

BioR — Class 3

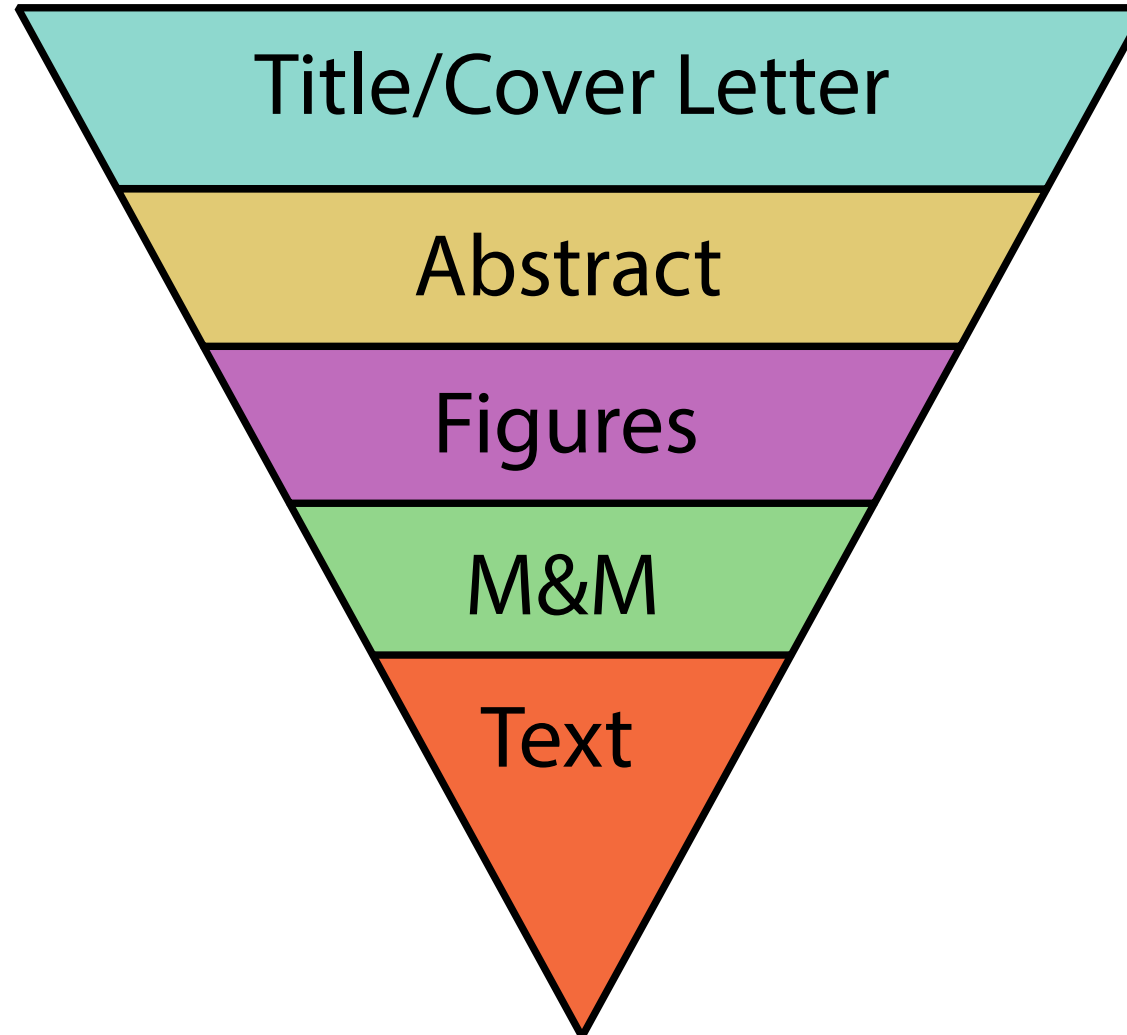
Data Visualization with ggplot2

Today you will learn

How to build clear figures from tidy tables using the grammar of graphics: data → aesthetics → geoms → scales → facets → theme.

You will also learn practical patterns for: mapping vs setting aesthetics, grouping time series, adding uncertainty, and exporting publication-ready figures.



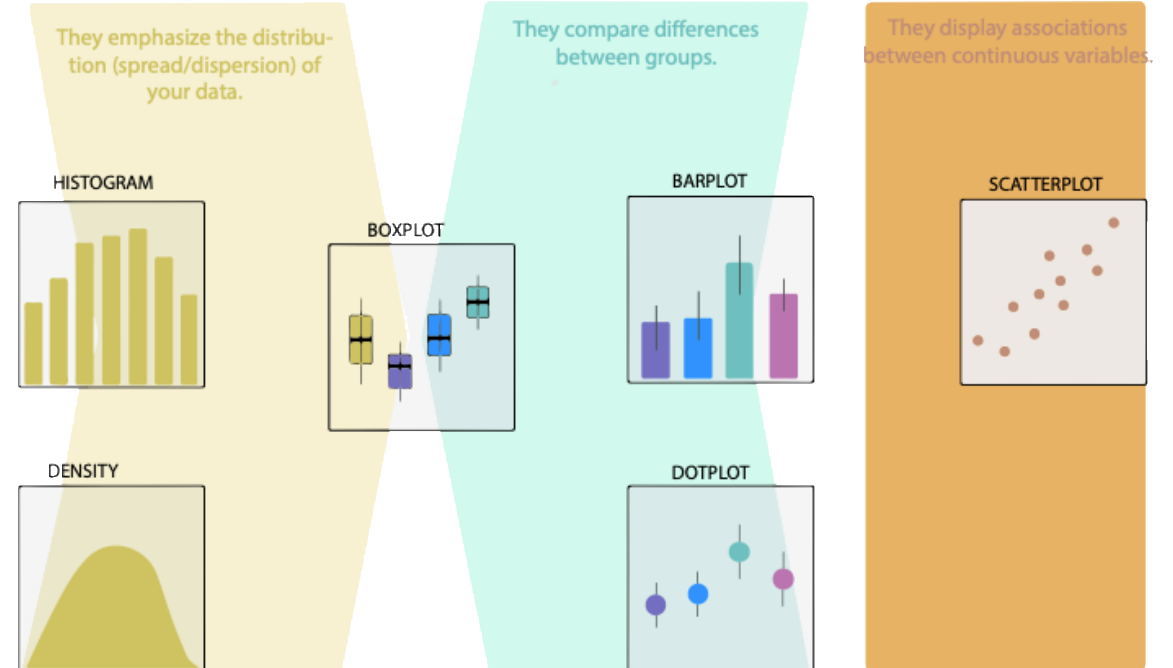


Agenda + learning outcomes

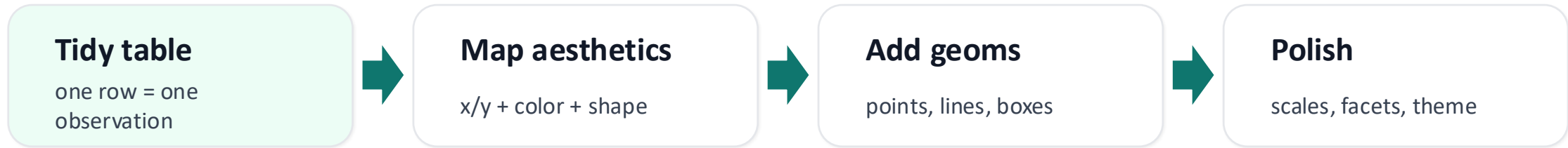
- The ggplot2 “grammar of graphics”: components & workflow
- Core syntax: `ggplot()` + `aes()` + `geom_*()`
- Aesthetics: mapping vs setting (color, size, alpha)
- Facets and grouping for biological datasets
- Themes, labels, and exporting figures (`ggsave`)
- Hands-on exercises with BioR dataset

By the end, you can...

- Choose an appropriate plot for a biological question
- Build plots systematically using layers
- Diagnose common ggplot mistakes (grouping, aesthetics, factors)
- Save figures with correct size and resolution



A practical pipeline



Key idea

ggplot2 is “declarative”: you describe how variables map to visual properties.

When your data are tidy, ggplot2 becomes fast and predictable: you can reuse the same code patterns across many biological datasets.

Think in components

- Data: the table you plot
- Aesthetics (aes): how variables map to x, y, color, shape, size
- Geoms: marks (points, lines, bars, boxes)
- Scales: control breaks, labels, transformations, palettes
- Facets: small multiples (split by groups)
- Theme: non-data styling (fonts, grids, background, legend)

```
ggplot(data) +  
  + aes(x, y, color, ...)  
  
  + geom_*()  
  
  + scale_*()  
  
  + facet_*()  
  
  + theme_*() + labs()
```

Same building blocks → many plot types

Think in components

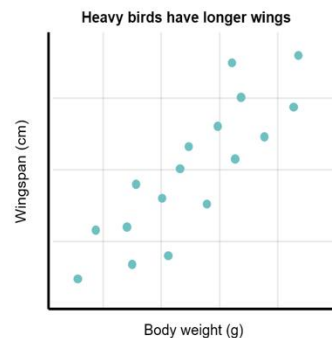
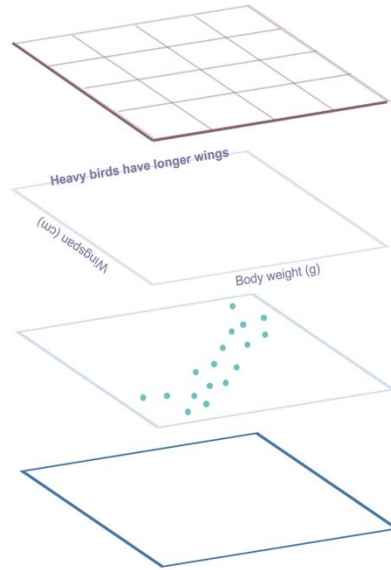
MAKING A GRAPH WITH GGPLOT2

Customise the look of your plot with themes
(pre-made or your own!);
`+ theme_bw()`

Add labels and titles:
`+ labs(x = "Body weight (g)", y = "Wingspan (cm)",
title = "Heavy birds have longer wings")`

Specify the type of graph and the variables to use:
`+ geom_point(aes(x = body.weight, y = wingspan))`

Plot the device containing your data:
`ggplot(data = birds)`



- Data: the table you plot
- Aesthetics (aes): how variables map to x, y, color, shape, size
- Geoms: marks (points, lines, bars, boxes)
- Scales: control breaks, labels, transformations, palettes
- Facets: small multiples (split by groups)
- Theme: non-data styling (fonts, grids, background, legend)

The ggplot template

Template: map variables → add layers → polish

```
library(ggplot2)

# 1) start with data + aes
p <- ggplot(dat, aes(x = temperature, y = biomass, color =
treatment)) +
  geom_point(alpha = 0.7) +
  geom_smooth(method = "lm", se = TRUE) +
  facet_wrap(~ site) +
  labs(x = "Temperature (°C)", y = "Biomass (g m-2)") +
  theme_minimal()

# 2) print or save
p
```

Rules of thumb

- Keep data tidy (one observation per row)
- Start simple, then add layers
- Use facets for comparisons
- Make labels explicit (units!)
- Save with consistent size + dpi

Common plot types for biology

Relationships

Scatter: `geom_point()`
Trend: `geom_smooth()`

Use when you ask: “How does Y change with X?”

Distributions

Box/violin: `geom_boxplot()`,
`geom_violin()`
Histogram: `geom_histogram()`

Use when you ask: “How are values spread?”

Time / gradients

Lines: `geom_line()`
Points: `geom_point()`

Use when you ask: “How does it change over time?”

Three “daily driver” patterns

```
# Scatter
ggplot(dat, aes(temperature, biomass, color = treatment)) +
  geom_point()

# Boxplot
ggplot(dat, aes(treatment, biomass)) +
  geom_boxplot()

# Time series (requires grouping!)
ggplot(dat, aes(day, biomass, color = treatment, group = rep)) +
  geom_line() + geom_point()
```

A common beginner mistake

Mapping (inside aes())

Use for variables: color = treatment, shape = species, size = abundance.
This creates a legend.

```
# MAPPING → legend  
ggplot(dat, aes(temperature, biomass, color =  
treatment)) +  
  geom_point(size = 2, alpha = 0.7)
```

Setting (outside aes())

Use for constants: color = "steelblue", alpha = 0.6.
This does NOT create a legend.

```
# SETTING → no legend  
ggplot(dat, aes(temperature, biomass)) +  
  geom_point(color = "steelblue", size = 2,  
alpha = 0.7)
```

Make comparisons readable

- Facets create small multiples (e.g., one panel per site)
- Grouping tells ggplot which points belong to the same line
- For repeated measures: group = rep (or sample_id)

Time series: grouping is NOT optional

```
# Facets: one panel per site
ggplot(dat, aes(day, biomass, color = treatment, group = rep)) +
  geom_line(alpha = 0.6) +
  geom_point(size = 1.5) +
  facet_wrap(~ site, nrow = 1) +
  labs(x = "Day", y = "Biomass") +
  theme_bw()
```

Show uncertainty responsibly

Regression / smoothing

`geom_smooth()` can fit models per group.
For linear relationships: `method = "lm"`.
Use `se = TRUE` to show uncertainty bands.

Group summaries

For mean \pm SE/CI by treatment/day, use `stat_summary()` or `summarise()` + `geom_errorbar()`.
Always state what the error bars represent.

```
# Linear model trend per treatment
ggplot(dat, aes(temperature, biomass, color = treatment)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = TRUE)

# Mean  $\pm$  SE by day
ggplot(dat, aes(day, biomass, color = treatment)) +
  stat_summary(fun = mean, geom = "line") +
  stat_summary(fun.data = mean_se, geom = "errorbar", width = 0.5)
```

Scales, units, and legends

- Scales control: breaks, labels, transformations (e.g., log10)
- Always label axes with units (e.g., °C, g m⁻²)
- If color encodes biology, choose a palette that stays readable when printed

```
ggplot(dat, aes(temperature, biomass, color = treatment)) +  
  geom_point(alpha = 0.7) +  
  scale_y_continuous(trans = "log10") +  
  labs(  
    title = "Biomass vs temperature",  
    x = "Temperature (°C)",  
    y = "Biomass (g m-2, log10)",  
    color = "Treatment"  
  ) +  
  theme_minimal()
```

Polish + save

Themes

Use a consistent theme across figures (e.g., `theme_bw()`, `theme_minimal()`).
Then modify details with `theme()`: text size, legend position, gridlines.

Export

Use `ggsave()` and always specify width/height and dpi for bitmap outputs.
Prefer PDF for vector graphics when possible.

```
p <- ggplot(dat, aes(temperature, biomass, color = treatment)) +  
  geom_point() +  
  theme_bw(base_size = 12) +  
  theme(legend.position = "top")  
  
# Save (bitmap)  
ggsave("outputs/fig_biomass_temp.png", p, width = 6, height = 4, dpi = 300)  
  
# Save (vector)  
ggsave("outputs/fig_biomass_temp.pdf", p, width = 6, height = 4)
```

Reproduce this end-to-end

```
library(tidyverse)
library(here)

# 1) Load tidy dataset
dat <- readr::read_csv(here("data", "BioR_Class3_dataset.csv"))

# 2) Basic plot
p <- ggplot(dat, aes(temperature, biomass, color = treatment)) +
  geom_point(alpha = 0.7) +
  geom_smooth(method = "lm", se = TRUE) +
  facet_wrap(~ site) +
  labs(x = "Temperature (°C)", y = "Biomass (g m-2)") +
  theme_minimal()

# 3) Save
ggsave(here("outputs", "fig_biomass_temp.png"), p, width = 7, height = 4, dpi = 300)
```

Common ggplot2 issues

Symptoms → likely cause

- “object not found” → typo in column name; check `names(dat)`
- Lines look wrong → missing group aesthetic (use `group = rep` or `sample_id`)
- No legend → you set color outside `aes()`
- Weird ordering → convert to factor and set levels
- “+” errors → put + at end of the line, not the start

```
# Check columns
names(dat)

# Fix ordering
dat <- dat %>%
  mutate(treatment = factor(treatment, levels =
c("CTRL", "HEAT")))

# Grouping for lines
ggplot(dat, aes(day, biomass, group = rep,
color = treatment)) +
  geom_line()

# Mapping vs setting
# aes(color = treatment) vs color =
"steelblue"
```

Common ggplot2 Add-on Pack (Biologists' favorites)

Use these when your plots need to be paper-ready fast:

ggrepel: readable point labels

(species/sites/samples)

patchwork: combine plots into multi-panel figures
(A–D)

viridis: colorblind-friendly palettes
(discrete/continuous)

scales: pretty axis labels (units, %, scientific
notation)

ggbeeswarm: show replicates clearly (better than
bars)

broom: tidy model outputs for plotting slopes/CI

```
# ggrepel: labels without overlap
library(ggrepel)
geom_text_repel()

# patchwork: multi-panel figures
library(patchwork)
(p1 | p2) / p3

# viridis: safer colours
library(viridis)
scale_color_viridis_d()

# scales: nicer axes
library(scales)
label_number(accuracy = 0.1)

# export (manuscript-friendly)
ggsave("fig1.png", p, width=6, height=4,
dpi=300)
```

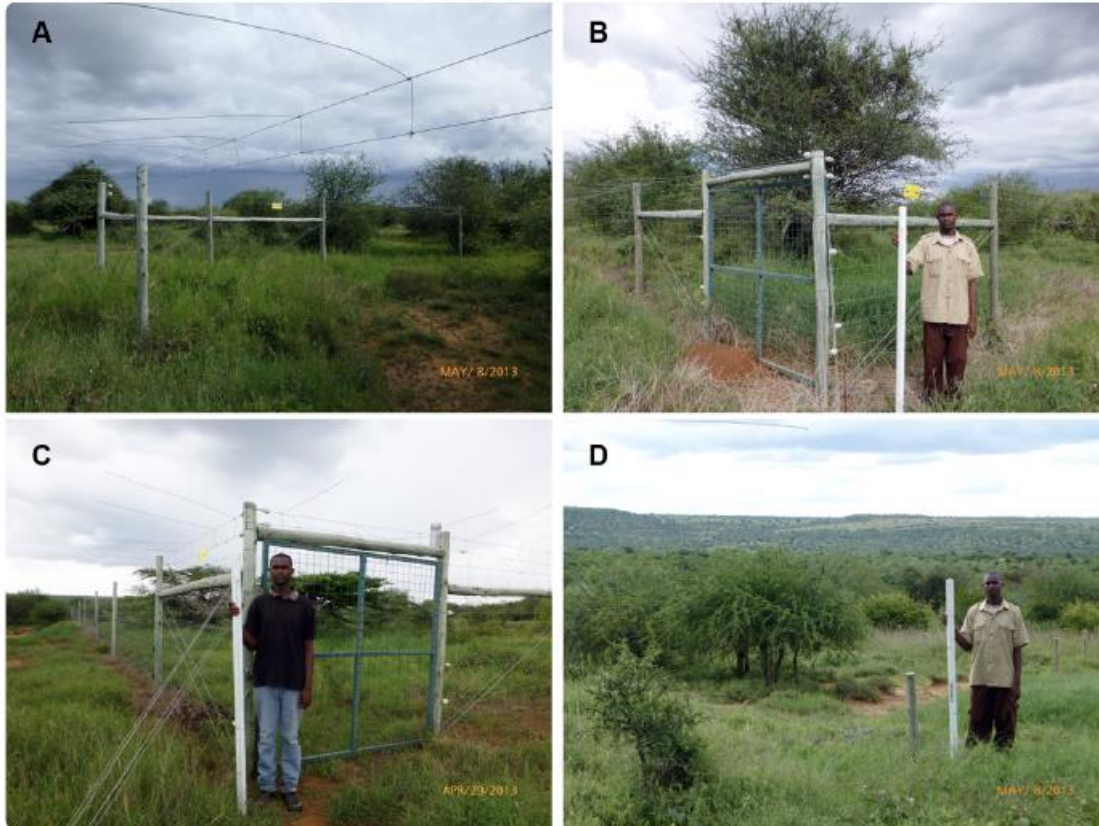
Before you submit a figure

- Does the figure answer one clear question?
- Are axes labeled with units?
- Is color encoding necessary and readable in grayscale?
- Is grouping correct (especially for repeated measures)?
- Are uncertainty and sample sizes communicated?
- Is the export size/resolution appropriate (dpi, dimensions)?

Next time

We'll connect plots to data manipulation workflows (summaries, joins, and reshaping) and build an end-to-end reproducible script.

Practice: Herbivores exclusion experiment



Experiment design: herbivore exclusions

Goal

Understand how herbivores influence vegetation by excluding different size classes.

Treatments (4 plot types)

- **Megaherbivore enclosure** (top-left): ~2 m high wires to exclude **elephants**
- **Mesoherbivore enclosure** (top-right): fence starts ~30 cm above ground to exclude **medium herbivores** (e.g., impala)
- **Full enclosure** (bottom-left): fence reaches the ground to exclude **all mammalian herbivores**
- **Control** (bottom-right): **no fencing**, natural grazing/browsing