Exercise Sheet: Generating and Measuring Data

Data, Data Storage, Data Collection

Exercise 1 – Remote Working and Employee Stress

A research team wants to investigate the following question:

Does working from home reduce employees' stress during the working day?

The team proposes the following study:

1. Recruit volunteers through an advertisement posted on the company's internal website.
2. Ask each volunteer whether they prefer to work from home or from the office.
3. Ask participants to complete a stress questionnaire at the end of each working day, specifying whether they worked from home or in the office on that day.

(**Q1**) What is the target population, what is the access frame, and what is the sample?

The target population is the group about which the researchers want to draw conclusions, here all employees of the company.
The access frame is the employees who actually see the internal advertisement and are able to volunteer. It can be narrower than the full target population if, for instance, some employees rarely check the internal website.
The sample is the set of employees who volunteer and actually participate in the study.

(**Q2**) Is the collected data observational, experimental or simulated? Explain.

It is observational. Participants choose their working location rather than being randomly assigned to work from home or from the office.

(**Q3**) Identify at least two distinct sources of error or limitation in this study. For each, explain why collecting more participants would or would not solve the problem.

Possible answers include:

- volunteer/self-selection, which can make the sample unrepresentative;
- confounding caused by participants choosing their working location (e.g., employees with young children may be more likely to work from home, and they may also experience different levels of stress.);
- measurement bias in the questionnaire;
- random measurement variation;
- low temporal resolution caused by measuring only once per day;
- limited external validity if the volunteers come from only one department or team.

Increasing the number of participants can reduce random sampling variation and can improve the precision of estimates. It does not automatically solve systematic problems such as:

- self-selection or coverage problems;
- confounding caused by the study design;
- measurement bias in the questionnaire.

Exercise 2 – Personalized Product Recommendation

An online retailer wants to know whether a new *personalized product recommendation* feature increases average order value. The data team lets customers opt in to a beta program to try the new feature, then compares the average order value of beta users against non-beta users over the following month.

(**Q1**) Is the collected data observational, experimental or simulated? Explain.

It is observational. Customers choose for themselves whether to join the beta program, so the researcher does not control who receives the feature.

(**Q2**) Beta users spend, on average, 40% more per order than non-beta users. Identify the independent and dependent variables. Propose one plausible confounding variable.

The independent variable is whether the user uses the personalized product recommendation and the dependent variable is the order value. Customers who opt in to try new features are plausibly more engaged, frequent shoppers to begin with (e.g., loyalty-program members, or people who already browse the site more). This existing engagement level could independently drive both the decision to opt in *and* higher spending, producing a correlation between using the feature and order value with no causal link between them.

(**Q3**) Redesign the study so that it supports a causal claim. What, concretely, would the retailer need to change? Does it change the status (observational, experimental or simulated) of the data?

Instead of letting customers self-select into the beta program, the retailer should *randomly assign* customers to see the new recommendation feature or not. This spreads baseline engagement level (and any other confounder) evenly across both groups, on average, so any remaining difference in order value can be attributed to the feature itself. The redesign does change the status of the data: since the retailer now controls which customers receive the feature, rather than customers choosing for themselves, the data becomes *experimental* rather than observational.

(**Q4**) Suppose the retailer randomly assigns the feature to customers who have already made at least one purchase in the last year. How well will the study generalize to all customers? Explain.

Random assignment fixes comparability *within* the group being studied, but it does not address who is *in* that group in the first place. Here, the target population (all customers) and the group actually studied (customers with a purchase in the last year) differ, so the result may not generalize to first-time visitors. This is a question of random *sampling*, not random assignment.

Exercise 3 – Fitness Tracker for Heart Rate

A fitness tracker measures heart rate by sampling once every 2 seconds.

(**Q1**) A brief, genuine spike in heart rate lasting under one second (e.g., from a sudden startle) occurs. Will the tracker record it? Is this a bias, a precision problem, or something else?

Depending on when the tracker samples the value, it might measure the peak of the spike, a part of the spike, or miss it entirely if the spike occurs between two sampling instants. This is neither bias (a systematic offset) nor imprecision (scatter around an average). It is a distinct failure mode coming from undersampling: the instrument can be perfectly accurate and still fail to represent a real, fast event because it was not looking often enough.

(**Q2**) What is the sampling frequency of the monitoring system? What is the highest heart-rate-related frequency this tracker can faithfully capture, according to the Nyquist–Shannon sampling theorem?

One measurement is taken every 2 seconds, so

$$f_s = \frac{1}{2} = 0.5 \text{ Hz}.$$

By Nyquist–Shannon sampling theorem, faithful reconstruction requires sampling at more than twice the highest frequency present in the signal. With a sampling rate of 0.5 Hz, the highest frequency that can be faithfully captured is just under 0.25 Hz.

(**Q3**) A signal $\cos(2\pi f_0 t)$ of true frequency $f_0 = 13$ Hz is sampled at $f_s = 10$ Hz. Identify the frequency $f_{\text{alias}} < f_s/2$ of a cosine that produces the exact same sampled values.

Since $\cos(2\pi \times 13n/10) = \cos(2\pi n + 2\pi \times 3n/10) = \cos(2\pi \times 3n/10)$ (using 2π-periodicity to drop the integer multiple of $2\pi n$), the samples are identical to those of a 3 Hz signal. So $f_{\text{alias}} = 3$ Hz.

(**Q4**) Generalize: for a signal of frequency f_0 sampled at rate f_s, propose a formula for the apparent frequency f_{alias} in terms of f_0 and f_s, valid when $f_s/2 < f_0 < f_s$. Check your formula against the previous question.

For $f_s/2 < f_0 < f_s$, the apparent frequency is $f_{\text{alias}} = f_s - f_0$ (the "fold-over" around $f_s/2$). Checking: $f_s - f_0 = 10 - 13 = -3$; taking the absolute value (frequency is non-negative) gives 3 Hz, matching the previous part. More generally, for any f_0, $f_{\text{alias}} = |f_0 - k f_s|$ for the integer k that brings this value into $[0, f_s/2]$.

Exercise 4 – Monte Carlo: How Many Draws Do We Need?

In this exercise, we assume known Chebyshev's inequality

$$\Pr\left(|X - \mathbb{E}[X]| \geq \epsilon\right) \leq \frac{\text{Var}(X)}{\epsilon^2}.$$

One standard proof relies on Markov's inequality.

(**Q1**) Let $X_1, \ldots, X_n$ be independent Bernoulli random variables of parameter p, and let $\hat{p} = \frac{1}{n}\sum_{i=1}^{n} X_i$. Using Chebyshev's inequality, show that, for any $\epsilon > 0$,

$$\Pr\left(|\hat{p} - p| \geq \epsilon\right) \leq \frac{1}{4n\epsilon^2}.$$

By Chebyshev's inequality,

$$\Pr\left(|\hat{p} - \mathbb{E}[\hat{p}]| \geq \epsilon\right) \leq \frac{\mathrm{Var}(\hat{p})}{\epsilon^2}.$$

By linearity, $\mathbb{E}[\hat{p}] = p$ and by independence, $\mathrm{Var}(\hat{p}) = \frac{p(1-p)}{n}$. Now, bound $p(1-p)$ by $\frac{1}{4}$. It follows that

$$\Pr\left(|\hat{p} - p| \geq \epsilon\right) \leq \frac{1}{4n\epsilon^2}.$$

(**Q2**) Get a bound for $|\hat{p} - p|$ that holds with probability at least $1 - \frac{1}{t^2}$. What can you conclude for the order of the error? What hypothesis from the lecture was not needed?

We want a bound B such that

$$\Pr\left(|\hat{p} - \mathbb{E}[\hat{p}]| < B\right) \geq 1 - \frac{1}{t^2},$$

That is

$$\Pr\left(|\hat{p} - \mathbb{E}[\hat{p}]| \geq B\right) \leq \frac{1}{t^2}.$$

Thus, set

$$\frac{1}{t^2} = \frac{1}{4n\epsilon^2}.$$

Solving for ϵ, we obtain

$$\epsilon = \frac{t}{2\sqrt{n}}.$$

Thus, with any fixed confidence level, the error is of order $1/\sqrt{n}$.
The $1/\sqrt{n}$ rate does not require the Central Limit Theorem: it follows directly from Chebyshev's inequality.

(**Q3**) You want to estimate π using the disk method from class, with an error of at most $\epsilon = 0.01$, and you want this guarantee to hold with probability at least 95% (i.e., the bound above should be at most 0.05). How many draws n do you need?

Let $\hat{p}$ be the fraction of sampled points falling inside the quarter-disk. Then $p = \frac{\pi}{4}$, and $\hat{\pi} = 4\hat{p}$. Therefore

$$|\hat{\pi} - \pi| = 4|\hat{p} - p|.$$

To guarantee an error of at most 0.01 on π, we therefore need

$$|\hat{p} - p| \leq \frac{0.01}{4} = 0.0025.$$

Using the bound from above, we require

$$\frac{1}{4n(0.0025)^2} \leq 0.05.$$

Hence

$$n \geq \frac{1}{4 \times 0.0025^2 \times 0.05} = 800\,000.$$

Thus the Chebyshev bound requires 800 000 draws.

(**Q4**) Your colleague argues: "Since Chebyshev only requires independence and finite variance, and gives a valid bound for *any* n, we should always use it instead of a normal approximation, since the normal approximation requires n to be large." Identify one cost of following this advice.

Recall that approximately 95% of a normal distribution $X \sim \mathcal{N}(\mu, \sigma^2)$ lies within 1.96 standard deviations of its mean:

$$\Pr\left(|X - \mu| \leq 1.96\sigma\right) \approx 0.95.$$

Here we have $X = \hat{p}$, $\mu = p$ and $\sigma^2 = \frac{p(1-p)}{n}$. Assuming $p = \frac{\pi}{4}$, solving $1.96\sqrt{\frac{p(1-p)}{n}} = 0.0025$ for n gives

$$n = \frac{\pi}{4}\left(1 - \frac{\pi}{4}\right)\left(\frac{1.96}{0.0025}\right)^2 \approx 100\,000.$$

Chebyshev's bound is valid for any n, but it is much *looser* than the normal-approximation bound for the same confidence level: it requires roughly 8 times more samples than the CLT-based confidence interval to guarantee the same error at the same confidence level.

Exercise 5 – A Confounded Comparison (Simpson's Paradox)

A hospital compares two treatments, A and B, for a condition, testing each on patients classified as having either a Mild or Severe case.

	Mild cases		Severe cases	
	Treated	Recovered	Treated	Recovered
Treatment A	180	171	20	6
Treatment B	20	19	180	90

(**Q1**) Compute the recovery rate for Treatment A and for Treatment B, separately within Mild cases and within Severe cases. Which treatment looks better in *each* subgroup?

Mild cases: A recovers $171/180 = 95.0\%$; B recovers $19/20 = 95.0\%$. (Equal in this subgroup.)
Severe cases: A recovers $6/20 = 30.0\%$; B recovers $90/180 = 50.0\%$. B is better among severe cases.
So B is at least as good as A in both subgroups.

(**Q2**) Now compute the *overall* recovery rate for each treatment, ignoring the Mild/Severe distinction. Which treatment now looks better?

Treatment A overall: $(171 + 6)/(180 + 20) = 177/200 = 88.5\%$.
Treatment B overall: $(19 + 90)/(20 + 180) = 109/200 = 54.5\%$.
Treatment A now looks dramatically *better* overall, despite being no better than B in either subgroup individually — the ordering has reversed.

(**Q3**) Explain this reversal using the vocabulary of confounding introduced in class. Identify the independent and dependent variables, as well as the confound.

Here, the independent variable is treatment assigned, and the dependent variable is the recovery status. Severity of the case is a confound: it strongly affects both which treatment a patient is likely to receive (Treatment A is given mostly to Mild cases; Treatment B mostly to Severe cases, perhaps because doctors reserve B for harder cases) and the probability of recovery (Mild cases recover far more often regardless of treatment). Because the confound is distributed so unevenly between the two treatment groups, the raw comparison of the dependent and independent variables is dominated by severity rather than by any real effect of the treatment.

(**Q4**) Using the idea of random assignment from class, explain precisely what change to this study's design would prevent this reversal from being possible.

Randomly assigning patients to Treatment A or B, regardless of severity, would ensure that Mild and Severe cases are represented in *similar proportions* in both treatment groups, on average. This would prevent severity from being unevenly distributed between groups, so it could no longer produce a reversal like the one seen here: the overall comparison would then align with the subgroup comparisons instead of contradicting them.

Exercise 6 – A Monte Carlo Simulation for Hashing

A student wants to estimate the probability that inserting n random keys into a hash table with m buckets produces at least one collision (two keys hashed to the same bucket), by simulation:

```
table = set()

def estimate_collision_prob(n, m, trials):
    collision_count = 0
    for t in range(trials):
        collided = False
        for i in range(n):
            h = random_int(0, m - 1)
            if h in table:
                collided = True
            table.add(h)
        if collided:
            collision_count += 1
    return collision_count / trials
```

Run with $n = 20$, $m = 1000$, and `trials` = 10,000, this code returns an estimated collision probability noticeably *higher* than the true value. A classmate points out a bug.

(**Q1**) Identify the bug. Propose a corrected version.

`table` is created *once*, outside the loop over trials, instead of being reset at the start of each trial. As a result, buckets filled by earlier trials remain filled for all later trials: keys from trial 1 are still "in the table" during trial 5000, making a collision more and more likely to be (spuriously) detected as `trials` progresses. A correct version would look like this:

```
def estimate_collision_prob(n, m, trials):
    collision_count = 0
    for t in range(trials):
        table = set()
        collided = False
        for i in range(n):
            h = random_int(0, m - 1)
            if h in table:
                collided = True
            table.add(h)
        if collided:
            collision_count += 1
    return collision_count / trials
```

(**Q2**) Is this a bias problem, a variance problem, both, or something the bias/variance framework doesn't capture at all? Justify.

It is a bias problem: every trial after the first is systematically more likely to report a collision than it should, since it inherits the accumulated state of all previous trials, so $\mathbb{E}[\hat{p}]$ is pushed above the true collision probability, growing worse as `trials` increases. But the bug is worse than an ordinary bias, because it also breaks an assumption the whole Monte Carlo framework depends on: that the trials $X_1, \ldots, X_n$ are *independent*. Here they are not — trial t's outcome depends on every earlier trial's insertions — so the formulas for $\mathrm{Var}(\hat{p})$, $SE(\hat{p})$, and the confidence interval derived in class no longer apply at all, not just the point estimate.

(**Q3**) Now that the bug is fixed, the student claims: "Now that each trial is correct and independent, using more trials will always shrink my error at the same $1/\sqrt{n}$ rate we saw in class." Is there a hidden assumption in this claim that could still fail?

The $1/\sqrt{n}$ rate from class assumes each *draw* is independent and identically distributed — but here, that requires `random_int` to be a genuinely independent random draw on every call, which in practice depends on the underlying pseudorandom number generator (PRNG) being properly seeded and not, e.g., reseeded identically at the start of every trial (another classic bug: seeding the PRNG with a fixed value inside the trial loop would make every trial produce the exact same sequence of hashes, destroying independence *between* trials just as surely as the shared-`table` bug did, even though each trial looks correct in isolation).