Missing Data

PSY 410: Data Science for Psychology

Dr. Sara Weston

2026-05-13

Setup

Why missing data matters

The data you don’t have

In a typical longitudinal psychology study, 30–50% of participants drop out before the final wave.

. . .

If you just delete their data, you might be throwing away the most important part of the story — because **who drops out** is rarely random.

. . .

Today we learn to detect, explore, and handle missing data honestly.

Types of missing data

Explicit missing: NA

Explicit missing means you can see the `NA`:

```r
survey <- tibble(
  participant = 1:5,
  age = c(25, NA, 30, 22, NA),
  depression = c(12, 18, NA, 10, 15)
)
```

```
survey
```

```
# A tibble: 5 x 3
  participant   age depression
        <int> <dbl>      <dbl>
1           1    25         12
2           2    NA         18
3           3    30         NA
4           4    22         10
5           5    NA         15
```

The NA values are obvious.

Implicit missing: Rows don't exist

Implicit missing means entire rows are absent:

```
appointments <- tibble(
  name = c("Alice", "Bob", "Alice", "Carol"),
  day = c("Mon", "Mon", "Wed", "Wed"),
  attended = c(TRUE, TRUE, TRUE, TRUE)
)

appointments
```

```
# A tibble: 4 x 3
  name  day   attended
  <chr> <chr> <lgl>
1 Alice Mon   TRUE
2 Bob   Mon   TRUE
3 Alice Wed   TRUE
4 Carol Wed   TRUE
```

. . .

Who didn't show up? You can't tell because they're not in the data!

Why implicit missing matters

In longitudinal studies, missing rows often mean dropout:

```
# Three-wave study
longitudinal <- tibble(
  id = c(1, 1, 1, 2, 2, 3),  # Person 2 missing wave 3, person 3 missing waves 2 and 3
  wave = c(1, 2, 3, 1, 2, 1),
  depression = c(20, 15, 12, 25, 22, 18)
)

longitudinal
```

```
# A tibble: 6 x 3
     id  wave depression
  <dbl> <dbl>      <dbl>
1     1     1         20
2     1     2         15
3     1     3         12
4     2     1         25
5     2     2         22
6     3     1         18
```

Person 2 and 3's missing waves are **implicit** — they're not `NA`, they're just absent.

Exploring missing data

Checking for NAs

```
survey <- tibble(
  id = 1:6,
  age = c(25, NA, 30, 22, NA, 28),
  depression = c(12, 18, NA, 10, 15, NA),
  anxiety = c(15, 20, 12, NA, 18, 16)
)

# Check if any NAs exist
any(is.na(survey))
```

```
[1] TRUE
```

. . .

```
# Count total NAs
sum(is.na(survey))
```

```
[1] 5
```

NAs by column

```
# Count NAs in each column
survey |>
  summarize(
    age_missing = sum(is.na(age)),
    depression_missing = sum(is.na(depression)),
    anxiety_missing = sum(is.na(anxiety))
  )
```

```
# A tibble: 1 x 3
  age_missing depression_missing anxiety_missing
        <int>              <int>           <int>
1           2                  2               1
```

Better approach: across()

```
survey |>
  summarize(
    across(
      everything(),
      ~sum(is.na(.x))
    )
  )
```

```
# A tibble: 1 x 4
     id   age depression anxiety
  <int> <int>      <int>   <int>
1     0     2          2       1
```

. . .

Or as proportions:

```
survey |>
  summarize(
    across(
      everything(),
      ~mean(is.na(.x))
    )
  )
```

```
# A tibble: 1 x 4
     id   age depression anxiety
  <dbl> <dbl>      <dbl>   <dbl>
1     0 0.333      0.333   0.167
```

Which rows have missing data?

```
# Show only rows with any NA
survey |>
  filter(if_any(everything(), is.na))
```

```
# A tibble: 5 x 4
     id   age depression anxiety
  <int> <dbl>      <dbl>   <dbl>
1     2    NA         18      20
2     3    30         NA      12
3     4    22         10      NA
4     5    NA         15      18
5     6    28         NA      16
```

. . .

```
# Show only complete cases
survey |>
  filter(if_all(everything(), ~!is.na(.x)))
```

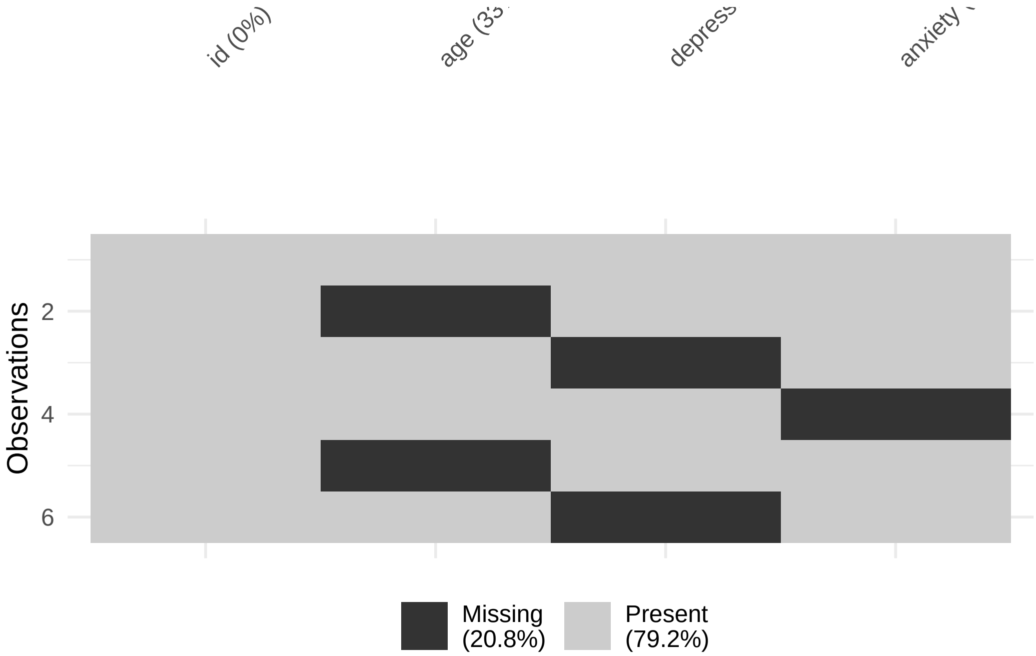
```
# A tibble: 1 x 4
     id   age depression anxiety
  <int> <dbl>      <dbl>   <dbl>
1     1    25         12      15
```

Visualizing missingness

The `naniar` package provides great visualization tools:

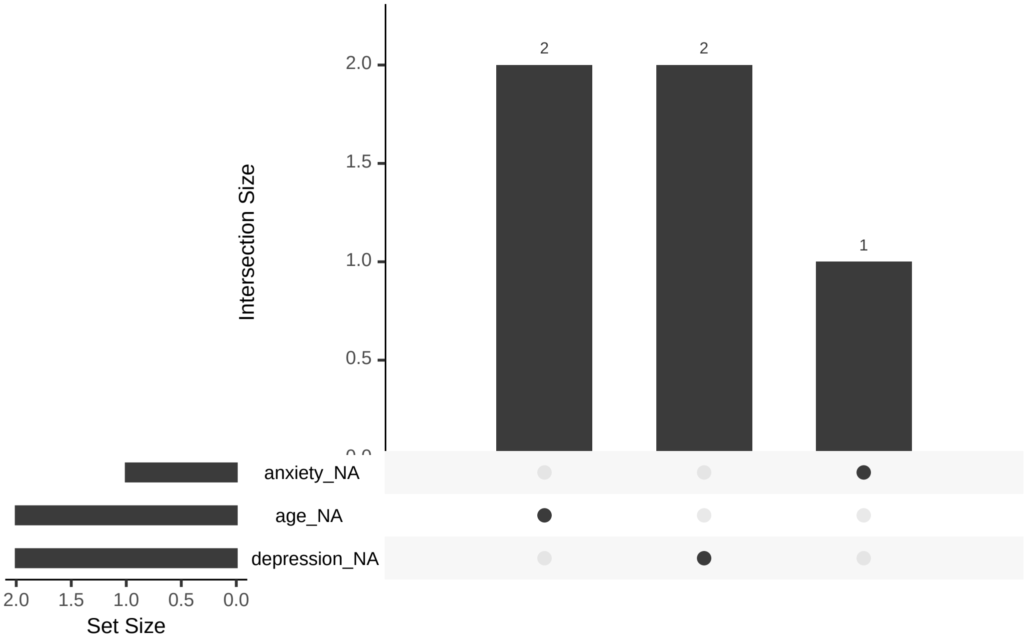
```
library(naniar)

# Visual summary
vis_miss(survey)
```



Patterns of missingness

```
# Are certain combinations of variables missing together?
gg_miss_upset(survey)
```



. . .

This shows which combinations of variables are missing in the same rows.

Handling missing data

Strategy 1: Complete case analysis

Complete case analysis (listwise deletion) removes any row with *any* `NA`:

```
# Base R way
na.omit(survey)
```

```
# A tibble: 1 x 4
     id   age depression anxiety
  <int> <dbl>      <dbl>   <dbl>
1     1    25         12      15
```

. . .

```
# tidyverse way
survey |>
  drop_na()
```

```
# A tibble: 1 x 4
     id   age depression anxiety
  <int> <dbl>      <dbl>   <dbl>
1     1    25         12      15
```

Dangers of complete case analysis

```r
# Started with 6 participants
nrow(survey)
```

```
[1] 6
```

```r
# Only 2 complete cases
survey |>
  drop_na() |>
  nrow()
```

```
[1] 1
```

. . .

> ⚠ Warning
>
> You just lost 67% of your data!

Selective dropping

Only drop rows missing *specific* variables:

```r
# Drop only if depression is missing
survey |>
  drop_na(depression)
```

```
# A tibble: 4 x 4
     id   age depression anxiety
  <int> <dbl>      <dbl>   <dbl>
1     1    25         12      15
2     2    NA         18      20
3     4    22         10      NA
4     5    NA         15      18
```

. . .

```
# Drop only if depression OR anxiety is missing
survey |>
  drop_na(depression, anxiety)
```

```
# A tibble: 3 x 4
     id   age depression anxiety
  <int> <dbl>      <dbl>   <dbl>
1     1    25         12      15
2     2    NA         18      20
3     5    NA         15      18
```

When is dropping okay?

Dropping is fine when:

- Missing data is rare (< 5-10%)
- Missingness is truly random
- You have adequate sample size

. . .

Be cautious when:

- Missing data is common (> 20%)
- Certain groups have more missingness (systematic bias)
- Sample size is small

Strategy 2: Filling values

Sometimes you can **reasonably fill** missing values:

```
# Carry forward the last observation
time_series <- tibble(
  day = 1:5,
  mood = c(5, NA, NA, 4, 6)
)

time_series |>
  fill(mood)  # Fills downward by default
```

```
# A tibble: 5 x 2
    day  mood
  <int> <dbl>
1     1     5
2     2     5
3     3     5
4     4     4
5     5     6
```

Fill directions

```
# Fill upward
time_series |>
  fill(mood, .direction = "up")
```

```
# A tibble: 5 x 2
    day  mood
  <int> <dbl>
1     1     5
2     2     4
3     3     4
4     4     4
5     5     6
```

. . .

```
# Fill both ways
time_series |>
  fill(mood, .direction = "updown")
```

```
# A tibble: 5 x 2
    day  mood
  <int> <dbl>
1     1     5
2     2     4
3     3     4
4     4     4
5     5     6
```

When is filling okay?

Filling makes sense for:

- Time series with repeated measures (carry forward last observation)
- Grouping variables that apply to multiple rows
- Values that don't change often

. . .

> ⚠ Warning
>
> **NEVER fill** when it means making up data you don't have!

Strategy 3: Replace with a specific value

```
# Replace NAs with a value
survey |>
  mutate(
    age = replace_na(age, 99),  # Code 99 = "no response"
    depression = replace_na(depression, -999)  # Obvious invalid code
  )
```

```
# A tibble: 6 x 4
     id   age depression anxiety
  <int> <dbl>      <dbl>   <dbl>
1     1    25         12      15
2     2    99         18      20
3     3    30       -999      12
4     4    22         10      NA
5     5    99         15      18
6     6    28       -999      16
```

. . .

> ❗ Important
>
> If you use placeholder codes, **document them clearly** and make sure they can't be mistaken for real data.

Strategy 4: Leave them as NA

Often the best approach is to **keep NAs** and handle them in analysis:

```
# Most functions have na.rm argument
survey |>
  summarize(
    mean_age = mean(age, na.rm = TRUE),
    mean_depression = mean(depression, na.rm = TRUE)
  )
```

```
# A tibble: 1 x 2
  mean_age mean_depression
     <dbl>           <dbl>
1     26.2            13.8
```

. . .

This is transparent about what data you have.

Implicit missing → Explicit missing

The problem with implicit missing

```
study_completion <- tibble(
  participant = c(1, 1, 1, 2, 2, 3, 3),
  timepoint = c(1, 2, 3, 1, 2, 1, 3),
  depression = c(20, 15, 12, 25, 22, 18, 14)
)

study_completion
```

```
# A tibble: 7 x 3
  participant timepoint depression
        <dbl>     <dbl>      <dbl>
1           1         1         20
2           1         2         15
3           1         3         12
4           2         1         25
```

```
5              2           2           22
6              3           1           18
7              3           3           14
```

. . .

Who's missing which timepoints? Hard to tell.

complete(): Make implicit missing explicit

```
study_completion |>
  complete(participant, timepoint)
```

```
# A tibble: 9 x 3
  participant timepoint depression
        <dbl>     <dbl>      <dbl>
1           1         1         20
2           1         2         15
3           1         3         12
4           2         1         25
5           2         2         22
6           2         3         NA
7           3         1         18
8           3         2         NA
9           3         3         14
```

. . .

Now we can see: Participant 2 missing timepoint 3, Participant 3 missing timepoint 2.

Why this matters

Makes dropout visible for analysis:

```
study_completion |>
  complete(participant, timepoint) |>
  group_by(timepoint) |>
  summarize(
    n_completed = sum(!is.na(depression)),
    n_missing = sum(is.na(depression))
  )
```

```
# A tibble: 3 x 3
  timepoint n_completed n_missing
      <dbl>       <int>     <int>
1         1           3         0
2         2           2         1
3         3           2         1
```

Filling after completing

Common pattern: make implicit explicit, then fill with a value:

```
study_completion |>
  complete(participant, timepoint) |>
  mutate(
    completed = if_else(is.na(depression), FALSE, TRUE)
  )
```

```
# A tibble: 9 x 4
  participant timepoint depression completed
        <dbl>     <dbl>      <dbl> <lgl>
1           1         1         20 TRUE
2           1         2         15 TRUE
3           1         3         12 TRUE
4           2         1         25 TRUE
5           2         2         22 TRUE
6           2         3         NA FALSE
7           3         1         18 TRUE
8           3         2         NA FALSE
9           3         3         14 TRUE
```

Pair coding break

Your turn: Analyze missing data patterns

You have survey data from a therapy study:

```
therapy_survey <- tibble(
  id = 1:8,
  age = c(25, 30, NA, 22, 28, NA, 35, 26),
  baseline_depression = c(22, 25, 18, 20, 24, 19, NA, 21),
```

```r
  followup_depression = c(12, 23, NA, 15, NA, NA, NA, 16),
  satisfaction = c(4, 3, NA, 5, 4, NA, NA, 5)
)
```

1. How many participants are missing baseline data? Followup data?
2. How many participants have **complete data** (no NAs anywhere)?
3. Create a version that drops rows missing followup data
4. What percentage of participants completed the followup?

Time: 10 minutes

Psychology-specific considerations

Missing data mechanisms

Statisticians distinguish three types:

1. **MCAR (Missing Completely At Random)** — Missingness unrelated to anything
2. **MAR (Missing At Random)** — Missingness related to observed variables
3. **MNAR (Missing Not At Random)** — Missingness related to the missing value itself

Example: Depression study

MCAR: Computer randomly failed to save 5% of responses

- No bias introduced

. . .

MAR: Older participants more likely to skip online surveys

- Can account for this by including age as a predictor

. . .

MNAR: People with severe depression skip the depression questionnaire

- **This is a problem** — missing values are related to what you're measuring

Why it matters

- **MCAR:** Complete case analysis is fine (but you lose power)
- **MAR:** More sophisticated methods can help (beyond this course)
- **MNAR:** No easy fix — missing data is fundamentally informative

. . .

> Tip
>
> **Your job:** Always **document and report** how much data is missing and why you think it's missing.

Attrition analysis

In longitudinal studies, always check *who* drops out:

```r
baseline <- tibble(
  id = 1:10,
  condition = rep(c("Treatment", "Control"), each = 5),
  baseline_depression = rnorm(10, 20, 5)
)

completers <- tibble(
  id = c(1, 2, 3, 7, 8, 9, 10)  # 4, 5, 6 dropped out
)

# Who dropped out?
baseline |>
  anti_join(completers, by = "id")
```

```
# A tibble: 3 x 3
     id condition baseline_depression
  <int> <chr>                   <dbl>
1     4 Treatment                23.9
2     5 Treatment                16.0
3     6 Control                  23.3
```

Attrition by condition

```
baseline |>
  anti_join(completers, by = "id") |>
  count(condition)
```

```
# A tibble: 2 x 2
  condition     n
  <chr>     <int>
1 Control       1
2 Treatment     2
```

. . .

All 3 dropouts from Treatment condition — this could bias results!

Reporting missing data

In your write-up, report:

1. **How much data is missing** (by variable)
2. **Patterns of missingness** (related to other variables?)
3. **How you handled it** (dropped? kept as NA?)
4. **Potential biases** (who's missing? does it matter?)

. . .

Example:

> "Eight participants (12%) did not complete the follow-up assessment. Dropout was unrelated to baseline depression scores (t = 1.2, p = .24) or treatment condition (^{2} = 0.8, p = .37). Analyses used complete case analysis (N = 60)."

Advanced topic: Multiple imputation

Beyond this course

More sophisticated approaches exist for handling missing data:

- **Multiple imputation** — create multiple plausible versions of missing data
- **Maximum likelihood** — estimate parameters using all available data

- **Bayesian methods** — incorporate uncertainty about missing values

. . .

Packages in R: `mice`, `Amelia`, `missForest`

. . .

> ℹ Note
>
> We won't cover these methods, but know they exist for when you need them in future research!

End-of-deck exercise

Practice: Longitudinal missing data

You have a three-wave study:

```r
longitudinal_study <- tibble(
  participant = c(1, 1, 1, 2, 2, 3, 3, 3, 4, 4),
  wave = c(1, 2, 3, 1, 2, 1, 2, 3, 1, 3),
  depression = c(25, 20, 15, 30, 28, 22, 18, 16, 20, NA),
  anxiety = c(28, 25, 22, 32, NA, 24, 20, 18, 22, 19)
)

demographics <- tibble(
  participant = 1:4,
  age = c(22, 30, 25, 28),
  condition = c("Treatment", "Control", "Treatment", "Control")
)
```

Your tasks

1. Make implicit missing waves **explicit** using `complete()`
2. Join the demographics data
3. Count how many assessments each participant completed
4. Check if completion differs by treatment condition
5. Compute mean depression change (wave 1 to wave 3) for participants with complete data on those waves

Wrapping up

Decision tree for missing data

1. **How much is missing?**
   - < 5%: Usually safe to drop
   - 5-20%: Investigate patterns
   - 20%: Be very careful
2. **Why is it missing?**
   - Random: Less concerning
   - Systematic: Potentially biasing
3. **What's your plan?**
   - Drop complete cases?
   - Drop specific variables?
   - Keep as NA and use `na.rm`?
   - Fill (carefully)?
4. **Document everything!**

Key takeaways

1. **Missing data is normal** in psychology research
2. **Explicit vs implicit** missing — make implicit explicit with `complete()`
3. **Explore patterns** before deciding how to handle
4. **Complete case analysis** (dropping rows) is simple but can lose power
5. **Never make up data** — be transparent about missingness
6. **Document your decisions** — report what's missing and why
7. **Check for bias** — does missingness relate to key variables?

Functions cheat sheet

Function	Purpose
`is.na()`	Check if values are missing
`drop_na()`	Remove rows with NAs
`replace_na()`	Replace NAs with a value
`fill()`	Fill NAs with nearby values
`complete()`	Make implicit missing explicit

Function	Purpose
`na.omit()`	Remove rows with NAs (base R)
`naniar::vis_miss()`	Visualize missing data

Before next class

Read:

- R4DS Ch 11: Communication

Do:

- Submit Assignment 7
- Check your final project for missing data
- Draft your final project report

The one thing to remember

Missing data isn't a problem to solve — it's information about your study. Treat it that way.

See you Wednesday for storytelling with data!