はじめに

この教材 RC bootcampはリザバー計算 (Reservoir computing; RC)の入門用に作成されています。 特にRCの分野で登場する概念や手法を、初心者でも素早く習得できるように様々な章が用意されています。

Jupyter Notebook

この教材ではプログラミング言語の一種であるPythonを使用します。 またより長期的にはJupyter Notebookを用いて、インタラクティブに実験を実装する手法の習得を目指します。 以下、設定・実行方法を難易度順に代表的なものを記載します。 お好きな方法でセットアップしてください。

方法1. Google Colaboratory + Google Drive (難易度: 初級)

Google Colaboratory (Colab) はJupyter Notebookのフロントエンドインターフェースです。 ブラウザ上で完結し、かつ主要なパッケージ (`numpy` 、 `matplotlib` 等) はインストール済みなので特別な設定は特に必要なく、 **プログラミング未経験者におすすめできる** 方法です。 また https://rc-bootcamp.github.io/ のColabリンクとは異なり、こちらの方法では **Google Drive上にノートブックを配置する** ので、後から編集や実行結果の保存が可能です。 一方で計算リソースが無料版だと限定的な点、時間が経過するとインスタンスがシャットダウンしてしまう点から、ある程度慣れたら **以降に紹介するローカルで実行する方法の導入を推奨します**。

0. Googleのアカウントが必要ですので用意してください。
1. RC bootcampの教材 (zipファイル) をダウンロードし展開してください。
2. Google driveにこのノートブックが含まれたフォルダ全体をアップロードしてください。
3. `.ipynb` と書かれているファイルがnotebookですので開いてください。
4. カーネルを起動後 `Shift+Enter` でセル(四角で区切られたコード) を実行できます。
5. `drive.mount("/content/gdrive")` と書かれているセルについて、 `if False:` の部分を `if True:` に書き換えてください。また必要に応じてGoogle Driveのパスも変更してください (`%cd /content/gdrive/My Drive/...` の箇所)。
6. 上記のセルを実行し認証してください。認証のポップアップが表示されるので許可してください。これでGoogle Driveのディレクトリに接続してそのリソースを利用できるようになります。

このうち、5.の変更は、Google Drive上で動かすために重要ですので忘れずに行ってください。

方法2. uv + VSCode (難易度: 中級)

パッケージ管理ツール `uv` を用いて仮想環境を構築しVSCode上で実行する方法で、本格的に研究・開発を行う人におすすめする方法です。 汎用性が高くVSCodeの強力なパッケージの支援を受けられるので、初学者でも挑戦しても問題ないと思います (実際慣れればそこまで難しくないです)。

Step 1. uvのインストール

`uv` はpythonのパッケージの管理ツールです。 プロジェクト毎にパッケージをインストールできるだけでなく、そのパッケージのバージョンを記録するため配布にも活用できます (condaと似ていますがuvは極めて高速に動作します)。 詳細はドキュメントの手順にしたがってインストールしてください。 以下最低限の手順を説明します。

LinuxやMacOSの場合

コンソールを開き、以下のコマンドでインストールできます。

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

インストールが完了したら

```
uv help
```

でインストールされているか確認してください。

Windowsの場合

PowerShellを起動し (スタートメニューを開き、「Windows PowerShell」と入力し、Windows PowerShell が表示されるので起動できます。) 以下のコマンドでインストールできます。

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

インストールが完了したら、

```
uv help
```

でインストールされているか確認してください。

Step 2. VSCodeの設定

Visual Studio Code (VSCode)はMicrosoftが開発するクロスプラットフォームのIDEです。 インストーラをダウンロード＆実行してください。 すると `code` コマンドが使用できるようになるので、対象のフォルダをドラッグするか、以下のコマンドで対象のディレクトリを開いてください (Windowsの場合はPowerShellを使ってください)。

```
code PATH_TO_DIR
```

VSCodeの特長はその拡張機能の豊富さです。 拡張機能は左側の `Extensions` タブよりインストールできます。 最低限のおすすめのパッケージは `.vscode/extensions.json` に

記載しておきました。 VSCodeの拡張機能のタブを開くとそのパッケージが表示されるので、クリックするだけでインストールできます。

Step 3. 仮想環境の構築

フォルダをVSCodeで開くと左側にフォルダの中身の一覧が表示されます。
`Ctrl+Shift+@` を押すと下画面にターミナルが開くと思います。 そこで次のコマンドを入力して仮想環境を構築してください。

```
uv sync
```

`uv.lock` に書かれている情報をもとに、同じディレクトリの `.venv` と書かれたフォルダが作成され仮想環境が構築されます。 同時にこの教材で使われるPythonカーネルとPythonのパッケージが自動的にインストールされれます。 数分かかると思うので、インストールし終わったら起動し `Ctrl+Shft+p` を押し、 `Python: Select Interpreter` から使うカーネル (今回は `.venv`)を指定します。 `.ipynb` のカーネルも同様で、右上に表示されている `Select Kernel` から対象のPython Kernelを指定してください。

なおVSCode以外でもLinuxかMacOSの場合はから以下のコマンドを実行すると仮想環境に入れます。

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

Windowsの場合は以下のコマンドを実行してください。

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

Step 4. その他VSCodeの設定 (オプション)

オススメの拡張機能にある `Ruff` はPythonのLintingツールで、自動的にコードを整形してくれます。 VSCodeでは `.vscode/settings.json` に以下の設定を記載しておくと、ファイルを保存するたびに自動的にコードが整形されます。

```
{
    "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}",
}
```