Setting up a minimal neovim environment for data science code development

A neovim IDE for R, Python, and Julia

Ronald (Ryy) Glenn Thomas

12/1/22

Table of contents

Photo by Nathan Waters on Unsplash

Introduction

Neovim (a fork of Vim) is a text editor that has several advantages for data science code development. One of the main attractions is that it is open source and has a number of useful plugins to facilitate working on R, python, and julia code. Also, its modal, keyboard-centric system allows text and code

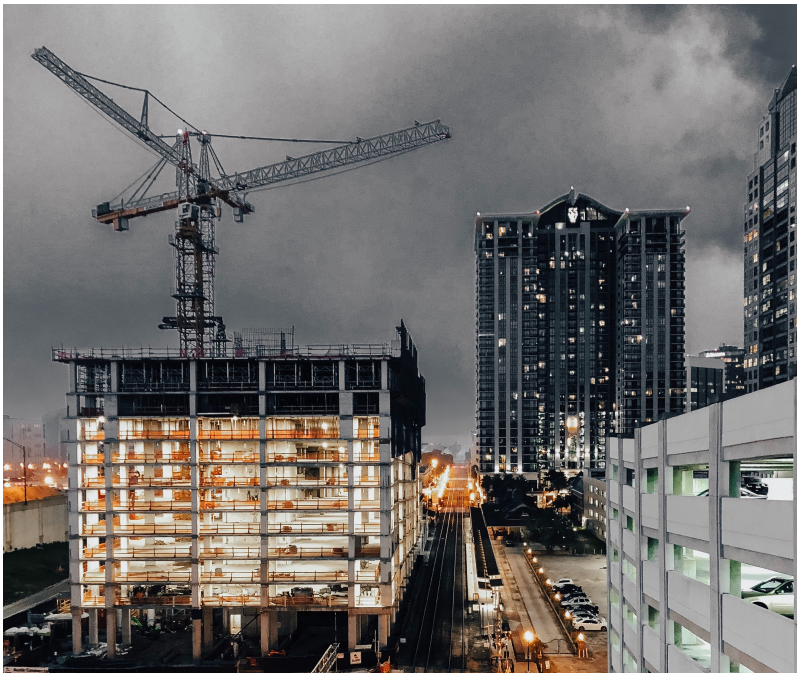

Figure 1: under construction

manipulation at potentially far greater speed than conventional, mouse-centric, systems.

In this post we describe both a minimal, yet functional setup, as well as a more extensive setup utilizing several of the newest neovim-only plugins, for neovim to allow IDE style code editing and REML interaction for the three primary data science coding tools: R, Python, and Julia.

Our presentation here is for a macos environment. Appendix one contains required adjustments for a ubuntu linux environment.

Step one: Install the latest stable version of neovim.

With minimal effort we can install both the terminal and GUI versions of neovim. The simplist approach is to use home-

brew:

```
> brew install neovim neovim-qt
```

Set up convenience aliases in `zsh`.

```
> alias ng = neovim-qt
> alias nt = neovim
```

(numonic: the `t` in `nt` is for terminal, the `g` in `ng` is for GUI)

Step 2: Configure neovim

The standard location for `neovim` configuration files on "unix-like" systems is `~/.config/nvim`. The main config file is either init.vim (VimL) or init.lua (Lua). In this post we'll focus on lua based configuration.

Specifically, the following code block creates an `nvim` subdirectory under `~/.config` and initialize a configuration file `init.lua`.

Here is the file hierarchy we'll construct. In fact all the code could be bundled into the `init.lua` file, but this approach is clearer and cleaner.

```
tree --charset=ascii
.
|-- init.lua
|-- leap-config.lua
|-- lua
|   |-- basics.lua
|   |-- nvim-R-config.lua
|   |-- nvim-cmp-config.lua
|   |-- nvim-telescope-config.lua
|   |-- nvim-tree-config.lua
|-- my_snippets
|   |-- giles.tex.snipppets
|   |-- mail.snippets
|   |-- rmd.snippets
```

```
|   |-- snippets.snippets
|   |-- tex.snippets
|   |-- text.snippets
|   `-- txt.snippets
|-- spell
|   |-- en.utf-8.add
|   `-- en.utf-8.add.spl
> cd .config
> mkdir nvim
> cd nvim
> touch init.lua
```

install the `paq` plugin manager

```
git clone --depth=1 https://github.com/savq/paq-nvim.git \
    ~/.local/share/nvim/site/pack/paqs/start/paq-nvim
:PaqInstall
```

Add the following code to `init.lua` list the plugins needed to be installed from `github` and "feed" them to `paq` for installation.

Nvim-R, Leap, UltiSnips, and vimtex need additional configuration. The required code is contained in bespoke files under the `lua` directory.

to use `paq-nvim` to manage plugins (maximally minimal)

```
require "paq" {
"savq/paq-nvim";
"junegunn/fzf";
'voldikss/vim-floaterm';
'preservim/nerdcommenter';
"NLKNguyen/papercolor-theme";
"SirVer/ultisnips";
"honza/vim-snippets";
"ggandor/leap.nvim";
"jalvesaq/Nvim-R";
'davidhalter/jedi-vim';
```

```
"lervag/vimtex";
"owickstrom/vim-colors-paramount"
}
require('basics')
require('nvim-R-config')
require('leap').add_default_mappings()
vim.keymap.del({'x', 'o'}, 'x')
vim.keymap.del({'x', 'o'}, 'X')
```

plugin discussions

keymaps

```
vim.cmd([[
set number relativenumber
set textwidth=80
set cursorline
set iskeyword-=_
set hlsearch
set splitright
set hidden
```

```
set incsearch
set noswapfile
set showmatch
set ignorecase
set smartcase
set gdefault
filetype plugin on

"completion
"        for text: S-Tab launches pop-up words
"   for R and Rmd completion is automatic
"ultisnips,
"   launch with C-j, move forward with C-j, move backward with C-k
"   open ultisnips file with <leader>u

set dictionary+=/usr/share/dict/words
let g:UltiSnipsSnippetDirectories=["UltiSnips", "my_snippets"]

let g:UltiSnipsExpandTrigger="<c-j>"
let g:UltiSnipsJumpForwardTrigger="<c-j>"
let g:UltiSnipsJumpBackwardTrigger="<c-k>"

let g:UltiSnipsEditSplit="vertical"
let g:UltiSnipsUsePythonVersion = 3
"nnoremap <leader>u :UltiSnipsEdit<cr>
set completeopt=longest,menuone

inoremap <expr> <TAB> pumvisible() ? "\<C-n>" : "\<TAB>"
inoremap <expr> <S-TAB> pumvisible() ? "\<C-p>" : "\<TAB>"
inoremap <expr> <CR> pumvisible() ? "\<C-y>" : "\<C-g>u\<CR>"
let R_set_omnifunc = ["r",  "rmd", "quarto",  "rhelp"]
let R_auto_omni = ["r", "rmd", "rhelp"]
]])
vim.g.mapleader = ","
vim.g.maplocalleader = " "
local map = vim.keymap.set
local opts = {noremap = true}
map('n', ':', ';', opts)
map('n', ';', ':', opts)
```

```
map('n', '<Space><leader>','<C-u>', opts)
map('n', '<leader>u',':UltiSnipsEdit<cr>', opts)
map('n', '<Space><Space>','<C-d>', opts)
map('n', '-','$', opts)
map('n', '<leader>f','vipgq', opts)
map('n', '<leader>v','edit ~/.config/nvim/init.lua<cr>', opts)
map('n', '<leader>a','ggVG', opts)
map('n', '<leader>t',':tab split<cr>', opts)
map('n', '<leader>y',':vert sb2<cr>', opts)
map('n', '<leader>0',':ls!<CR>:b<Space>', opts)
map('n', '<leader><leader>','<C-w>w', opts)
map('n', '<leader>1','<C-w>:b1<cr>', opts)
map('n', '<leader>2','<C-w>:b2<cr>', opts)
map('n', '<leader>3','<C-w>:b3<cr>', opts)
map('t',  'ZZ', "q('no')<CR>", opts)
map('t',  'ZQ', "q('no')<CR>", opts)
map('v',  '-', '$', opts)
map('t',  '<leader>0','<C-\\><C-n><C-w>:ls!<cr>:b<Space>', opts)
map('t',  '<Escape>','<C-\\><C-n>', opts)
map('t',  '<leader><leader>','<C-\\><C-n><C-w>w', opts)
map('i',  '<Esc>', '<Esc>`^', opts)
map('i',  '<S-Tab>', '<C-x><C-k>', opts)

vim.cmd([[
"    copy clipboard to register x for safe keeping
nnoremap <leader>x :let @x=@*
"    paste registers into terminal
tnoremap <expr> <C-R> '<C-\><C-N>"'.nr2char(getchar()).'pi'
set background=light
colorscheme paramount
]])
```

Set up R

```
vim.cmd([[
let $FZF_DEFAULT_COMMAND = 'rg --files --hidden'
let R_auto_start = 2
let R_hl_term = 0
let R_clear_line = 1
let R_pdfviewer = "zathura"
let R_assign = 2
let R_latexcmd = ['xelatex']
augroup rmarkdown
autocmd!
autocmd FileType rmd,r nnoremap <buffer> <CR>  :call SendLineToR("down")<CR>
autocmd FileType rmd,r nnoremap <buffer> <space>l :call SendChunkToR("silent","down")<cr>
autocmd FileType rmd,r nnoremap <buffer> <space>' :call RMakeRmd("default")<cr>
autocmd FileType rmd,r noremap <space>s :call RAction("str")<cr>
autocmd FileType rmd,r noremap <space>i :call RAction("dim")<cr>
autocmd FileType rmd,r noremap <space>h :call RAction("head")<cr>
autocmd FileType rmd,r noremap <space>p :call RAction("print")<cr>
autocmd FileType rmd,r noremap <space>q :call RAction("length")<cr>
autocmd FileType rmd,r noremap <space>n :call RAction("nvim.names")<cr>
autocmd FileType rmd,r noremap <space>c :call RAction("cnt")<cr>
autocmd FileType rmd,r noremap <space>k :call PreviousRChunk()<cr>
autocmd FileType rmd,r noremap <space>j :call NextRChunk()<cr>
augroup END
]])
```