Using Overleaf to make Trees

Alan Munn

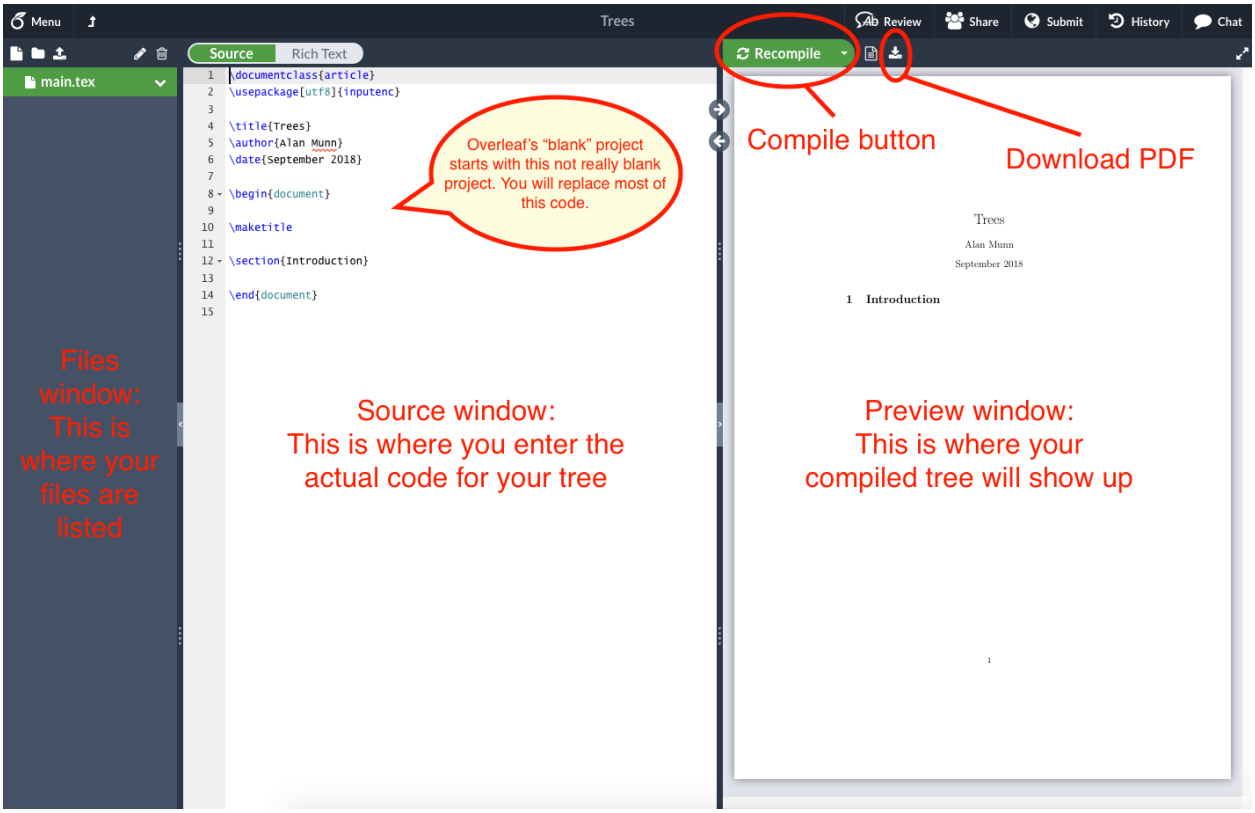
September 2018

1 Introduction

The Overleaf site http://overleaf.com/ is a cloud-based LaTeX system. LaTeX is a complete typesetting system which converts a plain text document with embedded commands (called a *source file*) into a fully formatted PDF document. The Overleaf site can be used to create complete documents using LaTeX, but in these notes I will simply explain how to use it to produce PDF (or other format) images that can be easily placed into other documents such as MSWord or Powerpoint. The advantage of this is that it can be used to quickly draw nice trees to include in Word or Powerpoint documents without having to actually learn LaTeX fully or install it on your own computer. In this guide I will explain how to use the Overleaf to make nice trees.

1.1 Using Overleaf

Overleaf is a free cloud-based document sharing system, very similar to Google Docs: you can create documents, save them in the cloud and share them with others and sync them with your Dropbox folder, if you use Dropbox. You sign up with your email address and then access the site through any web browser, just like Google Docs. When you first open an new Overleaf project, you should see something like the following:



There are three main parts to this display: on the lefthand side will be the list of files/projects/-folders that you have created. The middle part is called the *source window*, and this is where you enter the actual code to generate your tree. The righthand part is the *preview window*, and this shows the output of the compiled code.

The first time you login to Overleaf it displays a sample source document, so your preview may look slightly different from the image above.

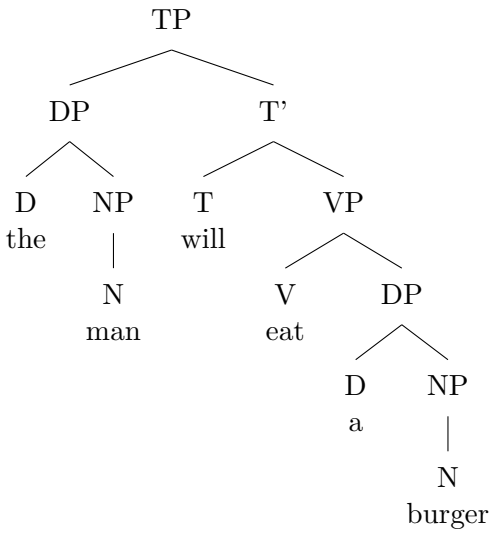
I have created a basic template document that you can use on Overleaf. Click on the following link and it will lead you to the document: LIN 434 Tree Template.

Now you are ready to input a simple tree into the system. Copy the code from the template document and paste it into a new file the source window, completely replacing any other text that might already be there.

Listing 1: A simple document

```
\documentclass[12pt,border=5pt,varwidth=true]{standalone}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[linguistics]{forest}
\begin{document}
\begin{forest}
[TP
    [DP [D\\the ] [NP [N\\man ]]]
    [T'
       [T\\will ]
       [VP
          [V\\eat ]
          [DP [D\\a ] [NP [N\\burger ]]]
       ]
    ]
]
\end{forest}
\end{document}
```

Now click on the Recompile button, and you should see the following:



If all has gone well, you have successfully created your first tree. You can now download that tree as a PDF file by clicking on the Download PDF icon located beside the Compile button and insert it as an image into your Word document. (See the Appendix at the end for how to do this.) Now that you've created a simple tree, I'll explain a bit more how the LaTeX and `forest` notation works.

2 Some basic things to understand about LaTeX

For people who are used to using a word processor like Microsoft Word, the idea that your source document is separate from your formatted document is a bit odd at first. The basic thing to understand about a LaTeX document is the following:

Formatting is separated from content!
Formatting should be associated with a semantics!

In reality what does this mean? Well basically, if you want LaTeX to format something a particular way, you should tell it what *kind* of thing it is (give it a semantics) and then associate the formatting with the kind rather than the actual thing.

Every LaTeX document begins with a statement of what kind of document it is. This is the `\documentclass` command. In the example above, the `\documentclass` has been set to `standalone`, which is a special class designed to make standalone images rather than full documents. This is the best document class for the purpose of making single images to be inserted into Word documents. The main font size of a document is set by the document class itself, and is set by putting it in square brackets before the `{class-name}` part of the command. So the first line in our sample document means "Use the `standalone` class with the font size `12 pt`."

Although there are many basic things that LaTeX can do, many useful things are implemented as separate *packages*. Each package is loaded separately in the what is called the *preamble* of the document. Extra packages are loaded using the `\usepackage{}` command. In the sample document above, we are loading a few packages. The commands `\usepackage[T1]{fontenc}` and `\usepackage[utf8]{inputenc}` load packages that are used to allow the use of accented characters in your document, and you should generally always use them. Next we load the `forest` package. This is the package that is needed to draw trees in LaTeX.

You should add all of these lines to every tree document you create.

3 Tree notation

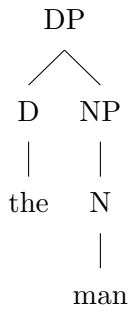
The `forest` package allows you to enter a labelled bracketing of a tree and output a tree. The rules for `forest` bracketing are simple:

(1) *Bracketing Rules*
   a. Every tree begins with `\begin{forest}` and ends with `\end{forest}`.
   b. Labels are suffixed to brackets; you can't have a space between a bracket and a label.
   c. All brackets must be balanced, i.e., every left bracket needs a closing right bracket.
   d. There should be no blank lines inside the `{forest}` environment.

Here, therefore, is the code for a simple tree, which produces the output in (2b).

(2) a.
```
\begin{forest}
[DP [D [the] ] [NP [N [man] ] ] ]
\end{forest}
```

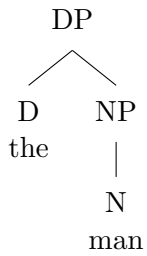
b.



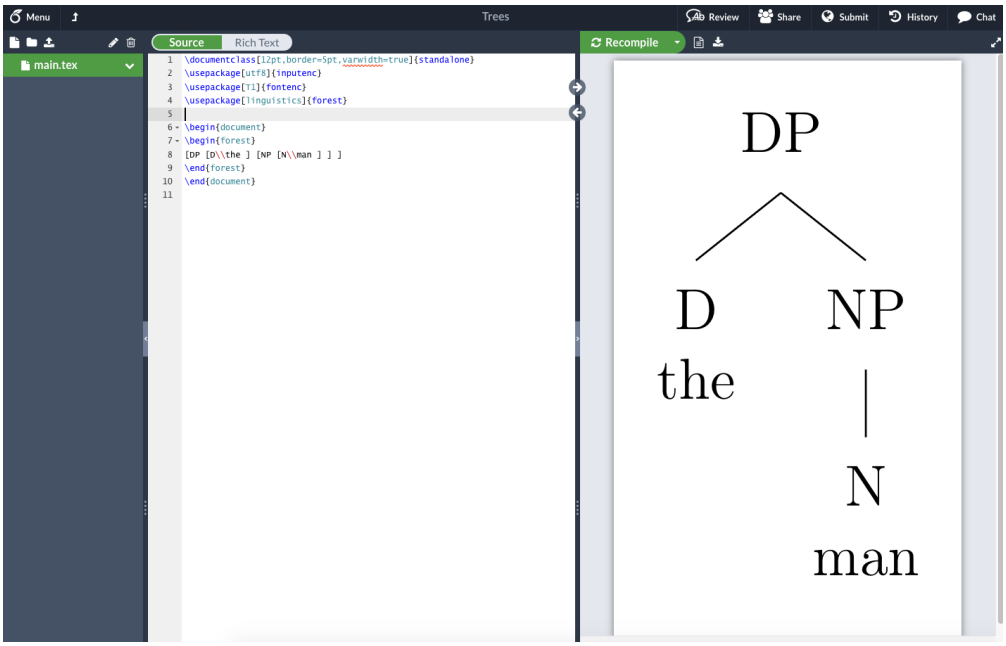
One thing to notice in this tree is that there are branches between the terminal nodes and their labels. If you don't want this (and I generally don't, since the labels are labels of the words) you can include the terminal as part of the label by connecting the two with a linebreak, which in LaTeX is `\\`. Tree (2) could then be coded as (3a) and appear as (3b).

(3) a.
```
\begin{forest}
[DP [D\\the ] [NP [N\\man ] ] ]
\end{forest}
```

b.



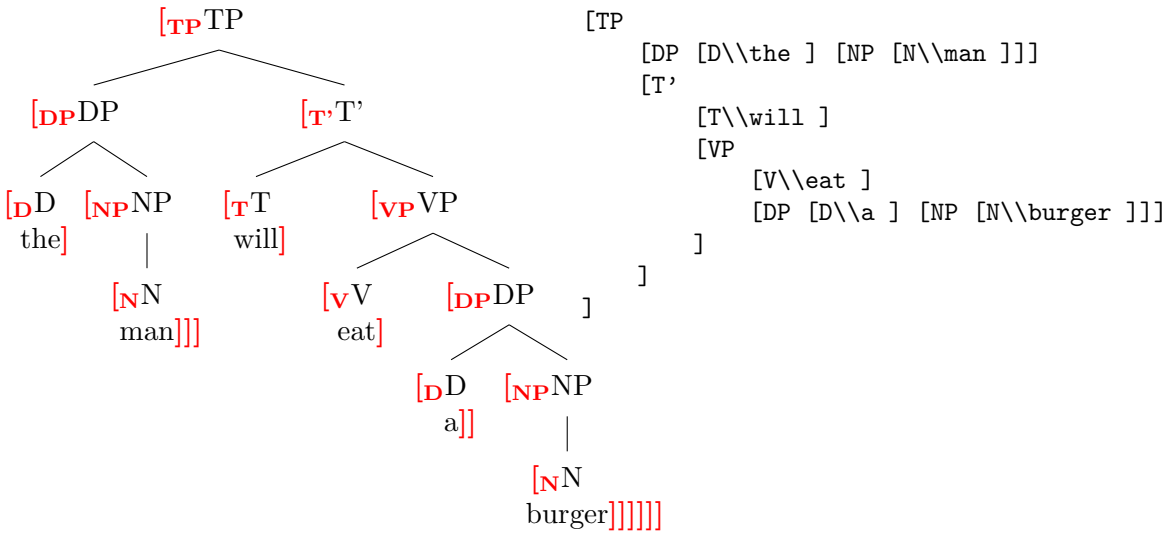
If you cut and paste this tree into your document (replacing everything between `\begin{document}...\end{document}`), and click on the compile button, you should see the following:



If your display doesn't look exactly like the one in the picture because the image in the preview window is too small or too large, you can change the size of the preview to fit the window either by height or by width. The buttons to do this are contextual and will appear if you move your cursor to the top left corner of the preview as shown below:

3.1 How to convert a tree to brackets

It's helpful to understand how to convert a tree into a bracketed structure. The basic idea is that every node in the tree gets a pair of brackets, and the label for the node is the label for the left bracket of the pair. So you should draw your tree by hand and then starting from the top, create brackets around the beginning and ends of nodes. For the sample tree shown above, we could do the following. In this tree, I've shown the bracketing in red as well as the tree.

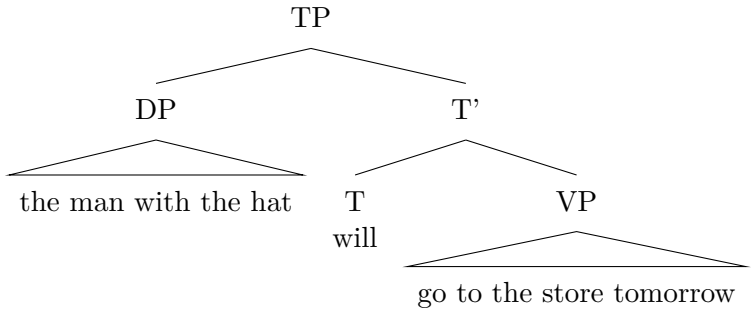


3.2 More `forest` commands

The `forest` package can also make closed triangles for grouping nodes under a single node. For this you need to use the add the `roof` command. Here is an example. The text you want to be under the triangle is enclosed inside `{...}` as the node label. This is then followed by a comma (which is important) and the `roof` command.

(4) a.
```
\begin{forest}
[TP [DP [{the man with the hat},roof  ]]
            [T'
                [T\\will ]
                [VP  [{go to the store tomorrow},roof  ] ]
            ]
      ]
\end{forest}
```

b.



4 Some useful latex commands

In order to format the text in a tree, it is necessary to be able to use a few LaTeX commands. Almost all LaTeX commands begin with a backslash (\) character. If a command takes an argument, the argument is enclosed in braces (`{}`). Here are some basic commands that are useful for tree drawing.

(5) Some useful LaTeX commands

a. subscripts, e.g. t_i use `t$_{i}$`
b. superscripts e.g. t^i use `t$^{i}$`
c. to start a new line in a label, use `\\`
d. to keep words together as a group, enclose them in braces `{ }`
e. to display a curly bracket, precede it with a \ (e.g. to produce {some text} you need to type `\{some text\}`)

(6) Formatting commands

a. to make the font sans serif put `\sffamily` at the beginning of the document
b. to make something *italics* enclose it in `\textit{}`
c. to make something **bold** enclose it in `\textbf{}`
d. to make something SMALL CAPS enclose it in `\textsc{}`
e. to change the color to red use `\textcolor{red}{}` and put the text you want red inside the second set braces

5 Some examples

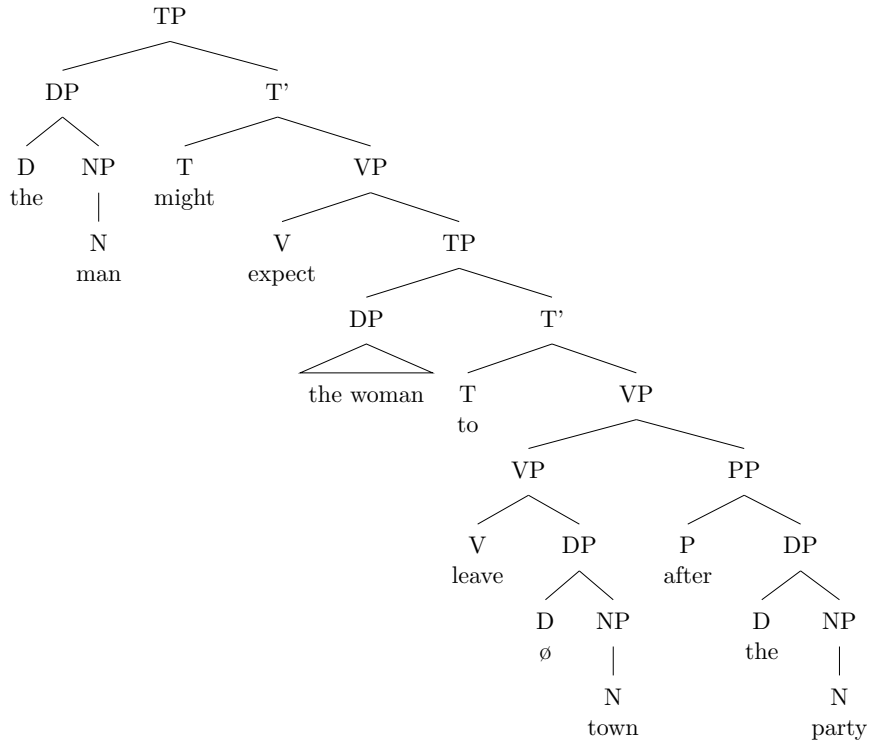
Here are some examples of more trees. For large trees it's helpful to write out the bracketed structure on separate lines with indenting to show the partial structure. This will allow you to find mismatched brackets more easily (and you *will* mismatch brackets…)

(7) a.
```
\begin{forest}
          [TP
          [DP [D\\the ] [NP [N\\man ] ] ]
          [T' [T\\might ]
            [VP [V\\expect ]
                 [TP [DP [{the woman},triangle] ]
                   [T'
                     [T\\to ]
                     [VP [VP
                          [V\\leave ] [DP [D\\{\o} ] [NP [N\\town ] ] ] ]
                          [PP [P\\after ]
                               [DP [D\\the ] [NP [N\\party ] ] ]
                          ]
                     ]]
                  ]
             ]
          ]
       ]
\end{forest}
```

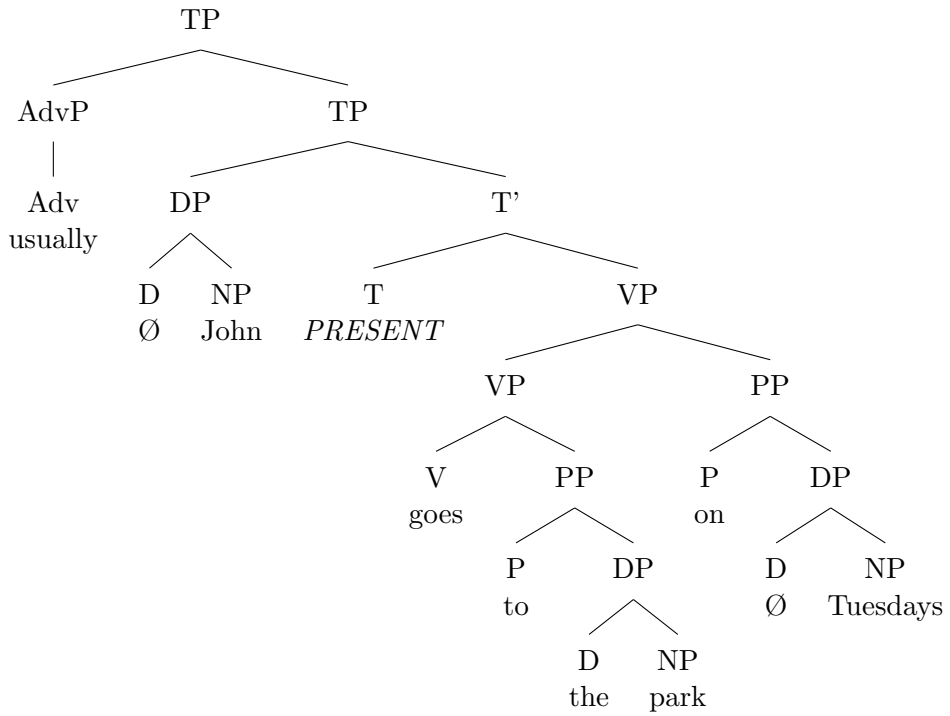
b.
```
\begin{forest}
        [TP
       [AdvP [Adv\\usually ] ]
       [TP
          [DP [D\\{\O} ] [NP\\John ] ]
          [T'
            [T\\\emph{PRESENT} ]
            [VP [VP
              [V\\goes ] [PP [P\\to ]
                              [DP [D\\the ] [NP\\park ] ]
                          ]
                    ]
                    [PP [P\\on ]
                         [DP [D\\{\O} ] [NP\\Tuesdays ] ]
                    ]
             ]
            ]
          ]
        ]
\end{forest}
```

c.
```
\begin{forest}
        [TP
       [DP  [NP\\Hanako ] [D\\ga ] ]
       [T'
          [VP
            [CP
              [TP
                [DP  [NP\\Yoshi ] [D\\ga ] ]
                [T'
                  [VP [DP  [NP\\Yuki ] [D\\-o  ]  ]
                       [V\\mi ] ]
                  [T\\ta ] ] ]
              [C\\to ]
            ]
            [V\\it ]
          ]
          [T\\ta  ]
       ]
    ]
\end{forest}
```

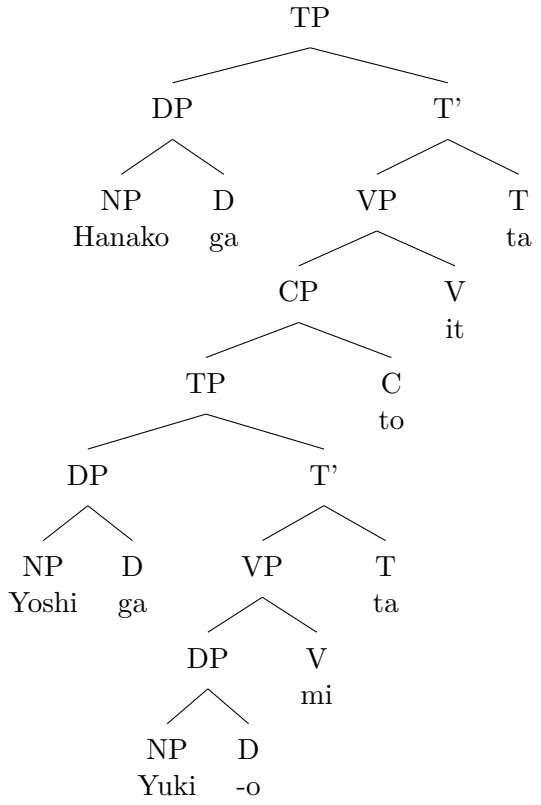
(8) a.



b.


TP
AdvP
TP
Adv
usually
DP
T'
D
Ø
NP
John
T
PRESENT
VP
VP
PP
V
goes
PP
P
on
DP
P
to
DP
D
Ø
NP
Tuesdays
D
the
NP
park

c.


TP
DP
T'
NP
Hanako
D
ga
VP
T
ta
CP
V
it
TP
C
to
DP
T'
NP
Yoshi
D
ga
VP
T
ta
DP
V
mi
NP
Yuki
D
-o

6 Really advanced stuff

LaTeX can do all sorts of really amazing things, and is especially good for representing mathematics. We can use this facility to make even fancier trees, and also make shorthands for them. For example, as noted above, using square brackets in LaTeX is complicated since the square bracket is used for other things within LaTeX itself. However, we can create a command which automatically puts something in square brackets (useful for annotating nodes with features.) It requires the `amsmath` package to be added to the included packages.

(9) Command to enclose something in square brackets

a. Command definition:
```
\newcommand{\SB}[1]{$[\text{#1}]$}
```

b. Command usage:
```
\SB{some text}
```

c. Output:

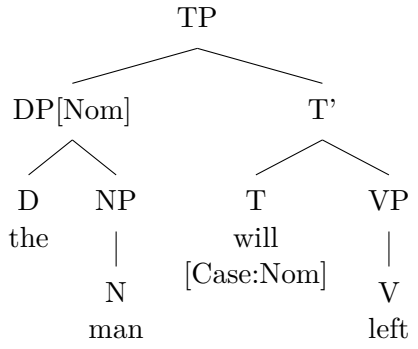
[some text]

If you put this command in the first line of your LaTeX source, you can then add features to trees using the `\SB` command, as in the following example:

(10) a.
```
\newcommand{\SB}[1]{$[\text{#1}]$}

\begin{forest}
[TP [DP\SB{Nom} [D\\the ] [NP [N\\man ] ] ]
[T' [T\\will\\\SB{Case:Nom} ] [VP [V\\left ] ] ] ]
\end{forest}
```

b.



If we want to put a bunch of features in a list, we can use the array command, and automatically expandable square brackets using the `\left`[and `\right`] commands. Here is an example:

(11) Feature bundle using an array

a.
```
$\left[\begin{array}{c}
\text{Tense}\\
\text{Nom}\\
\text{3PSg}\\
\end{array}\right]$
```

b. Output

$$\left[\begin{array}{c} \text{Tense} \\ \text{Nom} \\ \text{3PSg} \end{array}\right]$$

7 More information

These are just some examples of what can be done with LaTeX. They are not intended to be a complete introduction to all of its power. There are plenty of good introductions to be found on the web. For a very minimal introduction aimed at Mac users by me, see A Beginner's Guide to LaTeX (on the Mac). Another more extensive guide also for linguistics is by Adam Liter and can be found here: LaTeX workshop (for linguists). More general online books by Nicola Talbot can be found here: LaTeX Books from Dickinmaw Books. A very good printed book such as Marc van Dongen's *LaTeX and Friends* (Springer, 2012) might also be useful.

Appendix How to insert PDF images into MSWord documents

Once you've created your tree, you can easily insert it into a Microsoft Word document.

- On a Mac you can simply drag the PDF file onto your document or choose "Insert Picture from File".
- On a PC you can choose "Insert Object" and then select Adobe Acrobat Document and choose the file.