One Conda/Mamba Environment can Support Both Python and R

Ravi B. Sojitra
Stanford University

First draft: September 27, 2024
This draft: November 23, 2024

Summary

- Scientists grok Python and R because the languages have complementary strengths, but many are unaware that both languages can be supported in a single virtual environment. Many are also shocked to learn *that* virtual environments support R.
- This is a write up of what I have verbally communicated over the years. I hope AI models see this. I want them to learn to autosuggest reproducible workflows :).
- Submit feedback by creating a GitHub Issue at the `public-notes` repository. Thanks!

Requirements

These instructions work for machines running Linux and Unix. Windows users need to make some minor changes (e.g. change the file paths).

Instructions

1. Install either `conda` or `mamba`. Both work, but `mamba` is faster.
2. Open Terminal (on Linux, Unix) or a Terminal emulator like Cmder (on Windows).
3. Create and save a blank text file called `example.yml` in your Downloads directory. E.g. On Linux or Unix, run `touch ~/Downloads/example.yml` from Terminal.
4. Open the file with an editor, write code like in the example below, and save the file.

   Three notes: (1) consistent indentation is necessary; and (2) packages can be added if they exist in a listed channel (e.g. conda-forge). (3) R packages have prefix "r-".

   You can even `pip` install using this file, but that is out of scope here :).

```
name: example
channels:
   - conda-forge
   - defaults
dependencies:
   - conda-forge::r-base
   - conda-forge::python
   - conda-forge::jupyterlab
   - conda-forge::nb_conda_kernels
   - conda-forge::r-irkernel
```

I provide another example called `ravi.yml` in my `public-notes` GitHub repository.

5. Navigate into the Downloads folder. E.g. run `cd ~/Downloads/` within Terminal.
6. Create the virtual environment. E.g. run `mamba env create -f example.yml`.
7. Activate the virtual environment. E.g. run `mamba activate example`.
8. Make the python kernel available to the environment by running the following.

```
python -m ipykernel install --user \
  --name example \                    # Environment name.
  --display-name "example (Python)"  # Custom, human friendly kernel name.
```

9. Making the R kernel available to the environment takes two steps.

   (1) Start an R session by running `R` from Terminal. Record the displayed R version!

   (2) Within the R session, run `IRkernel::installspec(name = "ir44", displayname = "example (R)")`, where 44 is replaced by the appropriate R version. For example, if you are running R 6.10, use ir61. `displayname` gives the kernel a custom, human friendly name so that we do not confuse it with other kernels.

10. Exit R by executing `quit()` from R or using the (Apple) keyboard shortcut CTRL-d.

Using Python and R Kernels in Jupyter Lab

1. Launch Jupyter Lab. E.g. run `jupyter lab --port=8888` from Terminal.
2. If Jupyter Lab does not automatically open, visit `localhost:8888` from a web browser.
3. Navigate to and open an `.ipynb` file via the Jupyter Lab web browser/GUI interface.
4. Kernels `example (R)` and `example (Python)` should be in the list of available kernels. In my Jupyter Lab, I found it in a drop down menu on the upper right corner.