Chapter 1

ACAL Proceedings: Quick Reference Guide for LaTeX

Michael Diercks
Pomona College

Franny Brogan
Pomona College

Hazel Mitchley
Rutgers University

Here is where you write the abstract of the paper.

Change author. Change title. To appear in: Change Volume Editor. Change volume title. Berlin: Language Science Press. [preliminary page numbering]

Contents

1 Introduction

Note: This document is designed to be read as a template on Overleaf: double clicking on something on the right side of your screen will show you the source code to the left. So you can read the formatted document on the right of the screen, then look to the left to see the code that generated it. If this function stops working, just refresh the webpage.

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. Overleaf is a tool for writing in LaTeX. In Overleaf, the .tex document is in the middle side of your screen; this 'code' is **compiled** to generate typeset document is on the right, which can be downloaded as a PDF by clicking the 'Download PDF' icon at the top of the preview.

When writing a .tex document 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') are used to create different kinds of formatting.

Double click on this sentence in the preview and see where the cursor moved to in the .tex document to the left of this preview. Then read immediately below this paragraph to see what a "comment" is using the % symbol.

1.1 What is LaTeX and why am I using it?

These are very good questions, please check out the ACAL website for more complete answers. The main reason why we write in LaTeX is that it is required by Language Science Press, the press that publishes ACAL proceedings. We discuss why on our website. A few basic comments are in order, however, so that our explanations below are interpretable.

Aside from directly annotating text with formatting commands, another difference between apps like Microsoft Word and LaTeX is that LaTeX is more similar to the code that makes Word work, 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. Overleaf is just one of these apps, which we have chosen for this template because it is convenient in many ways (though it has restrictions too, e.g. requiring internet access).

LaTeX itself has a limited degree of functionality - it does many things, but not all the things. As a result, many people have created packages—which are essentially plug-ins, or add-ons—which allow LaTeX to do some extra stuff. Much of the explanations that follow use packages designed by linguists that allow linguists create formatting in their papers that matches their needs: numbered examples, trees, IPA symbols, charts, etc.[1]

1.2 What is this document?

This is a Quick Reference Guide for writing proceedings papers in LaTeX for the Annual Conference on African Linguistics. This guide is published as an Overleaf template.

- This guide provides examples of formatting you may need for an ACAL proceedings paper, e.g. trees, numbered examples, etc.
- A copy of this project is now in your Overleaf account (i.e. it belongs to you), and you can now edit this document 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 ACAL Proceedings Paper Template offers the same advantages without this (long) explanatory document.
- Don't worry about breaking anything - you can always open this template again on Overleaf to see this initial version again. So edit away!

[1] At the very left of your screen, you will see a list of files that make up the LaTeX project. All .tex documents begin with a **preamble** (the part of the .tex file before the command beginning the document at the top of the text in the .tex file). In this template, we chose to simplify by having the preamble refer to two files that are listed on the left side of your screen: AcalProceedingsTemplatePackages.tex, and AcalProceedingsTemplateCommands.tex. This is where we added the packages we use and the commands we designed for this template (or borrowed from other scholars). You can click on those files to see what things look like in there, but if you are a beginner, we recommend not worrying about that until later.

[2] This also means you will probably want to rename the document - you can do this by hovering your cursor over the title "ACAL Proceedings 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 AcalProceedingsQuickReference.tex file in the leftmost panel by clicking on the down arrow.

2 Some basic things you will want to know

For a full LaTeX tutorial, see Pedro Martin's tutorial. But this section *very* quickly outlines things you may want to know how to do to write ACAL proceedings papers.

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

The chart in Table 1 is structured with the most general sorts of text formatting first, with linguistics-specific symbols/formatting at the bottom. Note that the LSP style guide forbids the use of underlining, so do not format anything with underlining.

Table 1: Selected commands for common ACAL formatting needs

Symbol/Annotation	Example	Code
Ellipsis	…	`\dots`
Underscore	X_Y	`X\textunderscore{}Y`
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}`
Circle something	(something)	`\circled{something}`
Highlight something	something	`\hl{something}`
Null	∅	`\nothing`
Hash	#	`\#`
Trace with index	t_k	`\tr{k}`
Bar-level node	X′	`X\xbar`
Head Node	X°	`X\xhead`
Underline	*do not use*	*do not use*

- If you want to make a bulleted list, look at how this list is formatted in the .tex document using the "itemize" environment. Notice that `\begin{itemize}` starts the bulleted list, `\item` introduces a bullet, and `\end{itemize}` ends the bulleted list.
- As you've already seen if you are paying attention to the .tex document on

the left of your screen, sections, subsections, and sub-subsections 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.[3]

2.2 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. 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. If you still cannot figure out how to solve your LaTeX puzzle, contact the ACAL LaTeX team at acal.latex@gmail.com.

2.3 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 - use one for a single quote 'like this' and two for quotation marks "like this." For reference, the grave accent is the one that appears on the mid vowel here: ò.
- 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 visual warnings that pop up as you type if you have left a bracket open. (To see this,

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

you can try to highlight something with `\hl{}` in the preceding sentence and leave off the final bracket, Overleaf will highlight text in yellow until the final bracket is added.)

- 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:

(1) This example is auto-numbered - if you uncomment the example above that is in green in the .tex document (by deleting the % before the example) and recompile, the numbering will change.

The `gb4e` package that Language Science Press uses for numbered examples allows for inter-linear glossing and translations:

(2) 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 (3):

(3) Wekesa se-a-la-ba a-kula ka-ma-indi ta.
Wekesa NEG-SM-PST-be SM-buy 6-6-maize NEG
'Wekesa will not be buying maize.'

Some things to notice in the .tex document for the examples above - an example is started with the `\ea` and ended with `\z` - omitting either of these will generate an error. The language and gloss lines are started with the command `\gll`, with line breaks (`\\`) ending both the language line and the gloss line. The translation line is introduced with the command `\glt`, and notice that there is no line break

at the end of the translation line. If you want to label an example as coming from a specific language, Language Science Press places the language name above the example, as is commented out in example (3).

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

(4) a. This is the first example.
b. hii ni m-fano w-a pili, kwa Ki-swahili
this is 3-example 3-Assoc second, of 7-Swahili
'This is a second example, in Swahili.'

Again, some details to notice in (4): first, notice the commands `\begin{xlist}` and `\end{xlist}` that begin and end the sublist inside the example. It is important that the sublist be generated *inside* the numbered example, so `\begin{xlist}` comes after `\ea` and `\end{xlist}` comes before `\z`. Each example inside the sublist is introduced with the command `\ex`, and each of these examples uses normal gb4e formatting. So you can see that the commands to format (4b) are the same as were used to generate (3).

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). However, in Language Science Press trees are generated as figures (see §8, and tables are listed as tables (see §7).

Sebastian Nordhoff at Language Science Press has created a helpful detailed document about how to format numbered examples: if you want/need more detail than we outline here, you can consult that document.

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 (3) and (4) 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" has some predetermined portions (according to LSP's style guide) as well as a portion

where you use some text description to describe the thing you are labeling:

- Section label: `\label{sec:author:yourlabel}`
- Table label: `\label{tab:author:yourlabel}`
- Figure label: `\label{fig:author:yourlabel}`
- Example label: `\label{ex:author:yourlabel}`

2. You must then use one of these commands to create the cross-reference.
   - Section reference (will read '§#' in output):
     `\sectref{sec:author:yourlabel}`
   - Table reference (will read 'Table #' in output):
     `\tabref{tab:author:yourlabel}`
   - Figure reference (will read 'Figure #' in output):
     `\figref{fig:author:yourlabel}`
   - Example reference with parens — note caps (will read '(#)' in output):
     `\REF{ex:author:yourlabel}`
   - Example reference without parens — note caps (will read '#' in output):
     `\ref{ex:author:yourlabel}`
3. Some examples:[4]
   - the Swahili example above can be referenced as (4b). If I wanted to refer to it without parentheses, I can use this command: 4b.
   - The aforementioned examples occur in §3, which describes how to use numbered examples.
   - In §8, Figure 1 shows how to draw a syntax tree.
   - In §7, Table 4 shows how to center-align text in cells of a table.
   - These are some selected examples - throughout the entire document you will see these kinds of cross-references being utilized, so you can look at numerous examples if you so wish.

Notably, you can only use any of the reference commands (such as the `\REF{LabelName}` command for numbered examples) with labels that already exist somewhere in your paper - attempting to build a cross-reference to a label that

[4] Look at the .tex on the left to see how to write the commands, and click the hyperlinks in the cross-references to take you to the portion of the document where the label is created.

does not exist will not stop your document from compiling, but it will result in an additional 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.

5 Using IPA symbols

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

This template is built on Language Science Press's template, which which allows you to enter IPA symbols directly into your .tex document and it appropriately typesets it when creating a PDF. 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.[5]
- 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.[6]
- The web app shapecatcher does something similar; you draw the symbol you want and it identifies possible matches and their unicode identifiers.

6 Citing References

This section is not a proper introduction to all of how citations and bibliographies work in LaTeX. Rather, we attempt to give you the basics you need to know

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

[6] If detexify says "mathmode" below the command, it means that you should put the command between $ signs - i.e. $\your.command$

in order to generate citations and a bibliography for an ACAL proceedings paper.

6.1 How bibliographies work

You will notice on the left side of the Overleaf display there is a list of files. The file titled "localbibliography.bib" is the bibliography file, and contains references that can be cited in your document. Click on it to see what a LaTeX reference list looks like, and then click on "AcalProceedingsQuickReference.tex" to come back here.

As you write, you can type commands into your text which will do two things: first, they will generate citations in the text to the paper you cite, and second, when your document compiles they will add that reference to your bibliography at the end of your paper. So if you want to make reference to Bresnan & Mchombo (1987), enter the citation command as we did in the .tex document, which references the file bibliographic entry in "localbibliography.bib" and generates the appropriate citation here in this paragraph, and adds the Bresnan & Mchombo paper to the bibliography/reference list at the end of this document.

The subsections that follow explain in more detail how to do this in your own paper.

6.2 Building a bibliography in a .bib file

There are many ways to build your reference list: we outline two major ones here.[7]

First, if you have already created bibliographic references, you can use Nordhoff (2018) (which is located here) to convert those references to a format appropriate for LaTeX. You can then copy and paste those references into "localbibliography.bib" which makes them available to cite in your paper.

Second, you can download a reference manager, which produces reference lists that look like what you see in "localbibliography.bib" but which gives you a user-friendly interface. If you are using a Mac, BibDesk is an excellent application that will manage your citations - a brief introductory video is available on

[7] If you consistently write in LaTeX you eventually would build a large .bib file that you use on every project; our goal in this explanation is mainly to help novices learn enough to write their current paper.

Youtube. If you are using a PC, we recommend using Jabref, which has the same functionality. Those reference managers will allow you to build a .bib file, which can be copied/pasted into "localbibliography.bib."

6.3 What is a cite key?

If you use the Bib converter, you will notice that the first line of a bibliographic entry is auto-generated as "AuthorDate" which is "Bloomfield1925" in the example below (which is the first example in the Bib converter sample).

@article{Bloomfield1925,
author = Bloomfield, Leonard,
journal = Language,
number = 4,
pages = 130–156,
title = On the sound-system of central Algonquian,
volume = 1,
year = 1925
}

This circled text above is the "cite key," which is an identifier that you use to cite that reference (this can in fact be any string of text that you choose).

6.4 Citing sources

The two most important citation commands are `\citet{}` and `\citep{}`. The cite key of the reference goes between the curly brackets.

- For 'Author (Date)' format, use `\citet{}`.
  e.g.: writing `\citet{Lahiri2000}` produces Lahiri (2000), because the cite key 'Lahiri2000' refers to the associated reference in the localbibliography.bib file you can see in the file structure to the left.
- For a parenthetical citation, use `\citep{}`.
  e.g.: writing `\citep{Lahiri2000}` produces (Lahiri 2000), again because the cite key 'Lahiri2000' refers to the associated reference in the localbibliography.bib file you can see in the file structure to the left.

Both of these commands can be modified to add extra information.

- To add a page number after the year, put the page number in square brack-

ets *before* your curly brackets.
e.g.: `\citep[87]{Doke1923}` produces (Doke 1923: 87)

- To add information before the citation, put this in square brackets before the brackets with the page number.
  e.g.: `\citep[Zulu,][87]{Doke1923}` produces (Zulu, Doke 1923: 87)
- If you want information before the citation, but without a page number, just leave the page number blank.
  e.g. `\citep[Zulu,][]{Doke1923}` produces (Zulu, Doke 1923)
- Multiple cite keys separated by commas will produce multiple citations, assuming that you have put references associated with those cite keys into localbibliography.bib.
  e.g. `\citep{Doke1923,Lahiri2000}` produces (Doke 1923; Lahiri 2000)

You can check out this helpful reference sheet that gives many details about citation commands for any additional variants that you want.

7 Basic tables

Tables are very useful for formatting data and examples, but counter-intuitive to produce in LaTeX as compared to producing them in familiar word processors. This section lays out the basics; check out this useful wiki on how to construct LaTeX tables if you want some addition information, or Overleaf's tables tutorial.

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. Per LSP's guidelines, tables should not have vertical borders in ACAL Proceedings papers. Let's take a look at some simple examples below. Please note, tables are moved around by LaTeX in the typeset document on the right of your screen to where they typeset most naturally, so you may have to look to the next page for a table. Also, please pay attention to the commented-out text that appears in green in the .tex document around each table's code, because we have explained what each part of the code for the table does.

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

Table 2 below.

Table 2: A descriptive caption here

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

Table 3 is a table that's identical to the one above, but with 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.

Table 3: This table separates headers with a double line

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

Finally, Table 4 is a table identical to the one above, but with text center-aligned.

Table 4: This table has center-aligned text

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

In general, LSP's style guide disallows both horizontal borders and vertical borders between cells in a table (apart from horizontal border separating the headers from the rest of the cells, which is coded with `\midrule`.

7.1 If a table runs off the page

It is not uncommon to generate a table that can't fit straightforwardly on to the page, like table 5. You can fix this by using the `\fittable{}` command, as

exemplified in table 6. If you look at the .tex document, you'll see that `\fittable{` comes *before* `\begin{tabular}`, and the closing brace } for this command comes after `\end{tabular}`.

Table 5: This table is too long

Header 1	Header 2	Header 3	Header 4
Long piece of data 1	Long piece of data 2	Long piece of data 3	Long piece of data 4
Long piece of data 5	Long piece of data 6	Long piece of data 7	Long piece of data 8

Table 6: Long table with the \fittable command

Header 1	Header 2	Header 3	Header 4
Long piece of data 1	Long piece of data 2	Long piece of data 3	Long piece of data 4
Long piece of data 5	Long piece of data 6	Long piece of data 7	Long piece of data 8

We don't go into more depth here in order to keep the length of this overview constrained, but LaTeX allows for endless tinkering with the formatting if you need those kinds of adjustments in your paper. For a comprehensive "how-to" guide, see Overleaf's fantastic tables tutorial.

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.[8]

The code for trees uses a standard bracketing notation, so every open bracket has to have a corresponding closing bracket later. This also means that you have to be careful not to forget the second curly bracket at the very end of the code for the tree, matching the open bracket that is part of the `\forest{}` command.

[8] It's worth noting that Pedro Martin's tutorial does not use the `forest` package, instead using `qtree`. This means that the commands used to build trees here are different than what he uses in his tutorial.

Figure 1: First example of a tree with Forest. Double-click on this to take you to the code in the .tex document to the left.

The code for Figure 1 is formatted so that each level is on the same line. This is not necessary; the code for Figure 2 has the same exact bracket structures, just with line breaks and indents to visually clarify the levels of the trees. You'll see in the preview that these trees are identical, despite the code in the .tex document being formatted differently.

Figure 2: Same tree as Figure 1, with code formatted differently in the .tex document on the left

Please format your code something more like Figure 2 than Figure 1, largely because as the trees get larger it can get tougher and tougher to interpret your code and editors may need to wrestle with that at some point. We encourage you to try editing the trees above and see what happens!

An important note: do not include empty lines in the code for the tree - as you can see, if you add an empty line to the trees above and attempt to recompile, you will get an error. `forest` attempts to parse all lines so empty lines confuse it. But extra white spaces within a line do not make a difference to `forest`.

In the example below, we've drawn a slightly bigger tree, which includes an empty node (which is created with an open-close bracket sequence with nothing inside it.

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 Figure 4, where the top node in the tree is specified as having "nice empty nodes:"

This is really only helpful for a tree where there are empty branching nodes. We suspect many syntacticians will prefer the nice empty nodes option when there are empty nodes ... but it can cause some messes with more complex trees, so

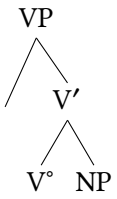


Figure 3: Slightly bigger tree

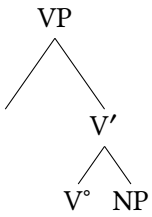


Figure 4: Example of "nice empty nodes"

there are times when it is advisable to simply live with some slightly off-angle brackets (unless someone is a master LaTeX user, but this document is not for them).

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 square brackets followed by the code ", roof" as seen in Figure 5. (The comma is part of the code, don't leave it out.)

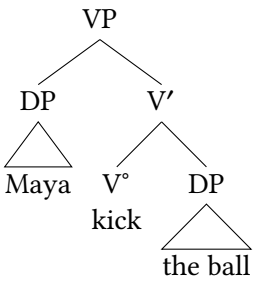


Figure 5: Example with texts in multiple nodes

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).

A head-final tree is illustrated in Figure 6, where the difference is what you would expect: the node for the head follows the node for the complement of the head.

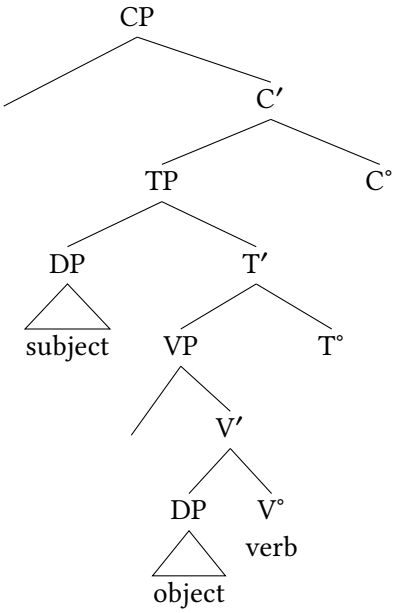


Figure 6: Example of head-final tree

8.2 Annotating Trees

8.2.1 Basic 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 in Figure 7 we named the triangled DP "SUBJ" using the command ", name=SUBJ"and the trace in Spec,VP as "SpecVP" using the command ", name=SpecVP". (Like for triangles above, the comma is part of the command, so don't leave it out!) We then used those names in the command to draw the arrow, which is the last line of code in the tree.

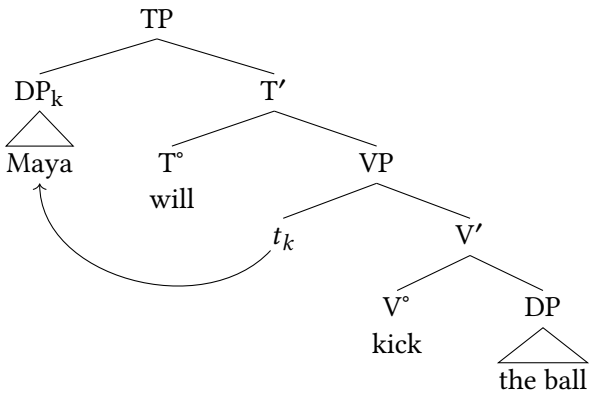


Figure 7: Example of arrows in a tree

8.3 Want to get more fancy with your trees?

There are a host of additional commands and packages you can use to decorate your trees and/or re-arrange the spacing of nodes, branches, etc. We don't go into detail here because this document is aimed at those just starting to use LaTeX. If you have a particular need for your paper with respect to tree-formatting, you can use Google to find a solution, you can read Guido Vanden Wyngaerd's quick-start guide to `forest` for linguists, or you can email the ACAL LaTeX committee at acal.latex@gmail.com.

9 Phonology Things

9.1 SPE Rules

Typing up SPE-style rules in Word/Google Docs can be annoying, 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.

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:

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

Keep in mind that you can 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 (6) above.

As you may know, the English Aspiration Rule in (6) 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, by using the command `\oneof{Context 1 \\ Context 2}`. Double click on rule in (6) to check out the code in the .tex document.

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

Click here to see a template of this rule (in the comments of the .tex doc-

ument) which you can copy-paste into your document to edit.

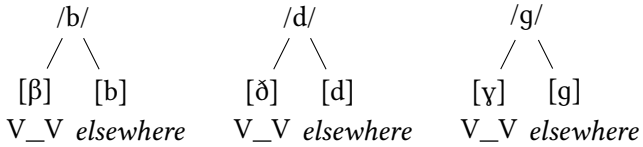
9.2 Phonemicization diagrams

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:

(8) A simple set of phonemicization diagrams

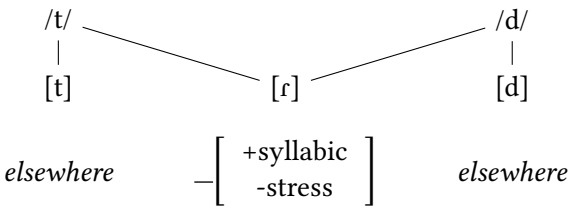
(9) A slightly more complex set of phonemicization diagram



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 (10).

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

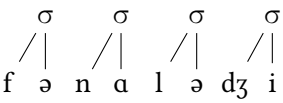


9.3 Autosegmental diagrams

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.

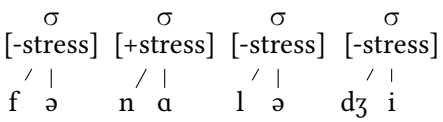
For the sake of simplicity, this template typesets autosegmental diagrams using the `TikZ` package, which we introduced in §9.2. Let's start with a basic example that illustrates the syllabic division of the word *phonology*, [fə.nɑ.lə.dʒi].

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



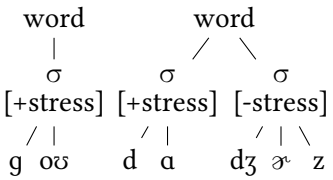
Now, using the `\stackanchor{}{}` command in the `stackengine` package, we can add a **stress tier** under our syllable nodes to show that [fə.nɑ.lə.dʒi] has antepenultimate stress. Notice that we've also shifted all our nodes (except for the left-most group) over .5 on the x-axis to make room for the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

(12) 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]:

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



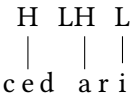
To build more tiers, simply add additional sets of nodes and association lines accordingly.

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:

(14) Underlying tonal associations for /cedari/

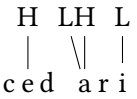
It's recommended that you play around with spacing and node placement here. Similar to the procedure laid out in §9.2, we like to start by anchoring the segments themselves and subsequently placing the tones to match the x values for their corresponding vowels. Note that one advantage of using the `TikZ` 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**:

(15) 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:

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



9.4 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:

(17) 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:

(18) 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):

(19) Simple tableau with incorrect constraint ranking

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

10 Conclusion

Our Quick Reference Guide is quite long! Ah the irony. But we hope it helps you find your way around formatting of ACAL Proceedings papers. There are a host

of relevant resources online, and if you hit a confusing puzzle you can almost always google "my problem latex" and find relevant articles on Stackexchange and other similar sites. This guide is not meant to be comprehensive at all, but rather is meant to introduce novices to the central issues necessary to write a paper for the ACAL Proceedings. If you have questions, please feel free to contact the ACAL LaTeX committee at acal.latex@gmail.com.

Thank you for your efforts to keep our proceedings Open Access with Language Science Press!

Abbreviations

It is a required part of volumes with Language Science Press that an abbreviations list be included, so you cannot omit this section.

Acknowledgements

Paper acknowledgements must be listed here, not in a footnote in the paper. This Quick Reference Guide and the corresponding Paper Template borrow directly (with permission) from corresponding documents written by Michael Diercks and Franny Brogan for linguistics students at Pomona College. Diercks' original LaTeX teaching materials were created with assistance from Claire Halpert, Nico Baier, Jason Zentz, Maddy Bossi, and Mica Clausen. Our gratitude to the ACAL LaTeX committee for their input to this template (especially Peter Jenks, Ken Steimel, and Matthew Faytak).

References

Bresnan, Joan & Sam Mchombo. 1987. Topic, pronoun, and agreement in Chichewa. *Language* 63. 741–782.

Doke, Clement M. 1923. A dissertation on the phonetics of the zulu language. *Bulletin of the School of Oriental and African Studies* 2(4). 687–729.

Lahiri, Aditi (ed.). 2000. *Analogy, leveling, markedness: Principles of change in phonology and morphology* (Trends in Linguistics 127). Berlin: Mouton de Gruyter.

Nordhoff, Sebastian. 2018. *Bibtex generator*. Follow the link to generate BibTeX. http://glottotopia.org/doc2tex/doc2bib.