Summary

Last modified: 2026-11-13 12:24:54 (PST)

> **AI-Assisted Authorship**
>
> This chapter was written with the assistance of GitHub Copilot, which expanded on outlined ideas and draft notes provided by the author. The content represents the author's perspective and has been reviewed for accuracy, but the detailed prose was generated through AI assistance.

This chapter lists the book's main rules as a checklist. Use the checklist when you revise a draft: read the draft once for each group of rules, and fix each passage that breaks a rule. Each rule is stated in a few words here. Follow its link for the reasons and the exceptions.

Terms and notation

- Define every technical term explicitly, before its first use, and follow the definition with examples and counterexamples.
- Give each mathematical statement an example that applies it.
- Prefer interval notation to a double inequality: $x \in (a, b)$ rather than $a < x < b$.
- Abstract each sub-expression you are not manipulating to a symbol, such as μ for $\mathbb{E}[X]$.
- Do one operation per step in a derivation, and justify each step.
- Give applying a definition its own step; never skip it.
- Show why you need a result before you derive it, and color each result where it is used.
- Put one equals sign on each line: split a chain such as $a = b = c$ into the rows of an aligned display.
- Distinguish a function from its value.

- Distinguish a random variable from its value: Y_i before the data, y_i after, and call each a "variable" or a "value" in the words too.
- Use the notation that matches a quantity's name, and keep the pairing the same everywhere.
- Write each symbol through a shared macro.
- Write out notational shorthands, such as the limits of a sum.
- Put words between formulas, and start a sentence with a word, not a symbol.
- Introduce a theorem with a complete sentence, and state it so that it stands alone.
- Spell out a number used as an adjective, and write the numeral for a number used as a name.

Claims and evidence

- Support every claim with a citation or with data.
- Do not substitute flippancy for support, such as calling a step "trivial".
- Cite a relevant source that is trustworthy.
- Copy exact wording with caution, even from your own earlier work.
- Cite a source once per passage.
- Link liberally to the sources and tools that you mention.
- Quote the original rather than a source that quotes it.
- Put the content first: name the topic, not its author, in headings and sentences.
- Date a claim that can change, such as what a software package supports.

Papers

- Write to clarify your thinking: write each question or idea down as it comes.
- Follow the standard structure of a research paper.
- Organize the Results section to fit the design of the study.
- Label variables in plain words in figures and tables.
- Use descriptive subsection headings.
- Avoid forward references: move the content ahead of the sentence that uses it.
- Lead with what is true: do not state a claim and then retract it.
- Put each link to a video next to the text that it supports.
- Follow the journal's conventions for a manuscript, and check the submission checklist before you submit.

Words

- Prefer plain words to Latin-derived ones where a plain word says the same thing.
- Avoid vague and metaphorical language.
- Follow an abstract statement with an example.
- Replace the word "thing" with the object, quantity, step, or idea that it stands for.
- Give a technical word one meaning.
- Minimize unnecessary jargon.
- Edit out clusters of clichéd words and sentence shapes, which make prose seem generic or machine-drafted.
- Say what something is before what it is not.

Writing for every reader

- Use literal language.
- Use common words, and define technical ones.
- Say when a common word has a technical meaning, such as "significant" or "bias".
- Spell out abbreviations, and do not clip a term, such as "cost" for "cost function".
- Write positive statements, and avoid double negatives.
- Rewrite noun strings and hidden verbs.
- Put one idea in each sentence.
- State every step and every point.
- Say what numbers and symbols mean.
- Use examples that every reader knows.
- Keep a predictable structure.
- Present ideas in more than one form.
- Test your writing with a reader.

Sentences

- Keep relative pronouns such as "that" and "which".
- Follow "this" or "that" with the noun that it refers to.
- Follow a quantifier such as "both" or "each" with its noun.
- Avoid garden-path sentences.
- Name the object of a relational noun.
- Lead with the subject.
- Start a sentence from what the reader already knows, and end it with the new information.
- Make every word tell: cut words that carry no meaning, but keep detail that does.
- Limit complex sentence structures.

- Do not rule out readings that no reader would make: when you keep a contrast, say who holds the other reading and cite a source.
- Write about the subject, not about the text.
- Put a list of three or more phrases in bullet points.
- Write to communicate, not to impress: use the plainest form that still carries the full meaning.

Teaching materials

- Build the lecture from exercises and solutions.
- Ask the question before giving the definition.
- Present a derivation as an exercise and solution, then the theorem and its proof.
- Give each defined term its own div.
- State the notation and its alternatives.
- Lead with the simplest correct form of a definition.
- Reuse one written version of each result.
- Use real data and compute every number.
- Pair exploratory data analysis with plots.
- Show each idea applied and in a figure.
- Fade worked examples into exercises within a unit.
- Leave a step to the reader only as an exercise.
- Refute common misconceptions where the concept is defined or first used.
- Revisit earlier units in later exercises.
- Write labs in both R and Python that run on any computer.
- Make figures accessible.
- Structure slides for the reader.
- Credit another course's material in a collapsed "Source" callout, not in running prose.
- Do not formally acknowledge material borrowed from your own work, and remove any existing credit to your own work, but keep what that work itself credits.

Software text

- Say what changed and why in a pull request or commit message.
- Comment why, not what.
- Write release notes by their effect on the user.

Thinking

- Write the question down first.

- Send a written question that gives the reader the context to answer it.
- Draft the plan before you run the analysis.