Composing LaTeX with Vim

David White

June 8, 2021

Intro to Vim

Motivating principles of (G)Vim:

- Enable the user to control the application and edit documents completely on the keyboard with minimal displacement of the hands.
- Incorporate the usual functionality of the mouse upon which beginners can fall back.

These aims are fulfilled by the *modal* nature of Vim: keypresses can perform actions other than inserting text depending on what mode we are in.
The LaTeX-suite plugin adds to Vim many new keymappings which insert TeX-specific structures.

Component Software

We will need to download and install/set up the following components:

1. **GVim:** MacOS ships with the terminal-based text editor Vim. GVim, branded as **MacVim** for MacOS, provides a GUI for this editor, providing e.g.
   - normal mouse functionality,
   - application dropdown menus,
   - context menus.

   Since we won't make any use of the terminal editor here, I will frequently refer to GVim simply as Vim.
2. **Pathogen:** A plugin manager which will allow us to "drop in" plugins like the LaTeX-suite with minimal configuration.
3. **Vim plugins:**
   - `vim-latex` – This is the plugin which will load all our LaTeX-related macros upon opening any `.tex` file.
   - `vim-sensible` – A collection of broadly desirable default settings for Vim. For our purposes this will operate "under the hood", and we won't have any cause for considering what it does.

Installing GVim

1. Navigate to `https://macvim-dev.github.io/macvim/`.
2. Click "Download" and save the `.dmg` file to your desired location.
3. Install from the `.dmg` as you would any standard MacOS app: find the `.dmg` file in Finder, double click it and follow the (minimal) instructions.

Using GVim from CLI

GVim can be run from Terminal via the command `mvim` when the latter is put in your `PATH` environment variable. To do this:

1. Open Terminal and run
   `echo $SHELL`
   - If the result is `/bin/zsh`, you are running Z Shell and will edit the file `~/.zprofile` in the next step.
   - If the result is `/bin/bash` you are running Bash (and are likely behind on OS updates). You will edit the file `~/.bash_profile` in the next step.
2. Run the command
   `echo 'PATH="/Applications/MacVim.app/Contents/bin/mvim:$PATH"'`
   before the occurence of `export PATH`. If there is no such occurence, put `export` at the beginning of the added line above.
3. Restart Terminal

Installing Pathogen

In `Terminal` run the commands

```
mkdir -p ~/.vim/autoload ~/.vim/bundle && \
curl -LSso ~/.vim/autoload/pathogen.vim
https://tpo.pe/pathogen.vim
```

Note: The backslash character denotes the unique instance of a newline/carriage return and should be retained if pasting all of the above on the command line.

Installing the plugins

1. In Terminal navigate to directory which will house Vim plugins by running

   ```
   cd ~/.vim/bundle
   ```

2. Run the following to copy the (executable) source code for the Vim plugins:

   ```
   git clone https://github.com/tpope/vim-sensible.git && \
   git clone https://github.com/vim-latex/vim-latex.git
   ```

   Again, the backslash must precede a newline/carriage return on the command line.

This has created two local source respositories on your machine:

- `~/.vim/bundle/vim-sensible`
- `~/.vim/bundle/vim-latex`

By running `git pull` within each of these directories, the repositories can be updated to match the latest version on github.

Configuring Vim

Every time Vim starts, all the commands in your `~/.vimrc` will be run. We will add commands to load Pathogen and establish some defaults to make our `.tex` files easy to read.
To test the installation and get our first practice, we will perform these edits with Vim:

1. In Terminal run `mvim ~/.vimrc`
2. Hit the `i` key in order to insert text.
3. Type in the following:
```
set wrap linebreak nolist
set number
set shiftwidth=4
execute pathogen#infect()
colorscheme slate
```
4. Hit <ESC> to reenter normal mode.
5. Type `:wq` and hit <RET> to save and quit.

Configuring Vim cont.

To make Vim your default editor for `.tex` files:

1. Navigate to any `.tex` file in Finder.
2. "Right-click" that `.tex` file and select "View Information".
3. Under "Open with:" select MacVim and click the button "Modify all".

Normal mode

Normal mode is where we begin. In this mode we can

- execute commands by typing `:<cmd>` and hitting `<Ret>`.
  E.g. `:wq` executes `[w]rite` (i.e. save) and then `[q]uit`.
- Navigate the document.
- Perform certain kinds of editing/file control.
- Enter the other modes.

From any other mode, we return to normal via `<ESC>`.

Normal mode

Common navigation commands:

- `<ln #>gg` – go to line <ln #>. Omitting the number goes to first line.
- `G` – go to last line
- `j` – down one line
- `k` – up one line
- `h` – left one character/column
- `l` – right one character/column
- `w` – forward one word
- `b` – back one word
- `e` – jump to end of word
- `0` – go to start of line
- `$` – go to end of line

Normal mode

Edit commands:

- `x` – delete character
- `dw` – delete word
- `d$` – delete to end of line
- `dd` – delete line
- `cw` – replace to end of word
- `cc` – replace line
- `c$` – replace to end of line
- `yy` – copy (yank) line
- `p` – paste
- `“+p` – paste from clipboard

Note: The last thing deleted in Vim is what gets pasted by `p`.

Normal mode

Control commands:

- `u` – undo
- `<CTRL> + R` – redo
- `/` – search/find
- `n` – cycle forward to next search result
- `N` – cycle backward to next search result
- `»` – indent line one shiftwidth
- `«` – unindent line one shiftwidth

Insert mode

Insert mode is for writing text.

Keys to enter insert mode:

- `i` – in front of the cursor
- `I` – start of current line
- `a` – after the cursor
- `A` – end of current line
- `o` – on a new line below
- `O` – on a new line above

Visual mode

Visual mode is where we select text and perform actions on those selections.

Enter visual mode: `v`

Things to do in visual mode:

- Use the navigation keys to highlight text.
- Use the editing keys to cut/manipulate selections.
- Execute commands limited to current selection.
- "+ followed by `y` or `x` resp. copies or cuts to clipboard.

Note: Selecting text can be done via the mouse in the usual manner, and doing so takes us into visual mode. Right clicking a selection provides a useful context menu.

Vim: helpful stuff

Reference: `https://vim.rtorr.com/`
Tips:

- One uses `<ESC>` frequently in Vim and `<CTRL> + J` frequently for moving between placeholders in LaTeX. It may therefore be advantageous to remap:
  - `<CAP>` to `<ESC>`;
  - `<CMD>` or `<OPT>` to `<CTRL>`.

  This can be done at `System Preferences > Keyboard > Modifier Keys`.
  Remap `<CAP>` to `<ESC>` (and maybe `<CMD>` to `<CTRL>`).
- We added the colorscheme "slate" to the `.vimrc` earlier. There are other colorschemes preinstalled with Vim and a vast quantity available online. Some provide better syntax highlighting for LaTeX than others, and personal taste is certainly a factor. You may wish to experiment.

Going further with Vim

- Separate files can be opened for editing in GVim either within tabs (`:tabn[ew]`) or in tiled buffers (`:b[uffer]`).
- File buffers can be compared in `diff` mode.

Latex-Suite configuration

Latex-Suite is can be configured to user preferences by creating a copy of the file

`~/.vim/bundle/vim-latex/ftplugin/latex-suite/texrc`

in a new directory

`~/.vim/ftplugin/tex`

and editing this copy.

Following are some "sensible" additions to that file so that `.tex` files are automatically saved when compiling/viewing:

```
map <leader>c :up<cr>:call Tex_RunLaTeX()<cr>
map <leader>v :up<cr>:call Tex_RunLaTeX()<cr>:call
Tex_ViewLaTeX()<cr>
```

Now \c saves and compiles, and \v saves, compiles and opens the pdf in the default viewer.

Latex-Suite macros

There are several types of macros for generating LaTeX:

1. Following the leader.
2. Three-character codes.
3. Function keys.

Leader macros

The "leader" in insert mode is ‘ (right of the 1 key).

Following this with most alphabetical characters produces Greek letters. E.g. ‘a produces α, and ‘q produces θ.

Numerals and special characters produce various symbols:

- ‘2 yields $\sqrt{\cdot}$
- ‘6 yields ∂
- ‘/ yields `\frac`

Placeholders

Many macros write one or more placeholders in the form `<+(opt name)+>` according to the LaTeX syntax.

To move to next place holder, press `<CTRL> J`.

We can get enviroments, headings and formatting commands using capitalized 3-character codes.

1. Enviroment codes begin with `E`
2. Organizational heading codes begin with `S` (for Section)
3. Text formatting codes begin with `F`

The next two letters are determined roughly as follows:

- For one-word, unprefixed names, the first two letters of the name.
  E.g. `EEQ` produces the equation environment and `FEM` produces `\emph`.
- For multi-word or prefixed names, the intial letters of the first two elements.
  E.g. `SSS` produces `\SubSection` and `EDM` produces the `displaymath` environment.
- For text formatting commands `\text<2 ltrs>`, those two terminal letters are used.
  E.g. `FBF` produces `\textbf`.

Function Keys

To be able to utilize this functionality, go to `System Preferences > Keyboard` and ensure that the box for using `F1`, `F2` etc. as standard function keys is checked.

- When the cursor is in contact with a recognized environment name, pressing `F5` creates that environment.
- Pressing `F7` creates a text or math formatting command out of whatever word the cursor is in contact with, whether a recognized command or not (useful for user newcommands).

When these function keys are pressed on an empty line, a numbered list of options is presented.

Note: This functionality requires the function keys to be set as such in the MacOS keyboard settings.

Doubled Keys

Double typing certain keys results in an appropriate expansion of the syntax. E.g.

- Typing `^^` produces `^{}`.
- Typing produces `\left( <++> \right)`.

Build process control

The best way to control/customize the build process is with a `makefile`.