CHAPTER 1

The Scientific Study of Politics

GOVT 10: Quantitative Political Analysis

> This chapter introduces the fundamental question: why should we use data and statistics to study politics? We explore how systematic, quantitative analysis allows us to answer questions that intuition and casual observation cannot. Human cognition is subject to predictable biases that distort our understanding of political phenomena. The scientific method, applied to political questions, provides tools for overcoming these limitations. We also introduce R, the programming language we will use throughout this course, and learn to work with the basic building blocks of data analysis: variables, vectors, and data frames.

1 The Limits of Intuition

Consider a question that campaigns spend millions of dollars trying to answer: do negative political advertisements actually persuade voters? Your intuition might suggest that of course they work, since campaigns would not spend so much money on them otherwise. Or perhaps you believe that people hate negative ads and they simply turn voters off. Maybe it depends on the voter and the ad. All of these answers sound reasonable, but which is correct? How would we know?

This is precisely the kind of question that separates casual political observation from political science. Without systematic analysis, we are left with competing intuitions and no way to resolve them. The problem is not that intuition is useless. Rather, the problem is that human cognition is subject to predictable biases that systematically distort our understanding of political phenomena.

The availability bias leads us to remember vivid, unusual events while forgetting the mundane. A single dramatic campaign ad sticks in memory while thousands of forgettable ads disappear from consciousness. Confirmation bias causes us to notice evidence that confirms what we already believe while dismissing contradictory information. If you think negative ads are effective, you will remember the ones that appeared to "work" and forget the ones that did not.

Perhaps most insidiously, we engage in selection on the dependent variable. We observe outcomes but not counterfactuals. We see that a candidate who ran negative ads won their election, but we do not see what would have happened if they had not run those ads. Finally, confounding relationships between variables mislead us constantly. Candidates who are behind in polls tend to go negative, so negative ads correlate with losing. But that does not mean the ads caused the loss.

The solution to these problems is systematic, quantitative analysis. By collecting data carefully, measuring variables precisely, and using statistical tools to analyze patterns, we can move beyond

intuition toward genuine understanding of political phenomena.

2 Introduction to R

R is a programming language designed for statistical computing and data analysis. It is free, open-source, and widely used in academia and data science. We will use R throughout this course to explore data, calculate statistics, and test hypotheses about political phenomena.

2.1 Variables and Assignment

In R, we store values in variables using the assignment operator <−. Think of variables as labeled containers that hold information. Once we assign a value to a variable, we can refer to that value by name.

```
approval_rating <- 42
president <- "Biden"
is_democrat <- TRUE
approval_rating
```

```
[1] 42
```

```
president
```

```
[1] "Biden"
```

> **Why This Syntax?**
>
> **Q: Why <− instead of =?**
>
> A: Both work, but <− is the R convention. It makes clear that you're putting a value INTO a container (the arrow points where data goes). Many R users reserve = for function arguments only.
>
> **Q: Why quotes around "Biden" but not around 42?**
>
> A: Quotes signal TEXT. `"Biden"` is the literal characters B-i-d-e-n. Without quotes, R would look for a variable named Biden. Numbers don't need quotes because R recognizes them as numbers.
>
> **Q: Why type the variable name alone on a line?**
>
> A: That tells R to SHOW you what's inside. Assignment (`x <- 5`) is silent. Typing just `x` says "print x's contents."
>
> **Q: Why underscores in variable names?**
>
> A: Spaces aren't allowed. `approval rating` would be two separate things. `approval_rating` is one clear name.

What this code does: We create three variables: `approval_rating` stores a number (42), `president` stores text ("Biden"), and `is_democrat` stores a logical value (TRUE). The assignment operator `<-` places the value on the right into the variable on the left. When we type a variable name alone, R displays its contents.

Good variable names are descriptive and use underscores between words. Names like `voter_turnout` or `dem_vote_share` are much clearer than abbreviations like `vt` or `dvs`. Clear naming makes code readable and reduces errors.

2.2 Data Types

R has several fundamental data types, each corresponding to different kinds of information. Numeric values store numbers and correspond to interval or ratio measurement. Character values store text (always enclosed in quotes) and correspond to nominal categories. Logical values store TRUE or FALSE and represent binary conditions.

```
class(42)
```

```
[1] "numeric"
```

```
class("Democrat")
```

```
[1] "character"
```

```
class(TRUE)
```

```
[1] "logical"
```

What this code does: The `class()` function tells us what type of data we have. The number 42 is "numeric," the text "Democrat" is "character," and TRUE is "logical." Knowing data types matters because different operations apply to different types.

The data type determines what operations R will allow. We can add numbers but not text. Attempting to add "Texas" + "California" would produce an error, because addition is not defined for character data.

2.3 Vectors: Collections of Values

A vector is an ordered collection of values of the same type. Vectors are fundamental to R. Nearly everything in R is built from vectors, and most operations work on entire vectors at once.

```
approval <- c(42, 44, 41, 39, 43)
states <- c("NH", "PA", "MI", "WI", "AZ")
approval
```

```
[1] 42 44 41 39 43
```

```
states
```

```
[1] "NH" "PA" "MI" "WI" "AZ"
```

What this code does: The `c()` function (for "combine") creates vectors by joining values together. We create a numeric vector of approval ratings and a character vector of state abbreviations. Each vector stores multiple values in a specific order.

All elements of a vector must be the same type. If you try to mix types, R will convert them to a common type, usually character. This automatic conversion (called coercion) can cause unexpected problems if you are not aware of it.

```
mixed <- c(42, "Democrat", TRUE)
mixed
```

```
[1] "42"       "Democrat" "TRUE"
```

```
class(mixed)
```

```
[1] "character"
```

What this code does: When we combine a number (42), text ("Democrat"), and logical (TRUE) in one vector, R converts everything to character. Notice that 42 and TRUE now appear in quotes. This is why we need separate vectors for different types of data.

2.4 Data Frames: Rectangular Data

A data frame combines multiple vectors into a rectangular dataset, like a spreadsheet. Rows are observations (individual voters, countries, elections) and columns are variables (age, party, turnout). Data frames are the primary structure for storing political data.

```
treaties <- tibble(
  treaty = c("Paris Agreement", "NAFTA", "NATO", "Kyoto Protocol"),
  year_signed = c(2015, 1994, 1949, 1997),
  us_member = c(FALSE, TRUE, TRUE, FALSE),
  member_count = c(195, 3, 31, 192)
)
treaties
```

```
# A tibble: 4 x 4
  treaty          year_signed us_member member_count
  <chr>                 <dbl> <lgl>            <dbl>
1 Paris Agreement        2015 FALSE              195
2 NAFTA                  1994 TRUE                 3
3 NATO                   1949 TRUE                31
4 Kyoto Protocol         1997 FALSE              192
```

Reading Tibble Output Step-by-Step

When you print a tibble, R displays helpful information:

Line 1: Dimensions

- `# A tibble: 4 × 4` tells you there are 4 rows and 4 columns

Line 2: Column names and types

- `<chr>` = character (text)
- `<dbl>` = double (numeric with decimals)
- `<int>` = integer (whole numbers)
- `<lgl>` = logical (TRUE/FALSE)

Rows: The data

- Each numbered row is one observation
- Values align under their column headers

Why this matters: The column types tell you what operations are valid. You can do math with `<dbl>` and `<int>`, but not with `<chr>`. If a column you expect to be numeric shows as `<chr>`, you have a data problem.

What this code does: The `tibble()` function creates a data frame with four columns: treaty names (character), years signed (numeric), US membership status (logical), and member counts (numeric). Each row represents one international treaty. The data frame keeps these vectors aligned so row 1 of each column refers to the same treaty.

3 Working with Political Data

Once we have data stored in R, we need tools to explore and summarize it. The most basic descriptions involve measures of central tendency (where is the center of the data?) and simple counts.

3.1 Summary Statistics

```
approval <- c(42, 44, 41, 39, 43, 45, 44, 42, 40, 38, 41, 43)
mean(approval)
```

```
[1] 41.83333
```

```
median(approval)
```

```
[1] 42
```

What this code does: We calculate the mean (arithmetic average) and median (middle value) of approval ratings over 12 months. The mean of 41.83 and median of 42 are close, suggesting a roughly symmetric distribution without extreme outliers pulling the average in one direction.

The mean and median can differ substantially when data are skewed or contain outliers. Consider this example with international aid budgets, where one country's contribution dwarfs the others.

```r
aid_millions <- c(150, 200, 180, 175, 190, 210, 5000)
mean(aid_millions)
```

```
[1] 872.1429
```

```r
median(aid_millions)
```

```
[1] 190
```

> **What this code does:** The mean ($872 million) is pulled up dramatically by one extremely large value (presumably a major donor like the US). The median ($190 million) better represents the "typical" country's contribution. This is why median income is often more informative than mean income when discussing economic well-being.

3.2 Other Useful Functions

R provides many functions for exploring data. The `summary()` function gives a quick overview of a vector's distribution. The `length()` function counts elements. The `sum()`, `min()`, and `max()` functions provide other basic statistics.

```r
summary(approval)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  38.00   40.75   42.00   41.83   43.25   45.00
```

```r
length(approval)
```

```
[1] 12
```

```r
sum(approval)
```

```
[1] 502
```

> **What this code does:** The `summary()` function shows six key statistics: minimum, first quartile, median, mean, third quartile, and maximum. The `length()` tells us we have 12 observations. The `sum()` adds all values together (502 total approval points across 12 months).

3.3 Handling Missing Data

Real political data often has missing values, represented in R as `NA` (Not Available). Survey respondents skip questions, countries fail to report statistics, and records get lost. Understanding how R handles missing data is essential.

```r
responses <- c(4, 5, NA, 3, 4, NA, 5)
mean(responses)
```

```
[1] NA
```

```r
mean(responses, na.rm = TRUE)
```

```
[1] 4.2
```

```r
sum(is.na(responses))
```

```
[1] 2
```

What this code does: When data contains NA values, `mean()` returns NA by default (R cannot calculate an average that includes unknown values). Adding `na.rm = TRUE` tells R to remove missing values before calculating. The `is.na()` function identifies which values are missing, and `sum()` counts them (2 missing values here).

4 Common Mistakes and How to Avoid Them

Mistake 1: Confusing Correlation with Causation

This is the most important lesson in the course and the error we will return to repeatedly. Just because two things occur together does not mean one causes the other. Countries with more telephones per capita have higher life expectancy. Does owning a telephone make you live longer? Of course not. Wealthy countries have both more telephones and better healthcare. The correlation is real, but the causal interpretation is nonsense. We will explore this concept in depth, with detailed examples, in Chapter 4.

Mistake 2: Relying on Anecdotal Evidence

We remember stories, not statistics. A single dramatic example can seem more compelling than systematic evidence from thousands of observations. ”My neighbor switched parties after seeing that ad” is not evidence that the ad worked in general. It is a single data point that may not be representative of anything broader. Systematic evidence trumps anecdote every time.

Mistake 3: Ignoring Uncertainty

A poll shows Candidate A at 48% and Candidate B at 46%. Is A winning? Not necessarily. The margin of error matters. If the poll has a margin of error of plus or minus 3 percentage points, the two candidates are statistically tied. We will learn in later chapters how to quantify and communicate uncertainty properly.

5 R Functions Reference

Function	Purpose	Example
`c()`	Create vector	`c(1, 2, 3)`
`length()`	Count elements	`length(x)`
`sum()`	Add values	`sum(x, na.rm = TRUE)`
`mean()`	Calculate average	`mean(x, na.rm = TRUE)`
`median()`	Find middle value	`median(x)`
`min()`, `max()`	Find extremes	`min(x)`, `max(x)`
`summary()`	Quick statistics	`summary(x)`
`class()`	Check data type	`class(x)`
`is.na()`	Check for missing	`is.na(x)`
`tibble()`	Create data frame	`tibble(a = 1:3)`

Table 1: Core R Functions for Chapter 1

6 Practice Problems

Question 1

A researcher finds that countries with more international airports have higher rates of obesity. Does this mean airports cause obesity? Propose two alternative explanations for this correlation that do not involve a direct causal relationship between airports and weight gain.

Question 2

Consider this vector of congressional approval ratings: `c(18, 22, 15, 25, 19, NA, 21, 17)`. What value will `mean()` return by default? How would you calculate the mean excluding the missing value? How many missing values are in this vector?

Question 3

You create the following vector: `mixed <- c(42, "Republican", TRUE)`. What data type will `mixed` be, and why? What happens when you try to calculate `mean(mixed)`?

Question 4 (Computational)

Create a vector called `turnout` containing voter turnout rates for 6 countries: 67.5, 72.1, 58.3, 69.8, 71.2, 55.0. Calculate the mean, median, minimum, and maximum. Which measure of central tendency would you report if one country had a turnout of 95% and you wanted

to describe ”typical” turnout?

For Further Study

This chapter has introduced the foundational ideas behind quantitative political analysis. Students interested in the philosophy of causation should explore Judea Pearl’s accessible book *The Book of Why*, which explains causal thinking without heavy mathematics. For developing R skills further, the free online textbook *R for Data Science* at r4ds.hadley.nz provides excellent tutorials. For a deeper treatment of research design in political science, Kellstedt and Whitten’s *The Fundamentals of Political Science Research* is a standard reference.