LaTeX reference management

Brian Buccola & Alanah McKillen
August 20, 2012
McGill University

This document describes how to manage and cite references in LaTeX. We begin by discussing LaTeX's native `\cite` command and document–internal method of encoding reference entries ("bibitems"). We then introduce a more robust and flexible method: using a package (`natbib`) for additional citation commands and style options, and BibTeX for standardized, external reference management.

1 LaTeX's native reference management

LaTeX provides a simple method of creating and citing reference entries ("bibitems") on the fly, with a clean and compact look. This is useful for a document with just a few references that you'll probably never need again, e.g., a class assignment, presentation, etc. Although this method is limited, its simplicity and speed are an advantage, and it's useful to understand and tease apart LaTeX's own reference system before diving into more advanced systems.

LaTeX provides a command called `\cite` for automatically citing references, and an environment called `thebibliography` for listing references, which are denoted by the `bibitem` command. The latter has the syntax `\bibitem{citekey}reference_data`, where the *citekey* is argument taken by the `\cite` command.

The following is a working minimal example:[1]

Example 1

```
According to \cite{chomsky1959} ...
...
\begin{thebibliography}{9}
    \bibitem{chomsky1959}
        Noam Chomsky.
        \emph{On certain formal properties of grammars}.
        1959.
        Information and Control 2, 137--167.
\end{thebibliography}
```

[1]The `9` in the first argument to `thebibliography` specifies the number of reference entries you have, but in a strange way: the actual number (9 here) is meaningless; what matters is the number of digits (1 here). Thus, `9` (or equivalently, `4`) means only one digit is required for each entry, i.e., because there are 1–9 total entries, whereas `99` (or equivalently, `37`) means two digits are required for each entry, and so on. In other words, you're specifying the number of places needed.

By default, this produces inline *bracketed numerals* for citations: "According to [1] . . . ". You can modify a bibitem's citation style with an option: `\bibitem[Chom59]{chomsky1959}` would produce "[Chom59]" instead of "[1]", both in the paragraph and in the reference list.

Note that the reference data for a bibitem is plain text with no structure, as far as LaTeX is concerned (linebreaks are ignored). This means (1) LaTeX cannot extract info about the author, journal, year of publication, etc., hence why the numeric citation style is so crude, and (2) it's up to you to remember to add all relevant data, as well as formatting, e.g., italics/emphasis.

As a solution to these difficulties, it's usually best to use a package like `natbib`, together with the BibTeX reference management system.

2 Package help and BibTeX

`natbib` provides an array of citation commands that effectively supersede LaTeX's `\cite` command by allowing you to cite the author and date, just the author, just the date, the date in parentheses, etc. In addition, `natbib` provides the command `bibliographystyle`, which allows you to automatically format all citations (inline and in the reference list) according to a style guide, provided you have the `.bst` style file. (Most TeX distributions come with `apa` and many others.)

Of course, `natbib` must somehow be able to figure out a reference's author, date, etc. from the reference entry. While it's possible to format a bibitem (LaTeX's native entry format) in such a way that `natbib` can parse and extract the data properly, a much easier way is to use BibTeX, which allows you to keep all your references in a single text file (`.bib` extension) organized in a standardized format.

BibTeX provides standardized organization of many different publication types, including `article`, `inproceedings`, `phdthesis`, and `unpublished`, each of which has its own set of required and optional fields. A BibTeX file can be created and edited with a simple text editor; however, an easier way is to use a helper application like BibDesk or JabRef as a frontend to your `.bib` file. These applications have all the required and optional fields programmed into them, so that you don't have to look them up.

The following is a minimal working example of a BibTeX file—let's call it `myrefs.bib`.

Example 2

```
@ARTICLE{chomsky1959,
  author = {Noam Chomsky},
  title = {On Certain Formal Properties of Grammars},
  journal = {Information and Control},
  year = {1959},
  volume = {2},
  pages = {137--167},
}
```

To use `natbib`, simply add the `natbib` package to your preamble, specify your `.bib` file, and specify the bibliography style you'd like to use (e.g., `apa`).[2]

Example 3

```
\usepackage{natbib}
...
According to \citet{chomsky1959} ... After \citeyear{chomsky1959},
\citeauthor{chomsky1959} went on to ...
...
\bibliography{myrefs}
\bibliographystyle{apa}
```

To compile your document—let's call it `mydoc.tex`—run the following:[3]

Example 4

```
pdflatex mydoc.tex
bibtex mydoc
pdflatex mydoc.tex
pdflatex mydoc.tex
```

What's going on here? Essentially: (1) the first `pdflatex` compile gathers up all the citekeys and the bibfile and writes them to `mydoc.aux`; (2) the `bibtex` command reads `mydoc.aux` and `myrefs.bib` to determine the reference list; (3) the second `pdflatex` compile inserts the reference list but without proper labels (you'll see things like "According to **??**"); (4) the final `pdflatex` compile fixes all labels.

Read the BibTEX documentation for further info. For a comprehensive list of all `natbib` commands and options, check out the `natbib` documentation. For a quick and easy cheat sheet, go to: `http://merkel.zoneo.net/Latex/natbib.php`.

3 Tips, tricks, and caveats

- Some bibliography styles decapitalize all letters other than the very first. For example, `title = {Quantification and ACD}` gets formatted as *Quantification and acd*. This is a design feature, not flaw: it leaves formatting to the style file, not the bibliography file. To hardcode capitalization into your bibliography entry, just surround the capitalized letter in curly brackets: `{ACD}`, `{N}ew {Y}ork`, etc.
- You can use standard LATEX commands for formatting (certain parts of) reference entries. This is necessary for special characters (`g\"{o}del` for "Gödel") and anything you need italicized (`An Analysis of \emph{Even}` for "An analysis of *even*").

[2]Note that there is no `.bib` extension given for the bibfile. Note also that, written as such, the file must be located either in the same directory as the file you're compiling, or in the default path searched by BibTEX. To be safe, you can simply include the full path name, e.g., `/home/john/references/myrefs`.

[3]`pdflatex` can be substituted by `latex`. For TEXshop users: typeset first using "LaTeX", then "BibTeX", then "LaTeX" twice, which can be found under the "Typeset" menu.

- The various cite commands can take *multiple* citekeys as arguments, separated by commas. So instead of `(see \cite{ref1}; \cite{ref2})`, you can write `(see \cite{ref1,ref2})`.
- To add a reference to the reference list that you do not actually cite in the document, use `\nocite{citekey}`. To add *all* references from the specified `.bib` file to the reference list, use `\nocite{*}`. This is useful for keeping a clean–looking copy of all your references, testing out different bibliography styles, double–checking that your entries are properly formatted, etc.
- The space between entries in the reference list can be modified with the `\bibsep` value, e.g., add `\setlength{\bibsep}{0pt}` to your preamble to have a single linebreak (no space) between entries.
- You can create your own macro for possessive citations (e.g., "In Chomsky's (1959) opinion") as follows:

  Example 5

  ```
  \newcommand{\citetposs}[1]{\citeauthor{#1}'s \citeyearpar{#1}}
  ...
  In \citetposs{chomsky1959} opinion ...
  ```

- Some excellent free and open–source frontends for BibTEX are JabRef (Windows, Linux, Mac OS X) and BibDesk (Mac OS X). They include features like classifying entries by keyword, creating sub–bibliographies (e.g., for small projects), import/export of various formats, automatic customized citekey generation, searching and extracting references from databases (Google Scholar, arXiv, CiteseerX), etc.