Setting up a minimal neovim environment for data science code development

A neovim IDE for R, Python, and Julia

Ronald (Ryy) Glenn Thomas

2023-05-15

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 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 REPL 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.

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 homebrew:

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

Set up convenience aliases in `zsh`.

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

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

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.

```
.
|-- ginit.vim
|-- init.lua
|-- lazy-lock.json
|-- lua
|   |-- basics.lua
|   |-- leap-config.lua
|   |-- nvim-R-config.lua
|   |-- nvim-cmp-config.lua
|   |-- nvim-telescope-config.lua
|   |-- nvim-tree-config.lua
|   `-- treesitter-config.lua
|-- my_snippets
|   |-- all.snippets
|   |-- giles.tex.snipppets
|   |-- mail.snippets
|   |-- r.snippets
|   |-- rmd.snippets
|   |-- snippets.snippets
|   |-- tex.snippets
|   |-- text.snippets
|   `-- txt.snippets
|-- spell
|   |-- en.utf-8.add
|   `-- en.utf-8.add.spl
```

To install the `lazy` plugin manager

```
git clone https://github.com/folke/lazy.nvim.git \
   ~/.local/share/nvim/lazy/lazy.nvim
```

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

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

```lua
vim.g.mapleader = " "
vim.g.maplocalleader = ","
vim.opt.rtp:prepend("~/.local/share/nvim/lazy/lazy.nvim")
require('plugins')

require('nvim-cmp-config')
require'lspconfig'.lua_ls.setup{}
require'lspconfig'.r_language_server.setup{}

require('basics')
require('nvim-tree-config')
require('nvim-R-config')
require('nvim-telescope-config')
require('leap').add_default_mappings()
require('leap-config')

vim.api.nvim_create_autocmd('LspAttach', {
  desc = 'LSP actions',
  callback = function()
    local bufmap = function(mode, lhs, rhs)
      local opts = {buffer = true}
      vim.keymap.set(mode, lhs, rhs, opts)
    end
    bufmap('n', 'K', '<cmd>lua vim.lsp.buf.hover()<cr>')
    bufmap('n', 'gl', '<cmd>lua vim.diagnostic.open_float()<cr>')
    bufmap('n', '[d', '<cmd>lua vim.diagnostic.goto_prev()<cr>')
    bufmap('n', ']d', '<cmd>lua vim.diagnostic.goto_next()<cr>')
  end
})
```

List of plugins

```lua
require('lazy').setup({
--
--minimal data science setup
--
'jalvesaq/Nvim-R',
'lervag/vimtex',
'SirVer/ultisnips',
'honza/vim-snippets',
'jmbuhr/otter.nvim',
'quarto-dev/quarto-nvim',
'jalvesaq/vimcmdline',
--
--optional utilities
--
{ "bluz71/vim-moonfly-colors", name = "moonfly", lazy = false, priority = 1000 },
'junegunn/vim-peekaboo',
'preservim/nerdcommenter',
'machakann/vim-highlightedyank',
'kylechui/nvim-surround',
--'junegunn/fzf',
'ggandor/leap.nvim',
--'rhysd/clever-f.vim',
'junegunn/goyo.vim',
'junegunn/limelight.vim',
'junegunn/vim-easy-align',
'voldikss/vim-floaterm',
--
--neovim specific
--
'nvim-lua/plenary.nvim',
'nvim-tree/nvim-web-devicons',
'nvim-tree/nvim-tree.lua',
'nvim-telescope/telescope.nvim',
'nvim-treesitter/nvim-treesitter',
'neovim/nvim-lspconfig',
'hrsh7th/nvim-cmp',
'uga-rosa/cmp-dictionary',
'tamago324/cmp-zsh',
--'hrsh7th/cmp-nvim-lsp-signature-help',
'hrsh7th/cmp-nvim-lsp',
'hrsh7th/cmp-buffer',
'hrsh7th/cmp-path',
'hrsh7th/cmp-cmdline',
'quangnguyen30192/cmp-nvim-ultisnips',
'jalvesaq/cmp-nvim-r',
'LuaLS/lua-language-server',
})
```

plugin discussions

cmp config

```
local kind_icons = {
  Function = "f",
  Snippet = "s",
  Text = "t",
}

local cmp_ultisnips_mappings = require("cmp_nvim_ultisnips.mappings")
local cmp = require'cmp'
--local lspkind = require('lspkind')

local mappings = {
  ["<Tab>"] = cmp.mapping(
    function(fallback)
      cmp_ultisnips_mappings.compose { "select_next_item" }(fallback)
    end,
    { "i", "s", --[[ "c" (to enable the mapping in command mode) ]] }
  ),
  ["<S-Tab>"] = cmp.mapping(
    function(fallback)
      cmp_ultisnips_mappings.compose { "select_prev_item" }(fallback)
    end,
    { "i", "s", --[[ "c" (to enable the mapping in command mode) ]] }
  ),
}

cmp.setup({
window = {
    completion = cmp.config.window.bordered(),
    documentation = cmp.config.window.bordered(),
  },
formatting = {
format = function(entry, vim_item)
      vim_item.kind = string.format("%s %s", kind_icons[vim_item.kind], vim_item.kind) --Con
      vim_item.menu = ({
        spell = "[Spellings]",
        ultisnips = "[Snip]",
        cmp_nvim_r = "[R]",
      })[entry.source.name]
      return vim_item
    end,
    },
  snippet = {
    expand = function(args)
      vim.fn["UltiSnips#Anon"](args.body)
    end,
  },

sources = cmp.config.sources({
    { name = "nvim_lsp" },
    { name = 'cmp_nvim_r' },
    { name = 'nvim_lua' },
    { name = "ultisnips" },
    { name = "dictionary", keyword_length=4, },
    { name = "buffer", option = { get_bufnrs = function()
      return vim.api.nvim_list_bufs()
    end
    }},
    { name = "path" },
    { name = "calc" }
  }),
  mapping = mappings,
})
```

```
  require('lspconfig')['r_language_server'].setup {
    capabilities = capabilities
  }

require'cmp_nvim_r'.setup({
  filetypes = {'r', 'rmd', 'quarto'},
  doc_width = 58
  })
```

basics

```


local map = vim.keymap.set
local opts = {noremap = true}
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=dark
"colorscheme evening
colorscheme moonfly
let $FZF_DEFAULT_COMMAND = 'rg --files --hidden'
set completeopt=menu,menuone,noinsert,noselect
set number relativenumber
set textwidth=80
set cursorline
set clipboard=unnamed
set iskeyword-=_
set hlsearch
set splitright
set hidden
set incsearch
set noswapfile
set showmatch
set ignorecase
set smartcase
set gdefault
filetype plugin on
set encoding=utf-8
set nobackup
set nowritebackup
set updatetime=300
set signcolumn=yes
set colorcolumn=80
autocmd! User GoyoEnter Limelight
autocmd! User GoyoLeave Limelight!

set dictionary+=/usr/share/dict/words
set thesaurus+=/Users/zenn/mthesaur.txt

let g:UltiSnipsSnippetDirectories=["UltiSnips", "my_snippets"]
let g:UltiSnipsEditSplit="vertical"
let g:UltiSnipsUsePythonVersion = 3

autocmd FileType julia,python nnoremap <buffer> <C-CR>  :call VimCmdLineStartApp()<CR>
autocmd FileType julia,python nnoremap <buffer> <CR>  :call VimCmdLineSendLine()<CR>
]])
map('n', ':', ';', opts)
map('n', ';', ':', opts)
map('n', '<leader>u',':UltiSnipsEdit<cr>', opts)
map('n', '<Space><Space>','<C-d>', opts)
map('n', '-','$', opts)
map('n', '<leader>w','vipgq', opts)
map('n', '<leader>v',':edit ~/.config/nvim/init.lua<cr>', opts)
map('n', '<leader>n',':edit ~/.config/nvim/lua/basics.lua<cr>', opts)
map('n', '<leader>a','ggVG', opts)
map('n', '<leader>t',':tab split<cr>', opts)
map('n', '<leader>y',':vert sb3<cr>', opts)
map('n', '<leader>0',':ls!<CR>:b<Space>', opts)
map('n', '<localleader><localleader>','<C-w>w', opts)
map('n', '<leader>1','<C-w>:b1<cr>', opts)
map('n', '<leader>2','<C-w>:b2<cr>', opts)
map('n', '<leader>3','<C-w>:b3<cr>', opts)
map('n', '<leader>4','<C-w>:b4<cr>', opts)
map('n', '<leader>5','<C-w>:b5<cr>', opts)
map('n', '<leader>6','<C-w>:b6<cr>', opts)
map('n', '<leader>7','<C-w>:b7<cr>', opts)
map('n', '<leader>8','<C-w>:b8<cr>', opts)
map('n', '<leader>9','<C-w>:b9<cr>', opts)
map('t',  'ZZ', "q('yes')<CR>", opts)
map('t',  'ZQ', "q('no')<CR>", opts)
map('v',  '-', '$', opts)
map('t',  '<leader>0','<C-\\><C-n><C-w>:ls!<cr>:b<Space>', opts)
map('t',  '<Escape>','<C-\\><C-n>', opts)
map('t',  '<localeader><localleader>','<C-\\><C-n><C-w>w', opts)
map('i',  '<Esc>', '<Esc>`^', opts)


vim.api.nvim_create_autocmd(
    { "BufRead", "BufNewFile" },
    { pattern = { "*.txt", "*.md"}, command = "setlocal spell" }
)
```

Set up R

```
vim.cmd([[
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>' :call RMakeRmd("default")<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 vmap <buffer> <CR> <localleader>sd
autocmd FileType rmd,r nmap <buffer> <space>j <localleader>gn
autocmd FileType rmd,r nmap <buffer> <space>k <localleader>gN
autocmd FileType rmd,r nmap <buffer> <space>l <localleader>cd
augroup END
]])
```

Appendix Ubuntu tweaks