LaTeX for Linguistics
(Unfinished December 2018 draft)

Nicholas LaCara
University of Toronto

Contents

1 Background

- LaTeX (pronounced [ˈlɑ.tɛx] or [ˈlɑ.tɛk], [ˈleɪ.tɛk]. . .) is a popular system for typesetting technical documents in the fields of math, physics, engineering, and linguistics.
- It has native support for typesetting complex mathematical formulas:

$$i\hbar\frac{\partial}{\partial t}|\Psi(\mathbf{r},t)\rangle = \hat{H}|\Psi(\mathbf{r},t)\rangle$$

- Many of LaTeX's features are useful for linguists and all academic and technical writers:
  - Built-in sectioning and reference commands allow for easy structuring of documents and intra-document cross-references.
  - The inbuilt math support is used for technical notation in all subfields.
  - The bibliography system (BibTeX) allows for easy citation management and outputs formatted reference sections.
- It has been expanded in various ways over the years to accommodate Linguistics, as well:
  - Extensions (known as packages) for drawing trees, tableaux, and other linguistic representations have been developed which allow for creating consistent, professional-looking diagrams and graphics.
  - Other extensions allow for simple input of IPA characters

1.1 Why use LaTeX?

LaTeX is known to have a fairly steep learning curve, especially for those who are not used to coding or programming, though I think part of this reputation is undeserved.

- Unlike what-you-see-is-what-you-get word processors (WYSIWYG; Microsoft Word, LibreOffice Write, and Google Docs), LaTeX makes it difficult to format your documents in an arbitrary way.
- Things that are easy to do in Word are often hard to do
- Part of the philosophy of using LaTeX, however, is maintaining consistent formatting throughout a document. Thus, somethings are intentionally difficult to change.
- Many users coming from a WYSIWYG background are not used to having so much control taken away from them.

LaTeX has a number of technical features that

- Built-in support for technical document structures and cross-referencing.
- Macros and custom commands allow for easy and consistent formatting throughout a document.
- Many linguistics-specific packages for typesetting and referencing examples, drawing trees, creating Optimality Theory tableaux
- Lack of formatting options, in principle, allows for focus on document structure.

Furthermore, it is free software (no cost, no commercial limitations).

- In principle, this means that it will be available in perpetuity for free.
- .tex files can be opened with any text editor on any computer.
- LaTeX code will produce the same output no matter what system it is compiled on.

1.2 What it's not

- LaTeX is a scripting language that instructs an interpretor on how to format documents.
- LaTeXis not a word processor.

1.3 This document

- The goal of this document is to provide novice and inexperienced users with a guide for using LaTeX to create and typeset linguistics documents, though it is probably also of use to those outside of linguistics.
- The source code for this document has, as a result, been kept relatively simple and straightforward to make it easy to compare with the compiled document; only the minimal number of packages necessary have been used to create this document.

2 Getting it

2.1 MacOS

- There are a couple of different ways of installing LaTeX on MacOS.
- A simple way is to use the MacTeX distribution.
- Alternatively, one can use MacPorts to install TeXlive.

2.2 Windows

- Windows users can download and install MiKTEX.

2.3 Linux

- Most popular GNU/Linux distributions (Ubuntu, Fedora, Linux Mint...) include LATEX in their repositories. The following commands will install the full LATEX distribution (around 4ish GB):
  - Ubuntu: `$ sudo apt-get install texlive-full`
  - Fedora: `# yum install texlive-scheme-full`

2.4 The internet

- If you aren't ready to take the plunge, if you can't get the MikTEX installer to work, or if you just want to play around, there are websites that allow you to use LATEX online.
- Overleaf (`https://www.overleaf.com/`) allows for collaborative online editing of documents.

3 Workflow

3.1 Compiling a document

- One of the main ways that using LATEX differs from using a word processor is the need to *compile* a document. That is, it is necessary to convert the LATEX code into a human-readable .pdf file.
- There are different ways to compile a document, using different compilers. The compiler you need to use might depend on the packages you choose; most full LATEX installations will install all three following ones.
  - I raise this now since it raises compatibility issues for some packages that are useful for linguistics.
  - I'll point these compatibility issues out when they are relevant.
- The traditional way is to send your document to the LATEXcompiler. This creates a .dvi file that must be converted to a .ps file and then to a .pdf file.

```
$ latex document.tex
$ dvips document.dvi
$ ps2pdf document.ps
```

- The script `latexmk` automates this process (see more below).
- However, pdfLaTeX will produce a .pdf file directly. The resulting .pdf file is generally cleaner than th

```
$ pdflatex document.tex
```

- Another option is XeLaTeX, which will produce a .pdf directly, as well. XeLaTeX provides several additional extensions to LaTeX, including native UTF8 support and the ability to use all the fonts installed on your computer.

```
$ xelatex document.tex
```

- Because of how the system interacts with external databases for things such as citations and document-internal references, you may need to compile the document more than once.

3.2 Compiling with BibTeX and other

- One of the chief advantages to using LaTeX is that it automates compiling bibliographies from in-line citations. This is usually done with a bibliography management system known as BibTeX (see Section 6).
- Because of the way this works, you will have to compile your document several times.
  - The first compilation identifies the citations in your document and creates a list of these citations to search for in a bibliography file (this list is stored in a .aux file).
  - Running BibTeX will take the list of citations and search for them in the bibliography file, creating the references section for the document you are creating (which is stored in a .bbl file).
  - Compiling the document a second time will integrate the contents of the .bbl file with your original document.
  - Compiling the document a third time will link the in-line citations to the references that have been added to your document.

- This means that if you are using plain LaTeX, you will have to use the following commands to produce a .pdf file if you change the citations in the document:

```
$ latex document.tex
$ bibtex document.aux
$ latex document.tex
$ latex document.tex
$ dvips document.dvi
$ ps2pdf document.ps
```

- As noted above, the script `latexmk` automates this process, detecting changes to your .aux and .bbl files and running commands as necessary.
- Using a LaTeX-specific editor can also automate this process.

3.3 Using a LaTeX-specific editor

-

4 Basics of a LaTeX document

- A LaTeX document is just a plain text document containing LaTeX code.
- That code is processed by an interpreter that creates a formatted document from that plain text code.

4.1 Preamble

4.1.1 Document classes

- The document class tells LaTeX what kind of document you are creating. This can affect various commands made available to you.
- The document class must be declared at the beginning of your document with the `documentclass` command.

(1) `\documentclass[<options>]{<class>}`

- For our purposes, the `article` class will be sufficient, but there are several other classes available, including `book`, `memoir`, and `letter`.
- There are a number of options, which can be separated by commas:

(2)

Option:	*Function*:
`10pt`	Sets font size to 10pt
`11pt`	Sets font size to 11pt
`12pt`	Sets font size to 12pt
`a4paper`	Sets paper size to A4
`letterpaper`	Sets paper size to US Letter
`twoside`	Sets different margins on even and odd pages

- To create an 11pt document on letter paper, start your document with the following command:

(3)
```
\documentclass[11pt, letterpaper]{article}
```

4.1.2 Title, author, and date

- You can declare the title of your document and the author with the `\title` and `\author` commands.

(4)
```
\title{\LaTeX\ for Linguists}
```
(5)
```
\author{Nicholas LaCara}
```

- You'll if you want to include information like your affiliation, you can add that by including a line break with the `\\` command.

(6)
```
\author{Nicholas LaCara \\ University of Toronto}
```

However, there is no extra command for this.

- By default, the title will print the date that the document was compiled on. If you want to change that, you can use the `\date` command.

(7)
```
\date{Septembruary 35, 2068}
```

4.1.3 Packages

- After you declare the document class and title information, you can call packages that you will use in the document.
- This is done with the `\usepackage` command.

(8)
```
\usepackage[<options>]{<package>}
```

- I'll discuss specific packages in more detail in Section 7.

4.1.4 Other things

- One of the best things about LaTeX is the ability to define custom commands, which helps simplify repetitive formatting tasks as well as create shortcuts for various.
- This should usually be done in the preamble before the start of the document, but after you call various packages you might want to use.
- This is done with the `\newcommand` command (or the `\renewcommand` command if a command needs to be redefined).

  (9) `\newcommand{<command_name>}[no_of_arguments]{<definition>}`

- For instance, if you have to write the term 'verb phrase ellipsis' many times, you might define a `\VPE` command that does most the work for you:

  (10) `\newcommand{\VPE}{verb phrase ellipsis}`

- You can do more complicated things, too. For instance, if you want to create a command that **highlights text** so you can remember to come back and fix it later, you can define the following `\highlight` command:

  (11) `\newcommand{\highlight}[1]{\underline{\textbf{#1}}}`

  This basically says: Make a new command called `\highlight` that takes a single argument. Make the first argument bold and underline it.

4.2 Body

- You begin the body of your document with the `\begin{document}` command.

4.2.1 The title

- In order to typeset your title, use the `\maketitle` command.
- This will begin a new page and print the document titles, the author's name, and the date.

4.2.2 Document structure and sections

- The `article` class includes the `\part`, `\section`, `\subsection`, `\subsubsection`, `\paragraph`, and `\subparagraph` commands by default.
- These all have the same basic syntax: `\section{<title>}`
- If you throw the `\appendix` command, all subsequent sections will be labeled with letters rather than numbers.
- These also all interact with LaTeX's built-in cross-referencing system. Sectioning commands can be labeled and referred to in other places. This sentence is in Section 4.2.2.

(12) `\subsubsection{Document structure and sections}\label{S:Structure}`

(13) `This sentence is in Section \ref{S:Structure}.`

- If you want to change the formatting of section headings, you can use a package like `fancyhdr`.

4.2.3 Bibliography

- If you are using a bibliography (see Section 6 below), you should put the bibliography at the end of the document.
- Assuming you are using an external bibliography for references, you can use this command to tell BibTeX where that bibliography is: `\bibliography{<path_to_bib_file>}`

4.2.4 Ending the document

- When you reach the end of your document, you must call the `\end{document}` command.
- Any text after the `\end{document}` will be ignored.

5 Native LaTeX commands

5.1 Text formatting

- `\textbf` makes text bold.
- `\textit` makes text italic. However, you should generally use `\emph` for this purpose.
- `\textsc` sets text in small caps.

- `\textsf` sets text in sans serif.
- `\texttt` sets text in monospace.
- `\underline` underlines text (but it won't break across lines).

5.2 Text size

The following are the standard sizes provided by the `article` class:

`\tiny`	Call me Ishmael.
`\scriptsize`	Call me Ishmael.
`\footnotesize`	Call me Ishmael.
`\small`:	Call me Ishmael.
`\normalsize`:	Call me Ishmael.
`\large`:	Call me Ishmael.
`\Large`:	Call me Ishmael.
`\LARGE`:	Call me Ishmael.
`\huge`:	Call me Ishmael.
`\Huge`:	Call me Ishmael.

The syntax of these is a bit different from other LaTeX commands. They can be used inside brackets:

(14) `{\Large Call me Ishmael.}`

Or, you can introduce them as an environment:

(15) `\begin{Large} Call me Ishmael. \end{Large}`

5.3 Labeling and cross-references

5.4 List environments

5.5 Tables

- To make a table, use the `tabular` environment.

5.6 Footnotes

- For most purposes, the `\footnote` command will do what you need.
- Inserting a footnote into a document is easy.[1]

[1] Here is a footnote.

(16) `Inserting a footnote into a document is easy.\footnote{Here is a footnote.}`

- There are a few places where putting footnotes will not work, like section titles.

5.7 Changing fonts

- The default font in LaTeX is Computer Modern. It is instantly recognizable, but it has not aged well.
- Most LaTeX distributions make available a large number of alternative fonts, however.

6 Bibliography management and citations

- LaTeX has some native capacity for in-text citations, but
- Probably the most common package for bibliography and references is natbib.

7 Packages

- Here I describe a number of useful packages, with an emphasis on those that are useful or important for linguistics.
- These packages extend LaTeX's abilities by adding more commands. If you do not include the relevant packages in your preamble, these commands will not work and your document will not compile.

7.1 Example packages

- There are two or three packages that are in common use.
- All of them allow for automatic spacing of glosses in examples.

7.1.1 linguex

- Probably the most common example package, and the easiest to use.

7.1.2 gb4e

- gb4e is a bit more robust than linguex. It is the package that is being used in this document.
- It actually supports various kinds of syntax, including standard LaTeX formatting for environments and commands.

```
\begin{exe}
  \ex[*]{This is an example unglossed.}
  \ex
    \begin{xlist}
      \ex[]{This is a subexample.}
      \ex[]{\gll \'Este es un otro  ejemplo. \\
                   this is a  other example \\
            \glt ‘This is another example.’}
    \end{xlist}
\end{exe}
```

- However, Example formatting is not very flexible.
- It intentionally makes it so that superscripts and subscripts can be introduced with ˆ and _ outside mathmode. Some people like this feature, but it will screw up the placement of superscripts and subscripts in mathmode, which makes some formulas look bad. It also interferes with the code in many packages!

7.1.3 expex

- If you want total control over the formatting of your examples, `expex` can do just about anyting.
- However, it is the least intuitive and user friendly of the packages discussed here. It has it's own syntax that is notably different from standard LaTeX syntax.

```
\ex
  \ljudge* This is an example unglossed.
\xe
\pex~
  \a This is a subexample.
  \a
    \begingl
      \gla \'Este es un otro ejemplo. //
      \glb This is a other example //
      \glft ‘This is another example’//
    \endgl
\xe
```

- It also introduces its own cross-referencing system, separate from the one built in to LaTeX (though you can still use the built-in one).

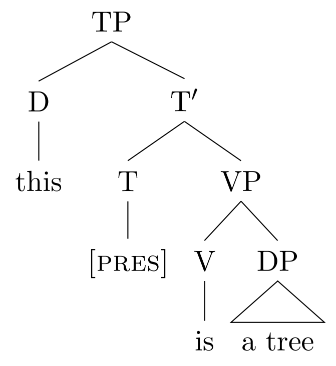
7.2 Trees

- There are a lot of options in this domain, depending on your needs.

7.2.1 qtree and variants

- Qtree uses bracket notation to create trees. This makes it fairly straightforward to use.
- This syntax has recently been re-implemented with the TikZ drawing package using the `tikz-qtree` package.

(17)
```
\begin{tikzpicture}
  \Tree [.TP [.D this ] [.T$'$ [.T \textsc{[pres]} ] %
    [.VP [.V is ] [.DP \edge[roof]; {a tree} ] ] ] ]
\end{tikzpicture}
```

(18)


- Many people swear by TikZ, though I find the bracket notation too cumbersome for more complicated trees. I also find the syntax for drawing a bit too unintuitive, but that's me.i

7.2.2 pst-jtree

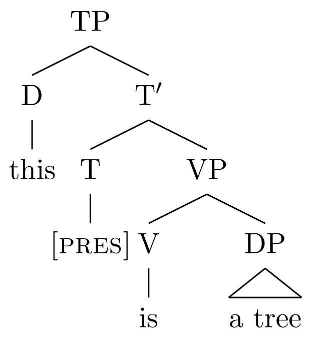
- jTree is not very user-friendly, but it is very powerful and, with practice, will let you draw most anything.[2]

(19)
```
\begin{jtree}
  \! = {TP}
    <left>[]{D}(<vert>[]{this})           ^<right>[]{T$'$}
    <left>[]{T}(<vert>[]{\textsc{[pres]}})  ^<right>[]{VP}
```

[2]It was created with producing multidominant trees in mind.

```
      <left>[]{V}(<vert>[]{is})                ^<right>[]{DP}
      <vartri>[]{a tree}.
    \end{jtree}
```

(20)



- It allows (but sometimes requires) very fine-grained control how a tree appears.
- The big drawback to jTree is that it uses PSTricks for drawing, which is incompatible with pdfLaTeX. You must use (plain) LaTeX to compile documents that use jTree.

7.3 Tableaux

- It is possible to press `tabular` environments into service here, but it can be hard to do fancy (and important) stuff like draw dashed lines between unranked constraints.

		/patak/	NoCoda	: Max	Dep
a.	→	pataka		:	*
b.		pata		: *	
c.		patak	*	:	

- However, if you want professional looking tableaux, Nathan Sanders' `OTtablx` package creates excellent looking tableaux. You can even make complicated comparative tableaux:

		/patak/	NoCoda	Max	Dep
a.	☞	pataka	0	0	1
b.		pata	0	W 1	L 0
c.		patak	W 1	0	L 0

- As with jTree, mentioned above, `OTtablx` uses PSTricks, and so it cannot be used with pdfLaTeX.

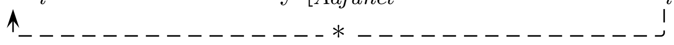
7.4 Phonetic symbols

- If you need to use IPA symbols in your document, the `tipa` package is absolutely excellent.
- The code `\textipa{[f@"nE.RIks]}` produces the following output: [fəˈnɛ.ɾɪks]

7.5 Drawing

- I use the `pst-node` package from PSTricks for drawing arrows between things.

(21) * Who$_i$ did Bill meet Sally [$_{Adjunct}$ before he talked to t_i]?



- As mentioned above, you can also use `tikz` to draw.

7.6 Semantic formulas

- LaTeX's native mathmode provides a lot of functionality for typesetting semantic formulas and denotations. A few extra packages are occasionally necessary for
- The package `stmaryrd` provides standard denotation brackets ⟦.⟧
- The packages `amssymb` and `amsmath` also provide additional functionality

(22)
```
$\llbracket\mbox{the}\rrbracket = \lambda P. \lambda Q. \exists x. %
         [Q(x) \wedge \forall y. [P(y) \rightarrow x = y]]$
```

(23) $[\![\text{the}]\!] = \lambda P.\lambda Q.\exists x.[Q(x) \wedge \forall y.[P(y) \rightarrow x = y]]$

7.7 Non-English languages and scripts

- Owing to its age, LaTeX's support for non-Latin scripts is not particularly good.
  - LaTeX was first released in 1983. Unicode would not come into existence until the late 80s.
  - [NOTE ABOUT FONT SUPPORT] Additionally, most modern font formats (like TrueType and OpenType) had not been invented yet.
  - For various technical reasons, there are different econdings for typefaces.
- If you want to diacritics or accent marks on Latin characters, there is fairly good native support for this if you are willing to use LaTeX's built-in commands.

- If you want to type in characters directly, you must use the package `inpuntenc`, with the `utf8` option:

  ```
  \usepackage[utf8]{inputenc}
  ```

  If you don't do this, nothing will appear if you type a character like `ö` .

- If you want to change various section headings, dates, hyphenation rules, and other language-specific parts of the document, use the `babel` package:

  ```
  \usepackage[<language>]{babel}
  ```

- However, if you want/need to use non-Latin characters, you should really consider using XƎLaTeX! XƎLaTeX supports modern UTF8 encoding natively and works with your system fonts.

7.8 Fonts and typographical stuff

- There are a number of fonts and typefaces that come with the full installation of LaTeX; the Danish TeX User Group keeps a pretty good list at `http://www.tug.dk/FontCatalogue/`.
- If you want to use Times both in regular text and in mathmode, use either the package `mathptmx` or the packages `newtxtext` and `newtxmath` together.

7.9 Graphics and images

- Another place that LaTeX shows its age is in how it handles graphics, and this is probably one of the places it is most frustrating to use.
- LaTeX has no native support for graphics. Graphics and images must be imported with the `graphicx` package.
- The frustrating thing is that the graphics formats you can use in a document are different depending on which method you use to compile your document.
  - latex $\rightarrow$ dvips $\rightarrow$ ps2pdf
    You can import postscript graphics (.eps files), and nothing else. If you want to use graphics with this mode, you must convert them to .eps files first.
  - pdflatex
    You can import jpeg, pdf, and png files, amongst others. It can also use .eps file (but secretly converts them to .pdf format).

    – xelatex
      XƎLaTeX supports most graphics formats.

- The reason this leads to frustration is that you might need to use a specific compiler for other reasons. For instance, if you use `pstricks` for drawing trees, you won't be able to use pdfLaTeX.
- If you need to import and manipulate .pdf documents, the `pdfpages` is extremely useful.

8 Slides and posters with Beamer

- You can use LaTeX to create presentation slides and posters using the Beamer document class.[3]

8.1 Presentations with Beamer

- Beamer is designed to create presentation slides (*frames* in its terms).
- Beamer includes special commands for creating and laying out slides, controlling the flow of information.
- Each slide is defined in a frame environment, which contains standard LaTeX code:

```
\begin{frame}{This is the title of the slide}
  The following list will appear after a pause:\pause
    \begin{itemize}
      \item This is the first item.
      \item This is the second item.
    \end{itemize}
\end{frame}
```

- The output is .pdf file that can be viewed in any .pdf viewer.

8.2 Make a real handout with the `beamerhandout` package

- Printing out your slides 6-up is for PowerPoint users. The `beamerhandout` package makes it easy to convert your slides into a true handout!

8.3 Posters using `beamerposter`

- The `beamerposter` package makes one huge Beamer frame to be used as a poster.

[3]The name *beamer* comes from the German loanword for a projector.

9 Additional resources

- The LaTeX Wikibook is a good, free resource: `https://en.wikibooks.org/wiki/LaTeX/`
  - The chapter on Linguistics has more examples and some different package examples.
- The ShareLaTeX documentation at `https://www.sharelatex.com/learn/Main_Page` is very comprehensive!
- When things fail, check out StackExchange!