Typsetting Academic Manuscripts

A Practical Introduction to LaTeX

Derron Borders

October 25, 2024

Contents

Preface

When I started my master's in linguistics at the University of Utah, I was introduced to LaTeX by a couple of professors. LaTeX is a software system for typesetting documents. LaTeX markup describes the content and layout of the document, as opposed to the formatted text found in What-you-see-is-what-you-get (WYSIWYG) word processors like Google Docs, LibreOffice Writer, and Microsoft Word.

It was very useful, especially since linguistics uses a lot of special formatting in subfields like syntax and semantics. I used it to format my master's thesis at the U because there was already a U of Utah thesis class and I only had to worry about the writing instead of the formatting. In fact, I continued to use LaTeX throughout my second master's degree at Bowling Green State University because the documents it makes are just so beautiful, compared to those created by Microsoft Word or Google Docs. There are APA 7th Edition classes (as well as many many others) and utilizing BetterBib with Zotero makes incorporating citations a breeze.

I use Visual Studio Code (VSCode) in conjunction with MacTeX, MiKTeX, LaTeX Workshop, Zotero, and BetterBib to organize and prepare my manuscripts. While it may seem like there is a steep learning curve, it has absolutely changed my life and has made the process of writing so much more fun! I am writing this manual (using LaTeX by the way) as a way to help any academic writers who are looking for a way to focus more on writing than on the formatting of their papers. As someone who lives with ADHD, it can be easy to get distracted when writing and I need organization and a process. LaTeX has allowed me to streamline a process for writing that allows me to produce manuscripts with a variety of style parameters (e.g., APA, Chicago, MLA), without having to constantly relearn and verify if my citations, references, footnotes, and endnotes are formatted correctly, due to the various packages available to LaTeX users.

As a academic in the field of Adult Learning and Leadership, I've designed this manual to be most accessible to those scholars who are not in fields like STEM, linguistics, and other specialized fields that use complex formulas and graphics in their manuscripts. This manual is a general overview of the process I use to create manuscripts for publishing that might include basic tables and figures. While this isn't geared towards those who are in the STEM field, it should still be an accessible introduction to LaTeX. If you are planning on writing a thesis or dissertation, I highly recommend learning LaTeX.

I do plan on trying to keep this document updated and hopefully adding a glossary and index section to help with navigation. You can click on the section titles in the table of comments to go directly to that section. All URLs are also clickable.

Part I
Getting Started with LaTeX

1 Introduction to LaTeX

LaTeX (pronounced *LAY-tek* or *LAH-tek*) is a **document preparation system** used for typesetting high-quality documents, especially those containing complex mathematical equations, technical content, and academic papers.

At its core, LaTeX is a markup language that defines how your content should be formatted, much like HTML or Markdown. Unlike a word processor, where you adjust formatting visually (e.g., changing fonts or margins directly), LaTeX separates **content** from **style**. You write your document in a plain text file using LaTeX commands to describe the structure (e.g., sections, equations, or references), and the system compiles it into a polished PDF output.

It was originally developed by Leslie Lamport in the 1980s as an extension of TeX, a typesetting system created by Donald Knuth. Since then, it has become a standard for academic writing in fields such as mathematics, computer science, engineering, and the natural sciences.

1.1 Why Use LaTeX?

LaTeX offers several benefits that make it a preferred tool for academics, researchers, and technical writers:

1. **Professional Typesetting and Layout**
   - LaTeX excels at producing beautifully formatted text with minimal effort.
   - It handles complex structures–like footnotes, references, and equations–elegantly and consistently.
   - Outputs are publication-ready, making it the go-to tool for academic journals and conference papers.
2. **Superior Handling of Mathematical Content**
   - LaTeX is unmatched when it comes to typesetting mathematical symbols, equations, and formulas.
   - Whether you need simple equations or complex multi-line mathematical expressions, LaTeX renders them precisely.
   - This makes it especially valuable in fields such as mathematics, physics, and engineering.
3. **Automated Document Management**
   - LaTeX automatically handles references, citations, cross-referencing, table of contents, lists of figures, and bibliographies with ease.
   - You can focus on writing content without worrying about keeping track of formatting and page layout.
4. **Scalability for Large Projects**

- LaTeX is ideal for long documents like dissertations, theses, and books.
- It allows you to manage content across multiple files and compile everything into one seamless output.
- Collaboration on complex projects is easier, as LaTeX files are plain text and version control friendly (e.g., with Git).

5. **Consistency Across Devices**
   - Unlike word processors that can behave differently across operating systems, LaTeX ensures consistent formatting regardless of platform.
   - A LaTeX document compiled on one computer will look identical when compiled on another.
6. **Open-Source and Free**
   - LaTeX is free to use and has extensive community support.
   - There are numerous templates available for research papers, theses, CVs, and presentations.
   - It is supported by tools like Overleaf, a web-based LaTeX editor that facilitates collaboration and online editing.
7. **Separation of Content and Formatting**
   - With LaTeX, you focus on content creation the system takes care of the formatting based on pre-defined document classes (like `article` or `book`).
   - This separation allows you to maintain consistent styling throughout the document without repetitive formatting work.

1.2 When Should You Use LaTeX?

LaTeX is particularly useful for:

- Writing academic papers, research articles, and theses
- Creating technical documents with equations, tables, and figures
- Producing CVs or resumes with clean and professional layouts
- Formatting scientific presentations using Beamer
- Writing books or long-form content with structured chapters

1.3 Challenges of Using LaTeX

While LaTeX offers many benefits, it also comes with a learning curve:

- It can feel intimidating for beginners since it relies on commands and coding rather than WYSIWYG (What You See Is What You Get) editing.

- Debugging errors can be frustrating, as small syntax mistakes (e.g., a missing bracket) can prevent the document from compiling.
- For quick, simple documents, a word processor like Microsoft Word may be more convenient.

Despite its initial complexity, LaTeX becomes an invaluable tool once mastered. It provides precise control over document structure and formatting, making it the best choice for academics, researchers, and technical professionals. The time invested in learning LaTeX pays off in terms of polished documents that are well-suited for publication or presentation.

In summary, if you are working on academic, technical, or mathematical content, LaTeX is worth the effort!

1.4 Comparison with Word Processors (MS Word vs. LaTeX)

When deciding whether to use LaTeX or a traditional word processor like Microsoft Word, it's helpful to compare the two based on their strengths and limitations. See the Table 1 on the next page.

Aspect	Microsoft Word	LaTeX
Ease of Use	WYSIWYG interface (What You See Is What You Get). Ideal for beginners with little to no learning curve.	Markup-based language; requires learning LaTeX commands. Can be intimidating for beginners but offers more flexibility once mastered.
Formatting Control	Limited control over advanced layout formatting; changes are made manually.	Precise and automated formatting. Handles complex layouts like equations, tables, and references consistently.
Mathematical Equations	Basic support for simple equations via built-in equation editor. Not ideal for complex mathematical typesetting.	Designed for professional-quality mathematical typesetting, including multi-line equations and matrices.
Document Length	Struggles with large documents (theses, books); prone to lag or crashes with many references and figures.	Handles long documents efficiently; ideal for theses, books, and multi-file projects with cross-references.
Collaboration	Good for real-time collaboration through cloud services like OneDrive. Issues with version control and merging changes.	Supports collaboration through Overleaf or Git-based version control. Text-based files make it easy to track changes.
References and Bibliography	Manual citation management or use of third-party tools (e.g., EndNote). Can be tedious to manage references.	Uses BibTeX or BibLaTeX to automate citations and bibliography management. Cross-referencing is seamless.
Consistency Across Devices	Formatting may vary depending on device, operating system, or Word version.	Ensures identical output regardless of device or platform. Compiled PDF will look the same everywhere.
Cost	Requires a paid Microsoft Office license.	Free and open-source.

Table 1: Comparison of Microsoft Word and LaTeX

2 Setting Up Your LaTeX Environment

Before you can begin using LaTeX, you need to install a **LaTeX distribution** and choose a suitable **LaTeX editor**. This section gives an overview of distributions and editors. Utilize the directions for each program in order to install it on your computer.

2.1 Installing LaTeX Distributions

A LaTeX distribution provides all the tools necessary to compile LaTeX code into a PDF. Below are some of the most popular distributions:

- **TeX Live** (Windows, macOS, Linux) The most widely-used distribution, especially on Linux systems. `https://www.tug.org/texlive/`
- **MiKTeX** (Windows) A lightweight distribution preferred on Windows platforms. It downloads required packages on the fly. `https://miktex.org/`
- **MacTeX** (macOS) A complete distribution for macOS, including TeX Live and additional tools like TeXShop. `https://tug.org/mactex/`

After installation, you may need to update the distribution to ensure you have the latest packages and tools.

2.2 Choosing a LaTeX Editor

Once your LaTeX distribution is installed, you need an editor to write your LaTeX code. Here are some popular options:

- **Overleaf** (Web-based) A cloud-based platform that allows you to write and compile LaTeX documents online. Great for collaboration and doesn't require local installation. `https://www.overleaf.com/`
- **TeXShop** (macOS) A simple and easy-to-use editor bundled with MacTeX. It offers a straightforward environment for writing and compiling LaTeX documents.
- **TeXstudio** (Windows, macOS, Linux) A cross-platform editor with a lot of built-in features, such as syntax highlighting and auto-completion. Ideal for intermediate users. `https://www.texstudio.org/`
- **VSCode with LaTeX Workshop Extension** (Windows, macOS, Linux) A powerful code editor that can be extended with the LaTeX Workshop extension, providing an integrated writing and compiling experience. `https://code.visualstudio.com/`

3 Your First LaTeX Document: Hello World

3.1 Overview

Creating your first LaTeX document is a great way to get familiar with the basics. In this section, I'll walk through the steps required to write and compile a basic LaTeX document that displays the message: **Hello, world!**

3.2 Step 1: Create a New LaTeX File

Follow these steps to create your first LaTeX document:

1. Open your LaTeX editor (Overleaf, TeXstudio, TeXShop, or VSCode).
2. Create a new file and name it `hello_world.tex`.
3. Type the following code into the new file:

```
\documentclass{article}
\begin{document}
Hello, world!
\end{document}
```

3.3 Explanation of the Code

Here's a breakdown of the components used in this simple LaTeX document:

- `\documentclass{article}`: This command specifies the type of document. Here, we use the `article` class, which is ideal for shorter documents such as papers, assignments, or reports.
- `\begin{document} ... \end{document}`: The content of your document goes between these two commands.
- `Hello, world!`: This is the content that will appear in the PDF output.

3.4 Step 2: Compile the Document

After typing the code, save the file and compile it to generate a PDF. The steps to compile will vary depending on the editor you are using:

1. In **Overleaf**: Click the **Recompile** button.
2. In **TeXstudio**, **TexShop**, and **VSCode**: Click the **Compile** or **Build** button.
3. Alternatively, you can compile from the command line by opening a terminal and typing:

```
pdflatex hello_world.tex
```

3.5 Step 3: View the Output

After successful compilation, you should see a PDF with the following text:

Hello, world!

3.6 Troubleshooting

If your document doesn't compile successfully, try the following troubleshooting tips:

- Make sure all braces {} and parentheses are properly closed.
- Ensure the file is saved with the `.tex` extension.
- Verify that your LaTeX distribution (such as TeX Live, MiKTeX, or MacTeX) is correctly installed.

Congratulations! You've successfully created and compiled your first LaTeX document. This simple example demonstrates the core structure of a LaTeX file. In the upcoming sections, we will explore more advanced features and formatting options to help you create professional academic documents.

4 Anatomy of a LaTeX Document

Every LaTeX document follows a well-defined structure. Understanding the anatomy of a LaTeX document helps you organize your writing efficiently and make use of LaTeX's powerful typesetting features. In this section, we will explore the essential components of a LaTeX document, including document classes, the preamble, and the document structure.

4.1 Document Classes (article, report, book, etc.)

The `documentclass` command defines the type of document you are creating. It influences the overall layout, margins, section formatting, and font sizes. Commonly used document classes include:

- **article**: Used for shorter documents such as research papers, essays, and assignments.
- **report**: Suitable for longer documents like technical reports, theses, or project documentation.
- **book**: Ideal for books or very long documents with chapters and sections.
- **letter**: Used for typesetting formal letters.
- **beamer**: Used for presentations.

Example of using the `article` class:

```
\documentclass{article}
```

You can also customize the document class by adding options in square brackets. For example:

```
\documentclass[12pt, a4paper]{report}
```

This example sets the font size to 12 points and the paper size to A4.

4.2 Preamble: Packages and Document Settings

The preamble is the section of the LaTeX document that appears before the `\begin{document}` command. In this part, you can load additional packages, set document options, and define custom commands. Below are some common elements of the preamble.

4.2.1 Loading Packages

Packages extend LaTeX's functionality. Some commonly used packages include:

- `amsmath`: Provides advanced mathematical features.
- `graphicx`: Allows you to include images.
- `hyperref`: Makes URLs and references clickable.

Example of loading packages:

```
\usepackage{amsmath}
\usepackage{graphicx}
\usepackage{hyperref}
```

4.2.2 Customizing Document Settings

You can adjust layout settings in the preamble. For example, the `geometry` package allows you to modify margins:

```
\usepackage[margin=1in]{geometry}
```

You can also define your own commands and environments:

```
\newcommand{\boldit}[1]{\textbf{\textit{#1}}}
```

This command creates a shortcut to make text both bold and italic.

4.3 Structure: Title, Abstract, Sections, and Paragraphs

The content of your document is enclosed between the `\begin{document}` and `\end{document}` commands. Inside this environment, you organize your content using sections, paragraphs, and additional formatting elements.

4.3.1 Title and Author Information

You can add a title, author name, and date using the following commands:

```
\title{My First \LaTeX{} Document}
\author{John Doe}
\date{\today}
```

Use the `\maketitle` command to generate the title block in the document:

```
\begin{document}
\maketitle
\end{document}
```

4.3.2 Abstract

The abstract provides a brief summary of the document. It is placed inside the `abstract` environment:

```
\begin{abstract}
This is a brief summary of the document.
\end{abstract}
```

4.3.3 Sections and Subsections

Use sections and subsections to organize your document. LaTeX automatically numbers the sections and generates a table of contents if required.

```
\section{Introduction}
This is the introduction.

\subsection{Background}
This is the background section.

\subsubsection{Details}
This section contains more detailed information.
```

4.3.4 Paragraphs and Line Breaks

Separate paragraphs with a blank line. If you need a line break without starting a new paragraph, use \\:

```
This is the first line.\\
This is the second line after a line break.
```

4.4 Complete Example: Basic LaTeX Document

Here is a complete example of a simple LaTeX document:

```
\documentclass{article}
\usepackage{amsmath}
\usepackage{graphicx}
\usepackage{hyperref}

\title{Anatomy of a \LaTeX{} Document}
\author{Jane Smith}
\date{\today}

\begin{document}

\maketitle

\begin{abstract}
```

```
This document provides an overview of the structure of a basic \LaTeX{} document.
\end{abstract}

\section{Introduction}
\LaTeX{} is a powerful tool for typesetting documents.

\subsection{Background}
This section provides background information.

\subsubsection{Details}
Detailed information goes here.

\section{Conclusion}
This is the conclusion of the document.

\end{document}
```

Conclusion

The anatomy of a LaTeX document consists of three main parts: the document class, the preamble, and the main content. Understanding how these components work will help you structure and organize your documents efficiently. In the next sections, we will explore more advanced formatting options and LaTeX features.

5 Typesetting a Manuscript Using the apa7 Class

The `apa7` document class is designed to follow the formatting guidelines of the American Psychological Association (APA) 7th Edition. It provides an easy way to create APA-compliant papers, including proper margins, font sizes, headings, and reference management. This section explains how to use the `apa7` class to typeset a manuscript.

5.1 Installing the apa7 Class

Most LaTeX distributions (such as TeX Live, MiKTeX, and MacTeX) come with the `apa7` class pre-installed. If you encounter errors related to the class not being found, try updating your LaTeX distribution or manually installing the class by running:

```
tlmgr install apa7  % For TeX Live
```

5.2 Using the apa7 Document Class

To use the `apa7` class, declare it at the beginning of your document with:

```
\documentclass[man]{apa7}
```

The `man` option specifies that the document is a manuscript. Other options include:

- `jou`: For journal submissions.

- `doc`: For documents (e.g., term papers).
- `12pt`: For font size (default is 12pt).

5.3 Preamble: Essential Packages for APA Formatting

In addition to the `apa7` class, you may need to load some useful packages:

```
\usepackage{csquotes}    % For proper quotation formatting
\usepackage[backend=biber,style=apa]{biblatex} % APA-style bibliography
\addbibresource{references.bib} % Bibliography file
```

The `csquotes` package ensures proper quotation marks, and `biblatex` allows APA-style citation management.

5.4 Title Page, Abstract, and Main Body

Below is an example structure of an APA-compliant manuscript using the `apa7` class.

```
\documentclass[man]{apa7}
\usepackage{csquotes}
\usepackage[backend=biber,style=apa]{biblatex}
\addbibresource{references.bib}

\title{An Example APA Manuscript}
\shorttitle{APA Manuscript}
\author{John Doe}
\affiliation{University of Example}
\authornote{This research was supported by...}

\abstract{
This is a sample abstract. It provides a brief overview of the research
topic, methods, and findings. Abstracts should be concise and informative.
}

\begin{document}

\maketitle

\section{Introduction}
The introduction explains the background of the study and its objectives.
In-text citations should follow the APA format,for example:
\textcite{doe2020} argues that...

\section{Method}
The method section describes the research design, participants, procedures,
and measures used in the study.

\section{Results}
```

```
The results section presents the findings of the study. Statistical results
should follow APA guidelines.

\section{Discussion}
The discussion interprets the results, relates them to previous research,
and suggests future directions.

\printbibliography

\end{document}
```

5.5 Explanation of Key Components

- `\title`: Specifies the title of the manuscript.
- `\shorttitle`: Provides a shortened version of the title for running heads.
- `\author` and `\affiliation`: Lists the author's name and institutional affiliation.
- `\authornote`: Adds a note about funding, acknowledgments, or conflicts of interest.
- `\abstract`: Defines the abstract.
- `\maketitle`: Generates the title page.
- `\printbibliography`: Outputs the bibliography based on the BibTeX file.

5.6 Citations and References

The `biblatex` package allows you to manage APA-style references easily. Add your references in a `references.bib` file:

```
@article{doe2020,
  author = {Doe, John},
  title = {A Study on Sample Manuscripts},
  journal = {Journal of Example Research},
  year = {2020},
  volume = {15},
  number = {3},
  pages = {23-45},
  doi = {10.xxxx/journal.2020}
}
```

You can cite this reference in the document using:

```
\textcite{doe2020}
```

5.7 Formatting and Submissions

The `apa7` class ensures proper APA formatting for margins, font size, line spacing, and section headings. For journal or class submissions, make sure to follow any additional guidelines provided by the institution or publisher.

Conclusion

Using the `apa7` class simplifies the process of creating APA-compliant manuscripts. With features such as automatic formatting, sectioning, and bibliography management, LaTeX offers a powerful tool for students, researchers, and authors following APA guidelines.

Part II
Essential LaTeX Features

6 Working with Text and Fonts

LaTeX provides extensive options for formatting text and customizing fonts. This section will guide you through the basics of text formatting, working with lists, and customizing fonts and layouts to meet your document's needs.

6.1 Basic Text Formatting: Italics, Bold, and Underlines

Here are the most common ways to format text in LaTeX:

- **Italics:** Use the `\textit{...}` command to italicize text.
- **Bold:** Use the `\textbf{...}` command for bold text.
- **Underline:** Use the `\underline{...}` command to underline text.

Example:

```
This is \textit{italicized} text.
This is \textbf{bold} text.
This is \underline{underlined} text.
```

The output will appear as:
This is *italicized* text. This is **bold** text. This is underlined text.

6.1.1 Combining Multiple Styles

You can combine different styles by nesting commands:

```
This is \textbf{\textit{bold and italicized}} text.
```

The output will be:
This is ***bold and italicized*** text.

6.2 Lists (Itemized, Enumerated, and Descriptions)

LaTeX provides several list environments to organize content.

6.2.1 Itemized Lists

Use the `itemize` environment to create bulleted lists:

```
\begin{itemize}
    \item First item
    \item Second item
\end{itemize}
```

Output:

- First item
- Second item

6.2.2 Enumerated Lists

Use the `enumerate` environment to create numbered lists:

```
\begin{enumerate}
    \item First item
    \item Second item
\end{enumerate}
```

Output:

1. First item
2. Second item

6.2.3 Description Lists

The `description` environment allows for labeled lists:

```
\begin{description}
    \item[First] Description of the first item.
    \item[Second] Description of the second item.
\end{description}
```

Output:

First Description of the first item.

Second Description of the second item.

6.3 Customizing Fonts and Layout Styles

LaTeX allows you to change fonts and layouts to customize your document.

6.3.1 Changing the Font Family

You can switch between different font families using the following commands:

```
\textrm{Roman family}   % Default serif font
\textsf{Sans-serif}     % Sans-serif font
\texttt{Monospace}      % Monospace font
```

Output:
Roman family Sans-serif Monospace

6.3.2 Changing Font Size

Font size can be set globally or locally.

Global Font Size: Set in the document class:

```
\documentclass[12pt]{article}
```

Local Font Size: Use the following commands:

```
{\tiny Tiny text}
{\small Small text}
{\normalsize Normal text}
{\large Large text}
{\Huge Huge text}
```

Output:

Tiny text Small text Normal text Large text Huge text

6.3.3 Customizing Fonts with Packages

You can change the fonts in your document by loading packages in the preamble:

```
\usepackage{times}     % Use Times New Roman
\usepackage{helvet}    % Use Helvetica
\usepackage{courier}   % Use Courier
```

If you need more font options, consider using the `fontspec` package (for XeLaTeX or LuaLaTeX):

```
\usepackage{fontspec}
\setmainfont{Georgia}  % Use Georgia font
```

6.3.4 Line Spacing and Paragraph Layout

Use the `setspace` package to adjust line spacing:

```
\usepackage{setspace}
\onehalfspacing  % 1.5 line spacing
```

Control paragraph indentation and spacing with these commands:

```
\setlength{\parindent}{0pt}   % No indentation
\setlength{\parskip}{10pt}    % 10pt space between paragraphs
```

6.3.5 Customizing Headers and Footers

The `fancyhdr` package allows you to customize headers and footers:

```
\usepackage{fancyhdr}
\pagestyle{fancy}
\fancyhead[L]{Left Header}  % Left-aligned header
\fancyhead[C]{Center Header}  % Centered header
\fancyhead[R]{Right Header}  % Right-aligned header
\fancyfoot[C]{\thepage}  % Centered page number in footer
```

Conclusion

LaTeX offers powerful tools for text formatting and layout customization. With the ability to manage lists, change fonts, and adjust layouts, you can produce professional-looking documents tailored to your needs. In the next sections, we will explore advanced features, including tables, figures, and mathematical content.

7 Including Tables and Figures

Tables and figures are essential components of many documents, especially academic papers. LaTeX provides flexible environments for creating, positioning, and referencing tables and figures. This section will guide you through using the `tabular` environment for tables, adding captions to figures, and properly referencing them within your document.

7.1 Creating Tables with `tabular`

The `tabular` environment is used to create simple tables. You can specify the alignment of columns and use vertical and horizontal lines to structure the table.

7.1.1 Basic Table Example

Here is a simple 2x2 table with left-aligned columns and horizontal borders:

```
\begin{tabular}{|l|l|}
\hline
Item & Description \\ \hline
Apples & Red or green fruit \\ \hline
Bananas & Yellow fruit \\ \hline
\end{tabular}
```

Output:

Item	Description
Apples	Red or green fruit
Bananas	Yellow fruit

7.1.2 Column Alignment

You can align columns to the left (`l`), center (`c`), or right (`r`). Use `|` to add vertical borders between columns.

Example:

```
\begin{tabular}{|c|r|l|}
\hline
Center & Right & Left \\ \hline
1 & 2 & 3 \\ \hline
\end{tabular}
```

Center	Right	Left
1	2	3

7.1.3 Adding Captions and Labels to Tables

Use the `table` environment to add captions and labels for referencing:

```
\begin{table}[h]
\centering
\begin{tabular}{|l|l|}
\hline
Item & Description \\ \hline
Apples & Red or green fruit \\ \hline
Bananas & Yellow fruit \\ \hline
\end{tabular}
\caption{A sample fruit table}
\label{tab:fruit}
\end{table}
```

Item	Description
Apples	Red or green fruit
Bananas	Yellow fruit

Table 2: A sample fruit table

7.2 Positioning and Captioning Figures

Figures can be included using the `figure` environment with the `graphicx` package. This allows you to position, caption, and reference figures effectively.

7.2.1 Including an Image

Use the `\includegraphics` command to include an image:

```
\usepackage{graphicx}

\begin{figure}[h]
```

```
\centering
\includegraphics[width=0.5\textwidth]{example-image.png}
\caption{Sample figure with caption}
\label{fig:sample}
\end{figure}
```

This code adds a figure centered on the page, scales it to half the text width, and includes a caption and label.

7.2.2 Positioning Figures

You can control the placement of figures using placement specifiers:

- `h`: Place the figure **here**, at the current location.
- `t`: Place the figure at the **top** of the page.
- `b`: Place the figure at the **bottom** of the page.
- `p`: Place the figure on a **separate** page for floats (figures or tables).

Example:

```
\begin{figure}[t]
\centering
\includegraphics[width=\textwidth]{example-image.png}
\caption{A figure at the top of the page}
\label{fig:topfigure}
\end{figure}
```

7.3 Referencing Figures and Tables in Text

LaTeX makes it easy to reference tables and figures. Use the `\label` command inside the `table` or `figure` environment and the `\ref` or `\autoref` command to refer to them in your text.

Example:

```
As shown in Table \ref{tab:fruit}, apples are a common fruit.
```

Output: As shown in Table 2, apples are a common fruit.

Similarly, you can reference figures:

```
Figure \ref{fig:sample} illustrates the sample image used.
```

Output: Figure 1 illustrates the sample image used.

7.3.1 Using `\autoref` for Automatic Labels

The `hyperref` package allows you to use `\autoref`, which automatically inserts the appropriate label (e.g., "Figure", "Table"):

```
\usepackage{hyperref}

As shown in \autoref{tab:fruit}, apples are a common fruit.
```

Output: As shown in Table 2, apples are a common fruit.

Conclusion

Tables and figures are crucial elements in many documents. LaTeX provides powerful tools to structure, position, and reference them accurately. The `tabular` and `figure` environments allow you to create professional-looking tables and figures with captions and labels. In the next section, we will explore how to manage citations and bibliographies effectively.

8 Citations and Bibliography Management

LaTeX provides powerful tools to manage citations and bibliographies, making it easy to reference academic works and produce properly formatted bibliographies. This section will cover the use of '\cite' and '\bibitem' for simple references, as well as managing references using the more advanced 'biblatex' package with the 'biber' backend. We will also explore formatting options for popular citation styles like APA, IEEE, and Chicago.

8.1 Using `\cite` and `\bibitem` for References

The simplest way to add citations in LaTeX is by using the 'thebibliography' environment with '\cite' and '\bibitem' entry corresponds to a reference, which can be cited in the text using the '\cite' command.

8.1.1 Example of thebibliography Environment

Here is an example of how to use '\thebibliography' and '\bibitem':

```
\begin{thebibliography}{99}

\bibitem{lamport1994}
  Lamport, L. (1994).
  \textit{\LaTeX{}: A Document Preparation System}.
  Addison-Wesley.

\bibitem{knuth1986}
  Knuth, D. E. (1986).
  \textit{The TeXbook}.
  Addison-Wesley.

\end{thebibliography}
```

To cite one of the references in the text, use the '[**?**] command:

```
Lamport developed \LaTeX{} \cite{lamport1994}.
```

Output: Lamport developed LaTeX [**?**].

While this method is simple, it becomes difficult to manage when working with many references. For larger projects, it is recommended to use BibTeX or 'biblatex'.

8.2 Managing References with BibTeX and BibLaTeX

BibTeX and 'biblatex' offer more powerful ways to manage bibliographies. BibTeX uses '.bib' files to store references, while 'biblatex' (with 'biber' as its backend) provides more flexibility and customization.

8.2.1 Creating a .bib File

A '.bib' file contains bibliographic information for all references in your document. Here is an example of a '.bib' file named 'references.bib':

```
@book{lamport1994,
  author    = {Leslie Lamport},
  title     = {LaTeX: A Document Preparation System},
  year      = {1994},
  publisher = {Addison-Wesley}
}

@book{knuth1986,
  author    = {Donald E. Knuth},
  title     = {The TeXbook},
  year      = {1986},
  publisher = {Addison-Wesley}
}
```

8.2.2 Using BibTeX with LaTeX

To use BibTeX, follow these steps:

1. Include the following commands in your LaTeX document:

   ```
   \bibliographystyle{plain}
   \bibliography{references}
   ```

2. Compile the document using:

   (a) 'pdflatex'

   (b) 'bibtex'

   (c) 'pdflatex' (twice)

The 'plain' style produces a numeric bibliography, but you can change the style to 'ieeetr', 'apalike', or other formats.

8.3 Using biblatex with Biber

The 'biblatex' package offers greater control over citations and bibliography formatting. It works with 'biber' as its backend for better compatibility with modern citation styles.

8.3.1 Setting Up BibLaTeX and Biber

1. Include the 'biblatex' package in the preamble:

```
\usepackage[backend=biber,style=apa]{biblatex}
\addbibresource{references.bib}
```

2. Compile the document using:
   (a) 'pdflatex'
   (b) 'biber'
   (c) 'pdflatex' (twice)

The 'backend=biber' option ensures that the bibliography is processed using 'biber', and the 'style=apa' option applies the APA citation style. Other styles include 'ieee' and 'chicago'.

8.3.2 Citing References with biblatex

Biblatex has different citation commands to cite references. For example, you can use the '\cite' and '\textcite' commands to cite references in different ways:

```
According to \textcite{lamport1994}, LaTeX is a powerful tool.
It is also discussed in other texts \cite{knuth1986}.
```

Output: According to Lamport (1994), LaTeX is a powerful tool. It is also discussed in other texts (Knuth, 1986).

You can find a cheatsheet for biblatex in the following document: `https://tug.ctan.org/info/biblatex-cheatsheet/biblatex-cheatsheet.pdf`

8.3.3 Printing the Bibliography

Use the '\printbibliography' command to print the bibliography:

```
\printbibliography
```

8.4 Formatting the Bibliography (APA, IEEE, Chicago, etc.)

The 'biblatex' package supports various citation styles. Here are some examples:

- **APA Style:**

```
\usepackage[backend=biber,style=apa]{biblatex}
```

  This style is used for social sciences and follows APA 7th edition guidelines.

- **IEEE Style:**

```
\usepackage[backend=biber,style=ieee]{biblatex}
```

IEEE style is commonly used in technical and engineering fields.

- **Chicago Style:**

```
\usepackage[backend=biber,style=chicago-authordate]{biblatex}
```

Chicago style is popular in humanities and follows the author-date format.

8.5 Troubleshooting Tips

If your bibliography is not displaying correctly, try the following:

- Ensure the '.bib' file is in the same directory as your LaTeX document.
- Run 'biber' instead of 'bibtex' if you are using 'biblatex'.
- Compile the document multiple times to update cross-references.
- Check the log file for errors related to bibliography compilation.

Conclusion

Managing citations and bibliographies in LaTeX can be done using 'thebibliography' for simple projects or BibTeX and 'biblatex' for more complex documents. Using 'biblatex' with 'biber' provides greater control and supports a wide range of citation styles such as APA, IEEE, and Chicago. With the proper setup, LaTeX can handle all your citation and referencing needs efficiently.

Part III
Formatting Academic Manuscripts

9 Structuring a Large Document

When working on large documents like theses, reports, or books, LaTeX offers powerful tools to organize the structure. This section covers how to use chapter and sectioning commands, generate a table of contents, lists of figures and tables, and create appendices.

9.1 Chapter and Sectioning Commands

LaTeX provides hierarchical commands to divide your document into sections. The most common ones are:

- `\part`: Divides the document into major parts (used in books).

- \chapter: Used for chapters (only in the report or book class).
- \section: Creates a new section.
- \subsection: Creates a subsection under a section.
- \subsubsection: Creates a smaller subsection.
- \paragraph and \subparagraph: For even finer divisions.

Example:

```
\chapter{Introduction}
This is the first chapter.

\section{Background}
This section provides background information.

\subsection{Previous Research}
This subsection covers related work.

\subsubsection{Key Findings}
Detailed information goes here.
```

Each section will be numbered automatically, and LaTeX will manage the hierarchy of the headings for you.

9.2 Table of Contents, List of Figures, and List of Tables

In longer documents, it is essential to provide a table of contents (ToC) and lists for figures and tables. LaTeX makes it easy to generate these automatically.

9.2.1 Table of Contents

Include the following command where you want the table of contents to appear:

```
\tableofcontents
```

The table of contents will include all chapters, sections, and subsections by default. It is typically placed after the title page.

9.2.2 List of Figures and List of Tables

To generate lists of figures and tables, use the following commands:

```
\listoffigures
\listoftables
```

These lists will include all figures and tables in the document along with their captions and page numbers.

9.2.3 Updating the Table of Contents, Figures, and Tables

If you make changes to the document structure, recompile the document twice to ensure that the ToC, list of figures, and list of tables are updated correctly.

9.3 Creating Appendices

Appendices are used to include supplementary information at the end of the document. In LaTeX, you can create appendices using the `appendix` environment or the `\appendix` command.

9.3.1 Using the `appendix` Command

The simplest way to create appendices is by using the `\appendix` command, which changes the sectioning to start with "Appendix" instead of a chapter or section number.

Example:

```
\appendix

\chapter{Additional Data}
This appendix contains extra data.

\section{Details of the Experiment}
Here are further details about the experiment.
```

9.3.2 Using the `appendices` Environment

The `appendices` package provides additional formatting options for appendices. Load it in the preamble:

```
\usepackage{appendix}
```

Then use the `appendices` environment:

```
\begin{appendices}
    \section{First Appendix}
    This is the first appendix.

    \section{Second Appendix}
    This is the second appendix.
\end{appendices}
```

9.4 Cross-Referencing Chapters, Sections, and Appendices

LaTeX makes it easy to reference different parts of your document. Use the `\label` command to label sections, figures, or tables, and the `\ref` command to refer to them.

Example:

```
\chapter{Introduction}
\label{ch:intro}

As discussed in Chapter \ref{ch:intro}, this is the introduction.
```

Output: As discussed in Section 1, this is the introduction.

9.5 Putting It All Together: Complete Example of a Structured Document

Here is a complete example of how to structure a LaTeX document with chapters, sections, a table of contents, and appendices:

```
\documentclass{report}
\usepackage{graphicx}
\usepackage{appendix}

\title{A Sample Report}
\author{John Doe}
\date{\today}

\begin{document}

\maketitle
\tableofcontents
\listoffigures
\listoftables

\chapter{Introduction}
This is the first chapter.

\section{Background}
Some background information.

\subsection{Previous Research}
Discussion of related work.

\chapter{Methodology}
This chapter describes the methods used.

\chapter{Results}
Results of the study are presented here.

\appendix
\chapter{Supplementary Data}
This appendix provides additional data.

\end{document}
```

Conclusion

Properly structuring a large document in LaTeX helps manage its complexity and ensures professional formatting. LaTeX handles the numbering of chapters, sections, figures, and tables automatically, and it makes it easy to create a table of contents and appendices. With these tools, you can produce polished reports, theses, or books efficiently.

10 Page Layout and Margins

In LaTeX, controlling the layout and appearance of a document, such as margins, headers, page numbering, and title pages, is essential for producing professional-looking results. This section covers how to adjust page layout and headers, set page numbering styles, and format the title page and abstract.

10.1 Adjusting Page Layout and Headers

LaTeX provides multiple ways to adjust the page layout, including margins and headers. One of the most common ways to customize these elements is through the 'geometry' and 'fancyhdr' packages.

10.1.1 Adjusting Margins with the `geometry` Package

The 'geometry' package makes it easy to set custom margins and paper sizes.

Example:

```
\usepackage[a4paper, margin=1in]{geometry}
```

This example sets the page size to A4 and the margins to 1 inch on all sides. You can also adjust individual margins:

```
\usepackage[
  top=2cm,
  bottom=2.5cm,
  left=3cm,
  right=3cm
]{geometry}
```

This code sets different margins for each side of the page.

10.1.2 Customizing Headers and Footers with `fancyhdr`

The 'fancyhdr' package allows you to modify the headers and footers.

Load the package:

```
\usepackage{fancyhdr}
\pagestyle{fancy}
```

Customize the headers and footers:

```
\fancyhead[L]{Left Header}   % Left-aligned header
\fancyhead[C]{Center Header} % Center-aligned header
\fancyhead[R]{Right Header}  % Right-aligned header
\fancyfoot[C]{\thepage}      % Page number in footer
```

Use the '\pagestyleempty' command if you want to remove headers and footers from a particular page.

10.2 Page Numbering Styles

LaTeX allows you to customize page numbering styles. You can use Roman numerals, Arabic numerals, or even letters.

10.2.1 Changing Page Numbering Styles

Use the following commands to set different page numbering styles:

```
\pagenumbering{arabic}   % Arabic numerals (1, 2, 3, ...)
\pagenumbering{roman}    % Lowercase Roman numerals (i, ii, iii, ...)
\pagenumbering{Roman}    % Uppercase Roman numerals (I, II, III, ...)
\pagenumbering{alph}     % Lowercase letters (a, b, c, ...)
\pagenumbering{Alph}     % Uppercase letters (A, B, C, ...)
```

10.2.2 Starting Page Numbers from a Specific Value

To start numbering from a specific page or number, use:

```
\setcounter{page}{3}
```

This example starts the page numbering from page 3.

10.2.3 Suppressing Page Numbers on Some Pages

To suppress page numbers on certain pages, use the following command:

```
\thispagestyle{empty}
```

This will remove the page number from the current page only.

10.3 Title Page and Abstract Layouts

In longer documents like theses or reports, a title page and an abstract are usually included. LaTeX makes it easy to create a title page using the ‘’ command, and abstracts are defined within the ‘abstract’ environment.

10.3.1 Creating a Title Page

A basic title page can be created using the following commands in the preamble:

```
\title{My First \LaTeX{} Document}
\author{John Doe}
\date{\today}
```

To generate the title page, include the ‘’ command in the document:

```
\begin{document}
\maketitle
\end{document}
```

This will create a title page with the title, author, and date.

10.3.2 Customizing the Title Page

You can further customize the title page by adding additional fields, such as an institution or subtitle:

```
\title{A Comprehensive Report}
\author{John Doe \\ Department of Computer Science}
\date{\today}
```

10.3.3 Creating an Abstract

The 'abstract' environment is used to create an abstract for your document. Place it right after the title page:

```
\begin{abstract}
This is a sample abstract. It provides a brief overview of the research topic,
methods, and findings.
\end{abstract}
```

10.3.4 Creating a Separate Title Page

If you need to create a title page that is separate from the main content, you can use the 'titlepage' environment:

```
\begin{titlepage}
\centering
{\LARGE My First \LaTeX{} Document\par}
\vspace{1cm}
{\large John Doe\par}
\vfill
{\large \today\par}
\end{titlepage}
```

This creates a standalone title page with customized layout options.

Conclusion

Adjusting the page layout, headers, page numbers, and title page can significantly enhance the appearance of your LaTeX document. The 'geometry' and 'fancyhdr' packages provide precise control over layout and headers, while LaTeX's native commands allow you to customize page numbering and create professional title pages and abstracts.

11 Handling Errors and Debugging

When working with LaTeX, it's common to encounter errors or warnings during compilation. These issues can range from simple syntax mistakes to missing files or package conflicts. This section covers common LaTeX errors, explains what they mean, and provides tips for troubleshooting and resolving compilation issues effectively.

11.1 Common LaTeX Errors and Warnings

Understanding common LaTeX errors and warnings can help you quickly identify and resolve issues.

11.1.1 Missing or Unmatched Braces

Error message:

```
! Missing } inserted.
```

This error occurs when an opening brace { or closing brace } is missing or unmatched. LaTeX expects balanced braces, so adding or removing a brace incorrectly triggers this error.

Solution: Check for unmatched braces and ensure each opening brace { has a corresponding closing brace }.

11.1.2 Undefined Control Sequence

Error message:

```
! Undefined control sequence.
```

This error occurs when LaTeX encounters a command that it does not recognize, often due to a typo or a missing package.

Solution: Verify the spelling of the command and ensure the necessary package is loaded. For example:

```
\usepackage{graphicx}  % Required for \includegraphics
```

11.1.3 File Not Found

Error message:

```
! LaTeX Error: File ‘example-image.png’ not found.
```

This error occurs when LaTeX cannot locate a required file, such as an image or ‘.bib’ file.

Solution: Ensure the file exists in the correct directory, and check that the filename is spelled correctly, including the file extension.

11.1.4 Package Conflict or Missing Package

Error message:

```
! LaTeX Error: Option clash for package ‘geometry’.
```

This occurs when the same package is loaded multiple times with conflicting options.

Solution: Ensure that the package is loaded only once. If you need to modify its options, do so in a single `\usepackage` command.

11.1.5 Overfull or Underfull Boxes

Warnings:

```
Overfull \hbox (3.0pt too wide) detected at line 23.
Underfull \vbox (badness 10000) has occurred while \output.
```

These warnings indicate that text or content does not fit properly within the defined boxes (horizontal or vertical).

Solution: Try adjusting margins or line breaks, or use the `\sloppy` command to allow LaTeX to handle spacing more leniently.

11.1.6 Bibliography Issues with BibTeX or Biber

Error message:

```
! Package biblatex Error: File ‘references.bib’ not found.
```

This error occurs when the bibliography file is missing or not correctly referenced.

Solution: Ensure the '.bib' file is in the same directory and check the path in your document:

```
\addbibresource{references.bib}
```

Compile using `biber` if you are using `biblatex`:

```
pdflatex yourfile.tex
biber yourfile
pdflatex yourfile.tex
pdflatex yourfile.tex
```

11.2 Tips for Troubleshooting Compilation Issues

Here are some practical tips to troubleshoot and resolve LaTeX compilation issues effectively.

11.2.1 Check the Log File

When LaTeX encounters an error, it generates a log file (`.log`) with detailed information about the issue. Open the log file and look for the first occurrence of `!`, which indicates the location of the error.

11.2.2 Compile Incrementally

If your document is large, compile it incrementally to isolate errors. Comment out sections of your document using:

```
% \section{This section is temporarily disabled}
```

Compile smaller portions until you identify the problematic section.

11.2.3 Use an Online Editor (e.g., Overleaf)

Online editors like Overleaf provide real-time error feedback and simplify troubleshooting by pointing directly to the problematic line.

11.2.4 Update LaTeX Distribution and Packages

Outdated packages or LaTeX distributions can cause errors. Ensure your LaTeX distribution (TeX Live, MiKTeX, MacTeX) and all installed packages are up to date.

For TeX Live:

```
tlmgr update --all
```

11.2.5 Check for Package Conflicts

If you encounter unexplained errors, check for package conflicts. Try commenting out recent `\usepackage` commands to identify conflicting packages.

11.2.6 Recompile Multiple Times

Some LaTeX features (like cross-references and bibliographies) require multiple compilations. If cross-references appear as "??" or the table of contents is not updated, compile the document multiple times.

11.2.7 Use the `-interaction=nonstopmode` Option

If you are working on a large document, it can be helpful to compile in "non-stop" mode, so the process continues even if errors occur. Run the following command in the terminal:

```
pdflatex -interaction=nonstopmode yourfile.tex
```

11.2.8 Seek Help from the Community

If you are unable to resolve an error, consult the LaTeX community. Forums like `https://tex.stackexchange.c` provide valuable resources and solutions to common problems.

Conclusion

Handling errors and debugging in LaTeX can be challenging, but by understanding common errors and using effective troubleshooting techniques, you can quickly resolve most issues. Make use of log files, incremental compilation, and community resources to streamline your workflow. With practice, you'll become more comfortable handling LaTeX errors and maintaining smooth document production.

Part IV
Advanced Features

12 Managing Complex Documents

Large documents such as theses, dissertations, or books are easier to manage if they are broken into smaller, modular components. LaTeX offers several ways to work with multiple files, including the use of '\input', '\include', and the 'subfiles' package. This section covers how to import external files, manage cross-references across files, and use subfiles to create modular documents.

12.1 Importing External Files with `\input` and `\include`

LaTeX allows you to split a document into multiple files and import them into a master document. This keeps the project organized, especially when working with long documents.

12.1.1 Using `\input`

The '\input' command inserts the contents of an external file as if it were written in the main document. This command is useful for including smaller sections or chapters.

Example:

```
\input{introduction.tex}
```

The 'introduction.tex' file could contain:

```
\section{Introduction}
This is the introduction to the document.
```

Advantages of `\input`:

- Supports multiple imports without page breaks.
- Changes in the imported files are reflected immediately upon recompilation.

12.1.2 Using `\include`

The '\include' command is similar to '\input', but it inserts the content with a page break. It is useful for including entire chapters or sections.

Example:

```
\include{chapter1.tex}
```

The 'chapter1.tex' file could contain:

```
\chapter{First Chapter}
This is the first chapter of the book.
```

Advantages of `\include`:

- Automatically starts each included file on a new page.

- Works with the '\includeonly' command to compile specific sections selectively.

Example using '\includeonly':

```
\includeonly{chapter1, chapter2}
```

This command ensures only 'chapter1.tex' and 'chapter2.tex' are compiled, saving time when working on large projects.

12.2 Cross-Referencing Across Multiple Files

When working with multiple files, you can still use LaTeX's cross-referencing system to reference sections, figures, and tables from other files.

12.2.1 Setting Up Labels and References

Add '\label' commands in the appropriate places:

```
% In introduction.tex
\section{Introduction}
\label{sec:intro}
This section introduces the topic.
```

Use the '\ref' command to reference it from another file:

```
% In another section
As discussed in Section \ref{sec:intro}, the topic is introduced.
```

Output: As discussed in Section 1, the topic is introduced.

12.2.2 Managing Cross-References with the xr Package

The 'xr' package allows you to reference labels from external documents.

1. Add the 'xr' package to your preamble:

```
\usepackage{xr}
\externaldocument{another_document}
```

2. Use '\ref' to reference sections from the external document:

```
See Section \ref{sec:intro} of the referenced document.
```

12.3 Using Subfiles for Modular Documents

The 'subfiles' package is designed to manage large documents that are split into multiple files, each of which can be compiled independently. This is particularly useful when different contributors are working on different parts of the document.

12.3.1 Setting Up the Master Document

1. Load the 'subfiles' package in the master document:

```
\documentclass{report}
\usepackage{subfiles}

\begin{document}
\tableofcontents

\subfile{chapter1.tex}
\subfile{chapter2.tex}

\end{document}
```

2. Each included chapter or section must also load the 'subfiles' package.

12.3.2 Creating Subfiles

Here is an example of a subfile ('chapter1.tex'):

```
\documentclass[../main.tex]{subfiles}

\begin{document}
\chapter{First Chapter}
This is the first chapter of the document.
\end{document}
```

In this setup:

- The path to the master document is provided as an optional argument ('../main.tex').
- Each subfile can be compiled individually for testing.

12.3.3 Advantages of Using Subfiles

- Each section or chapter can be compiled separately, making debugging easier.
- The master document compiles all subfiles into a single cohesive document.
- Contributors can work on individual subfiles without interfering with the master document.

Conclusion

Managing complex documents in LaTeX is made easier with the use of '\input', '\include', and 'subfiles'. These tools allow you to split your project into smaller, manageable files, improving organization and making collaboration easier. Cross-referencing between files ensures consistency throughout the document, and the 'subfiles' package allows modular compilation. These techniques are essential for efficiently working on large projects like theses, books, or technical reports.

13 Advanced Packages for Academics

LaTeX offers a variety of advanced packages that cater to the needs of academics working with technical documents. This section introduces two powerful packages: 'TikZ' for drawing diagrams and 'listings' packages for creating code listings.

13.1 Drawing with TikZ for Diagrams

The 'TikZ' package allows you to create high-quality diagrams directly within LaTeX. It can be used to draw graphs, flowcharts, and geometric shapes.

13.1.1 Setting Up TikZ

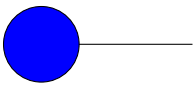
Include the TikZ library in your preamble:

```
\usepackage{tikz}
\usetikzlibrary{arrows,shapes}
```

13.1.2 Drawing Basic Shapes

Example of drawing a simple diagram:

```
\begin{tikzpicture}
  \draw (0,0) -- (2,0); % Draw a line from (0,0) to (2,0)
  \draw[fill=blue] (0,0) circle (0.5); % Blue filled circle at (0,0)
\end{tikzpicture}
```



13.1.3 Creating a Flowchart

Example of a simple flowchart:

```
\begin{tikzpicture}[node distance=2cm]
  \node (start) [rectangle, draw] {Start};
  \node (process) [circle, draw, below of=start] {Process};
  \node (end) [rectangle, draw, below of=process] {End};

  \draw[->] (start) -- (process);
  \draw[->] (process) -- (end);
\end{tikzpicture}
```

13.1.4 Creating Code Listings with the `listings` Package

The 'listings' package allows you to display code snippets with syntax highlighting.

Include the package in your preamble:

```
\usepackage{listings}
```

Example of a code listing:

```
\begin{lstlisting}[language=Python, caption=Python Example]
def add(a, b):
    return a + b

print(add(2, 3))
\end{lstlisting}
```

Listing 1: Python Example

```
def add(a, b):
    return a + b

print(add(2, 3))
```

Conclusion

The 'TikZ' and 'listings' packages provide powerful tools for academic writing. Whether you need to format units, create complex diagrams, or display code snippets, LaTeX offers a wide range of solutions to meet your needs. With these advanced packages, you can produce polished, professional documents with ease.

Part V
Polishing and Publishing

14 Customizing LaTeX Output

Customizing LaTeX output allows you to tailor your document to specific requirements, such as creating shortcuts through custom commands, defining reusable environments, and styling your document for journal submissions. This section covers how to create custom commands and environments, and how to format your document according to journal guidelines.

14.1 Creating Custom Commands and Environments

LaTeX allows users to define their own commands and environments, making the writing process more efficient and ensuring consistency across the document.

14.1.1 Creating Custom Commands

You can create reusable commands using the '\newcommand' syntax. This is useful for repetitive text, symbols, or phrases.

```
\newcommand{\R}{\mathbb{R}} % Shortcut for the real numbers symbol
\newcommand{\highlight}[1]{\textbf{#1}} % Bold text shortcut
```

Usage in the document:

```
The set of real numbers is denoted by $\R$.
This is \highlight{important}.
```

Output: The set of real numbers is denoted by $\mathbb{R}$. This is **important**.

You can also define commands with optional parameters:

```
\newcommand{\unit}[2][m]{#2~\si{#1}}
```

Usage:

```
\unit[kg]{5} % Output: 5 kg
```

14.1.2 Creating Custom Environments

You can define your own environments using the '\newenvironment' command. Custom environments are useful for creating reusable blocks such as definitions or theorems.

Example of a custom environment:

```
\newenvironment{definition}[1][Definition]
  {\par\noindent\textbf{#1:} \itshape}
  {\par}
```

Usage in the document:

```
\begin{definition}
A prime number is a natural number greater than 1 that has no positive
divisors other than 1 and itself.
\end{definition}
```

Output: **Definition:** *A prime number is a natural number greater than 1 that has no positive divisors other than 1 and itself.*

14.2 Styling for Journal Submissions

Many academic journals have strict formatting guidelines. LaTeX allows you to style your document according to these requirements by loading specific document classes or defining styles manually.

14.2.1 Using Journal Templates and Document Classes

Many journals provide LaTeX templates or document classes. These templates often come with pre-defined formatting for margins, fonts, and citations.

Example of using a journal-specific class:

```
\documentclass[12pt,twocolumn]{elsarticle} % Elsevier journal template
\usepackage{hyperref}
\begin{document}
\title{A Study on \LaTeX{} Customization}
\author{John Doe}
\maketitle
\end{document}
```

In this example, the 'elsarticle' class formats the document according to Elsevier's journal guidelines.

14.2.2 Customizing Headers, Footers, and Page Layouts

If your target journal does not provide a specific template, you can use the 'geometry' and 'fancyhdr' packages to configure your layout.

Example:

```
\usepackage[margin=1in]{geometry}
\usepackage{fancyhdr}
\pagestyle{fancy}
\fancyhead[L]{Journal of Examples}
\fancyhead[R]{\thepage}
\fancyfoot[C]{Confidential Submission}
```

This setup customizes the header to include the journal's name on the left and the page number on the right, with a footer that displays "Confidential Submission."

14.2.3 Setting Line Spacing and Font Size

Use the 'setspace' package to adjust line spacing, as required by many journals:

```
\usepackage{setspace}
\doublespacing % Set double spacing
```

You can also use font packages to change the document font if the journal has specific requirements:

```
\usepackage{times} % Use Times New Roman
```

14.2.4 Adding a Title Page and Abstract

Most journal submissions require a title page and an abstract. You can create a title page using the following commands:

```
\title{A Study on Custom \LaTeX{} Output}
\author{John Doe \\ Department of Mathematics, University of Examples}
\date{\today}
\maketitle
\begin{abstract}
This study explores methods for customizing \LaTeX{} output,
including creating custom commands and environments.
\end{abstract}
```

This will generate a title page followed by an abstract, formatted according to typical journal submission standards.

14.2.5 Including a Bibliography with BibLaTeX

Many journals require specific citation styles. Use the 'biblatex' package to manage your references:

```
\usepackage[backend=biber,style=apa]{biblatex}
\addbibresource{references.bib}
```

This example applies the APA citation style. You can change the style to meet journal requirements, such as IEEE or Chicago.

Conclusion

Customizing LaTeX output is essential for meeting the requirements of academic journals and ensuring consistency in your document. Creating custom commands and environments streamlines your workflow, while LaTeX's flexibility allows you to style your document for specific journal submissions. With these tools, you can produce high-quality documents tailored to any publication's standards.

15 Exporting and Publishing

Once your LaTeX document is complete, you'll need to export it to a distributable format, often as a PDF, and prepare it for submission or collaboration. This section covers the process of exporting PDFs, preparing your document for journal or conference submissions, and using Overleaf for collaborative writing.

15.1 Exporting PDFs and Sharing Documents

PDF is the most common output format for LaTeX documents, especially for academic and professional publishing. Exporting to PDF ensures that your document looks the same on any device and platform.

15.1.1 Exporting to PDF Using Your Editor

Most LaTeX editors provide a simple way to compile and export your document to PDF:

- **Overleaf:** Click the **Recompile** button to generate a PDF.
- **TeXstudio:** Use the shortcut 'F5' or click **Build & View**.
- **VSCode:** Use the 'Ctrl + Alt + B' (or 'Cmd + Alt + B' on macOS) shortcut to compile with the LaTeX Workshop extension.

15.1.2 Exporting from the Command Line

You can also compile your LaTeX file from the command line:

```
pdflatex mydocument.tex
```

If you are using 'biblatex' with 'biber', follow this sequence:

```
pdflatex mydocument.tex
biber mydocument
pdflatex mydocument.tex
pdflatex mydocument.tex
```

This ensures that all citations and cross-references are processed correctly.

15.1.3 Sharing Your PDF Document

Once your PDF is ready, you can share it via:

- **Email**: Attach the PDF to an email.
- **Cloud Storage**: Upload to Google Drive, Dropbox, or OneDrive.
- **GitHub or GitLab**: Share it through a version control platform if you are working on a collaborative project.

15.2 Preparing for Journal or Conference Submissions

When submitting to a journal or conference, it's important to follow the formatting guidelines provided by the publisher or organizer. Here are some key steps to ensure your document meets submission requirements.

15.2.1 Using Journal or Conference Templates

Many journals and conferences provide LaTeX templates to ensure uniform formatting. Search the journal's website or conference portal for the appropriate template. Some examples include:

- **Elsevier Journals**: Use the 'elsarticle' template.
- **IEEE Conferences**: Use the 'ieeeconf' template.
- **ACM Journals**: Use the 'acmart' template.

Example using the IEEE template:

```
\documentclass[conference]{IEEEtran}
\title{A Study on Exporting and Publishing}
\author{John Doe}
\date{}

\begin{document}
\maketitle
\begin{abstract}
This paper discusses exporting \LaTeX{} documents and preparing them for publication.
\end{abstract}
\end{document}
```

15.2.2 Reviewing Guidelines

Ensure that your document follows the submission guidelines, including:

- **Margins and Spacing**: Use the appropriate layout settings.
- **File Format**: Submit in PDF unless otherwise specified.
- **Bibliography Style**: Use the correct citation style (e.g., APA, IEEE).
- **Plagiarism Check**: Some journals require plagiarism screening before submission.

15.2.3 Handling Revisions and Resubmissions

If your submission requires revisions, make the necessary changes in LaTeX and regenerate the PDF. Use version control (e.g., Git) to track changes and resubmit your updated document.

15.3 Using Overleaf for Collaborative Writing

Overleaf is a cloud-based LaTeX editor that makes it easy to collaborate on LaTeX projects in real-time. It provides an intuitive interface and supports seamless sharing with co-authors.

15.3.1 Creating a Project on Overleaf

1. Go to `https://overleaf.com` and create an account.
2. Click **New Project** and choose:
    - **Blank Project**: Start from scratch.
    - **Upload Project**: Upload existing LaTeX files.
    - **From a Template**: Use a journal or conference template.

15.3.2 Sharing Your Project with Co-Authors

To collaborate with others:

1. Open your project.
2. Click **Share** at the top right.
3. Enter the email addresses of your collaborators and set permissions (e.g., read-only or edit access).

Your collaborators will receive an email invitation to join the project.

15.3.3 Real-Time Collaboration

Overleaf supports real-time editing, meaning multiple authors can work on the same document simultaneously. Changes are reflected immediately, and Overleaf tracks the version history of the project.

15.3.4 Compiling and Exporting in Overleaf

Overleaf automatically compiles your document when changes are made. You can download the final PDF by clicking **Download PDF**.

15.3.5 Integrating Overleaf with GitHub

Overleaf can sync with GitHub for version control:

1. In Overleaf, go to **Menu** > **GitHub**.
2. Connect your GitHub account and select a repository.
3. Push or pull changes between Overleaf and GitHub.

Conclusion

Exporting and publishing LaTeX documents is an essential part of academic writing. Whether you are submitting to a journal or collaborating with co-authors, LaTeX offers robust tools to ensure your document is polished and ready for distribution. Using Overleaf simplifies collaboration, while following journal guidelines ensures that your work meets submission requirements. With the right tools and workflows, you can efficiently manage the publishing process from start to finish.

16 Templates for Common Academic Documents

Using templates simplifies the process of formatting academic documents, ensuring they meet institutional or journal requirements. This section covers templates for research papers, theses/dissertations, and presentations with Beamer. Each template provides a starting point with appropriate formatting, layout, and structure.

16.1 Research Paper Template

A research paper template follows common formatting requirements such as title pages, abstract, sections, citations, and references. The following template demonstrates the typical structure for an article.

16.1.1 Basic Research Paper Template

```
\documentclass[12pt]{article}
\usepackage{amsmath, graphicx, hyperref, natbib}

\title{A Comprehensive Study on \LaTeX{} Templates}
\author{John Doe \\ Department of Computer Science, Example University}
\date{\today}

\begin{document}
\maketitle

\begin{abstract}
This paper discusses the use of \LaTeX{} templates for academic writing,
including research papers, theses, and presentations.
\end{abstract}

\section{Introduction}
\LaTeX{} offers various templates for academic writing to streamline the
formatting process. This paper demonstrates the structure of a typical
research paper.

\section{Methodology}
The methodology section describes the research approach.

\section{Results}
```

```
The results section presents the findings.

\section{Conclusion}
This paper highlights the benefits of using \LaTeX{} templates.

\bibliographystyle{plain}
\bibliography{references}
\end{document}
```

16.1.2 Explanation

- **Title and Author Information**: '\title', '\author', and '\date' define the title page.
- **Abstract**: The 'abstract' environment provides a summary of the paper.
- **Sections**: '\section' commands divide the paper into logical sections.
- **References**: Use the 'natbib' package to manage citations, and include a '.bib' file for the bibliography.

16.2 Thesis/Dissertation Template

Theses and dissertations typically require a more complex structure, including a title page, abstract, acknowledgments, table of contents, and chapters.

16.2.1 Basic Thesis/Dissertation Template

```
\documentclass[12pt]{report}
\usepackage{graphicx, hyperref, setspace}

\title{A Sample Thesis Template}
\author{John Doe}
\date{\today}

\begin{document}
\maketitle

\begin{abstract}
This thesis demonstrates how to use \LaTeX{} templates for dissertations.
\end{abstract}

\chapter*{Acknowledgments}
I would like to thank everyone who supported me during my research.

\tableofcontents

\chapter{Introduction}
This chapter introduces the thesis and its objectives.
```

```
\chapter{Literature Review}
This chapter discusses previous work in the field.

\chapter{Methodology}
Details about the research methodology used in the thesis.

\chapter{Results}
Presentation of results obtained from the research.

\chapter{Conclusion}
Summary of findings and suggestions for future work.

\bibliographystyle{plain}
\bibliography{references}

\appendix
\chapter{Appendix}
This appendix contains supplementary material.
\end{document}
```

16.2.2 Explanation

- **Chapters**: Use '\chapter' to organize the content into logical sections.
- **Table of Contents**: Automatically generated with '\tableofcontents'.
- **Appendix**: Use '\appendix' to add supplementary materials.
- **Line Spacing**: Add '\doublespacing' from the 'setspace' package if required.

16.3 Presentation with Beamer

The Beamer class is used to create high-quality presentations directly in LaTeX. It provides tools to format slides, include figures, and animate content.

16.3.1 Basic Beamer Presentation Template

```
\documentclass{beamer}
\usepackage{graphicx}

\title{A \LaTeX{} Presentation on Templates}
\author{John Doe}
\date{\today}

\begin{document}

\frame{\titlepage}

\begin{frame}{Outline}
```

```
\tableofcontents
\end{frame}

\section{Introduction}
\begin{frame}{Introduction}
This presentation discusses how to use templates in \LaTeX{}.
\end{frame}

\section{Research Paper Template}
\begin{frame}{Research Paper Template}
A structured template simplifies the writing process.
\end{frame}

\section{Thesis/Dissertation Template}
\begin{frame}{Thesis/Dissertation Template}
Thesis templates include chapters, abstract, and acknowledgments.
\end{frame}

\section{Conclusion}
\begin{frame}{Conclusion}
Using \LaTeX{} templates ensures consistency and professionalism.
\end{frame}

\end{document}
```

16.3.2 Explanation

- **Title Page**: Automatically generated using '\titlepage'.
- **Table of Contents**: The '\tableofcontents' command creates an outline slide.
- **Sections and Frames**: Use '\section' and '\frame' for slide content.

Conclusion

Templates simplify the creation of academic documents by ensuring consistent formatting and structure. Whether you are working on a research paper, thesis, or presentation, LaTeX provides a range of templates to meet your needs. These templates ensure that your work meets academic standards while reducing the time spent on formatting.

Part VI
Using VSCode, Github, and Zotero with LaTeX

17 Setting Up VSCode to Use with LaTeX

Visual Studio Code (VSCode) is a powerful, free code editor that can be used for LaTeX with the help of extensions. This is the LaTeX editor that I use to prepare all of my documents. Below are step-by-step instructions to set up VSCode for LaTeX development.

17.1 Step 1: Install VSCode

If you don't already have VSCode installed, download it from the official website: `https://code.visualstudio.com/` Follow the installation instructions for your operating system (Windows, macOS, or Linux).

17.2 Step 2: Install a LaTeX Distribution

Ensure you have a LaTeX distribution installed. Choose one based on your platform:

- **TeX Live**: Suitable for Linux and cross-platform users `https://www.tug.org/texlive/`
- **MiKTeX**: Recommended for Windows users `https://miktex.org/`
- **MacTeX**: Best for macOS users `https://tug.org/mactex/`

17.3 Step 3: Installing Perl on Windows

Some features of LaTeX distributions, such as **TeX Live**, rely on Perl scripts for important tasks, including package management and updates. If you are using TeX Live on Windows and working with VSCode, you may encounter errors indicating that Perl is missing. Follow the steps below to install Perl and ensure your LaTeX setup works smoothly.

Check if Perl is Installed

To see if Perl is already installed on your system:

1. Open the `Command Prompt` by typing `cmd` in the search bar.
2. Type the following command:

```
perl --version
```

If Perl is installed, this command will display the version number. If it is not installed, you will need to download and install it.

Install Perl on Windows

- Visit the official Strawberry Perl website: `https://strawberryperl.com/`
- Download the latest version of Strawberry Perl, which provides a complete Perl environment for Windows.
- Run the installer and follow the on-screen instructions.
- Ensure that the option to **add Perl to the PATH environment variable** is checked during installation.

Verify the Installation

After installation, verify that Perl is correctly set up:

1. Open the `Command Prompt` again.
2. Run the following command:

```
perl --version
```

If the installation was successful, you should see the version number of Perl displayed.

Ensure VSCode Can Use Perl

VSCode relies on your system's PATH variable to find installed tools. If Perl was added to the PATH during installation, VSCode should now be able to access it. Restart both VSCode and your computer to ensure the changes take effect.

Troubleshooting Tips

- If you still encounter issues, make sure Perl is correctly listed in the PATH. You can verify this by running the following command in the Command Prompt:

```
echo %PATH%
```

- If Perl is not listed, add it manually to the PATH via: `Control Panel` → `System` → `Advanced System Settings` → `Environment Variables`.

17.4 Step 4: Install the LaTeX Workshop Extension

The **LaTeX Workshop** extension provides all the necessary tools to write, compile, and preview LaTeX documents in VSCode.

1. Open VSCode.
2. Click on the **Extensions** icon on the left sidebar (or press `Ctrl+Shift+X`).
3. Search for **LaTeX Workshop** and click **Install**.

17.5 Step 5: Configure the LaTeX Workshop Extension

The LaTeX Workshop extension provides many features, including PDF preview, syntax highlighting, and automatic compilation. You may need to adjust some settings to ensure smooth operation.

1. Open the Command Palette (`Ctrl+Shift+P` or `Cmd+Shift+P` on macOS).
2. Search for `Preferences: Open Settings (JSON)`.
3. Add the following configuration to your `settings.json` file to customize the build process:

```
"latex-workshop.latex.tools": [
    {
        "name": "pdflatex",
        "command": "pdflatex",
        "args": [
            "-synctex=1",
            "-interaction=nonstopmode",
            "-file-line-error",
            "%DOC%"
        ]
    }
],
"latex-workshop.latex.recipes": [
    {
        "name": "pdflatex",
        "tools": ["pdflatex"]
    }
]
```

This configuration ensures that VSCode uses `pdflatex` to compile your documents with error reporting and SyncTeX enabled for PDF synchronization. Alternatively you can see and copy my `settings.json` code in my GitHub Repository: `https://github.com/derronborders/adultlearninglatex.git`

17.6 Step 6: Verify the Setup with a Sample Document

To test your setup:

1. Create a new file in VSCode and save it as `hello_world.tex`.
2. Type the following LaTeX code:

```
\documentclass{article}
\begin{document}
Hello, world!
\end{document}
```

1. Press `Ctrl+Alt+B` (or `Cmd+Alt+B` on macOS) to build the document.
2. The PDF output should appear in the preview pane.

17.7 Step 7: Troubleshooting Common Issues

- If VSCode cannot find the `pdflatex` command, ensure that the LaTeX distribution is installed correctly and that the path to its binaries is included in your system's `PATH` variable.
- If the PDF preview does not appear, try restarting VSCode and ensure that the `latex-workshop.latex.r` configuration is correct.

17.8 Other Useful Extensions

There are other useful LaTeX extensions that one can use in VSCode. Another one that I use is `LTeX+ - LanguageTool grammar/spell checking`, which is useful for catching basic spelling and grammar errors.

18 Using VSCode with GitHub for LaTeX Projects

In this section, I will guide you through the process of integrating VSCode with GitHub to back up and collaborate on your LaTeX projects. GitHub provides a version control system that ensures your work is safely stored and allows you to work with others efficiently.

18.1 Step 1: Install Git

First, make sure you have **Git** installed, as it is required to use GitHub.

1. Download Git from the official website: `https://git-scm.com/`
2. Run the installer and select the default options.
3. Verify the installation by opening the `Command Prompt` or `Terminal` and typing:

```
git --version
```

If Git is installed correctly, this command will display the version number.

18.2 Step 2: Set Up a GitHub Account

- Go to `https://github.com/` and sign up for a free GitHub account if you don't have one.
- Once signed in, click on **Repositories** → **New** to create a new repository.
- Give the repository a name (e.g., `MyLaTeXProject`) and select the option **Initialize this repository with a README**.

18.3 Step 3: Install the GitHub Extension for VSCode

1. Open VSCode.
2. Click on the **Extensions** icon (or press `Ctrl+Shift+X`).
3. Search for **GitHub Repositories** and install it.

18.4 Step 4: Configure Git in VSCode

1. Open the `Command Palette` in VSCode (`Ctrl+Shift+P` or `Cmd+Shift+P` on macOS).
2. Type `Git: Clone` and press `Enter`.
3. Paste the URL of your GitHub repository (e.g., `https://github.com/username/MyLaTeXProject.git`).
4. Choose a local folder where the repository will be cloned.

18.5 Step 5: Initialize a Git Repository for Your Project

If your project is not yet connected to Git:

1. Open your LaTeX project folder in VSCode.
2. Open the `Terminal` within VSCode (`Ctrl+‘`) and type:

```
git init
git add .
git commit -m "Initial commit"
```

This initializes the Git repository and commits all current files.

18.6 Step 6: Push Your Project to GitHub

1. Add your GitHub repository as a remote by typing:

```
git remote add origin https://github.com/username/MyLaTeXProject.git
git branch -M main
git push -u origin main
```

This command uploads your project to GitHub.

18.7 Step 7: Back Up Your Work with Regular Commits

As you make changes to your LaTeX project:

1. Use the following commands to save your changes:

```
git add .
git commit -m "Describe your changes"
git push
```

These commands stage, commit, and push your changes to GitHub.

18.8 Step 8: Collaborate with Others

Collaborating on a project is simple with GitHub:

- Share your repository's URL with collaborators.
- They can clone the repository using the `Git: Clone` command in VSCode.
- Collaborators can create their own branches using:

```
git checkout -b new-branch-name
```

- Changes can be merged into the main branch through **pull requests** on GitHub.

18.9 Step 9: Handle Merge Conflicts

Merge conflicts can occur when multiple people edit the same part of a document. Use these steps to resolve conflicts:

1. Pull the latest changes from GitHub:

```
git pull
```

2. If a conflict occurs, VSCode will highlight the conflicting sections.
3. Choose the correct version or manually merge the changes.
4. After resolving the conflict, commit the changes:

```
git add .
git commit -m "Resolved merge conflict"
git push
```

19 Utilizing Zotero to Manage References and Integrate with LaTeX

Zotero is a powerful reference management tool that helps organize research materials, generate citations, and create bibliographies. It integrates seamlessly with LaTeX through the Better BibTeX extension, allowing users to export '.bib' files of their libraries for use in LaTeX documents. This section provides a step-by-step guide to managing references with Zotero and using Better BibTeX to generate '.bib' files.

19.1 Setting Up Zotero for Reference Management

Zotero simplifies reference management by allowing users to store and organize citations, PDFs, and notes in one place.

19.1.1 Installing Zotero

1. Go to `https://www.zotero.org/` and download Zotero.
2. Install the application following the on-screen instructions.
3. Create an account to enable syncing and cloud backups.

19.1.2 Adding References to Zotero

There are several ways to add references to your Zotero library:

1. **Manually**: Click the green + icon and select **Book, Article, or Other Item** to add details manually.
2. **Web Browser Plugin**: Install the Zotero Connector plugin for your browser to quickly add items from websites, databases, and journals.
3. **Import Files**: Drag and drop PDFs or other documents into the Zotero interface. Zotero can extract metadata to create a reference entry.

Organize references into **collections** or **tags** to manage them effectively.

19.2 Integrating Zotero with Better BibTeX for LaTeX

Better BibTeX (BBT) is a Zotero plugin that enhances the export of '.bib' files and ensures compatibility with LaTeX.

19.2.1 Installing Better BibTeX

1. Open Zotero and go to **Tools** > **Add-ons**.
2. In the Add-ons Manager, click **Get Add-ons** and search for **Better BibTeX** (or download it from `https://retorque.re/zotero-better-bibtex/`).
3. Restart Zotero to enable Better BibTeX.

19.2.2 Creating a .bib File of Your Zotero Library

Once Better BibTeX is installed, follow these steps to export your Zotero references as a '.bib' file:

1. Right-click a **collection** or **individual reference** in Zotero.
2. Select **Export Library** or **Export Item**.
3. Choose **Better BibTeX** from the export format options.
4. Save the '.bib' file in the same directory as your LaTeX project.

19.2.3 Generating a Dynamic .bib File with Auto-Export

Better BibTeX also supports **automatic exports** to keep your '.bib' file synchronized with Zotero. This is useful when you are actively adding references to your library during a project.

1. In Zotero, go to **Tools** > **Preferences** > **Better BibTeX**.
2. Under the **Automatic Export** tab, click **New Auto-export**.
3. Choose a **collection** and set the export format to **Better BibTeX**.
4. Select a destination for the '.bib' file. Ensure that it matches the directory of your LaTeX project.
5. Enable **Auto-Export on Change** to keep the '.bib' file updated automatically.

19.3 Using Zotero-Generated .bib Files in LaTeX

After creating the '.bib' file, follow these steps to use it in your LaTeX document:

19.3.1 Adding the .bib File to Your Project

1. Place the '.bib' file in the same directory as your LaTeX source file.
2. In your LaTeX document preamble, add:

```
\usepackage[backend=biber,style=apa]{biblatex}
\addbibresource{references.bib}
```

19.3.2 Citing References in LaTeX

Use the '\cite' or '\textcite' commands to cite references:

```
\cite{lamport1994} % Inline citation
\textcite{knuth1986} % In-text citation with author name
```

19.3.3 Compiling Your Document with Biber

If you are using 'biblatex' with Better BibTeX, follow this compilation sequence:

```
pdflatex mydocument.tex
biber mydocument
pdflatex mydocument.tex
pdflatex mydocument.tex
```

This ensures that citations and bibliographies are correctly processed.

19.4 Troubleshooting Zotero-LaTeX Integration

Here are some common issues and solutions:

- **Missing Bibliography Entries**: Ensure that the '.bib' file is correctly linked in the LaTeX document using '\addbibresource'.
- **Biber Errors**: Verify that the compilation sequence includes 'biber' if you are using 'biblatex'.
- **Duplicate Entries**: Use the Zotero **Duplicate Items** feature to merge duplicate references before exporting the '.bib' file.

Conclusion

Using Zotero with Better BibTeX simplifies reference management and ensures smooth integration with LaTeX. By exporting '.bib' files from Zotero, you can maintain an organized bibliography and efficiently cite sources in your LaTeX projects. Auto-export features in Better BibTeX further streamline the process, keeping your bibliography up-to-date with minimal effort.

A Additional Resources

There are so many resources out there to help you on your journey of mastering LaTeX. Whenever you run into an issue, it is very likely you are not the first one to have this issue. The first thing I do is run to Google and ask my question.

Below are some additional resources that you can turn to when you are looking to typset something in a certain way and do not know.

- `www.ctan.org` - Most of the packages that are created for LaTeX are archived on this website. If you are having trouble with a certain package, going to that package's documentation is a good start.
- `www.overleaf.com` - Overleaf has a lot of tutorials on how to do things with a lot of examples. You can also find a ton of templates on there as well.
- `tex.stackexchange.com` - This website is invaluable and without it I would not have been able to finish my thesis in LaTeX. There are a lot of knowledgeable people on here and if you put the effort into asking your questions, they will put the effort into helping you. Most times, someone has already asked the question I'm looking for, so make sure it is in your bookmarks!
- `www.reddit.com/r/latex` - R/Latex is a good place to ask questions or search for solutions for to just discuss LaTeX in general.
- `https://github.com/derronborders/adultlearninglatex.git` - I created this GitHub repository to provide templates and other additional resources for using LaTeX including the files for creating this manual.