4. Basic figures with ggplot2

Recreating figures from a published sepsis study

Contents

Setup . . . 1
Boxplots over time . . . 2
Anatomy of a ggplot . . . 2
Build the published version, one step at a time . . . 2
Reuse the figure for another variable . . . 4
Try it: pattern 15 (Figure 5a) . . . 4
Follow patients over time . . . 5
Average paths . . . 5
A trajectory plot . . . 5
Individual patients . . . 6
Explore patients with `ggplotly()` . . . 6
Heatmap: all 30 patterns at a glance . . . 7
Optional: six heatmaps side by side (Figure 2a–f) . . . 8
Keep a reference handy . . . 9

This lesson uses public figure data from Chenoweth et al. (2024), a study of blood gene expression in adults with sepsis in Kumasi, Ghana. We will make boxplots, patient trajectories, and a heatmap with `ggplot2`.

The data come from the authors' Zenodo record. The data notes describe our preparation. These are teaching recreations, so compare them with the published figures as you go.

Allow **35 minutes** for the main examples. Leave the interactive plot and six-panel heatmap for extra practice if time is short.

Setup

Each page starts a fresh R session. Run this cell first. In local R, install `ggplot2`, `plotly`, and `patchwork` once with `install.packages(c("ggplot2", "plotly", "patchwork"))`.

```
library(ggplot2)

# One row is a blood sample; patient_id links repeat samples.
samples <- read.csv("data/ghana_sepsis_samples.csv", na.strings = c("", "NA"))
heat <- read.csv("data/ghana_sepsis_heatmaps.csv", na.strings = c("", "NA"))

samples$outcome <- factor(samples$outcome,
                          levels = c("Healthy", "Survived", "Died"))
# Keep visits in time order.
samples$visit <- factor(samples$visit,
                        levels = c("H", "0h", "6h", "24h", "48h", "72h",
                                   "28d", "6m", "12m"))
```

```r
outcome_colours <- c(Healthy = "#E3B505", Survived = "#CDBB98", Died = "#C8102E")

table(samples$visit, samples$outcome)
```

`samples` has one row per **blood sample**, not per person. Healthy donors have one sample (`H`); patients have up to eight, linked by `patient_id`. Visits are measured in hours (`h`), days (`d`), and months (`m`) after enrolment. The factor levels keep them in time order instead of alphabetical order.

`PC1` and `PC2` summarize gene expression. `P1` to `P30` are CoGAPS pattern values between 0 and 1, showing how strongly each gene-expression pattern appears in a sample. They are analysis results, not laboratory measurements. `heat` contains the published heatmap values.

Boxplots over time

A boxplot summarizes a numeric variable within groups. Here the x-axis is visit, the y-axis is PC1, and the fill distinguishes outcomes.

```r
ggplot(samples, aes(x = visit, y = PC1, fill = outcome)) +
  geom_boxplot()
```

The line is the median and the box spans the middle 50% of samples.

Question: why is there only one box at `H`, and only survivor boxes after `72h`? Check the counts printed in Setup.

Anatomy of a ggplot

`ggplot()` chooses the data, and `aes()` maps columns to axes and other visual properties. A **geom** is the drawing layer: boxes here, or points and lines later.

Read this example as a guide. We build it one piece at a time in the next section.

```r
ggplot(samples, aes(x = visit, y = PC1, fill = outcome)) + # Data and mappings
  geom_boxplot() +                                         # Geom: draw boxes
  scale_fill_manual(values = outcome_colours) +            # Scale: choose colours
  labs(x = "Visit", y = "PC1 score") +                     # Labels: name the axes
  theme_classic()                                          # Theme: set appearance
```

`ggplot()` plus `geom_boxplot()` already makes the simple boxplot above. Colours, labels, and appearance have defaults; the last three lines customize them. Each `+` adds a plot component. The `|>` in the table pipeline passes a result to the next function instead. Put `+` at the end of each line except the last.

Inside `aes()`, a property varies with a column: `fill = outcome` assigns colours by outcome. Outside `aes()`, it is a fixed setting: `geom_boxplot(colour = "grey25")` gives every box a grey outline. Column names inside `aes()` are not quoted; fixed colour names are.

Question: which line would you change to rename the y-axis without changing the values plotted?

Build the published version, one step at a time

We will bring the plot closer to Figure 1f. Run each step and look at the result before adding the next.

1. Set the whiskers and box widths.

```r
# Save the plot so the next steps can add to it.
figure <- ggplot(samples, aes(x = visit, y = PC1, fill = outcome)) +
  geom_boxplot(coef = Inf, width = 0.7,
               position = position_dodge2(preserve = "single"))
```

```
figure
```

`coef = Inf` puts whiskers at the minimum and maximum, as in the paper. Try changing `width` from `0.7` to `0.4`.

2. Set the colours and legend labels.

```
# Reuse the outcome colours from Setup.
figure <- figure +
  scale_fill_manual(
    values = outcome_colours,
    breaks = c("Healthy", "Survived", "Died"),
    labels = c("Healthy donor", "Survived 28 days", "Died by 28 days")
  )

figure
```

Change one colour in `outcome_colours`, then rerun this step.

3. Add labels and a source credit.

```
# Give the axes, legend, and source a readable label.
figure <- figure +
  labs(
    title = "Later samples from survivors resemble healthy donors",
    subtitle = "Ghana sepsis cohort | Principal component 1 by visit and 28-day outcome",
    x = "Visit (H = healthy donor; hours, days, or months after enrolment)",
    y = "PC1 score", fill = NULL,
    caption = paste(
      "Source: Chenoweth et al. (2024), Nat Commun; source data on Zenodo (CC BY 4.0).",
      "Whiskers show the minimum and maximum. Each visit includes only the samples
       ↪ available.",
      sep = "\n"
    )
  )

figure
```

The caption includes the source credit; `"\n"` starts a new line.

4. Set the appearance.

```
# Apply the complete theme before making individual changes.
figure <- figure +
  theme_classic(base_size = 12) +
  theme(
    legend.position = "bottom",
    plot.title = element_text(face = "bold"),
    plot.caption = element_text(hjust = 0, colour = "#52616B")
  )

figure
```

`theme_classic()` sets the overall style. `theme()` changes individual parts. Keep that order, or the complete theme replaces your adjustments.

! Compare the samples available

Healthy donors were younger than patients (median age 26 versus 47 years). Patients who died by day 28 have no samples after 72 hours. Later survivor visits include only 41 samples at 28 days, 25 at 6 months, and 14 at 12 months.
The boxes therefore compare changing groups as well as times. They do not show that every patient recovered or establish what caused the differences.

Reuse the figure for another variable

An additional `aes()` can replace one mapping. Figure 2g uses pattern 23, which the authors link to septic shock.

```
# Change the y variable while keeping the other settings.
figure +
  aes(y = P23) +
  labs(
    title = "Pattern 23 is higher in patients who died",
    subtitle = "Ghana sepsis cohort | Pattern 23, which the authors link to septic
     ↪  shock",
    y = "Pattern 23 value"
  )
```

Pattern 23 is higher in patients who died at each visit through 72 hours. The paper reports an association with 28-day mortality from enrolment; this does not establish causation.

Try it: pattern 15 (Figure 5a)

Allow five minutes. Fill the blanks to plot pattern 15 by visit and outcome, with whiskers extending to the minimum and maximum.

```
ggplot(samples, aes(x = visit, y = ______, fill = ______)) +
  geom_boxplot(coef = ______, position = position_dodge2(preserve = "single")) +
  scale_fill_manual(values = outcome_colours) +
  labs(x = "Visit", y = "Pattern 15 value", fill = "28-day outcome") +
  theme_classic()
```

i Hint

Use `P15` for y and `outcome` for fill. `Inf` extends the whiskers.

Solution.

Solution

```
ggplot(samples, aes(x = visit, y = P15, fill = outcome)) +
  geom_boxplot(coef = Inf, position = position_dodge2(preserve = "single")) +
  scale_fill_manual(values = outcome_colours) +
  labs(x = "Visit", y = "Pattern 15 value", fill = "28-day outcome") +
  theme_classic()
```

Pattern 15 is higher in patients who died at visits through 72 hours. The authors link it to developing neutrophils also seen in severe COVID-19. Compare Figure 5a, or try `figure + aes(y = P15)`.

Follow patients over time

A trajectory joins observations in time order. Figure 6 compares pattern 23, linked to septic shock, with pattern 4, linked to healthy myeloid cells.

Average paths

`aggregate()` calculates each outcome group's mean at each visit.

```
patients <- subset(samples, outcome != "Healthy")
# geom_path() joins rows in their current order.
patients <- patients[order(patients$patient_id, patients$visit), ]

# Use the samples available in each group and visit.
mean_paths <- aggregate(cbind(P23, P4) ~ outcome + visit,
                        data = patients, FUN = mean)
mean_paths <- mean_paths[order(mean_paths$outcome, mean_paths$visit), ]

mean_paths
```

`geom_path()` follows row order. `geom_line()` follows x order, which would scramble a path through time. Neither the group mean nor the later visits represent the same number of patients throughout.

A trajectory plot

```
visit_sizes <- c("H" = 2.6, "0h" = 2.8, "6h" = 2.5, "24h" = 2.2, "48h" = 1.9,
                 "72h" = 1.6, "28d" = 1.3, "6m" = 1.0, "12m" = 0.8)
line_colours <- c(Survived = "#6B5530", Died = "#7A0019")

# Points show samples; thicker paths show group means.
trajectory <- ggplot(samples, aes(x = P23, y = P4)) +
  geom_point(aes(fill = outcome, size = visit), shape = 21,
             colour = "grey25", stroke = 0.2, alpha = 0.8) +
  geom_path(data = mean_paths, aes(colour = outcome), linewidth = 1.3,
            arrow = arrow(length = unit(0.3, "cm"), type = "closed")) +
  scale_fill_manual(values = outcome_colours) +
  scale_colour_manual(values = line_colours, guide = "none") +
  scale_size_manual(values = visit_sizes) +
  labs(
    x = "Pattern 23 value (linked to septic shock)",
    y = "Pattern 4 value (linked to health)",
    fill = "28-day outcome", size = "Visit",
    caption = "Lines join each group's mean at each visit; arrows point to the last
     ↪ visit."
  ) +
  theme_classic(base_size = 12)

trajectory
```

The plot adds two layers: individual samples, then group means with arrows toward the last visit. `data = mean_paths` supplies a different dataset for that layer while retaining the x and y mappings.

As in the anatomy example, fill varies by outcome inside `aes()`, while the grey outline is fixed outside it.

Try changing `shape = 21` to 22. Then change `stroke = 0.2` to 1.

The survivor mean moves towards the healthy donors. Averages hide individual movement, and later survivor means come from fewer people.

Individual patients

`group = patient_id` gives each patient a path. `facet_wrap()` puts outcomes in separate panels.

```
# Give each patient a separate path.
paths <- ggplot(patients, aes(x = P23, y = P4, group = patient_id,
                              colour = outcome)) +
  geom_path(alpha = 0.4) +
  geom_point(size = 0.8) +
  facet_wrap(~ outcome) +
  scale_colour_manual(values = line_colours, guide = "none") +
  labs(x = "Pattern 23 value", y = "Pattern 4 value") +
  theme_classic(base_size = 12)

paths
```

The lines are tangled because there are more than 100 patients. Can you follow one patient from the first sample to the last?

Explore patients with `ggplotly()`

Optional: click through patient paths

`ggplotly()` converts a ggplot to an interactive HTML plot. `highlight_key()` links visits from one patient; `text` supplies hover labels. The package's message about masked functions is expected.

```
library(plotly)

# Link all visits from the same patient.
patient_key <- highlight_key(patients, ~patient_id)

clickable <- ggplot(patient_key, aes(
    x = P23, y = P4, group = patient_id, colour = outcome,
    text = paste0("Patient ", patient_id,
                  "<br>Visit: ", visit,
                  "<br>Pattern 23: ", round(P23, 2),
                  "<br>Pattern 4: ", round(P4, 2))
  )) +
  geom_path(alpha = 0.3) +
  geom_point(size = 1) +
  facet_wrap(~ outcome) +
  scale_colour_manual(values = line_colours) +
  labs(x = "Pattern 23 value", y = "Pattern 4 value", colour = "28-day outcome") +
  theme_classic(base_size = 12)

interactive_paths <- ggplotly(clickable, tooltip = "text")
highlight(interactive_paths, on = "plotly_click", color = "black")
```

Hover for the patient and visit. Click a point to highlight the patient's path; double-click empty space to clear it. Find **S67615**, a survivor shown in Figure 6b, and follow its path towards the healthy area by 28 days. Click a legend entry to hide a group, or add `selectize = TRUE` to `highlight()` to search by patient ID.

Arrows do not survive the ggplotly conversion, so use the hover visit labels. Keep `paths` for static exports and `interactive_paths` for interactive HTML. A PNG, SVG, PDF, Word picture, or PowerPoint picture cannot keep these controls. In local R the widget opens in the Viewer or browser; the PDF lesson prints its code. See the Plotly linking guide.

Heatmap: all 30 patterns at a glance

A heatmap uses colour to show values in a grid. Figure 2a has one row per pattern and one column per group and visit. Each z-score compares a group's mean with that pattern's mean across the 14 columns. Compare colours within a row: each row is standardized separately.

```
pattern_order <- c(3, 17, 4, 12, 27, 25, 10, 29, 21, 19, 6, 18, 11, 2, 22,
                   20, 14, 26, 8, 13, 16, 7, 1, 23, 15, 5, 9, 30, 28, 24)
# Reverse the levels so the first pattern appears at the top.
heat$pattern <- factor(heat$pattern, levels = rev(pattern_order))

zscores <- subset(heat, panel == "CoGAPS z-score")
zscores$group <- factor(zscores$group, levels = c("Healthy", "Survived", "Died"))
zscores$column <- factor(zscores$column, levels = levels(samples$visit))

pattern_heatmap <- ggplot(zscores, aes(x = column, y = pattern, fill = value)) +
  geom_tile(colour = "white", linewidth = 0.3) +
  facet_grid(. ~ group, scales = "free_x", space = "free_x") +
  scale_fill_gradientn(colours = c("#FFFFCC", "#FEB24C", "#F03B20", "#800026")) +
  labs(x = NULL, y = "Pattern", fill = "z-score") +
  theme_minimal(base_size = 11) +
  theme(panel.grid = element_blank())

pattern_heatmap
```

The factor levels reproduce the published row order, with separate panels for Healthy, Survived, and Died.

Find patterns 23 and 26. Pattern 23 is darker in the Died columns throughout. Pattern 26 is dark early in both patient groups, then fades in survivors. The values are the authors' group means standardized within each pattern; you could calculate them from `samples` using group means and `scale()`.

Optional: six heatmaps side by side (Figure 2a–f)

Extension: separate colour scales

Allow another ten minutes. Each panel uses the same pattern rows, but a separate fill scale. The function below builds one panel; `patchwork` joins them together.

```
library(patchwork)

# Give each panel its own fill scale before joining the plots.
heat_panel <- function(panel_name, high_colour, fill_label) {
  panel_data <- subset(heat, panel == panel_name)
  panel_data$column <- factor(panel_data$column,
                              levels = unique(panel_data$column))
  ggplot(panel_data, aes(x = column, y = pattern, fill = value)) +
    geom_tile(colour = "white", linewidth = 0.3) +
    scale_fill_gradient(low = "white", high = high_colour) +
    labs(title = panel_name, x = NULL, y = NULL, fill = fill_label) +
    theme_minimal(base_size = 9) +
    theme(
      panel.grid = element_blank(),
      axis.text.y = element_blank(),
      axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5),
      legend.position = "bottom"
    )
}

six_panels <- pattern_heatmap +
  labs(title = "CoGAPS z-score") +
  theme(legend.position = "bottom") +
  heat_panel("AUROC", "#08306B", "AUROC") +
  heat_panel("CIBERSORT", "#00441B", "r") +
  heat_panel("Hematology", "#00441B", "r") +
  heat_panel("Chemistry and vitals", "#7F2704", "r") +
  heat_panel("Proteins", "#67000D", "r") +
  plot_layout(widths = c(14, 5, 6, 7, 7, 12)) +
  plot_annotation(
    caption = "Source: Chenoweth et al. (2024), Nat Commun; source data on Zenodo
    ↪  (CC BY 4.0)."
  )

six_panels
```

The panels share the same pattern rows.
AUROC measures separation of survivors from non-survivors at the first five visits: 0.5 is chance and 1 is perfect. The other panels show Spearman correlations with estimated cell proportions, blood counts, chemistry and vital signs, or plasma proteins.
The sequential correlation scales match the paper. For a scale centred on zero, try `scale_fill_gradient2(low = "#2166AC", mid = "white", high = "#B2182B")`. Pattern 23 has the highest AUROC at the first four visits. For glucose, sodium, and blood urea nitrogen (BUN), it is the darkest row in the chemistry panel; for the Glasgow Coma Scale (GCS), it is the lightest. These are associations, not causes.

Keep a reference handy

The ggplot2 cheatsheet summarizes geoms, scales, facets, labels, and themes on two pages. More are available at Posit's cheatsheet page.

← 3. Building a Table 1 5. Exporting to Word and PowerPoint →