

Scientific writing

Lecture 2

16.10.26

References

- cite
- it is better to cite more than less
- cite precisely: not just the book
- you should know what you cite
- do not make bibliography by hand, there are tools for that (bibliography, biber, natbib)
- make bibliography when reading

References

- cite
- it is better to cite more than less
- cite precisely: not just the book
- you should know what you cite
- do not make bibliography by hand, there are tools for that (bibliography, biber, natbib)
- make bibliography when reading

Did you know?

Citations used to be footnotes.
There are many consequences of it.

References

- cite
- it is better to cite more than less
- cite precisely: not just the book
- you should know what you cite
- do not make bibliography by hand, there are tools for that (bibliography, biber, natbib)
- make bibliography when reading

Did you know?

Citations used to be footnotes.

There are many consequences of it.

Citation is not a noun.

- × see [2]
- ✓ see recent survey [2].
- ✓ see Knuth et al. [2]. (there are tools for rendering lastnames)

[Bad] There are many $\mathcal{O}(n \log n)$ sorting algorithms.

[OK] There are many $\mathcal{O}(n \log n)$ sorting algorithms [1, 2, 3].

[Good] There are many $\mathcal{O}(n \log n)$ sorting algorithms, which is discussed already in standard textbooks [7, Chapter 3], [8, Chapter 7].

[Over the top but OK] There are many $\mathcal{O}(n \log n)$ sorting algorithms [1, 2, 3, 4, 5, 6, 7], and discussion in standard textbooks [7, Chapter 3], [8, Chapter 7].

[Over the top] There are many $\mathcal{O}(n \log n)$ sorting algorithms [1, 2, 3, 4, 5, 6, 7], and discussion in standard textbooks [7, Chapter 3], [8, Chapter 7]. see also a recent survey that analyses in detail those algorithms under different measures [8] and recent developments in quicksort [9].

[Bad] There are many $\mathcal{O}(n \log n)$ sorting algorithms.

[OK] There are many $\mathcal{O}(n \log n)$ sorting algorithms [1, 2, 3].

[Good] There are many $\mathcal{O}(n \log n)$ sorting algorithms, which is discussed already in standard textbooks [7, Chapter 3], [8, Chapter 7].

[Over the top but OK] There are many $\mathcal{O}(n \log n)$ sorting algorithms [1, 2, 3, 4, 5, 6, 7], and discussion in standard textbooks [7, Chapter 3], [8, Chapter 7].

[Over the top] There are many $\mathcal{O}(n \log n)$ sorting algorithms [1, 2, 3, 4, 5, 6, 7], and discussion in standard textbooks [7, Chapter 3], [8, Chapter 7]. see also a recent survey that analyses in detail those algorithms under different measures [8] and recent developments in quicksort [9].

[Good] The number of both swaps comparisons were analyzed separately for quicksort [4], different implementations optimize those two measures, both bounded by $\mathcal{O}(n \log n)$ [4, Theorem 2, Theorem 5].

The purpose of notation

Good notation helps.

The purpose of notation

Good notation helps.

Bad notation distracts.

The purpose of notation

Good notation helps.

Bad notation distracts.

There used to be special notation for permutations, combinations, combinations with repetitions, choices, choices with repetitions. . .

The purpose of notation

Good notation helps.

Bad notation distracts.

There used to be special notation for permutations, combinations, combinations with repetitions, choices, choices with repetitions. . .

Difficult to work with, difficult connections

Replacing this all with $\binom{n}{k}$ made computation possible.

- Plan the notation in advance.

- Plan the notation in advance.
- Define macros and commands: changes may be needed.

- Plan the notation in advance.
- Define macros and commands:
changes may be needed.
- Reuse previous and established notation.

- Plan the notation in advance.
- Define macros and commands:
changes may be needed.
- Reuse previous and established notation.
- Use notation to expose connections.

- Plan the notation in advance.
- Define macros and commands:
changes may be needed.
- Reuse previous and established notation.
- Use notation to expose connections.
- Give different objects different symbols.

- Plan the notation in advance.
- Define macros and commands: changes may be needed.
- Reuse previous and established notation.
- Use notation to expose connections.
- Give different objects different symbols.
- Avoid unnecessary indices.

- Plan the notation in advance.
- Define macros and commands: changes may be needed.
- Reuse previous and established notation.
- Use notation to expose connections.
- Give different objects different symbols.
- Avoid unnecessary indices.
- Avoid clashes.

- Plan the notation in advance.
- Define macros and commands: changes may be needed.
- Reuse previous and established notation.
- Use notation to expose connections.
- Give different objects different symbols.
- Avoid unnecessary indices.
- Avoid clashes.
- Avoid visual overloading, especially boldface.

- Plan the notation in advance.
- Define macros and commands: changes may be needed.
- Reuse previous and established notation.
- Use notation to expose connections.
- Give different objects different symbols.
- Avoid unnecessary indices.
- Avoid clashes.
- Avoid visual overloading, especially boldface.
- Be prepared to revise the notation.

- Plan the notation in advance.
- Define macros and commands: changes may be needed.
- Reuse previous and established notation.
- Use notation to expose connections.
- Give different objects different symbols.
- Avoid unnecessary indices.
- Avoid clashes.
- Avoid visual overloading, especially boldface.
- Be prepared to revise the notation.

- Plan the notation in advance.
- Define macros and commands: changes may be needed.
- Reuse previous and established notation.
- Use notation to expose connections.
- Give different objects different symbols.
- Avoid unnecessary indices.
- Avoid clashes.
- Avoid visual overloading, especially boldface.
- Be prepared to revise the notation.

A useful test

Can a reader reconstruct the meaning of a symbol without searching several pages backwards?

Use commands and macros

A recurring mathematical object

Repeated manually

$\mathcal{G}(V,E)$

Defined once

`\newcommand{\graph}{\mathcal{G}}`

Use commands and macros

A recurring mathematical object

Repeated manually

```
\mathcal{G}(V,E)
```

Defined once

```
\newcommand{\graph}{\mathcal{G}}
```

Example 2: an operator

Ad hoc

```
$OPT(I)$
```

Semantic command

```
\DeclareMathOperator{\OPT}{OPT}
```

3. Reuse established notation

3. Reuse established notation

Example 1: graph size

Avoid

$$|V| = p, \quad |E| = q$$

Prefer

$$|V| = n, \quad |E| = m$$

3. Reuse established notation

Example 1: graph size

Avoid

$$|V| = p, \quad |E| = q$$

Prefer

$$|V| = n, \quad |E| = m$$

Example 2: string length

Avoid

$$\ell(w) = k$$

Prefer

$$|w| = n$$

Use notation to expose connections

Use notation to expose connections

A graph and a subgraph

Avoid

$$G = (V, E), \quad H = (W, F)$$

Prefer

$$G = (V, E), \quad G' = (V', E')$$

Use notation to expose connections

A graph and a subgraph

Avoid

Prefer

$$G = (V, E), \quad H = (W, F)$$

$$G = (V, E), \quad G' = (V', E')$$

Successive elements

Avoid

Prefer

$$x, \quad y, \quad z$$

$$x^{(0)}, \quad x^{(1)}, \quad x^{(2)}$$

Preserve order/meaning

Avoid

Prefer

$$A, B, C, n = |A|, m = |B|, k = |C|$$

$$\begin{aligned} k &= |A|, \ell = |B|, m = |C| \\ a &= |A|, b = |B|, c = |C| \\ n_A &= |A|, n_B = |B|, n_C = |C| \end{aligned}$$

Use notation to expose connections

A graph and a subgraph

Avoid

Prefer

$$G = (V, E), \quad H = (W, F)$$

$$G = (V, E), \quad G' = (V', E')$$

Successive elements

Avoid

Prefer

$$x, \quad y, \quad z$$

$$x^{(0)}, \quad x^{(1)}, \quad x^{(2)}$$

Preserve order/meaning

Avoid

Prefer

$$A, B, C, n = |A|, m = |B|, k = |C|$$

$$\begin{aligned} k &= |A|, \ell = |B|, m = |C| \\ a &= |A|, b = |B|, c = |C| \\ n_A &= |A|, n_B = |B|, n_C = |C| \end{aligned}$$

Elements of elements of ...

$\mathbb{V}$: vector space $V \subseteq \mathbb{V}$: set of vectors $\vec{v} = (v_1, \dots, v_k) \in \mathbb{V}$: vector v_i : coordinate

Use different symbols for different things

m, n : numbers

i, j : indices

p, q : dividers

x, y : variables

Respect standard conventions

Conventional notation gives the reader information.

Respect standard conventions

Conventional notation gives the reader information.

Example 1: indices and constants

Avoid

$$a = \sum_{x=1}^n b_x$$

Prefer

$$A = \sum_{i=1}^n b_i$$

Respect standard conventions

Conventional notation gives the reader information.

Example 1: indices and constants

Avoid

$$a = \sum_{x=1}^n b_x$$

Prefer

$$A = \sum_{i=1}^n b_i$$

Example 2: sets and elements

Avoid

$$x \in y$$

Prefer

$$x \in X$$

Avoid boldface

Bold symbols are easy to miss, hard to distinguish in speech, and fragile under copying or conversion.

Avoid boldface

Bold symbols are easy to miss, hard to distinguish in speech, and fragile under copying or conversion.

Vectors and coordinates

Avoid

$\mathbf{x}$, $\mathbf{x}$

Prefer

x , $\vec{x}$

Avoid boldface

Bold symbols are easy to miss, hard to distinguish in speech, and fragile under copying or conversion.

Vectors and coordinates

Avoid

x , $\mathbf{x}$

Prefer

x , $\vec{x}$

Example 2: word versus set of words

Avoid

w , $\mathbf{w}$

Prefer

w , $\mathcal{W}$

Avoid boldface

Bold symbols are easy to miss, hard to distinguish in speech, and fragile under copying or conversion.

Vectors and coordinates

Avoid

x , $\mathbf{x}$

Prefer

x , $\vec{x}$

Example 2: word versus set of words

Avoid

w , $\mathbf{w}$

Prefer

w , W

Example 3: matrix versus entry

Avoid

a , $\mathbf{a}$

Prefer

A , a_{ij}

Excessive indexing

Vectors

Avoid

$\vec{v}_1, \vec{v}_2, \vec{v}_3; v_{1,i}, v_{2,j}, v_{3,k};$

Prefer

$\vec{v}, \vec{v}', \vec{v}''; v_i, v'_j, v''_k$

Excessive indexing

Vectors

Avoid

Prefer

$\vec{v}_1, \vec{v}_2, \vec{v}_3; v_{1,i}, v_{2,j}, v_{3,k};$

$\vec{v}, \vec{v}', \vec{v}''; v_i, v'_j, v''_k$

Example 2: local context already fixes the graph

Avoid

Prefer

$d_{G,i}(v)$

$d_i(v)$

Excessive indexing

Vectors

Avoid

Prefer

$$\vec{v}_1, \vec{v}_2, \vec{v}_3; v_{1,i}, v_{2,j}, v_{3,k};$$

$$\vec{v}, \vec{v}', \vec{v}''; v_i, v'_j, v''_k$$

Example 2: local context already fixes the graph

Avoid

Prefer

$$d_{G,i}(v)$$

$$d_i(v)$$

Example 3: name the object instead

Avoid

Prefer

$$S_{i,j,k,\ell}$$

$$B_{i,j}$$

Typography

There is reason to use WYSINWYG.

There is some science/engineering in typography.

Typography

There is reason to use WYSINWYG.

There is some science/engineering in typography.

Some things should be corrected by hand

- widows and orphans
- splitting of paragraphs
- ...

Focus on what distract the Reader

Typography

There is reason to use WYSINWYG.

There is some science/engineering in typography.

Some things should be corrected by hand

- widows and orphans
- splitting of paragraphs
- ...

Focus on what distract the Reader

Some simple rules:

- use boldface to make something visible at first glance
- use italics or slanted (`\emph`) to stress something in text (while reading)
- use semantic and not syntactic tags
- do not use underline

Typography

There is reason to use WYSINWYG.

There is some science/engineering in typography.

Some things should be corrected by hand

- widows and orphans
- splitting of paragraphs
- ...

Focus on what distract the Reader

Some simple rules:

- use boldface to make something visible at first glance
- use italics or slanted (`\emph`) to stress something in text (while reading)
- use semantic and not syntactic tags
- do not use underline

Do not touch if you do not understand.

Guidelines for Writing Equations

Mathematical notation provides precision at key moments—it should not be overused. This is also a piece of writing.

Guidelines for Writing Equations

Mathematical notation provides precision at key moments—it should not be overused. This is also a piece of writing.

- **Do not begin a sentence with a symbol:**

- $\times$ $f(x)$ is an increasing function.
- $\checkmark$ The function $f(x)$ is increasing.

Guidelines for Writing Equations

Mathematical notation provides precision at key moments—it should not be overused. This is also a piece of writing.

- **Do not begin a sentence with a symbol:**

- $\times$ $f(x)$ is an increasing function.
- $\checkmark$ The function $f(x)$ is increasing.

- **Separate symbols with text:**

- $\times$ For every $x \in A$, $x^2 > 0$.
- $\checkmark$ For every element $x \in A$, the inequality $x^2 > 0$ holds.

Guidelines for Writing Equations

Mathematical notation provides precision at key moments—it should not be overused. This is also a piece of writing.

- **Do not begin a sentence with a symbol:**

- $\times$ $f(x)$ is an increasing function.
- $\checkmark$ The function $f(x)$ is increasing.

- **Separate symbols with text:**

- $\times$ For every $x \in A$, $x^2 > 0$.
- $\checkmark$ For every element $x \in A$, the inequality $x^2 > 0$ holds.

- **Numbering:** Number only those equations that are referenced later in the text.

You can use your own tags (do not overdo), subtags, etc.

- This can help: Invariants I1, I2, ...
- Properties P1, P2, ...
- it looks bad to have equation (1) and nothing more
- subtags (3a), (3b),

L^AT_EX is a powerful tool—you can hurt yourself with it.

If you do not know what it does or do not understand it, do not touch it.

L^AT_EX is a powerful tool—you can hurt yourself with it.

If you do not know what it does or do not understand it, do not touch it.

Formatting

Good habits in source code $\Rightarrow$ easier document management.

- $\times \times$ one line with automatic line wrapping
- $\times$ one continuous line
- $\checkmark$ line breaks at semantic boundaries (end of a sentence, subordinate clause, thought, etc.)

There are tools for integrating with git and showing changes (latexdiff). They assume such formatting.

L^AT_EX is a powerful tool—you can hurt yourself with it.

If you do not know what it does or do not understand it, do not touch it.

Formatting

Good habits in source code $\Rightarrow$ easier document management.

- $\times \times$ one line with automatic line wrapping
- $\times$ one continuous line
- $\checkmark$ line breaks at semantic boundaries (end of a sentence, subordinate clause, thought, etc.)

There are tools for integrating with git and showing changes (latexdiff). They assume such formatting.

- $\checkmark$ indentation
- macros and commands: do not overuse them
- $\times$ copying all commands from a previous paper

Presentation Formats

- visualizations: very helpful for building intuition
- tables are essential
- plots are very useful
 - make plots/graphs large
 - do not rely on color or test for B&W

Presentation of Data and Results

Presentation Formats

- visualizations: very helpful for building intuition
- tables are essential
- plots are very useful
 - make plots/graphs large
 - do not rely on color or test for B&W

Tools

Tools are available for this.

Principle

The entire process of obtaining the results must be verifiable and reproducible.

Principle

The entire process of obtaining the results must be verifiable and reproducible.

- data
- computed parameters
- implementations

Principle

The entire process of obtaining the results must be verifiable and reproducible.

- data
- computed parameters
- implementations

Availability

It must be accessible and it must work.

This also applies to the solutions being compared.

Reproducibility: Data and Results

Principle

The entire process of obtaining the results must be verifiable and reproducible.

- data
- computed parameters
- implementations

Availability

It must be accessible and it must work.

This also applies to the solutions being compared.

Results

There should be a metric that allows us to assess the “quality” of a solution.

Artifact

- a bootable artifact with all libraries, verified
- ...
- compiled binaries

Artifact

- a bootable artifact with all libraries, verified
- ...
- compiled binaries

User manual

Supply user manual. With examples.

Who is your audience?

Think about user experience.

Work environment

Writing takes time: make sure that you are not interrupted.

Easy reading is damn hard writing.

Work environment

Writing takes time: make sure that you are not interrupted.

Easy reading is damn hard writing.

In the end you will type it, some things might be better done offline.

Work environment

Writing takes time: make sure that you are not interrupted.

Easy reading is damn hard writing.

In the end you will type it, some things might be better done offline.

D. Knuth wrote papers in hand and then typed them. . .

Work environment

Writing takes time: make sure that you are not interrupted.

Easy reading is damn hard writing.

In the end you will type it, some things might be better done offline.

D. Knuth wrote papers in hand and then typed them. . .

- complex tables
- figures (scan and ask LLM)
- complex equations (there are tools that translate hand-written math to $\text{\LaTeX}$)
- work outline. . .

Work environment

Writing takes time: make sure that you are not interrupted.

Easy reading is damn hard writing.

In the end you will type it, some things might be better done offline.

D. Knuth wrote papers in hand and then typed them. . .

- complex tables
- figures (scan and ask LLM)
- complex equations (there are tools that translate hand-written math to $\text{\LaTeX}$)
- work outline. . .

✓ There will be corrections, iterations.

× First batch, a good batch.

Example: chess engine

Research topic

develop and evaluate chess engine

uses search trees + heuristics: evaluates positions using Neural Network

learns through self play

Example: chess engine

Research topic

develop and evaluate chess engine

uses search trees + heuristics: evaluates positions using Neural Network

learns through self play

Question

Can we train a chess engine through self-play on PC-class computer.

Example: chess engine

Research topic

develop and evaluate chess engine

uses search trees + heuristics: evaluates positions using Neural Network

learns through self play

Question

Can we train a chess engine through self-play on PC-class computer.

How to evaluate: compare with existing engines? Compare progress?

Example: chess engine

Research topic

develop and evaluate chess engine

uses search trees + heuristics: evaluates positions using Neural Network

learns through self play

Question

Can we train a chess engine through self-play on PC-class computer.

How to evaluate: compare with existing engines? Compare progress?

Context

Chess: history of chess engines

different approaches (tree search vs NN, heuristics for evaluating a position vs NN)

many chess-specific details on NN (small updates, 2 players, ResNet)

Tree search

general approach, including heuristics
chess-specific heuristics

Tree search

general approach, including heuristics
chess-specific heuristics

Some specifics

Coding matters
NNUE: special NN (many zeroes, small updates)
chess-specific heuristics

- ➊ Intro
- ➋ Chess engines history
 - ➊ classic
 - ➋ pinnacles (DeepMind, Stockfish)
 - ➌ AlphaZero
 - ➊ selfplay
 - ➋ why (science vs practice)
 - ➌ initial controversies?
 - ➍ state of the art (mixed systems)
 - ➎ some comments on Stockfish development

- ➊ Intro
- ➋ Chess engines history
 - ➊ classic
 - ➋ pinnacles (DeepMind, Stockfish)
 - ➌ AlphaZero
 - ➊ selfplay
 - ➋ why (science vs practice)
 - ➌ initial controversies?
 - ➍ state of the art (mixed systems)
 - ➎ some comments on Stockfish development
- ➌ Tree search (in general); general heuristics

- ➊ Intro
- ➋ Chess engines history
 - ➊ classic
 - ➋ pinnacles (DeepMind, Stockfish)
 - ➌ AlphaZero
 - ➊ selfplay
 - ➋ why (science vs practice)
 - ➌ initial controversies?
 - ➍ state of the art (mixed systems)
 - ➎ some comments on Stockfish development
- ➌ Tree search (in general); general heuristics
- ➍ Chess specific heuristics
 - ➊ books of openings
 - ➋ closures
 - ➌ tree specific heuristics

- 1 Intro
- 2 Chess engines history
 - 1 classic
 - 2 pinnacles (DeepMind, Stockfish)
 - 3 AlphaZero
 - 1 selfplay
 - 2 why (science vs practice)
 - 3 initial controversies?
 - 4 state of the art (mixed systems)
 - 5 some comments on Stockfish development
- 3 Tree search (in general); general heuristics
- 4 Chess specific heuristics
 - 1 books of openings
 - 2 closures
 - 3 tree specific heuristics
- 5 NN in chess
 - 1 ResNet
 - 2 NNUE
 - 3 describe coding

- 1 Intro
- 2 Chess engines history
 - 1 classic
 - 2 pinnacles (DeepMind, Stockfish)
 - 3 AlphaZero
 - 1 selfplay
 - 2 why (science vs practice)
 - 3 initial controversies?
 - 4 state of the art (mixed systems)
 - 5 some comments on Stockfish development
- 3 Tree search (in general); general heuristics
- 4 Chess specific heuristics
 - 1 books of openings
 - 2 closures
 - 3 tree specific heuristics
- 5 NN in chess
 - 1 ResNet
 - 2 NNUE
 - 3 describe coding
- 6 Architecture of the engine + learning
 - 1 tree search + NNUE evaluation
 - 2 self-play, NNUE updates

- 1 Intro
- 2 Chess engines history
 - 1 classic
 - 2 pinnacles (DeepMind, Stockfish)
 - 3 AlphaZero
 - 1 selfplay
 - 2 why (science vs practice)
 - 3 initial controversies?
 - 4 state of the art (mixed systems)
 - 5 some comments on Stockfish development
- 3 Tree search (in general); general heuristics
- 4 Chess specific heuristics
 - 1 books of openings
 - 2 closures
 - 3 tree specific heuristics
- 5 NN in chess
 - 1 ResNet
 - 2 NNUE
 - 3 describe coding
- 6 Architecture of the engine + learning
 - 1 tree search + NNUE evaluation
 - 2 self-play, NNUE updates
- 7 Experiments

7 Experiments

- 1 some problems

8 Further heuristics

9 Evaluation

- 1 progress measured (games with previous versions)

7 Experiments

- 1 some problems

8 Further heuristics

9 Evaluation

- 1 progress measured (games with previous versions)

- This is authors decision
- OK for thesis
- in scientific paper: describe this together, in architecture?

7 Experiments

- 1 some problems

8 Further heuristics

9 Evaluation

- 1 progress measured (games with previous versions)

- This is authors decision
- OK for thesis
- in scientific paper: describe this together, in architecture?

Plot twisters are too popular in crime nowadays.