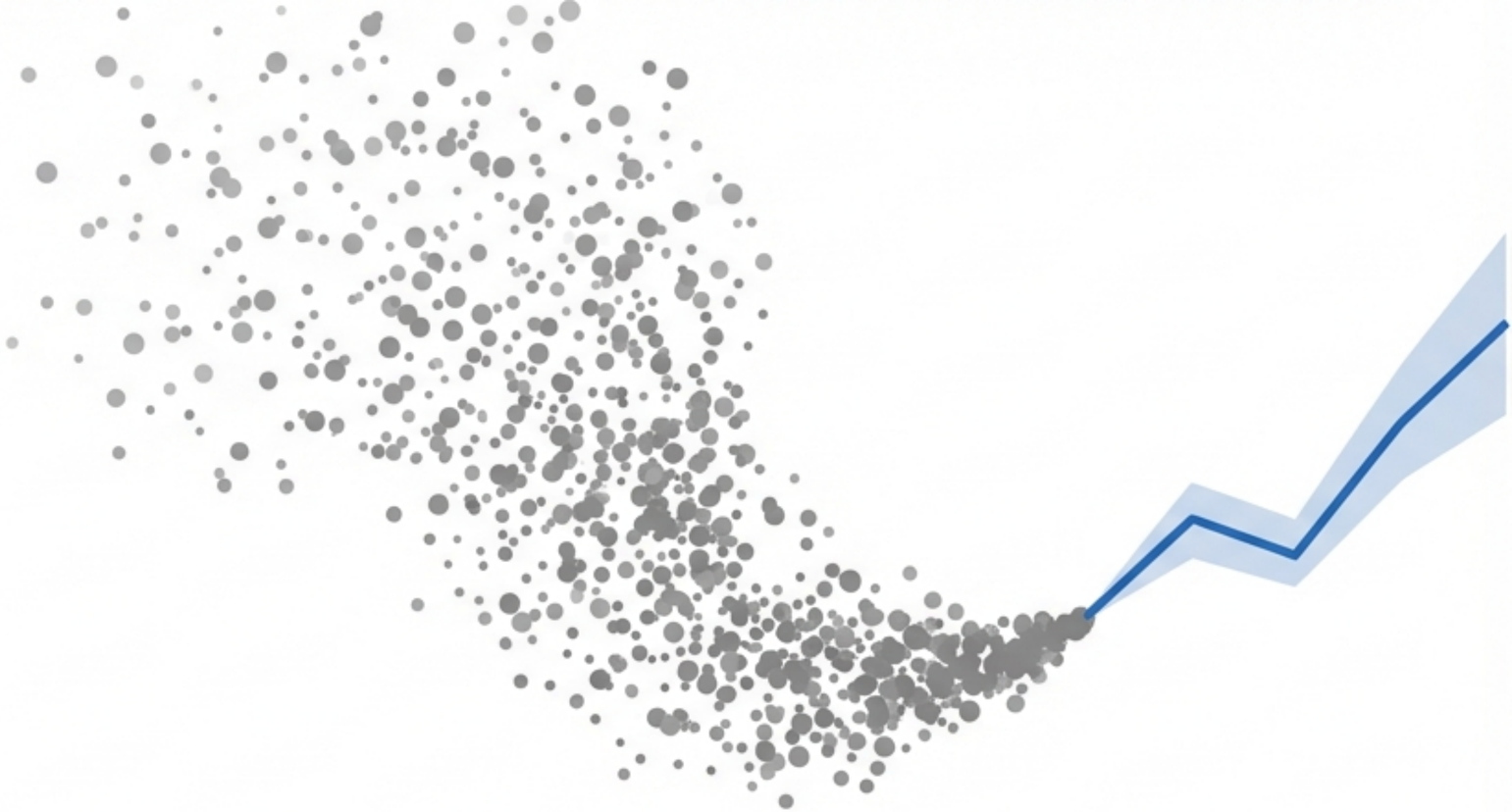
R PROGRAMMING AND
APPLICATIONS TO FINANCE



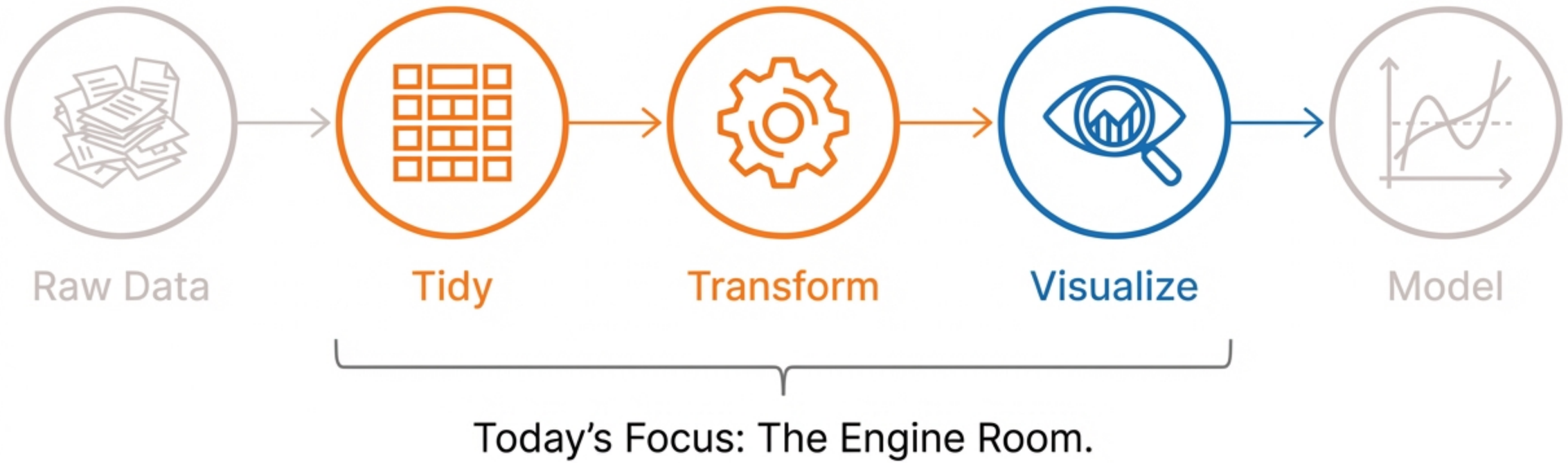
# Lecture 2
# Data Manipulation & The Grammar of Graphics

The Financial Data Pipeline: From Chaos to Clarity

R in Finance
NotebookLM

# The Analyst’s Workflow

From Raw Tickers to Alpha. Before we can model volatility or optimize portfolios, we we must structure and explore the data.



# The Philosophy of Tidy Data

## Messy / Wide

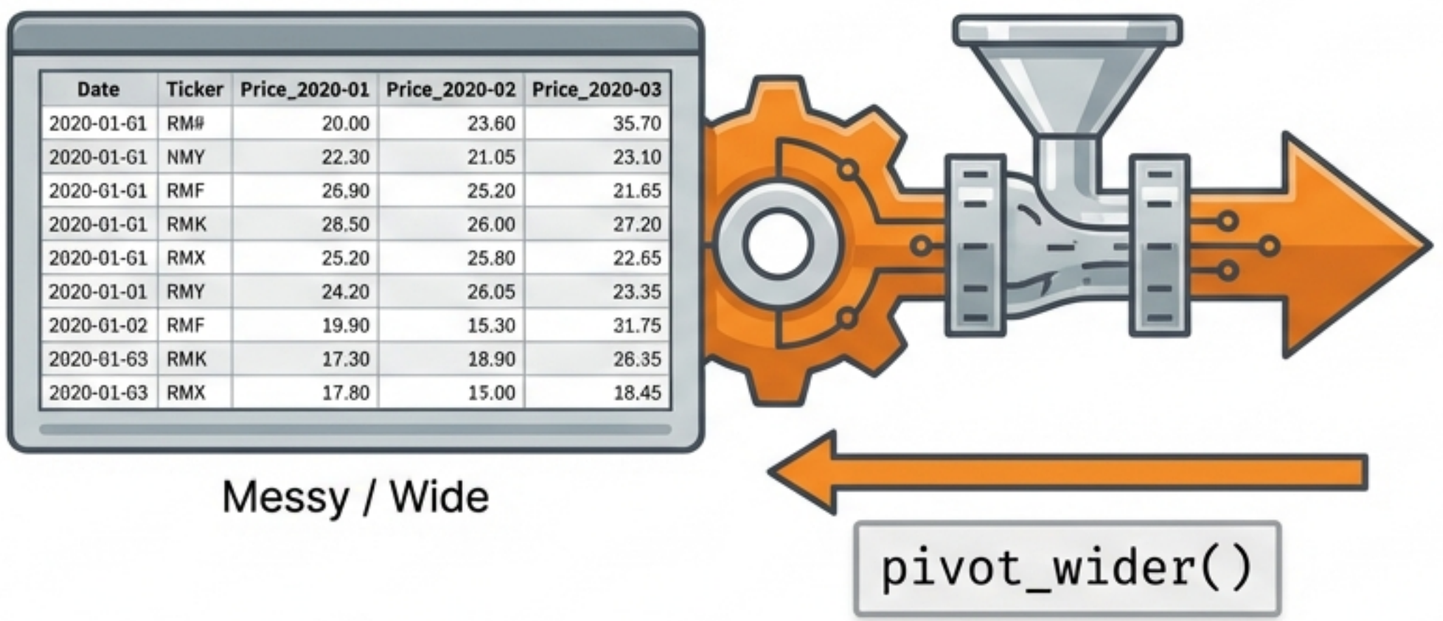
| Ticker | 2020-01 | 2020-02 | 2020-03 |
|---|---|---|---|
| AAPL | 299.80 | 299.80 | at |
| GOOG | 1438.20 | 1438.20 | ar ds_c hien ing |
| MSFT | 309.51 | 309.51 | 309.51 |
| AAPL | 299.80 | 299.80 | 296.70 |
| MSFT | 1440.30 | 1440.30 | 1440.77 |

1. Variables = Columns
2. Observations = Rows
3. Values = Cells

## Tidy / Long

| Date | Ticker | Price |
|---|---|---|
| 2020-01-01 | AAPL | 299.80 |
| 2020-01-01 | GOOG | 1438.20 |
| 2020-02-01 | AAPL | 309.51 |
| 2020-02-01 | GOOG | 1440.30 |
| 2020-02-01 | GOOG | 1440.30 |
| 2020-02-01 | AAPL | 309.5 |
| ... | | |

# Reshaping Reality: **Pivoting**




| Date | Ticker | Price_2020-01 | Price_2020-02 | Price_2020-03 |
|---|---|---|---|---|
| 2020-01-01 | RM9 | 20.00 | 23.60 | 35.70 |
| 2020-01-01 | NMY | 22.30 | 21.05 | 23.10 |
| 2020-01-01 | RMF | 26.90 | 25.20 | 21.65 |
| 2020-01-01 | RMK | 28.50 | 26.00 | 27.20 |
| 2020-01-01 | RMX | 25.20 | 25.80 | 22.65 |
| 2020-01-01 | RMY | 24.20 | 26.05 | 23.35 |
| 2020-01-02 | RMF | 19.90 | 15.30 | 31.75 |
| 2020-01-03 | RMK | 17.30 | 18.90 | 26.35 |
| 2020-01-03 | RMX | 17.80 | 15.00 | 18.45 |

Messy / Wide

| Date | Ticker | Price |
|---|---|---|
| 2020-01-01 | RM9 | 35.90 |
| 2020-01-01 | NMY | 21.60 |
| 2020-01-01 | RMF | 21.50 |
| 2020-01-01 | RMK | 22.63 |
| 2020-01-01 | RMX | 26.20 |
| 2020-01-01 | RMY | 32.05 |
| 2020-01-01 | RMF | 22.90 |
| 2020-01-01 | RMK | 21.50 |
| 2020-01-01 | RMX | 26.09 |
| 2020-01-01 | RMX | 21.20 |
| 2020-01-01 | RMY | 21.70 |
| 2020-01-02 | RMY | 21.85 |
| 2020-01-03 | RMF | 22.65 |
| 2020-01-03 | RMK | 28.50 |
| 2020-01-03 | RMX | 21.00 |
| 2020-01-03 | RMK | 25.60 |
| 2020-01-03 | RMX | 21.80 |
| 2020-01-03 | RMY | 22.70 |
| 2020-01-03 | RMY | 22.25 |
| 2020-01-03 | RMF | 22.45 |
| 2020-01-03 | RMK | 23.00 |
| 2020-01-03 | RMR | 26.65 |
| 2020-01-03 | RNX | 21.65 |
| 2020-01-03 | RMR | 21.50 |

Tidy / Long

Financial time series often arrive in Wide format (Excel-friendly) but require Long format for modeling and ggplot2.

# The Grammar of Data: The Pipe

Data (Subject)


Function (Verb)

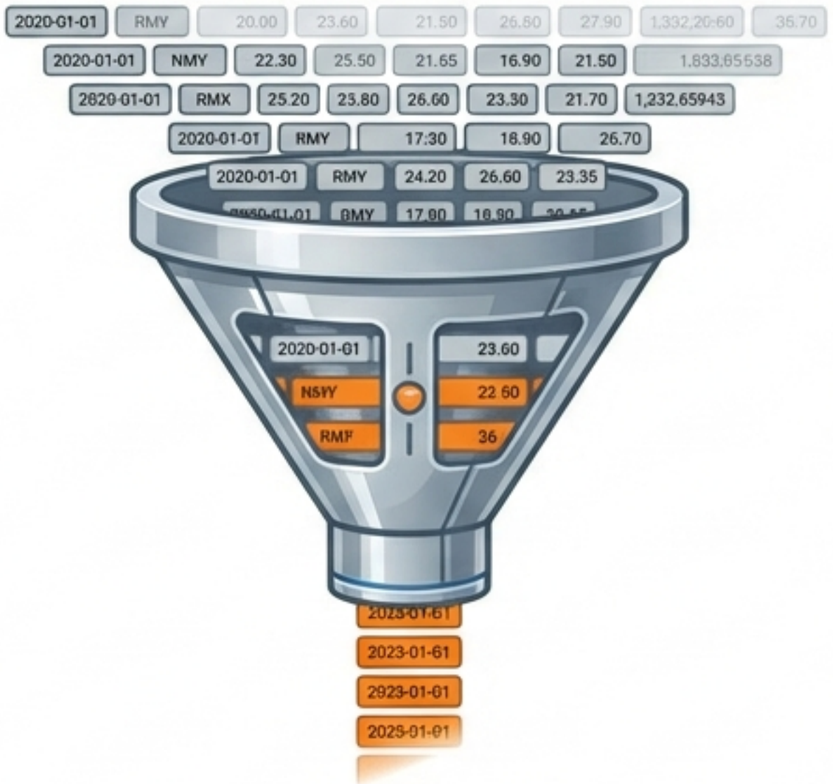
**Nested (Hard to read):**

```
summarize(group_by(filter(data,
  Return > 0), Ticker),
  Mean = mean(Return))
```

**Piped (Readable):**

```
data |> filter(Return > 0) |>
  group_by(Ticker) |>
  summarize(Mean = mean(Return))
```

# The Toolkit: Extracting Cases & Variables



| Date | Ticker | Open | High | Low | Close | Adjusted_Close | Volume |
|---|---|---|---|---|---|---|---|
| 2020-01-01 | RM# | 20.00 | 23.60 | 23.60 | 36.90 | 36.90 | 36,541 |
| 2020-01-01 | NMY | 22.30 | 21.05 | 21.05 | 21.50 | 21.60 | 16,867 |
| 2020-01-01 | RMF | 22.30 | 22.03 | 22.63 | 32.03 | 32.03 | 1,284 |
| 2020-01-01 | RMK | 26.90 | 25.20 | 21.50 | 22.90 | 26.90 | 41,882 |
| 2020-01-01 | RMF | 26.90 | 26.00 | 22.20 | 21.20 | 22.00 | 2,822 |
| 2020-01-01 | RMX | 25.20 | 25.80 | 26.00 | 21.70 | 21.70 | 7,502 |
| 2020-01-01 | RMY | 24.20 | 26.05 | 23.33 | 22.65 | 22.65 | 3,807 |
| 2020-01-01 | RMY | 24.20 | 26.05 | 26.05 | 23.35 | 28.50 | 130,500 |
| 2020-01-01 | RMF | 17.30 | 15.30 | 31.75 | 21.80 | 21.80 | 1,026 |
| 2020-01-02 | RMY | 17.30 | 18.90 | 22.90 | 22.70 | 22.70 | 3,880 |
| 2020-01-03 | RMX | 17.80 | 18.90 | 22.25 | 22.25 | 26.65 | 7,682 |
| 2020-01-03 | RMX | 17.80 | 15.00 | 22.00 | 23.00 | 21.50 | 4,900 |

Extract rows (cases).

```
data |> filter(Date > '2023-01-01')
```

Extract columns (variables).

```
data |> select(Date, Adjusted_Close)
```

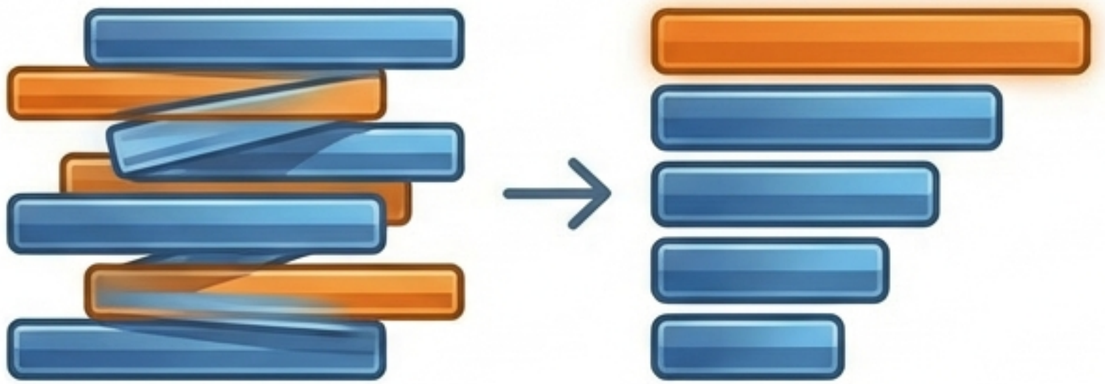
# The Toolkit: Deriving & Arranging

## `mutate()`


Create new variables.

```
data |> mutate(LogRet =
  log(Price / lag(Price)))
```

## `arrange()`

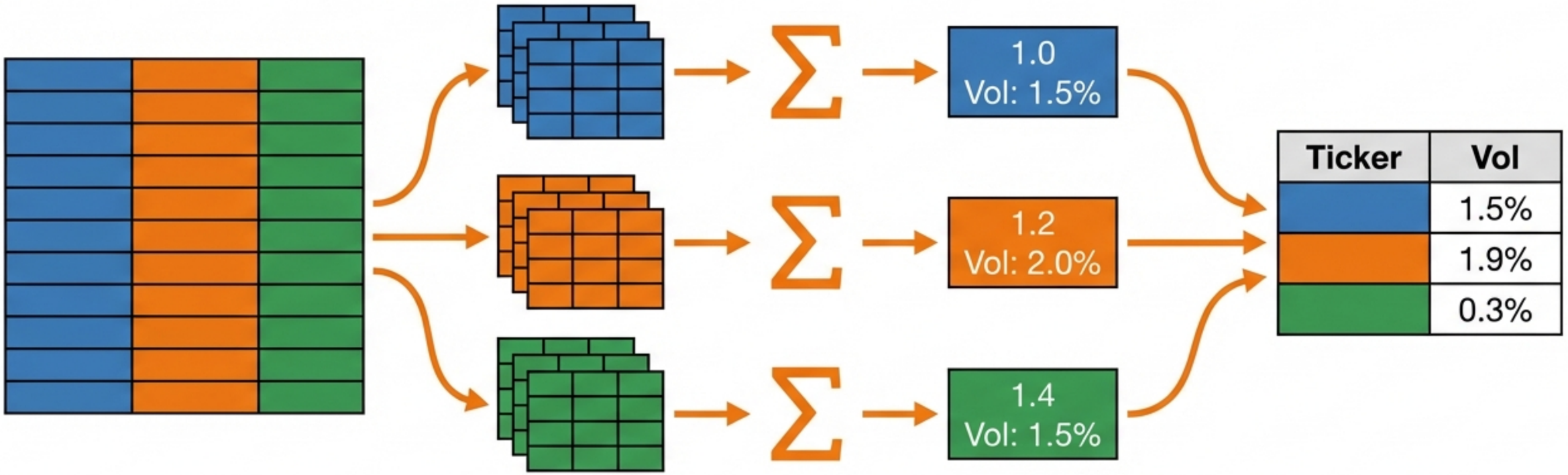


Reorder rows.

```
data |> arrange(desc(LogRet))
```

Modern Financial Architect

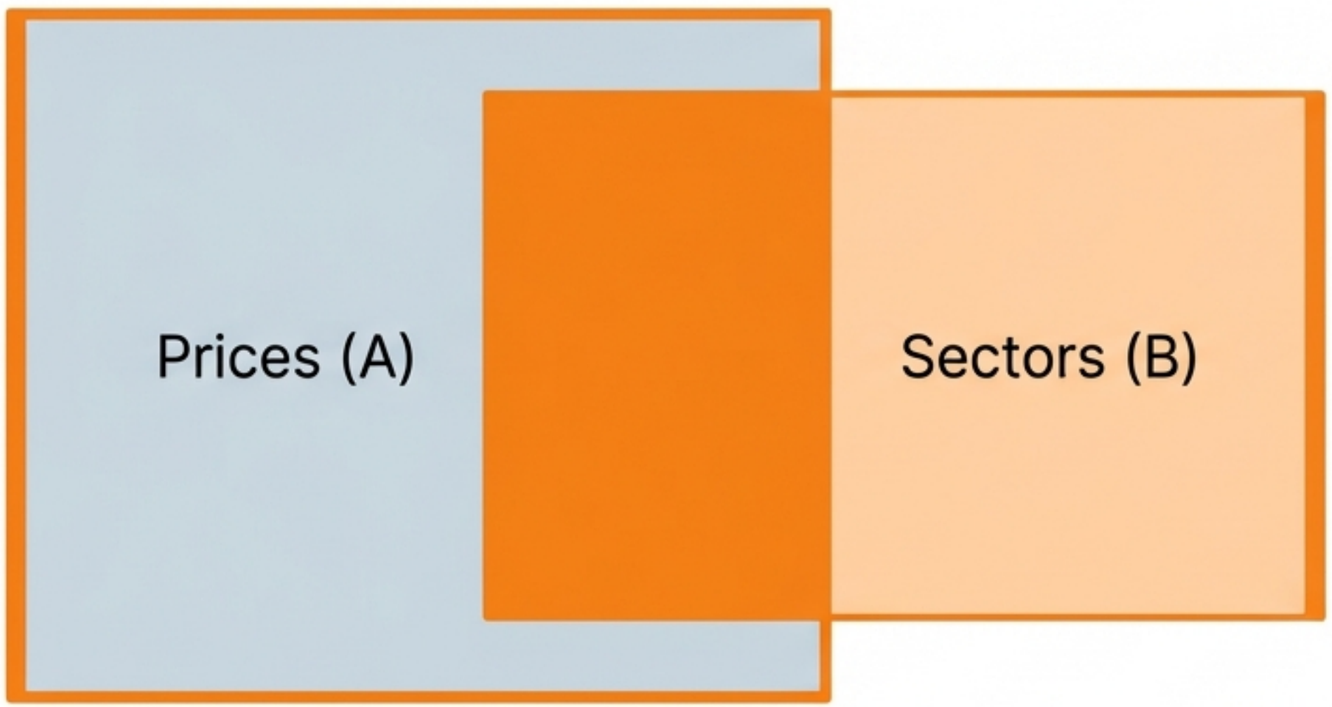
# The Power Combo: Group & Summarize



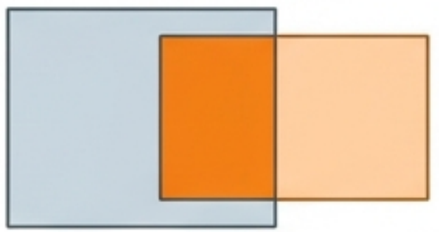

```
data |>
  group_by(Ticker) |>
  summarize(Vol = sd(LogRet))
```

# Relational Data: Joins

Combining data from multiple sources based on shared keys.

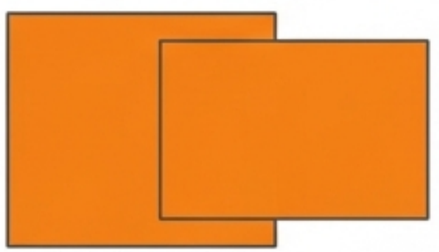



`left_join(A, B)`: Keep all Prices, add Sector info where available.



**Inner Join**

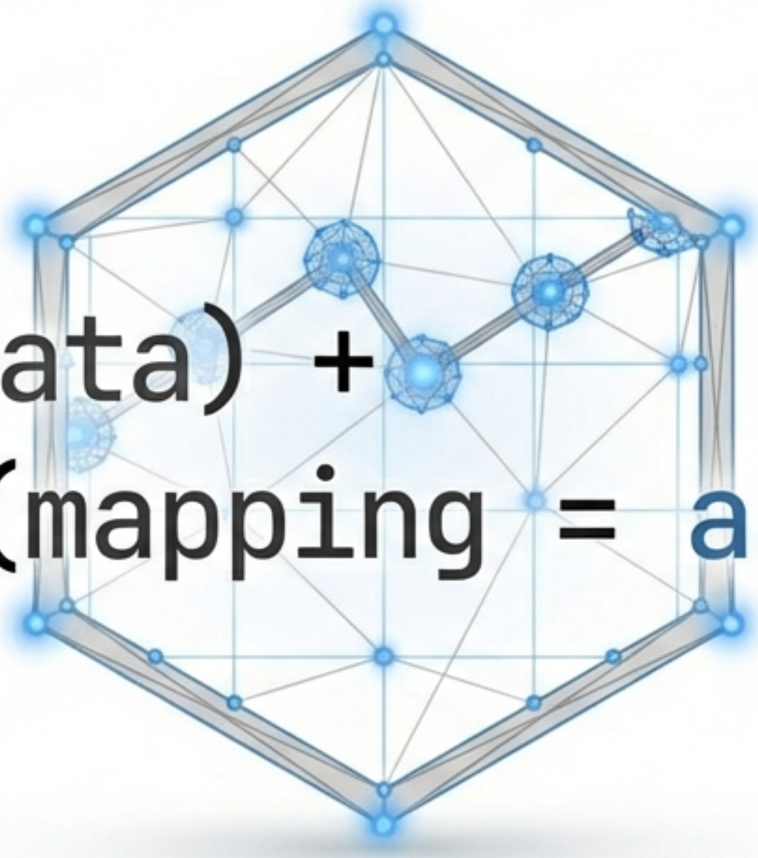
`inner_join(A, B)`: Only keep matching rows.



**Full Join**

`full_join(A, B)`: Keep all rows from both.

# The Grammar of Graphics

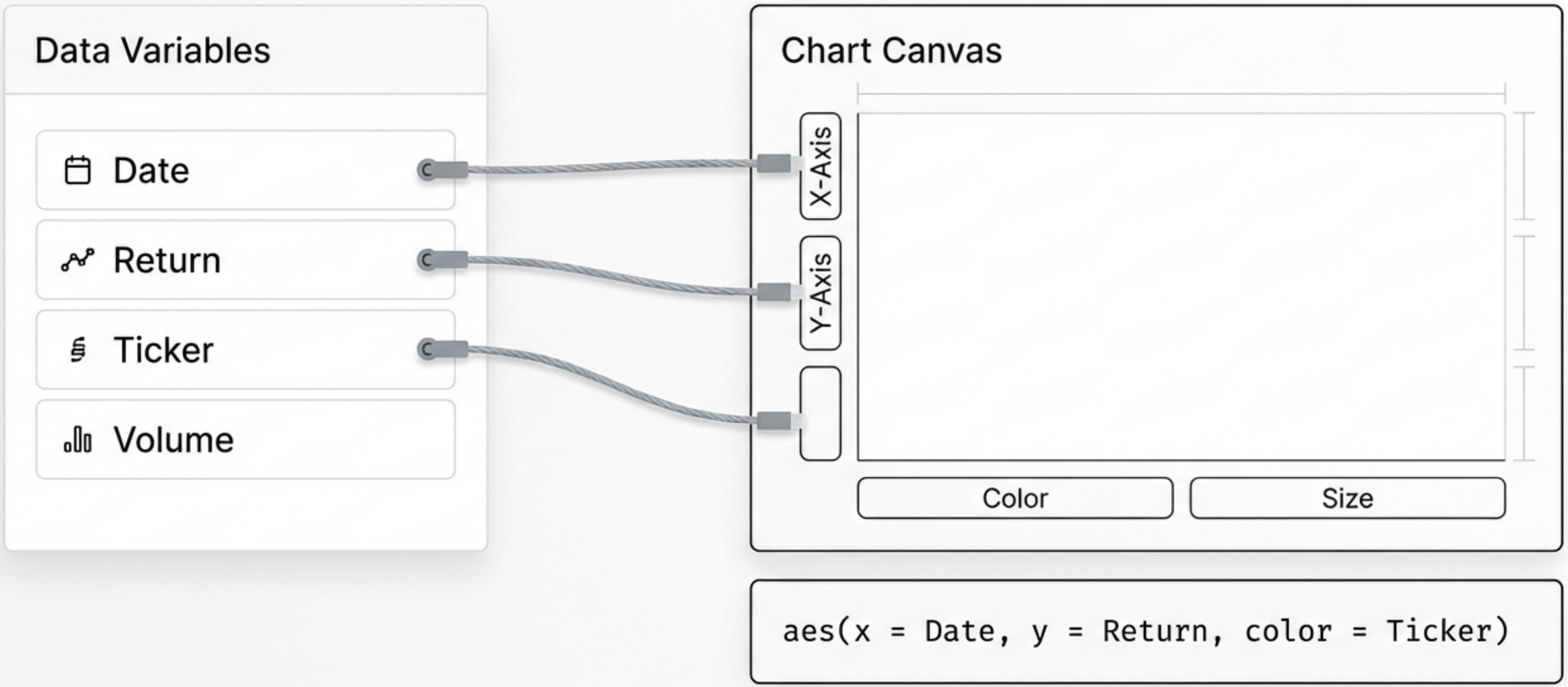


```
ggplot(data) +
  <geom>(mapping = aes(x, y))
```

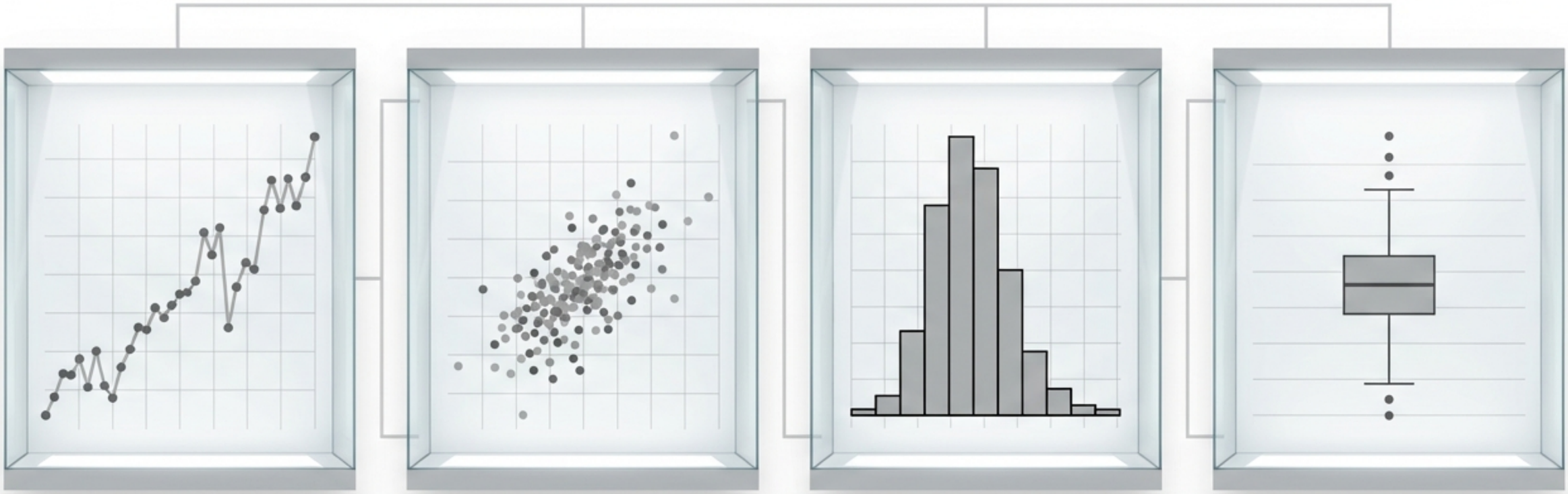
We do not just draw charts. We build them, layer by layer.

1. Data
2. Aesthetics (Mappings)
3. Geometries (Objects)

# Layer 1: Data & Aesthetics



# Layer 2: Geometries



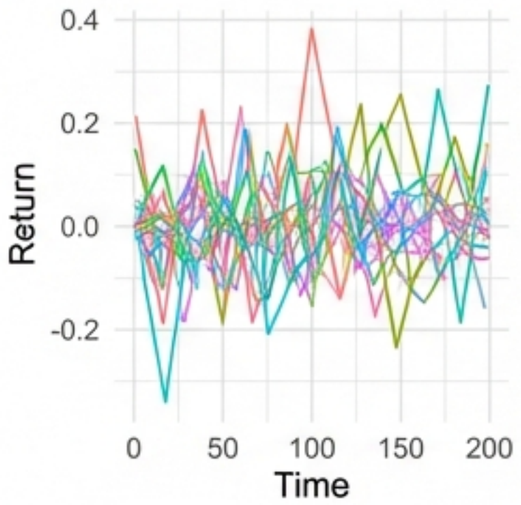
`geom_line()`
(Time Series)

`geom_point()`
(Risk/Return)

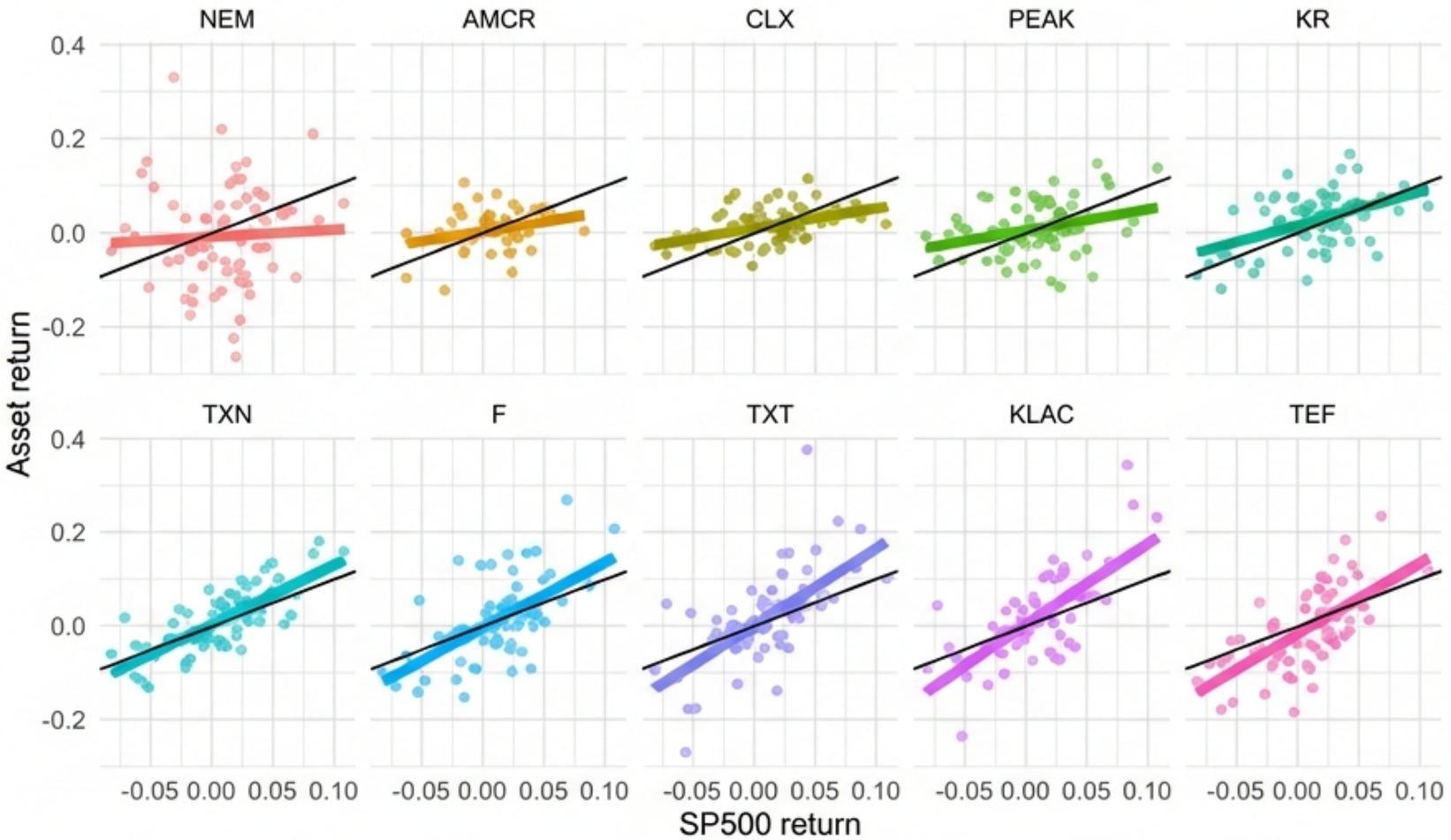
`geom_histogram()`
(Returns Dist)

`geom_boxplot()`
(Volatility)

# Layer 3: Faceting (Small Multiples)

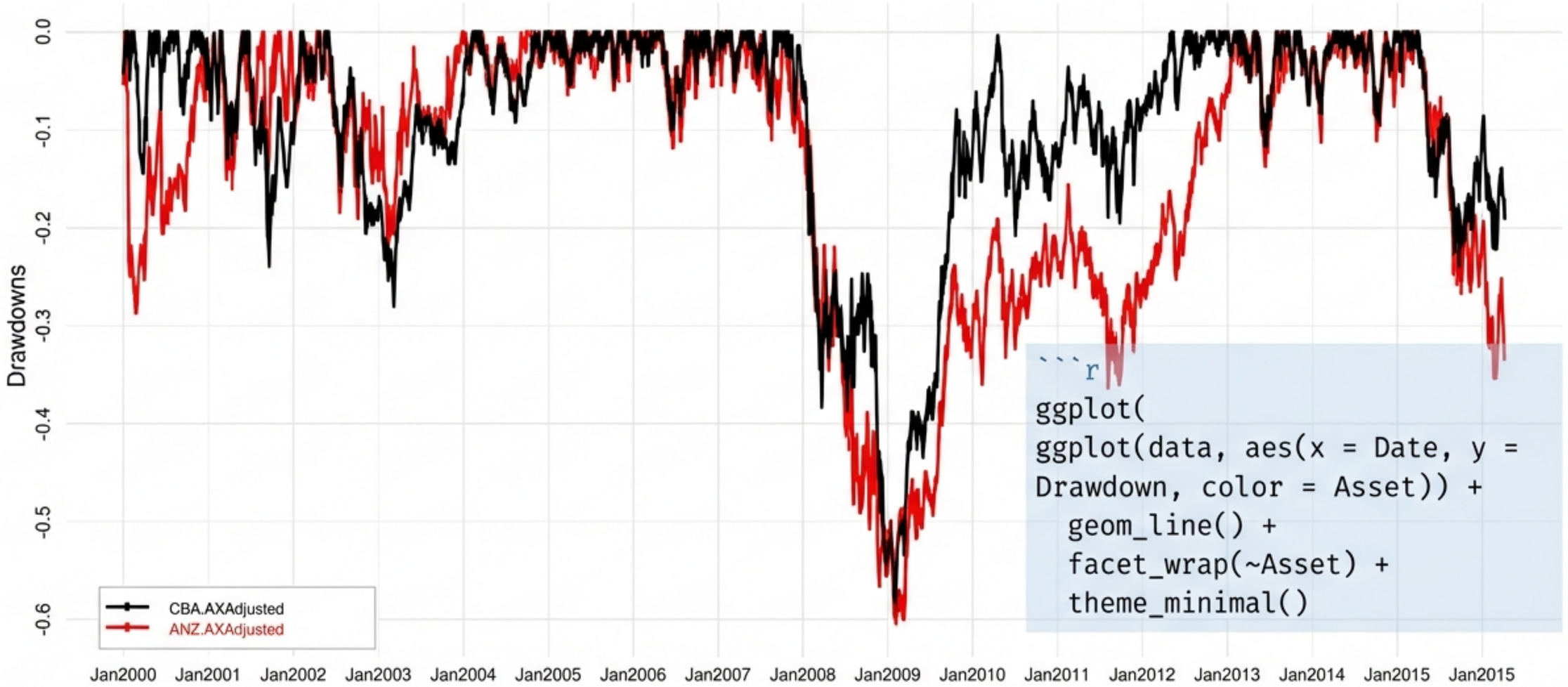



Unreadable Spaghetti Plot




```
+ facet_wrap(~Ticker)
```

# The Chef d'Oeuvre




```r
ggplot(
ggplot(data, aes(x = Date, y =
Drawdown, color = Asset)) +
  geom_line() +
  facet_wrap(~Asset) +
  theme_minimal()
```

# Summary & Next Steps

## Tidy

Structure data (Long format) for analysis.

## Transform

Filter, mutate, and aggregate with dplyr.

## Visualize

Build plots layer-by-layer with ggplot2.

**Seminar Tasks:**

- Benchmarking dplyr vs data.frame
- Building the Chef d’Oeuvre graphic

**Home Assignment:**

- Analyze a stock’s performance using these tools.