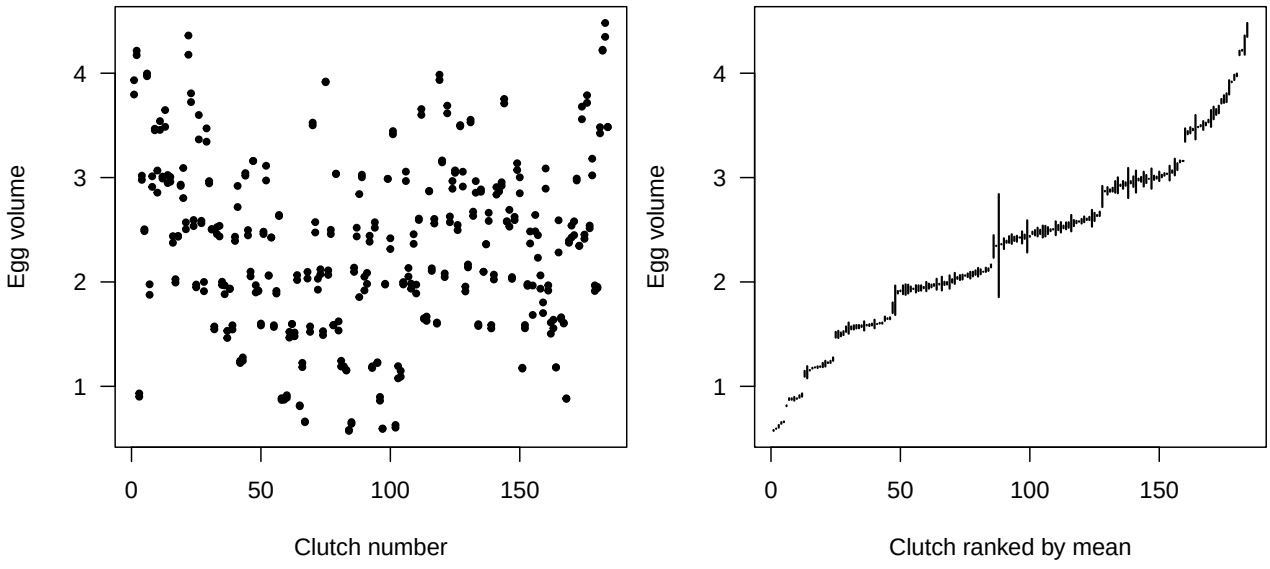

Figure 6.7: Egg volumes of the American coot. Left: the two measured eggs of each clutch against the clutch number. Right: clutches ranked by their mean volume, with the two eggs of each clutch joined by a segment.

We apply the same ratio estimator, with the estimated clutch totals $\hat{t}_i = M_i \bar{y}_i$:

```
coots_res <- cluster_ratio(data = coots,
                           cname = "clutch",
                           csize = "csize",
                           yvar = "volume")
coots_res$table
```

Ratio Estimation: Working Table
$n = 184$

i	m_i	$\bar{y}_i$	$\hat{t}_i$ [1]	M_i	$\hat{B}M_i$ [2]	e_i	e_i^2
1	2.00	3.86	50.24	13.00	32.38	17.86	318.92
2	2.00	4.19	54.52	13.00	32.38	22.15	490.48
3	2.00	0.92	5.50	6.00	14.94	-9.45	89.23
4	2.00	3.00	32.98	11.00	27.40	5.59	31.20
5	2.00	2.50	24.96	10.00	24.91	0.05	0.00
6	2.00	3.98	51.80	13.00	32.38	19.42	377.05
...	...	...	...	...	...	...	...
Sum	**368.00**	**...**	**4,375.95**	**1,757.00**	**4,375.95**	**0.00**	**11,439.58**

[1] $\hat{t}_i = \bar{y}_i\, M_i$.
[2] $\hat{B} = \sum_i \hat{t}_i / \sum_i M_i = 2.49$, $\hat{B}M_i = \hat{B}\, M_i$, $s_e^2 = 62.51$.

$$\bar{y}_r = \frac{\sum_i \hat{t}_i}{\sum_i M_i} = \frac{4,375.95}{1,757.00} = 2.49$$

$$\text{SE}(\bar{y}_r) = \frac{1}{\bar{x}}\sqrt{\frac{s_e^2}{n}} = \frac{1}{9.55}\sqrt{\frac{62.51}{184}} = 0.06$$

Only a subset of eggs is measured in each clutch, so the working table carries an m_i column. The function estimates the total volume of each clutch by multiplying the clutch size M_i by the mean $\bar{y}_i$ of the measured eggs. The total number of clutches N is unknown but presumably very large, so the psu-level fpc is $1 - n/N \approx 1$ and the within-clutch term, with its factor $1/(nN)$, is negligible. No within-cluster variance column appears: with `N = Inf`, s_e^2 carries all of the subsampling noise on its own.

```
coots_res$estimate |> format_est_gt("ratio_mean")
```

$\bar{y}_r$	SE($\bar{y}_r$)	95% CI Lower	95% CI Upper
2.49	0.06	2.37	2.61

The estimated mean egg volume is 2.491, with standard error 0.061, a coefficient of variation of only 0.0245.

6.6.4 Example: farm acreage, with states as clusters (agpop.csv)

Neither example above actually exercises the within-cluster term: `algebra` is one-stage, and `coots` has an unknown N. To see it at work we need a population whose clusters we can count. The `agpop.csv` file provides one: its 3,059

counties (dropping those coded `-99`) sit inside exactly $N = 50$ states, so the states are clusters of known number and known sizes M_i.

Drawing the sample

We take a two-stage sample: an SRS of $n = 10$ states, followed by an SRS of $m_i \approx 30\%$ of the counties in each selected state.

```
agpop <- acres_in_thousands(read.csv("data/agpop.csv"))
agpop <- agpop[!is.na(agpop$acres92), ]

## cluster sizes M_i: the number of counties in each state
Mi_all <- table(agpop$state)
agpop$Mi <- as.integer(Mi_all[agpop$state])

set.seed(2026)
states <- sample(names(Mi_all), 10)                 # stage 1: n = 10 clusters
agclus <- do.call(rbind, lapply(states, function(st) {
  rows <- which(agpop$state == st)                  # stage 2: m_i = 0.3 M_i
  agpop[sample(rows, max(1, round(0.30 * length(rows)))), ]
}))

c(clusters = length(states), counties = nrow(agclus))
```

```
clusters counties
      10      184
```

Computing the estimate directly

Everything the estimator needs is a cluster-level summary: the known size M_i of each sampled state, the number of counties actually measured in it (m_i), their mean $\bar{y}_i$ and variance s_i^2, and the estimated state total $\hat{t}_i = M_i \bar{y}_i$.

```
n  <- length(states)                                          # clusters drawn
N  <- 50                                                     # clusters in all

Mi    <- tapply(agclus$Mi,      agclus$state, function(x) x[1])  # cluster sizes
mi    <- tapply(agclus$acres92, agclus$state, length)            # counties measured
ybari <- tapply(agclus$acres92, agclus$state, mean)              # subsample means
si2   <- tapply(agclus$acres92, agclus$state, var)               # within-cluster vars
thati <- Mi * ybari                                          # estimated totals

clus_summary <- data.frame(Mi, mi, ybari, si2, thati)

clus_summary |>
  gt(rownames_to_stub = TRUE) |>
  cols_label(Mi    = md("$M_i$"),    mi  = md("$m_i$"),
             ybari = md("$\\bar y_i$"), si2 = md("$s_i^2$"),
             thati = md("$\\hat t_i$")) |>
  fmt_auto(data = clus_summary, columns = c("ybari", "si2", "thati"))
```

Table 6.3: Cluster-level summaries for the 10 sampled states

	M_i	m_i	$\bar{y}_i$	s_i^2	$\hat{t}_i$
CA	58	17	443.89	203,331.78	25,745.46
NC	100	30	90.67	3,996.75	9,066.94
NE	93	28	489.72	100,362.46	45,543.98
NJ	21	6	40.47	2,483.35	849.86
NV	15	4	213.67	41,391.58	3,204.99
NY	59	18	123.28	9,047.88	7,273.64
OK	77	23	395.78	21,374.08	30,475.10
PA	65	20	104.15	5,516.48	6,769.90
UT	29	9	396.64	130,980.12	11,502.66
VA	96	29	67.00	2,568.08	6,431.95

The states drawn range from 15 to 100 counties, of which 4 to 30 were visited. s_i^2 is large throughout: county acreages vary enormously even inside a single state, and that is the variability the second-stage term accounts for.

The ratio estimate divides the estimated totals by the cluster sizes:

```
Mbar  <- mean(Mi)                  # average size of the sampled clusters
ybarr <- sum(thati) / sum(Mi)      # ybar_r = sum(t_i hat) / sum(M_i)

c(Mbar = Mbar, ybar_r = ybarr)
```

```
    Mbar   ybar_r
 61.3000 239.5832
```

Its variance has the two pieces from the formula above. The **ratio** piece measures how the estimated totals scatter about the fitted line $\hat{\bar{y}}_r M_i$:

```
## ratio piece: the variance of the ratio estimate over the 10 sampled states
ei  <- thati - ybarr * Mi          # ratio residuals
se2 <- sum(ei^2) / (n - 1)         # s_e^2
V_ratio <- (1 - n / N) * se2 / (n * Mbar^2)

c(s_e2 = se2, V_ratio = V_ratio)
```

```
       s_e2     V_ratio
1.650481e+08 3.513821e+03
```

The **stratified** piece treats those same 10 states as strata, of sizes M_i inside the $M = \sum_i M_i$ counties they contain, and is then scaled by the cluster sampling fraction n/N:

```
## stratified piece: strata = the sampled states, weights W_i = M_i / M
M     <- sum(Mi)                                  # elements in the 10 states
vi    <- (Mi / M)^2 * (1 - mi / Mi) * si2 / mi    # each stratum's contribution
V_str <- sum(vi)                                  # the stratified variance

c(M = M, n_Mbar = n * Mbar, V_str = V_str, scaled = (n / N) * V_str)
```

```
       M    n_Mbar     V_str    scaled
613.00000 613.00000 180.37073  36.07415
```

Adding the two gives the standard error, and the interval uses $n - 1$ degrees of freedom, since the clusters, not the counties, are the units that were randomized.

```
SE  <- sqrt(V_ratio + (n / N) * V_str)
mem <- qt(0.975, df = n - 1) * SE

c(Est. = ybarr, S.E. = SE, ci.low = ybarr - mem, ci.upp = ybarr + mem)
```

```
     Est.      S.E.    ci.low    ci.upp
239.58317  59.58099 104.80160 374.36475
```

The estimated mean 1992 acreage per county is 239.6 thousand acres, with a 95% confidence interval of [104.8, 374.4] thousand acres. The stratified piece contributes 1% of the variance; almost all of it comes from the differences between the states. The interval is wide: only 10 states were visited, so $n - 1 = 9$ degrees of freedom is all the between-cluster scatter provides.

Using the function `cluster_ratio`

The function performs exactly these steps, provided N is supplied.

```
clus_res <- cluster_ratio(data = agclus,
                          cname = "state",
                          csize = "Mi",
                          yvar = "acres92",
                          N = 50)

clus_res$table
```

Ratio Estimation: Working Table
$n = 10$

i	m_i	$\bar{y}_i$	s_i^2	$\hat{t}_i$ [1]	M_i	$\hat{B}M_i$ [2]	e_i	e_i^2	$\hat{v}_i$ [3]
1	17.00	443.89	203,331.78	25,745.46	58.00	13,895.82	11,849.63	1.40×10^8	75.69
2	30.00	90.67	3,996.75	9,066.94	100.00	23,958.32	-14,891.37	2.22×10^8	2.48
3	28.00	489.72	100,362.46	45,543.98	93.00	22,281.23	23,262.75	5.41×10^8	57.66
4	6.00	40.47	2,483.35	849.86	21.00	5,031.25	-4,181.39	1.75×10^7	0.35
5	4.00	213.67	41,391.58	3,204.99	15.00	3,593.75	-388.75	1.51×10^5	4.54
6	18.00	123.28	9,047.88	7,273.64	59.00	14,135.41	-6,861.77	4.71×10^7	3.24
...	...	...	...	...	...	...	...	...	...
Sum	**184.00**	**...**	**...**	**146,864.48**	**613.00**	**146,864.48**	**0.00**	**1.49×10^9**	**180.37**

[1] $\hat{t}_i = \bar{y}_i\, M_i$.
[2] $\hat{B} = \sum_i \hat{t}_i / \sum_i M_i = 239.58$, $\hat{B}M_i = \hat{B}\, M_i$, $s_e^2 = 1.65 \times 10^8$.
[3] $\hat{v}_i = \left(\dfrac{M_i}{M}\right)^2 \left(1 - \dfrac{m_i}{M_i}\right) \dfrac{s_i^2}{m_i}, \quad M = \sum_i M_i = n\bar{M}, \quad \sum_i \hat{v}_i = \hat{V}_{\text{str}}.$

$$\bar{y}_r = \frac{\sum_i \hat{t}_i}{\sum_i M_i} = \frac{1.47 \times 10^5}{613.00} = 239.58$$

$$\text{SE}(\bar{y}_r) = \sqrt{\left(1 - \frac{n}{N}\right) \frac{s_e^2}{n\,\bar{M}^2} + \frac{n}{N} \sum_i \hat{v}_i} = \sqrt{\left(1 - \frac{10}{50}\right) \frac{1.65 \times 10^8}{10 \times 61.30^2} + 36.07} = 59.58$$

Because this sample is two-stage *and* N is finite, the working table carries the three extra columns computed above: m_i, the number of counties measured in each state; s_i^2, the variance among them; and $\hat{v}_i = (M_i/M)^2(1 - m_i/M_i)s_i^2/m_i$, that state's contribution to the stratified variance. The Sum of the $\hat{v}_i$ column is $\hat{V}_{\text{str}}$ itself, and the note under the table spells the standard error out as $\hat{V}_{\text{ratio}} + (n/N)\hat{V}_{\text{str}}$.

```
clus_res$estimate |> format_est_gt("ratio_mean")
```

$\bar{y}_r$	SE($\bar{y}_r$)	95% CI Lower	95% CI Upper
239.58	59.58	104.80	374.36

The estimate, standard error and interval reproduce the direct computation above.

6.7 Interactive Demonstration: Clustering by State or at Random

The simulation for this section runs as a standalone app: Two-Stage Cluster Sampling (ratio estimator) with SRS. It opens in a new tab, together with its documentation and the full list of apps.

7 Unequal Probability Sampling

Unequal probability sampling (UPS) selects units with different probabilities, usually in proportion to a **size variable** known for every unit in the population. Large units contribute more to a total, so we give them a larger chance of being selected, and correct for this in the estimator. This chapter builds UPS in three steps, each a special case or an extension of the previous one:

1. **The Hansen–Hurwitz (HH) estimator**: the selection probabilities ψ_i are built from an auxiliary variable x_i, and we estimate the total or mean of y_i (Section 7.2).
2. **One-stage cluster sampling**: the units are clusters (psus), y_i is the cluster total t_i, and x_i is the cluster size M_i (Section 7.4).
3. **Two-stage cluster sampling**: the cluster total t_i is not observed, but estimated from a subsample of the cluster (Section 7.5).

Along the way, Section 7.3 compares the HH estimator with the sample average and the ratio estimator computed from a simple random sample of the same size.

Sampling **with replacement** (UPSWR) is used throughout, since it makes the variance easy to estimate: we make n independent draws, and on each draw unit i is selected with probability ψ_i, where $\sum_{i=1}^{N} \psi_i = 1$. A unit may be selected more than once, and it then appears in the sample once per draw. Sampling without replacement is briefly discussed at the end (Section 7.6).

7.1 Functions and Packages

The estimating functions used throughout this book (in this chapter `upswr_est` and `upswr_total` for sampling with replacement, `cluster_upswr` for clusters sampled with replacement, `upswo_est` for sampling without replacement, `srs_est`, `ratio_est` and `reg_est` for comparison, and the `format_est_gt` table-formatting helper) live in a single shared file, `samplingestimate.r`, which we source below.

The full source of `samplingestimate.r` is listed in Appendix C.

7.2 The Hansen–Hurwitz Estimator

7.2.1 Theory

The population has N units. Unit i has a study value y_i, with population total $t = \sum_{i=1}^{N} y_i$ and mean $\bar{y}_U = t/N$, and an auxiliary variable $x_i > 0$ whose total $t_x = \sum_{i=1}^{N} x_i$ is known. If y_i is roughly proportional to x_i (for example,

the number of physicians of a county and its population), we construct the selection probabilities from x_i:

$$\psi_i = \frac{x_i}{t_x}. \tag{7.1}$$

The ratio of each unit. Define, for every unit of the population,

$$b_i = \frac{y_i}{x_i}, \qquad i = 1, \dots, N, \tag{7.2}$$

and let $B = t/t_x = \bar{y}_U/\bar{x}_U$ be the population ratio. Since $y_i = x_i b_i$, B is the mean of the b_i weighted by x_i,

$$B = \frac{\sum_{i=1}^{N} x_i b_i}{\sum_{i=1}^{N} x_i}, \tag{7.3}$$

and once B is known, the mean and the total follow,

$$\bar{y}_U = \bar{x}_U\, B, \qquad t = t_x\, B, \qquad \bar{x}_U = \frac{t_x}{N}. \tag{7.4}$$

So estimating B is enough: we build an estimator $\hat{B}$ and convert it to $\hat{\bar{y}} = \bar{x}_U \hat{B}$ and $\hat{t} = t_x \hat{B}$. Everything below, for HH, the ratio estimator and the sample average, is written in terms of the b_i.

The sample $\mathcal{R}$ and its observations b_j^*. We make n independent draws with replacement; on each draw, unit i is selected with probability $\psi_i = x_i/t_x$. The sample, denoted $\mathcal{R}$ (not $\mathcal{S}$), is the list of the n draws **including repeats**: a unit drawn twice appears twice. Let b_j^* denote the ratio b of the unit selected on the jth draw, that is, the jth observation in $\mathcal{R}$ ($j = 1, \dots, n$). The star reminds us that b_j^* is indexed by the draw, not by the population unit: b_j^* is random, while $b_1, \dots, b_N$ are fixed numbers.

Sampling distribution of b_j^*. On every draw, b_j^* equals b_i exactly when unit i is selected, so b_j^* is a discrete random variable with distribution

Unit selected on draw j	1	2	⋯	N
Value of b_j^*	b_1	b_2	⋯	b_N
Probability	$\psi_1 = x_1/t_x$	$\psi_2 = x_2/t_x$	⋯	$\psi_N = x_N/t_x$

(if several units have the same value of b_i, their probabilities add up). Since the draws are made with replacement, the probabilities do not change from draw to draw, and $b_1^*, \dots, b_n^*$ are **independent and identically distributed** (iid) with this distribution. Its mean and variance are

$$E(b_j^*) = \sum_{i=1}^{N} \frac{x_i}{t_x} \cdot \frac{y_i}{x_i} = \frac{t}{t_x} = B, \qquad S^2_{b^*} = V(b_j^*) = \sum_{i=1}^{N} \psi_i\, (b_i - B)^2. \tag{7.5}$$

A single draw is therefore an unbiased estimate of B, and it is a precise one when the b_i are nearly the same for all units, that is, when y_i is nearly proportional to x_i.

Hansen–Hurwitz estimator of B. The HH estimator is the average of the n observations in $\mathcal{R}$, the **mean of the**

ratios:

$$\boxed{\hat{B}_{\text{HH}} = \bar{b}^* = \frac{1}{n}\sum_{j=1}^{n} b_j^*}. \tag{7.6}$$

Being the mean of n iid copies of b_j^*, it is unbiased, and its variance is that of one draw divided by n:

$$E(\hat{B}_{\text{HH}}) = B, \qquad V(\hat{B}_{\text{HH}}) = \frac{S_{b^*}^2}{n} = \frac{1}{nN}\sum_{i=1}^{N} w_i^x\,(b_i - B)^2, \qquad w_i^x = \frac{x_i}{\bar{x}_U} = N\psi_i. \tag{7.7}$$

Here w_i^x is the size of unit i relative to the average, so that $\psi_i = w_i^x/N$, and the variance is a sum of terms involving only w_i^x and $b_i - B$. There is no finite population correction, since the draws are independent. For the same reason, the sample variance of the b_j^* is an unbiased estimate of $S_{b^*}^2$, which gives

$$\boxed{\widehat{\text{SE}}(\hat{B}_{\text{HH}}) = \frac{s_{b^*}}{\sqrt{n}}}, \qquad s_{b^*}^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(b_j^* - \bar{b}^*\right)^2. \tag{7.8}$$

Mean and total. Multiplying by the known $\bar{x}_U$ and t_x, the estimator, its standard error and its estimated standard error all scale by the same constant:

$$\begin{array}{lll} \hat{\bar{y}}_{\text{HH}} = \bar{x}_U\,\hat{B}_{\text{HH}} = \bar{x}_U\,\bar{b}^*, & \text{SE}(\hat{\bar{y}}_{\text{HH}}) = \bar{x}_U\,\dfrac{S_{b^*}}{\sqrt{n}}, & \widehat{\text{SE}}(\hat{\bar{y}}_{\text{HH}}) = \bar{x}_U\,\dfrac{s_{b^*}}{\sqrt{n}}, \\ \hat{t}_{\text{HH}} = t_x\,\hat{B}_{\text{HH}} = t_x\,\bar{b}^*, & \text{SE}(\hat{t}_{\text{HH}}) = t_x\,\dfrac{S_{b^*}}{\sqrt{n}}, & \widehat{\text{SE}}(\hat{t}_{\text{HH}}) = t_x\,\dfrac{s_{b^*}}{\sqrt{n}}. \end{array} \tag{7.9}$$

Section 7.2.4 writes $\hat{t}_{\text{HH}}$ directly in terms of the selection probabilities ψ_i.

Variance of the mean and total. The variances of $\hat{\bar{y}}_{\text{HH}}$ and $\hat{t}_{\text{HH}}$ are $\bar{x}_U^2$ and t_x^2 times $V(\hat{B}_{\text{HH}})$. The variance is small when all the ratios b_i are close to B, that is, when y_i is nearly proportional to x_i.

7.2.2 Computation Example: Revenue from AI Token Usage

An AI company has $N = 6$ enterprise customers. From its usage logs it knows, for every customer, the number of tokens x_i (in millions) the customer processed last month, but the revenue y_i (in thousands of dollars) that the customer generated has to be audited from the billing records, which is costly. Revenue is roughly proportional to token usage, so we draw $n = 5$ customers with replacement, with probabilities $\psi_i = x_i/t_x$ proportional to the tokens, and estimate the revenue per million tokens B, the mean revenue per customer $\bar{y}_U$, and the total revenue t.

Customer	x_i (million tokens)	$\psi_i = x_i/t_x$	y_i (revenue, \$ thousand)	$b_i = y_i/x_i$
A	2	0.02	11	5.5000
B	5	0.05	24	4.8000
C	8	0.08	41	5.1250
D	15	0.15	70	4.6667
E	30	0.30	155	5.1667
F	40	0.40	190	4.7500
Total	100	1.00	491	
True $B = \sum y_i / \sum x_i$				4.9100

Customer	x_i (million tokens)	$\psi_i = x_i/t_x$	y_i (revenue, \$ thousand)	$b_i = y_i/x_i$

The y_i and b_i columns are unknown to the company before the audit; they are shown here so that we can check the estimates. The largest customer F, with $\psi_F = 0.4$, is likely to be drawn more than once; in our sample it is drawn twice:

$$\mathcal{R} = \{F,\ E,\ F,\ D,\ C\}. \tag{7.10}$$

Both copies of F are kept: each is a separate draw, so each is a separate observation b_j^*. Only these customers are audited.

Draw j	Customer	x_j^*	y_j^*	$b_j^* = y_j^*/x_j^*$	$b_j^* - \bar{b}^*$	$(b_j^* - \bar{b}^*)^2$
1	F	40	190	4.7500	-0.1417	0.020069
2	E	30	155	5.1667	0.2750	0.075625
3	F	40	190	4.7500	-0.1417	0.020069
4	D	15	70	4.6667	-0.2250	0.050625
5	C	8	41	5.1250	0.2333	0.054444
	Sum			24.4583	0.0000	0.220833

Step 1: HH estimate of B, the mean of the ratios. From the column b_j^*,

$$\hat{B}_{\text{HH}} = \bar{b}^* = \frac{1}{5}\sum_{j=1}^{5} b_j^* = \frac{24.4583}{5} = 4.8917. \tag{7.11}$$

Step 2: the HH estimates of the mean and the total. Multiply by the known mean and total of the tokens, $\bar{x}_U = 16.667$ and $t_x = 100$:

$$\hat{\bar{y}}_{\text{HH}} = \bar{x}_U\,\hat{B}_{\text{HH}} = 16.667 \times 4.8917 = 81.53, \qquad \hat{t}_{\text{HH}} = t_x\,\hat{B}_{\text{HH}} = 100 \times 4.8917 = 489.2. \tag{7.12}$$

The true values are $B = 4.91$, $\bar{y}_U = 81.83$ and $t = 491$.

Step 3: the sample variance of the ratios. From the last column,

$$s_{b^*}^2 = \frac{1}{n-1}\sum_{j=1}^{5}\left(b_j^* - \bar{b}^*\right)^2 = \frac{0.220833}{4} = 0.055208, \qquad s_{b^*} = 0.23496. \tag{7.13}$$

Step 4: the estimated standard errors.

$$\begin{aligned}
\widehat{\text{SE}}(\hat{B}_{\text{HH}}) &= \frac{s_{b^*}}{\sqrt{n}} = 0.10508, \\
\widehat{\text{SE}}(\hat{\bar{y}}_{\text{HH}}) &= \bar{x}_U\,\frac{s_{b^*}}{\sqrt{n}} = 16.667 \times 0.10508 = 1.75, \\
\widehat{\text{SE}}(\hat{t}_{\text{HH}}) &= t_x\,\frac{s_{b^*}}{\sqrt{n}} = 100 \times 0.10508 = 10.5.
\end{aligned} \tag{7.14}$$

For comparison, the theoretical standard error is $\mathrm{SE}(\hat{B}_{\mathrm{HH}}) = \sqrt{S_{b^*}^2/n} = \sqrt{0.05015/5} = 0.10015$, that is, $\mathrm{SE}(\hat{\bar{y}}_{\mathrm{HH}}) = 1.67$. The estimate 81.53 is 0.17 estimated standard errors below $\bar{y}_U = 81.83$.

The same HH estimates from the shared function `upswr_est` (Section 7.2.3 uses it for real data). The audited sample is stored as a data set `token`, with one row per draw (customer F appears twice) and columns `y` (revenue) and `x` (tokens). `upswr_est` takes these two columns and returns the estimate of B, with its SE and confidence interval; the mean and the total are $\bar{x}_U = t_x/N = 100/6$ and $t_x = 100$ times the estimate:

```
token <- data.frame(y = c(190, 155, 190, 70, 41),   # draws F, E, F, D, C
                    x = c(40, 30, 40, 15, 8))
token
```

```
    y  x
1 190 40
2 155 30
3 190 40
4  70 15
5  41  8
```

```
B_hat <- upswr_est(token$y, token$x, show.details = FALSE)$estimate   # estimate of B
B_hat |> format_est_gt("hh_B")
```

$\hat{B}_{\mathrm{HH}}$	$\mathrm{SE}(\hat{B}_{\mathrm{HH}})$	95% CI Lower	95% CI Upper
4.89	0.11	4.60	5.18

```
(B_hat * 100 / 6) |> format_est_gt("hh_ybar")   # mean of y: times xbar_U = t_x / N
```

$\hat{\bar{y}}_{\mathrm{HH}}$	$\mathrm{SE}(\hat{\bar{y}}_{\mathrm{HH}})$	95% CI Lower	95% CI Upper
81.53	1.75	76.67	86.39

```
(B_hat * 100) |> format_est_gt("hh_total")      # total of y: times t_x
```

$\hat{t}_{\mathrm{HH}}$	$\mathrm{SE}(\hat{t}_{\mathrm{HH}})$	95% CI Lower	95% CI Upper
489.17	10.51	459.99	518.34

7.2.3 Example: Physicians in U.S. Counties (`statepop.csv`)

The file `statepop.csv` contains data from an unequal-probability sample of 100 U.S. counties. Counties were chosen with probabilities proportional to their populations, $\psi_i = x_i/t_x$ with x_i the population of county i and $t_x = 255{,}077{,}536$ the population of the US. We want to estimate the total number of physicians t in the US. Because sampling was done **with replacement**, large counties like Los Angeles occur multiple times in the sample.

Table 7.1: Sample of U.S. county population and physician counts

state	county	popn	phys	psi
AL	Wilcox	13672	4	5.360×10^{-5}
AZ	Maricopa	2209567	4320	8.662×10^{-3}
AZ	Maricopa	2209567	4320	8.662×10^{-3}
AZ	Pinal	120786	61	4.735×10^{-4}
AR	Garland	76100	131	2.983×10^{-4}
AR	Mississippi	55060	48	2.159×10^{-4}
CA	Contra_Costa	840585	1761	3.295×10^{-3}
CA	Kern	587680	682	2.304×10^{-3}
CA	Los_Angeles	9053645	23677	3.549×10^{-2}
CA	Los_Angeles	9053645	23677	3.549×10^{-2}

Showing the first 10 of 100 rows; the full table is in the HTML version of the book.

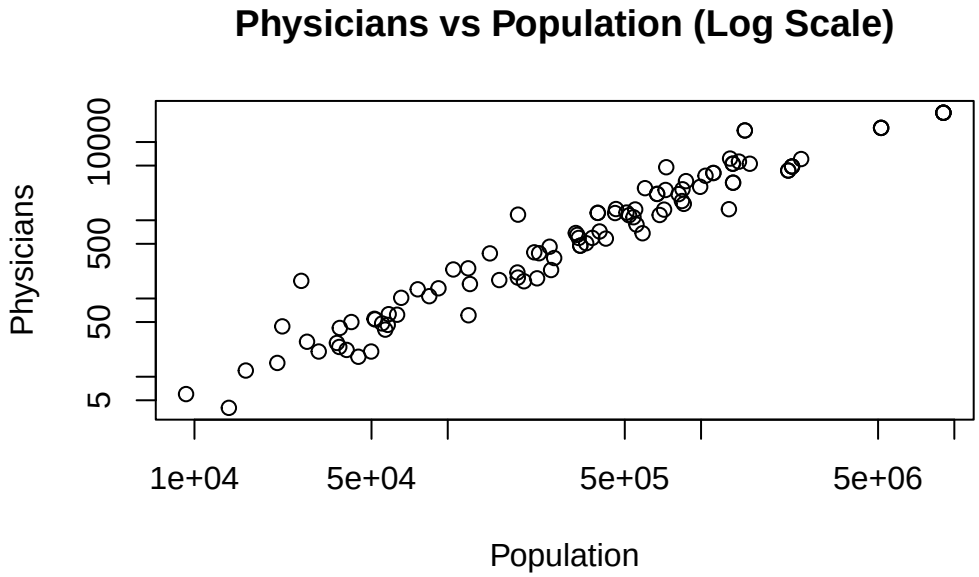

Figure 7.1: Physicians vs. population (log-log scale), sample of U.S. counties

The number of physicians is roughly proportional to the population, which is why this is a good size variable:

All three estimates come from the same quantity, the ratio $b_j^* = \text{phys}_j^*/\text{popn}_j^*$ of physicians per person of the county selected on draw j. To avoid tiny numbers we measure the population in thousands of people, so that b_j^* is the number of physicians per 1,000 people. The function `upswr_est` takes the physicians and populations of the 100 draws and estimates B; the mean and the total follow by multiplying by $\bar{x}_U = t_x/N$ and t_x (the number of counties is $N = 3141$).

Physicians per 1,000 people, $\hat{B}_{\text{HH}} = \bar{b}^*$. The working table lists the ratios b_j^* of the first draws, their deviations from $\hat{B}_{\text{HH}}$, and the squared deviations, whose sum gives $s^2_{b^*}$:

```
N <- 3141   # counties in the US
popn_in_thousands <- popn / 1000   # county populations in thousands
popn_US_in_thousands <- popn_US / 1000
res_phys_ratio_hh <- upswr_est(phys, popn_in_thousands)
res_phys_ratio_hh$table
```

UPSWR Hansen-Hurwitz Estimation: Working Table
$n = 100$

i	b_j^*	$\hat{B}_{\text{HH}}$	$b_j^* - \hat{B}_{\text{HH}}$	$(b_j^* - \hat{B}_{\text{HH}})^2$[1]
1	0.29	2.24	-1.94	3.78
2	1.96	2.24	-0.28	0.08
3	1.96	2.24	-0.28	0.08
4	0.51	2.24	-1.73	3.00
5	1.72	2.24	-0.51	0.26
6	0.87	2.24	-1.36	1.86
...	...	...	...	...
Sum	**223.58**	**2.24**	**0.00**	**260.81**

[1] $s^2_{b^*} = \sum_i (b_j^* - \hat{B}_{\text{HH}})^2/(n-1) = 2.63$.

$$\hat{B}_{\text{HH}} = \frac{\sum_i b_j^*}{n} = \frac{223.58}{100} = 2.24$$

$$\text{SE}(\hat{B}_{\text{HH}}) = \frac{s_{b^*}}{\sqrt{n}} = \frac{1.62}{\sqrt{100}} = 0.16$$

```
res_phys_ratio_hh$estimate |> format_est_gt("hh_B")
```

$\hat{B}_{\text{HH}}$	SE($\hat{B}_{\text{HH}}$)	95% CI Lower	95% CI Upper
2.24	0.16	1.91	2.56

Output Note: Each draw gives an unbiased estimate of the national number of physicians per 1,000 people, and their average is $\hat{B}_{\text{HH}} = 2.236$ physicians per 1,000 people, with a 95% interval of [1.914, 2.558].

Physicians per county, $\hat{\bar{y}}_{\text{HH}} = \bar{x}_U \hat{B}_{\text{HH}}$, with $\bar{x}_U = t_x/N$ the average population of a county:

```
res_phys_county <- res_phys_ratio_hh$estimate * popn_US_in_thousands / N   # xbar_U times B
res_phys_county |> format_est_gt("hh_ybar")
```

$\hat{\bar{y}}_{\text{HH}}$	SE($\hat{\bar{y}}_{\text{HH}}$)	95% CI Lower	95% CI Upper
181.57	13.18	155.41	207.72

Output Note: This is the estimate per 1,000 people times the average county population in thousands, $\bar{x}_U = 81.2$, and its standard error is multiplied by the same factor. It averages over *counties*, each weighted equally, so the many small rural counties pull it down to about 182 physicians per county.

Total physicians in the US, $\hat{t}_{\text{HH}} = t_x \hat{B}_{\text{HH}}$, with t_x the US population (in thousands):

```
res_phys_total <- res_phys_ratio_hh$estimate * popn_US_in_thousands   # t_x times B
res_phys_total |> format_est_gt("hh_total")
```

$\hat{t}_{\text{HH}}$	SE($\hat{t}_{\text{HH}}$)	95% CI Lower	95% CI Upper
570,304.30	41,401.23	488,155.27	652,453.32

The same total with the normalized size $\tilde{x}_i = \psi_i$ (Section 7.2.4), with $t_{\tilde{x}} = 1$; the population need not be in thousands now, since ψ_i has no units:

```
upswr_est(phys, psi, show.details = FALSE)$estimate |>
  format_est_gt("hh_total")
```

$\hat{t}_{\text{HH}}$	SE($\hat{t}_{\text{HH}}$)	95% CI Lower	95% CI Upper
570,304.30	41,401.23	488,155.27	652,453.32

Output Note: This is the estimate per 1,000 people times the US population in thousands, so it estimates the national total number of physicians: about 570,304 physicians in the US, with a 95% interval of [488,155, 652,453]. The three estimates carry the same information: the standard errors are in the same proportion as the estimates.

Comparison with estimates based on an SRS sample

Let's compare our UPSWR results against estimates based on a Simple Random Sample of 100 counties, including the ratio estimator that uses the county population at the estimation stage.

```r
# Read SRS sample, treating -99 as NA
counties <- read.csv("data/counties.csv", na.strings = "-99")
totpop <- 255077536
N <- 3141 # Total number of counties

# 1. Naive SRS estimate for total
res_counties_srs <- srs_est(counties$physician, N = N, estimate = "total", show.details =
↪  FALSE)
res_counties_srs$estimate |> format_est_gt("total")
```

$\hat{t}$	SE($\hat{t}$)	95% CI Lower	95% CI Upper
933,410.97	491,982.79	-42,789.62	1,909,611.56

```r
# 2. Ratio estimate for total using SRS data
res_counties_ratio <- ratio_est(counties$physician, counties$totpop, xbarU = totpop/N, N = N,
↪  estimate = "total")
res_counties_ratio$table
```

Ratio Estimation: Working Table
$n = 100$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	24.00	36,023.00	90.31	-66.31	4,397.47
2	44.00	73,524.00	184.33	-140.33	19,693.17
3	7.00	6,408.00	16.07	-9.07	82.18
4	11.00	19,261.00	48.29	-37.29	1,390.49
5	3.00	7,649.00	19.18	-16.18	261.69
6	327.00	188,377.00	472.28	-145.28	21,106.52
...	...	...	...	...	...
Sum	**29,717.00**	**11,853,116.00**	**29,717.00**	**0.00**	**17,054,519.52**

[1] $\hat{B} = \sum_i y_i / \sum_i x_i = 0.00$, $\hat{y}_i = \hat{B}\, x_i$, $s_e^2 = 1.72 \times 10^5$.
$\hat{t} = \hat{B}\, \bar{x}_U\, N = 0.00 \times 81,209.02 \times 3,141 = 6.40 \times 10^5$
$\text{SE}(\hat{t}) = \text{SE}(\hat{B})\, \bar{x}_U\, N = 3.45 \times 10^{-4} \times 81,209.02 \times 3,141 = 87,885.27$

```r
res_counties_ratio$estimate |> format_est_gt("total")
```

$\hat{t}$	SE($\hat{t}$)	95% CI Lower	95% CI Upper
639,506.03	87,885.27	465,122.60	813,889.46

```
# 3. Regression estimate for total using SRS data
res_counties_reg <- reg_est(counties$physician, counties$totpop, xbarU = totpop/N, N = N,
 ↪  estimate = "total")
res_counties_reg$table
```

Regression Estimation: Working Table
$n = 100$

i	y_i	x_i	$\hat{y}_i$ [1]	e_i	e_i^2
1	24.00	36,023.00	52.56	-28.56	815.88
2	44.00	73,524.00	163.74	-119.74	14,337.75
3	7.00	6,408.00	-35.23	42.23	1,783.70
4	11.00	19,261.00	2.87	8.13	66.09
5	3.00	7,649.00	-31.55	34.55	1,194.03
6	327.00	188,377.00	504.24	-177.24	31,413.03
...	...	...	...	...	...
Sum	**29,717.00**	**11,853,116.00**	**29,717.00**	**0.00**	**11,349,761.43**

[1] $\hat{B}_0 = -54.23$, $\hat{B}_1 = 0.00$, $\hat{y}_i = \hat{B}_0 + \hat{B}_1 x_i$, $s_e^2 = 1.16 \times 10^5$.
$\hat{t}_{reg} = N\bar{y}_{reg} = 3,141 \times 186.52 = 5.86 \times 10^5$
$\text{SE}(\hat{t}_{reg}) = N \times \text{SE}(\bar{y}_{reg}) = 3,141 \times 33.49 = 1.05 \times 10^5$

```
res_counties_reg$estimate |> format_est_gt("reg_total")
```

$\hat{t}_{reg}$	$\text{SE}(\hat{t}_{reg})$	95% CI Lower	95% CI Upper
585,870.60	105,177.42	377,149.44	794,591.76

Output Note: The true value is **532,638**. Notice how poorly the naive SRS estimate performs (predicting over 800,000) because it ignores county sizes. Both the SRS Ratio and SRS Regression methods perform much better by incorporating auxiliary population data, as does the HH estimator above, which uses the county sizes at the design stage.

Summary. The estimates of the total number of physicians in the US from all the methods above:

Table 7.2: Estimates of the total number of physicians in the US (true value 532,638)

Method	Esti-mate	SE	95% CI lower	95% CI upper	Estimate − true
Sample total, $N\bar{y}$ (SRS)	933,411	491,983	-42,790	1,909,612	400,773
Ratio estimator (SRS)	639,506	87,885	465,123	813,889	106,868
Regression estimator (SRS)	585,871	105,177	377,149	794,592	53,233
Hansen–Hurwitz, $t_x \hat{B}_{\text{HH}}$ (UPSWR)	570,304	41,401	488,155	652,453	37,666

Table 7.2: Estimates of the total number of physicians in the US (true value 532,638)

Method	Esti-mate	SE	95% CI lower	95% CI upper	Estimate − true

The naive SRS total ignores the county sizes: it is far too large, with a huge SE. The ratio and regression estimators use the county populations at the estimation stage and have much smaller SEs, and HH, which uses them at the design stage, has the smallest SE and the smallest error here, even though its sample is of the same size, 100 counties. Section 7.3 explains when using the size variable at the design stage pays off.

7.2.4 The HH Estimator of the Total in Terms of ψ_i

The selection probabilities need not come from a size variable: any $\psi_i > 0$ with $\sum_{i=1}^{N} \psi_i = 1$ define a UPSWR design. Let y_j^* and ψ_j^* be the study value and the selection probability of the unit selected on draw j. Dividing y_j^* by its selection probability gives an estimate of the total from that draw alone,

$$u_j = \frac{y_j^*}{\psi_j^*}, \qquad E(u_j) = \sum_{i=1}^{N} \psi_i \frac{y_i}{\psi_i} = \sum_{i=1}^{N} y_i = t, \tag{7.15}$$

and the u_j are iid, since the draws are independent. The HH estimator of the total is their average:

$$\boxed{\hat{t}_{\text{HH}} = \bar{u} = \frac{1}{n}\sum_{j=1}^{n} \frac{y_j^*}{\psi_j^*}, \qquad \widehat{\text{SE}}(\hat{t}_{\text{HH}}) = \frac{s_u}{\sqrt{n}}, \qquad s_u^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(\frac{y_j^*}{\psi_j^*} - \hat{t}_{\text{HH}}\right)^2}, \tag{7.16}$$

with variance

$$V(\hat{t}_{\text{HH}}) = \frac{1}{n}\sum_{i=1}^{N} \psi_i \left(\frac{y_i}{\psi_i} - t\right)^2. \tag{7.17}$$

The mean is estimated by $\hat{t}_{\text{HH}}/N$. A unit with a small ψ_i is rarely drawn, but when it is, its value is multiplied by the large factor $1/\psi_i$; this is what makes u_j unbiased. This form is what the function `upswr_total` computes.

Equivalence with $\hat{B}_{\text{HH}}$. The formula is the estimator of Section 7.2.1 with the selection probabilities themselves as the size variable. Take the **normalized size**

$$\tilde{x}_i = \psi_i, \qquad \text{so that} \qquad t_{\tilde{x}} = \sum_{i=1}^{N} \tilde{x}_i = 1 \quad \text{and} \quad \frac{\tilde{x}_i}{t_{\tilde{x}}} = \psi_i. \tag{7.18}$$

The ratios and the population ratio for this size variable are

$$\tilde{b}_i = \frac{y_i}{\tilde{x}_i} = \frac{y_i}{\psi_i}, \qquad \tilde{B} = \frac{t}{t_{\tilde{x}}} = t, \tag{7.19}$$

so each observation $\tilde{b}_j^* = y_j^*/\psi_j^* = u_j$, and

$$\hat{\tilde{B}}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n} \tilde{b}_j^* = \hat{t}_{\text{HH}}, \qquad \hat{t}_{\text{HH}} = t_{\tilde{x}}\, \hat{\tilde{B}}_{\text{HH}} = \hat{\tilde{B}}_{\text{HH}}, \qquad s_{\tilde{b}^*} = s_u. \tag{7.20}$$

With a normalized size, the "ratio" $\tilde{B}$ *is* the total, and the estimator, its SE and its variance are those of Section 7.2.1. Conversely, when $\psi_i = x_i/t_x$, every draw gives $y_j^*/\psi_j^* = t_x\, y_j^*/x_j^* = t_x\, b_j^*$, so $\hat{t}_{\text{HH}} = t_x\, \bar{b}^*$ and $s_u = t_x\, s_{b^*}$, the formulas for the total in Section 7.2.1. Rescaling a size variable to total 1 therefore does not change the estimator of the total, since ψ_i depends on x_i only through x_i/t_x.

7.2.4.1 Example: Revenue from AI Token Usage, Using ψ_i

We repeat the AI token example of Section 7.2.2 with the formula in terms of ψ_i. The same sample $\mathcal{R} = \{F, E, F, D, C\}$ is used, but each draw now gives an estimate $u_j = y_j^*/\psi_j^*$ of the **total** revenue, and the HH estimate is their average.

Draw j	Customer	y_j^*	ψ_j^*	$u_j = y_j^*/\psi_j^*$	$u_j - \hat{t}_{\text{HH}}$	$(u_j - \hat{t}_{\text{HH}})^2$
1	F	190	0.40	475.0	-14.2	200.7
2	E	155	0.30	516.7	27.5	756.3
3	F	190	0.40	475.0	-14.2	200.7
4	D	70	0.15	466.7	-22.5	506.2
5	C	41	0.08	512.5	23.3	544.4
	Sum			2,445.8	0.0	2,208.3

Step 1: HH estimate of the total.

$$\hat{t}_{\text{HH}} = \bar{u} = \frac{1}{n}\sum_{j=1}^{n} \frac{y_j^*}{\psi_j^*} = \frac{2445.8}{5} = 489.2, \tag{7.21}$$

the same as $t_x \hat{B}_{\text{HH}} = 489.2$ in Section 7.2.2.

Step 2: its standard error. From the last column,

$$s_u^2 = \frac{1}{n-1}\sum_{j=1}^{5} \left(u_j - \hat{t}_{\text{HH}}\right)^2 = \frac{2208.3}{4} = 552.1, \qquad \widehat{\text{SE}}(\hat{t}_{\text{HH}}) = \frac{s_u}{\sqrt{n}} = 10.5, \tag{7.22}$$

again equal to $t_x\, s_{b^*}/\sqrt{n} = 10.5$ (here $u_j = t_x\, b_j^*$).

Step 3: the mean and B. Divide by N and by t_x: $\hat{\bar{y}}_{\text{HH}} = \hat{t}_{\text{HH}}/N = 81.53$ and $\hat{B}_{\text{HH}} = \hat{t}_{\text{HH}}/t_x = 4.8917$.

The same estimates from `upswr_est`, with the normalized size $\tilde{x}_i = \psi_i$ of Section 7.2.4, whose total is $t_{\tilde{x}} = 1$: the ratios are the u_j, and their mean $\hat{\tilde{B}}_{\text{HH}}$ is $\hat{t}_{\text{HH}}$ itself. So `xbar_pop` and `tx_pop` are left out, and the call returns the estimate of the total; the estimates of $\bar{y}_U$ and B follow by dividing the estimate and its SE by $N = 6$ and by $t_x = 100$:

```
token$psi <- token$x / 100   # psi_i = x_i / t_x
t_hat <- upswr_est(token$y, token$psi, show.details = FALSE)$estimate   # estimate of t
t_hat |> format_est_gt("hh_total")
```

$\hat{t}_{\text{HH}}$	$\text{SE}(\hat{t}_{\text{HH}})$	95% CI Lower	95% CI Upper
489.17	10.51	459.99	518.34

```
(t_hat / 6) |> format_est_gt("hh_ybar")        # mean y: divide by N
```

$\hat{\bar{y}}_{\text{HH}}$	$\text{SE}(\hat{\bar{y}}_{\text{HH}})$	95% CI Lower	95% CI Upper
81.53	1.75	76.67	86.39

```
(t_hat / 100) |> format_est_gt("hh_B")         # B: divide by t_x
```

$\hat{B}_{\text{HH}}$	$\text{SE}(\hat{B}_{\text{HH}})$	95% CI Lower	95% CI Upper
4.89	0.11	4.60	5.18

7.3 Comparison of the HH, Ratio, and Average Estimators of the Total

The HH estimator uses the size variable x_i at the **design** stage, through the selection probabilities ψ_i. With a simple random sample, the same information can instead be used at the **estimation** stage, by the ratio estimator, or ignored, by the sample average. This section compares the three estimators of the total t: their variances, a toy example in which each of them can win, a simulation with the `agpop` data, and an app.

7.3.1 Theory

The sample average and the ratio estimator use a simple random sample (SRS) $\mathcal{S}$ of n units drawn without replacement; HH uses n draws with replacement, with selection probabilities $\psi_i = x_i/t_x$. The ratio estimator and HH are best compared through the **residuals about the line through the origin**,

$$d_i = y_i - Bx_i, \qquad B = \frac{t}{t_x}, \qquad \sum_{i=1}^{N} d_i = t - Bt_x = 0. \tag{7.23}$$

- The **sample average** ignores x_i: $\hat{t}_{\text{avg}} = N\bar{y}$, with the SRS variance

$$V(\hat{t}_{\text{avg}}) = \left(1 - \frac{n}{N}\right)\frac{N^2 S_y^2}{n}, \qquad S_y^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(y_i - \bar{y}_U\right)^2. \tag{7.24}$$

- The **ratio estimator** (Section 5.3.1) is $\hat{t}_r = t_x\,\bar{y}/\bar{x}$, and its (large-$n$) variance is the SRS variance of the residuals, $(1-\frac{n}{N})\,N^2 S_d^2/n$ with $S_d^2 = \frac{1}{N-1}\sum_{i=1}^{N} d_i^2$. Since an SRS gives every unit the same selection probability $1/N$, it can be written in the same form as the HH variance below:

$$V(\hat{t}_r) \approx \left(1 - \frac{n}{N}\right)\frac{N}{N-1}\cdot\frac{1}{n}\sum_{i=1}^{N}\frac{d_i^2}{1/N} \approx \left(1 - \frac{n}{N}\right)\frac{1}{n}\sum_{i=1}^{N}\frac{d_i^2}{1/N}. \tag{7.25}$$

- The **HH estimator** is $\hat{t}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n} y_j^*/\psi_j^*$ (Section 7.2.4), with variance $\frac{1}{n}\sum_{i=1}^{N}\psi_i\,(y_i/\psi_i - t)^2$. With $\psi_i = x_i/t_x$, each one-draw deviation is a scaled residual,

$$\frac{y_i}{\psi_i} - t = \frac{t_x}{x_i}\left(y_i - \frac{t}{t_x}\,x_i\right) = \frac{d_i}{\psi_i}, \qquad \text{so} \qquad V(\hat{t}_{\text{HH}}) = \frac{1}{n}\sum_{i=1}^{N}\frac{d_i^2}{\psi_i}. \tag{7.26}$$

Table 7.3: Estimators of the total t from SRS and UPSWR samples of n units, with $d_i = y_i - Bx_i$.

Estimator	Estimator of t	Variance	x_i is used at	Unbiased?
Sample average (SRS)	$\hat{t}_{\text{avg}} = N\bar{y}$	$\left(1-\frac{n}{N}\right)\frac{N^2 S_y^2}{n}$	not used	yes
Ratio (SRS)	$\hat{t}_r = t_x\,\frac{\bar{y}}{\bar{x}}$	$\approx \left(1-\frac{n}{N}\right)\frac{1}{n}\sum_{i=1}^{N}\frac{d_i^2}{1/N}$	the **estimation** stage	slightly biased
HH (UPSWR)	$\hat{t}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n}\frac{y_j^*}{\psi_j^*}$	$\frac{1}{n}\sum_{i=1}^{N}\frac{d_i^2}{\psi_i}$	the **design** stage, $\psi_i = x_i/t_x$	exactly unbiased

For the mean, divide the estimators by N and the variances by N^2; for $B = t/t_x$, divide by t_x and t_x^2.

On a common scale. The ratio and HH variances differ only in the selection probability that divides d_i^2: $1/N$ for every unit in an SRS, ψ_i under UPSWR. Writing the sample average the same way, $N^2 S_y^2/n \approx \frac{1}{n}\sum_{i=1}^{N}(y_i - \bar{y}_U)^2/(1/N)$, and ignoring the finite population correction, each variance is $\frac{1}{n}$ times a sum over the units of one term, a squared deviation divided by the unit's selection probability:

Estimator	Term of unit i	Same, as $N\times$ (squared deviation) / (relative size)	Spread of y about
Sample average	$\frac{(y_i - \bar{y}_U)^2}{1/N}$	$\frac{N(y_i - \bar{y}_U)^2}{1}$	the flat line $\bar{y}_U$
Ratio	$\frac{d_i^2}{1/N}$	$\frac{N d_i^2}{1}$	the line $y = Bx$
HH	$\frac{d_i^2}{\psi_i}$	$\frac{N d_i^2}{x_i/\bar{x}_U} = \frac{N d_i^2}{w_i^x}$	the line $y = Bx$

Because $\psi_i = x_i/t_x = \frac{1}{N}\,x_i/\bar{x}_U$, the last column shows the contrast most directly: the three terms share the factor N and differ only in what divides the squared deviation, namely 1 for the two SRS estimators and the relative size $w_i^x = x_i/\bar{x}_U$ for HH.

Ratio versus average. Since $y_i - \bar{y}_U = d_i + B(x_i - \bar{x}_U)$, the two terms differ by $B(x_i - \bar{x}_U)$. When y_i is roughly proportional to x_i, the residuals d_i are small and the spread of y comes mostly from that of x, which the ratio estimator removes. When y_i is not related to x_i, the residual $d_i = (y_i - \bar{y}_U) - B(x_i - \bar{x}_U)$ *adds* the spread of x to that of y, and the sample average does better.

HH versus ratio. Both use the same squared residuals d_i^2; they differ only in what they are divided by, the equal

probability $1/N$ for the ratio estimator and ψ_i for HH. A unit larger than average ($\psi_i > 1/N$) has its residual **downweighted** under HH, and a smaller one upweighted. The reason is that HH gives equal weights to the draws because the large units are already drawn more often, whereas the SRS estimators give every unit the same chance. UPSWR therefore wins when the large residuals d_i belong to the large units, that is, when the spread of y_i about the line grows with x_i, as it does for county acreages. When the spread of y_i is the same for all x_i, the SRS ratio estimator does better, and when y_i is not related to x_i at all, all the d_i are large and both can be worse than the plain sample average. The toy example in Section 7.3.2 illustrates these cases.

When does UPS beat the ratio estimator? Ignoring the finite population correction, the difference of the two variances is a single sum,

$$V(\hat{t}_r) - V(\hat{t}_{\text{HH}}) \approx \frac{1}{n}\sum_{i=1}^{N} d_i^2 \left(N - \frac{1}{\psi_i}\right) = \frac{N}{n}\sum_{i=1}^{N} d_i^2 \left(1 - \frac{\bar{x}_U}{x_i}\right), \tag{7.27}$$

so HH has the smaller variance exactly when

$$\boxed{H_d = \frac{\sum_{i=1}^{N} d_i^2}{\sum_{i=1}^{N} d_i^2/x_i} > \bar{x}_U}, \tag{7.28}$$

that is, when the harmonic mean of the sizes, weighted by the squared residuals, exceeds their ordinary mean: the units with large residuals are larger than average. In terms of the features of the population, suppose the spread of y about the line $y = Bx$ grows like a power of the size, $d_i^2 \approx \sigma^2 x_i^{\gamma}$. Then the sum above is proportional to $\sum_i x_i^{\gamma-1}(x_i - \bar{x}_U)$, a multiple of the covariance of $x_i^{\gamma-1}$ with x_i, whose sign is that of $\gamma - 1$:

Spread of y about the line $y = Bx$	γ	Better estimator
Constant: $\text{SD}(d_i)$ does not depend on x_i	0	ratio (SRS)
$\text{SD}(d_i) \propto \sqrt{x_i}$: $V(y_i) \propto x_i$, as for sums of x_i independent pieces	1	tie
$\text{SD}(d_i) \propto x_i$: a constant coefficient of variation, $\text{SD}(y_i) \propto Bx_i$	2	HH (UPSWR)

Figure 7.2 shows the three cases for one skewed population of sizes, with the same line $y = Bx$ and the same average spread about it; only the way the spread grows with x differs.

Many skewed populations with a natural size, such as revenues by customer size, acreages by county area, or counts by population, have a roughly constant coefficient of variation ($\gamma \approx 2$), which is why PPS works well for them. A plot of y against x from a census or a pilot sample shows which case applies: if the scatter about the line fans out in proportion to x, sample with PPS; if its width stays about the same, use an SRS with the ratio estimator. Three further points favour the SRS ratio estimator: its finite population correction $1 - n/N$, which matters when the sampling fraction is large; the repeated units of sampling with replacement (a πps design without replacement, Section 7.6, removes this); and a line with an intercept, when y is not proportional to x, where both rely on the wrong line and the regression estimator with an SRS is safer. In the toy example, H_d is 529 for the first population and 423 for the second, both above $\bar{x}_U = 400$, so HH beats the ratio estimator in both, as the working tables below confirm.

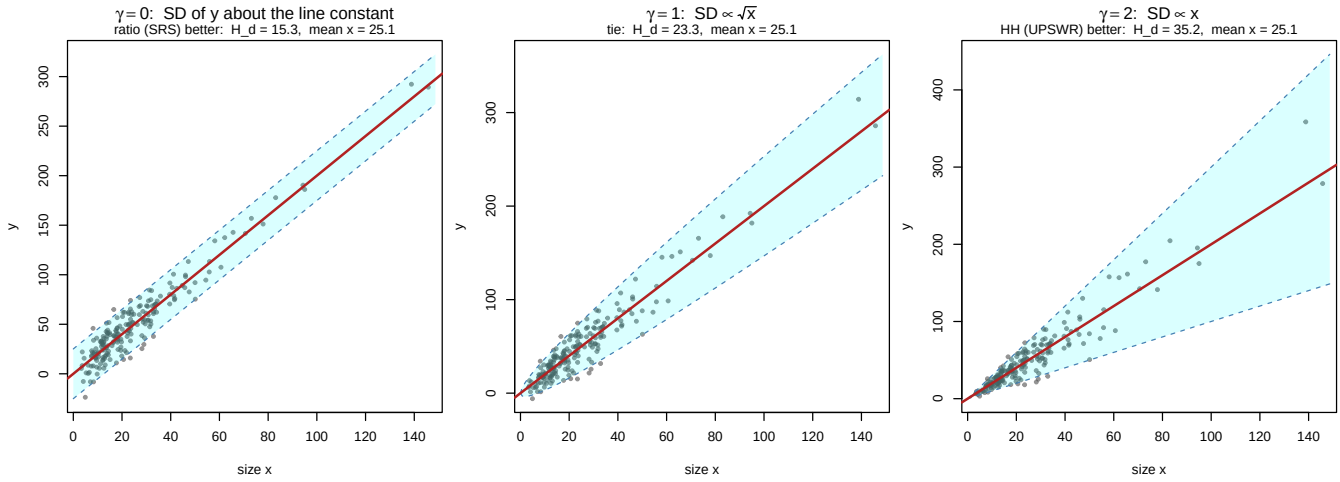

Figure 7.2: Three populations with the same sizes x_i and the same line $y = Bx$ (red), whose spread about the line (dashed band, ± 2 SD) is constant ($\gamma = 0$), grows like $\sqrt{x}$ ($\gamma = 1$), or grows like x ($\gamma = 2$). The subtitle compares H_d with the mean size $\bar{x}_U$: HH beats the SRS ratio estimator only when $H_d > \bar{x}_U$.

7.3.2 Toy Example: Four Supermarkets

A town has four supermarkets, with floor areas x_i from $100\,\mathrm{m}^2$ to $1000\,\mathrm{m}^2$, so $t_x = 1600$, and the selection probabilities are $\psi_i = x_i/t_x = 0.0625,\ 0.125,\ 0.1875,\ 0.625$, against $1/N = 0.25$ for each store in an SRS. We want to estimate the total monthly sales t of the four stores; $B = t/t_x$ is the sales per square metre, and the line $y = Bx$ gives the residuals $d_i = y_i - Bx_i$.

To compare the three estimators of t in Section 7.3, we work out their variances for a **single draw** ($n = 1$); for n draws, each is divided by n. For the two SRS estimators we ignore the finite population correction, so a single draw has the variance of one value drawn with equal probabilities $1/N$ (hence the divisor N rather than $N - 1$). The one-draw variance of each estimator is the sum over the stores of its term:

Estimator	Design	Term of store i
Sample average	SRS	$N(y_i - \bar{y}_U)^2$
Ratio	SRS	$N d_i^2$
HH	PPS, $\psi_i = x_i/t_x$	$d_i^2/\psi_i = N d_i^2/w_i^x$, with $w_i^x = x_i/\bar{x}_U$

For each population below, a single table works out the three variances side by side; the **Variance** row is the sum of each column.

Sales proportional to floor area. The monthly sales (thousands of dollars) are nearly proportional to the floor areas, as in the slides for this chapter. Here $t = 300$, $\bar{y}_U = 75$ and $B = 300/1600 = 0.1875$.

The working table for the three one-draw variances:

Store	x_i	y_i	$\psi_i = x_i/t_x$	$w_i^x = x_i/\bar{x}_U$	$d_i = y_i - Bx_i$	$N(y_i - \bar{y}_U)^2$	$N d_i^2$	$\frac{d_i^2}{\psi_i} = \frac{N d_i^2}{w_i^x}$
A	100	11	0.0625	0.25	-7.75	16,384	240.2	961
B	200	20	0.1250	0.50	-17.50	12,100	1,225.0	2,450
C	300	24	0.1875	0.75	-32.25	10,404	4,160.2	5,547
D	1,000	245	0.6250	2.50	57.50	115,600	13,225.0	5,290
Variance (Sum)	1,600	300	1.0000	4.00	0.00	154,488	18,850.5	14,248

for example because the large store D is a warehouse outlet with few customers, while the total is still $t = 300$, so $B = 0.1875$ again. The floor area is then a poor size variable, but we still draw with $\psi_i = x_i/t_x$:

Store	x_i	y_i	$\psi_i = x_i/t_x$	$w_i^x = x_i/\bar{x}_U$	$d_i = y_i - Bx_i$	$N(y_i - \bar{y}_U)^2$	Nd_i^2	$\frac{d_i^2}{\psi_i} = \frac{Nd_i^2}{w_i^x}$
A	100	60	0.0625	0.25	41.25	900	6,806.2	27,225
B	200	80	0.1250	0.50	42.50	100	7,225.0	14,450
C	300	90	0.1875	0.75	33.75	900	4,556.2	6,075
D	1,000	70	0.6250	2.50	-117.50	100	55,225.0	22,090
Variance (Sum)	1,600	300	1.0000	4.00	0.00	2,000	73,812.5	69,840

The ordering is reversed. The sales hardly vary, so the sample average has a tiny variance, 2,000. The line $y = Bx$ fits the sales badly, so all the residuals d_i are large, and the ratio estimator and HH, with variances 73,800 and 69,800, are more than 34 times worse than the sample average. Dividing by ψ_i instead of $1/N$ helps little: it downweights the large negative residual of store D, but upweights the positive residuals of the small stores, by $0.25/0.0625 = 4$ for store A. Unequal probability sampling is only as good as the size variable used to construct ψ_i: before using PPS, check that y_i is roughly proportional to x_i.

7.3.3 Simulation Study using `agpop` Data

Let's simulate 2000 random samples of 300 counties to see how UPSWR, with ψ_i proportional to the acres in 1987 (x_i), empirically compares to SRS for estimating the total acres in 1992 (y_i).

```
# Read population data
agpop <- acres_in_thousands(read.csv("data/agpop.csv", na.strings = "-99"))
agpop <- agpop |> filter(!is.na(acres92), !is.na(acres87))

N <- nrow(agpop)
total_acres92 <- sum(agpop$acres92)
total_acres87 <- sum(agpop$acres87)
acres87 <- agpop$acres87

# Storage vectors for simulation
est_srs300 <- rep(0, 2000)
est_srs300_reg_acres87 <- rep(0, 2000)
est_srs300_ratio_acres87 <- rep(0, 2000)
est_ups300_acres87 <- rep(0, 2000)

for (i in 1:2000) {
  # SRS Simulation
  srs300 <- sample(1:N, size = 300)
  acres92_srs300 <- agpop$acres92[srs300]
  acres87_srs300 <- agpop$acres87[srs300]
```

```r
  est_srs300[i] <- srs_est(acres92_srs300, N = N, estimate = "total", show.details =
↪  FALSE)$estimate[1]
  est_srs300_reg_acres87[i] <- reg_est(acres92_srs300, acres87_srs300,
                                       xbarU = total_acres87/N, N = N, estimate = "total",
                                       ↪  show.details = FALSE)$estimate[1]

  est_srs300_ratio_acres87[i] <- ratio_est(acres92_srs300, acres87_srs300,
                                       xbarU = total_acres87/N, N = N, estimate = "total",
                                       ↪  show.details = FALSE)$estimate[1]

  # UPSWR Simulation (Probability proportional to acres87)
  ups300 <- sample(1:N, size = 300, prob = acres87, replace = TRUE)
  acres92_ups300 <- agpop$acres92[ups300]

  psi <- (acres87 / sum(acres87))[ups300]
  est_ups300_acres87[i] <- upswr_total(acres92_ups300, psi, show.details = FALSE)$estimate[1]
}
```

Output Note: The red line represents the true total acreage, and the line inside each violin is the median. The naive SRS estimator has massive variance. The SRS ratio and regression estimators, which use the 1987 acres at the estimation stage, and UPSWR, which uses them at the design stage, drastically reduce the variance, centering tightly around the true mean; the R^2 in the legend is the proportion of the variance of the naive SRS average that each method removes.

7.3.4 Shinylive App Comparing SRS and UPSWR

The same simulation runs as a standalone app: SRS (average and ratio) with UPSWR (Hansen–Hurwitz) and UPSWO (Horvitz–Thompson). It opens in a new tab, together with its documentation and the full list of apps. Its "Variance reduction vs. SRS average" legend reports the variances of the ratio, HH and HT estimators, simulated and theoretical, relative to the plain SRS average, and illustrates the comparison of the variances in Section 7.3. The app describes all the estimators through the ratios b_i, and for the UPSWO estimator it shows the coverage of the confidence intervals based on the Hájek and the with-replacement variance estimators (Section 7.6).

7.4 One-Stage Cluster Sampling with PPS

7.4.1 Theory

In cluster sampling, the units of the HH estimator are **clusters** (primary sampling units, psus), and the study value of unit i is the **cluster total** $t_i = \sum_{j=1}^{M_i} y_{ij}$. In a one-stage design, every element (secondary sampling unit, ssu) of a selected cluster is observed, so t_i is known for the sampled clusters. The auxiliary variable is the **cluster size** M_i, known for all clusters. So the notation of Section 7.2.1 changes as

$$y_i \to t_i, \qquad x_i \to M_i, \qquad t_x \to M_0 = \sum_{i=1}^{N} M_i, \qquad \bar{x}_U \to \bar{M}_U = \frac{M_0}{N}, \qquad \psi_i = \frac{M_i}{M_0}. \tag{7.29}$$

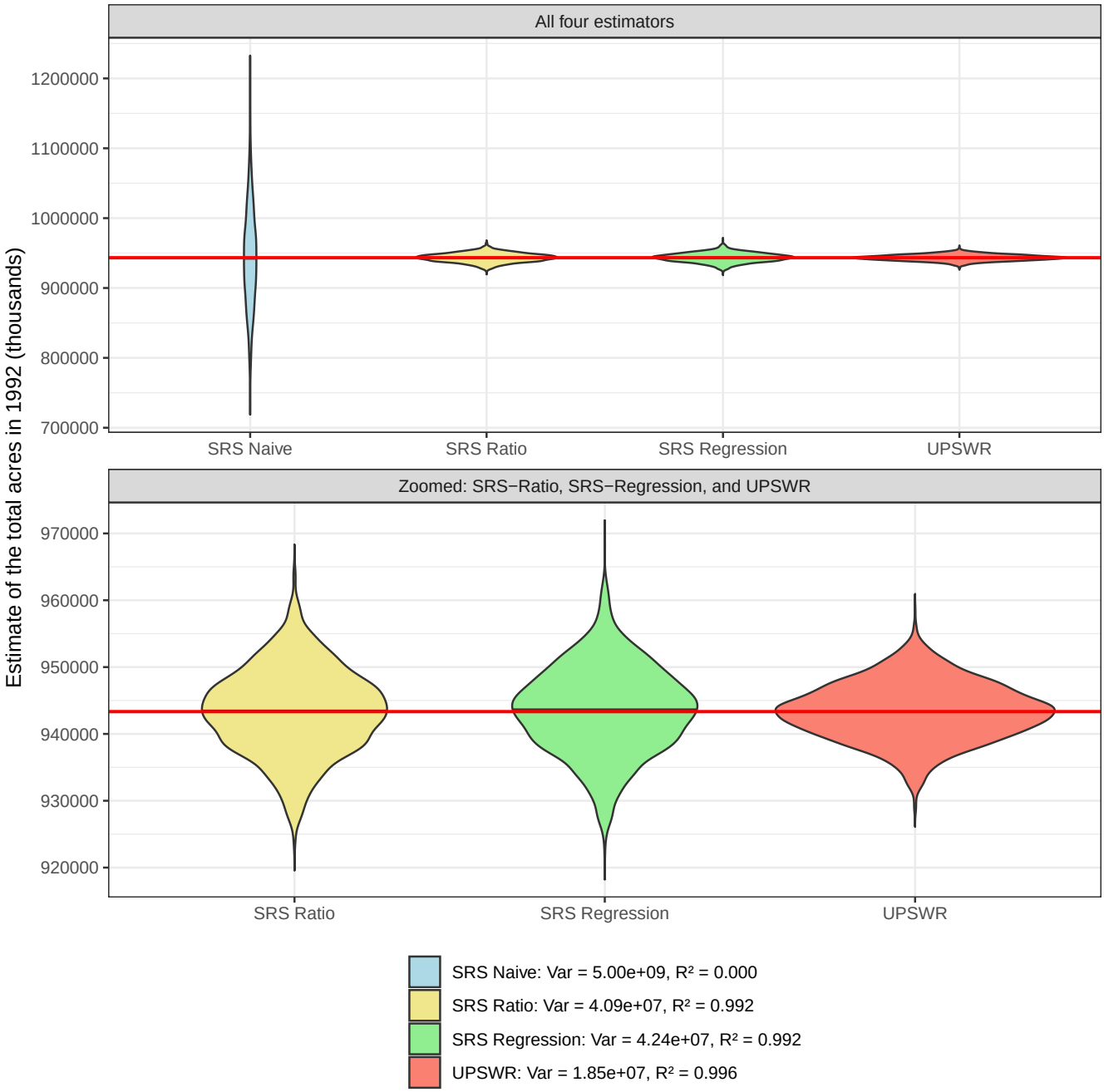

Figure 7.3: Estimator distribution: naive SRS vs. SRS ratio vs. SRS regression vs. UPSWR (2000 simulations), full and zoomed views. The legend gives the variance of each estimator and its R^2 relative to the naive SRS average.

The ratio of cluster i is

$$b_i = \frac{t_i}{M_i} = \bar{y}_i, \tag{7.30}$$

the **mean per element (ssu-level mean)** of cluster i, and $B = t/M_0$ is the mean per element of the whole population, which we denote also by $\bar{y}_U$. Let $b_j^* = \bar{y}_j^*$ be the mean per element of the cluster selected on draw j. Then everything in Section 7.2.1 applies, with $\hat{B}_{\text{HH}}$ written in the cluster notation as $\hat{\bar{y}}_{\text{HH}}$:

$$\boxed{\hat{\bar{y}}_{\text{HH}} = \hat{B}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n} \bar{y}_j^*}, \qquad \widehat{\text{SE}}(\hat{\bar{y}}_{\text{HH}}) = \frac{s_{\bar{y}^*}}{\sqrt{n}}, \qquad s_{\bar{y}^*}^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(\bar{y}_j^* - \hat{\bar{y}}_{\text{HH}}\right)^2. \tag{7.31}$$

That is, the estimate of the mean per element is the plain average of the means of the sampled clusters: large clusters are selected more often, and each selected cluster is then weighted equally. Multiplying by the known constants gives the estimates of the three quantities of interest, with their estimated SEs multiplied by the same constants:

Quantity	Estimate	Estimated SE
Mean per element (ssu-level mean), $\bar{y}_U = B = t/M_0$	$\hat{\bar{y}}_{\text{HH}}$	$s_{\bar{y}^*}/\sqrt{n}$
Mean per cluster (psu-level mean of the totals), $\bar{t}_U = t/N$	$\hat{\bar{t}}_{\text{HH}} = \bar{M}_U\,\hat{\bar{y}}_{\text{HH}}$	$\bar{M}_U\, s_{\bar{y}^*}/\sqrt{n}$
Total, $t = \sum_{i=1}^{N} t_i$	$\hat{t}_{\text{HH}} = M_0\,\hat{\bar{y}}_{\text{HH}}$	$M_0\, s_{\bar{y}^*}/\sqrt{n}$

The function `cluster_upswr(ti, Mi, N, M0, estimate)` computes these estimates. It simply calls `upswr_est` with $y = t$ (the cluster totals) and $x = M$ (the cluster sizes), $\bar{x}_U = \bar{M}_U$ and $t_x = M_0$, and reports the working table and the results in the cluster notation; `estimate` is `"ybar"` (mean per element), `"tbar"` (mean per cluster) or `"total"`:

```
cluster_upswr <- function (ti, Mi, N, M0, estimate = c ("ybar", "tbar", "total"),
                           show.details = TRUE, cluster = NULL, t_label = "t_j^*",
                           decimals = 4)
{
    estimate <- match.arg (estimate)
    Mbar_U <- M0 / N
    res <- upswr_est (ti, Mi, show.details = FALSE)      # estimates ybar_U
    res$estimate <- res$estimate * switch (estimate, ybar = 1, tbar = Mbar_U, total = M0)
    if (!show.details) return (res)

    ## working table on the scale of the cluster means ybar_j*
    n <- length (ti)
    ybar_j <- ti / Mi
    ybar_hh <- mean (ybar_j)
    dev <- ybar_j - ybar_hh
    s2 <- sum (dev^2) / (n - 1)
    se_ybar <- sqrt (s2 / n)
    working <- data.frame (
        j    = c (seq_len (n), "Sum"),
```

```
        cl   = c (if (is.null (cluster)) rep ("", n) else as.character (cluster), ""),
        M    = c (Mi, NA),
        t    = c (ti, NA),
        ybar = c (ybar_j, sum (ybar_j)),
        dev  = c (dev, sum (dev)),
        dev2 = c (dev^2, sum (dev^2))
    )
    working <- truncate_working (working, id_col = "j", head_n = 10)

    yh <- "\\hat{\\bar y}_{\\mathrm{HH}}"
    th <- "\\hat{\\bar t}_{\\mathrm{HH}}"
    Th <- "\\hat t_{\\mathrm{HH}}"
    fn <- function (x, d = decimals) fmt_num (x, decimals = d)
    table <- gt (working) |>
        tab_header (title = "UPSWR Cluster Sampling (Hansen-Hurwitz): Working Table",
                    subtitle = md (sprintf ("$n = %d$ draws, $N = %s$, $M_0 = %s$, $\\bar M_U =
                     ↪  M_0/N = %s$",
                                            n, fn (N, 0), fn (M0, 0), fn (Mbar_U, 2)))) |>
        cols_label (
            j    = md ("$j$"),
            cl   = "Cluster",
            M    = md ("$M_j^*$"),
            t    = md (sprintf ("$%s$", t_label)),
            ybar = md (sprintf ("$\\bar y_j^* = %s/M_j^*$", t_label)),
            dev  = md (sprintf ("$\\bar y_j^* - %s$", yh)),
            dev2 = md (sprintf ("$(\\bar y_j^* - %s)^2$", yh))
        ) |>
        fmt_number (columns = c ("M", "t"), decimals = if (all (ti == round (ti))) 0 else 1) |>
        fmt_auto (data = working, columns = c ("ybar", "dev", "dev2"), decimals = decimals) |>
        sub_missing (missing_text = "") |>
        tab_style (style = cell_text (weight = "bold"),
                   locations = cells_body (rows = j == "Sum")) |>
        tab_footnote (
            footnote = md (sprintf ("$s_{\\bar y^*}^2 = \\sum_j (\\bar y_j^* - %s)^2/(n-1) =
             ↪  %s$, $s_{\\bar y^*} = %s$.",
                                    yh, fn (s2), fn (sqrt (s2)))),
            locations = cells_column_labels (columns = dev2)) |>
        tab_source_note (md (sprintf (
            "Mean per element: $%s = \\dfrac{\\sum_j \\bar y_j^*}{n} = \\dfrac{%s}{%d} = %s$,
             ↪  $\\mathrm{SE}(%s) = \\dfrac{s_{\\bar y^*}}{\\sqrt n} = %s$",
            yh, fn (sum (ybar_j)), n, fn (ybar_hh), yh, fn (se_ybar)))) |>
        tab_source_note (md (sprintf (
            "Mean per cluster: $%s = \\bar M_U\\,%s = %s \\times %s = %s$, $\\mathrm{SE}(%s) =
             ↪  %s$",
            th, yh, fn (Mbar_U, 2), fn (ybar_hh), fn (Mbar_U * ybar_hh, 2), th, fn (Mbar_U *
↪  se_ybar, 2)))) |>
        tab_source_note (md (sprintf (
            "Total: $%s = M_0\\,%s = %s \\times %s = %s$, $\\mathrm{SE}(%s) = %s$",
            Th, yh, fn (M0, 0), fn (ybar_hh), fn (M0 * ybar_hh, 2), Th, fn (M0 * se_ybar, 2))))
    if (is.null (cluster)) table <- cols_hide (table, "cl")
```

```
    list (estimate = res$estimate, table = table)
}
```

The estimator is most precise when the cluster means $\bar{y}_i$ vary little, that is, when t_i is nearly proportional to M_i. Note that this differs from the ratio estimator $\sum_{i\in\mathcal{S}} t_i / \sum_{i\in\mathcal{S}} M_i$ of cluster sampling with a simple random sample of clusters: here the cluster sizes have been used at the design stage, so a plain average replaces the ratio.

7.4.2 Example: Falls in Long-Term Care Homes

A health authority wants to estimate the number of falls over the past year of the residents of its $N = 40$ long-term care homes. From the registry it knows the number of residents M_i of every home, and $M_0 = 3{,}000$ residents in total, so $\bar{M}_U = M_0/N = 75$ residents per home. Visiting a home to review the incident reports of all its residents is costly, so $n = 8$ draws of homes are made with replacement with probabilities $\psi_i = M_i/M_0$ proportional to the number of residents. The sample is a one-stage cluster sample: every resident of a selected home is reviewed, and t_i is the number of falls in home i. (The data are made up for illustration.) Home H12 was drawn twice; a repeated draw is kept as a separate observation, and since all its residents are reviewed, it has the same t_i:

Table 7.4: Falls in the sampled long-term care homes, $n = 8$ draws

Draw j	Home	M_j^* (residents)	$\psi_j^* = M_j^*/M_0$	t_j^* (falls)	$\bar{y}_j^* = t_j^*/M_j^*$ (falls per resident)
1	H12	180	0.060	160	0.8889
2	H03	60	0.020	49	0.8167
3	H27	240	0.080	205	0.8542
4	H12	180	0.060	160	0.8889
5	H31	95	0.032	70	0.7368
6	H08	120	0.040	118	0.9833
7	H19	45	0.015	30	0.6667
8	H36	150	0.050	140	0.9333

We estimate $\bar{y}_U$, the number of falls per resident, with `cluster_upswr`, from the totals t_j^* and the sizes M_j^* of the sampled homes. The working table lists the cluster means $\bar{y}_j^* = t_j^*/M_j^*$, their deviations from $\hat{\bar{y}}_{\mathrm{HH}}$, and the squared deviations, and its notes give all three estimates:

```
res_falls_B <- cluster_upswr(ti, Mi, N = N, M0 = M0, estimate = "ybar", cluster = home)
res_falls_B$table
```

UPSWR Cluster Sampling (Hansen-Hurwitz): Working Table
$n = 8$ draws, $N = 40$, $M_0 = 3,000$, $\bar{M}_U = M_0/N = 75.00$

j	Cluster	M_j^*	t_j^*	$\bar{y}_j^* = t_j^*/M_j^*$	$\bar{y}_j^* - \hat{\bar{y}}_{\text{HH}}$	$(\bar{y}_j^* - \hat{\bar{y}}_{\text{HH}})^2$[1]
1	H12	180	160	0.8889	0.0428	0.0018
2	H03	60	49	0.8167	-0.0294	0.0009
3	H27	240	205	0.8542	0.0081	0.0001
4	H12	180	160	0.8889	0.0428	0.0018
5	H31	95	70	0.7368	-0.1093	0.0119
6	H08	120	118	0.9833	0.1372	0.0188
7	H19	45	30	0.6667	-0.1794	0.0322
8	H36	150	140	0.9333	0.0872	0.0076
Sum				**6.7688**	**0.0000**	**0.0752**

[1] $s^2_{\bar{y}^*} = \sum_j (\bar{y}_j^* - \hat{\bar{y}}_{\text{HH}})^2/(n-1) = 0.0107$, $s_{\bar{y}^*} = 0.1036$.

Mean per element: $\hat{\bar{y}}_{\text{HH}} = \dfrac{\sum_j \bar{y}_j^*}{n} = \dfrac{6.7688}{8} = 0.8461$, $\text{SE}(\hat{\bar{y}}_{\text{HH}}) = \dfrac{s_{\bar{y}^*}}{\sqrt{n}} = 0.0366$

Mean per cluster: $\hat{\bar{t}}_{\text{HH}} = \bar{M}_U \, \hat{\bar{y}}_{\text{HH}} = 75.00 \times 0.8461 = 63.46$, $\text{SE}(\hat{\bar{t}}_{\text{HH}}) = 2.75$

Total: $\hat{t}_{\text{HH}} = M_0 \, \hat{\bar{y}}_{\text{HH}} = 3,000 \times 0.8461 = 2,538.29$, $\text{SE}(\hat{t}_{\text{HH}}) = 109.91$

```
res_falls_B$estimate |> format_est_gt("hh_ybar")
```

$\hat{\bar{y}}_{\text{HH}}$	$\text{SE}(\hat{\bar{y}}_{\text{HH}})$	95% CI Lower	95% CI Upper
0.85	0.04	0.76	0.93

The number of falls per home is $\bar{M}_U = 75$ times as large, and the total number of falls is $M_0 = 3{,}000$ times as large:

```
res_falls_home <- cluster_upswr(ti, Mi, N = N, M0 = M0, estimate = "tbar", show.details =
↪  FALSE)
res_falls_home$estimate |> format_est_gt("hh_tbar")
```

$\hat{\bar{t}}_{\text{HH}}$	$\text{SE}(\hat{\bar{t}}_{\text{HH}})$	95% CI Lower	95% CI Upper
63.46	2.75	56.96	69.95

```
res_falls_total <- cluster_upswr(ti, Mi, N = N, M0 = M0, estimate = "total", show.details =
↪  FALSE)
res_falls_total$estimate |> format_est_gt("hh_total")
```

$\hat{t}_{\text{HH}}$	SE($\hat{t}_{\text{HH}}$)	95% CI Lower	95% CI Upper
2,538.29	109.91	2,278.39	2,798.20

Output Note: The estimated number of falls per resident is $\hat{\bar{y}}_{\text{HH}} = 0.846$ (SE 0.037), the plain average of the eight cluster means in the working table. A typical home has $\hat{\bar{t}}_{\text{HH}} = 63.5$ falls per year (SE 2.7), and the 40 homes together have about $\hat{t}_{\text{HH}} = 2,538$ falls (SE 110, 95% interval [2,278, 2,798]). The three statements carry the same information: the estimates and the standard errors are in the same proportion.

7.5 Two-Stage Cluster Sampling with PPS

7.5.1 Theory

In a two-stage design, we select n clusters with replacement with probabilities $\psi_i = M_i/M_0$, and then take a simple random sample of m_i elements from the M_i elements of the cluster **on every draw**. The cluster total t_i is no longer observed, so we cannot compute the cluster mean $b_i = t_i/M_i$ of Section 7.4.1. We estimate it from the subsample,

$$\bar{y}_i = \frac{1}{m_i}\sum_{j=1}^{m_i} y_{ij}, \qquad \hat{t}_i = M_i\,\bar{y}_i, \tag{7.32}$$

where $\bar{y}_i$ is the mean of the *subsample*. The one-stage formulas carry over with the subsample mean $\bar{y}^*_j$ of draw j in place of the cluster mean b^*_j:

$$\boxed{\hat{\bar{y}}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n}\bar{y}^*_j}, \qquad \widehat{\text{SE}}(\hat{\bar{y}}_{\text{HH}}) = \frac{s_{\bar{y}^*}}{\sqrt{n}}, \qquad s^2_{\bar{y}^*} = \frac{1}{n-1}\sum_{j=1}^{n}\left(\bar{y}^*_j - \hat{\bar{y}}_{\text{HH}}\right)^2, \tag{7.33}$$

and the estimates of the mean per element, the mean per cluster and the total are $\hat{\bar{y}}_{\text{HH}}$, $\hat{\bar{t}}_{\text{HH}} = \bar{M}_U\hat{\bar{y}}_{\text{HH}}$ and $\hat{t}_{\text{HH}} = M_0\hat{\bar{y}}_{\text{HH}}$, with their SEs multiplied by the same constants, as in the table of Section 7.4.1. Equivalently, each draw estimates the total by $\hat{t}^*_j/\psi^*_j = M_0\,\bar{y}^*_j$.

There are two differences from the one-stage case. First, a cluster that is drawn twice is subsampled **independently** on each draw, so the two draws give different subsample means $\bar{y}^*_j$, and both are kept as separate observations. Second, **no second-stage formula is needed for the standard error.** The n draws are independent, and the values $\bar{y}^*_j$ differ between draws for two reasons, the choice of the clusters and the choice of the elements within them. The sample variance $s^2_{\bar{y}^*}$ of the values $\bar{y}^*_j$ captures both, so the variance of the two-stage estimator is computed in exactly the same way as that of the one-stage estimator. This is a major simplification compared with two-stage SRS of clusters, where the variance has a between-cluster and a within-cluster piece. Of course, a larger subsample m_i makes $\bar{y}_i$ closer to t_i/M_i and reduces the variance, but it does so through $s^2_{\bar{y}^*}$.

Why the within-cluster variation is absorbed into $V(\bar{y}^*_j)$. Write $\bar{y}_{iU} = t_i/M_i$ for the true mean of cluster i, S^2_i for the variance of its M_i elements, and $V_{2i} = \left(1 - \frac{m_i}{M_i}\right)\frac{S^2_i}{m_i}$ for the variance of the subsample mean $\bar{y}_i$ of an SRS of m_i elements. The value $\bar{y}^*_j$ of draw j has two sources of randomness: which cluster i is drawn (with

probability ψ_i), and which m_i elements are subsampled from it. Conditioning on the cluster, $E(\bar{y}_j^* \mid i) = \bar{y}_{iU}$ and $V(\bar{y}_j^* \mid i) = V_{2i}$, so

$$E(\bar{y}_j^*) = \sum_{i=1}^{N} \psi_i \, \bar{y}_{iU} = \sum_{i=1}^{N} \frac{M_i}{M_0} \frac{t_i}{M_i} = \bar{y}_U, \tag{7.34}$$

and, by the law of total variance,

$$V(\bar{y}_j^*) = \underbrace{V(E[\bar{y}_j^* \mid i])}_{\text{which cluster}} + \underbrace{E(V[\bar{y}_j^* \mid i])}_{\text{which elements}} = \underbrace{\sum_{i=1}^{N} \psi_i \, (\bar{y}_{iU} - \bar{y}_U)^2}_{\text{between clusters}} + \underbrace{\sum_{i=1}^{N} \psi_i \, V_{2i}}_{\text{within clusters}}. \tag{7.35}$$

In one-stage sampling $V_{2i} = 0$ and only the between-cluster term remains. Because the n draws are independent and identically distributed, so are $\bar{y}_1^*, \ldots, \bar{y}_n^*$, and their sample variance is unbiased for the whole variance:

$$E\left(s^2_{\bar{y}^*}\right) = V(\bar{y}_j^*) = \text{between} + \text{within}, \qquad E\left[\hat{V}(\hat{\bar{y}}_{\text{HH}})\right] = E\left(\frac{s^2_{\bar{y}^*}}{n}\right) = \frac{V(\bar{y}_j^*)}{n} = V(\hat{\bar{y}}_{\text{HH}}). \tag{7.36}$$

The within-cluster variation is visible in the spread of the values $\bar{y}_j^*$: two draws of clusters with the same true mean still give different $\bar{y}_j^*$ because of the subsampling (home H12 in the example below gives 0.95 on one draw and 0.80 on the other). So no estimate of S_i^2 is needed, and the method works even with $m_i = 1$. This relies on three conditions: the clusters are drawn **with replacement**, a cluster drawn twice is **subsampled again, independently**, on each draw (reusing one subsample would make those two draws dependent), and the subsample mean is **unbiased** for the cluster mean, for example with an SRS within the cluster.

Compare with two-stage sampling of clusters without replacement. There, the cluster-level values are not iid: the between-cluster formula applied to the estimated totals carries the finite population correction $(1 - n/N)$, and captures only a fraction (about n/N) of the within-cluster variance, so the rest must be added back with an estimate of S_i^2. With replacement there is no finite population correction, and nothing is lost. The price is that $s^2_{\bar{y}^*}$ estimates only the **sum** of the two components: a single sample does not tell how much of the variance is due to the choice of the clusters and how much to the subsampling, which is what one would need to decide between more clusters and larger subsamples m_i.

7.5.2 Example: Falls in Long-Term Care Homes, Two Stages

Suppose the health authority of Section 7.4.2 cannot afford to review all the residents of a selected home. On each of the same $n = 8$ draws of homes, with probabilities $\psi_i = M_i/M_0$, a simple random sample of m_i residents is drawn, and the number of falls of each sampled resident over the past year is recorded. The data below are made up for illustration. Home H12 was drawn twice, and the two subsamples were drawn independently, so the two draws have different subsample means. The table gives the total number of falls of the m_j^* sampled residents, their mean $\bar{y}_j^*$, and the estimated total of the home $\hat{t}_j^* = M_j^* \bar{y}_j^*$:

Table 7.5: Two-stage sample of long-term care homes, $n = 8$ draws

Draw j	Home	M_j^*	$\psi_j^* = M_j^*/M_0$	m_j^*	Falls in subsample	$\bar{y}_j^*$	$\hat{t}_j^* = M_j^* \bar{y}_j^*$
1	H12	180	0.060	20	19	0.950	171.0
2	H03	60	0.020	20	15	0.750	45.0
3	H27	240	0.080	20	16	0.800	192.0
4	H12	180	0.060	20	16	0.800	144.0
5	H31	95	0.032	20	18	0.900	85.5
6	H08	120	0.040	20	21	1.050	126.0
7	H19	45	0.015	15	10	0.667	30.0
8	H36	150	0.050	20	17	0.850	127.5

The data enter `cluster_upswr` through the estimated totals $\hat{t}_j^*$ and the sizes M_j^*, whose ratios are the subsample means $\bar{y}_j^*$ (`t_label` names the totals column $\hat{t}_j^*$). The working table lists them, their deviations from $\hat{\bar{y}}_{\text{HH}}$ and the squared deviations:

```
res_falls2_B <- cluster_upswr(ti_hat2, Mi2, N = N, M0 = M0, estimate = "ybar", cluster = home2,
                              t_label = "\\hat t_j^*")
res_falls2_B$table
```

UPSWR Cluster Sampling (Hansen-Hurwitz): Working Table
$n = 8$ draws, $N = 40$, $M_0 = 3,000$, $\bar{M}_U = M_0/N = 75.00$

j	Cluster	M_j^*	$\hat{t}_j^*$	$\bar{y}_j^* = \hat{t}_j^*/M_j^*$	$\bar{y}_j^* - \hat{\bar{y}}_{\text{HH}}$	$(\bar{y}_j^* - \hat{\bar{y}}_{\text{HH}})^2$[1]
1	H12	180.0	171.0	0.9500	0.1042	0.0109
2	H03	60.0	45.0	0.7500	-0.0958	0.0092
3	H27	240.0	192.0	0.8000	-0.0458	0.0021
4	H12	180.0	144.0	0.8000	-0.0458	0.0021
5	H31	95.0	85.5	0.9000	0.0542	0.0029
6	H08	120.0	126.0	1.0500	0.2042	0.0417
7	H19	45.0	30.0	0.6667	-0.1792	0.0321
8	H36	150.0	127.5	0.8500	0.0042	0.0000
Sum				**6.7667**	**0.0000**	**0.1010**

[1] $s^2_{\bar{y}^*} = \sum_j (\bar{y}_j^* - \hat{\bar{y}}_{\text{HH}})^2/(n-1) = 0.0144$, $s_{\bar{y}^*} = 0.1201$.

Mean per element: $\hat{\bar{y}}_{\text{HH}} = \dfrac{\sum_j \bar{y}_j^*}{n} = \dfrac{6.7667}{8} = 0.8458$, $\text{SE}(\hat{\bar{y}}_{\text{HH}}) = \dfrac{s_{\bar{y}^*}}{\sqrt{n}} = 0.0425$

Mean per cluster: $\hat{\bar{t}}_{\text{HH}} = \bar{M}_U \, \hat{\bar{y}}_{\text{HH}} = 75.00 \times 0.8458 = 63.44$, $\text{SE}(\hat{\bar{t}}_{\text{HH}}) = 3.18$

Total: $\hat{t}_{\text{HH}} = M_0 \, \hat{\bar{y}}_{\text{HH}} = 3,000 \times 0.8458 = 2,537.50$, $\text{SE}(\hat{t}_{\text{HH}}) = 127.39$

```
res_falls2_B$estimate |> format_est_gt("hh_ybar")
```

$\hat{\bar{y}}_{\text{HH}}$	SE($\hat{\bar{y}}_{\text{HH}}$)	95% CI Lower	95% CI Upper
0.85	0.04	0.75	0.95

Multiplying by $\bar{M}_U = 75$ and $M_0 = 3{,}000$:

```
res_falls2_home <- cluster_upswr(ti_hat2, Mi2, N = N, M0 = M0, estimate = "tbar", show.details
↪  = FALSE)
res_falls2_home$estimate |> format_est_gt("hh_tbar")
```

$\hat{\bar{t}}_{\text{HH}}$	SE($\hat{\bar{t}}_{\text{HH}}$)	95% CI Lower	95% CI Upper
63.44	3.18	55.91	70.97

```
res_falls2_total <- cluster_upswr(ti_hat2, Mi2, N = N, M0 = M0, estimate = "total",
↪  show.details = FALSE)
res_falls2_total$estimate |> format_est_gt("hh_total")
```

$\hat{t}_{\text{HH}}$	SE($\hat{t}_{\text{HH}}$)	95% CI Lower	95% CI Upper
2,537.50	127.39	2,236.28	2,838.72

Output Note: The estimated number of falls per resident is $\hat{\bar{y}}_{\text{HH}} = 0.846$ (SE 0.042), the plain average of the eight subsample means. A typical home has 63.4 falls per year (SE 3.2), and the 40 homes together have about 2,538 falls (SE 127). The standard error is computed exactly as in the one-stage example, from the eight values $\bar{y}_j^*$ alone. Here it is a little larger than the one-stage SE of 0.037, as expected, since the subsample means carry the extra sampling noise of the second stage (for home H12, the two draws gave $\bar{y} = 0.95$ and 0.80); in a single sample this is not guaranteed.

7.6 UPS Without Replacement

7.6.1 Review of UPSWR and the Horvitz–Thompson Estimator

Review of UPSWR in terms of ψ_i. With replacement, each of the n draws selects unit i with probability ψ_i, and draw j gives the unbiased estimate y_j^*/ψ_j^* of the total t, since $E(y_j^*/\psi_j^*) = \sum_i \psi_i \, y_i/\psi_i = t$. The HH estimator averages the n estimates,

$$\hat{t}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n} \frac{y_j^*}{\psi_j^*}, \qquad \widehat{\text{SE}}(\hat{t}_{\text{HH}}) = \frac{s_u}{\sqrt{n}}, \qquad s_u^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(\frac{y_j^*}{\psi_j^*} - \hat{t}_{\text{HH}}\right)^2, \tag{7.37}$$

where the SE is easy because the n estimates are iid. With $\psi_i = x_i/t_x$, $\frac{y_j^*}{\psi_j^*} = t_x\, b_j^*$, which gives the estimator $\hat{t}_{\text{HH}} = t_x \bar{b}^*$ of Section 7.2.1. The drawback of UPSWR is that a large unit can be drawn several times, which is wasteful.

Without replacement. Now n **distinct** units are selected, and the design is described by the **inclusion probabilities**

$$\pi_i = P(\text{unit } i \in \mathcal{S}), \qquad \sum_{i=1}^{N} \pi_i = n, \tag{7.38}$$

for a sample of fixed size n. For a size variable x_i, we take $\pi_i = n\, x_i/t_x$ (inclusion probability proportional to size, πps), which is $n\psi_i$ for $\psi_i = x_i/t_x$. A unit with $n\, x_i/t_x > 1$ is included with certainty, $\pi_i = 1$, and the other probabilities are recomputed from the remaining units.

The Horvitz–Thompson (HT) estimator. Weighting each sampled unit by $1/\pi_i$ gives

$$\boxed{\hat{t}_{\text{HT}} = \sum_{i \in \mathcal{S}} \frac{y_i}{\pi_i}}. \tag{7.39}$$

It has the same form as the HH estimator with $n\psi_i$ replaced by π_i (the two coincide if no unit is drawn twice). *It is unbiased.* Let $I_i = 1$ if unit i is in the sample and 0 otherwise, so that $E(I_i) = P(I_i = 1) = \pi_i$. Then $\hat{t}_{\text{HT}} = \sum_{i=1}^{N} I_i\, y_i/\pi_i$, and

$$E(\hat{t}_{\text{HT}}) = \sum_{i=1}^{N} \frac{y_i}{\pi_i}\, E(I_i) = \sum_{i=1}^{N} \frac{y_i}{\pi_i}\, \pi_i = \sum_{i=1}^{N} y_i = t. \tag{7.40}$$

This needs only $\pi_i > 0$ for all units. The estimator of the mean per unit is $\hat{t}_{\text{HT}}/N$. When the inclusion probabilities are proportional to a size variable x_i, $\pi_i = n\, x_i/t_x$, the estimator of the ratio $B = t/t_x$ is, as for UPSWR, the HT total divided by t_x,

$$\hat{B}_{\text{HT}} = \frac{\hat{t}_{\text{HT}}}{t_x} = \frac{1}{t_x} \sum_{i \in \mathcal{S}} \frac{y_i}{\pi_i} = \frac{1}{n} \sum_{i \in \mathcal{S}} \frac{y_i}{x_i} = \frac{1}{n} \sum_{i \in \mathcal{S}} b_i, \tag{7.41}$$

the plain average of the ratios $b_i = y_i/x_i$ of the n sampled units, with $\widehat{\text{SE}}(\hat{B}_{\text{HT}}) = \widehat{\text{SE}}(\hat{t}_{\text{HT}})/t_x$. In cluster sampling, $y_i = t_i$ and $x_i = M_i$, so that $\hat{B}_{\text{HT}} = \hat{t}_{\text{HT}}/M_0$ is the mean per element (Section 7.4.1).

Variances. For a fixed sample size, with the joint inclusion probabilities $\pi_{ij} = P(i, j \in \mathcal{S})$,

$$V(\hat{t}_{\text{HT}}) = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\pi_i \pi_j - \pi_{ij}) \left(\frac{y_i}{\pi_i} - \frac{y_j}{\pi_j} \right)^2 \qquad \text{(Sen–Yates–Grundy)}, \tag{7.42}$$

which is estimated unbiasedly by summing over the sampled pairs only, each divided by π_{ij}:

$$\hat{V}_{\text{SYG}}(\hat{t}_{\text{HT}}) = \frac{1}{2} \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}} \frac{\pi_i \pi_j - \pi_{ij}}{\pi_{ij}} \left(\frac{y_i}{\pi_i} - \frac{y_j}{\pi_j} \right)^2. \tag{7.43}$$

The joint probabilities π_{ij} are hard to compute for most sampling methods, so approximations that need only π_i are used:

- **Hájek's approximation** (for designs with high entropy), obtained by $\pi_{ij} \approx \pi_i \pi_j \left[1 - \frac{(1-\pi_i)(1-\pi_j)}{\sum_k \pi_k (1-\pi_k)}\right]$ in the SYG estimator:

$$\hat{V}_{\mathrm{H}}(\hat{t}_{\mathrm{HT}}) = \frac{n}{n-1} \sum_{i \in \mathcal{S}} (1-\pi_i) \left(\frac{y_i}{\pi_i} - A \right)^2, \qquad A = \frac{\sum_{i \in \mathcal{S}} (1-\pi_i)\, y_i/\pi_i}{\sum_{i \in \mathcal{S}} (1-\pi_i)}. \tag{7.44}$$

- **With-replacement approximation:** treat the sample as n draws with $\psi_i = \pi_i / n$ and use the UPSWR formula,

$$\hat{V}_{\mathrm{WR}}(\hat{t}_{\mathrm{HT}}) = \frac{n}{n-1} \sum_{i \in \mathcal{S}} \left(\frac{y_i}{\pi_i} - \frac{\hat{t}_{\mathrm{HT}}}{n} \right)^2. \tag{7.45}$$

  It ignores the factors $1 - \pi_i$ that account for sampling without replacement, so it overestimates the variance, the more so the larger the π_i.

For the mean per unit, divide the estimate and the SE by N; for the ratio $\hat{B}_{\mathrm{HT}}$, divide by t_x.

7.6.2 Selecting a Sample with Given Inclusion Probabilities

The HT estimator needs only the inclusion probabilities π_i, but to use it we must be able to **draw** a sample without replacement whose inclusion probabilities are exactly the π_i we chose. This is less obvious than it looks, because many different designs have the same π_i (they differ in the joint probabilities π_{ij}), and the most natural procedure does not give them.

Step 1: compute the π_i. For a size variable x_i and a sample of size n, start with $\pi_i = n\, x_i / t_x$. A unit with $\pi_i \geq 1$ is taken with certainty, $\pi_i = 1$; remove the certainty units, and recompute $\pi_i = n' x_i / t'_x$ for the others, with n' the number of places left and t'_x the total size of the remaining units; repeat until all $\pi_i \leq 1$. The function `inclusionprobabilities(x, n)` of the R package `sampling` does this. For example, with $N = 6$ units of sizes $x = (40, 25, 15, 10, 6, 4)$, $t_x = 100$ and $n = 3$, unit 1 has $3 \times 40/100 = 1.2 > 1$, so it is a certainty unit, and the other two places are shared by units 2–6 with $\pi_i = 2x_i/60$:

```
library(sampling)
x_toy <- c(40, 25, 15, 10, 6, 4)
pik_toy <- inclusionprobabilities(x_toy, 3)
round(pik_toy, 4)
```

```
[1] 1.0000 0.8333 0.5000 0.3333 0.2000 0.1333
```

```
sum(pik_toy)
```

```
[1] 3
```

The natural procedure does not work. Drawing n times with probabilities $\psi_i = x_i / t_x$, and redrawing whenever a unit already in the sample comes up (draw by draw, without replacement), does **not** give $\pi_i = n\psi_i$. For example, with $N = 3$, $x = (2, 1, 1)$ and $n = 2$, the target is $\pi = (1, 0.5, 0.5)$, but unit 1 is in the sample with probability $P(1 \text{ on draw } 1) + P(1 \text{ on draw } 2) = 0.5 + 2 \times 0.25 \times \frac{0.5}{0.75} = 0.833$: when a small unit is drawn first, the large unit only has probability $2/3$ on the second draw.

Step 2: draw the sample. There are many methods for drawing a fixed-size sample without replacement with any given π_i (with $0 < \pi_i \leq 1$ and $\sum_i \pi_i = n$ an integer); see Tillé, *Sampling Algorithms* (Springer, 2006), for a survey. Two of them are simple and general.

- **Systematic πps sampling** (Madow, 1949). Lay the units along a line in some order, unit i occupying an interval of length π_i, so that the line has length $\sum_i \pi_i = n$. Draw one random start $u \sim \text{Uniform}(0, 1)$, and select the units whose intervals contain the n points $u,\ u+1,\ \ldots,\ u+n-1$. Since the points are spaced 1 apart and an interval has length $\pi_i \leq 1$, it contains at most one point, and it contains one with probability exactly π_i. This is the same line as for UPSWR (Section 7.2.1), with n equally spaced points instead of n independent ones; see

  1.

- **The pivotal method** (Deville and Tillé, 1998). Process the probabilities two at a time. Take two units i and j whose current probabilities $a = \pi_i$ and $b = \pi_j$ are strictly between 0 and 1, and let them "fight" for their combined probability $a + b$:

  - if $a + b < 1$: with probability $\frac{a}{a+b}$ set $(\pi_i, \pi_j) \leftarrow (a + b, 0)$, otherwise $(0, a + b)$;
  - if $a + b \geq 1$: with probability $\frac{1-b}{2-a-b}$ set $(\pi_i, \pi_j) \leftarrow (1, a + b - 1)$, otherwise $(a + b - 1, 1)$.

  Each step decides at least one of the two units (its probability becomes 0 or 1), keeps the sum n, and keeps the **expected** value of every probability. For example, if $a + b < 1$, the new π_i has expectation $\frac{a}{a+b}(a+b) + \frac{b}{a+b} \cdot 0 = a$, and if $a+b \geq 1$ it has expectation $\frac{1-b}{2-a-b} \cdot 1 + \frac{1-a}{2-a-b}(a+b-1) = \frac{a(2-a-b)}{2-a-b} = a$. After at most $N - 1$ steps every probability is 0 or 1, the units with 1 form a sample of size n, and unit i is in it with probability equal to its original π_i.

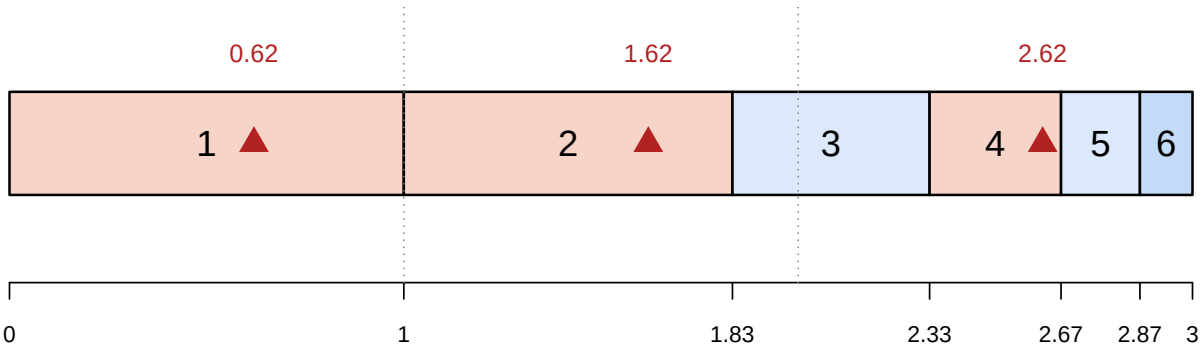

Figure 7.4: Systematic πps sampling of $n = 3$ of the six toy units: unit i occupies an interval of length π_i on $[0, 3]$, and the units containing the points $u,\ u+1,\ u+2$ (red triangles, here $u = 0.62$) are selected: units 1, 2 and 4.

The pivotal method is an instance of a **general procedure**, the *splitting method* of Deville and Tillé (1998): at each step, write the current vector of probabilities as a random choice between two vectors, $\pi = \lambda\, \pi^{(1)} + (1 - \lambda)\, \pi^{(2)}$, each with sum n and components in $[0, 1]$, and move to $\pi^{(1)}$ with probability λ, otherwise to $\pi^{(2)}$. The expected value of every component is kept, and the procedure stops when all components are 0 or 1. Almost all fixed-size methods can be described in this way; they differ in how the vector is split, which determines the joint probabilities π_{ij}. The main methods, all in the R package `sampling`, are:

Method	R function	Remarks
Systematic πps	`UPsystematic(pik)`	simplest; many $\pi_{ij} = 0$
Randomized systematic	`UPrandomsystematic(pik)`	random order of the units first, so fewer $\pi_{ij} = 0$
Pivotal method	`UPpivotal(pik)`	sequential, two units at a time
Brewer's method	`UPbrewer(pik)`	draw by draw with adjusted probabilities; $\pi_{ij} > 0$
Sampford's method	`UPsampford(pik)`	rejective: redraw until n distinct units; π_{ij} computable
Tillé's elimination	`UPtille(pik)`	eliminates units one at a time
Maximum entropy (conditional Poisson)	`UPmaxentropy(pik)`	most "random" design with the π_i; used in Section 7.6.3
Poisson sampling	`UPpoisson(pik)`	each unit independently with probability π_i; random sample size

The choice matters for the variance estimation of Section 7.6.1: the SYG estimator needs $\pi_{ij} > 0$ for every pair, which fails for systematic sampling, and the Hájek approximation is accurate for designs of high entropy, such as maximum entropy, Brewer, Sampford and Tillé sampling. Systematic sampling is often used in practice for its simplicity, with a with-replacement or Hájek variance estimate.

Implementing the two methods. The following functions implement systematic πps sampling and the pivotal method directly from their descriptions, and each returns the indices of the sampled units. We draw 20,000 samples with each method from the toy population and compare the proportion of samples that contain each unit with the target π_i:

```
## systematic pi-ps: points u, u+1, ..., u+n-1 on the cumulated pi_i
sys_pps <- function(pik) {
  n <- round(sum(pik))
  cum <- c(0, cumsum(pik))
  u <- runif(1)
  findInterval(u + 0:(n - 1), cum, left.open = TRUE)
}

## pivotal method: two undecided units fight for their combined probability
pivotal_pps <- function(pik, eps = 1e-9) {
  repeat {
    open <- which(pik > eps & pik < 1 - eps)   # undecided units
    if (length(open) < 2) break
    i <- open[1]; j <- open[2]
    a <- pik[i]; b <- pik[j]
    if (a + b < 1) {
      if (runif(1) < a / (a + b)) { pik[i] <- a + b; pik[j] <- 0 }
      else                        { pik[i] <- 0;     pik[j] <- a + b }
    } else {
      if (runif(1) < (1 - b) / (2 - a - b)) { pik[i] <- 1;         pik[j] <- a + b - 1 }
      else                                  { pik[i] <- a + b - 1; pik[j] <- 1 }
    }
```

```
  }
  which(pik > 1 - eps)
}

set.seed(348)
sys_pps(pik_toy)
```

```
[1] 1 2 4
```

```
pivotal_pps(pik_toy)
```

```
[1] 1 3 4
```

```
R_sel <- 20000
freq <- function(draw) tabulate(unlist(replicate(R_sel, draw(pik_toy), simplify = FALSE)),
 ↪  length(pik_toy)) / R_sel
knitr::kable(rbind(`target pi_i` = pik_toy, systematic = freq(sys_pps), pivotal =
 ↪  freq(pivotal_pps),
                   `UPbrewer` = freq(function(p) which(UPbrewer(p) == 1))),
             digits = 3, col.names = paste("unit", 1:6))
```

Table 7.6: Target inclusion probabilities and the proportion of 20,000 samples containing each unit

	unit 1	unit 2	unit 3	unit 4	unit 5	unit 6
target pi_i	1	0.833	0.500	0.333	0.200	0.133
systematic	1	0.836	0.502	0.332	0.198	0.132
pivotal	1	0.836	0.497	0.336	0.200	0.131
UPbrewer	1	0.838	0.498	0.331	0.200	0.133

All three methods reproduce the target π_i up to simulation error. They are different designs, however, as the joint inclusion probabilities show. For systematic sampling (in this order of the units) they are, from `UPsystematicpi2`:

```
knitr::kable(UPsystematicpi2(pik_toy), digits = 3, col.names = paste("unit", 1:6))
```

Table 7.7: Joint inclusion probabilities π_{ij} of systematic πps sampling for the toy population (diagonal: π_i)

unit 1	unit 2	unit 3	unit 4	unit 5	unit 6
1.000	0.833	0.500	0.333	0.200	0.133
0.833	0.833	0.333	0.333	0.167	0.000
0.500	0.333	0.500	0.000	0.033	0.133
0.333	0.333	0.000	0.333	0.000	0.000
0.200	0.167	0.033	0.000	0.200	0.000
0.133	0.000	0.133	0.000	0.000	0.133

Several pairs, such as units 3 and 4, are never selected together ($\pi_{ij} = 0$): their intervals, $(1.833, 2.333]$ and $(2.333, 2.667]$, are adjacent with a total length $0.833 < 1$, so two points spaced 1 apart cannot fall in both. For these pairs the SYG variance estimator cannot be computed, which is why a with-replacement or Hájek approximation is used with systematic sampling, or the order of the units is randomized first.

7.6.3 Example: Acres of Farmland in Counties (`agpop.csv`)

We draw a sample of $n = 100$ of the counties of `agpop` without replacement, with inclusion probabilities $\pi_i = 100\, x_i / t_x$ proportional to the acres in 1987 (x_i), and estimate the mean acres of farmland in 1992 (y_i) per county. (The few counties with no farmland in 1987 cannot be selected and are left out.) The sample is drawn by maximum entropy sampling, a fixed-size design with these inclusion probabilities, from the R package `sampling` (Section 7.6.2); a county with $100 x_i / t_x > 1$ would be included with certainty (none is here).

```
library(sampling)
agpop <- acres_in_thousands(read.csv("data/agpop.csv", na.strings = "-99"))
agpop <- agpop |> filter(!is.na(acres92), !is.na(acres87), acres87 > 0)
N <- nrow(agpop)
n <- 100
tx <- sum(agpop$acres87)
pik <- inclusionprobabilities(agpop$acres87, n)    # pi_i = n x_i / t_x

set.seed(348)
q_me <- UPMEqfromw(UPMEpiktildefrompik(pik), n)    # set-up of the sampling design
s <- which(UPMEsfromq(q_me) == 1)                  # indices of the n sampled counties
y <- agpop$acres92[s]; x <- agpop$acres87[s]; p <- pik[s]
```

The mean per county, estimated by HT with the two variance estimators (the plain sample mean ignores the inclusion probabilities and is shown for contrast), and the ratio $\hat{B}_{\text{HT}} = \hat{t}_{\text{HT}} / t_x$, the acres in 1992 per acre in 1987:

```
est_ht_hajek <- upswo_est(y, p, N, method = "mean", variance = "hajek")
est_ht_wr    <- upswo_est(y, p, N, method = "mean", variance = "wr")
est_B_hajek <- upswo_est(y, p, N, tx, method = "ratio", variance = "hajek")
est_B_wr    <- upswo_est(y, p, N, tx, method = "ratio", variance = "wr")
```

To judge the variance estimators, we repeat the sampling 2000 times, and compute the standard deviation of the 2000 HT estimates, which is the true standard error of the design, and the average of the estimated standard errors:

```
R <- 2000
sim <- matrix(0, R, 5, dimnames = list(NULL, c("est", "se_hajek", "se_wr", "cover_hajek",
 ↪  "cover_wr")))
for (i in 1:R) {
  si <- which(UPMEsfromq(q_me) == 1)
  yi <- agpop$acres92[si]
  eh <- upswo_est(yi, pik[si], N, method = "mean", variance = "hajek")
  ew <- upswo_est(yi, pik[si], N, method = "mean", variance = "wr")
  sim[i, ] <- c(eh["Est."], eh["S.E."], ew["S.E."],
```

```
              eh["ci.low"] <= mean(agpop$acres92) & mean(agpop$acres92) <= eh["ci.upp"],
              ew["ci.low"] <= mean(agpop$acres92) & mean(agpop$acres92) <= ew["ci.upp"])
}
```

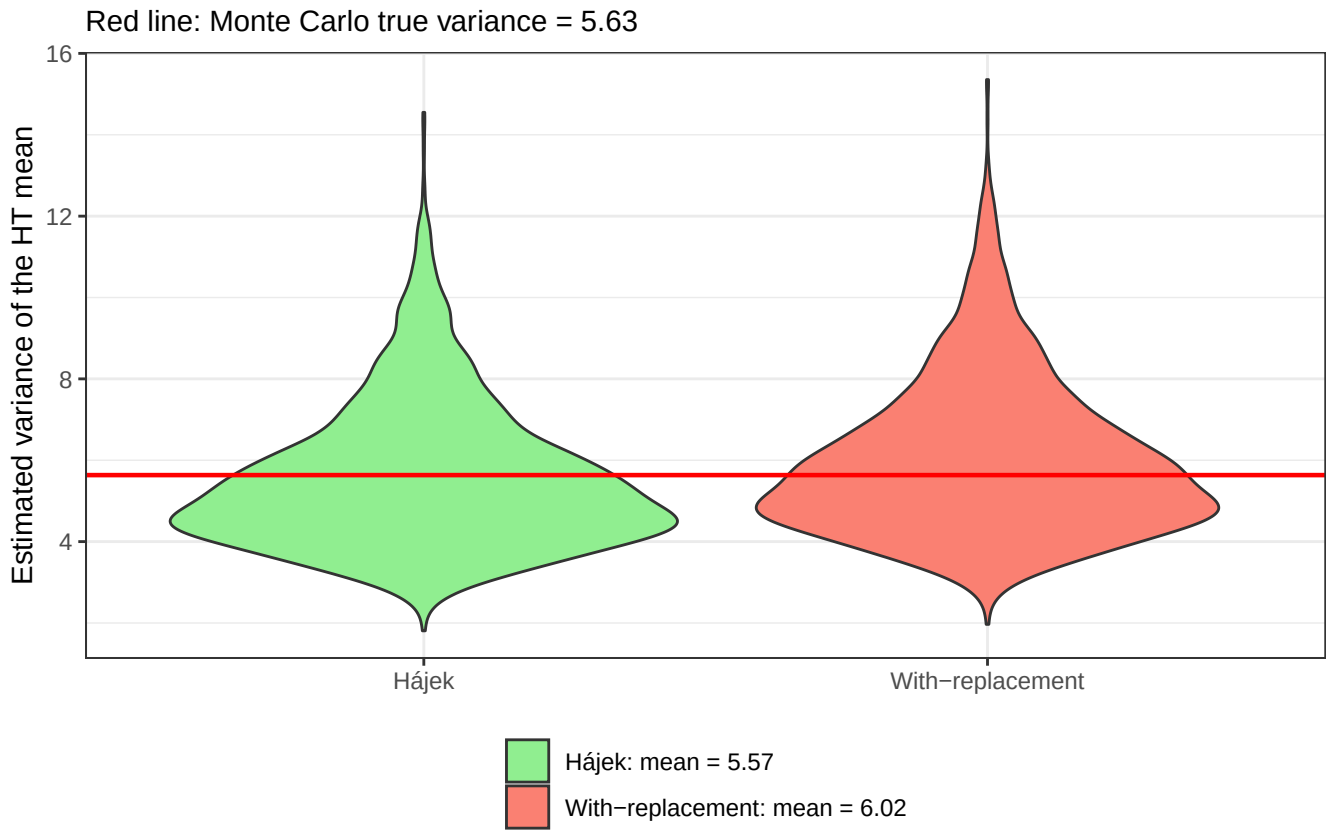

Figure 7.5: Estimated variances of the HT mean in 2000 repeated samples: Hájek and with-replacement approximations. The red line is the Monte Carlo estimate of the true variance, the variance of the 2000 HT estimates.

Table 7.8: Estimates of the mean acres of farmland in 1992 per county from one sample of 100 counties drawn without replacement

Method	Estimate	Estimated SE
Sample mean (ignores π_i)	834.3	100.53
HT, Hájek variance	315.8	1.98
HT, with-replacement variance	315.8	2.06

Output Note: The true mean is 310.1 thousand acres per county. The plain sample mean, 834.3, is far off because it gives every sampled county the same weight, although the large counties had a larger chance of being selected; the HT estimate 315.8 corrects for that. The Hájek and with-replacement SEs are 1.98 and 2.06. Over the 2000 repeated samples the true standard error of the HT mean is 2.37, and the average estimated SEs are 2.33 for the Hájek

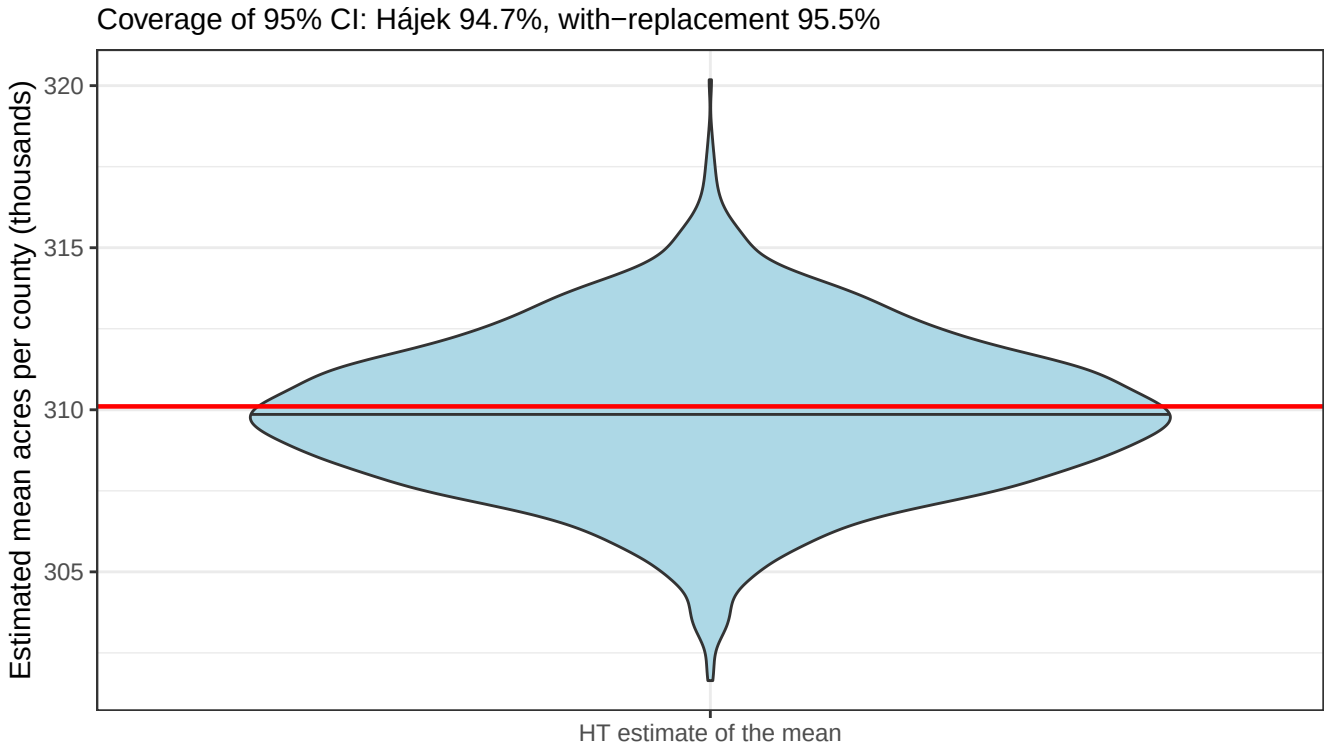

Figure 7.6: HT estimates of the mean acres per county in 2000 repeated samples. The red line is the true mean; the coverage rates of the 95% confidence intervals with the Hájek and the with-replacement variance estimates are given in the subtitle.

approximation and 2.43 for the with-replacement approximation, which is a little too large, as expected. The violin plots show the whole distribution of the two estimated variances, with the true variance (the variance of the 2000 HT estimates) as a red line: both are centred close to it, the with-replacement one slightly above. The HT estimates of the mean are centred at the true mean, as unbiasedness implies. The 95% confidence intervals, estimate $\pm\, t_{n-1}$ SE, cover the true mean in 94.7% of the samples with the Hájek variance and in 95.5% with the with-replacement variance; the latter is a little above the nominal 95% because its variance is too large. The ratio of the total acres in 1992 to that in 1987 is $\hat{B}_{\text{HT}} = \hat{t}_{\text{HT}}/t_x = 0.998$ (SE 0.006, Hájek).

7.7 Summary and Discussion on UPS

7.7.1 Summary of the Designs and Estimators

Design	Selection	Estimator of the total	Estimated SE
UPSWR (Section 7.2)	n independent draws, unit i with probability ψ_i	$\hat{t}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n} y_j^*/\psi_j^*$	$s_u/\sqrt{n}$, from the n values y_j^*/ψ_j^*
One- and two-stage cluster UPSWR, $\psi_i = M_i/M_0$ (Section 7.4, Section 7.5)	clusters drawn with replacement; in two stages, a fresh SRS of elements on every draw	$\hat{t}_{\text{HH}} = M_0\,\hat{\bar{y}}_{\text{HH}}$, $\hat{\bar{y}}_{\text{HH}} = \frac{1}{n}\sum_j \bar{y}_j^*$	$M_0\, s_{\bar{y}^*}/\sqrt{n}$; no second-stage term
UPSWO (Section 7.6)	n distinct units with inclusion probabilities π_i	$\hat{t}_{\text{HT}} = \sum_{i\in\mathcal{S}} y_i/\pi_i$	SYG (needs π_{ij}), or the Hájek or with-replacement approximation

In all three, a unit's weight is the inverse of its selection probability, and the estimator is precise when y_i is nearly proportional to the size x_i that defines the probabilities.

7.7.2 When to Use Unequal Probability Sampling

> ⚠ Remark: UPS is risky unless y is tied to the size variable
>
> The gain from unequal probability sampling depends entirely on how close the ratios $b_i = y_i/x_i$ are to a constant, that is, on y_i being roughly proportional to the size variable x_i used for ψ_i or π_i. Then the residuals $d_i = y_i - Bx_i$ in Table 7.3 are small, and so is the variance, as for the county acreages. Otherwise,
>
> - the estimate can be **much worse than a plain SRS average**, as in the second population of Section 7.3.2, where the sales are unrelated to the floor area;
> - a unit with a small x_i but a large y_i gets a very large weight $1/\psi_i$ (or $1/\pi_i$) when it is selected, so the estimates are unstable, and the sample variance of the b_j^* that estimates the SE tends to be too small, so that the intervals under-cover.
>
> UPS is therefore used mostly when the size is a good proxy of what is measured: selecting the clusters of a multi-stage survey with probabilities proportional to their number of households or students, auditing by the

monetary value of the items, or surveying establishments by their employment. When the size is only loosely related to y_i, use simple random or stratified sampling, with a ratio or regression estimator if x_i is known, which is much safer than using x_i at the design stage.

7.7.3 Practical Issues of Sampling Without Replacement

Sampling without replacement avoids drawing a large unit twice, but it is harder than UPSWR, in the selection and above all in the analysis.

Selecting the sample is no longer a real difficulty. Software draws a fixed-size sample with given π_i in one line, for example `UPbrewer(pik)` or `UPmaxentropy(pik)` in the R package `sampling`, and systematic πps sampling is easy to do by hand (Section 7.6.2). The two points that need care are to make the units with $n\,x_i/t_x \geq 1$ certainty units, and not to use the natural "draw with ψ_i and redraw on duplicates" procedure, which does not give the intended π_i.

Estimating the variance is the real cost. The unbiased SYG estimator needs the joint inclusion probabilities π_{ij}, which are hard to compute for most designs, and for systematic sampling some π_{ij} are 0, so the estimator cannot be computed at all. In practice the π_{ij} are avoided:

- **Small sampling fraction: analyse as with replacement.** With and without replacement differ only through factors of about $1-\pi_i$; when all π_i are small, the with-replacement variance $\hat{V}_{\text{WR}}$ is only slightly conservative. This is the standard practice of large surveys and survey software, which treat the first-stage units as drawn with replacement.
- **Some large π_i: a high-entropy design with Hájek's approximation.** Maximum entropy (conditional Poisson), Brewer, Sampford, Tillé and randomized systematic sampling have high entropy, and for them the Hájek approximation $\hat{V}_{\text{H}}$, which needs only the π_i, is accurate. This is the combination used in Section 7.6.3.
- **Avoid plain systematic sampling of a fixed list** when a design-based variance estimate is needed; randomize the order of the units first.

Simpler alternatives give most of the benefit of πps sampling with an easy analysis:

- **UPSWR** itself: when n/N is small, repeated draws are rare and the simple HH analysis of Section 7.2 loses almost nothing.
- **Stratification by size** with an SRS within each stratum (and a larger sampling fraction in the strata of large units), analysed with the usual stratified formulas.
- **Poisson sampling**: each unit is included independently with probability π_i, so $\pi_{ij} = \pi_i\pi_j$ and $\hat{V}(\hat{t}_{\text{HT}}) = \sum_{i\in\mathcal{S}}(1-\pi_i)\,y_i^2/\pi_i^2$ is simple, at the price of a random sample size.

In short, sampling without replacement pays off mainly when the sampling fraction is large, so that UPSWR would waste a noticeable part of the sample on repeated draws of the large units.

A List of Important Formulas

This appendix collects the boxed formulas of the chapters, together with the few companion formulas they rely on, sorted by topic. Each topic starts with its notation; the link after each formula leads to the section where it is derived. All designs draw units **without replacement** unless stated otherwise (UPSWR draws with replacement).

A.1 General Notation

Symbol	Meaning
N, n	number of units in the population $\mathcal{U}$ and in the sample $\mathcal{S}$
y_i	value of the study variable for unit i
$t = \sum_{i=1}^{N} y_i,\ \bar{y}_U = t/N$	population total and population mean
$S^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \bar{y}_U)^2$	population variance
$\bar{y} = \frac{1}{n}\sum_{i\in\mathcal{S}} y_i,\ s^2 = \frac{1}{n-1}\sum_{i\in\mathcal{S}}(y_i - \bar{y})^2$	sample mean and sample variance
$1 - n/N$	finite population correction (fpc)
$\hat{V}(\cdot),\ \mathrm{SE}(\cdot) = \sqrt{\hat{V}(\cdot)}$	estimated variance and standard error
$z_{\alpha/2},\ t_{\alpha/2,\,df}$	upper $\alpha/2$ quantiles of the standard normal and the t distribution

Confidence interval. For any estimator $\hat{\theta}$ in this list, an approximate $100(1-\alpha)\%$ confidence interval is

$$\boxed{\hat{\theta} \pm t_{\alpha/2,\,df}\,\mathrm{SE}(\hat{\theta})} \tag{A.1}$$

with $df = n - 1$ for an SRS ($n - H$ for H strata); for large samples $z_{\alpha/2}$ may replace $t_{\alpha/2,\,df}$ (Section 3.4).

A.2 Simple Random Sampling

Notation: $\hat{p}$ is the sample proportion of a 0/1 variable; e is the margin of error.

Mean (Section 3.2):

$$\boxed{\hat{V}(\bar{y}) = \left(1 - \frac{n}{N}\right)\frac{s^2}{n}, \qquad \mathrm{SE}(\bar{y}) = \sqrt{\left(1 - \frac{n}{N}\right)\frac{s^2}{n}}} \tag{A.2}$$

Total (Section 3.2):

$$\boxed{\hat{t} = N\bar{y}, \qquad \hat{V}(\hat{t}) = N^2\left(1 - \frac{n}{N}\right)\frac{s^2}{n}, \qquad \mathrm{SE}(\hat{t}) = N\,\mathrm{SE}(\bar{y})} \tag{A.3}$$

Proportion (Section 3.2); the number of units with the characteristic is estimated by $N\hat{p}$ with SE $N\,\mathrm{SE}(\hat{p})$:

$$\boxed{\hat{V}(\hat{p}) = \left(1 - \frac{n}{N}\right)\frac{\hat{p}(1-\hat{p})}{n-1}} \tag{A.4}$$

Sample size for margin of error e at level $1-\alpha$, with $n_0 = (z_{\alpha/2}S/e)^2$ (Section 3.5):

$$\boxed{n = \frac{n_0}{1 + n_0/N} = \frac{z_{\alpha/2}^2 S^2}{e^2 + z_{\alpha/2}^2 S^2/N}} \tag{A.5}$$

A.3 Stratified Random Sampling

Notation: stratum $h = 1, \ldots, H$ has N_h units, of which n_h are sampled by SRS; $\pi_h = N_h/N$ is the stratum weight; $\bar{y}_h$, s_h^2 and $\hat{p}_h$ are the sample mean, variance and proportion in stratum h; S_h^2 is the population variance in stratum h; c_h is the cost of sampling one unit in stratum h.

Estimators of the total and the mean (Section 4.2):

$$\boxed{\hat{t}_{\text{str}} = \sum_{h=1}^{H} N_h \bar{y}_h, \qquad \bar{y}_{\text{str}} = \frac{\hat{t}_{\text{str}}}{N} = \sum_{h=1}^{H} \pi_h \bar{y}_h} \tag{A.6}$$

Variance and SE (v_h is the contribution of stratum h; $\hat{V}(\hat{t}_{\text{str}}) = N^2\hat{V}(\bar{y}_{\text{str}})$) (Section 4.2):

$$\boxed{\hat{V}(\bar{y}_{\text{str}}) = \sum_{h=1}^{H} v_h, \qquad v_h = \left(1 - \frac{n_h}{N_h}\right)\pi_h^2\frac{s_h^2}{n_h}, \qquad \mathrm{SE}(\bar{y}_{\text{str}}) = \sqrt{\hat{V}(\bar{y}_{\text{str}})}} \tag{A.7}$$

Proportion (Section 4.4):

$$\boxed{\begin{aligned} \hat{p}_{\text{str}} &= \sum_{h=1}^{H} \frac{N_h}{N}\hat{p}_h, \\ \hat{V}(\hat{p}_{\text{str}}) &= \sum_{h=1}^{H} \left(1 - \frac{n_h}{N_h}\right)\left(\frac{N_h}{N}\right)^2 \frac{\hat{p}_h(1-\hat{p}_h)}{n_h - 1} \end{aligned}} \tag{A.8}$$

Proportional allocation $n_h = n\pi_h$ (Section 4.7):

$$\boxed{V_{\text{prop}}(\bar{y}_{\text{str}}) = \left(1 - \frac{n}{N}\right)\frac{1}{n}\sum_{h=1}^{H} \pi_h S_h^2} \tag{A.9}$$

Gain over SRS under proportional allocation; SSB and SSTO are the between-strata and total sums of squares of the population (Section 4.7):

$$\boxed{\text{percentage of variance reduction} \approx \frac{\text{SSB}}{\text{SSTO}} = R^2_{y|\text{strata}}} \tag{A.10}$$

Optimal allocation for a fixed cost (Section 4.8):

$$\boxed{n_h = \left(\frac{N_h S_h / \sqrt{c_h}}{\sum_{l=1}^{H} N_l S_l / \sqrt{c_l}} \right) n, \qquad n_h \propto \frac{N_h S_h}{\sqrt{c_h}}} \tag{A.11}$$

Neyman allocation (equal costs c_h) and its variance (Section 4.8):

$$\boxed{n_h = n \frac{N_h S_h}{\sum_{l=1}^{H} N_l S_l}}, \qquad \boxed{V_{\text{opt}}(\bar{y}_{\text{str}}) = \frac{1}{n} \left(\sum_{h=1}^{H} \pi_h S_h \right)^2 - \frac{1}{N} \sum_{h=1}^{H} \pi_h S_h^2} \tag{A.12}$$

A.4 Ratio and Regression Estimation

Notation: SRS of n pairs (x_i, y_i); the auxiliary variable x has known population total t_x and mean $\bar{x}_U = t_x/N$; $\bar{x}$, $\bar{y}$ are the sample means; $\hat{t}_y = N\bar{y}$, $\hat{t}_x = N\bar{x}$.

Ratio estimators (Section 5.3.1):

$$\boxed{\hat{B} = \frac{\bar{y}}{\bar{x}} = \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i} = \frac{\hat{t}_y}{\hat{t}_x}, \qquad \hat{t}_{yr} = \hat{B}\, t_x = N \hat{\bar{y}}_r, \qquad \hat{\bar{y}}_r = \hat{B}\, \bar{x}_U} \tag{A.13}$$

Ratio variances, with residuals $e_i = y_i - \hat{B}x_i$ and $s_e^2 = \frac{1}{n-1} \sum_{i=1}^{n} e_i^2$ (Section 5.3.1):

$$\boxed{\begin{aligned} \hat{V}(\hat{B}) &= \frac{1}{\bar{x}^2} \left(1 - \frac{n}{N}\right) \frac{s_e^2}{n}, \\ \hat{V}(\hat{\bar{y}}_r) &= \bar{x}_U^2\, \hat{V}(\hat{B}) = \left(\frac{\bar{x}_U}{\bar{x}}\right)^2 \left(1 - \frac{n}{N}\right) \frac{s_e^2}{n}, \\ \hat{V}(\hat{t}_{yr}) &= N^2\, \hat{V}(\hat{\bar{y}}_r) \end{aligned}} \tag{A.14}$$

Regression estimators, with the least-squares fit $\hat{B}_1 = \sum(x_i - \bar{x})(y_i - \bar{y}) / \sum(x_i - \bar{x})^2$ and $\hat{B}_0 = \bar{y} - \hat{B}_1 \bar{x}$ (Section 5.4.1):

$$\boxed{\hat{\bar{y}}_{reg} = \hat{B}_0 + \hat{B}_1 \bar{x}_U = \bar{y} + \hat{B}_1 \left(\bar{x}_U - \bar{x}\right), \qquad \hat{t}_{reg} = N \hat{\bar{y}}_{reg} = N\hat{B}_0 + \hat{B}_1 t_x} \tag{A.15}$$

Regression variance (Section 5.4.1):

$$\boxed{\hat{V}(\hat{\bar{y}}_{reg}) = \left(1 - \frac{n}{N}\right)\frac{s_e^2}{n}, \qquad s_e^2 = \frac{1}{n-2}\sum_{i=1}^{n}\left(y_i - \hat{B}_0 - \hat{B}_1 x_i\right)^2} \tag{A.16}$$

Post-stratification: an SRS sorted afterwards into H groups of known sizes N_h, $\pi_h = N_h/N$; $\bar{y}_h$ and s_h^2 are the mean and variance of the sampled units in group h, and $n_h^{post} = n\pi_h$ is its expected sample size (Section 5.5.1):

$$\boxed{\bar{y}_{post} = \sum_{h=1}^{H}\pi_h\,\bar{y}_h, \qquad \hat{t}_{post} = N\bar{y}_{post}} \tag{A.17}$$

$$\boxed{\hat{V}(\bar{y}_{post}) = \sum_{h=1}^{H}\pi_h^2\left(1 - \frac{n_h^{post}}{N_h}\right)\frac{s_h^2}{n_h^{post}} = \left(1 - \frac{n}{N}\right)\frac{1}{n}\sum_{h=1}^{H}\pi_h s_h^2} \tag{A.18}$$

Domain mean: the n_d sampled units $\mathcal{S}_d$ that fall in domain d have mean $\bar{y}_d$ (Section 5.6.1):

$$\boxed{\hat{V}(\bar{y}_d) = \left(1 - \frac{n}{N}\right)\frac{s_d^2}{n_d}, \qquad s_d^2 = \frac{1}{n_d - 1}\sum_{i\in\mathcal{S}_d}(y_i - \bar{y}_d)^2} \tag{A.19}$$

Prediction estimator with a fitted model $\hat{y}_i = \hat{m}(x_i)$ for every population unit (Section 5.7):

$$\boxed{\hat{\bar{y}}_{\text{pred}} = \frac{1}{N}\sum_{i=1}^{N}\hat{y}_i, \qquad \hat{t}_{y,\text{pred}} = \sum_{i=1}^{N}\hat{y}_i} \tag{A.20}$$

A.5 Cluster Sampling

Notation: an SRS $\mathcal{S}$ of n of the N clusters (psus); cluster i has M_i elements (ssus), total t_i and mean $\bar{y}_i$; $M_0 = \sum_{i=1}^{N} M_i$; $\bar{M} = \frac{1}{n}\sum_{i\in\mathcal{S}} M_i$ is the average size of the sampled clusters; s_t^2 is the sample variance of the t_i. In two-stage sampling m_i of the M_i elements are subsampled, with mean $\bar{y}_i$ and variance s_i^2, and $\hat{t}_i = M_i\bar{y}_i$.

One stage, equal sizes $M_i = M$: an SRS of cluster totals (Section 6.3):

$$\boxed{\hat{t} = \frac{N}{n}\sum_{i\in\mathcal{S}} t_i, \qquad \text{SE}(\hat{t}) = N\sqrt{\left(1 - \frac{n}{N}\right)\frac{s_t^2}{n}}, \qquad \hat{\bar{y}} = \frac{\hat{t}}{NM}} \tag{A.21}$$

One stage, unequal sizes: ratio estimator of the mean per element (Section 6.5):

$$\boxed{\hat{\bar{y}}_r = \frac{\sum_{i\in\mathcal{S}} t_i}{\sum_{i\in\mathcal{S}} M_i} = \frac{\sum_{i\in\mathcal{S}} M_i\bar{y}_i}{\sum_{i\in\mathcal{S}} M_i}} \tag{A.22}$$

$$\boxed{\mathrm{SE}(\hat{\bar{y}}_r) = \sqrt{\left(1-\frac{n}{N}\right)\frac{s_e^2}{n\bar{M}^2}}, \qquad s_e^2 = \frac{1}{n-1}\sum_{i\in\mathcal{S}} e_i^2, \qquad e_i = t_i - \hat{\bar{y}}_r M_i} \tag{A.23}$$

Two stages: ratio estimator (Section 6.6):

$$\boxed{\hat{\bar{y}}_r = \frac{\sum_{i\in\mathcal{S}} \hat{t}_i}{\sum_{i\in\mathcal{S}} M_i} = \frac{\sum_{i\in\mathcal{S}} M_i \bar{y}_i}{\sum_{i\in\mathcal{S}} M_i}} \tag{A.24}$$

Two stages: variance, a ratio piece plus n/N times a stratified piece, with $e_i = \hat{t}_i - \hat{\bar{y}}_r M_i$ and $M = n\bar{M}$ (Section 6.6):

$$\boxed{\begin{aligned} &\hat{V}(\hat{\bar{y}}_r) = \hat{V}_{\text{ratio}} + \frac{n}{N}\hat{V}_{\text{str}}, \\ &\hat{V}_{\text{ratio}} = \left(1-\frac{n}{N}\right)\frac{s_e^2}{n\bar{M}^2}, \qquad \hat{V}_{\text{str}} = \sum_{i\in\mathcal{S}}\left(\frac{M_i}{M}\right)^2\left(1-\frac{m_i}{M_i}\right)\frac{s_i^2}{m_i} \end{aligned}} \tag{A.25}$$

A.6 Unequal Probability Sampling

Notation: n independent draws **with replacement** (UPSWR); on each draw, unit i is selected with probability $\psi_i = x_i/t_x$ for a size variable x_i; the unit selected on draw j has ratio $b_j^* = y_j^*/x_j^*$; $B = t/t_x$.

Hansen–Hurwitz (HH) estimator of B (Section 7.2.1):

$$\boxed{\hat{B}_{\text{HH}} = \bar{b}^* = \frac{1}{n}\sum_{j=1}^{n} b_j^*}, \qquad \boxed{\widehat{\mathrm{SE}}(\hat{B}_{\text{HH}}) = \frac{s_{b^*}}{\sqrt{n}}}, \qquad s_{b^*}^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(b_j^* - \bar{b}^*\right)^2 \tag{A.26}$$

HH mean and total: multiply the estimate and its SE by $\bar{x}_U$ or t_x (Section 7.2.1):

$$\boxed{\hat{\bar{y}}_{\text{HH}} = \bar{x}_U\,\bar{b}^*, \quad \widehat{\mathrm{SE}}(\hat{\bar{y}}_{\text{HH}}) = \bar{x}_U\frac{s_{b^*}}{\sqrt{n}}, \qquad \hat{t}_{\text{HH}} = t_x\,\bar{b}^*, \quad \widehat{\mathrm{SE}}(\hat{t}_{\text{HH}}) = t_x\frac{s_{b^*}}{\sqrt{n}}} \tag{A.27}$$

HH total for any selection probabilities ψ_i; y_j^* and ψ_j^* belong to the unit selected on draw j (Section 7.2.4):

$$\boxed{\hat{t}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n}\frac{y_j^*}{\psi_j^*}, \qquad \widehat{\mathrm{SE}}(\hat{t}_{\text{HH}}) = \frac{s_u}{\sqrt{n}}, \qquad s_u^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(\frac{y_j^*}{\psi_j^*} - \hat{t}_{\text{HH}}\right)^2} \tag{A.28}$$

UPS versus the ratio estimator: with residuals $d_i = y_i - Bx_i$, $B = t/t_x$, HH (with $\psi_i = x_i/t_x$) has a smaller variance than the SRS ratio estimator, ignoring the fpc, exactly when (Section 7.3.1)

$$\boxed{H_d = \frac{\sum_{i=1}^{N} d_i^2}{\sum_{i=1}^{N} d_i^2/x_i} > \bar{x}_U} \tag{A.29}$$

Cluster sampling with PPS, one or two stages (Section 7.4.1, Section 7.5.1): n clusters are drawn with replacement with $\psi_i = M_i/M_0$; $\bar{M}_U = M_0/N$. Here $\bar{y}_j^*$ is the mean per element of the cluster on draw j: its true mean t_j^*/M_j^* (one stage), or the mean of an SRS of m_j^* of its elements, $\hat{t}_j^*/M_j^*$, taken afresh on every draw (two stages). The formulas are the same in both designs:

$$\boxed{\hat{\bar{y}}_{\text{HH}} = \frac{1}{n}\sum_{j=1}^{n} \bar{y}_j^*, \qquad \widehat{\text{SE}}(\hat{\bar{y}}_{\text{HH}}) = \frac{s_{\bar{y}^*}}{\sqrt{n}}, \qquad s_{\bar{y}^*}^2 = \frac{1}{n-1}\sum_{j=1}^{n}\left(\bar{y}_j^* - \hat{\bar{y}}_{\text{HH}}\right)^2} \tag{A.30}$$

Quantity	Estimate	Estimated SE
Mean per element (ssu-level mean), $\bar{y}_U = t/M_0$	$\hat{\bar{y}}_{\text{HH}}$	$s_{\bar{y}^*}/\sqrt{n}$
Mean per cluster (psu-level mean of the totals), $\bar{t}_U = t/N$	$\hat{\bar{t}}_{\text{HH}} = \bar{M}_U\,\hat{\bar{y}}_{\text{HH}}$	$\bar{M}_U\, s_{\bar{y}^*}/\sqrt{n}$
Total, t	$\hat{t}_{\text{HH}} = M_0\,\hat{\bar{y}}_{\text{HH}}$	$M_0\, s_{\bar{y}^*}/\sqrt{n}$

No second-stage variance term is needed: with $\bar{y}_{iU} = t_i/M_i$ and $V_{2i} = \left(1 - \frac{m_i}{M_i}\right)\frac{S_i^2}{m_i}$ ($V_{2i} = 0$ in one stage),

$$V(\bar{y}_j^*) = \underbrace{\sum_{i=1}^{N} \psi_i\left(\bar{y}_{iU} - \bar{y}_U\right)^2}_{\text{between clusters}} + \underbrace{\sum_{i=1}^{N} \psi_i V_{2i}}_{\text{within clusters}}, \qquad E\left(s_{\bar{y}^*}^2\right) = V(\bar{y}_j^*), \tag{A.31}$$

since the draws $\bar{y}_1^*, \ldots, \bar{y}_n^*$ are iid.

Horvitz–Thompson (HT) estimator: without replacement, with inclusion probabilities $\pi_i = P(i \in \mathcal{S})$, for example $\pi_i = nx_i/t_x$ (Section 7.6.1):

$$\boxed{\hat{t}_{\text{HT}} = \sum_{i \in \mathcal{S}} \frac{y_i}{\pi_i}} \tag{A.32}$$

B List of Figures, Tables, and Apps

A consolidated index of every numbered figure and table in the book. Click any entry to jump to it in context.

B.1 Shinylive Apps

The simulation apps have their own archive: Shinylive Apps for Sampling Survey.

B.2 List of Figures

B.2.1 Introduction to Sampling Survey

- Figure 1.1 — Target population, sampling frame population and sampled population
- Figure 1.2 — A simple random sample
- Figure 1.3 — A stratified random sample
- Figure 1.4 — A cluster sample
- Figure 1.5 — Ratio estimation: an SRS, the fitted line and the predicted values
- Figure 1.6 — Unequal probability sampling with probabilities roughly proportional to y
- Figure 1.7 — Sampling distribution of the sample mean under an unequal-probability design
- Figure 1.8 — Unbiased, precise and accurate estimators as archers

B.2.2 Data Analysis with R

- Figure 2.1 — Histogram of 1992 farm acreage
- Figure 2.2 — 1992 vs. 1987 farm acreage, with the 1:1 line
- Figure 2.3 — The saved high-resolution PNG, re-inserted
- Figure 2.4 — Distribution of log(acres92): histogram with empirical density and normal fit

B.2.3 Simple Random Sampling

- Figure 3.1 — A simple random sample from a synthetic population
- Figure 3.2 — Farm acreage in 1992 for the SRS of 300 counties, with mean and median
- Figure 3.3 — Population distribution and sampling distribution of the SRS sample mean over 5000 repeated samples (n = 300)
- Figure 3.4 — First 100 simulated 95% confidence intervals for the SRS sample mean, with coverage failures highlighted

B.2.4 strata

- Figure 4.1 — A stratified random sample from a synthetic population
- Figure 4.2 — The six caribou strata and their sampled units
- Figure 4.3 — Prediction by the overall mean (SSTO) vs. by the stratum means (SSW)
- Figure 4.4 — Strata with large vs. small R^2
- Figure 4.5 — The variance $p(1-p)$ of a 0/1 variable
- Figure 4.6 — Strata formed by cutting an auxiliary variable
- Figure 4.7 — 1992 farm acreage by region stratum
- Figure 4.8 — Sampling distribution of the mean under SRS, proportional, and Neyman allocation (region strata, 2000 simulations)
- Figure 4.9 — 1992 farm acreage by acres82 quantile stratum
- Figure 4.10 — Sampling distribution of the mean under SRS, proportional, and Neyman allocation (acres82 strata, 2000 simulations)

B.2.5 ratio

- Figure 5.1 — Illustration of the benefits of ratio estimation
- Figure 5.2 — The toy France example with the ratio line and residuals
- Figure 5.3 — 1992 vs. 1987 farm acreage for the SRS of 300 counties, with the ratio line
- Figure 5.6 — Data suited to ratio estimation versus regression estimation
- Figure 5.7 — Illustration of the benefits of regression estimation
- Figure 5.10 — Illustration of the benefits of post-stratification
- Figure 5.11 — A prediction estimator with a flexible (loess) model
- Figure 5.4 — Cherry tree volume vs. diameter
- Figure 5.5 — Cherry tree volume vs. diameter with the fitted ratio line
- Figure 5.12 — 1992 vs. 1987 county acreage with the true ratio line
- Figure 5.13 — Estimator variance under repeated sampling: SRS vs. ratio estimator (2000 simulations)
- Figure 5.8 — Cherry tree volume vs. diameter with the fitted regression line
- Figure 5.9 — Field counts vs. photo counts of dead trees with the fitted regression line

B.2.6 cluster

- Figure 6.1 — A two-stage cluster sample from a synthetic population
- Figure 6.2 — Stratified sampling versus one-stage cluster sampling on the same population
- Figure 6.3 — Notation for cluster sampling: cells y_ij, psu totals t_i and sizes M_i
- Figure 6.4 — A two-stage cluster sample in the cell notation
- Figure 6.5 — Large versus small adjusted R² between clusters
- Figure 6.6 — Cluster totals against cluster sizes, with the ratio line
- Figure 6.7 — Coot egg volumes by clutch

B.2.7 ups

- Figure 7.1 — Physicians vs. population (log-log scale), sample of U.S. counties

- Figure 7.2 — Spread of y about the line y = Bx: when UPS beats the ratio estimator
- Figure 7.3 — Estimator distribution: naive SRS vs. SRS ratio vs. SRS regression vs. UPSWR (2000 simulations)
- 1 — Systematic πps sampling: units on a line of length n, selected by the points u, u+1, …, u+n−1

B.3 List of Tables

B.3.1 Data Analysis with R

- Table 2.1 — Regional summary of five variables, grouped by statistic
- Table 2.2 — Regional summary of five variables, grouped by variable
- Table 2.3 — Automatic descriptive summary of agpop variables by region
- Table 2.4 — Comparing column means: custom function vs. base R colMeans()

B.3.2 strata

- Table 4.1 — Sample survey data for agstrat.csv, including stratum and weight
- Table 4.2 — Sample summary statistics for acres92 by region stratum
- Table 4.3 — Neyman-optimal allocation proportions for the seals survey zones
- Table 4.4 — Population sizes and Neyman-optimal sample sizes by region stratum
- Table 4.5 — Simulated mean, variance, and variance reduction by allocation method (region strata)
- Table 4.6 — Simulated mean, variance, and variance reduction by allocation method (acres82 strata)

B.3.3 ratio

- Table 5.1 — Sample data: cherry tree volume and diameter
- Table 5.7 — Simulated variance and variance reduction: ratio estimator vs.
- Table 5.2 — Observed vs. population stratum proportions and post-stratified sample sizes

B.3.4 cluster

- Table 6.1 — Raw data: algebra classes (sample)
- Table 6.2 — Raw data: coot egg clutches (sample)
- Table 6.3 — Cluster-level summaries for the 10 sampled states

B.3.5 ups

- Table 7.1 — Sample of U.S. county population and physician counts

C Estimating Functions Source Code

The R functions used throughout this book to compute point estimates, standard errors, and confidence intervals — along with their gt-table formatting helpers — are defined in a single shared file, `samplingestimate.r`, which every chapter sources. Its full source is listed below for reference.

```
# ============================================================
# Shared estimation functions for the Elements of Sampling Survey
# book. Source this file from any chapter that needs point/SE/CI
# estimators or their gt-table formatting helpers, e.g.:
#
#   source("samplingestimate.r")
#
# ============================================================

suppressPackageStartupMessages({
    library(gt)
})

# ------------------------------------------------------------
# Shared gt-table helpers
# ------------------------------------------------------------

## Formats a gt table column as ordinary fixed-point, UNLESS its values are
## too small (< `small` in absolute value) or too large (>= `large`), in
## which case it switches that column to scientific notation (rendered as
## "m x 10^n" via gt's exp_style = "x10n") instead.
fmt_auto <- function (gt_tbl, data, columns, decimals = 2, small = 1e-3, large = 1e5)
{
    for (col in columns)
    {
        x <- data[[col]]
        x <- x[is.finite (x) & x != 0]
        if (length (x) == 0) next
        ## judge by the column's TYPICAL magnitude, not by its extremes: the
        ## Sum row is n times the size of a data row, and one residual that
        ## lands near zero should not push a whole column into scientific
        typical <- stats::median (abs (x))
        if (typical < small || typical >= large)
            gt_tbl <- fmt_scientific (gt_tbl, columns = tidyselect::all_of (col), decimals =
↪  decimals, exp_style = "x10n")
        else
            gt_tbl <- fmt_number (gt_tbl, columns = tidyselect::all_of (col), decimals =
↪  decimals)
    }
```

```
    gt_tbl
}

## Formats a single scalar for inline formula display (fixed-point with
## thousands separators, switching to "m \times 10^{n}" LaTeX scientific
## notation for very small or very large magnitudes, mirroring fmt_auto()'s
## use of exp_style = "x10n") --- used to show worked-example steps such as
## \bar y = sum/n with the actual numbers plugged in.
fmt_num <- function (x, decimals = 2, small = 1e-3, large = 1e5)
{
    use_sci <- is.finite (x) & x != 0 & (abs (x) < small | abs (x) >= large)
    exps    <- ifelse (x == 0, 0, floor (log10 (abs (x))))
    mant    <- ifelse (x == 0, 0, x / 10^exps)
    ifelse (use_sci,
            sprintf ("%s \\times 10^{%d}", formatC (mant, format = "f", digits = decimals),
             ↪  exps),
            formatC (x, format = "f", digits = decimals, big.mark = ","))
}

## The agriculture files (agpop.csv and the samples drawn from it: agsrs.csv,
## agstrat.csv, ...) record county acreages in acres, which run into the
## millions and make every working table below hard to read. This converts the
## acreage columns to THOUSANDS of acres, so a mean reads as 297.9 rather than
## 297,897, and a variance as 2.0e5 rather than 2.0e11. The file's missing-value
## code, -99, becomes NA first, so it cannot be silently rescaled to -0.099.
## d --- data frame just read from one of those files
## cols --- acreage columns to convert (those absent from d are ignored)
acres_in_thousands <- function (d, cols = c ("acres92", "acres87", "acres82"))
{
    cols <- intersect (cols, names (d))
    d[cols] <- lapply (d[cols], function (x) { x[x == -99] <- NA; x / 1000 })
    d
}

## Truncates a row-level working data.frame whose LAST row is a summary row
## (e.g. "Sum"/"Total"): if there are more than head_n data rows (excluding
## that summary row), keeps only the first head_n data rows, inserts one
## "..." placeholder row, then the summary row --- so long working tables
## stay readable. id_col --- name of the row-label column.
truncate_working <- function (working, id_col, head_n = 6)
{
    n <- nrow (working) - 1
    if (n <= head_n + 1) return (working)

    dots <- working[1, ]
    dots[1, ] <- NA
    dots[1, id_col] <- "..."

    rbind (working[seq_len (head_n), ], dots, working[nrow (working), ])
}
```

```r
# Helper function to format estimation outputs with appropriate mathematical headers
format_est_gt <- function(est_data, est_type = c("mean", "total", "ratio", "reg_mean",
 ↪  "reg_total", "ratio_mean", "domain_mean",
                                                  "hh_total", "hh_B", "hh_ybar", "hh_tbar")) {
  est_type <- match.arg(est_type)

  sym_map <- list(
    "mean"        = c(est = "$\\bar{y}$",          se = "$\\mathrm{SE}(\\bar{y})$"),
    "total"       = c(est = "$\\hat{t}$",          se = "$\\mathrm{SE}(\\hat{t})$"),
    "ratio"       = c(est = "$\\hat{B}$",          se = "$\\mathrm{SE}(\\hat{B})$"),
    "reg_mean"    = c(est = "$\\bar{y}_{reg}$",    se = "$\\mathrm{SE}(\\bar{y}_{reg})$"),
    "reg_total"   = c(est = "$\\hat{t}_{reg}$",    se = "$\\mathrm{SE}(\\hat{t}_{reg})$"),
    "ratio_mean"  = c(est = "$\\bar{y}_{r}$",      se = "$\\mathrm{SE}(\\bar{y}_{r})$"),
    "domain_mean" = c(est = "$\\bar{y}_{d}$",      se = "$\\mathrm{SE}(\\bar{y}_{d})$"),
    ## UPSWR (Hansen-Hurwitz) total
    "hh_total"      = c(est = "$\\hat{t}_{\\mathrm{HH}}$",
                        se  = "$\\mathrm{SE}(\\hat{t}_{\\mathrm{HH}})$"),
    ## UPSWR with psi_i = x_i / t_x (upswr_est): ratio B, mean of y per unit
    "hh_B"          = c(est = "$\\hat{B}_{\\mathrm{HH}}$",
                        se  = "$\\mathrm{SE}(\\hat{B}_{\\mathrm{HH}})$"),
    "hh_ybar"       = c(est = "$\\hat{\\bar{y}}_{\\mathrm{HH}}$",
                        se  = "$\\mathrm{SE}(\\hat{\\bar{y}}_{\\mathrm{HH}})$"),
    ## UPSWR of clusters (cluster_upswr): mean of the cluster totals
    "hh_tbar"       = c(est = "$\\hat{\\bar{t}}_{\\mathrm{HH}}$",
                        se  = "$\\mathrm{SE}(\\hat{\\bar{t}}_{\\mathrm{HH}})$")
  )

  est_sym <- sym_map[[est_type]]["est"]
  se_sym  <- sym_map[[est_type]]["se"]

  # Convert to data frame. Use unname() to strip hidden lm() coefficient names
  if (is.vector(est_data) && !is.list(est_data)) {
    df <- as.data.frame(t(unname(est_data)))
  } else {
    df <- as.data.frame(est_data)
  }

  # Force standard column names so gt() never gets confused
  colnames(df) <- c("Est.", "S.E.", "ci.low", "ci.upp")

  # Check if we should use rownames as a stub
  use_stub <- !is.null(rownames(df)) && !all(rownames(df) == as.character(1:nrow(df)))

  df |>
    gt(rownames_to_stub = use_stub) |>
    cols_label(
      Est.   = md(est_sym),
      S.E.   = md(se_sym),
      ci.low = md("95% CI Lower"),
      ci.upp = md("95% CI Upper")
    ) |>
```

```
    fmt_number(
      columns = everything(),
      decimals = 2
    ) |>
    tab_options(table.width = pct(70))
}

## Formats an lm object as two gt tables, replacing the plain-text
## summary(lmfit) output:
##   $coef --- coefficient table (Estimate, S.E., t value, p value)
##   $fit  --- one-row table of model fit statistics, including R^2
## returns list($coef, $fit)
format_lm_gt <- function (lmfit)
{
    s <- summary (lmfit)

    coef_df <- as.data.frame (s$coefficients)
    colnames (coef_df) <- c ("Estimate", "S.E.", "t value", "p value")

    coef_gt <- coef_df |>
        gt (rownames_to_stub = TRUE) |>
        fmt_number (columns = c ("Estimate", "S.E.", "t value"), decimals = 4) |>
        fmt_number (columns = "p value", decimals = 4) |>
        tab_header (title = "Coefficient Estimates") |>
        tab_options (table.width = pct(70))

    fstat <- s$fstatistic
    fit_df <- data.frame (
        R.squared     = s$r.squared,
        Adj.R.squared = s$adj.r.squared,
        Sigma         = s$sigma,
        F.statistic   = unname (fstat["value"]),
        p.value       = unname (pf (fstat["value"], fstat["numdf"], fstat["dendf"], lower.tail
        ↪  = FALSE))
    )

    fit_gt <- fit_df |>
        gt () |>
        cols_label (
            R.squared     = md ("$R^2$"),
            Adj.R.squared = md ("Adj. $R^2$"),
            Sigma         = md ("$\\hat\\sigma$"),
            F.statistic   = "F-statistic",
            p.value       = "p-value"
        ) |>
        fmt_number (columns = everything (), decimals = 4) |>
        tab_header (title = "Model Fit Summary") |>
        tab_options (table.width = pct(70))

    list (coef = coef_gt, fit = fit_gt)
}
```

```r
# ------------------------------------------------------------
# Base Estimators
# ------------------------------------------------------------

## sdata --- a vector of original survey data
## N --- population size
## estimate --- "mean" (default) returns the population MEAN estimate;
##              "total" returns the population TOTAL estimate (N * mean,
##              with SE/CI scaled accordingly); requires finite N
## show.details --- if TRUE (default), also return a gt table of the
##                  row-level working values y_i, the fitted value \hat y_i
##                  (constant, and equal to \bar y under the SRS model),
##                  e_i = y_i-\hat y_i, and e_i^2, with a Sum row and a
##                  footnote showing s_y^2; NULL otherwise
## col_labels --- named list (y, yhat, dev, ybar, s) of RAW latex: the
##                data, fitted and deviation columns, the symbol for the
##                sample mean, and for the sample SD; a partial list fills in
##                the remaining defaults, so callers can override just one label
## title --- the working table's title
## returns list ($estimate = c(Est., S.E., ci.low, ci.upp), $table)
srs_est <- function (sdata, N = Inf, estimate = c ("mean", "total"),
                     show.details = TRUE, col_labels = list (),
                     title = "SRS Mean Estimation: Working Table", decimals = 2)
{
    estimate <- match.arg (estimate)
    ## show `decimals` decimals in the working table and its notes
    fmt_num_default <- fmt_num
    fmt_num <- function (x, d = decimals) fmt_num_default (x, decimals = if (d == 2) decimals
↪   else d)
    col_labels <- modifyList (list (y = "y_i", yhat = "\\hat y_i", dev = "e_i",
                                    ybar = "\\bar y", s = "s_y"), col_labels)
    ybar_lab <- col_labels$ybar
    s_lab    <- col_labels$s

    n <- length (sdata)
    ybar <- mean (sdata)
    dev <- sdata - ybar
    s2y <- sum (dev^2) / (n - 1)
    se.ybar <- sqrt((1 - n / N)) * sd (sdata) / sqrt(n)

    if (estimate == "total")
    {
        if (!is.finite (N))
            stop ("N must be finite to compute estimate = \"total\"")
        est <- N * ybar
        se.est <- N * se.ybar
    }
    else
    {
        est <- ybar
```

```
    se.est <- se.ybar
}
mem <- qt (0.975, df = n - 1) * se.est
estimate_vec <- c (Est. = est, S.E. = se.est, ci.low = est - mem, ci.upp = est + mem)

if (!show.details)
    return (list (estimate = estimate_vec, table = NULL))

working <- data.frame (
    i    = c (seq_len (n), "Sum"),
    y    = c (sdata, sum (sdata)),
    yhat = rep (ybar, n + 1),
    dev  = c (dev, sum (dev)),
    dev2 = c (dev^2, sum (dev^2))
)
working <- truncate_working (working, id_col = "i")

table <- gt (working) |>
    tab_header (title = title, subtitle = md (sprintf ("$n = %d$", n))) |>
    cols_label (
        i    = md ("$i$"),
        y    = md (sprintf ("$%s$", col_labels$y)),
        yhat = md (sprintf ("$%s$", col_labels$yhat)),
        dev  = md (sprintf ("$%s$", col_labels$dev)),
        dev2 = md (sprintf ("$(%s-%s)^2$", col_labels$y, col_labels$yhat))
    ) |>
    cols_width (i ~ px (40), everything () ~ px (110)) |>
    fmt_auto (data = working, columns = c ("y", "yhat", "dev", "dev2"), decimals =
    ↪  decimals) |>
    sub_missing (missing_text = "...") |>
    tab_style (
        style     = cell_text (weight = "bold"),
        locations = cells_body (rows = i == "Sum")
    ) |>
    tab_footnote (
        footnote  = md (paste0 (
            if (col_labels$yhat != ybar_lab)
                sprintf ("$%s = %s$ for every unit under the SRS model; ", col_labels$yhat,
                ↪  ybar_lab),
            sprintf ("$%s^2 = \\sum_i (%s-%s)^2/(n-1) = %s$.",
                     s_lab, col_labels$y, col_labels$yhat, fmt_num (s2y)))),
        locations = cells_column_labels (columns = dev2)
    ) |>
    tab_source_note (
        source_note = md (if (estimate == "total")
            sprintf ("$\\hat t = N%s = %s \\times %s = %s$",
                     ybar_lab, fmt_num (N, 0), fmt_num (ybar), fmt_num (est))
        else
            sprintf ("$%s = \\dfrac{\\sum_i %s}{n} = \\dfrac{%s}{%d} = %s$",
                     ybar_lab, col_labels$y, fmt_num (sum (sdata)), n, fmt_num (ybar)))
    ) |>
```

```
        tab_source_note (
            source_note = md (if (estimate == "total")
                sprintf ("$\\mathrm{SE}(\\hat t) = N \\times \\mathrm{SE}(%s) = %s \\times %s =
                 ↪  %s$",
                        ybar_lab, fmt_num (N, 0), fmt_num (se.ybar), fmt_num (se.est))
            else if (is.finite (N))
                sprintf ("$\\mathrm{SE}(%s) = \\sqrt{1-\\dfrac{n}{N}}\\,\\dfrac{%s}{\\sqrt n} =
                 ↪  \\sqrt{1-\\dfrac{%d}{%d}}\\,\\dfrac{%s}{\\sqrt{%d}} = %s$",
                        ybar_lab, s_lab, n, N, fmt_num (sqrt (s2y)), n, fmt_num (se.ybar))
            else
                sprintf ("$\\mathrm{SE}(%s) = \\dfrac{%s}{\\sqrt n} = \\dfrac{%s}{\\sqrt{%d}} =
                 ↪  %s$",
                        ybar_lab, s_lab, fmt_num (sqrt (s2y)), n, fmt_num (se.ybar)))
        )

    list (estimate = estimate_vec, table = table)
}

## ydata --- observations of the variable of interest
## xdata --- observations of the auxiliary variable
## xbarU --- population mean of the auxiliary variable; required when
##           estimate is "mean" or "total"; ignored when estimate = "model"
## N --- population size
## estimate --- "mean" (default) returns the ratio estimate of the
##              population MEAN of y, Bhat*xbarU; "total" returns the ratio
##              estimate of the population TOTAL of y (requires finite N);
##              "model" returns the raw ratio coefficient Bhat = ybar/xbar
##              itself (with its own SE)
## show.details --- if TRUE (default), also return a gt table of the
##                  row-level working values y_i, x_i, fitted yhat_i =
##                  Bhat*x_i, residual e_i, and e_i^2, with a Sum row and a
##                  footnote showing Bhat and s_e^2; NULL otherwise
## col_labels --- named list (y, x, yhat) of RAW latex for those
##                working-table columns, so this function can be reused
##                (e.g. by cluster_ratio()) with
##                context-appropriate labels
## extra_col --- optional column spec, or list of column specs, adding
##               RAW-latex-labelled columns of per-unit `values` right before
##               the y column of the working table (e.g. the within-cluster
##               means $\bar y_i$ used by cluster_ratio() to build $\hat
##               t_i = \bar y_i M_i$); a spec is a list (label, values,
##               formula, total): `formula` (optional) is shown as a footnote
##               on the y column documenting that relationship, and
##               `total = TRUE` puts the column's sum in the Sum row
## tail_col --- optional column spec, or list of specs, in the same format,
##              appended AFTER the $e_i^2$ column (e.g. the per-cluster
##              second-stage variance contributions $\hat v_i$); each spec's
##              `formula` is footnoted on its own column
## var2 --- extra variance component to add to $\hat V(\hat B)\,\bar x^2$,
##          i.e. $\hat V(\hat B) = [(1-n/N) s_e^2/n + \mathrm{var2}]/\bar x^2$;
##          0 (default) gives the usual one-stage ratio variance. Used by
```

```
##          cluster_ratio() to add the within-cluster (second-stage) term
## var2_label --- RAW latex for that extra term, shown symbolically in the
##                standard-error source note when var2 > 0
## xbar_label --- RAW latex naming $\bar x$ in the standard-error note
##                (cluster_ratio() passes "\\bar M", the mean cluster size)
## returns list ($estimate = c(Est., S.E., ci.low, ci.upp), $table)
ratio_est <- function (ydata, xdata, xbarU = NULL, N = Inf,
                       estimate = c ("mean", "total", "model"), show.details = TRUE,
                       col_labels = list (y = "y_i", x = "x_i", yhat = "\\hat y_i"),
                       extra_col = NULL, B_label = "\\hat B",
                       tail_col = NULL, var2 = 0, var2_label = NULL,
                       xbar_label = "\\bar x")
{
  estimate <- match.arg (estimate)

  n <- length (xdata)
  xbar <- mean (xdata)
  ybar <- mean (ydata)
  B_hat <- ybar / xbar
  yhat <- B_hat * xdata
  e <- ydata - yhat
  var_e <- sum (e^2) / (n - 1)
  sd_B_hat <- sqrt ((1 - n/N) * var_e / n + var2) / xbar

  if (estimate == "model")
  {
      est <- B_hat
      sd_est <- sd_B_hat
  }
  else
  {
      if (is.null (xbarU))
          stop ("xbarU (population mean of x) is required when estimate is \"mean\" or
           ↪  \"total\"")
      if (estimate == "total")
      {
          if (!is.finite (N))
              stop ("N must be finite to compute estimate = \"total\"")
          est <- B_hat * xbarU * N
          sd_est <- sd_B_hat * xbarU * N
      }
      else
      {
          est <- B_hat * xbarU
          sd_est <- sd_B_hat * xbarU
      }
  }

  mem <- qt (0.975, df = n - 1) * sd_est
  estimate_vec <- c (Est. = est, S.E. = sd_est, ci.low = est - mem, ci.upp = est + mem)
```

```
if (!show.details)
    return (list (estimate = estimate_vec, table = NULL))

## a column spec is list (label, values, formula, total); accept either one
## spec or a list of them, so callers can add several columns at once
norm_specs <- function (x)
    if (is.null (x)) list () else if (!is.null (x$label)) list (x) else x
extras <- norm_specs (extra_col)
tails  <- norm_specs (tail_col)
nm     <- function (specs, prefix) if (length (specs)) paste0 (prefix, seq_along (specs))
↪  else character (0)
e_nms  <- nm (extras, "extra")
t_nms  <- nm (tails,  "tail")

add_specs <- function (cols, specs, nms)
{
    for (k in seq_along (specs))
        cols[[nms[k]]] <- c (specs[[k]]$values,
                             if (isTRUE (specs[[k]]$total)) sum (specs[[k]]$values) else NA)
    cols
}

working_cols <- add_specs (list (i = c (seq_len (n), "Sum")), extras, e_nms)
working_cols$y    <- c (ydata, sum (ydata))
working_cols$x    <- c (xdata, sum (xdata))
working_cols$yhat <- c (yhat, sum (yhat))
working_cols$e    <- c (e, sum (e))
working_cols$e2   <- c (e^2, sum (e^2))
working_cols <- add_specs (working_cols, tails, t_nms)
working <- as.data.frame (working_cols, check.names = FALSE)
working <- truncate_working (working, id_col = "i")

fmt_cols <- c (e_nms, "y", "x", "yhat", "e", "e2", t_nms)

label_args <- list (i = md ("$i$"))
for (k in seq_along (extras))
    label_args[[e_nms[k]]] <- md (sprintf ("$%s$", extras[[k]]$label))
label_args$y    <- md (sprintf ("$%s$", col_labels$y))
label_args$x    <- md (sprintf ("$%s$", col_labels$x))
label_args$yhat <- md (sprintf ("$%s$", col_labels$yhat))
label_args$e    <- md ("$e_i$")
label_args$e2   <- md ("$e_i^2$")
for (k in seq_along (tails))
    label_args[[t_nms[k]]] <- md (sprintf ("$%s$", tails[[k]]$label))

table <- gt (working) |>
    tab_header (title = "Ratio Estimation: Working Table", subtitle = md (sprintf ("$n =
    ↪  %d$", n)))
table <- do.call (cols_label, c (list (table), label_args))
table <- table |>
    cols_width (i ~ px (40), everything () ~ px (100)) |>
```

```
    fmt_auto (data = working, columns = fmt_cols, decimals = 2) |>
    sub_missing (missing_text = "...") |>
    tab_style (
        style     = cell_text (weight = "bold"),
        locations = cells_body (rows = i == "Sum")
    ) |>
    tab_footnote (
        footnote  = md (sprintf ("$\\hat B = \\sum_i %s / \\sum_i %s = %s$, $%s = \\hat B\\,
        ↪  %s$, $s_e^2 = %s$.",
                                 col_labels$y, col_labels$x, fmt_num (B_hat),
↪  col_labels$yhat, col_labels$x, fmt_num (var_e))),
        locations = cells_column_labels (columns = yhat)
    )
 ## an extra column's formula documents how it feeds the y column; a tail
 ## column's formula belongs on the tail column itself
 for (k in seq_along (extras))
    if (!is.null (extras[[k]]$formula))
        table <- table |>
            tab_footnote (
                footnote  = md (sprintf ("$%s$.", extras[[k]]$formula)),
                locations = cells_column_labels (columns = y)
            )
 for (k in seq_along (tails))
    if (!is.null (tails[[k]]$formula))
        table <- table |>
            tab_footnote (
                footnote  = md (sprintf ("$%s$.", tails[[k]]$formula)),
                locations = cells_column_labels (columns = tidyselect::all_of (t_nms[k]))
            )
 table <- table |>
    tab_source_note (
        source_note = md (if (estimate == "model")
            sprintf ("$%s = \\dfrac{\\sum_i %s}{\\sum_i %s} = \\dfrac{%s}{%s} = %s$",
                     B_label, col_labels$y, col_labels$x, fmt_num (sum (ydata)), fmt_num
↪  (sum (xdata)), fmt_num (B_hat))
        else if (estimate == "total")
            sprintf ("$\\hat t = \\hat B\\,\\bar x_U\\,N = %s \\times %s \\times %s = %s$",
                     fmt_num (B_hat), fmt_num (xbarU), fmt_num (N, 0), fmt_num (est))
        else
            sprintf ("$\\bar y_r = \\hat B\\,\\bar x_U = %s \\times %s = %s$",
                     fmt_num (B_hat), fmt_num (xbarU), fmt_num (est)))
    ) |>
    tab_source_note (
        source_note = md (if (estimate == "model")
            (if (var2 > 0)
                ## two parts: the ratio variance over the sampled units, plus
                ## whatever second-stage variance the caller supplied
                sprintf ("$\\mathrm{SE}(%s) =
                 ↪  \\sqrt{\\left(1-\\dfrac{n}{N}\\right)\\dfrac{s_e^2}{n\\,%s^2} + %s} =
                 ↪  \\sqrt{\\left(1-\\dfrac{%d}{%d}\\right)\\dfrac{%s}{%d \\times %s^2} + %s}
                 ↪  = %s$",
```

```
                        B_label, xbar_label,
                        if (is.null (var2_label)) "\\mathrm{var}_2" else var2_label,
                        n, N, fmt_num (var_e), n, fmt_num (xbar),
                        fmt_num (var2 / xbar^2), fmt_num (sd_B_hat))
            else if (is.finite (N))
                sprintf ("$\\mathrm{SE}(%s) = \\dfrac{1}{\\bar
                 ↪  x}\\sqrt{\\left(1-\\dfrac{n}{N}\\right)\\dfrac{s_e^2}{n}} =
                 ↪  \\dfrac{1}{%s}\\sqrt{\\left(1-\\dfrac{%d}{%d}\\right)\\dfrac{%s}{%d}} =
                 ↪  %s$",
                        B_label, fmt_num (xbar), n, N, fmt_num (var_e), n, fmt_num
↪  (sd_B_hat))
            else
                sprintf ("$\\mathrm{SE}(%s) = \\dfrac{1}{\\bar x}\\sqrt{\\dfrac{s_e^2}{n}} =
                 ↪  \\dfrac{1}{%s}\\sqrt{\\dfrac{%s}{%d}} = %s$",
                        B_label, fmt_num (xbar), fmt_num (var_e), n, fmt_num (sd_B_hat)))
        else if (estimate == "total")
            sprintf ("$\\mathrm{SE}(\\hat t) = \\mathrm{SE}(\\hat B)\\,\\bar x_U\\,N = %s
             ↪  \\times %s \\times %s = %s$",
                    fmt_num (sd_B_hat), fmt_num (xbarU), fmt_num (N, 0), fmt_num (sd_est))
        else
            sprintf ("$\\mathrm{SE}(\\bar y_r) = \\mathrm{SE}(\\hat B)\\,\\bar x_U = %s
             ↪  \\times %s = %s$",
                    fmt_num (sd_B_hat), fmt_num (xbarU), fmt_num (sd_est)))
    )

  list (estimate = estimate_vec, table = table)
}

## ydata --- observations of the variable of interest
## xdata --- observations of the auxiliary variable
## xbarU --- population mean of the auxiliary variable; required when
##           estimate is "mean" or "total"; ignored when estimate = "model"
## N --- population size
## estimate --- "mean" (default) returns the regression estimate of the
##              population MEAN of y, Bhat0 + Bhat1*xbarU; "total" returns
##              the regression estimate of the population TOTAL of y
##              (requires finite N); "model" returns the raw fitted slope
##              Bhat1 itself (with its own SE from the lm fit)
## show.details --- if TRUE (default), also return a gt table of the
##                  row-level working values y_i, x_i, fitted yhat_i, e_i,
##                  e_i^2, with a Sum row and a footnote showing Bhat0,
##                  Bhat1, and s_e^2; NULL otherwise
## returns list ($estimate = c(Est., S.E., ci.low, ci.upp), $table)
reg_est <- function (ydata, xdata, xbarU = NULL, N = Inf,
                     estimate = c ("mean", "total", "model"), show.details = TRUE)
{
  estimate <- match.arg (estimate)

  n <- length (ydata)
  lmfit <- lm (ydata ~ xdata)
  Bhat <- lmfit$coefficients
```

```
yhat <- lmfit$fitted.values
e <- lmfit$residuals
SSe <- sum (e^2) / (n - 2)

if (estimate == "model")
{
    est <- unname (Bhat[2])
    sd_est <- summary (lmfit)$coefficients[2, "Std. Error"]
}
else
{
    if (is.null (xbarU))
        stop ("xbarU (population mean of x) is required when estimate is \"mean\" or
         ↪  \"total\"")
    yhat_reg <- unname (Bhat[1] + Bhat[2] * xbarU)
    se_yhat_reg <- sqrt ((1-n/N) * SSe / n)
    if (estimate == "total")
    {
        if (!is.finite (N))
            stop ("N must be finite to compute estimate = \"total\"")
        est <- yhat_reg * N
        sd_est <- se_yhat_reg * N
    }
    else
    {
        est <- yhat_reg
        sd_est <- se_yhat_reg
    }
}

mem <- qt (0.975, df = n - 2) * sd_est
estimate_vec <- c (Est. = est, S.E. = sd_est, ci.low = est - mem, ci.upp = est + mem)

if (!show.details)
    return (list (estimate = estimate_vec, table = NULL))

Sxx <- sum ((xdata - mean (xdata))^2)
Sxy <- sum ((xdata - mean (xdata)) * (ydata - mean (ydata)))

working <- data.frame (
    i    = c (seq_len (n), "Sum"),
    y    = c (ydata, sum (ydata)),
    x    = c (xdata, sum (xdata)),
    yhat = c (yhat, sum (yhat)),
    e    = c (e, sum (e)),
    e2   = c (e^2, sum (e^2))
)
working <- truncate_working (working, id_col = "i")

table <- gt (working) |>
    tab_header (title = "Regression Estimation: Working Table", subtitle = md (sprintf ("$n =
     ↪  %d$", n))) |>
```

```
    cols_label (
        i    = md ("$i$"),
        y    = md ("$y_i$"),
        x    = md ("$x_i$"),
        yhat = md ("$\\hat y_i$"),
        e    = md ("$e_i$"),
        e2   = md ("$e_i^2$")
    ) |>
    cols_width (i ~ px (40), everything () ~ px (100)) |>
    fmt_auto (data = working, columns = c ("y", "x", "yhat", "e", "e2"), decimals = 2) |>
    sub_missing (missing_text = "...") |>
    tab_style (
        style     = cell_text (weight = "bold"),
        locations = cells_body (rows = i == "Sum")
    ) |>
    tab_footnote (
        footnote  = md (sprintf ("$\\hat B_0 = %s$, $\\hat B_1 = %s$, $\\hat y_i = \\hat B_0
         ↪  + \\hat B_1 x_i$, $s_e^2 = %s$.",
                                 fmt_num (Bhat[1]), fmt_num (Bhat[2]), fmt_num (SSe))),
        locations = cells_column_labels (columns = yhat)
    ) |>
    tab_source_note (
        source_note = md (if (estimate == "model")
            sprintf ("$\\hat B_1 = \\dfrac{\\sum_i(x_i-\\bar x)(y_i-\\bar
             ↪  y)}{\\sum_i(x_i-\\bar x)^2} = \\dfrac{%s}{%s} = %s$",
                     fmt_num (Sxy), fmt_num (Sxx), fmt_num (unname (Bhat[2])))
        else if (estimate == "total")
            sprintf ("$\\hat t_{reg} = N\\bar y_{reg} = %s \\times %s = %s$",
                     fmt_num (N, 0), fmt_num (yhat_reg), fmt_num (est))
        else
            sprintf ("$\\bar y_{reg} = \\hat B_0+\\hat B_1\\bar x_U = %s + %s \\times %s =
             ↪  %s$",
                     fmt_num (Bhat[1]), fmt_num (Bhat[2]), fmt_num (xbarU), fmt_num
                      ↪  (yhat_reg)))
    ) |>
    tab_source_note (
        source_note = md (if (estimate == "model")
            sprintf ("$\\mathrm{SE}(\\hat B_1) = \\sqrt{\\dfrac{s_e^2}{\\sum_i(x_i-\\bar
             ↪  x)^2}} = \\sqrt{\\dfrac{%s}{%s}} = %s$",
                     fmt_num (SSe), fmt_num (Sxx), fmt_num (sd_est))
        else if (estimate == "total")
            sprintf ("$\\mathrm{SE}(\\hat t_{reg}) = N\\times\\mathrm{SE}(\\bar y_{reg}) = %s
             ↪  \\times %s = %s$",
                     fmt_num (N, 0), fmt_num (se_yhat_reg), fmt_num (sd_est))
        else if (is.finite (N))
            sprintf ("$\\mathrm{SE}(\\bar y_{reg}) =
             ↪  \\sqrt{\\left(1-\\dfrac{n}{N}\\right)\\dfrac{s_e^2}{n}} =
             ↪  \\sqrt{\\left(1-\\dfrac{%d}{%d}\\right)\\dfrac{%s}{%d}} = %s$",
                     n, N, fmt_num (SSe), n, fmt_num (se_yhat_reg))
        else
            sprintf ("$\\mathrm{SE}(\\bar y_{reg}) = \\sqrt{\\dfrac{s_e^2}{n}} =
             ↪  \\sqrt{\\dfrac{%s}{%d}} = %s$",
```

```
                      fmt_num (SSe), n, fmt_num (se_yhat_reg)))
      )

  list (estimate = estimate_vec, table = table)
}

## sdata --- a vector of original survey data in a domain
## N --- population size
## n --- total sample size (not the sample size in the domain)
## to find total, multiply domain size N_d to the estimate returned by this function
domain_est <- function (sdata, n, N = Inf)
{
    n_d <- length (sdata)
    ybar <- mean (sdata)
    se.ybar <- sqrt((1 - n / N)) * sd (sdata) / sqrt(n_d)
    mem <- qt (0.975, df = n_d - 1) * se.ybar
    c (Est. = ybar, S.E. = se.ybar, ci.low = ybar - mem, ci.upp = ybar + mem)
}

# ------------------------------------------------------------
# Stratified Estimator
# ------------------------------------------------------------

## Given per-stratum summaries, compute the stratified (or post-stratified)
## mean or total.
## estimate --- "mean" (default) returns the stratified MEAN estimate;
##              "total" returns the stratified TOTAL estimate
## show.details --- if TRUE (default), also return a gt table showing the
##                  stratum-by-stratum working, with a Total row
## for poststratification, use nh = n * Nh/N
## returns list ($estimate = c(Est., S.E., ci.low, ci.upp), $table)
str_est <- function (ybarh, sh, nh, Nh, estimate = c ("mean", "total"), show.details = TRUE)
{
    estimate <- match.arg (estimate)

    N    <- sum (Nh)
    Pi_h <- Nh / N
    v_h  <- (1 - nh / Nh) * Pi_h^2 * sh^2 / nh

    ybar   <- sum (ybarh * Pi_h)
    seybar <- sqrt (sum (v_h))

    if (estimate == "total")
    {
        est    <- N * ybar
        se.est <- N * seybar
    }
    else
    {
        est    <- ybar
        se.est <- seybar
```

```
}
mem          <- 1.96 * se.est
estimate_vec <- c (Est. = est, S.E. = se.est, ci.low = est - mem, ci.upp = est + mem)

if (!show.details)
    return (list (estimate = estimate_vec, table = NULL))

stratum <- names (Nh)
if (is.null (stratum)) stratum <- seq_along (Nh)

working <- data.frame (
    Stratum       = c (stratum, "Total"),
    Nh            = c (Nh, N),
    nh            = c (nh, sum (nh)),
    Pi_h          = c (Pi_h, sum (Pi_h)),
    ybarh         = c (ybarh, NA),
    sh2           = c (sh^2, NA),
    weighted_mean = c (Pi_h * ybarh, ybar),
    v_h           = c (v_h, seybar^2)
)

table <- gt (working) |>
    tab_header (title = "Stratified Mean Estimation: Working Table") |>
    cols_label (
        Stratum       = md ("Stratum"),
        Nh            = md ("$N_h$"),
        nh            = md ("$n_h$"),
        Pi_h          = md ("$\\pi_h$"),
        ybarh         = md ("$\\bar y_h$"),
        sh2           = md ("$s_h^2$"),
        weighted_mean = md ("$\\pi_h\\bar y_h$"),
        v_h           = md ("$v_h$")
    ) |>
    cols_width (c (Nh, nh) ~ px (60), everything () ~ px (110)) |>
    fmt_number (columns = c (Nh, nh), decimals = 0) |>
    fmt_number (columns = Pi_h, decimals = 2) |>
    fmt_auto (data = working, columns = c ("ybarh", "sh2", "weighted_mean", "v_h"),
    ↪  decimals = 2) |>
    sub_missing (missing_text = "") |>
    tab_style (
        style     = cell_text (weight = "bold"),
        locations = cells_body (rows = Stratum == "Total")
    ) |>
    tab_footnote (
        footnote  = md ("$\\pi_h = N_h/N$ is the stratum weight."),
        locations = cells_column_labels (columns = Pi_h)
    ) |>
    tab_footnote (
        footnote  = md ("$v_h = (1-n_h/N_h)\\,\\pi_h^2\\, s_h^2/n_h$ is stratum $h$'s
        ↪  contribution to $V(\\bar y_{str}) = \\sum_h v_h$."),
        locations = cells_column_labels (columns = v_h)
```

```
        ) |>
        tab_source_note (
            source_note = md (if (estimate == "total")
                sprintf ("$\\hat t_{str} = N\\bar y_{str} = %s \\times %s = %s$",
                         fmt_num (N, 0), fmt_num (ybar), fmt_num (est))
            else
                sprintf ("$\\bar y_{str} = \\sum_h \\pi_h\\bar y_h = %s$", fmt_num (ybar)))
        ) |>
        tab_source_note (
            source_note = md (if (estimate == "total")
                sprintf ("$\\mathrm{SE}(\\hat t_{str}) = N \\times \\mathrm{SE}(\\bar y_{str})
                 ↪  = %s \\times %s = %s$",
                         fmt_num (N, 0), fmt_num (seybar), fmt_num (se.est))
            else
                sprintf ("$\\mathrm{SE}(\\bar y_{str}) = \\sqrt{\\sum_h v_h} = \\sqrt{%s} =
                 ↪  %s$",
                         fmt_num (sum (v_h)), fmt_num (seybar)))
        )

    list (estimate = estimate_vec, table = table)
}

## this function finds statistical estimates given a dataset with sampling weight
# stratdata --- data.frame containing stratified sample
# y --- name of variable for which we want to estiamte population mean
# stratum --- name of variable that will be used as stratum variable
# weight --- name of variable indicating sampling weight
# note: from weights we can find Nh (see the code for formula)
# returns the same list ($estimate, $table) as str_est(), which
# this function calls after deriving ybarh/sh/nh/Nh from the raw data
str_est_data <- function (stratdata, y, stratum, weight, estimate = c ("mean", "total"),
 ↪  show.details = TRUE)
{
    ## compute stratum-wise data
    sh <- tapply (stratdata[, y], stratdata[,stratum], sd)
    ybarh <- tapply (stratdata[, y], stratdata[,stratum], mean)
    ## find population stratum size using sampling weight included in the data set
    Nh <- tapply (stratdata[, weight], stratdata[,stratum], sum)
    nh <- tapply (1:nrow(stratdata), stratdata[,stratum], length)

    str_est (ybarh, sh, nh, Nh, estimate = estimate, show.details = show.details)
}

# ------------------------------------------------------------
# Cluster (ratio-to-size) Estimator
# ------------------------------------------------------------

## Cluster ratio estimation (one-stage if Mi = mi, i.e. every element in the
## sampled cluster is measured; two-stage/ratio-to-size if only mi < Mi
## elements are subsampled, so t_hat_i = Mi * ybari is an ESTIMATED total).
## data --- original data frame for holding data
```

```r
## cname --- variable recording cluster (psu) identity
## csize --- variable recording cluster (psu) population size (not sample size)
## yvar --- variable of interest
## N --- total number of clusters (psus) in the population
## estimate --- "mean" (default) or "model" both return the raw ratio
##              Bhat = sum(t_hat_i)/sum(Mi) --- for this ratio-to-size
##              cluster estimator, that raw B IS the population MEAN per
##              element, so the two coincide; "total" returns the
##              population TOTAL of y, Bhat*Mtotal_U (requires Mtotal_U,
##              the population total of the cluster-size variable)
## Mtotal_U --- population total of the cluster-size variable; required
##              when estimate = "total"
## show.details --- if TRUE, also return a gt table of the cluster-level
##                  working values (m_i, ybar_i, s_i^2, t_hat_i, M_i,
##                  fitted, e_i, e_i^2, v_hat_i), via ratio_est()'s own
##                  working table
##
## Variance, in two readable parts (Lohr, Sampling: Design and Analysis):
##   $\hat V(\bar y_r) = \hat V_{\mathrm{ratio}} + (n/N)\, \hat V_{\mathrm{str}}$.
## The first part is the variance of the RATIO estimate $\sum \hat t_i /
## \sum M_i$ formed from the n sampled clusters,
##   $\hat V_{\mathrm{ratio}} = (1 - n/N)\, s_e^2 / (n \bar M^2)$, with
##   $s_e^2 = \sum_i (\hat t_i - \bar y_r M_i)^2/(n-1)$,
## and is all a one-stage sample needs. The second part is the variance of a
## STRATIFIED sample in which the n sampled clusters are the strata, of sizes
## $M_i$ inside $M = \sum_{i \in S} M_i = n \bar M$ elements:
##   $\hat V_{\mathrm{str}} = \sum_{i \in S} \hat v_i$,
##   $\hat v_i = (M_i/M)^2 (1 - m_i/M_i)\, s_i^2 / m_i$,
## where $m_i$ is the number of elements measured in cluster $i$ and $s_i^2$
## their sample variance. Sub-sampling makes $\hat t_i = M_i \bar y_i$ an
## estimate rather than a total, and $s_e^2$ recovers only the fraction
## $1 - n/N$ of that noise; the $(n/N)\hat V_{\mathrm{str}}$ term adds back
## the rest. It vanishes for a one-stage sample ($m_i = M_i$) and for an
## unknown or infinite $N$, and at $n = N$ it is the whole variance --- a
## census of the clusters IS a stratified sample, and the formula says so.
## returns list ($estimate = c(Est., S.E., ci.low, ci.upp), $table)
cluster_ratio <- function (data, cname, csize, yvar, N = Inf,
                           estimate = c ("mean", "total", "model"),
                           show.details = TRUE, Mtotal_U = NULL)
{
  estimate <- match.arg (estimate)

  clust <- data[, cname]
  ydata <- data[, yvar]

  ybari <- tapply (ydata, clust, mean)
  mi    <- tapply (ydata, clust, length)           # elements measured in cluster i
  si2   <- tapply (ydata, clust, function (v) if (length (v) > 1) var (v) else 0)
  Mi    <- tapply (data[, csize], clust, function (x) x[1])
  ## same as the cluster total if all Mi elements were measured (mi = Mi)
  t_hat_cls <- ybari * Mi
```

```
  n <- length (Mi)

  ## Second-stage contributions: treat the n sampled clusters as strata of
  ## sizes M_i within M = sum_{i in S} M_i = n Mbar elements, so sum(vi) is
  ## exactly the stratified variance V_str and the term to add is (n/N) V_str.
  ## ratio_est() divides by xbar^2 = Mbar^2, so var2 carries that factor back.
  M          <- sum (Mi)
  vi         <- (Mi / M)^2 * (1 - mi / Mi) * si2 / mi
  subsampled <- any (mi < Mi)
  two_stage  <- subsampled && is.finite (N)
  var2       <- if (two_stage) (n / N) * sum (vi) * mean (Mi)^2 else 0

  ybar_col <- list (label = "\\bar y_i", values = ybari, formula = "\\hat t_i = \\bar y_i \\,
↪  M_i")
  m_col    <- list (label = "m_i",   values = mi,  total = TRUE)
  s2_col   <- list (label = "s_i^2", values = si2)
  v_col    <- list (label = "\\hat v_i", values = vi, total = TRUE,
                    formula = paste0 ("\\hat v_i = \\left(\\dfrac{M_i}{M}\\right)^2",
                                      "\\left(1-\\dfrac{m_i}{M_i}\\right)\\dfrac{s_i^2}{m_i},",
                                      "\\quad M = \\sum_i M_i = n\\bar M,",
                                      "\\quad \\sum_i \\hat v_i = \\hat V_{\\mathrm{str}}"))

  ## m_i is shown whenever the clusters were sub-sampled; s_i^2 and v_hat_i
  ## only when they actually enter the standard error
  extras <- if (two_stage) list (m_col, ybar_col, s2_col)
            else if (subsampled) list (m_col, ybar_col)
            else list (ybar_col)
  tails  <- if (two_stage) list (v_col) else NULL
  v2_lab <- "\\dfrac{n}{N}\\sum_i \\hat v_i"

  if (estimate == "total")
  {
      if (is.null (Mtotal_U))
          stop ("Mtotal_U (population total of the cluster-size variable) is required when
           ↪  estimate = \"total\"")
      ratio_est (t_hat_cls, Mi, xbarU = Mtotal_U, N = N, estimate = "mean", show.details =
       ↪  show.details,
                 col_labels = list (y = "\\hat t_i", x = "M_i", yhat = "\\hat B M_i"),
                 extra_col = extras, tail_col = tails, var2 = var2, var2_label = v2_lab,
                 xbar_label = "\\bar M")
  }
  else
  {
      ratio_est (t_hat_cls, Mi, N = N, estimate = "model", show.details = show.details,
                 col_labels = list (y = "\\hat t_i", x = "M_i", yhat = "\\hat B M_i"),
                 extra_col = extras, tail_col = tails, var2 = var2, var2_label = v2_lab,
                 xbar_label = "\\bar M", B_label = "\\bar y_r")
  }
}

# ------------------------------------------------------------
```

```r
# UPS Estimators
# ------------------------------------------------------------

## Total estimation for datasets collected with UPS with Replacement
## (psi-estimator / Hansen-Hurwitz).
## estimate --- "total" (default) is this function's whole purpose, computed
##              as the SAMPLE MEAN of the expanded pseudo-values t_i/psi_i
##              (a with-replacement HT-type property); "mean" instead
##              divides that total by N (requires finite N)
upswr_total <- function (total, psi, N = Inf, estimate = c ("total", "mean"), show.details =
 ↪  TRUE)
{
  estimate <- match.arg (estimate)
  ## N = Inf: draws are with replacement, so no finite population correction
  res <- srs_est (total/psi, N = Inf, estimate = "mean", show.details = show.details,
                  col_labels = list (y = "t_i/\\psi_i", yhat = "\\hat t_{\\mathrm{HH}}",
                                     ybar = "\\hat t_{\\mathrm{HH}}", s = "s_u"),
                  title = "UPSWR Hansen-Hurwitz Estimation: Working Table")
  if (estimate == "mean")
  {
      if (!is.finite (N)) stop ("N must be finite to compute estimate = \"mean\" (mean total
       ↪  per psu)")
      res$estimate <- res$estimate / N
  }
  res
}

## UPSWR Hansen-Hurwitz estimation of B = t/t_x when the drawing probabilities
## are psi_i = x_i / t_x, built from the auxiliary variable x. Works with the
## ratios b_j* = y_j*/x_j* of the n draws (repeats kept): Bhat = mean(b_j*)
## estimates B with SE s_b*/sqrt(n) (no fpc, draws are independent).
## ydata, xdata --- y and x of the n draws
## decimals --- decimals shown in the working table
## returns list($estimate = c(Est., S.E., ci.low, ci.upp) of B, $table)
## To estimate the mean or the total of y, multiply $estimate by the known mean
## xbar_U or total t_x of x. With x = psi_i (total 1), B = t, so the estimate is
## already the total.
upswr_est <- function (ydata, xdata, show.details = TRUE, decimals = 2)
{
  srs_est (ydata/xdata, N = Inf, estimate = "mean", show.details = show.details,
           col_labels = list (y = "b_j^*", yhat = "\\hat B_{\\mathrm{HH}}",
                              dev = "b_j^*-\\hat B_{\\mathrm{HH}}",
                              ybar = "\\hat B_{\\mathrm{HH}}", s = "s_{b^*}"),
           title = "UPSWR Hansen-Hurwitz Estimation: Working Table",
           decimals = decimals)
}

## UPSWR (Hansen-Hurwitz) estimation for cluster sampling, with clusters
## drawn with replacement with probabilities psi_i = M_i / M0. It calls
## upswr_est() with y = t (cluster totals) and x = M (cluster sizes), so the
## ratio b_j* = t_j*/M_j* is the mean per element (ssu-level mean) ybar_j* of
```

```r
## the cluster of draw j, and reports the results in cluster notation:
##   ybar --- mean per element, ybar_U = t/M0, estimated by ybarhat_HH = mean(ybar_j*)
##   tbar --- mean per cluster (psu-level mean of the totals), tbar_U = t/N,
##            estimated by Mbar_U * ybarhat_HH
##   total --- population total t, estimated by M0 * ybarhat_HH
## ti --- cluster totals of the n draws (repeats kept); in a two-stage design,
##        the estimated totals M_i * ybar_i from the subsamples
## Mi --- cluster sizes of the n draws
## N, M0 --- number of clusters and number of elements in the population
## estimate --- which of the three quantities to return ("ybar" default)
## cluster --- optional labels of the drawn clusters, shown in the working table
## t_label --- RAW latex for the totals column ("\\hat t_j^*" for two stages)
## returns list ($estimate = c(Est., S.E., ci.low, ci.upp), $table)
cluster_upswr <- function (ti, Mi, N, M0, estimate = c ("ybar", "tbar", "total"),
                           show.details = TRUE, cluster = NULL, t_label = "t_j^*",
                           decimals = 4)
{
    estimate <- match.arg (estimate)
    Mbar_U <- M0 / N
    res <- upswr_est (ti, Mi, show.details = FALSE)      # estimates ybar_U
    res$estimate <- res$estimate * switch (estimate, ybar = 1, tbar = Mbar_U, total = M0)
    if (!show.details) return (res)

    ## working table on the scale of the cluster means ybar_j*
    n <- length (ti)
    ybar_j <- ti / Mi
    ybar_hh <- mean (ybar_j)
    dev <- ybar_j - ybar_hh
    s2 <- sum (dev^2) / (n - 1)
    se_ybar <- sqrt (s2 / n)
    working <- data.frame (
        j    = c (seq_len (n), "Sum"),
        cl   = c (if (is.null (cluster)) rep ("", n) else as.character (cluster), ""),
        M    = c (Mi, NA),
        t    = c (ti, NA),
        ybar = c (ybar_j, sum (ybar_j)),
        dev  = c (dev, sum (dev)),
        dev2 = c (dev^2, sum (dev^2))
    )
    working <- truncate_working (working, id_col = "j", head_n = 10)

    yh <- "\\hat{\\bar y}_{\\mathrm{HH}}"
    th <- "\\hat{\\bar t}_{\\mathrm{HH}}"
    Th <- "\\hat t_{\\mathrm{HH}}"
    fn <- function (x, d = decimals) fmt_num (x, decimals = d)
    table <- gt (working) |>
        tab_header (title = "UPSWR Cluster Sampling (Hansen-Hurwitz): Working Table",
                    subtitle = md (sprintf ("$n = %d$ draws, $N = %s$, $M_0 = %s$, $\\bar M_U =
                     ↪  M_0/N = %s$",
                                            n, fn (N, 0), fn (M0, 0), fn (Mbar_U, 2)))) |>
        cols_label (
```

```
            j    = md ("$j$"),
            cl   = "Cluster",
            M    = md ("$M_j^*$"),
            t    = md (sprintf ("$%s$", t_label)),
            ybar = md (sprintf ("$\\bar y_j^* = %s/M_j^*$", t_label)),
            dev  = md (sprintf ("$\\bar y_j^* - %s$", yh)),
            dev2 = md (sprintf ("$(\\bar y_j^* - %s)^2$", yh))
        ) |>
        fmt_number (columns = c ("M", "t"), decimals = if (all (ti == round (ti))) 0 else 1) |>
        fmt_auto (data = working, columns = c ("ybar", "dev", "dev2"), decimals = decimals) |>
        sub_missing (missing_text = "") |>
        tab_style (style = cell_text (weight = "bold"),
                   locations = cells_body (rows = j == "Sum")) |>
        tab_footnote (
            footnote = md (sprintf ("$s_{\\bar y^*}^2 = \\sum_j (\\bar y_j^* - %s)^2/(n-1) =
             ↪  %s$, $s_{\\bar y^*} = %s$.",
                                    yh, fn (s2), fn (sqrt (s2)))),
            locations = cells_column_labels (columns = dev2)) |>
        tab_source_note (md (sprintf (
            "Mean per element: $%s = \\dfrac{\\sum_j \\bar y_j^*}{n} = \\dfrac{%s}{%d} = %s$,
             ↪  $\\mathrm{SE}(%s) = \\dfrac{s_{\\bar y^*}}{\\sqrt n} = %s$",
            yh, fn (sum (ybar_j)), n, fn (ybar_hh), yh, fn (se_ybar)))) |>
        tab_source_note (md (sprintf (
            "Mean per cluster: $%s = \\bar M_U\\,%s = %s \\times %s = %s$, $\\mathrm{SE}(%s) =
             ↪  %s$",
            th, yh, fn (Mbar_U, 2), fn (ybar_hh), fn (Mbar_U * ybar_hh, 2), th, fn (Mbar_U *
↪  se_ybar, 2)))) |>
        tab_source_note (md (sprintf (
            "Total: $%s = M_0\\,%s = %s \\times %s = %s$, $\\mathrm{SE}(%s) = %s$",
            Th, yh, fn (M0, 0), fn (ybar_hh), fn (M0 * ybar_hh, 2), Th, fn (M0 * se_ybar, 2))))
    if (is.null (cluster)) table <- cols_hide (table, "cl")

    list (estimate = res$estimate, table = table)
}

## Variance estimate of the Horvitz-Thompson total sum(y/pik) for a fixed-size
## sample drawn without replacement with inclusion probabilities pik.
## y, pik --- values and inclusion probabilities of the n sampled units
## method --- "hajek": Hajek (1964) approximation, which needs only pik (it is
##             the Sen-Yates-Grundy estimator with pi_ij approximated by
##             pi_i pi_j [1 - (1 - pi_i)(1 - pi_j) / sum_k pi_k (1 - pi_k)])
##             v = n/(n-1) sum (1 - pi_i) (y_i/pi_i - A)^2,
##             A = sum (1 - pi_i) y_i/pi_i / sum (1 - pi_i)
##             "wr": the with-replacement approximation, treating the units as
##             n draws with psi_i = pik/n; ignores the factor (1 - pi_i),
##             so it is conservative
##             v = n/(n-1) sum (y_i/pi_i - that/n)^2
ht_var <- function (y, pik, method = c ("hajek", "wr"))
{
  method <- match.arg (method)
  n <- length (y)
```

```
  u <- y / pik
  if (method == "hajek")
  {
    A <- sum ((1 - pik) * u) / sum (1 - pik)
    n / (n - 1) * sum ((1 - pik) * (u - A)^2)
  }
  else
    n / (n - 1) * sum ((u - sum (u) / n)^2)
}

## Horvitz-Thompson estimation for UPS without replacement (fixed sample size),
## with pik = n x / tx_pop proportional to the size variable x.
## ydata, pik --- y and inclusion probabilities of the n sampled units
## N --- number of units in the population
## tx_pop --- population total of x (needed for method = "ratio")
## method --- "total": that = sum(y/pik)
##            "mean": that / N
##            "ratio": Bhat = that / tx_pop, the total of y per unit of x
## variance --- see ht_var()
upswo_est <- function (ydata, pik, N, tx_pop = NULL,
                       method = c ("total", "mean", "ratio"),
                       variance = c ("hajek", "wr"))
{
  method <- match.arg (method)
  variance <- match.arg (variance)
  n <- length (ydata)
  if (method == "ratio" && is.null (tx_pop))
    stop ("tx_pop is required for method = \"ratio\"")

  scale <- switch (method, total = 1, mean = 1 / N, ratio = 1 / tx_pop)
  est <- scale * sum (ydata / pik)
  se <- scale * sqrt (ht_var (ydata, pik, variance))
  mem <- qt (0.975, df = n - 1) * se
  c (Est. = est, S.E. = se, ci.low = est - mem, ci.upp = est + mem)
}
```