A TUTORIAL: WRITING, TOOLS, PAPERS AND CODE

Ziheng Chen[1]
[1] University of Trento
ziheng_ch@163.com

CONTENTS

ABSTRACT

This tutorial is designed for early-stage students aiming to publish in top-tier AI conferences and journals. It covers paper writing, submission, rebuttal, useful tools & tips, and essential codes & papers (for Riemannian deep learning). This document is continuously evolving. Readers are expected to be proficient in AI tools, such as ChatGPT, to assist with detailed aspects. If something is unclear to you, you can ask the ChatGPT for examples.

1 WRITING

The most important rule is to avoid using translators when reading papers. The best way to improve your writing is by reading academic papers directly. Only use translators when you truly cannot understand certain sentences. Over-reliance on translation tools will negatively impact your writing. For a concise introduction to LaTeX (approximately 40 minutes), refer to *The Not So Short Introduction to LaTeX*[1].

[1] https://tobi.oetiker.ch/lshort/lshort.pdf

A key indicator of a well-written paper is that it looks like top-tier papers, especially those related to your paper.

1.1 How to start with LaTeX

The best way is to follow a mature LaTeX paper. A good library is Arxiv, where each paper comes with its latex source code. You can find every interesting style shown in the Arxiv paper. A good tool is open-in-overleaf.

1.2 How to organize a paper

A good tutorial, https://github.com/hzwer/WritingAIPaper.

Do not place all the LaTeX code in a single file, as this can become difficult to manage when changing the template. Instead, structure your document by separating sections into different files, like this latex file. Refer to the following LaTeX source:

- Chen et al. (2024b) for a paper with an appendix.
- Chen et al. (2024a) for a paper with a separate supplement. Some conferences require the supplement to be uploaded separately.

1.3 Organizing References

Properly formatting references ensures consistency and professionalism in academic writing. Below are some essential guidelines for organizing references in LaTeX.

- Proper capitalization in titles: Ensure that specific terms, abbreviations, and acronyms are capitalized correctly in BibTeX entries. Since BibTeX automatically converts titles to lowercase, use curly brackets {} to preserve uppercase letters for proper nouns. Example:

  ```
  title={A {Lie} Group Approach to {Riemannian} Batch Normalization}
  ```

  Without curly brackets, "Lie" and "Riemannian" might be incorrectly lowercased.
- Use correct entry types: Choose the appropriate BibTeX entry type for different types of publications:
  - `@article` → for journal papers.
  - `@inproceedings` → for conference papers.
  - `@book` → for books.
  - `@phdthesis` → for dissertations.
  - `@misc` → for online resources (e.g., preprints, arXiv papers).

  arXiv papers can be cited via `@misc` or `@article`.
- Ensure consistent reference style: All references should follow a uniform style, particularly in formatting URLs, abbreviations, and journal/conference names.
  - URLs: If using URLs in references, ensure a consistent style, e.g., always using `url`.
  - Abbreviations: If abbreviating journal names (e.g., IEEE Transactions on Pattern Analysis and Machine Intelligence → IEEE TPAMI), ensure consistency across all references.
  - Page numbers, volume, and issue numbers: Make sure these details are consistently included or omitted across all references.

  Example of a journal article reference:

  ```
  @article{chen2024adaptive,
    title={Adaptive Log-{Euclidean} Metrics for {SPD} Matrix Learning},
    author={Chen, Ziheng and Song, Yue and Xu, Tianyang and Huang, Zhiwu and
    journal={IEEE TIP},
    year={2024},
  }
  ```

Example of a conference paper reference:

```
@inproceedings{chen2024liebn,
    title={A {Lie} Group Approach to {Riemannian} Batch Normalization},
    author={Ziheng Chen and Yue Song and Yunmei Liu and Nicu Sebe},
    booktitle={ICLR},
    year={2024}
}
```

Example of a reference with a URL:

```
@misc{chen2024liebn,
    title={A {Lie} Group Approach to {Riemannian} Batch Normalization},
    author={Ziheng Chen and Yue Song and Yunmei Liu and Nicu Sebe},
    booktitle={ICLR},
    url={https://openreview.net/forum?id=xxxxxx},
    year={2024}
}
```

- Check consistency with citations in the text: When citing a reference, use an appropriate citation command:
  - `\cite{key}` or `\citet{key}` for textual citations: Chen et al. (2024b) proposed this approach.
  - `\citep{key}` for parenthetical citations: This approach has been studied extensively (Chen et al., 2024b).

1.4 Equation Formatting Guidelines

Proper equation formatting is essential for clarity and readability in academic writing. Below are key guidelines to ensure consistency and proper indexing of equations.

- **Every equation should have an index.** Avoid using `equation*` whenever possible, as it prevents automatic equation numbering, making references more difficult. Instead of:

```
\begin{equation*}
E = mc^2
\end{equation*}
```

Use:

```
\begin{equation}
E = mc^2
\end{equation}
```

- **Difference between `aligned` and `align`:** Understanding when to use `aligned` vs. `align` helps maintain correct equation indexing:
  - Use `aligned` inside a single equation when you want to break an equation across multiple lines but retain one equation number. Example:

```
\begin{equation}
\begin{aligned}
a &= b + c + d + e + f \\
&\quad + g + h + i + j.
\end{aligned}
\end{equation}
```

  - Use `align` when writing multiple equations, each requiring a separate equation number. Example:

```
\begin{align}
a &= b + c + d, \\
x &= y + z.
\end{align}
```

Pay attention to the difference in **punctuation** between the two!

- **Use the `\operatorname` command properly.** Mathematical operators should be formatted correctly to distinguish them from variables. Use

```
\operatorname{Linear}(x), \quad \log(x)
```

Instead of:

```
Linear(x), \quad log(x)
```

- **notag.** Using `\notag` suppresses numbering for intermediate steps when necessary:

```
\begin{align}
f(x) &= a + b + c + d + e + f + g + h + i + j \notag \\
&\quad + k + l + m + n.
\end{align}
```

- **Cite theoretical results correctly.** When citing specific sections of a paper, use

```
\cite[Sec.xx]{reference}
```

- **Resizing equations using `\fontsize` or `\small`:** When equations are too long and resizing is necessary, use

```
{\fontsize{8}{5}\selectfont
\begin{equation}
...
\end{equation}
}
```

Alternatively, for slightly smaller equations, use:

```
{\small
\begin{equation}
...
\end{equation}
}
```

However, use resizing only when absolutely necessary to maintain readability.

1.5 Managing Abbreviations and Symbols

In case many abbreviations (such as `CNN`, `RNN`) and mathematical symbols (like $\mathcal{M}$, $\mathbb{R}^n$) are used, it could improve readability to index each of them.

The `glossaries-extra` package is a powerful tool to manage such terms consistently. It supports:

- Automatic generation of lists: acronyms, mathematical symbols, and general glossaries.
- Custom display styles: e.g., showing "Full Term (Abbr)" on first use, and only the abbreviation thereafter.
- Sorted entries and multiple glossary types (acronyms, notations, etc.)
- Hyperlinked entries and table of contents integration.

We provide a complete Overleaf example project demonstrating how to define and use acronyms and mathematical notations with `glossaries-extra`, including styling, sorting, and glossary printing. Please check Example Project for the example.

1.6 Useful Tips

In this section, we introduce several useful LaTeX tips that can improve the clarity, consistency, and efficiency of your academic writing.

- **Useful package:** Use `cleveref` for clever referencing. Instead of manually specifying the type of reference (`Figure \ref {xxx}`), the `cleveref` package automatically detects the type. Example usage:

  ```
  \usepackage{cleveref}
  ...
  \cref{fig:example, eq:example}
  ```

  This will output references like "Fig. 1" or "Eq. (2)" without explicitly writing "Figure" or "Equation." You can also customize the names of different reference types using `\crefname` (for lowercase) or `\Crefname` (for capitalized names). For example:

  ```
  \crefname{figure}{Fig.}{Figs.}
  \Crefname{figure}{Figure}{Figures}
  ```

  This tells LaTeX to use "Fig." when referring to a single figure, and "Figs." for plural. Similarly, "Figure" and "Figures" will be used when the reference begins a sentence.
- **Understand different citation commands:** LaTeX provides different citation commands depending on the desired citation style:
  - `\cite{key}`: Produces an in-text citation: Chen et al. (2025);
  - `\citep{key}`: Produces a parenthetical citation: (Chen et al., 2025);
  - `\citet{key}`: Produces an in-text citation: Chen et al. (2025).

  Choosing the correct format enhances readability and ensures consistency.
- **Using `\providecommand{}` for customized notations:** If you need to use a specific notation multiple times, defining it with `\providecommand` ensures consistency and allows for easy modification. Example:

  ```
  \providecommand{\loss}{\mathcal{L}}
  ```

  Then, you can use `\loss` throughout your document, ensuring uniform notation. Please check my previous paper for a plethora of examples (Chen et al., 2025).
- **Use `resizebox` cautiously:** The `resizebox` command is useful for shrinking tables or figures, but overuse can lead to unreadable content. Use it only when space constraints require it. Example:

  ```
  \resizebox{\textwidth}{!}{\begin{tabular}{c c c}
      A & B & C \\
      1 & 2 & 3
  \end{tabular}}
  ```

  Avoid using it if the table or figure remains legible without resizing.
- **Use `makecell` for better table formatting:** The `makecell` package allows multi-line cells in tables, improving readability. Example:

  ```
  \usepackage{makecell}
  \begin{tabular}{c|c}
      Column 1 & Column 2 \\ \hline
      A & \makecell{Line 1 \\ Line 2}
  \end{tabular}
  ```

- **Table formatting: Understanding `hline`, `midrule`, `bottomrule`, and `toprule`:** Tables can look more professional by using the appropriate horizontal rules:
  - `\hline`: A full horizontal line (can make the table look cluttered if overused).
  - `\toprule` and `\bottomrule` (from `booktabs` package): Used to create cleaner tables.
  - `\midrule`: Creates a thinner line between rows for better readability.

  Example of a well-formatted table:

```
\usepackage{booktabs}
\begin{tabular}{l c c}
    \toprule
    Method & Accuracy & F1-score \\
    \midrule
    Baseline & 85.3\% & 0.72 \\
    Proposed & 89.5\% & 0.79 \\
    \bottomrule
\end{tabular}
```

  Using `booktabs` produces aesthetically pleasing tables.

2 Submission

2.1 Submission

Carefully read the author instructions. Conference guidelines may change annually, so always verify the latest requirements. Avoid desk rejection, which is the last thing that should happen.

Many conferences explicitly prohibit certain LaTeX commands, especially those that alter formatting or spacing, such as `\vspace`, or custom margin settings. Violating these guidelines can lead to desk rejection.

Pay close attention to style requirements regarding:

- **Heading capitalization**: whether headings should be in sentence case or title case.
- **Figure and table captions**: some conferences require table captions to appear *above* the table and figure captions *below* the figure; others may specify differently.

Equally important is selecting the correct submission area and keyword tags. If you're uncertain, refer to related accepted papers and examine which subject areas and keywords they used, which often provides a reliable reference point.

2.2 Rebuttal

- Never apologize in a rebuttal. It is unnecessary. Focus on addressing the concerns of the reviewers rather than expressing regret.
- Useful resources are available online, such as https://zhuanlan.zhihu.com/p/602024489.
- Rebuttals on OpenReview can be written using HackMD, a Markdown-based editor similar to the LaTeX-based Overleaf. However, the LaTeX syntax in OpenReview may differ slightly from standard LaTeX, especially in the handling of whitespace. For example, to properly render superscripts and subscripts in math mode:
  - Correct: `$x ^{2}$` → x^2
  - Incorrect: `$x^{2}$` → may fail to render in HackMD

3 Tools and Tips

Most of these tools and resources can be easily found via Google.

3.1 Tools

- ChatGPT: It is as essential as breathing in the 21st century, especially for coding and writing. While it is a powerful tool, always verify its output. Mastering prompt engineering is key. For example, see gpt academic[2]. In paper writing, ChatGPT acts as an amplifier: if your writing is strong, it enhances it. Writing prompts[3] can also be useful.
- excel2latex: Converts Excel tables to LaTeX.
- Google PDF Reader (Chrome extension): Online academic PDF reader for quick reference checks.
- Grammarly (Chrome extension): Improves writing clarity and correctness.
- Mathpix Snipping Tool: Converts screenshots into LaTeX equations.
- TensorBoard: Visualizes training curves.
- HackMD: Online Markdown editor, similar to Overleaf. Learn to use team spaces—useful during rebuttals.
- Markdown Table Generator: Creates Markdown tables easily.
- Knowing the differences in LaTeX syntax between OpenReview and Markdown (useful for rebuttals).
- Open in OverLeaf: a chrome extension, where you can directly view the arxiv paper in Overleaf.

3.2 Tips

How to stay updated with the latest research:

- Scholar Inbox: ArXiv recommendation system.
- Follow top conferences, such as:
  - Computer Vision: ICCV, ECCV, CVPR
  - Machine Learning: ICML, ICLR, NeurIPS
  - ML Theory and Statistics: COLT, AISTATS, UAI
  - Manifold Theory: GSI (International Conference on Geometric Science of Information)
  - AAAI, IJCAI, MM, KDD
- Follow top journals, including:
  - Machine Learning and Vision: JMLR, TPAMI, IJCV
  - TNNLS, TIP
  - Matrix Theory: SIMAX (SIAM Journal on Matrix Analysis and Applications)

Gray indicates sub-top-tier conferences or journals.

4 Papers and Code

4.1 Papers

- Awesome-Riemannian-Deep-Learning
- Awesome-Riemannian-Optimization
- Awesome-Hyperbolic-Representation-and-Deep-Learning

[2] https://github.com/binary-husky/gpt_academic
[3] https://github.com/eseckel/ai-for-grant-writing/

4.2 CODE

SPD Manifold: The torch code sourced from SPDNetBN (Brooks et al., 2019), which does not allow batch computation for the matrix function, such as log, exp, power. It is refined in TSMNet (Kobler et al., 2022), LieBN (Chen et al., 2024b), and RMLR (Chen et al., 2024c), which allow for batch computation. RMLR is highly recommended.

- SPDNetBN (see the NeurIPS supplement)
- TSMNet
- LieBN
- RMLR

Grassmannian: The torch code sourced from GyroBN (Chen et al., 2025), which implements the gyro structure (Nguyen, 2022; Nguyen & Yang, 2023) for the PP and ONB Grassmannian.

- GyroBN

REFERENCES

Daniel Brooks, Olivier Schwander, Frédéric Barbaresco, Jean-Yves Schneider, and Matthieu Cord. Riemannian batch normalization for SPD neural networks. In *NeurIPS*, 2019.

Ziheng Chen, Yue Song, Gaowen Liu, Ramana Rao Kompella, Xiao-Jun Wu, and Nicu Sebe. Riemannian multinomial logistics regression for SPD neural networks. In *CVPR*, 2024a.

Ziheng Chen, Yue Song, Yunmei Liu, and Nicu Sebe. A Lie group approach to Riemannian batch normalization. In *ICLR*, 2024b.

Ziheng Chen, Yue Song, Rui Wang, Xiaojun Wu, and Nicu Sebe. RMLR: Extending multinomial logistic regression into general geometries. In *NeurIPS*, 2024c.

Ziheng Chen, Yue Song, Xiaojun Wu, and Nicu Sebe. Gyrogroup batch normalization. In *ICLR*, 2025.

Reinmar Kobler, Jun-ichiro Hirayama, Qibin Zhao, and Motoaki Kawanabe. SPD domain-specific batch normalization to crack interpretable unsupervised domain adaptation in EEG. In *NeurIPS*, 2022.

Xuan Son Nguyen. The gyro-structure of some matrix manifolds. In *NeurIPS*, 2022.

Xuan Son Nguyen and Shuo Yang. Building neural networks on matrix manifolds: A gyrovector space approach. In *ICML*, 2023.