Module 5: Visualization for health data

Data analysis with Python for health specialists

Yaé Ulrich Gaba

2026

AIRINA Labs

Principles of health data visualization

Before writing any plotting code, ask:

1. **What is the message?** “Malaria incidence is declining” — not “here is data.”
2. **Who is the audience?** Clinicians need different plots than policymakers.
3. **What comparison matters?** Between groups? Over time? Against a threshold?

Rules for every health figure:

- Label axes with units (“Systolic BP (mmHg)”, not “BP”)
- Use colorblind-friendly palettes
- No 3D charts, no pie charts, no dual y-axes
- Include sample sizes in titles or annotations

Setup: matplotlib and seaborn

```
import matplotlib.pyplot as plt
import seaborn as sns

# Clean style for all plots
sns.set_theme(style="whitegrid", font_scale=1.1)
plt.rcParams["figure.figsize"] = (8, 5)
plt.rcParams["figure.dpi"] = 100

# Colorblind-friendly palette (Bang Wong, 2011)
cb_palette = ["#0072B2", "#D55E00", "#009E73",
              "#CC79A7", "#F0E442", "#56B4E9"]
sns.set_palette(cb_palette)
```

These six colors are distinguishable by people with the most common forms of color vision deficiency.

Histograms: distribution of a variable

```
fig, ax = plt.subplots(figsize=(10, 5))
ax.hist(all_glucose, bins=40, edgecolor="white", alpha=0.7)
ax.axvline(100, color="orange", linestyle="--",
           label="Normal threshold (100 mg/dL)")
ax.axvline(126, color="red", linestyle="--",
           label="Diabetic threshold (126 mg/dL)")
ax.set_xlabel("Fasting glucose (mg/dL)")
ax.set_ylabel("Number of patients")
ax.set_title("Distribution of fasting glucose (n=1,000)")
ax.legend()
```

Add clinical threshold lines to give immediate context.

Box plots: group comparisons

```
fig, ax = plt.subplots(figsize=(10, 6))
sns.boxplot(data=clinical, x="age_group", y="bmi",
            hue="sex", order=["18-39", "40-59", "60+"])
ax.axhline(25, color="orange", linestyle="--",
           label="Overweight")
ax.axhline(30, color="red", linestyle="--",
           label="Obese")
ax.set_ylabel("BMI (kg/m\u00b2)")
ax.set_title(f"BMI by sex and age group (n={n})")
ax.legend()
```

Limitation

Box plots hide bimodal distributions. Consider `sns.violinplot()` or overlay `sns.stripplot()` for small samples.

Bar charts: counts and rates

```
# Horizontal bar chart: 15 lowest life expectancy countries
bottom15 = recent.nsmallest(15, "lifeExp")

fig, ax = plt.subplots(figsize=(10, 7))
ax.barh(bottom15["country"], bottom15["lifeExp"],
        color="#D55E00")
ax.set_xlabel("Life expectancy (years)")
ax.set_title("15 countries with lowest life expectancy, 2007")
ax.invert_yaxis()
for i, v in enumerate(bottom15["lifeExp"]):
    ax.text(v + 0.5, i, f"{v:.1f}", va="center")
```

Use **horizontal bars** when you have many categories to avoid rotated labels.

Line plots: epidemic curves

```
fig, ax = plt.subplots(figsize=(12, 5))
ax.bar(epi["date"], epi["new_cases"], alpha=0.4,
       label="Daily cases")
ax.plot(epi["date"], epi["rolling_7d"], color="#D55E00",
        linewidth=2, label="7-day average")
ax.set_ylabel("New cases per day")
ax.set_title("Epidemic curve: daily new cases")
ax.legend()
```

Clinical context

The epi curve is the single most important visualization during an outbreak. Its shape reveals the transmission pattern: point-source (sharp peak), continuous (plateau), or person-to-person (waves).

Scatter plots: relationships

```
# The classic Gapminder bubble chart
fig, ax = plt.subplots(figsize=(10, 7))
for continent in recent["continent"].unique():
    sub = recent[recent["continent"] == continent]
    ax.scatter(sub["gdpPercap"], sub["lifeExp"],
               s=sub["pop"] / 1e6, alpha=0.6,
               label=continent)
ax.set_xscale("log")
ax.set_xlabel("GDP per capita (log scale, USD)")
ax.set_ylabel("Life expectancy (years)")
ax.set_title("Wealth vs Health, 2007")
ax.legend()
```

Add regression lines with `sns.regplot()` and report the Pearson *r* in the title.

Kaplan-Meier survival curves

```
from lifelines import KaplanMeierFitter

kmf = KaplanMeierFitter()
kmf.fit(treatment_time, event_observed=treatment_event,
        label="Treatment (n=100)")
kmf.plot_survival_function(ax=ax, ci_show=True)
```

Survival analysis tracks **time until an event** (death, relapse). KM curves handle censored observations — patients still alive at study end.

Install: `pip install lifelines`

Heatmaps: correlations and co-occurrence

```
corr = health.corr()
mask = np.triu(np.ones_like(corr, dtype=bool))

fig, ax = plt.subplots(figsize=(9, 8))
sns.heatmap(corr, mask=mask, annot=True, fmt=".2f",
            cmap="RdBu_r", center=0, vmin=-1, vmax=1,
            square=True, ax=ax)
ax.set_title("Correlation matrix of health indicators")
```

Use `"RdBu_r"` (diverging) for correlations. Use `"YlOrRd"` (sequential) for disease burden.

Multi-panel figures: the health dashboard

```
fig, axes = plt.subplots(2, 2, figsize=(14, 10))
# Panel A: Box plot by continent
# Panel B: GDP vs Life Expectancy scatter
# Panel C: Life expectancy trends over time
# Panel D: Bottom 10 countries bar chart
plt.tight_layout()
plt.savefig("dashboard.png", dpi=150, bbox_inches="tight")
```

Combine related plots into a single figure for reports and publications. Save with `dpi=300` for print quality.

What you can do after this module

1. Create histograms with clinical threshold lines
2. Build box plots and bar charts for group comparisons
3. Plot epidemic curves with 7-day rolling averages
4. Generate scatter plots with regression lines and *r* values
5. Produce Kaplan-Meier survival curves
6. Combine everything into multi-panel dashboards

Next: Module 6 — Hypothesis testing

Exercises (Hour 3)

1. **Modify dashboard:** Rebuild the Gapminder dashboard for 2002 data
2. **Clinical dashboard:** 4 panels — BMI histogram, BP box plot by sex, HbA1c vs. age scatter, diagnosis frequency bar chart
3. **Survival:** Kaplan-Meier plot for 3 treatment groups with median survival annotations
4. **Mini-project:** Build a 4-panel health dashboard using Gapminder 2007 data