Pomona Linguistics: Quick Reference Guide for LaTeX

Michael Diercks and Franny Brogan
December 16, 2022

Abstract

Here is where you write the abstract of the paper.

Contents

1 Introduction

Read the formatted document on the right of the screen, then look to the left to see the code that generated it!

1.1 What is LaTeX?

LaTeX is a typesetting program that takes a text document (with the file type .tex) and converts it to a typeset document that can be easily made into a PDF. In Overleaf, the .tex document is on the left side of your screen and the typeset document is on the right, which can be downloaded as a PDF using the 'Download PDF' button right at the top of the preview.

In many ways LaTeX is like familiar word processors (like Microsoft Word or Google Docs) because it is a typesetting program that takes input via a keyboard and produces a document that can be printed on paper. There are important differences, though. When using LaTeX you directly annotate your text with commands that tell the program how to format the text. So if you want to write something in *italics* you write `\textit{italics}`, and if you want something **bolded** you write `\textbf{bolded}`. LaTeX then converts that 'code' into what looks like italics and bold. In this Overleaf template, you can look at the .tex file itself to see what commands ('code') is used to create different kinds of formatting.

1.2 What is this document?

This is a Quick Reference Guide (published as an Overleaf template) for writing papers in LaTeX for courses in Linguistics at Pomona College.[1] It serves two purposes: first, it provides examples of formatting you need to write papers in Phonology, Syntax, and Semantics/Pragmatics using LaTeX, such as trees, numbered examples, etc. Second, if you have opened this as an Overleaf template, a copy of this project is now in your Overleaf account, and you can now edit this document itself to be an actual paper that you write. This means there is no setup required, you can just erase this text and start typing your own paper, or use this document to practice and see what changes your edits make.[2] Designing a LaTeX project from scratch can be difficult; the point of this template is that this is already done for you, you can just start writing. Our Paper Template offers the same advantages without this (long) explanatory document.

- Don't worry about breaking anything - you can always open the template link again to see this initial version again. So edit away!
- **Overleaf pro-tip**: if you want to find a location in the .tex document where the commands are written for what you are seeing in the preview on the right side of your screen, just double-click on the text in the preview and it will take you to the relevant portion of the .tex document.
- Try this here - doubled click on this sentence in the preview and take a peek at where the cursor moved to in the .tex document to the left of this preview, to see what a "comment" is using the % symbol.

1.3 Some ways to work in LaTeX

Another difference between apps like Microsoft Word/Google Docs and LaTeX is that LaTeX is more similar to the code that makes Word/Google Docs function, rather than Word itself. LaTeX doesn't come with a user interface, and instead people have created many different sorts of apps to access and use LaTeX.

- **RECOMMENDED**: Overleaf (`https://www.overleaf.com`) is a cloud-based LaTeX editor that makes it quicker and easier to start writing in LaTeX. Full access requires a paid subscription, but the free accounts are sufficient for most student work. If you are reading this template in Overleaf, you have already found your

[1] This Quick Reference Guide and the corresponding Paper Template were written by Michael Diercks and Franny Brogan. Diercks' original LaTeX teaching materials were created with assistance from Claire Halpert, Nico Baier, Jason Zentz, Maddy Bossi, and Michael Clausen.

[2] This also means you will probably want to rename the document - you can do this by hovering your cursor over the title "Pomona Linguistics Quick Reference Guide' at the top of this page, and then clicking on the pencil icon that appears to the right of the title. You can also rename the PomLingQuickReference.tex file in the leftmost panel by clicking on the down arrow.

way here. If somehow you just found this document as a PDF and haven't found the Overleaf template, click here to open this document in Overleaf.

- The online LaTeX previewer is restricted in what it can do, but if you are just drawing some basic trees or equations it is a quick-start way to start working with LaTeX. The previewer allows you to preview what you wrote and download the images (or screenshot them), which can be inserted into a document you are working on in Microsoft Word or Google Docs.[3] There are other similar applications (often referred to as equation editors) that are available online or downloadable for Mac/PC. Diercks has a set of instructions for using the online LaTeX previewer on his website.
- If you want to install TeX on your own computer, you can download the distribution for Windows here (`http://miktex.org`) and for Mac here (`http://www.tug.org/mactex/`). We do NOT recommend this for beginners.

1.4 LaTeX tutorial

In this document we don't provide a complete tutorial on LaTeX: we don't need to, because Pedro Martins has already written a tutorial specifically for linguists that also happens to be the best beginner-oriented LaTeX tutorial that we have ever found. You can find it here: `http://ptmartins.info/latex/`.

1.5 Should I use LaTeX to write my papers?

The answer to this question is 'maybe.' There are upsides to using LaTeX for linguistics papers: it is easier to format complicated diagrams, it is easier to format interlinear glosses, it auto-numbers numbered examples, it can handle IPA fonts relatively well, it is much easier to create stable cross-references within your document. Many kinds of linguistics papers require complicated formatting that can be difficult (or impossible) to do in other applications. This is especially true for theoretical linguistics theses, which can become quite difficult to format. And especially for theses, some otherwise arduous tasks (like managing references) are relatively simple in LaTeX.

That said, the major downside to using LaTeX is that there is an initial learning curve. The existence of Overleaf (and resources like this guide and Pedro Martin's tutorial) make it more achievable to ascend the learning curve, but it is still not as simple as opening Google Docs and starting to type. Many people find the pay-offs to be worth the initial cost, but not everybody thinks so, often depending on what their particular formatting needs are (this is true among both professors and students).

So, to be as clear as possible: there is **no inherent moral virtue** to working in LaTeX vs. another application, and don't let anyone tell you otherwise. The main question is whether the application you are using serves the function that you need it to. We think the resources available make it achievable to write in LaTeX for those who want to learn how (and, that this will be helpful to do the things we ask for in classes), but 1) it is not a requirement for any Pomona linguistics course, 2) Microsoft Word can be used to write all linguistics papers,[4] and 3) we don't grade LaTeX submissions higher than those written any other way.

2 Some basic things you will want to know

For a full tutorial, see Pedro Martin's tutorial. But this section *very* quickly outlines things you may want to know how to do to write papers for our courses at Pomona.

2.1 Font Sizes

To change the font size for this entire document, you can go to the first line in the .tex document which gives the class of document `\documentclass{article}` and add a different font size as an option, for example, you can change it to `\documentclass[12pt]{article}` and recompile the document (the green button in the middle/top of your screen) and the document will be changed to 12pt font.

[3] That said, if you are writing your Syntax and/or Phonology papers in Google Docs, it is highly recommended that you switch to an application that is more suited to the task, e.g. Microsoft Word or LaTeX in Overleaf. Pomona College students can access Microsoft Word for free.

[4] Google Docs less so, but you can probably eke it out.

Here is a collection of font-size commands to change the size of text in a specific location in your paper:

- `\Huge`
- `\huge`
- `\LARGE`
- `\Large`
- `\large`
- `\normalsize` (default)
- `\small`
- `\footnotesize`
- `\scriptsize`
- `\tiny`

You can change the size of a font by {\Huge putting} {\huge it} inside brackets with {\LARGE one of these commands}, which can create a {\large lot of different font sizes.} Even {\tiny tiny ones}.

2.2 Useful commands for text formatting, some that we built for you

The chart below is structured with the most general sorts of text formatting first, with linguistics-specific symbols/formatting at the bottom.

(1)

Symbol/Annotation	**Example**	**Code**
Ellipsis	…	`\dots`
Subscript	NP_{i}	`NP\subs{i}`
Superscript	NP^{i}	`NP\supers{i}`
Bold	**bold**	`\textbf{bold}`
Italic	*italic*	`\textit{italic}`
Small Caps	SMALL CAPS	`\textsc{small caps}`
Strikeout	~~strikeout~~	`\sout{strikeout}`
Underline	underline	`\underline{underline}`
circle something in text	something	`\circled{something}`
Highlight something	something	`\hl{something}`
Null	$\varnothing$	`\nothing`
Theta	ϑ	`\texttheta`
Phi	φ	`\ph`
Hash	#	`\#`
Label a left bracket	$[_{VP}$ kick it]	`\Lb{VP}`
Label a right bracket	$[[\text{kick}]_{rt}\ \text{ed}]_{wd}$	`\Rb{rt}`, `\Rb{wd}`
Trace with index	t_k	`\tr{k}`
Bar-level node	X′	`X\1`
Head Node	X°	`X\0`

- If you want to make a bulleted list, look at how this list is formatted in the .tex document using the "itemize" environment.
- As you've already seen if you are paying attention to the .tex document on the left of your screen, sections, subsections, and sub-subsection are formatted with the commands `\section{}`, `\subsection{}`, and `\subsubsection{}`, respectively.
- Look at the .tex document to see how we bolded **this text** (and Overleaf has a shortcut to make it easy, Cmd-B on Macs, Ctrl-B on PCs). *Similarly for italics*, Overleaf provides a shortcut (Cmd-I on Macs, Ctrl-I on PCs).
- Write footnotes like this.[5]

2.3 How do I do X?

You likely will wonder how to do something that seems mysterious in LaTeX but that is so simple in other applications ... and it might possibly be driving you mad. Google is your friend. Type "bulleted list in latex" into Google and you will quickly find instructions for a bulleted list, and you can use this method to find out how to do almost anything you need to do. That said, we've attempted to provide instructions for everything you need for Pomona Linguistics inside this Quick Reference Guide.

2.4 Heads up! Some potential pitfalls

For all of LaTeX's conveniences, there are some annoyances that are puzzling until you figure them out.

- Quotation marks are easy to get wrong. You want them to display "like this" and not "like this," so use the grave accent symbol above your tab key to type front quotes.
- Because a character like the percentage symbol % actually means something in LaTeX (it creates a comment), you can't just use a percentage symbol on its own to mean a percentage symbol. Instead, you have to put a backslash in front of it, like we did in in the previous sentence (check out the .tex document, and also this note about other similar characters).
- It is very easy to forget to put the end bracket on a command. So you might write this `\hl{highlight this` instead of highlight this (see the .tex) and your document will show an error. Thankfully Overleaf has warnings that pop up as you type if you have left a bracket open.
- Because it's possible to write code that won't compile, it is recommended to compile your document frequently while writing. This way, when you make a mistake you won't have pages and pages to sort through to find where the mistake is. The keystroke Cmd-S (on Mac) and Ctrl-S (on PC) will instruct Overleaf to Recompile, we suggest making it a habit to do this frequently while you write.

3 Numbered examples

It is easy to create consecutively numbered examples;

(2) This example is auto-numbered - if you uncomment the example above and recompile, the numbering will change.

The `gb4e` package that we use for numbered examples allows for inter-linear glossing and translations:

(3) this is a language example
this is the morphological gloss
'This is the translation.'

`gb4e` automatically aligns glosses with the language example (each white space means a different word). This is useful for non-English examples like the Lubukusu sentence in (4):

[5] Hey, look at me, I'm a footnote.

(4) Peter se-a-la-ba a-kula sitabu ta. **Lubukusu**
Peter NEG-SA-TNS-be SA-buy book NEG
'Peter will not be buying a book.'

If you want multiple data examples to appear in one numbered example, you use the xlist environment, which you can see in the .tex document by double clicking on the example below in the preview:

(5) a. This is the first example.
b. This is the second example.

In many linguistics publications, *all* diagrams and data are presented as numbered examples (e.g. trees, tableaux, charts, sentences with interlinear glosses, data sets of words). You can see examples of this below for both phonology (§7) and syntax (§8).

4 Using cross-references

It is useful to be able to refer your reader to different sections/examples in your paper: for example, we may want to tell you that examples (4) and (5) are built using `gb4e`, or that if you want to learn about fonts you should read §5. There are two components of a cross-reference in text:

1. You must label a particular section or example that you intend to refer to using the command `\label{LabelName}` (where "LabelName" can be any text you choose to describe the example or section you are referring to),
2. You must then use the `\ref{LabelName}` command to create the cross-reference.

So if you look in the .tex at the example in (6), you will see that we have chosen the label name "LabelSample" to identify the example. And if you look in the preceding sentence, you see an instance of using the `\ref{}` command to create a cross-reference.

(6) A numbered example with a label

Notably, you can only use the `\ref{LabelName}` command with labels that already exist somewhere in your paper - attempting to build a cross-reference to a label that does not exist will not stop your document from compiling, but it will result in a Yellow Warning from Overleaf next to the Recompile button, and it will result in question marks where the cross-reference ought to be, like this: (**??**).

These cross-references are auto-generated every time you compile your document, so as your section/example numbers change as you write your paper, the cross-references update themselves. You will never have to renumber cross-references in your document again.

5 Fonts

(7) ðɪs ɪz haʊ tu raɪt ɪn ʔɑɪpijeɪ. (This is how to write in IPA.)

This template uses the package `fontspec`, which allows you to enter IPA symbols[6] directly into your .tex document and it appropriately typesets it when creating a PDF.[7] This doesn't tell you how to type the IPA symbols in the first place, though. This requires a non-LaTeX solution.

- **Recommended**: SIL IPA keyboard is downloadable on Mac and PC and gives you keystrokes for inserting IPA characters.[8]

[6] In fact, fontspec is not specific to IPA and can typeset any Unicode symbols as long as the font you are using has those symbols.

[7] In order to compile, the `fontspec` package requires a specific compiler (XeLaTeX), which is set in the Menu, which is next to the Overleaf logo at the top left of the screen. This is already done in this template so you don't have to do anything, but this note is here for any of you that start getting into the details of the template yourself.

[8] FYI there's no way to type a 'ɚ' via this keyboard (as far as we know), so utilize one of the methods below.

- IPA Palette
- Online IPA keyboard where you can type symbols and then copy/paste them into your document
- The web app detexify is a handy tool that allows you to draw the symbol you want and it shows you what packages you need for that symbol, and what commands create it in LaTeX.

If you **do** want a LaTeX-based solution for creating IPA symbols (i.e. if you don't want to use the keyboards above), the `TIPA` package is the most commonly-used system for processing IPA symbols in LaTeX. (This document is a good resource for the macro names (i.e. commands to draw a symbol) and shortcut characters you'll need if you want to use `TIPA`.) Crucially, `TIPA` phonetic symbols can be inputted in the following two ways (look at the .tex document to see how these are coded):

1. Input macro names in the normal text environment, e.g., ðɪs ɪz haʊ tu raɪt ɪn ʔɑpijeɪ.
2. Input macro names or *shortcut characters* within the following groups or environment:
   - `\textipa{...}`, e.g., ðɪs ɪz haʊ tu raɪt ɪn ʔɑpijeɪ.
   - `{\tipaencoding ...}`, e.g., ðɪs ɪz haʊ tu raɪt ɪn ʔɑpijeɪ.
   - `\begin{IPA} ... \end{IPA}`, e.g., ðɪs ɪz haʊ tu raɪt ɪn ʔɑpijeɪ.

In our estimation, `TIPA` is functional if you have to write the occasional IPA character. But the learning curve for `TIPA` is not much different than the learning curve for the IPA keyboard (and it is harder to read the code in a document coded with `TIPA`, as you can see in the examples in the list above), so if you are frequently using IPA symbols, you will likely find it easier to learn the IPA keyboard linked to above.

6 Basic tables

Tables are super useful for formatting data and examples, but super counter-intuitive to produce in LaTeX as compared to in Word/Google Docs. This section lays out the basics; see §10 for additional formatting tricks.

6.1 General Structure of Tables

The `tabular` environment is used to typeset tables. By default, LaTeX tables are drawn without any vertical or horizontal lines and column width is predetermined; this means that any settings beyond these defaults must be defined by you. It sounds complicated, but it's actually pretty cool to have so much control over what your tables look like. Let's take a look at some simple examples.

Let's start with a basic 3x3 table with single horizontal and vertical borders. Take a look at the .tex doc to see how borders and column alignment (i.e. whether your text is aligned on the left, right, or in the center of the column) are specified.

cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

Now, here's a table that's identical to the one above, but with bolded column headers separated by a double horizontal line. Take a look at the code in the .tex document to see how it's different from the code that generated the table above.

Header 1	**Header 2**	**Header 3**
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

Here's a table identical to the one above, but with text center-aligned.

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

Now let's add a title.

I'm a table with at title

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

I'm a table with a bigger title

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

6.2 How to put a table inside a numbered example

In many kinds of linguistics papers, tables are not formatted as a separate class of elements, but are instead simply another kind of numbered example. To do this, the only difference is that you include only the "tabular" environment (an "environment" is something that you "begin" and "end") and you omit the "table" environment. In addition, we added the option "[t]" to the tabular environment to align the top of the table with the numbered example.

(8)

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

6.3 How to auto-number your tables

Alternately, you may want LaTeX to automatically number your tables (not as a numbered example, but separately). You can do so by removing the '*' from the end of the `\caption` command for any tables you want to be auto-numbered:

Table 1: I'm an auto-numbered table

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

7 Phonology Things

7.1 Derivations

Now that you know how to make basic tables in the `tabular` environment, creating a derivation table is easy. Below is a template you can use as a foundation, applying what you learned in §6.1 to add or delete rows and columns as needed.

(9) Derivation table template

'Gloss1' /UF1/	'Gloss2' /UF2/	'Gloss3' /UF3/	'Gloss 4' /UF4/	**Gloss** **Underlying form of root**
				Morphology
				Rule name here Rule name here
				Phonology
				Rule name here Rule name here Rule name here Rule name here
[SF1]	[SF2]	[SF3]	[SF4]	**Surface form**

7.2 SPE Rules

Typing up SPE-style rules in Word/Google Docs is a pain in the butt, even if you're using Bruce Hayes's handy 'HowToBrackets' guide. Luckily, typesetting phonology rules in LaTeX is relatively straightforward. This template uses the package `phonrule`, which employs the basic command `\phon{target}{change}` and then builds on that for more complex rule types. Let's start simple and work our way up through the different rule formats you'll need to use in LGCS108 and beyond.

The command `\phon{target}{change}` has two arguments: the first is the **target**, or input, of the rule, and the second is the **change**, or output. Let's say we have a rule that raises /e/ to [i]. Here, '/e/' is our **target** and '[i]' is our **change** (double-click on the rule below to see the code for it):

e → i

Now, let's say this rule only applies in a particular **environment**, or context. We'll need to add an **environment** to our rule, which we can do by modifying our command and adding a third (and sometimes a fourth) argument. There are three distinct commands you can use when you want to add an environment to your rule; which one you choose depends on *where you want your environment to appear with respect to the place holder line (i.e., underscore)*: `\phonr`, `\phonl`, and `\phonb`.

- `\phonl{target}{change}{environment}` will place your environment **to the left** of your place holder line: e → i / u_
- `\phonr{target}{change}{environment}` will place your environment **to the right** of your place holder line: e → i / _]$_{\text{word}}$
- `\phonb{target}{change}{left environment}{right environment}` will place the two components of your environment **on either side** of your place holder line: e → i / u_]$_{\text{word}}$

Now, what if we want to write some or all parts of our rule using **feature specifications** instead of IPA symbols? The `\phonfeat[]{}` command allows you to insert feature matrices into your rules using the same argument structure we saw above. Let's take a look at an example:

(10) English Aspiration Rule: $\begin{bmatrix} \text{-voice} \\ \text{-delayed release} \end{bmatrix}$ → [+spread glottis] / [$_{\text{word}}$ _

Keep in mind that you an use the `\phonfeat[]{}` command to insert feature matrices of any size into any part

of your rule (the target, change, and/or environment) following the format presented in example (10) above.

As you may know, the English Aspiration Rule in (10) is incomplete; we need to add an additional environment. For rules that have more than one environment, we can add **curly brackets** to the environment of our rule. As far as we know, the `phonrule` package only allows you to add a single (open) curly bracket. Double click on rule in (10) to check out the code in the .tex document.

(11) English Aspiration Rule (revised): $\begin{bmatrix} \text{-voice} \\ \text{-delayed release} \end{bmatrix} \rightarrow [\text{+spread glottis}] \; / \; \left\{ \begin{array}{l} [_{\text{word}} \; \_ \\ [_{\sigma} \; \_ \begin{bmatrix} \text{+syllabic} \\ \text{+stress} \end{bmatrix} \end{array} \right.$

For the purposes of course papers and the like, the single-bracket version is probably fine, although you may want to check with your instructor. For more advanced LaTeX users who want to include both left and right curly brackets, there's a makeshift way to do so in math mode; see the example below:

(12) English Aspiration Rule (revised):

$$\begin{bmatrix} \text{-voice} \\ \text{-delayed release} \end{bmatrix} \rightarrow [+spreadglottis] \; / \; \left\{ \begin{array}{l} [_{\text{word}} \; \_ \\ [_{\sigma} \; \_ \begin{bmatrix} \text{+syllabic} \\ \text{+stress} \end{bmatrix} \end{array} \right\}$$

This work-around isn't ideal–particularly because the opening and closing curly brackets are of slightly different sizes–but should suffice for your purposes.

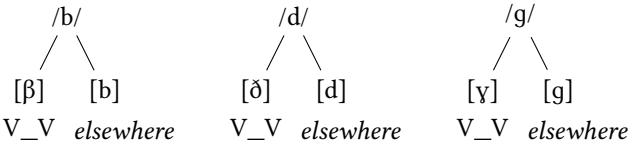
7.3 Phonemicization diagrams

For phonemicization diagrams, this template uses the `TikZ` package, which is a general tool for creating graphic elements in LaTeX. Once you understand the basic structure of these diagrams, adding additional tiers, changing the information conveyed in your tier(s), and modifying association lines is fairly simple.

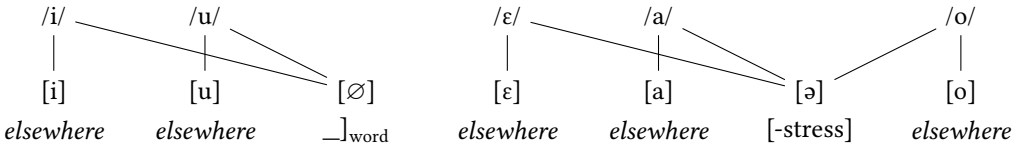
As you'll see in the .tex document, we're simply creating a series of "nodes", naming them, positioning them in relation to one another using (x,y) coordinates, and then specifying what text each node should contain. Once each node has been named, we use the `\draw` command to draw association lines between the nodes of our choice. Below are three sets of phonemicization diagrams of varying complexities so you can see how this works:

(13) A simple set of phonemicization diagrams

(14) A slightly more complex set of phonemicization diagram



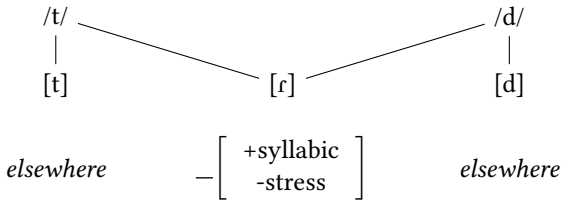
(15) A very complex set of phonemicization diagram



We recommend starting with your most "basic" tier (that is, the tier whose spatial positioning you're most sure of) and then positioning your other tiers accordingly. We like to start with the allophone tier and then position the corresponding phonemes and environments around it.

You can embed any component from the `phonrule` package in the *tikzpicture* environment when drawing phonemicization diagrams, such as feature matrices, as in (16).

(16) Neutralization of /t/ and /d/ in English

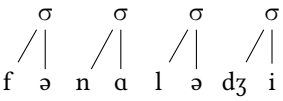


7.4 Autosegmental diagrams

There are various packages you an use to draw diagrams with autosegmental tiers in LaTeX. More advanced users may be interested in exploring the `pst-asr` package, which was designed to be used by linguists to typeset autosegmental representations (and generally renders them more aesthetically-pleasing than what you'll see here). Package contents can be found here.

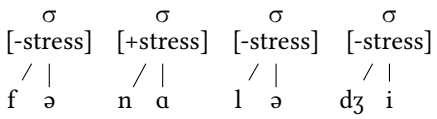
For the sake of simplicity, this template typesets autosegmental diagrams using the `TikZ` package, which we introduced in §7.3. Let's start with a basic example that illustrates the syllabic division of the word *phonology*, [fə.nɑ.lə.dʒi].

(17) Syllable associations for *phonology*, [fə.nɑ.lə.dʒi]



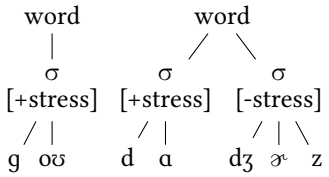
Now, using the `\stackanchor{}{}` command in the `stackengine` package, we can add a **stress tier** under our syllable nodes to show that [fə.nɑ.lə.dʒi] has antepenultimate stress. Notice that we've also shifted all our nodes (except for the left-most group) over .5 on the x-axis to make room for the stress tier:

(18) Syllable and stress associations for *phonology*, [fə.ˈnɑ.lə.dʒi]



Now let's build on this foundation, adding **structure above the syllable**. Suppose we want to show stress, syllable structure, and word division for the phrase *go, Dodgers!*, [ˈgoʊ.ˈdɑ.dʒɚz]:

(19) Word, syllable, and stress associations for *go, Dodgers!*, [ˈgoʊ.ˈdɑ.dʒɚz]



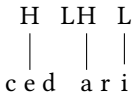
To build more tiers, simply add additional sets of nodes and association lines accordingly.

Finally, using the `TikZ` package, we can follow the procedure outlined above to create a tier for **tonal autosegments**. Below we see the hypothetical underlying form /cedeari/ with its tonal associations:

(20) Underlying tonal associations for /cedari/

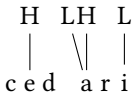
It's recommended that you play around with spacing and node placement here. Similar to the procedure laid out in §7.3, we like to start by anchoring the segments themselves and subsequently placing the tones to match the x values for their corresponding vowels. Note that one advantage of using the `TikZ` package for autosegmental diagrams is the ease with which you can control your association lines. For example, let's say we have a Syncope rule that deletes the second /e/ from the underlying form above, leaving its 'L' tone unassociated. We simply **delete or comment out the /e/ node** as well as **its tonal association line**:

(21) Tonal associations for /cedari/ after Syncope applies



Now imagine that, after Syncope applies, a Reassociation rule associates stray tones with the following vowel. We can show the reassociation of 'L' by connecting it to the [a] segment:

(22) Tonal associations for /cedari/ after Syncope and Reassociation



7.5 Tableaux

While you can certainly make beautiful Optimality-Theoretic tableaux in the `tabular` environment, the `ot-tableau` package makes this task a little bit easier. The `ot-tableau` package uses the `tableau` environment, which functions similarly to the `tabular` environment. Here's a simple tableau with three candidates and three constraints with a known ranking, as indicated by the solid vertical lines between constraints:

(23) Simple tableau with known constraint ranking

/stap/	*Complex	Anchor-IO	Contiguity-IO
a. stap	*!		
☞ b. sap			*
c. tap		*!	

Here's the same tableau, but with a dotted line separating the first and second constraints to indicate that the ranking relationship between these two constraints is unknown:

(24) Simple tableau with partially-known constraint ranking

/stap/	*Complex	Anchor-IO	Contiguity-IO
a. stap	*!		
☞ b. sap			*
c. tap		*!	

Finally, here's an example of a tableau where the incorrect constraint ranking gives us the wrong "winner". The 💣 marks the candidate that wins (but shouldn't), and the ☹ marks the candidate that should win (but doesn't):

(25) Simple tableau with incorrect constraint ranking

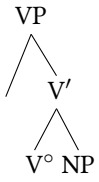
/stap/	*Complex	Contiguity-IO	Anchor-IO
a. stap	∗!		
☹ b. sap		∗!	
💣 c. tap			∗

8 Syntax Things

8.1 Trees trees trees

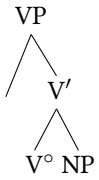
One of the biggest things syntacticians rely on LaTeX for is drawing trees. This template uses the package `forest`, so to draw a tree in this template you use the `\Forest{tree.in.here}` command. (look to the left at the .tex document for the code for the tree below)

(26)



The code for (26) is formatted so that each level of the tree is indented separately. This is not necessary; the code for (27) has the same exact bracket structures, just with no line breaks or indents. You'll see in the preview that these trees are identical, despite the code in the .tex document being formatted differently.

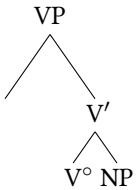
(27)



We recommend formatting your code something more like (26) than (27), largely because as the trees get larger it can get tougher and tougher to interpret your code. We encourage you to try editing the trees above and see what happens!

You will notice in the tree above that because there is an empty branching node, the branches don't all go at precisely the same angle. There are spacing options you can invoke to adjust this, as in the example below, where the top node in the tree is specified as having "nice empty nodes:"

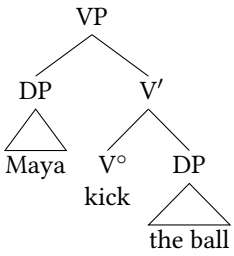
(28)



This is really only helpful for a tree where there are empty branching nodes. We suspect many syntax students will prefer the nice empty nodes option when there are empty nodes … but it can cause some messes with more complex trees, so there are times when it is advisable to simply live with some off-angle brackets. Please, please be willing to accept some mild imperfections in tree branch angles rather than spending hours and hours trying to fix it, at least for class papers.

If you want to include text under a node, you include a line break (two backslashes) and add the text, as with the verb below. If you want to include a triangle, you include the desired text in brackets followed by the code ", roof" as seen below.

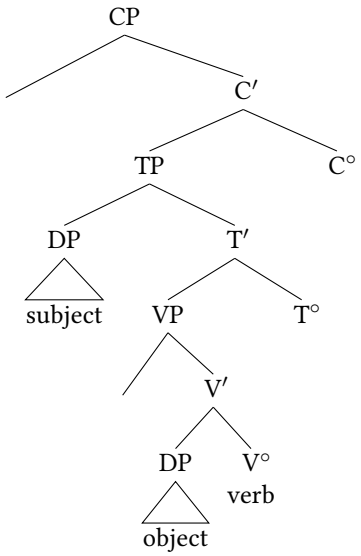
(29)



Some larger tree skeletons are commented out immediately below this paragraph in case you want a template to work from for a larger tree. Remember, to (un)comment batches of text in Overleaf, highlight the text and press Cmd-/ (Mac) or Ctrl-/ (PC).

Here's what a head-final tree looks like, where the difference is what you would expect: the node for the head follows the node for the complement of the head.

(30)

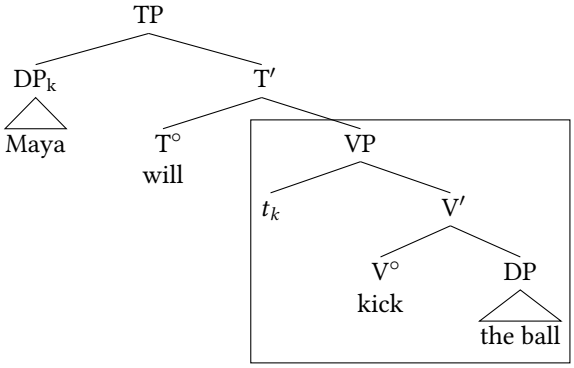


8.2 Annotating Trees

8.2.1 Boxes in Trees

At times people find it useful to draw a box around a subtree. The clunky code attached to the VP node below makes that happen, which you can copy and paste into your paper if you want to draw a box in a tree.

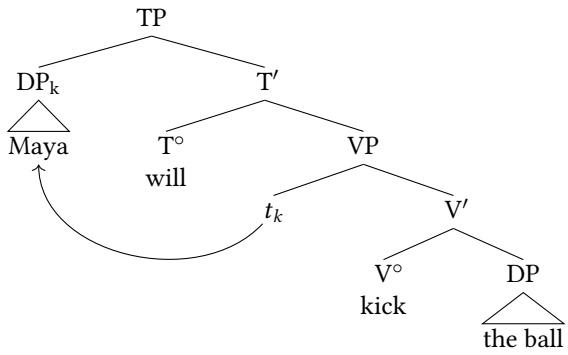
(31)



8.2.2 Arrows in Trees

Arrows are drawn in a tree with the command `\draw`. You must name specific nodes in your tree, and then you can draw an arrow from one node to the other in the last line of code in the tree. In the tree below we named the triangled DP "SUBJ" and the trace in Spec,VP as "SpecVP," and then used those names in the command to draw the arrow, which is the last line of code in the tree.

(32)

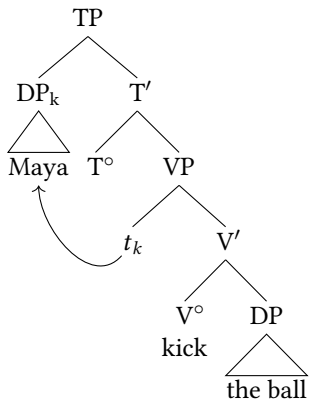


Please note, include arrows in a tree only when they serve to visually clarify the structure. They take extra time to draw, and traces/strikeout show the same exact information in a different notation. Be circumspect with how you spend your time here!

8.2.3 Spacing of Nodes in Trees

Forest will generally arrange the spacing around nodes for you, but it also allows you to specify it yourself. The "s sep" command controls horizontal spacing, "l" controls vertical spacing, and "inner sep" controls spacing around nodes. The tree in (33) is identical in structure to the one above in (32), but we have commented out the commands defining the node spacing. Uncomment it, and change the numbers there, and then recompile the document to see what the effects on the tree are.

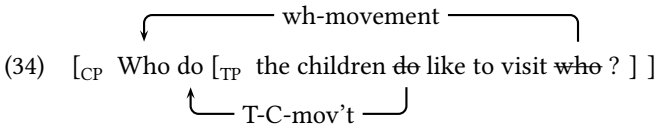
(33)



There is a lot of other kinds of formatting of arrows and trees that may be of interest to syntax students: we have another Overleaf template (the Forest Arrows Explainer that goes into much more detail for those who are interested.

8.3 Linear structures with arrows

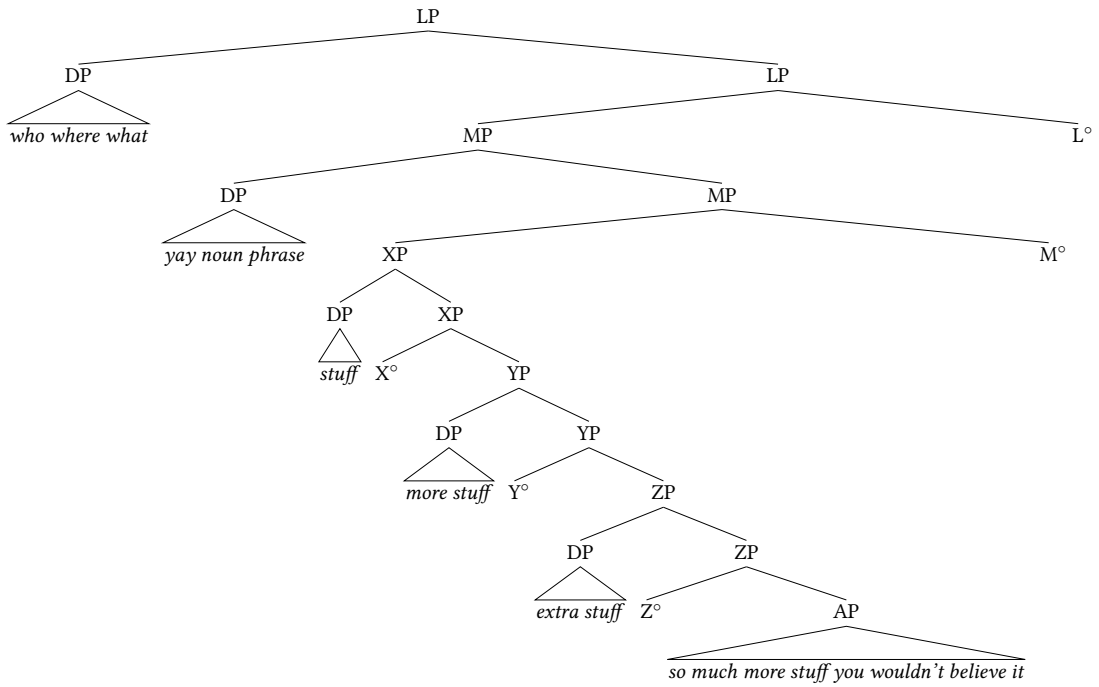
At times it is also nice to be able to present a linear syntactic structure that shows movement without drawing a whole tree, as in (34). If you are interested in these kinds of structures, see the Forest Arrows Explainer.

(34) 

8.4 What if a tree/figure is too big for the page?

Occasionally we run into issues where the tree we want to draw is too big to fit into the space on the page where we are putting it. There are a variety of ways that this can be dealt with in LaTeX, but we recommend using `\scalebox{number}{Tree/figure}`, where the number in the first argument of the command is the scale of the Tree (or other figure). In (35) we drew a tree that was too big for the page, and the scalebox command makes the whole tree smaller. You can change the number to see what effect it has on the tree. A number between 0 and 1 will shrink the tree, a number over 1 will embiggen the tree.

(35)



9 Semantics/Pragmatics Symbols

For Semantics and Pragmatics in Pomona LGCS you will mostly be using the tools introduced elsewhere in this guide. But it will also be useful to have a variety of particular symbols/graphs that are relevant for the formalisms used in Semantics and Pragmatics. We sketch them here in Table 2

descriptor	symbol	descriptor	symbol
logical 'and'	$\wedge$	logical 'or'	$\vee$
existential quantifier	$\exists$	universal quantifier	$\forall$
Negation	$\neg$	negation, other	$\sim$
denotation of 'this'	$[\![\text{this}]\!]$	alternate denotation of 'this'	$\|\text{this}\|$
long right arrow	$\longrightarrow$	right arrow	$\rightarrow$
right arrow w/ stroke	$\nrightarrow$	long right double arrow	$\Longrightarrow$
right double arrow	$\Rightarrow$	right double arrow with stroke	$\nRightarrow$
long left arrow	$\longleftarrow$	left arrow	$\leftarrow$
left arrow with stroke	$\nleftarrow$	long left double arrow	$\Longleftarrow$
left double arrow	$\Leftarrow$	left double arrow with stroke	$\nLeftarrow$
long left right arrow	$\longleftrightarrow$	long left right double arrow	$\Longleftrightarrow$
modal necessity operator	$\Box$	modal possibility operator	$\Diamond$
not equal	$\neq$	not approximately	$\not\approx$
approximately	$\approx$	lambda	λ
overscore, vinculum on X	$\bar{\text{X}}$ (copy/paste from here)	bullet between X and Y	$\text{X} \bullet \text{Y}$
intersection	$\cap$	union	$\cup$
superset of	$\supset$	not a superset of	$\not\supset$
subset of	$\subset$	not a subset of	$\not\subset$
superset of or equal to	$\supseteq$	not superset of or equal to	$\not\supseteq$
subset of or equal to	$\subseteq$	not subset of or equal to	$\not\subseteq$
element of	$\in$	not an element of	$\notin$
contains as member	$\ni$	does not contain as member	$\not\ni$
angle Brackets around X	$\langle \text{X} \rangle$	type theoretic notation	$\langle e \rangle$, $\langle e \langle e, t \rangle \rangle$
null / empty set	$\varnothing$	tick/checkmark	$\checkmark$
upwards arrow	$\uparrow$	downwards arrow	$\downarrow$

Table 2: Formal symbols useful for Semantics and Pragmatics Papers

10 Advanced tables

In §6.1, we showed you how to generate basic tables in LaTeX (if you haven't read through §6.1 yet, go back and do that now). This section provides guidelines for a few (mostly stylistic) modifications that may be useful for you. For a comprehensive "how-to" guide, see Overleaf's fantastic tables tutorial.

10.1 Controlling basic table formatting

- **Manually define column width**. Thus far, our column width has been determined by the widest cell. By using `p{}` when specifying your parameter, you can manually define the width of each individual column, inducing line breaks and/or justified columns.

Now my columns are wider

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

Now my columns are narrower

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

- **Control vertical borders**. Simply remove any '|' symbols from your parameter where you don't want vertical borders:

I'm a table without (vertical) borders

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

- **Control horizontal borders**. Simply remove any `\hline` commands where you don't want horizontal borders:

I'm a table without (horizontal) borders

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

I'm a table without any borders at all

Header 1	Header 2	Header 3
cell 1	cell 2	cell 3
cell 4	cell 5	cell 6
cell 7	cell 8	cell 9

10.2 Combining columns and rows in tables

Columns and rows can be combined in a bigger cell using the `\multicolumn{}{}{}` and `\multirow{}{}{}` commands.

The `\multicolumn{}{}{}` command takes three arguments:

1. The number of columns to be combined
2. Delimiters and alignment of the resulting cell - i.e. do you want borders, and whether it's left-, center-, or right-aligned
3. Text to be displayed inside the cell

Below we see that the columns making up the top row have been combined into one using the `\multicolumn{}{}{}` command.

(36)

Pomona/Pitzer Linguistics Professors		
Name	Initials	Name in IPA
Carmen	CF	ˈkɑɹ.mən
Mary	MP	ˈmɛ.ɹi
Mike	MD	ˈmaɪk
Nicole	NH	nə.ˈkol
Robin	RM	ˈɹɑ.bən
Franny	FB	ˈfɹæ.ni

The `\multirow{}{}{}` command also takes three arguments, but note that the second one is different than for `\multicolumn{}{}{}`:

1. The number of rows to be combined
2. The width of the column
3. Text to be displayed inside the cell

In the table below, we see that the rows in the first column have been combined:

(37)

Subfield	Professor(s)
Sociolinguistics	Carmen Nicole Franny
Phonology	Mary Franny
Phonetics	Nicole Franny
Syntax	Mike Robin
Fieldwork	Mary Mike
Psycholinguistics	Robin

Notice in the code above that for each multi-row cell, it must be paired with the equivalent number of rows in any other columns.

11 Conclusion

Welp, that wasn't brief.

But we hope it helps you find your way around formatting of linguistics papers.