

MODULE 3: PROGRAMMING FOR BIOLOGICAL DATA

# Session 4: Visualizing Clinical Data

Building Dashboards and QC Charts with ggplot2

**Instructor: Dr. Gu Haogao**

January 14, 2026

# The "Black Box" Problem

---

## ■ The Table

10,000 rows of numbers.

Computers love it.

Humans cannot spot errors.

## ↷ The Plot

Visual patterns.

Humans love it.

**"The outlier screams at you."**

# Introducing ggplot2



## The "Grammar of Graphics"

We build a plot layer by layer, just like Photoshop or a sandwich.

**Reproducible:** Code-based (not point-and-click).

**Scalable:** Handles millions of points.

**Standardized:** Consistent look for every report.

**Data visualization with ggplot2 :: CHEATSHEET**

**Basics**  
ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data set**, a **coordinate system**, and **geoms**—visual marks that represent data points.

**Geoms** Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

**GRAPHICAL PRIMITIVES**  
a <- ggplot(economics, aes(date, unemployed))  
b <- ggplot(seals, aes(x, long, y=lat))

a + geom\_blank() and a + expand\_limits()  
Ensure limits include values across all plots.

b + geom\_curve(aes(yend = lat + 1, xend = long + 1), curvature = 1) -> x, y, curve, angle, color, curvature, linetype, size

a + geom\_path(linewidth = "butt", linejoin = "round", linemrite = 1) -> x, y, alpha, color, group, linetype, size

b + geom\_rect(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1)) -> x, y, alpha, color, fill, group, subgroup, linetype, size

b + geom\_ribbon(aes(ymin = unemployed - 900, ymax = unemployed + 900)) -> x, y, alpha, color, fill, group, linetype, size

a + geom\_ribbon(aes(ymin = unemployed - 900, ymax = unemployed + 900)) -> x, y, alpha, color, fill, group, linetype, size

**TWO VARIABLES both continuous**  
e <- ggplot(mpg, aes(cty, hwy))

e + geom\_label(aes(label = cty, nudge\_x = 1, nudge\_y = 1)) -> x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom\_point() -> x, y, alpha, color, fill, shape, size, stroke

e + geom\_quantile() -> x, y, alpha, color, group, linetype, size, weight

e + geom\_rug(sides = "bl") -> x, y, alpha, color, linetype, size

e + geom\_smooth(method = lm) -> x, y, alpha, color, fill, group, linetype, size, weight

e + geom\_text(aes(label = cty, nudge\_x = 1, nudge\_y = 1)) -> x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

**continuous bivariate distribution**  
h <- ggplot(diamonds, aes(carat, price))

h + geom\_binned(binwidth = c(0.25, 500)) -> x, y, alpha, color, fill, linewidth, size, weight

h + geom\_hex2d() -> x, y, alpha, color, group, linetype, size

h + geom\_hex() -> x, y, alpha, color, fill, size

**continuous function**  
i <- ggplot(economics, aes(date, unemployed))

i + geom\_area() -> x, y, alpha, color, fill, linetype, size

i + geom\_line() -> x, y, alpha, color, group, linetype, size

i + geom\_step(direction = "hv") -> x, y, alpha, color, group, linetype, size

**visualizing error**  
df <- data.frame(grp = c("A", "B"), fit = 4.5, se = 1.2)  
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))

j + geom\_crossbar(fatten = 2) -> x, y, max, ymin, alpha, color, fill, group, linetype, size

j + geom\_boxplot() -> x, y, lower, middle, upper, max, ymin, alpha, color, fill, group, linetype, shape, size, weight

j + geom\_errorbar() -> x, y, min, max, alpha, color, fill, group, linetype, size, width  
Also geom\_errorbarh().

j + geom\_linerange() -> x, y, min, max, alpha, color, fill, group, linetype, size

j + geom\_pointrange() -> x, y, min, max, alpha, color, fill, group, linetype, shape, size

**maps**  
Draw the appropriate geometric object depending on the specific features present in the data. aes() arguments:  
map\_id, alpha, color, fill, linetype, linewidth.

nc <- sf::st\_read(system.file("shape/nc.shp", package = "sf"))  
ggplot(nc) +  
geom\_sf(aes(fill = AREA))

**THREE VARIABLES**  
seals\$z <- with(seals, sqrt(delta\_long^2 + delta\_lat^2)); i <- ggplot(seals, aes(long, lat))

i + geom\_contour(aes(z = z)) -> x, y, z, alpha, color, group, linetype, size, weight

i + geom\_bar() -> x, y, alpha, color, fill, linetype, size, weight

i + geom\_contour\_filled(aes(fill = z)) -> x, y, alpha, color, fill, group, linetype, size, subgroup

i + geom\_tile(aes(fill = z)) -> x, y, alpha, color, fill, linetype, size, width

**posit**  
CC BY SA Posit Software, PBC • info@posit.co • posit.co • Learn more at ggplot2.tidyverse.org • HTML cheatsheets at posit.tidytuesday.org • ggplot2 3.5.1 • Updated: 2024-05

# The 3 Pillars of a Plot

---



## 1. Data

The raw dataframe

`df`



## 2. Aesthetics

Mapping data to visuals

`aes(x=Date, y=Result)`



## 3. Geometry

The shape of the data

`geom_point()`

# The Syntax Template

---

```
ggplot(data = df, aes(x = col1, y = col2)) +  
  geom_point()
```

**The Golden Rule:** The + sign connects the layers. It MUST go at the end of the line!

# Step 0: Load the Library

---

ggplot2 is not in "Base R". We must load it from the shelf.

```
# install.packages("ggplot2")  <-- Run once only  
library(ggplot2)
```

# Geom 1: The Histogram (Distribution)

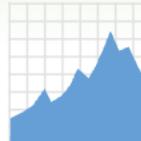
**Goal:** Check if patient data follows a Normal Distribution.

```
ggplot(df, aes(x = Result)) +  
  geom_histogram()
```

*Is it skewed? Are there two peaks (batch effect)?*

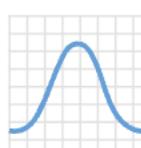
## ONE VARIABLE continuous

```
c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)
```



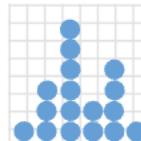
**c + geom\_area(stat = "bin")**

x, y, alpha, color, fill, linetype, size



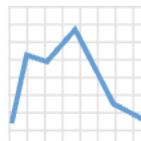
**c + geom\_density(kernel = "gaussian")**

x, y, alpha, color, fill, group, linetype, size, weight



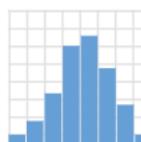
**c + geom\_dotplot()**

x, y, alpha, color, fill



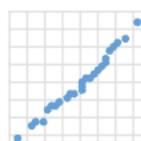
**c + geom\_freqpoly()**

x, y, alpha, color, group, linetype, size



**c + geom\_histogram(binwidth = 5)**

x, y, alpha, color, fill, linetype, size, weight



**c2 + geom\_qq(aes(sample = hwy))**

x, y, alpha, color, fill, linetype, size, weight

# Customizing the Histogram

---

Make it look scientific by adding color and binwidth.

```
ggplot(df, aes(x = Result)) +  
  geom_histogram(binwidth = 0.5,  
                 fill = "navy",  
                 color = "white")
```

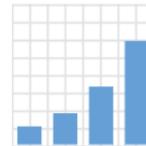
# Geom 2: The Boxplot (Comparisons)

**Goal:** Compare results between groups (Male vs Female, Machine A vs B).

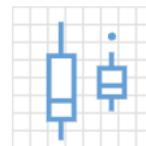
```
ggplot(df, aes(x = Gender,  
                y = Result,  
                fill = Gender)) +  
  geom_boxplot()
```

Shows Median, IQR, and Outliers.

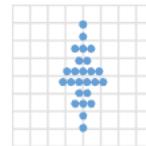
**one discrete, one continuous**  
`f <- ggplot(mpg, aes(class, hwy))`



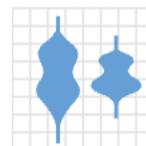
**f + geom\_col()**  
x, y, alpha, color, fill, group, linetype, size



**f + geom\_boxplot()**  
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight



**f + geom\_dotplot(binaxis = "y", stackdir = "center")**  
x, y, alpha, color, fill, group



**f + geom\_violin(scale = "area")**  
x, y, alpha, color, fill, group, linetype, size, weight

# Geom 3: The Scatter Plot (Correlation)

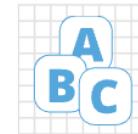
**Goal:** Comparing two continuous variables.

```
ggplot(df, aes(x = Age,  
                y = Viral_Load)) +  
  geom_point()
```

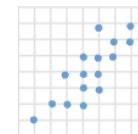
*Are older patients getting sicker?*

**TWO VARIABLES  
both continuous**

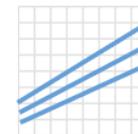
```
e <- ggplot(mpg, aes(cty, hwy))
```



**e + geom\_label(aes(label = cty)) -** x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust



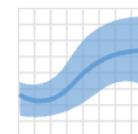
**e + geom\_point()**  
x, y, alpha, color, fill, shape, size, stroke



**e + geom\_quantile()**  
x, y, alpha, color, group, linetype, size, weight



**e + geom\_rug(sides = "bl")**  
x, y, alpha, color, linetype, size



**e + geom\_smooth(method = lm)**  
x, y, alpha, color, fill, group, linetype, size, weight



**e + geom\_text(aes(label = cty)) -** x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

# Geom 4: The Line Plot (Time Series)

---

**Goal:** The QC Levey-Jennings Chart.

```
ggplot(qc_data, aes(x = Date, y = Result)) +  
  geom_line() +  
  geom_point()
```

Tracking values over time to spot trends or drifts.

# Layering: Adding Thresholds

---

A chart needs context. We draw the "Limit Line".

```
ggplot(df, aes(x = Date, y = Result)) +  
  geom_line() +  
  geom_hline(yintercept = 6.0,  
             color = "red",  
             linetype = "dashed")
```

Any point above the red line is instantly recognizable as critical.

# Mapping Color to Data

---

Don't just color it blue. Color it by *meaning*.

```
# Inside aes()
ggplot(df, aes(x = Date,
                y = Result,
                color = Instrument)) +
  geom_line()
```

R automatically creates a legend and assigns colors to each Instrument.

# Faceting (Small Multiples)

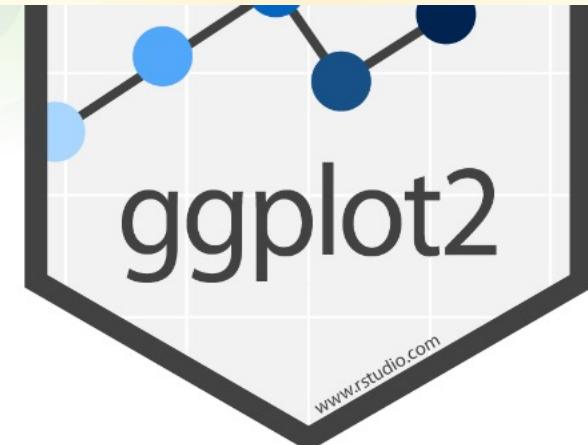
**Problem:** 10 instruments on one chart is messy.

**Solution:** Faceting.

```
ggplot(df, aes(x=Date, y=Res)) +  
  geom_line() +  
  facet_wrap(~ Instrument)
```

## Faceting

Facets divide a plot into subplots based on the values of one or more discrete variables.



```
t <- ggplot(mpg, aes(cty, hwy)) + geom_point()
```

||||| **t + facet\_grid(. ~ fl)**

Facet into columns based on fl.

===== **t + facet\_grid(year ~ .)**

Facet into rows based on year.

===== **t + facet\_grid(year ~ fl)**

Facet into both rows and columns.

===== **t + facet\_wrap(~ fl)**

Wrap facets into a rectangular layout.

# Themes (Publication Ready)

---

Remove the default grey background for a professional look.

```
plot + theme_bw()  
  
# OR  
  
plot + theme_minimal()
```

theme\_bw() is best for clinical reports (saves printer ink).

# Saving Plots

---

Don't take screenshots. Export high-res images programmatically.

```
ggsave("Monthly_QC_Report.png",
       width = 8,
       height = 5)
```

# Practice: The Levey-Jennings Plan

---

## 1. Base

Plot Points & Lines

## 2. Context

Add Mean Line

## 3. Limits

Add +/- 2SD Lines

# Summary of Geometries

---

- ⚠ geom\_point(): Scatter Plot
- ⚠ geom\_line(): Time Series
- ⚠ geom\_bar(): Counts
- ⚠ geom\_boxplot(): Stats Comparison
- ⚠ geom\_histogram(): Distribution

# Summary of Aesthetics

---

- ⚠ x, y: Position coordinates
- ⚠ color: Line/Point color (outline)
- ⚠ fill: Bar/Box color (inside)
- ⚠ size: Size of points
- ⚠ alpha: Transparency (0=invisible, 1=solid)

# Tutorial 4: The QC Trend Analysis

---

Goal: Build a Monthly Levey-Jennings Chart to compare two instruments.

- ⬆️ 1. Load Data
- ⬇️ 2. Visualize Trend
- ➡️ 3. Add QC Limits

# Step 1: Load Data

---

```
library(ggplot2) url <-  
"https://raw.githubusercontent.com/.../daily_qc_trend.csv" qc_data <-  
read.csv(url) # Check head(qc_data)
```

# Step 2: Basic Time Series

---

Ensure Date is recognized, then plot.

```
qc_data$Date <- as.Date(qc_data$Date)

ggplot(qc_data, aes(x = Date,
                     y = Result,
                     color = Instrument)) +
  geom_line() +
  geom_point()
```

# Step 3: Interpreting the Mess

---

The lines overlap. It is hard to see which machine is drifting.

**Solution:**

**Faceting.**

# Step 4: Faceting

---

Split into two panels for clarity.

```
ggplot(qc_data, aes(x = Date,  
                     y = Result,  
                     color = Instrument)) +  
  geom_line() +  
  geom_point() +  
  facet_wrap(~ Instrument)
```

# Step 5: Adding QC Limits

---

**Known Values:** Mean = 5.0, SD = 0.2

- ⚠️ Mean: 5.0 (Black Line)
- ⚠️ +2SD: 5.4 (Red Dashed)
- ⚠️ -2SD: 4.6 (Red Dashed)

# Step 6: Code for Limits

---

```
... + geom_hline(yintercept = 5.0, color = "black") + geom_hline(yintercept = 5.4, color = "red",  
linetype = "dashed") + geom_hline(yintercept = 4.6, color = "red", linetype = "dashed")
```

# Step 7: Final Polish

---

Add title and clean theme.

```
... +
theme_bw() +
labs(title = "Monthly Levey-Jennings Chart",
     y = "Glucose Level (mmol/L)",
     x = "Date")
```

# Analysis: The Diagnosis

---

## Machine B

Flat line.

Very precise.

**PASS**

## Machine A

Bouncing around.

Point on Jan 14 is 5.8 ( $> 3SD$ ).

**FAIL (Outlier)**

# Step 8: Export

---

Save your clinical report.

```
ggsave("QC_Dashboard_Jan2026.pdf",
       width = 10,
       height = 6)
```

[https://github.com/Koohoko/MSc\\_Module3\\_Programming\\_BioData\\_HKUSpacce\\_2026](https://github.com/Koohoko/MSc_Module3_Programming_BioData_HKUSpacce_2026)



## MSc Module 3: Programming for Biological Data (2026)

Instructor: Dr. Gu Haogao

Institution: SPH HKU / HKU SPACE Date: January 2026

### Course Overview

This repository contains the comprehensive preparation package for **Module 3**, tailored for Postgraduate Certificate in Bioinformatics for Medical Laboratory Technologists (MLTs). This plan integrates the "Code-as-Protocol" pedagogical approach.

### Repository Structure

- `lectures/` : Slides and R scripts used during the lecture.
- `tutorials/` : Interactive notebooks for hands-on practice.
- `data/` : Raw datasets used in exercises.
- `setup/` : Scripts to initialize the R environment.

### Quick Start

No software installation is required. We will use Google Colab for our "Dry Lab" sessions. Remember to **change runtime to R** while using colab (**Runtime -> Change runtime type -> R**).

- Session 1: Introduction to R (Part 1)
  - [Lecture Script](#)
  - [Tutorial Notebook](#): Open in Colab
- Session 2: Introduction to R (Part 2) - Data Frames & I/O
  - [Lecture Script](#)
  - [Tutorial Notebook](#): Open in Colab