Design Variables and the Grammar of Graphics

SMM635 - Week 2

Prof. Simone Santoni

Bayes Business School

Today's Journey

Part 1: Grammar of Graphics

- Framework & Philosophy
- Core Components
- Building Blocks

Part 2: Visual Forms

- Univariate Charts
- Bivariate Charts
- Multivariate Charts

Learning Objectives

By the end of today's session, you will:

1. **Understand** the grammar of graphics framework
2. **Map** data to visual variables effectively
3. **Build** complex visualizations from simple components
4. **Implement** layered graphics approaches
5. **Create** appropriate charts for different data types

Part 1: Grammar of Graphics

Moving Beyond Chart Types

How Do We Describe a Chart?

How Do We Describe a Chart?

Traditional Approach:

- Pie chart
- Bar chart
- Line chart
- Scatter plot

> **Note**
>
> **We can use labels or conceptual categories**

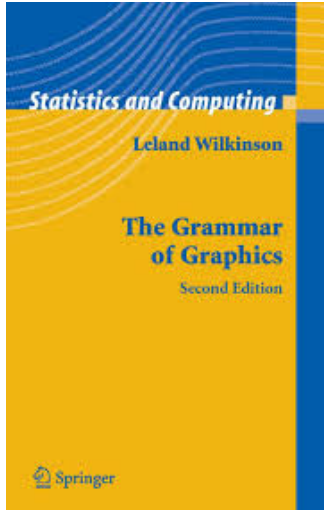
Grammar Approach:

- Data
- Aesthetics
- Geometries
- Scales
- Coordinates

> **Note**
>
> **We can refer to a chart's constitutive components**

What is Grammar of Graphics (GoG)?

"Grammar makes language expressive. A language consisting of words and no grammar expresses only as many ideas as there are words." - Leland Wilkinson



What's the Connection between GoG and ggplot2?

- **ggplot2** is an implementation of the Grammar of Graphics in R
- Created by Hadley Wickham based on Leland Wilkinson's framework
- The "gg" in ggplot2 stands for "Grammar of Graphics"
- Allows users to build plots **layer by layer** using the grammar components
- Instead of choosing from pre-made chart types, you **compose** visualizations from fundamental building blocks

The Power of GoG

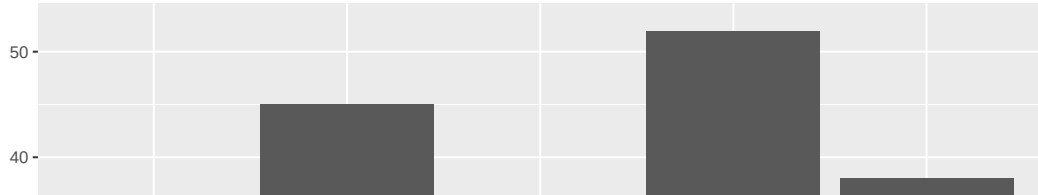
```
# Traditional thinking
make_pie_chart(data)
make_bar_chart(data)

# Grammar thinking
ggplot(data) +
  geom_bar() +
  coord_polar()  # Bar chart → Pie chart!
```

! Important

A pie chart is just a stacked bar chart in polar coordinates!

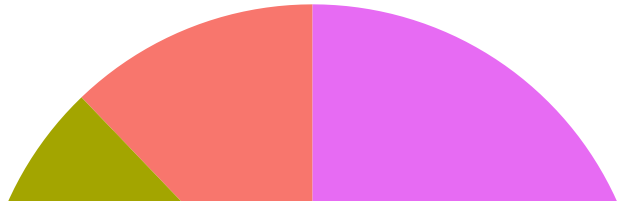
A Bar Chart

```r
library(ggplot2)

# Create data with five categories
data <- data.frame(
  category = c("A", "B", "C", "D", "E"),
  value = c(23, 45, 31, 52, 38)
)

# Create bar chart
ggplot(data, aes(x = category, y = value)) +
  geom_bar(stat = "identity")
```



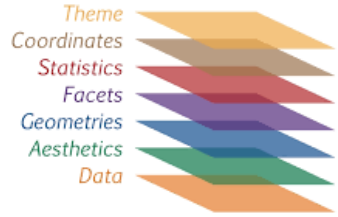
Pie Chart = Bar Chart + Polar Coordinates

```r
# Create data with five categories
data <- data.frame(
  category = c("A", "B", "C", "D", "E"),
  value = c(23, 45, 31, 52, 38)
)

# Create bar chart
ggplot(data, aes(x = "", y = value, fill = category)) +
  geom_bar(stat = "identity") +
  coord_polar(theta = "y", start = 0) +
  theme_void()
```



Core Components of the GoG

1. **DATA**: What we want to visualize
2. **AESTHETICS**: How we map data to visual properties
3. **GEOMETRIES**: The visual marks we use
4. **FACETS**: Creating small multiples
5. **STATISTICS**: How to transform or summarize the raw data
6. **COORDINATES**: The space we're working in
7. **THEMES**: Overall visual appearance



Source: https://r.qcbs.ca/

1. Data: The Foundation

```r
# Data is structured information
sales_data <- data.frame(
  month = c("Jan", "Feb", "Mar", "Apr'
  revenue = c(45000, 52000, 48000, 61(
  region = c("North", "North", "South'
)
```

Tip

Good visualization starts with well-structured data
Tidyverse is your friend!

2. Aesthetics: Visual Mappings

Mapping Data → Visual Properties

Data Variables

- Continuous values
- Categories
- Ordered factors
- Time series

Visual Variables

- Position (x, y)
- Size
- Color
- Shape
- Transparency
- Line type

date	region	sales	profit
2024-01-01	North	1250.50	325.15
2024-02-01	North	980.75	245.20
2024-01-01	South	1450.25	410.75
2024-02-01	South	1100.00	290.50

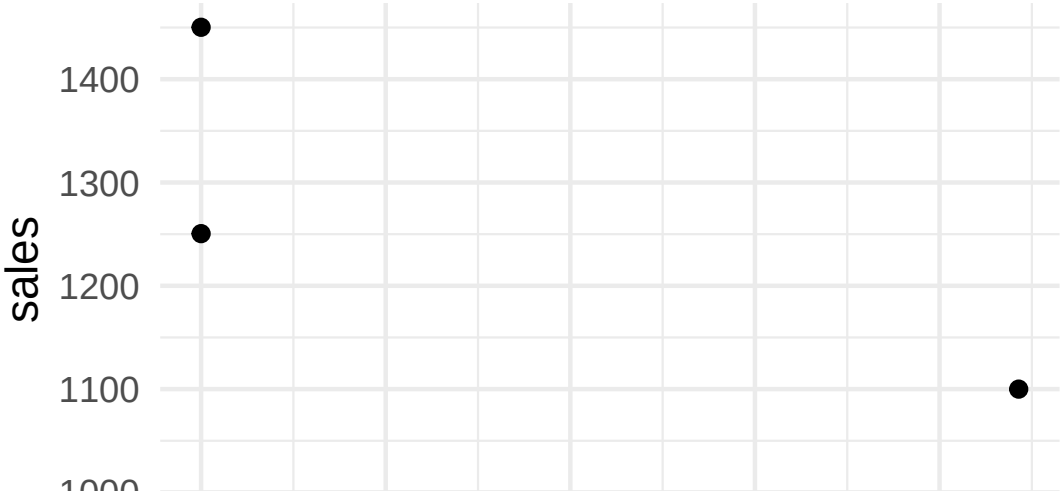
Visual Variables in Action

```
ggplot(sample_data, aes(
  x = date,         # Position
  y = sales,        # Position
  color = region,   # Color
  size = profit     # Size
)) +
  geom_point()
```

3. Geometries: Visual Marks

Points

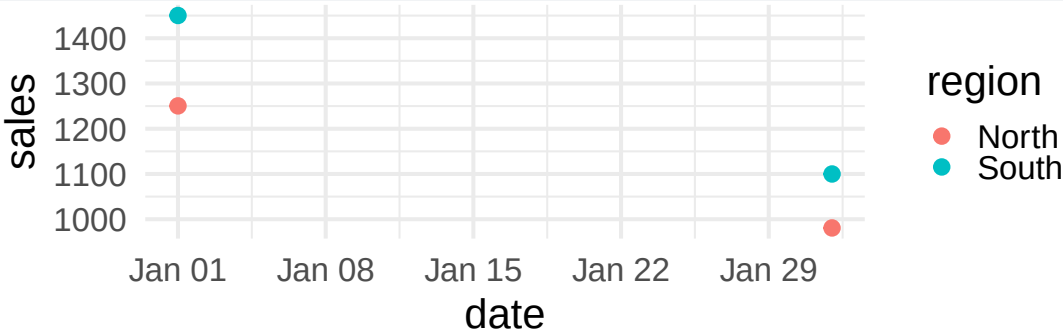
```
ggplot(sample_data, aes(x = date, y = sales)) +
  geom_point()
```



4. Facets: Small Multiples

No Facets

```
ggplot(sample_data, aes(x = date, y = sales, color = region)) +
  geom_point()
```



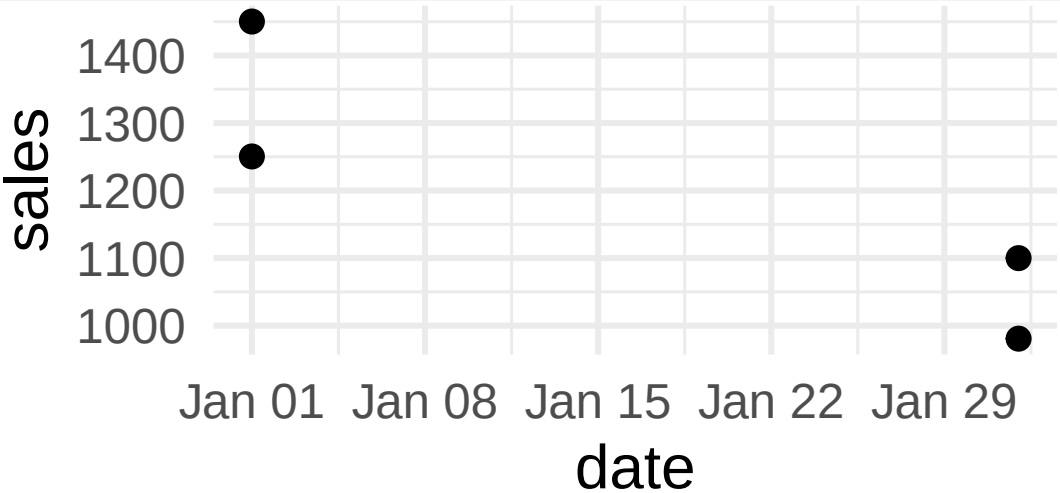
Note

All data in a single plot

5. Statistics: Transforming Data

Raw Data

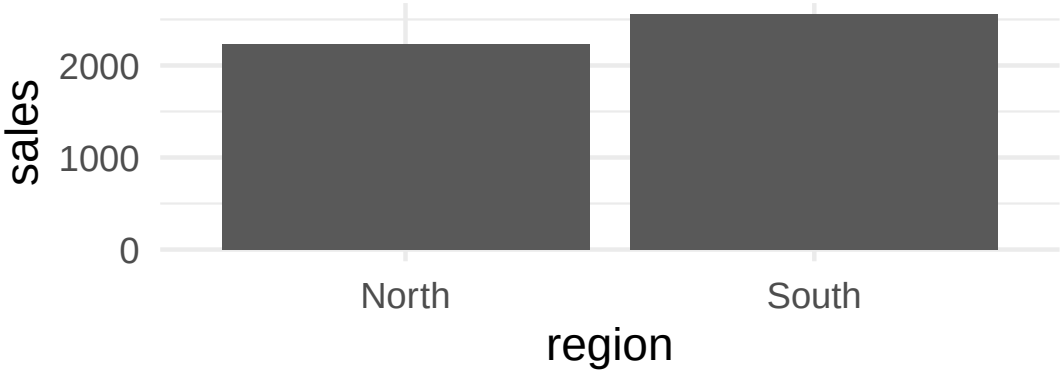
```
ggplot(sample_data, aes(x = date, y = sales)) +
  geom_point()
```



6. Coordinates: The Canvas

Cartesian (default)

```
ggplot(sample_data, aes(x = region, y = sales)) +
  geom_bar(stat = "identity") +
  coord_cartesian()
```

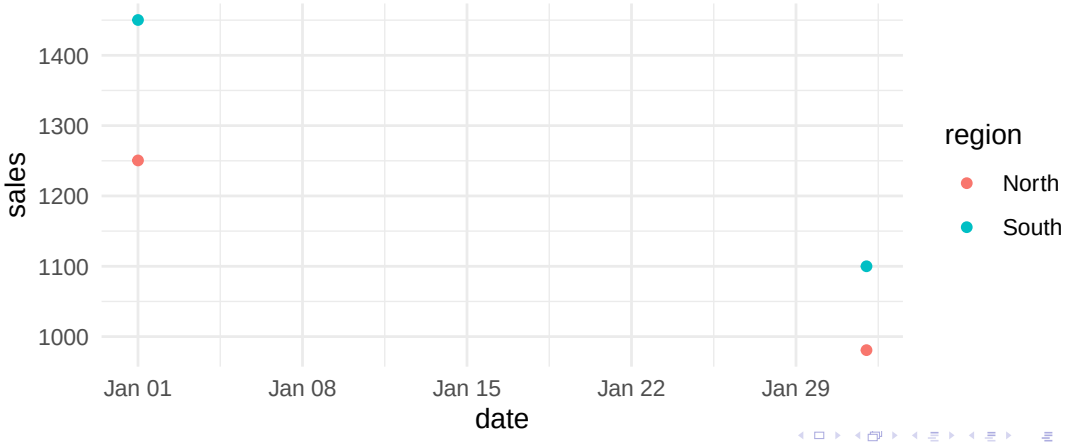


Note

7. Themes: Overall Visual Appearance

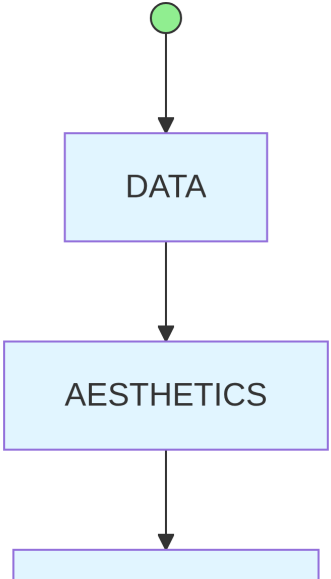
theme_minimal()

```
ggplot(sample_data, aes(x = date, y = sales, color = region)) +
  geom_point() +
  theme_minimal()
```

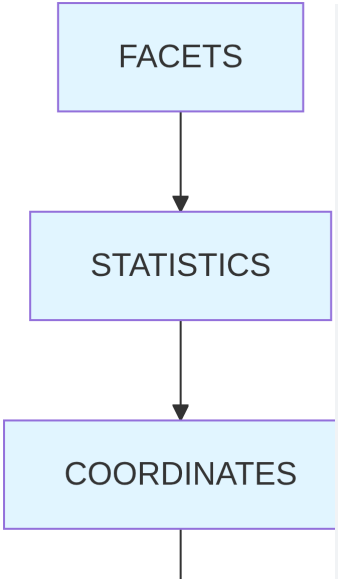


Building Complex from Simple

Part 1: Foundation



Part 2: Refinement

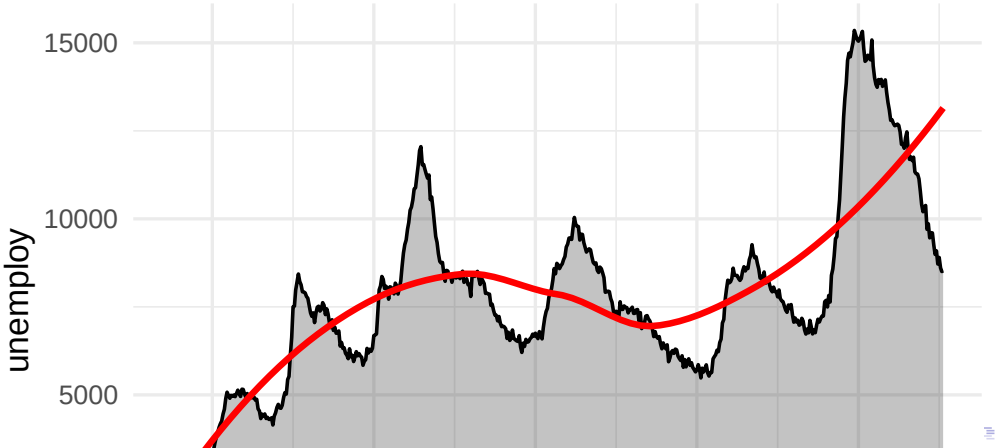


Example: Layer by Layer

```
# 1. Data + Aesthetics
ggplot(data, aes(x, y)) +

  # 2. Geometry
  geom_point() +

  # 3. Facets
  facet_wrap(~category) +

  # 4. Statistics
  geom_smooth() +

  # 5. Coordinates
  coord_cartesian() +

  # 6. Theme
```

Layering: The Power of Composition

```
ggplot(economics, aes(date, unemploy)) +
  geom_area(alpha = 0.3) +           # Layer 1: Area
  geom_line(size = 1.2) +            # Layer 2: Line
  geom_smooth(se = FALSE, col = "red") # Layer 3: Trend
```



Part 2: Visual Forms

From Simple to Complex

Univariate Charts

Exploring Single Variables

Continuous Data

- Histograms
- Density plots
- Box plots
- Violin plots

Categorical Data

- Bar charts
- Pie charts
- Waffle charts
- Dot plots

Univariate: Continuous Data

Histogram

```
ggplot(data, aes(x = value)) +
  geom_histogram(bins = 30)
```

Histograms divide data into bins and count observations in each bin.

- **Best for:** Understanding the distribution shape and identifying patterns
- **Shows:** Frequency, central tendency, spread, and skewness
- **Key parameter:** Number of bins affects granularity

Density

```
ggplot(data, aes(x = value)) +
  geom_density(fill = "skyblue", alpha = 0.5)
```

Density plots show a smoothed version of the distribution.

- **Best for:** Comparing multiple distributions, identifying modes
- **Shows:** Probability density across the range of values
- **Advantage:** Smooth curve makes patterns easier to see

Univariate: Categorical Data

Bar Chart

```
ggplot(data, aes(x = category)) +
  geom_bar()
```

Bar charts use bar length to encode category counts or values.

- **Best for:** Comparing categories, showing rankings
- **Shows:** Frequency or magnitude for each category
- **Advantage:** Easy to compare values, natural visual ordering

Pie Chart

```
ggplot(data, aes(x = "", fill = category)) +
  geom_bar() +
  coord_polar("y")
```

Pie charts show parts of a whole as slices of a circle.

- **Best for:** Showing proportions when there are few categories (2-5)
- **Shows:** Relative proportions and percentages
- **Limitation:** Difficult to compare similar-sized slices

Bivariate Charts

Exploring Relationships Between Two Variables

X Variable	Y Variable	Best Chart Types
Continuous	Continuous	Scatter plot, Line chart
Continuous	Categorical	Box plot, Violin plot
Categorical	Categorical	Heatmap, Grouped bars
Time	Continuous	Line chart, Area chart

Bivariate: Continuous × Continuous

Scatter Plot

```
ggplot(data, aes(x = height, y = weight)) +
  geom_point()
```

Scatter plots display individual data points in 2D space.

- **Best for:** Exploring relationships, identifying correlations, spotting outliers
- **Shows:** Direction, strength, and form of relationship between two variables
- **Key insight:** Patterns reveal linear, non-linear, or no correlation

With Trend

```
ggplot(data, aes(x = height, y = weight)) +
  geom_point() +
  geom_smooth(method = "lm")
```

Scatter plot with trend line adds a fitted model to show the relationship.

- **Best for:** Confirming correlation patterns, making predictions
- **Shows:** Overall trend and strength of linear relationship
- **Options:** Linear (lm), loess (local smoothing), or other methods

Bivariate: Categorical × Continuous

Grouped Box Plot

```
ggplot(data, aes(x = category, y = value)) +
  geom_boxplot()
```

Grouped box plots compare distributions across multiple categories.

- **Best for:** Comparing central tendency and spread across groups
- **Shows:** Median, quartiles, and outliers for each category
- **Advantage:** Compact representation of multiple distributions side-by-side

Violin Plot

```
ggplot(data, aes(x = category, y = value)) +
  geom_violin()
```

Violin plots combine box plots with kernel density estimation.

- **Best for:** Revealing distribution shapes and multimodality
- **Shows:** Full distribution shape for each category
- **Advantage:** More informative than box plots for complex distributions

Multivariate Charts

Beyond Two Dimensions

Strategies for encoding multiple variables:

1. **Color/Fill**: 3rd dimension
2. **Size**: 4th dimension
3. **Shape**: 5th dimension (categorical only)
4. **Faceting**: Create small multiples
5. **Animation**: Time as dimension

Multivariate Example: The Economics Dataset

Table 3: US Economic Time Series Data (1967-2015)

date	pce	pop	psavert	uempmed	unemploy
1967-07-01	506.7	198712	12.6	4.5	2944
1967-08-01	509.8	198911	12.6	4.7	2945
1967-09-01	515.6	199113	11.9	4.6	2958
1967-10-01	512.2	199311	12.9	4.9	3143
1967-11-01	517.4	199498	12.8	4.7	3066
1967-12-01	525.1	199657	11.8	4.8	3018

Multivariate Example: The Economics Dataset

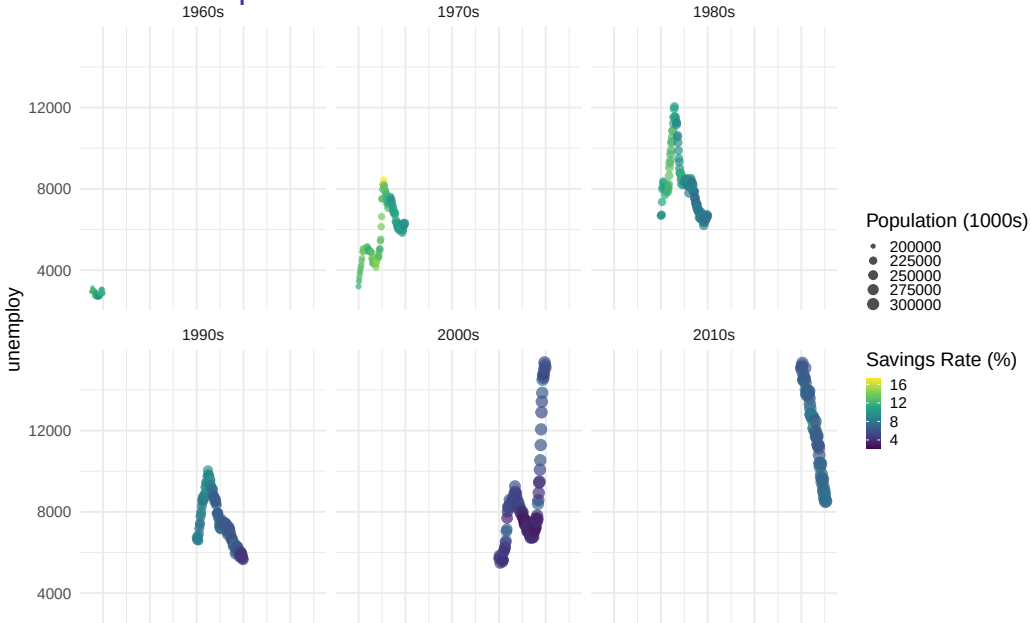
Note

Dataset Variables:

- **date**: Month of data collection
- **pce**: Personal consumption expenditures (billions USD)
- **pop**: Total population (thousands)
- **psavert**: Personal savings rate (%)
- **uempmed**: Median duration of unemployment (weeks)
- **unemploy**: Number of unemployed (thousands)

Multivariate Example

Multivariate Example



Best Practices for Multivariate

1. **Start simple**: Add dimensions gradually

Best Practices for Multivariate

1. **Start simple**: Add dimensions gradually
2. **Prioritize**: Most important variables get best encodings

Best Practices for Multivariate

1. **Start simple**: Add dimensions gradually
2. **Prioritize**: Most important variables get best encodings
3. **Test perception**: Can viewers decode all dimensions?

Best Practices for Multivariate

1. **Start simple**: Add dimensions gradually
2. **Prioritize**: Most important variables get best encodings
3. **Test perception**: Can viewers decode all dimensions?
4. **Consider alternatives**: Sometimes multiple simple charts $>$ one complex chart

Best Practices for Multivariate

1. **Start simple**: Add dimensions gradually
2. **Prioritize**: Most important variables get best encodings
3. **Test perception**: Can viewers decode all dimensions?
4. **Consider alternatives**: Sometimes multiple simple charts $>$ one complex chart
5. **Interactive solutions**: Tooltips, filtering, zooming

Putting It All Together

A Practical Workflow

Tip

1. **Understand your data**
   - Types of variables
   - Relationships to explore
2. **Choose appropriate forms**
   - Match chart to data type
   - Consider your message
3. **Apply the grammar**
   - Map variables to aesthetics
   - Layer geometries
   - Refine with scales

Key Takeaways

- The Grammar of Graphics provides a **systematic framework** for creating any visualization

Key Takeaways

- The Grammar of Graphics provides a **systematic framework** for creating any visualization
- Complex visualizations are built from **simple, reusable components**

Key Takeaways

- The Grammar of Graphics provides a **systematic framework** for creating any visualization
- Complex visualizations are built from **simple, reusable components**
- Visual variables (position, size, color, etc.) are tools for **encoding information**

Key Takeaways

- The Grammar of Graphics provides a **systematic framework** for creating any visualization
- Complex visualizations are built from **simple, reusable components**
- Visual variables (position, size, color, etc.) are tools for **encoding information**
- Choose chart types based on **data types and relationships**

Key Takeaways

- The Grammar of Graphics provides a **systematic framework** for creating any visualization
- Complex visualizations are built from **simple, reusable components**
- Visual variables (position, size, color, etc.) are tools for **encoding information**
- Choose chart types based on **data types and relationships**
- Iteration and layering lead to **rich, informative graphics**

Next Week

Topic 3: Exploratory Data Analysis

- EDA workflow and visualization
- Distribution visualization techniques
- Correlation and relationship exploration
- Time series exploration
- Case Study: Nomis Solutions

Homework

- Practice creating layered visualizations
- Experiment with different coordinate systems
- Read: Wickham's "Layered Grammar of Graphics"

Questions?

Let's explore the grammar together!

Simone.Santoni.1@city.ac.uk

Course website: https://simonesantoni.github.io/data-viz-smm635

Office hours: Wednesdays 3-5 PM