Portable Python-wrapped FV3GFS atmospheric model

Vulcan Climate Modeling, Seattle, WA

Geophysical Fluid Dynamics Laboratory, Princeton, NJ

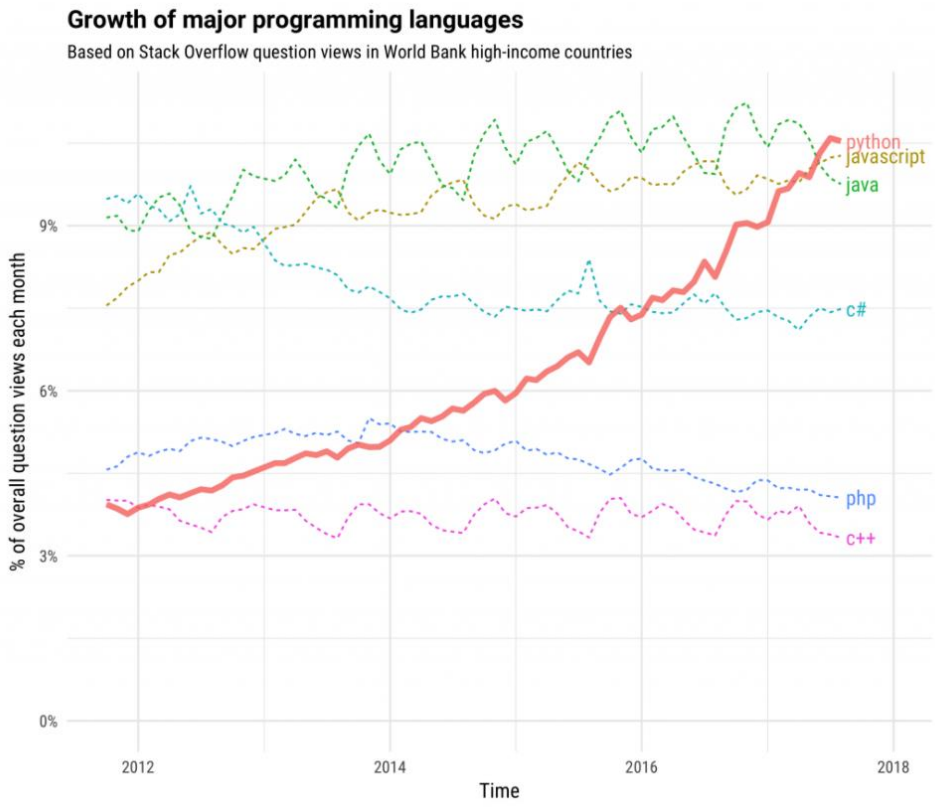
Jeremy McGibbon, Noah Brenowitz, Mark Cheeseman, Spencer Clark, Johann Dahm, Eddie Davis, Oliver Elbert, Rhea George, Lucas Harris, Brian M Henn, Anna Kwa, Andre Perkins, Oliver Watt-Meyer, Tobias Wicky, Christopher S. Bretherton, and Oliver Fuhrer

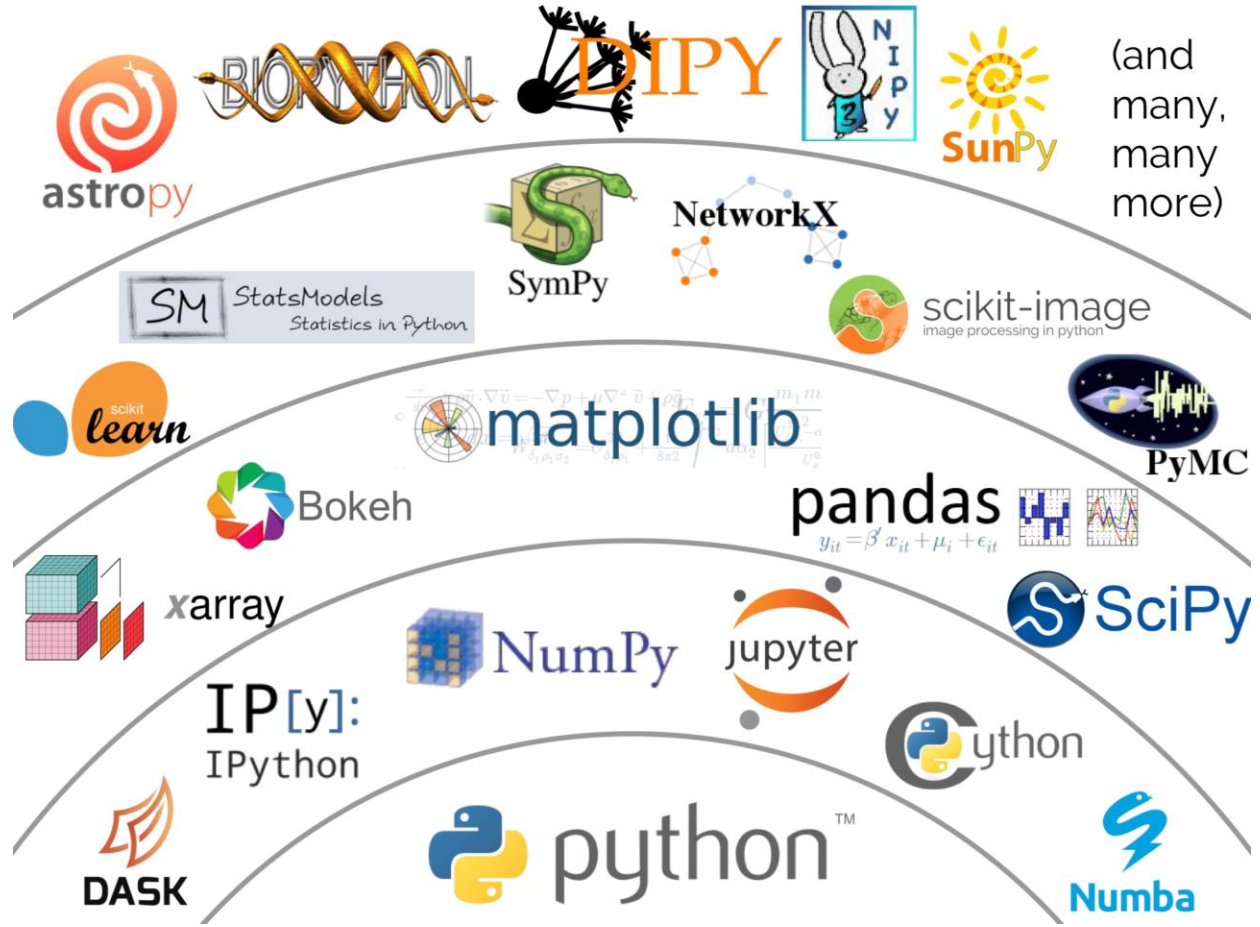
Why use Python in your model?

Python is popular. Why does that matter?

Programming language Python's popularity: Ahead of Java for first time but still trailing C

Python is now the second most popular language in the nearly 20-year-old Tiobe index.





Flask

TensorFlow

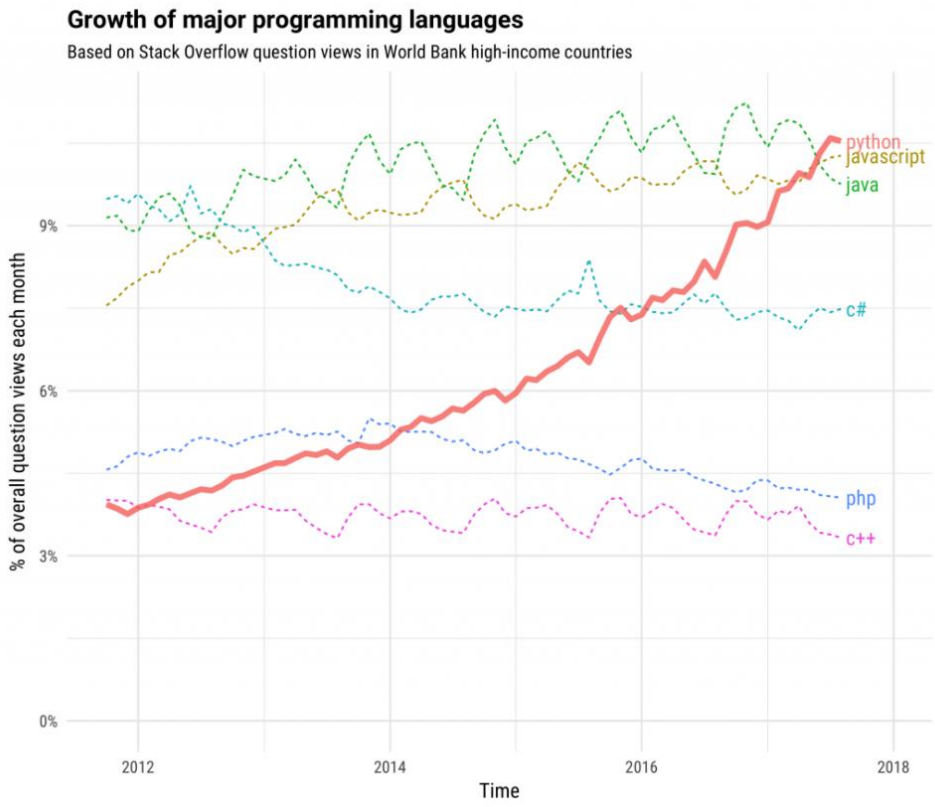
pytest

Credit: Jake VanderPlas, "The Unexpected Effectiveness of Python in Science", PyCon 2017

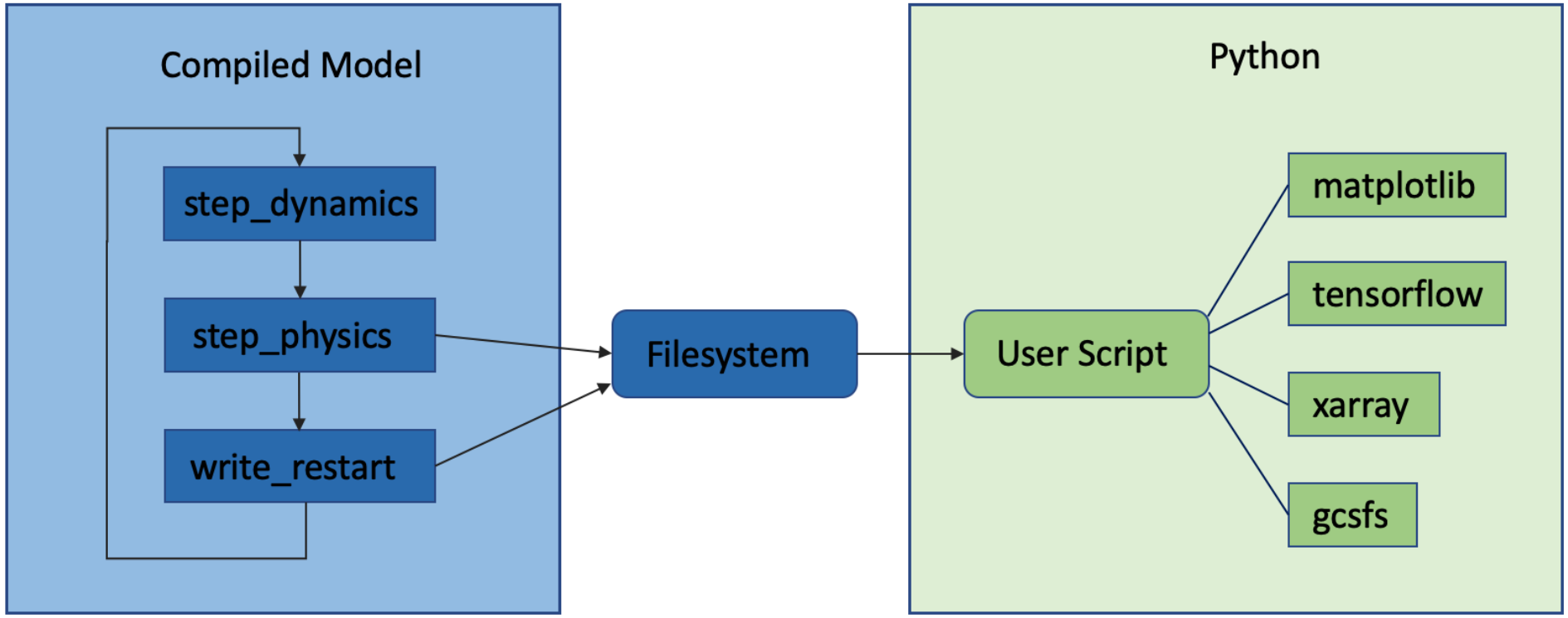
- You can use code written by others
  - Matplotlib, numpy, tensorflow, etc.
- Code can be understood and modified by more users
- More people can answer your questions
- Python calls Fortran, which is also popular

Programming language Python's popularity: Ahead of Java for first time but still trailing C

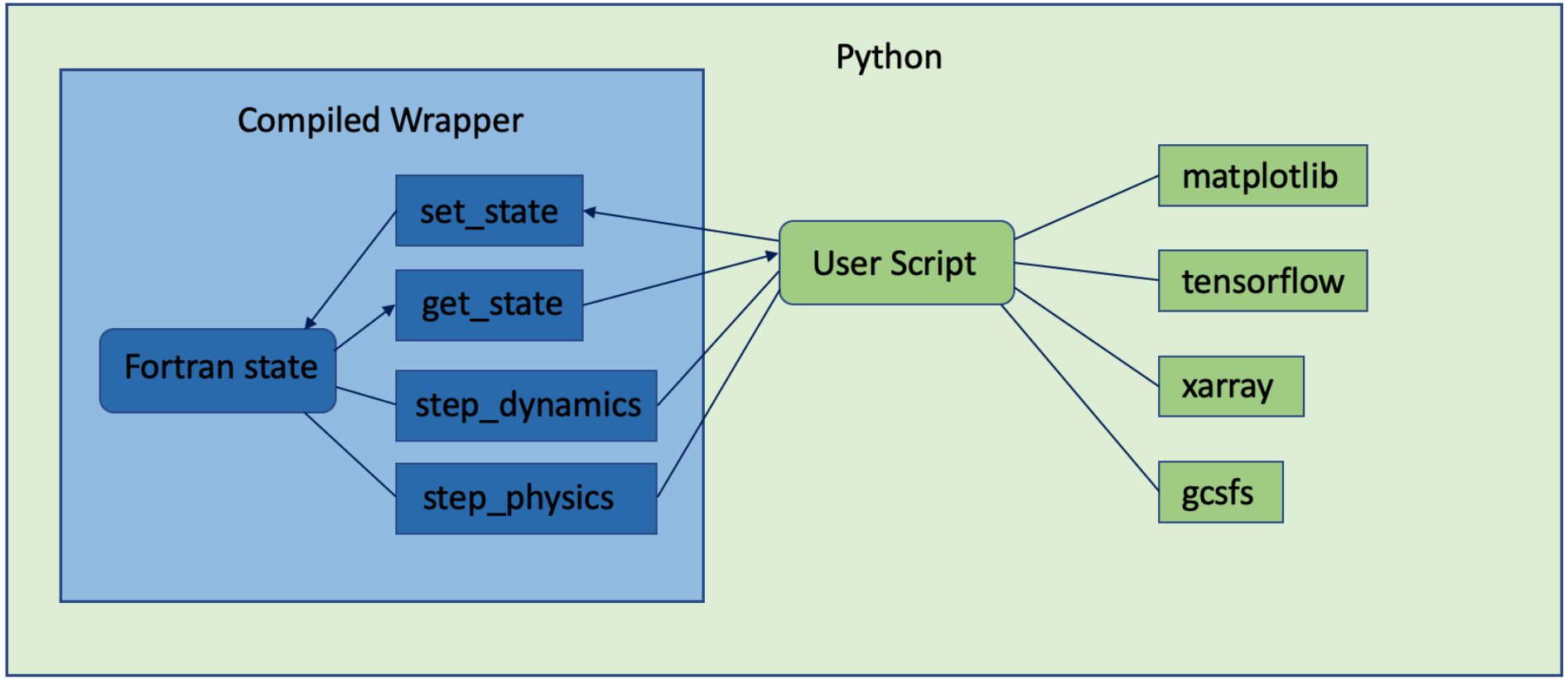
Python is now the second most popular language in the nearly 20-year-old Tiobe index.



Traditional Fortran-Python workflow



Wrapper Fortran-Python workflow



Concerns using Python

- Long term support
- Backwards-incompatible changes to the language

Why Python 4.0 won't be like Python 3.0

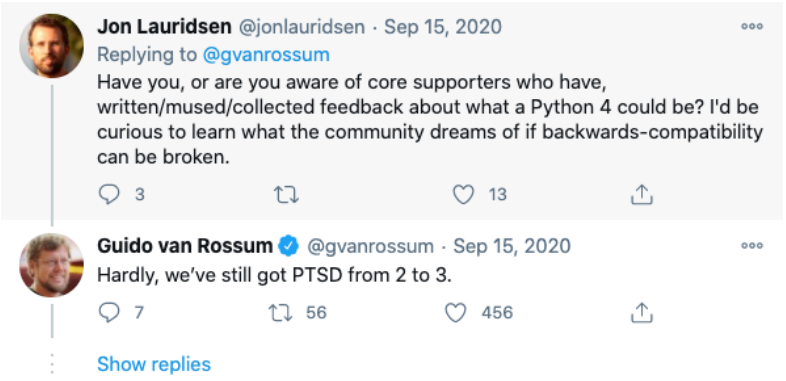
Nick Coghlan — 2014-08-17 05:30 — 57 Comments — Source

Newcomers to python-ideas occasionally make reference to the idea of "Python 4000" when proposing backwards incompatible changes that don't offer a clear migration path from currently legal Python 3 code. After all, we allowed that kind of change for Python 3.0, so why wouldn't we allow it for Python 4.0?

I've heard that question enough times now (including the more concerned phrasing "You made a big backwards compatibility break once, how do I know you won't do it again?"), that I figured I'd record my answer here, so I'd be able to refer people back to it in the future.

What are the current expectations for Python 4.0?

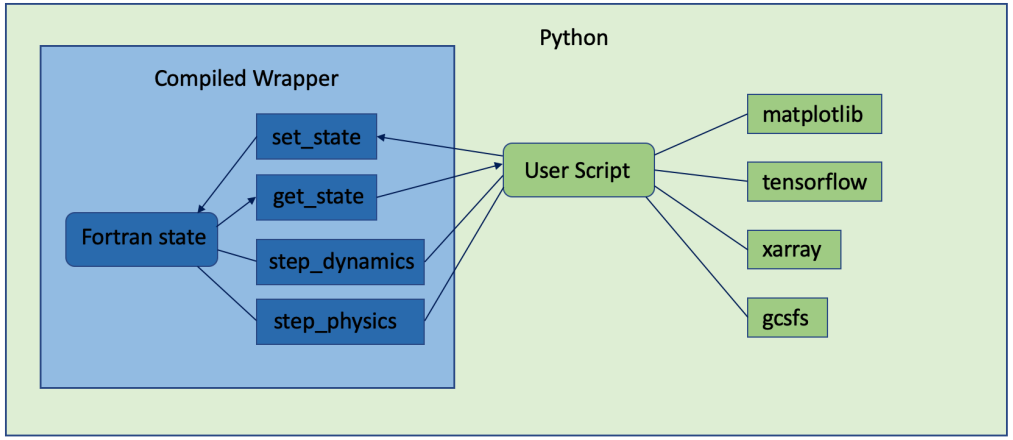
My current expectation is that Python 4.0 will merely be "the release that comes after Python 3.9". That's it. No profound changes to the language, no major backwards compatibility breaks - going from Python 3.9 to 4.0 should be as uneventful as going from Python 3.3 to 3.4 (or from 2.6 to 2.7). I even expect the stable Application Binary Interface (as first defined in PEP 384) to be preserved across the boundary.



What is fv3gfs-wrapper?

- Cython wrapper to encapsulate Fortran model in an importable compiled library
- Fortran has its own state you can copy/overwrite
- Routines to step the Fortran model forward

```
import fv3gfs.wrapper

if __name__ == "__main__":
    fv3gfs.wrapper.initialize()
    for i in range(fv3gfs.wrapper.get_step_count()):
        fv3gfs.wrapper.step_dynamics()
        fv3gfs.wrapper.step_physics()
        fv3gfs.wrapper.save_intermediate_restart_if_enabled()
    fv3fs.wrapper.cleanup()
```



Bit-for-bit Validation

- Identical output to Fortran model
- Makes sense – exactly the same Fortran code is called

```
import fv3gfs.wrapper

if __name__ == "__main__":
    fv3gfs.wrapper.initialize()
    for i in range(fv3gfs.wrapper.get_step_count()):
        fv3gfs.wrapper.step_dynamics()
        fv3gfs.wrapper.step_physics()
        fv3gfs.wrapper.save_intermediate_restart_if_enabled()
    fv3fs.wrapper.cleanup()
```

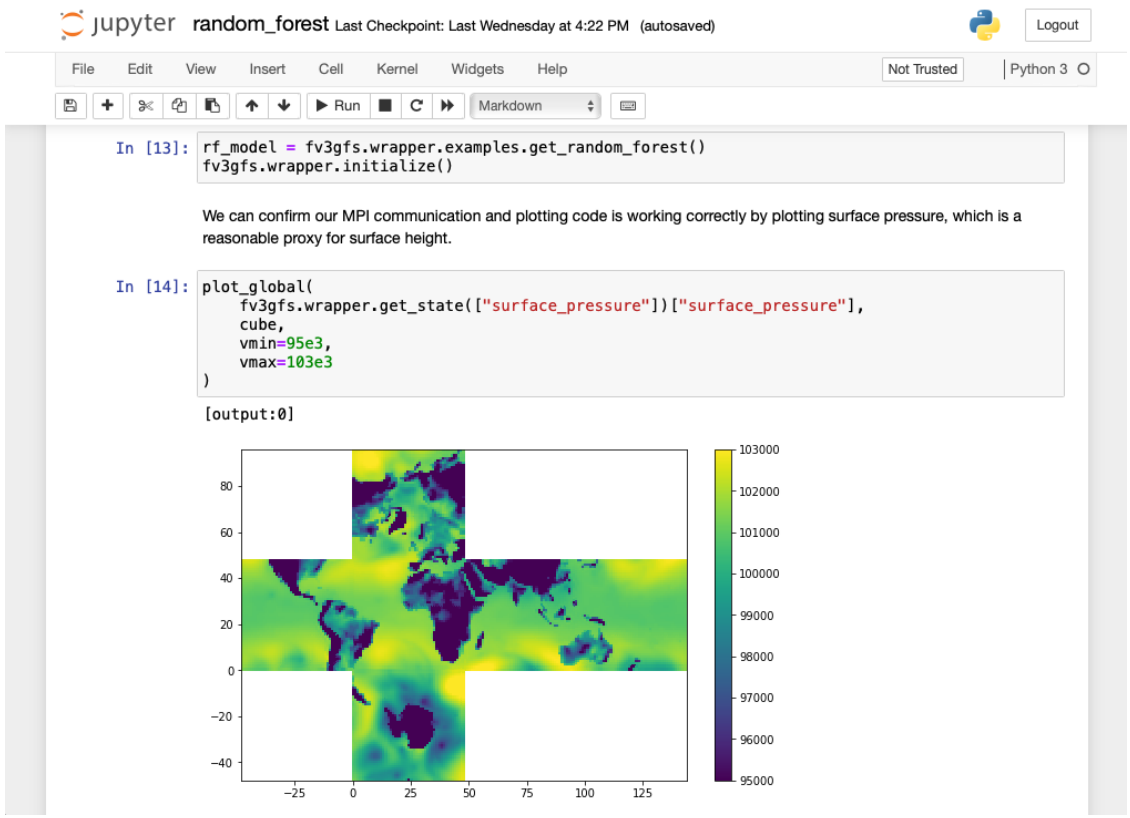
Online Machine Learning

- Base model can be easily modified
- get_state and set_state allow copying in and out from Fortran model state
- Allows access to external Python libraries (f90nml, scikit-learn) from within model

```python
import fv3gfs.wrapper
import fv3gfs.wrapper.examples
import f90nml
from datetime import timedelta

if __name__ == "__main__":
    # load timestep from the namelist
    namelist = f90nml.read("input.nml")
    timestep = timedelta(seconds=namelist["coupler_nml"]["dt_atmos"])
    # initialize the machine learning model
    rf_model = fv3gfs.wrapper.examples.get_random_forest()

    fv3gfs.wrapper.initialize()
    for i in range(fv3gfs.wrapper.get_step_count()):
        fv3gfs.wrapper.step_dynamics()
        fv3gfs.wrapper.step_physics()

        # apply an update from the machine learning model
        state = fv3gfs.wrapper.get_state(rf_model.inputs)
        rf_model.update(state, timestep=timestep)
        fv3gfs.wrapper.set_state(state)

        fv3gfs.wrapper.save_intermediate_restart_if_enabled()
    fv3gfs.wrapper.cleanup()
```

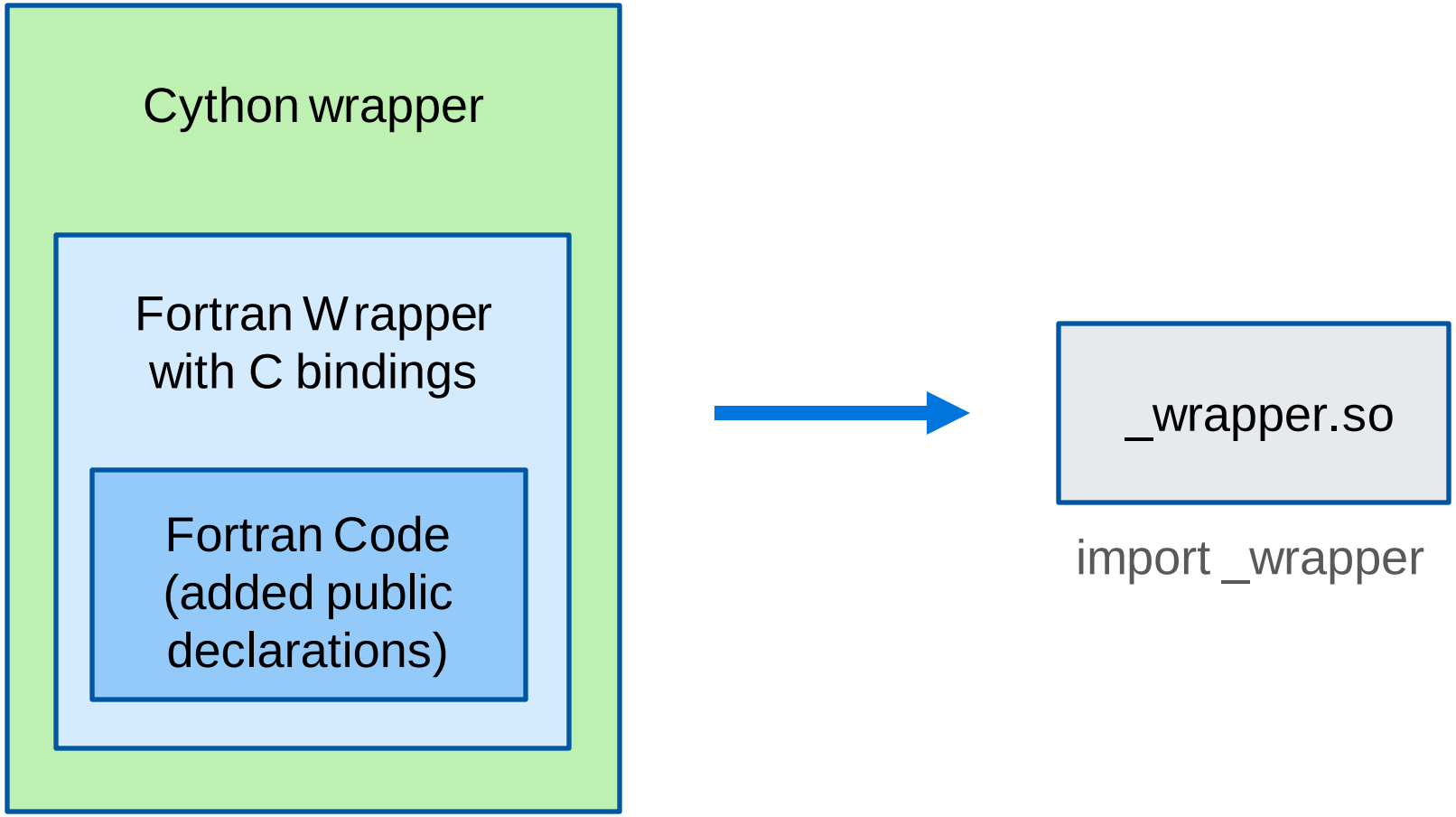
Interactive Development

- Can run in parallel Jupyter notebooks
- ipyparallel can run on many nodes on HPC
- mpi4py allows communication
- Useful for teaching/learning

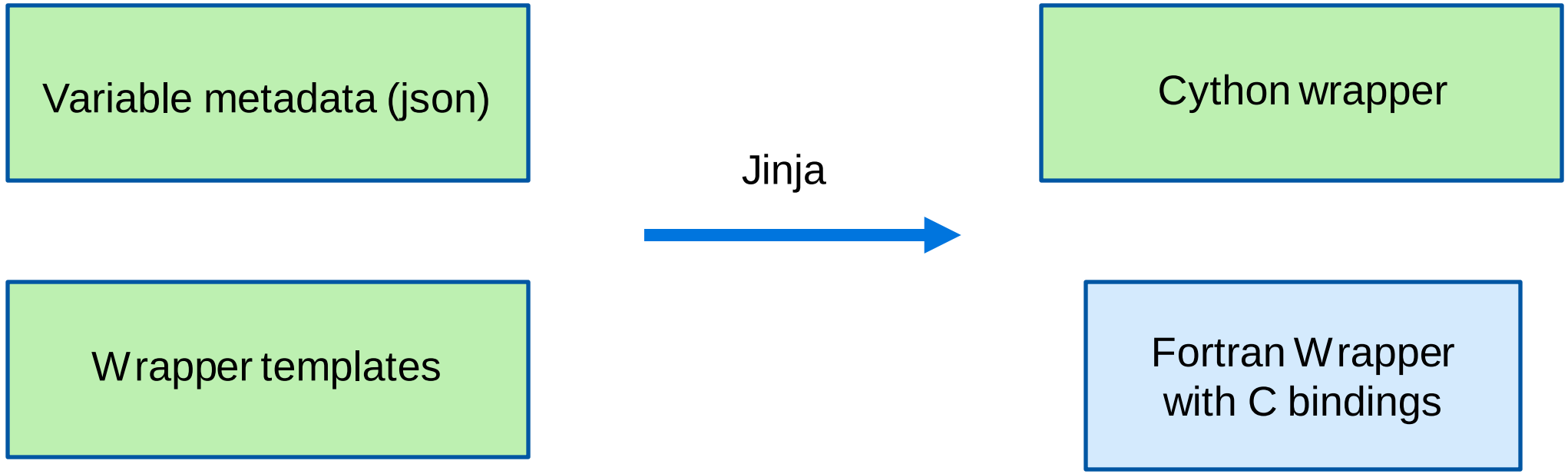


How is it done?

Compiled library structure



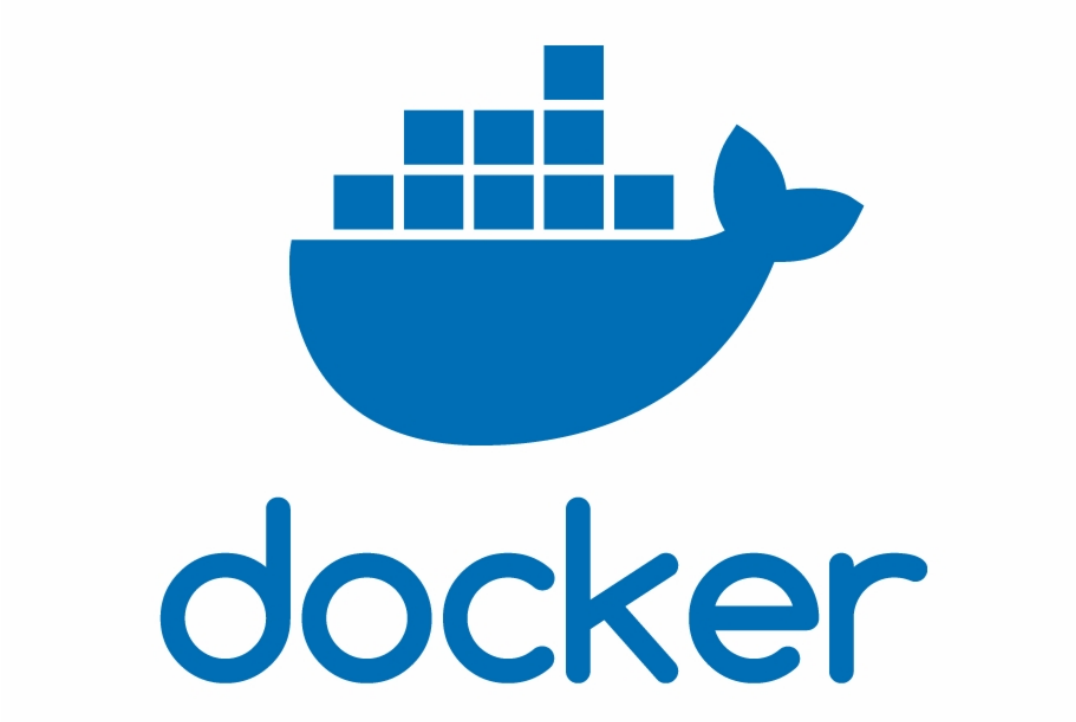
Code Generation



Portability and Docker

Why do we like Docker?

- Containerized FV3GFS in Docker runs reproducibly
- Docker image packages the runtime environment with the code
- Dockerfile acts as regression tested documentation for building the model
- Allows quickly and easily running a model on your laptop
- Pretty easy to learn



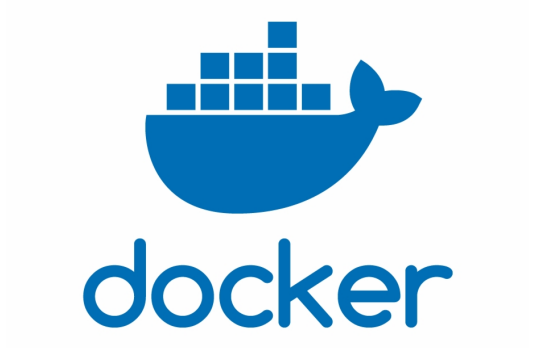
The Dockerfile

- Instructions to build an environment
- Base on an image with FROM
- Add sequential commands to set up the environment
- Not a complete definition
  - You build the environment at a specific point in your filesystem, used for COPY commands

```
FROM python:3.7.8-stretch

# install gcloud
RUN apt-get update && apt-get install -y  apt-transport-https ca-certificates gnupg curl gettext

RUN echo "deb [signed-by=/usr/share/keyrings/cloud.google.gpg] https://packages.cloud.google.com/
apt cloud-sdk main" | tee -a /etc/apt/sources.list.d/google-cloud-sdk.list &&\
    curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key --keyring /usr/share/
    keyrings/cloud.google.gpg add -
RUN apt-get update && apt-get install -y google-cloud-sdk

COPY constraints.txt /tmp/constraints.txt
COPY external/fv3fit/requirements.txt fv3fit/requirements.txt
RUN pip install -c /tmp/constraints.txt -r fv3fit/requirements.txt

COPY external/vcm/ /external/vcm/
COPY external/loaders/ /external/loaders/
COPY external/synth /external/synth
COPY external/fv3fit/ /fv3fit/

RUN apt-get update && apt-get install -y gfortran

COPY docker/fv3fit/entrypoint.sh /entrypoint.sh
RUN chmod +x /entrypoint.sh && \
    /entrypoint.sh

WORKDIR /fv3fit
ENTRYPOINT [ "/entrypoint.sh" ]
```

The Docker Image

- Fully defines an environment
- Built from a Dockerfile
  - `docker build -f Dockerfile -t image_name .`
- Can be:
  - used as a FROM target in Dockerfiles
  - pushed and pulled from the internet ("docker push" and "docker pull")
  - Used to start a Docker Container



The Docker Container

- A running environment
- Started from a Docker image
  - `docker run image_name <command>`
  - `docker run -it image_name bash`
- Used to run code in a defined environment
- NOT a virtual machine – much less expensive to run


```
(base) mcgibbon ~ $ docker run ubuntu:18.04 echo "hello"
hello
```

Workshop Contents

- Running the model in Jupyter notebooks to show features
- Running the model from the command-line
- More technical details about the wrapper
- A couple more features of Docker

Get in touch!

- Contact me at jeremym@vulcan.com or Spencer at spencerc@Vulcan.com
- Post issues to our Github repo https://github.com/VulcanClimateModeling/fv3gfs-wrapper
- Read more about our work https://www.vulcan.com/Special-Pages/Climate-Modeling.aspx

VULCAN

Geophysical Fluid Dynamics Laboratory
GFDL
NOAA

Let’s run this model!

* Commands prepared for OS X and Linux, need different commands for Windows

Get Docker

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

You can download and install Docker on multiple platforms. Refer to the following section and choose the best installation path for you.

Docker Desktop for Mac

A native application using the macOS sandbox security model which delivers all Docker tools to your Mac.

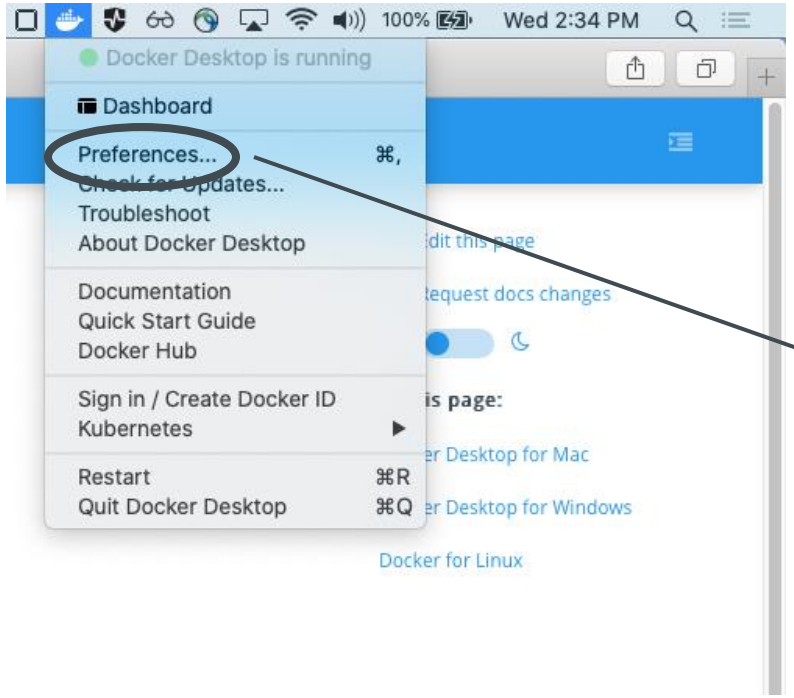
Docker Desktop for Windows

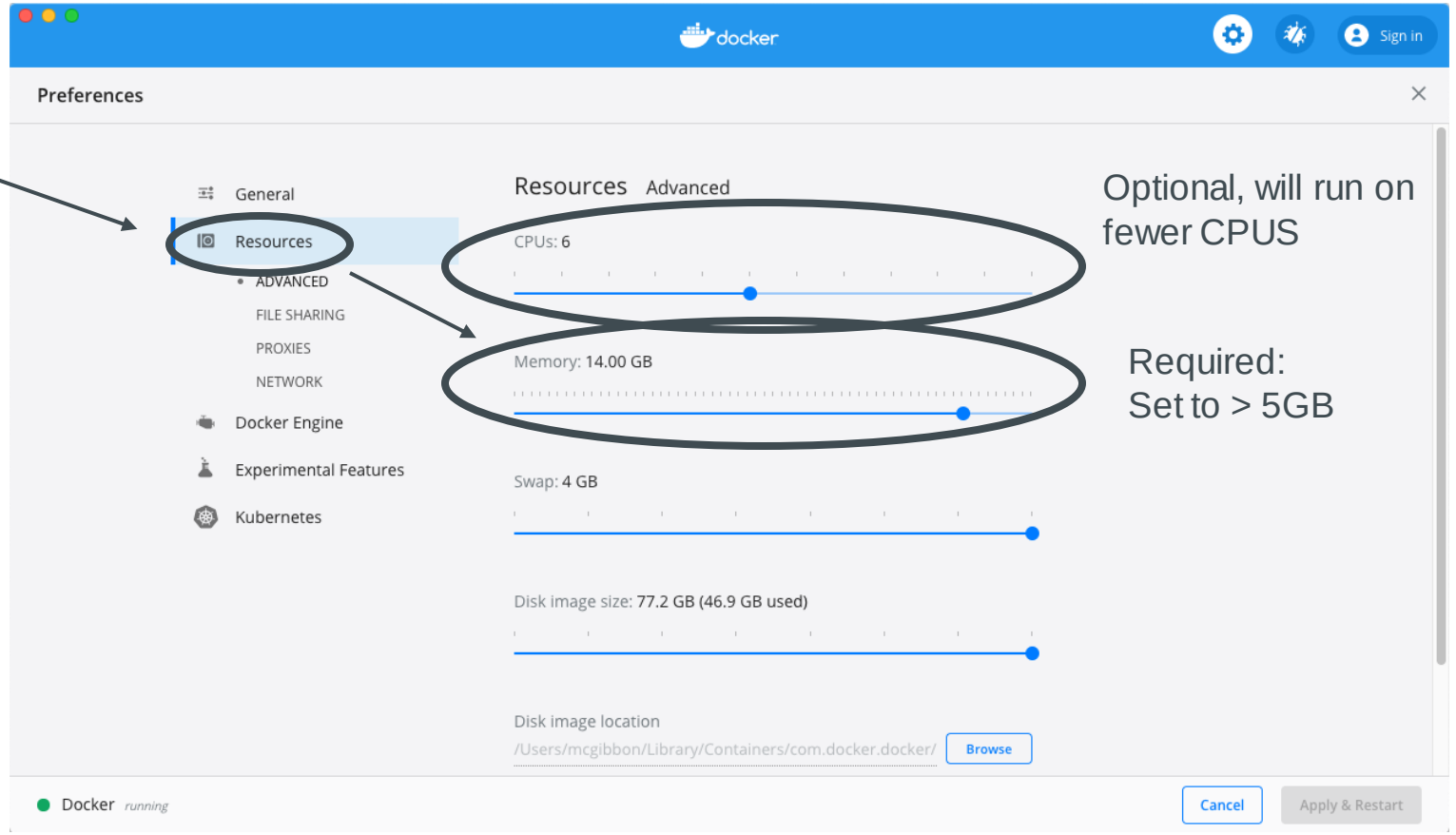
A native Windows application which delivers all Docker tools to your Windows computer.

Docker for Linux

Install Docker on a computer which already has a Linux distribution installed.

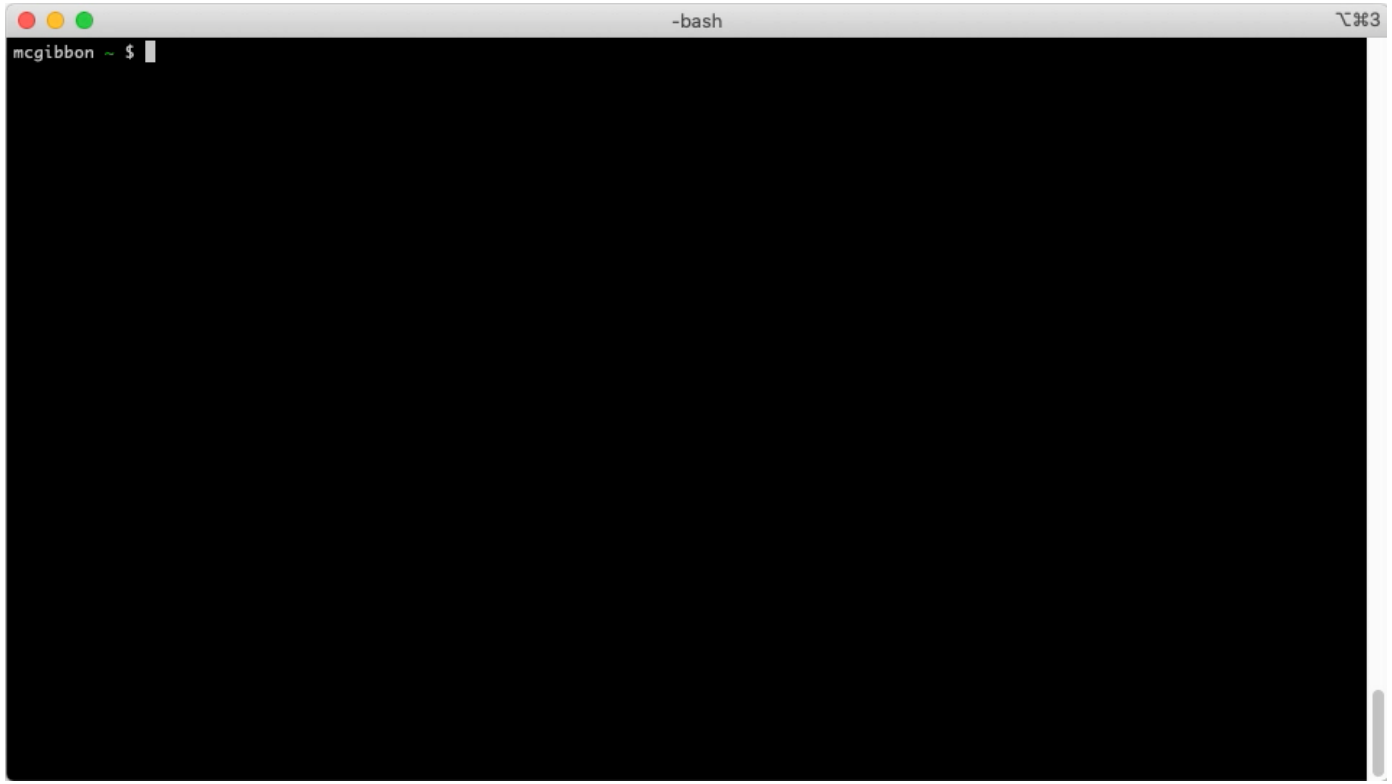
Docker, download, documentation, manual



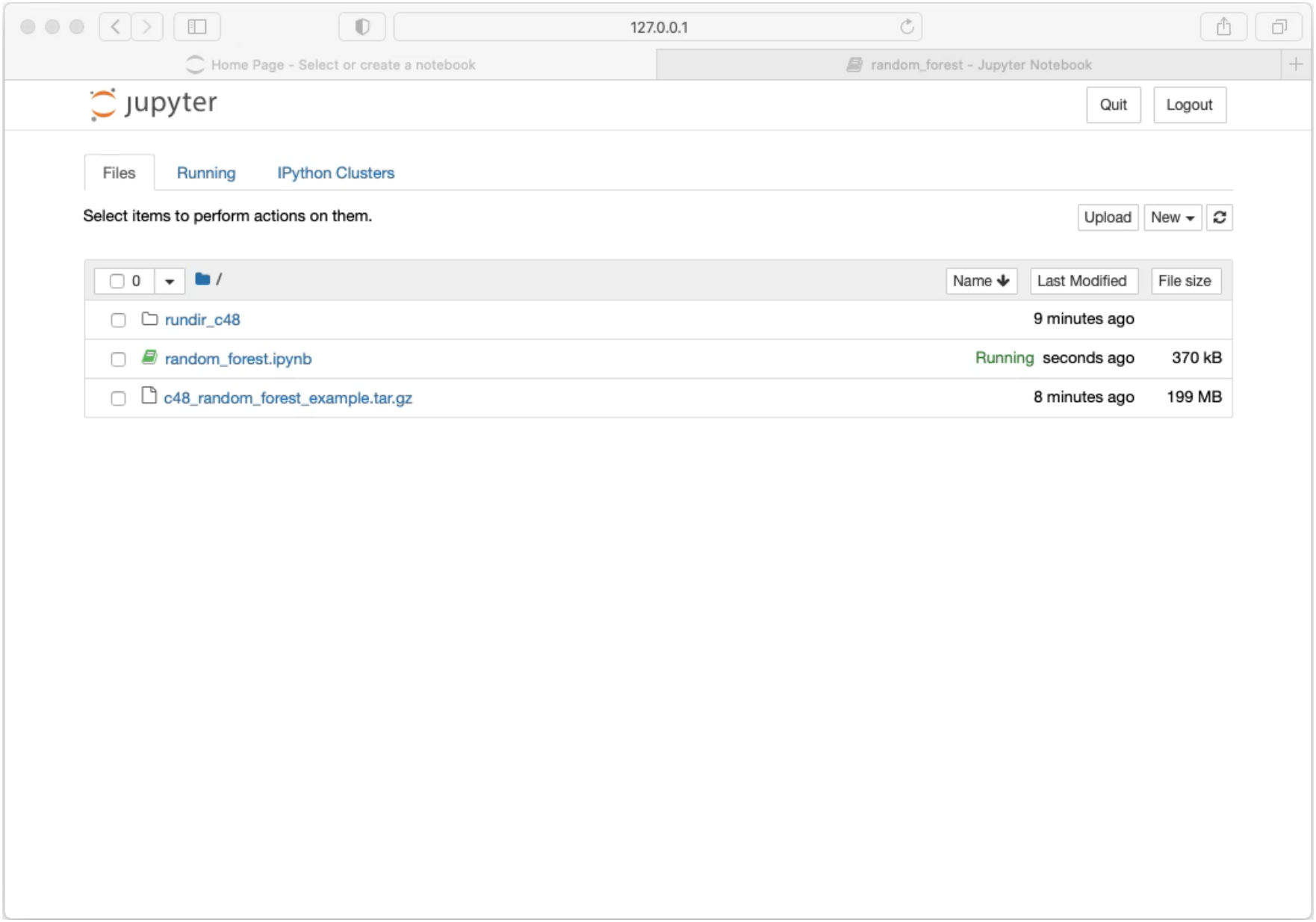


Allocate enough memory for Docker to run the model

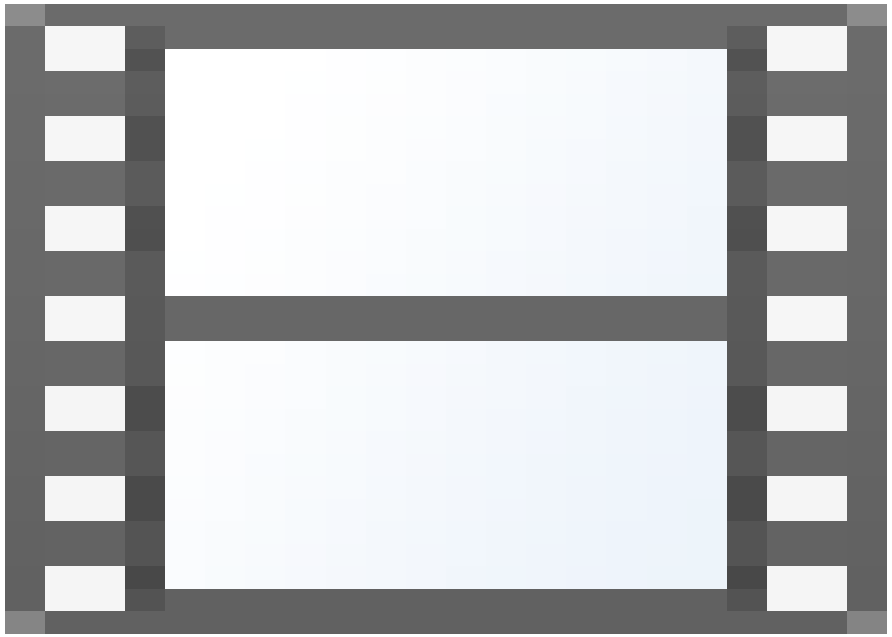
```
$ git clone https://github.com/VulcanClimateModeling/fv3gfs-wrapper.git
$ cd fv3gfs-wrapper
$ git checkout ams2021
$ cd examples/jupyter
$ make build run
```



Copy-paste the 127.0.0.1 URL into your browser


127.0.0.1
Home Page - Select or create a notebook
random_forest - Jupyter Notebook
jupyter
Quit
Logout
Files
Running
IPython Clusters
Select items to perform actions on them.
Upload
New
0
/
Name
Last Modified
File size
rundir_c48
9 minutes ago
random_forest.ipynb
Running seconds ago
370 kB
c48_random_forest_example.tar.gz
8 minutes ago
199 MB

```
$ git clone https://github.com/VulcanClimateModeling/fv3gfs-wrapper.git
$ cd fv3gfs-wrapper
$ git checkout ams2021
$ cd examples/runfiles
$ cat random_forest.py
$ wget https://storage.googleapis.com/vcm-ml-public/examples/c48_random_forest_example.tar.gz
$ tar -zxvf c48_random_forest_example.tar.gz
$ cp random_forest.py rundir_c48/
$ ls rundir_c48
$ docker run –w /rundir –v $(pwd)/rundir_c48:/rundir gcr.io/vcm-ml/fv3gfs-wrapper:ams2021 mpirun –n 6 python3 random_forest.py
```



https://github.com/VulcanClimateModeling/fv3gfs-wrapper

HISTORY.md	Remove initial fv3config cache from dockerfile (#186)	21 days ago
LICENSE	Split pure python code into subpackage (#33)	10 months ago
MANIFEST.in	Reduce coupling between fv3gfs and fv3util (#103)	4 months ago
Makefile	Update documentation and default build behavior for public reposit...	19 days ago
README.md	Update documentation and default build behavior for public reposit...	19 days ago
RELEASE.rst	Move wrapper and fv3util into namespaced packages under fv3gfs ...	4 months ago
dev_docker.sh	Use mpich only (#151)	2 months ago
fill_templates.py	Updates to allow fv3core to run with the wrapper (#142)	2 months ago
pyproject.toml	[VCMML-487] Use setup.py with pkg-config (#117)	4 months ago
requirements.txt	Remove initial fv3config cache from dockerfile (#186)	21 days ago
requirements_local.txt	Simplify python environment specification (#122)	4 months ago
setup.cfg	Move wrapper and fv3util into namespaced packages under fv3gfs ...	4 months ago
setup.py	Use mpich only (#151)	2 months ago
test_docker.sh	Use mpich only (#151)	2 months ago
tox.ini	Organize python wrapper into a package (#27)	14 months ago

Python 55.7% Fortran 29.4%
Shell 6.3% Makefile 5.6%
Dockerfile 3.0%

README.md

fv3gfs-wrapper (`import fv3gfs.wrapper`), is a Python wrapper for the FV3GFS global climate model.

See the documentation for more detailed instructions.

We also recommend you read the documentation for fv3gfs-util, which is used alongside fv3gfs-wrapper to provide Python model infrastructure and functionality.

This is currently alpha development software used for research. If you would like to contribute to or collaborate on the software, please get in touch with jeremym@vulcan.com or another developer.

All of this is documented, public, and open source

Take-aways

- We wrap FV3GFS (US weather model) in Python and containerize it in Docker
- You can run it on your laptop or workstation from the command-line or in a Jupyter notebook
- You can easily add online analysis code to modify model behavior
- We'll show you reproducible examples you can run at home

```python
import fv3gfs.wrapper

if __name__ == "__main__":
    fv3gfs.wrapper.initialize()
    for i in range(fv3gfs.wrapper.get_step_count()):
        fv3gfs.wrapper.step_dynamics()
        fv3gfs.wrapper.step_physics()
        fv3gfs.wrapper.save_intermediate_restart_if_enabled()
    fv3fs.wrapper.cleanup()
```