LaTeX for Linguistics

Nicholas LaCara
University of Toronto

Draft – April 2019

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 in LaTeX. Think of this as a good thing: Part of the philosophy of using LaTeXis maintaining consistent formatting throughout a document. This means that some formatting options are intentionally difficult to change on a whim.

- Many users coming from a WYSIWYG background are not used to having so much control taken away from them. But the idea is that an author should concentrate less on the formatting and more on the content and structure of the document they are composing.

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 not a word processor, and you really shouldn't use it like one.
- I once co-edited a NELS proceedings, and I edited a paper where the author had encoded section titles with commands like `\noindent\textbf{1. Introduction}` rather than using the built-in `\section` command that would have done this formatting for them.
- This is what is meant when people say that LaTeX allows the author to focus on the content and structure rather than formatting. The sectioning commands (see Section 4.2.3) ensure the same consistent formatting is always used while building a structure for the document.

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 7).

- 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

- There are many text editors designed specifically around composing LaTeX documents. While many hardcore coders seem to gravitate toward other editors, I think using a LaTeX-specific editor is good for beginners as they often provide a GUI that simplifies the many of the repetitive processes that go into creating and compiling a document.
- The editors available depend on your system.
  - TeXshop on MacOS is very popular.
  - Kile is a very versatile editor available on Linux.
- Additionally, Overleaf, mentioned in Section 2.4, also provides a web interface for compiling documents.

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 class

- 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, and the instructions in this document will be assume you are using `article`, but there are several other classes available, including `report`, `book`, `memoir`, and `letter`.
- The `article` class makes available 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
`a5paper`	Sets paper size to A5
`draft`	Marks material that flows into the right margin
`letterpaper`	Sets paper size to US Letter
`titlepage`	Causes `\maketitle` to print title on separate page
`twocolumn`	Prints text in two columns (but use the package `multicol` instead).
`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}
```

- There are no specific commands for other identifying information, unfortunately. If you want to include information like your affiliation, as I've done in this document, you can include a line break with the `\\` command.

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

- It is customary to include an initial footnote in which the the author acknowledges and thanks the people who have helped them. This can be included by adding the `\thanks` command to the title.[1]

(7) `\title{\LaTeX\ for Linguists\thanks{}}`

- 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.

(8) `\date{September 2019}`

4.1.3 Packages

- After you declare the document class and title information, you can call packages that you will use in the document.
- As discussed above, packages extend the base capabilities of LaTeX in various ways. You will probably want/need to use at least a few of them.
- You can call a package with the `\usepackage` command. This will load any package that comes with your TeX distribution; it will also load any package in the same directory as the .tex file.[2]

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

- I'll discuss specific packages in more detail in Section 6.

4.1.4 Custom macros

- 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.

[1]In prinicple, this can be added to the `\author` commmand or even the `\date` command.

[2]In principle, you can give this command a full path as an argument, but this is not likely necessary when you're starting out.

- This is done with the `\newcommand` command (or the `\renewcommand` command if a command needs to be redefined).

(10) `\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:

(11) `\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:

(12) `\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.

4.1.5 Other formatting

- You may want to add some commands to the preamble that set certain aspects of the document formatting, although there is not real reason to adjust these.
- Here are a few things you may want to do though:

(13)

Command:	Effect:
`\setlength{\parindent}{0em}`	Gets rid of indentation.

4.2 Body

- You begin the body of your document with the `\begin{document}` command and end it with the `\end{document}` command.
- The material between these commands will be typeset as the finally document.[3]

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.

[3]The document is technically an environment in LaTeX terminology; see Section 5.1.1.

4.2.2 Abstract and table of contents

4.2.3 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>}`
- By default, sections, subsubsections, and subsections will all appear in the table of contents.
- These all interact with LaTeX's built-in cross-referencing system (see Section 5.4). Sectioning commands can be labeled and referred to in other places. This sentence is in Section 4.2.3.

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

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

- If you want to change the formatting of section headings, you can use a package like `titlesec`.
- If you throw the `\appendix` command, all subsequent sections will be labeled with letters rather than numbers.

4.2.4 Bibliography

- If you are using a bibliography (see Section 7 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>}`

4.2.5 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 Command types

5.1.1 Commands and environments

- There are two types of ???
- Basic commands are . These are prefixed with '\' (backslash).
  - Obligatory arguments are usually placed in curly brackets (*e.g.*, `\textsc{text}`)
  - Optional arguments can often be given in square brackets (`\usepackage[showframe]{geometry}`)
  - Some packages take no arguments (`\noindent`)
- LaTeX also uses environments. The environment begins with a `\begin{<environment>}` command and is closed with a matching `\end{<environment>}` command.
  - Material in some environments will be formatted in a certain way (*e.g.*, everything between `\begin{sf}` and `\end{sf}` will be set in sans serif).
  - Some environments make available certain commands (*e.g.*, the list environments discussed in Section 5.5 or the `tabular` environment for making tables in Section 5.6).
  - Still others introduce elements referred to as 'floats', which encapsulate figures and tables

5.1.2 Text mode and math mode

5.2 Text formatting

These are a series of commands that change the font style or weight. For instance, to make text bold, one uses `\textbf{<text>}`; all of the following commands use the same syntax.

- `\emph` *emphasizes* text.
- `\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.
- `\textsl` makes text *slanted* (but it is not supported by many fonts).
- `\texttt` sets text in `monospace`.
- `\underline` underlines text (but it won't break across lines).

5.3 Text size

The usual way to change text size in LaTeX is to use commands referring to the relative size desired. The standard sizes provided by the `article` class are as follows:

Size:	Effect:
`\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:

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

Or, you can introduce them as an environment:

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

The point of using a size command rather than a specific font size is that these will scale when the default font size of the document is changed (so if you change from 10 point to 12 point, text set in `\large` will definitely be larger than 12 point.) It is possible to specify a specific font size instead of using one of the above size commands using the `\fontsize` command, if necessary.

5.4 Labeling and cross-references

5.5 List environments

5.6 Tables

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

5.7 Floats

5.8 Footnotes

- For most purposes, the `\footnote` command will do what you need.

- Inserting a footnote into a document is easy.[4]

(18) `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.

5.9 Diacritics and special symbols

- LaTeX provides a number of commands to allow for the input of accent marks, diacritics and other special symbols in text mode.

Symbol/Diacritic	**Command**	**Example**	**Output**
Acute accent	`\'`	`\'a`	á
Cedilla	`\c`	`\c{s}`	ş
Circumflex	`\^`	`\^o`	ô
Cup	`\u`	`\u{o}`	ŏ
Dot	`\.`	`\.z`	ż
Diaeresis	`\"`	`\"u`	ü
Grave accent	`` \` ``	`` \`e ``	è
Ring	`\r{u}`	`\r{u}`	z̊
Tilde	`\~`	`\~y`	ỹ
	`\=`	`\=y`	ā
	`\v`	`\v{c}`	č
	`\H`	`\H{s}`	a̋
Ash	`\ae`	`\ae{}`	æ
å	`\aa`	`\aa{}`	å
Eth	`\dh`	`\dh{}`	ð
Dotless i	`\i`	`\u{\i}`	ĭ
œ	`\oe`	`\oe`	œ
ø	`\o`	`\o`	ø
ß	`\ss`	`\ss`	ß

A hook diacritic (`\k`), engma (`\ng`), and thorn (`\th`) are available with the T1 font encoding; see Section 6.5.

- There are several other symbols available, too, some of which must be called by special commands because they are used as part of the TeX language. Here is a sample:

[4] Here is a footnote.

Backslash	`\textbackslash`	\
Copyright	`\textcopyright`	©
Trademark	`\texttrademark`	™
Percent	`\%`	%
Dollar	`\$`, `\textdollar`	$
Hash	`\#`	#
Pound	`\pounds`	£
Ellipsis	`\dots`	$\dots$
Braces	`\{ \}`	{ }

- Many, many other symbols are available in math mode. I won't list them all here, but there are several that are very common in linguistics worth noting:

Symbol	**Command**	**Example**	**Output**
Surd	`\surd`	`$\surd$`	$\surd$
Root	`\sqrt`	`$\sqrt{\mbox{root}}$`	$\sqrt{\mbox{root}}$
Exists	`\exists`	`$\exists x$`	$\exists x$
Universal	`\forall`	`$\forall y$`	$\forall y$
Element of	`\in`	`$x \in D$`	$x \in D$
Empty set	`\emptyset`	`$\emptyset$`	$\emptyset$
Not equal	`\neq`	`$x \neq y$`	$x \neq y$

- Arrows are also available in math mode:

`\leftarrow`	$\leftarrow$	`\Leftarrow`	$\Leftarrow$
`\rightarrow`	$\rightarrow$	`\Rightarrow`	$\Rightarrow$
`\leftrightarrow`	$\leftrightarrow$	`\Leftrightarrow`	$\Leftrightarrow$
`\uparrow`	$\uparrow$	`\Uparrow`	$\Uparrow$
`\downarrow`	$\downarrow$	`\Downarrow`	$\Downarrow$
`\updownarrow`	$\updownarrow$	`\Updownarrow`	$\Updownarrow$

- Greek letters are also available in math mode, called by their names. For example, `\alpha` = α, `\phi` = ϕ, `\lambda` = λ, and `\Sigma` = Σ.

6 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.

6.1 Page layout and margins

- To set page dimensions, including margins, use the `geometry` package.
- For headers and footers, use the `fancyhdr` package.

6.2 Example packages

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

6.2.1 gb4e

- `gb4e` is one of the more common example packages, and is the package that is being used for examples in this document.
- It actually supports various kinds of syntax, including standard LaTeX formatting for environments and commands.

(19)
```
\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}
```

(20) * This is an example unglossed.

(21) a. This is a subexample.

b. Éste es un otro ejemplo.
this is a other example
‘This is another example.’

- This involves a bit more typing than other example packages. Additionally, it is not really possible to format the examples any way other than the default (though that's rarely necessary).
- Some issues to look out for with this package:

- `gb4e` 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 formulae look bad.
- These redefinitions can also interfere with the code in many packages. Consequently, if you use `gb4e`, it needs to be the last package that you load!

6.2.2 linguex

- The other common example package is `linguex`. It has a reputation for being easy to use since it has a relatively simple (though non-standard) syntax:

(22)
```
\ex. *This is an example unglossed

\ex.
  \a.  This is a subexample.
  \bg. \'Este es un otro ejemplo. \\
       this is a other example \\
       ‘This is another example.’
```

- Blank lines are crucial, here – the package relies on these to know when an example is finished.

6.2.3 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
```

- `expex` also introduces its own cross-referencing system, separate from the one built in to LaTeX, but you can still use the built-in one; see Section 5.4.

6.3 Gloss abbreviations

- The example packages listed above do not have built in support for glosses.
- For glossing in examples, one can use the `leipzig` package. This not only provides appropriate small caps abbreviations, it also creates a list of all the

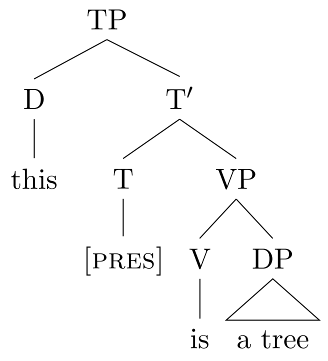
6.4 Trees

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

6.4.1 qtree and variants

- The `qtree` package uses bracket notation to create trees. This makes it fairly straightforward to use as long as you are comfortable using bracket notation (which if you probably are if you used phpSyntaxTree to make trees previously).
- This syntax has recently been re-implemented with the TikZ drawing package using the `tikz-qtree` package. It's almost certainly better to use this.

(23)
```
\begin{tikzpicture}
  \Tree [.TP [.D this ] [.T$'$ [.T \textsc{[pres]} ] %
    [.VP [.V is ] [.DP \edge[roof]; {a tree} ] ] ] ]
\end{tikzpicture}
```

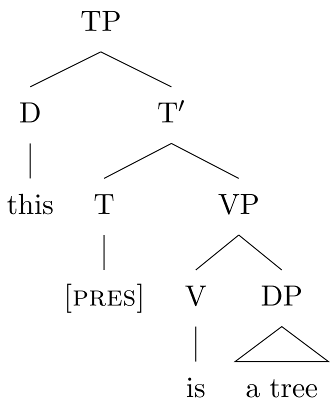
(24)


- Many people swear by TikZ for drawing. One major advantage of TikZ is that it works with any LaTeX compiler, which cannot be said of the older PSTricks package (see below)
- However, I find the bracket notation too cumbersome for more complicated trees. I also find the syntax for drawing arrows a bit too unintuitive, but that's me.

6.4.2 forest

- Another TikZ-based package that uses bracket notation is `forest`.
- It uses a slightly different bracket notation than `tikz-qtree`.

(25)
```
\begin{forest}
  [TP [D [this] ] [T$'$ [T [{\textsc{[pres]}}] ] %
    [VP [V [is] ] [DP [{a tree}, roof] ] ] ] ]
\end{forest}
```

(26)


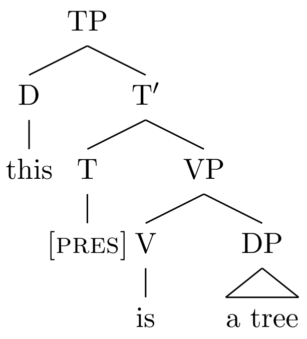
6.4.3 pst-jtree

- The `pst-jtree` package is a PSTricks-based tree-drawing package. Unlike the ones discussed above, it is does not use bracket notation but has it's own syntax.
- It is not known for being user-friendly, but it is very powerful and, with practice, will let you draw almost any tree.[5]

(27)
```
\begin{jtree}
  \! = {TP}
```

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

```
    <left>[]{D}(<vert>[]{this})            ^<right>[]{T$'$}
    <left>[]{T}(<vert>[]{\textsc{[pres]}})  ^<right>[]{VP}
    <left>[]{V}(<vert>[]{is})              ^<right>[]{DP}
    <vartri>[]{a tree}.
  \end{jtree}
```

(28)


- The big drawback to `pst-jtree` is that it uses PSTricks for drawing, which is incompatible with pdfLaTeX. You must use (plain) LaTeX to compile documents that use `pst-jtree`.
- Another issue is that `pst-jtree` sometimes requires the use to make very fine-grained control how a tree appears. Getting the spacing between nodes right can be a pain.
  - Packages like `qtree` automatically adjust the space between nodes, but sometimes this leads to comically big trees.

6.5 Fonts and typographical stuff

- The default font in LaTeX is Computer Modern. It is instantly recognizable, but has a reputation for not being pleasant to read.
- Depending on how you compile your document, your options for what fonts to use can be rather limited.
  - Due to its age, LaTeX wasn't designed with modern fonts in mind. This means that LaTeX cannot, in general, use any font available on your computer. Most fonts that you will use will come as part of your LaTeX distribution.
  - However, if you use XeLaTeX, you can use either LaTeX native fonts or any font installed on your computer.
- I discuss each of these options below.

6.5.1 Font selection under LaTeX and pdfLaTeX

- Most LaTeX distributions make available a large number of alternative fonts. The Danish TeX User Group keeps a pretty good list with many samples and details about each at the LaTeX Font Catalogue: `http://www.tug.dk/FontCatalogue/`.

- For various reasons, many authors must set documents in Times. If so, you should use either the package `mathptmx` or the packages `newtxtext` and `newtxmath` together, as this will ensure that both text and math will be set in Times.[6]

- Other popular options include Linux Libertine (using the `libertine` package) and Bitstream Charter (`\usepackage[bitstream-charter]{mathdesign}`)

- Many fonts other than Computer Modern (including those listed above) require using the `fontenc` package to ensure the right font encoding is used. If you choose a font from the LaTeX Font Catalogue, it will tell you how to use this package.

6.5.2 Using XeLaTeX

- XeLaTeX is a modern implementation of LaTeX that has full UTF8 support and can use TrueType and OpenType fonts installed on your computer.

  – As discussed in Section 3.1, you must use the XeLaTeXcompiler at the command line rather than LaTeX or pdfLaTeX.

- The usual way of choosing a font for use with XeLaTeX is with the `fontspec` package, though I recommend using `mathspec` because it will ensure that material in math mode will be set in the same font rather than in Computer Modern.

- To select fonts with `mathspec`, the following basic commands will get you started:

Command:	**Effect:**
`\setallmainfonts[<options>]{<font name>}`	Sets main font and math font.
`\setsansfont[<options>]{<font name>}`	Sets sans serif font.
`\setmonofont[<options>]{<font name>}`	Sets typewriter font.

- So, for example, if you want to set your document in Adobe's Arno Pro, you would use the command `\setallmainfonts{Arno Pro}`.

  – Be careful here, though. The font name has to match exactly what your system thinks the font is called. You may need to check your system's font settings to be sure.

[6]For various technical reasons that remain obscure to me, LaTeX uses distinct fonts in math mode and regular text. Not all fonts support math mode.

6.6 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.
  - Additionally, most modern font formats (like TrueType and OpenType) had not been invented yet.
    - This means that standard (pdf)LaTeX does not let you use any font on your computer. In general, you must use fonts set for use with LaTeX. See Section 6.5.
  - For various technical reasons, there are different econdings for typefaces. If you want to use a font other than the default
- 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 XeLaTeX! XeLaTeX supports modern UTF8 encoding natively and works with your system fonts.

6.7 Phonetic symbols

- If you need to use IPA symbols in your document, the `tipa` package is absolutely excellent.
- `tipa` provides many shortcut codes for typing IPA symbols quickly. For instance, the code `\textipa{[f@"nE.RIks]}` produces the following output: [fəˈnɛ.ɾɪks]

- The `tipa` package is not compatible with XƎLaTeX. If you want to use `tipa` codes with the `\textipa` command, use the `xunicode` package, which re-implements its functionality in XƎLaTeX.
  - To do this, you must use a font that includes IPA symbols. If your main font does not include these, you can specify an IPA-specific font using the following commands (assuming you are using `xunicode` and `fontspec`):

    ```
    \newfontfamily{\mytipa}[Scale=MatchLowercase]{<IPA Font Name>}
    \renewcommand\useTIPAfont{\mytipa}%
    ```

6.8 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

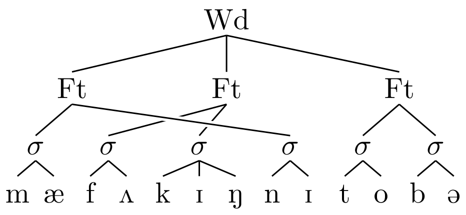
- As with jTree, mentioned above, `OTtablx` uses PSTricks, and so it cannot be used with pdfLaTeX.

6.9 Autosegmental diagrams

- The only package I know of specifically designed for drawing autosegmental diagrams is `pst-asr`, which uses PSTricks, though I'm sure there must be a TikZ package for drawing autosegmental diagrams as well.
- The code for `pst-asr` is complicated, and the learning curve is notoriously steep, but it makes great diagrams.

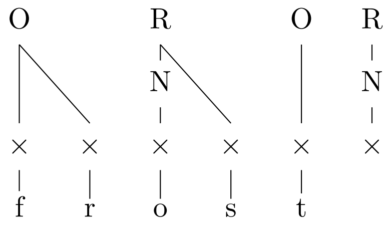
(29)
```
\newtier{foot}
\newtier{word}
\tiershortcuts
  \asr[reptype=nots,foot=(sy) 4.5ex (Ft), word=(sy) 10ex (Wd), xgap=1.25em]
  \2m{\textipa{\ae}}\2f{\textipa{2}}\3k{\textipa{I}}{\textipa{N}}
  \2n{\textipa{I}}\2t{\textipa{o}}\2b{\textipa{@}}|
  \@(5.75,foot){Ft}\-(2.5,sy)\-(5,sy)
  \@(1.5,foot){Ft}\-(0.5,sy)\-[border=1pt](7.5,sy)
  \@(10.5,foot){Ft}\-(9.5,sy)\-(11.5,sy)
  \@(5.75,word){Wd}\-(1.5,foot)\-(5.75,foot)\-(10.5,foot)
\endasr
```

(30) * Ma-fucking-nitoba



- As with `OTtablx` and `pst-jtree`, this package uses PSTricks and so cannot be compiled with pdfLaTeX.
- The `forest` package discussed in Section 6.4.2 can also draw autosegmental tree diagrams. The following is taken from the `forest` documentation:

(31)
```
\begin{forest} GP1 [
  [O[x[f]][x[r]]]
  [R[N[x[o]]][x[s]]]
  [O[x[t]]]
  [R[N[x]]]
]\end{forest}
```

(32)


6.10 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

(33)
```
$\llbracket\mbox{the}\rrbracket = \lambda P. \lambda Q. \exists x. %
    [Q(x) \wedge \forall y. [P(y) \rightarrow x = y]]$
```

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

6.11 Graphics and images

- Another place that LaTeX shows its age is in how it handles graphics,[7] 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 → dvips → 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
    XeLaTeX 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.

[7]The PDF standard dates to 1993; JPEG, to 1992; and GIF, to 1987. LaTeX was first released in 1983 and was not created with these modern image formats in mind.

6.12 Drawing

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

(35) * Who$_i$ did Bill meet Sally [$_{Adjunct}$ before he talked to t_i]?
$\uparrow$- - - - - - - - - - - - - - - * - - - - - - - - - - - - - - ⌟

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

7 Bibliography management and citations

- LaTeX has some native capacity for in-text citations, but this functionality is usually extended with various packages.
  - Probably the most common package for bibliography and references is `natbib`, which interfaces with BibTeX.
  - More recently, the `biblatex` package has been developed, interacting with the biber system. This system is more powerful and flexible than `natbib`.
- Each of these systems is fairly different under the hood, but most of the user-end functionality is essentially the same.
  - Users create a bibliography file, which is a database that contains information about references the author will use in a document.
  - Various citation commands provided by `natbib`/`biblatex` are responsible for citing the appropriate references in the bibliography.
  - When the document is compiled, LaTeX finds the references in the bibliography file that match those cited in the document.
  - Compiling the document adds the cited references to the references section and adds in-line citations to the document.

7.1 The bibliography file

7.2 Citing references

7.3 Formatting references and citations

8 Slides and posters with Beamer

- You can use LaTeX to create presentation slides and posters using the Beamer document class.[8]

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

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.

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!