Running VIC on GitHub Codespaces

Yifan Cheng, October 22, 2025

1 Table of Contents

2 Downloading the source code

First of all, let's go to your GitHub Codespaces. *Please remember to delete your existing Codespaces and open a new one.* Some configurations have been modified.

2.1 Download from GitHub

Go to the workspace directory

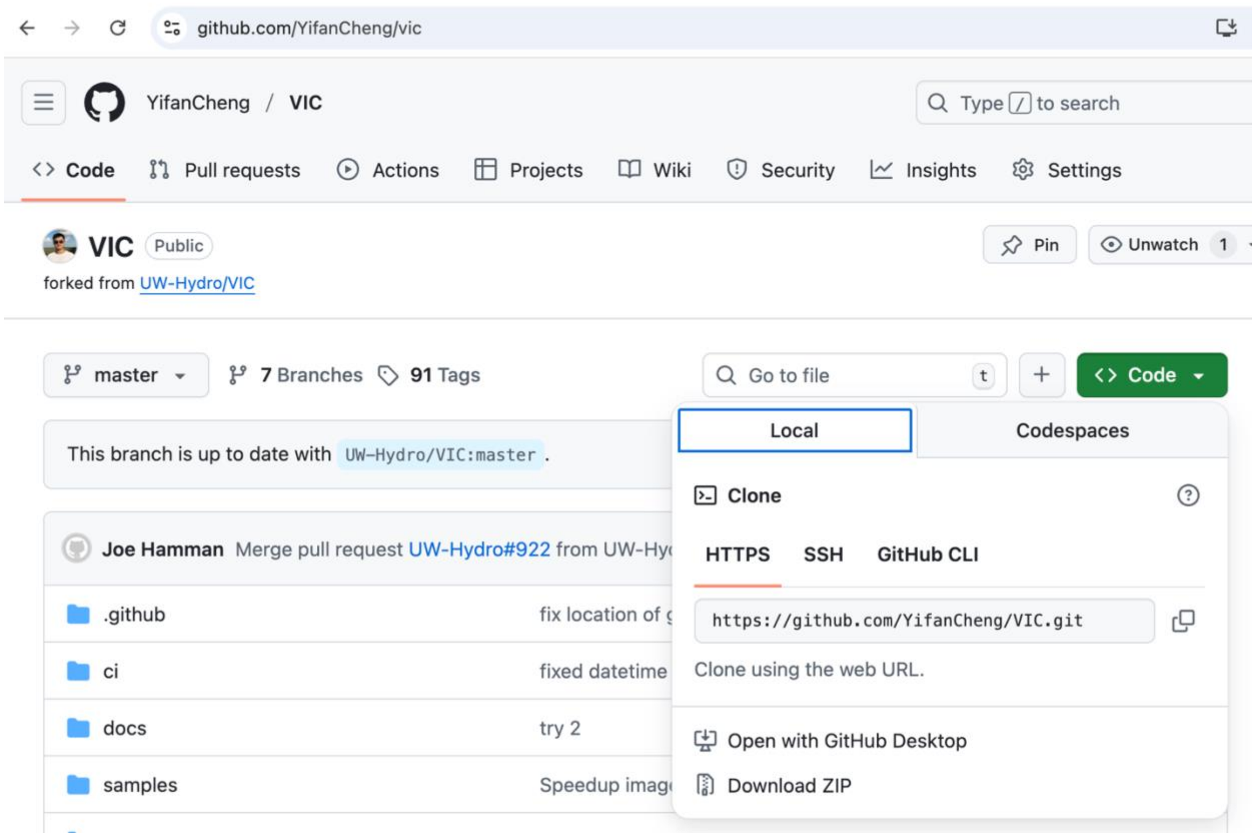
```
cd /workspaces
```

Let's download the source code of VIC using the following command

```
git clone https://github.com/YifanCheng/VIC.git
```

Notes:

1. *"git clone" can be used to download the source code from GitHub Repo. Usually, you can get the downloadable link by clicking the Green Button "Code" and then click "Local" (Figure 1 below).*
2. *After you successfully download the source code, you can see a new folder named "VIC" under the "/workspaces" directory.*



This branch is up to date with `UW-Hydro/VIC:master`.

Joe Hamman Merge pull request UW-Hydro#922 from UW-Hy

.github	fix location of g
ci	fixed datetime
docs	try 2
samples	Speedup imag

```
https://github.com/YifanCheng/VIC.git
```

Clone using the web URL.

Figure 1: Screenshot of GitHub

2.2 Check out the correct branch

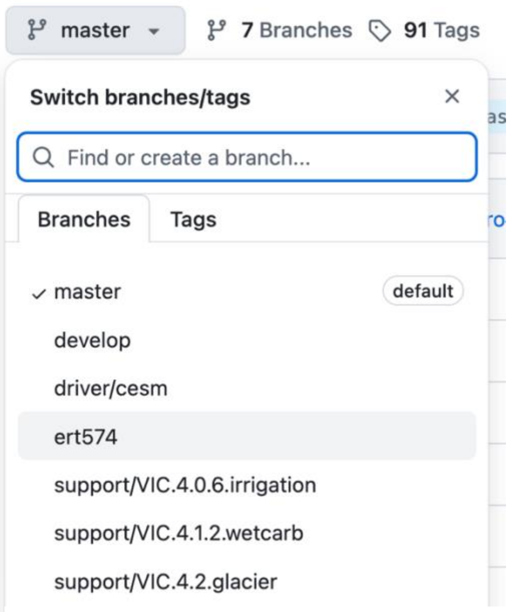


Figure 2: Screenshot of GitHub (Branch selection

The ***master*** branch usually contains the most updated and stable source code. However, it usually requires modifying the **makefile** to successful compiling the source code in different systems. In order to simplify the compiling process, I have edited the make file and tested it in the Codespaces. The updated code package was saved in a different branch, called “ert574”. You can actually see this branch in GitHub page (as shown in Figure 2).
First, we need to go to the model directory

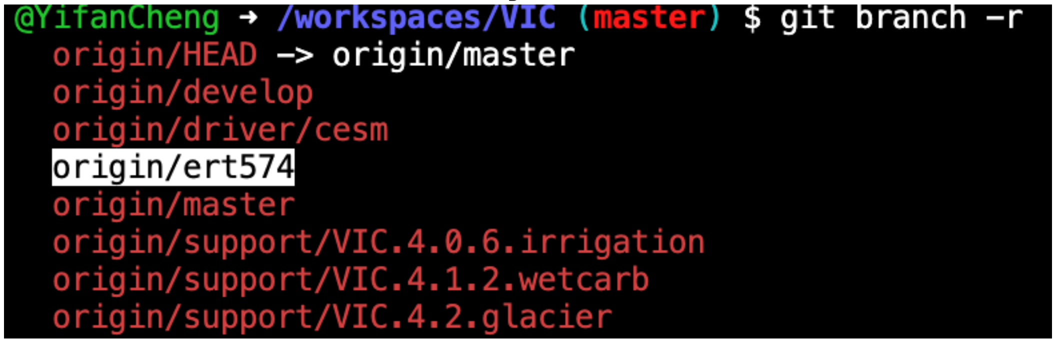
```
cd VIC/
```

Second, type

```
git branch
```

You can see the available branches in the local machine. However, we can only see the “master” branch, not the ert574 branch! *Why is that?* That is because in the “git clone” step, by default, we only download the code from the master (or main) branch. Now we need to fetch data from different branch. We can see what remote branches are available by typing

```
git branch -r
```

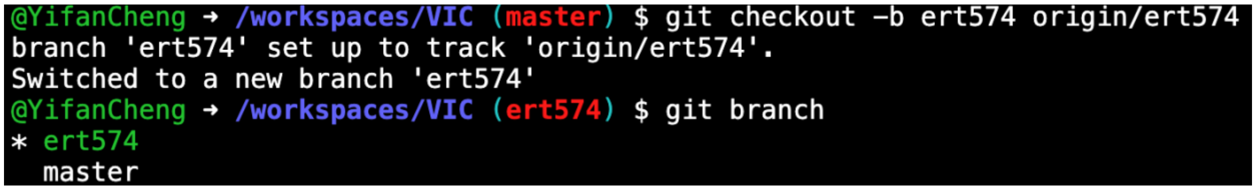


```
@YifanCheng ➜ /workspaces/VIC (master) $ git branch -r
  origin/HEAD -> origin/master
  origin/develop
  origin/driver/cesm
  origin/ert574
  origin/master
  origin/support/VIC.4.0.6.irrigation
  origin/support/VIC.4.1.2.wetcarb
  origin/support/VIC.4.2.glacier
```

Figure 3: Check remote branch

Now we would like to switch to the branch ert574,

```
git checkout -b ert574 origin/ert574
```



```
@YifanCheng ➜ /workspaces/VIC (master) $ git checkout -b ert574 origin/ert574
branch 'ert574' set up to track 'origin/ert574'.
Switched to a new branch 'ert574'
@YifanCheng ➜ /workspaces/VIC (ert574) $ git branch
* ert574
  master
```

Figure 4: Creating a new local branch based on remote branch

Note

1. *git checkout is used to switch between different branches. "-b" is to create a new branch. The first "ert574" means that we are creating a local branch named "ert574". "origin/ert574" means that the new local branch is based on the remote branch on GitHub.*

3 Compiling the source code

To run a hydrologic model, we will need to compile the code to get an executable file. The VIC model is mostly written in Fortran, so we would need to use a Fortran compiler. Please navigate to this following folder "`/workspaces/VIC/vic/drivers/image`" (from now on, I will only list the target folder. You would need to remember the command)
In this folder, you can see a **makefile**. What does **makefile** look like? Take a look at it.
To compile the code, type

```
make
```

Where is the executable file?

```
/workspaces/VIC/vic/drivers/image/vic_image.exe
```

4 Running example cases

Now we have successfully compiled the code! Well done! Let's run one example model case. Let's first download the example dataset.

4.1 Download example datasets

The model developer usually not only provide the source code, but also the example datasets so we can perform simple test to make sure that the code is compiled correctly, and the model output makes sense. The example dataset is stored in a GitHub repo. Please go to the folder `/workspaces` and download the following repo (*What's the command to download code from a GitHub repo?*)

```
https://github.com/UW-Hydro/VIC_sample_data.git
```

4.2 Prepare configuration files

Previously, we compiled the source code and get an executable file, which should be generalizable for all different applications. Therefore, we will need to pass along the important information, such as meteorological forcing data (such as air temperature, precipitation), land cover types (trees, grass, or bare ground), to the executable file. Configuration file is a text file that summarize this information, particularly the path and filenames to the meteorological forcing data and parameters to the executable file.
First, let's take a look at an example configuration file, located in the following folder

```
/workspaces/VIC_sample_data/image/Stehekin/parameters/
```

Let's use this following configuration file

```
global_param.Stehekin.L2015.txt
```

To modify the configuration file, let's locate this file in the side bar on the left and double click this file to activate the text editor in VS Studio.
The definition of each item in the global configuration file is available through the VIC documentation website (https://vic.readthedocs.io/en/master/Documentation/Drivers/Classic/GlobalParam/#vic-run-time-options-global-parameter-file). Take a look!

Quiz question:

If we would like to change the start date to 1949-01-03, what should we change?

In this configuration file, we will need to modify the several important information.
First, we need to replace "`${VIC_SAMPLE_DATA}`" using the correct path to the file,

Replace `${VIC_SAMPLE_DATA}` using `/workspaces/VIC_sample_data`

Note:

1. *We would need to replace it at multiple locations throughout the configuration file, including `DOMAIN`, `FORCING1`, `PARAMETERS`, and `RESULT_DIR`.*
2. *`DOMAIN`, `FORCING1`, `PARAMETERS` points to the corresponding input datasets, while `RESULT_DIR` points to the output directory.*
   a. *Does all input data exist? Please check.*
   b. *Does this output directory exist? If not, we would need to create this folder!*

```
mkdir /workspaces/VIC_sample_data/sample_image
```

4.3 Run the model

Now, it is finally time to run the model!
First, let's make sure that we are in the folder where configuration file is located. Let's type

```
/workspaces/VIC/vic/drivers/image/vic_image.exe -g
global_param.Stehekin.L2015.txt
```

If you did not catch that there is a typo in the "PARAMETERS" path, you would get the following error message:

```
[ERROR] ../shared_image/src/vic_start.c:59: errno: No such file or directory: Error opening /workspaces/VIC_sample_data/image/Stehekin/parameters/parameters.Stehekin.L2015.nc
```

How should we fix this bug? We need to replace the parameter file using

```
/workspaces/VIC_sample_data/image/Stehekin/parameters/params.Stehekin.L2015.nc
```

Note: The parameter file is "`params.Stehekin.L2015.nc`", not "`parameters.Stehekin.L2015.nc`".

4.4 Check output

Yayyyyy!! We should have successfully run the model. Let's check out the output using Python.
Let's create an empty notebook in the output folder.

```
/workspaces/VIC_sample_data/sample_image/week9_lab_checkout.ipynb
```

Below are some example code

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import xarray as xr
```

How can we quickly check the spatial map of this basin?
# check precipitation

```
ds.OUT_PREC.mean(dim='time').plot()
```

# check surface runoff

```
ds.OUT_RUNOFF.mean(dim='time').plot()
```

# check baseflow

```
ds.OUT_BASEFLOW.mean(dim='time').plot()
```

How can we take a look at the time series?

# check precipitation

```
ds.OUT_PREC.mean(dim=['lat', 'lon']).plot()
```

# surface runoff

```
ds.OUT_RUNOFF.mean(dim=['lat', 'lon']).plot()
```

# baseflow

```
ds.OUT_BASEFLOW.mean(dim=['lat', 'lon']).plot()
```

# evaporation

```
ds.OUT_EVAP.mean(dim=['lat', 'lon']).plot()
```

# soil moisture

```
ds.OUT_SOIL_MOIST.sel(nlayer=0).mean(dim=['lat', 'lon']).plot()
```

# Snow Water Equivalent (SWE)

```
ds.OUT_SWE.mean(dim=['lat', 'lon']).plot()
```

Quiz question:

Based on your exploration, where does the majority of precipitation go in this simulation? Runoff, evaporation, or snow?