Introduction

This RC bootcamp is designed as an introduction to reservoir computing (RC). It provides various chapters to help beginners quickly learn the key concepts and techniques in the field.

Jupyter Notebook

This material uses Python as the programming language. In the long run, we aim to help you master interactive experimentation using Jupyter Notebooks. Below are representative setup and execution methods listed in order of difficulty. Please choose your preferred method for setup.

Method 1. Google Colaboratory + Google Drive (beginner)

Google Colaboratory is a browser-based Jupyter Notebook frontend suitable for beginners. No additional setup is required beyond a Google account, as core packages (e.g., `numpy`, `matplotlib`) are preinstalled. Unlike the read-only link on https://rc-bootcamp.github.io/, this method allows you to save notebooks directly on Google Drive for editing and preserving results. Note that free-tier resources are limited and sessions time out after periods of inactivity, so we recommend transitioning to a local environment once you become familiar with the basics.

0. Create or sign in to your Google account.
1. Download and extract the RC bootcamp materials (zip file).
2. Upload the entire folder containing this notebook to Google Drive.
3. Open the notebook file with the `.ipynb` extension.
4. Start the kernel and run cells with `Shift+Enter`.
5. In the cell that mounts Drive (`drive.mount("/content/gdrive")`), change `if False:` to `if True:`, and adjust the Google Drive path (e.g., `%cd /content/gdrive/My Drive/...`) if needed.
6. Run that cell and authenticate when prompted to grant access to your Google Drive.

Step 5 is required to run the notebook on Google Drive; do not skip it.

Method 2. uv + VSCode (competent)

This method uses the package management tool `uv` to create a virtual environment and run it in VSCode. It is recommended for those who want to conduct research and development more seriously. Its versatility and robust VSCode extension support make it accessible even for beginners (it's not that difficult once you get used to it).

Step 1. Installing uv

`uv` is a tool for managing Python packages. You can install packages on a per-project basis, and it can also be used for distribution by recording package versions (similar to conda, but uv is extremely fast). Follow the steps in the documentation to install it. Below are the minimum steps.

Linux and MacOS

Open your terminal and install it using the following command:

```
curl -LsSf https://astral.sh/uv/install.sh | sh
```

Then, check if it has been installed by executing:

```
uv help
```

Windows

Open PowerShell (open the Start Menu, type "Windows PowerShell", and launch the Windows PowerShell application) and install using the following command:

```
powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/install.ps1 | iex"
```

Then, check if it has been installed by executing:

```
uv help
```

Step 2. Configuring VSCode

Visual Studio Code (VSCode) is a cross-platform IDE developed by Microsoft. Download and run the installer. Once installed, you can use the `code` command to either drag the target folder into VSCode or open the directory from your terminal (use PowerShell on Windows):

```
code PATH_TO_DIR
```

One of VSCode's key features is its rich extension ecosystem. You can install extensions from the `Extensions` tab on the left. The minimum recommended packages are listed in `.vscode/extensions.json`. When you open the Extensions tab in VSCode, these packages will appear in the recommendations and can be installed with a single click.

Step 3. Creating the virtual environment

When you open the folder in VSCode, the folder contents will appear on the left side. Press `Ctrl+Shift+@` to open a terminal at the bottom of the screen. Enter the following command to set up the virtual environment:

```
uv sync
```

Based on the information in `uv.lock`, a `.venv` folder will be created in the same directory and the virtual environment will be set up. The Python kernel and packages used in this material will be automatically installed. This may take a few minutes. Once installation is complete, press `Ctrl+Shift+P` and select `Python: Select Interpreter` to specify the kernel (`.venv` in this case). For `.ipynb` files, select the target Python kernel from `Select Kernel` in the upper right corner.

If you are using Linux or macOS, you can enter the virtual environment by running the following command:

```
source .venv/bin/activate
```

For Windows, run the following command:

```
.venv\Scripts\activate
```

Step 4. Additional VSCode configuration (Optional)

`Ruff` is a linting tool for Python that automatically formats your code. By adding the following settings to `.vscode/settings.json`, the code will be automatically formatted whenever you save a file.

```
{
    "notebook.output.wordWrap": true,
    "[python]": {
        "editor.formatOnSave": true,
        "editor.codeActionsOnSave": {
            "source.fixAll": "explicit",
            "source.organizeImports": "explicit"
        },
        "editor.defaultFormatter": "charliermarsh.ruff"
    },
    "notebook.formatOnSave.enabled": true,
    "notebook.codeActionsOnSave": {
        "notebook.source.fixAll": "explicit",
        "notebook.source.organizeImports": "explicit",
    },
    "jupyter.notebookFileRoot": "${fileDirname}",
}
```