Open Science Grid

Special Cases: Licenses, Interpreted Languages, and Containers for DHTC

Wednesday morning, 10:45 am

Christina Koch (ckoch5@wisc.edu)

Research Computing Facilitator

University of Wisconsin - Madison

Expanding Our Horizons

- Previously, we were using simple, open source code that could be easily compiled or built.
- This presentation discusses some special cases:
  - Licensed software
  - Running interpreted languages
  - Using containers

LICENSING

Licensing

- Many scientific softwares are licensed.
- Licenses are restrictive, particularly for high-throughput computing

Open Science Grid

License Variations

- Per machine or 'single-install’
- Per *running* instance of the software (per “job”)
- Per username / user
- Via a license server
  - can support 1 - 1000s of concurrently running processes (“seats”)

Open Science Grid

Licensing implications for DHTC

- Per machine or 'single-install': can’t be used for DHTC
- Per job: restrictive, limits the number of jobs you can have running, how do you access licenses from execute servers?
- Username: restrictive, could only run jobs on one system where your jobs run as *your username*

Open Science Grid

Approaches

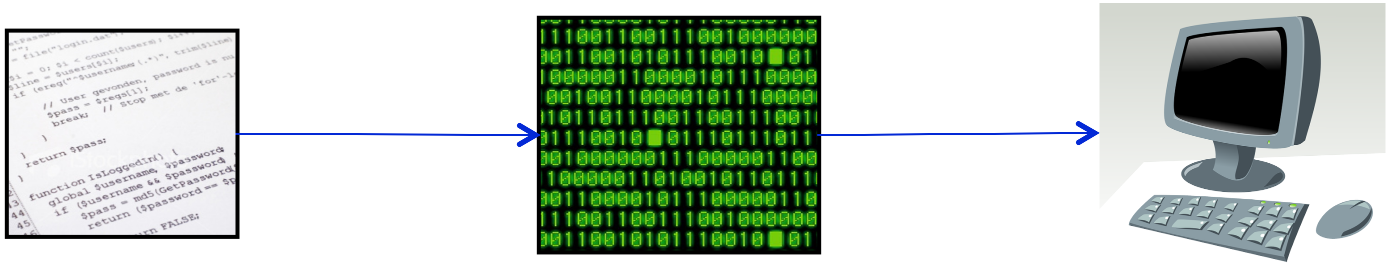
- Seek out open source alternatives
  - Python or R packages that emulate specific software behavior
  - If you can’t replace entire workflow, substitute free software where you can
- License-free workarounds (Matlab)
- Choose the least restrictive license possible

Open Science Grid

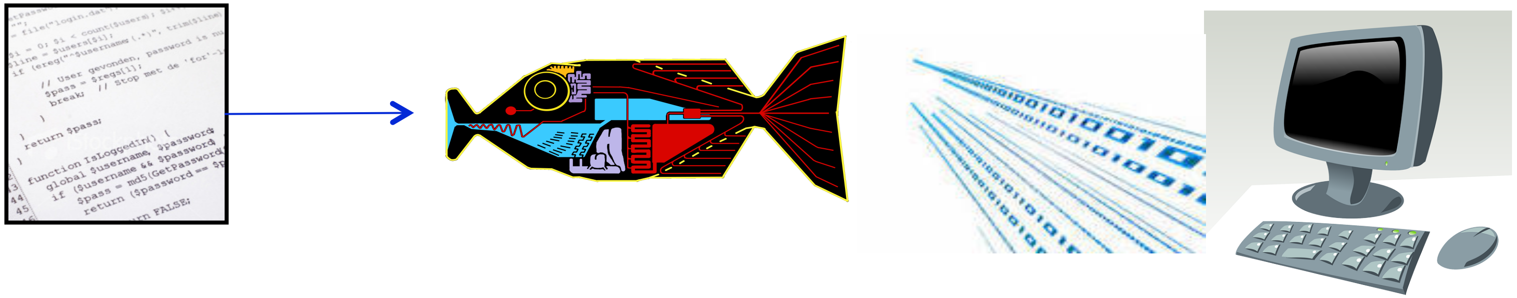
INTERPRETED LANGUAGES

Interpreted code

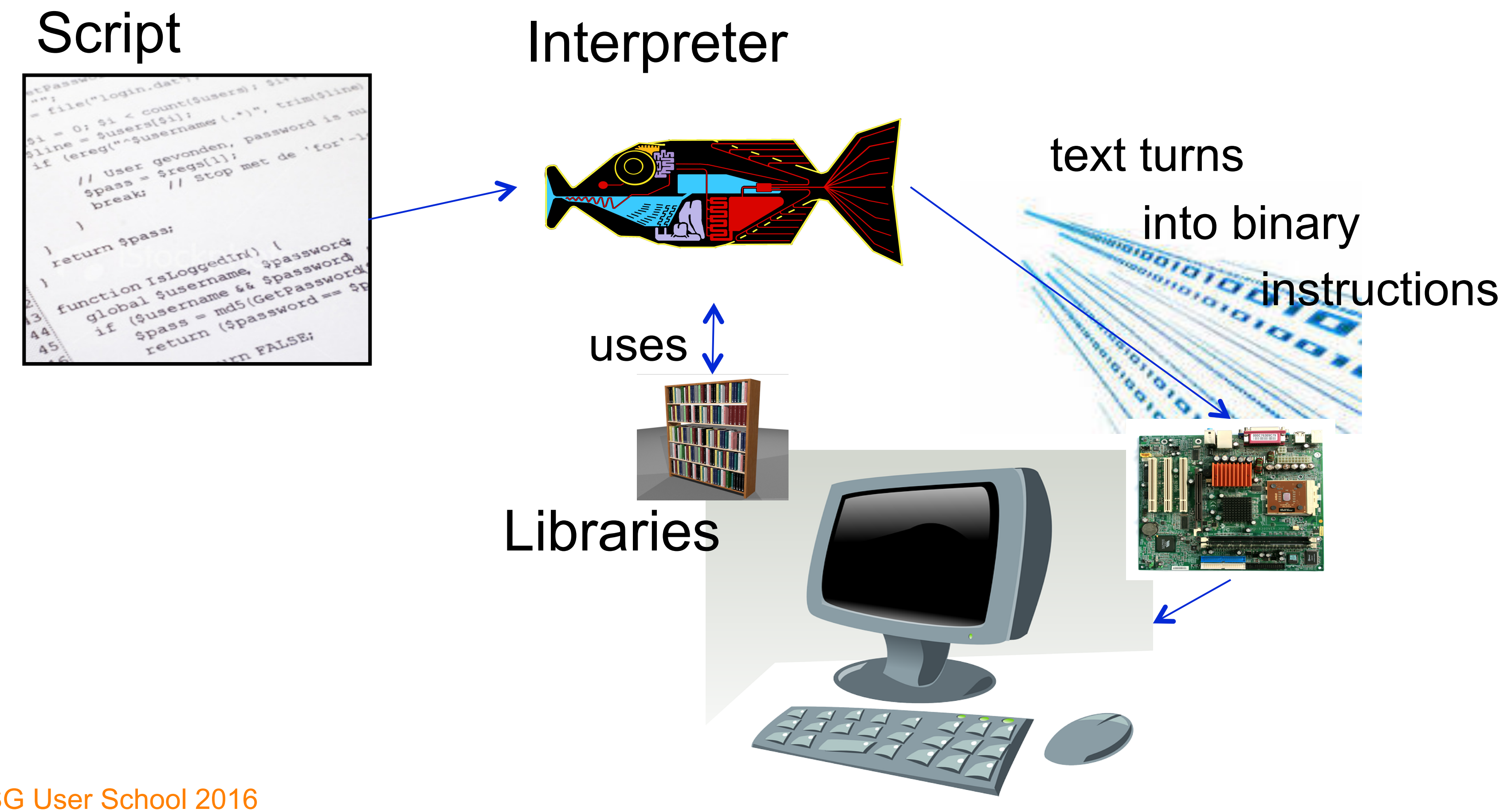
- Instead of being compiled and then run…



- …interpreted languages are translated into binary code "on the fly"

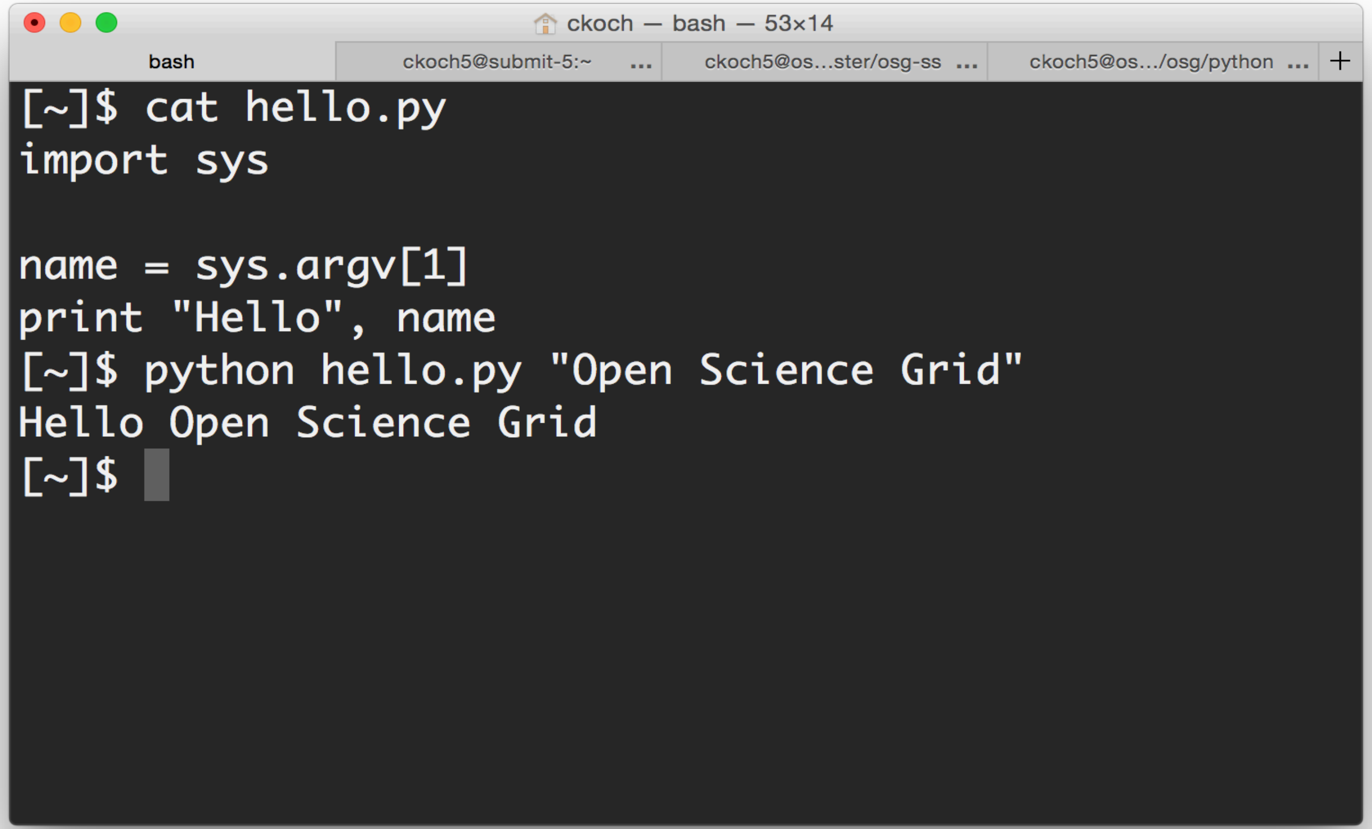


Interpretation



Open Science Grid

On the command line



```
[~]$ cat hello.py
import sys

name = sys.argv[1]
print "Hello", name
[~]$ python hello.py "Open Science Grid"
Hello Open Science Grid
[~]$
```

Open Science Grid

Common interpreted languages*

- Python
- R
- Julia
- Ruby
- Matlab
- Perl
- Javascript

*Note: the line between interpreted/compiled languages can be fuzzy. Many languages support both options, with one method being more common.

Running interpreted code in jobs

General procedure

- Need to bring along interpreter and script
- Use a wrapper script as the executable
- Wrapper script will:
  - “Install” the interpreter
  - Run the script using the local installation

Open Science Grid

Python on DHTC

1. Create a portable Python installation (optional)
2. Bring along:
   - pre-built installation OR Python source code
   - your Python code
3. Use a wrapper script to:
   - unpack pre-built install OR install from source
   - run your Python script

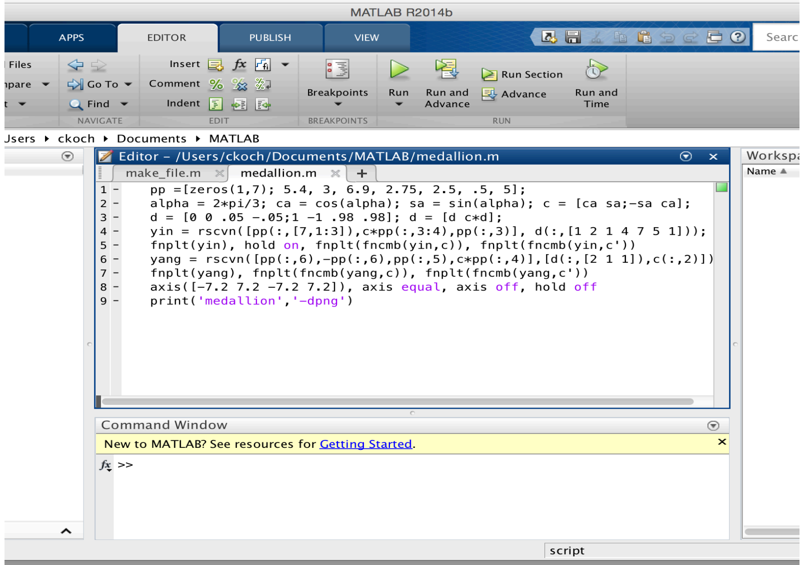
Matlab

- Wait a minute…isn't Matlab licensed?
- Yes, when interpreted on your computer using a normal Matlab installation.
- However, Matlab code can also be compiled.
- Once compiled, the code can be run without a license using a (free) set of files called the Matlab runtime (which acts like the interpreter).

Matlab contrast

Running Matlab on your computer

Uses license per instance



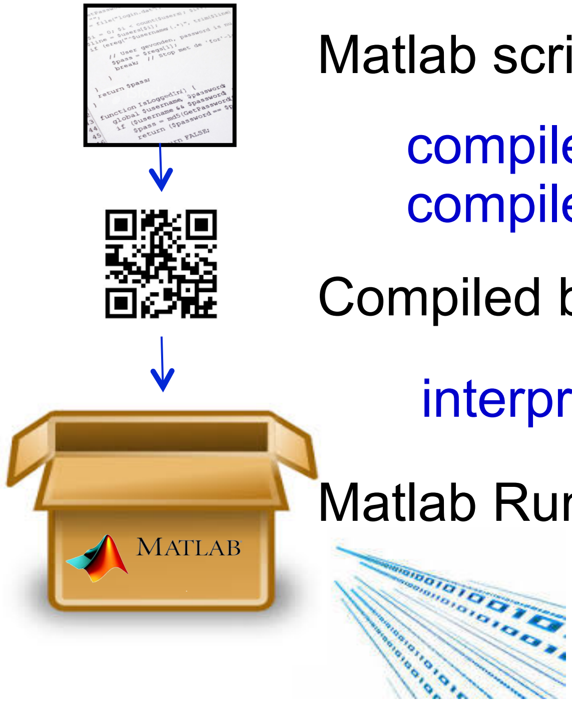
```
pp =[zeros(1,7); 5.4, 3, 6.9, 2.75, 2.5, .5, 5];
alpha = 2*pi/3; ca = cos(alpha); sa = sin(alpha); c = [ca sa;-sa ca];
d = [0 0 .05 -.05;1 -1 .98 .98]; d = [d c*d];
yin = rscvn([pp(:,[7,1:3]),c*pp(:,3:4),pp(:,3)], d(:,[1 2 1 4 7 5 1]));
fnplt(yin), hold on, fnplt(fncmb(yin,c)), fnplt(fncmb(yin,c'))
yang = rscvn([pp(:,6),-pp(:,6),pp(:,5),c*pp(:,4)],[d(:,[2 1 1]),c(:,2)])
fnplt(yang), fnplt(fncmb(yang,c)), fnplt(fncmb(yang,c'))
axis([-7.2 7.2 -7.2 7.2]), axis equal, axis off, hold off
print('medallion','-dpng')
```

New to MATLAB? See resources for Getting Started.

```
>>
```

Running Matlab on DHTC

Uses license once, runs many instances for free



Matlab script(s)

compiled w/ Matlab compiler (uses license)

Compiled binary

interpreted by

Matlab Runtime (free)

Open Science Grid

Matlab on DHTC

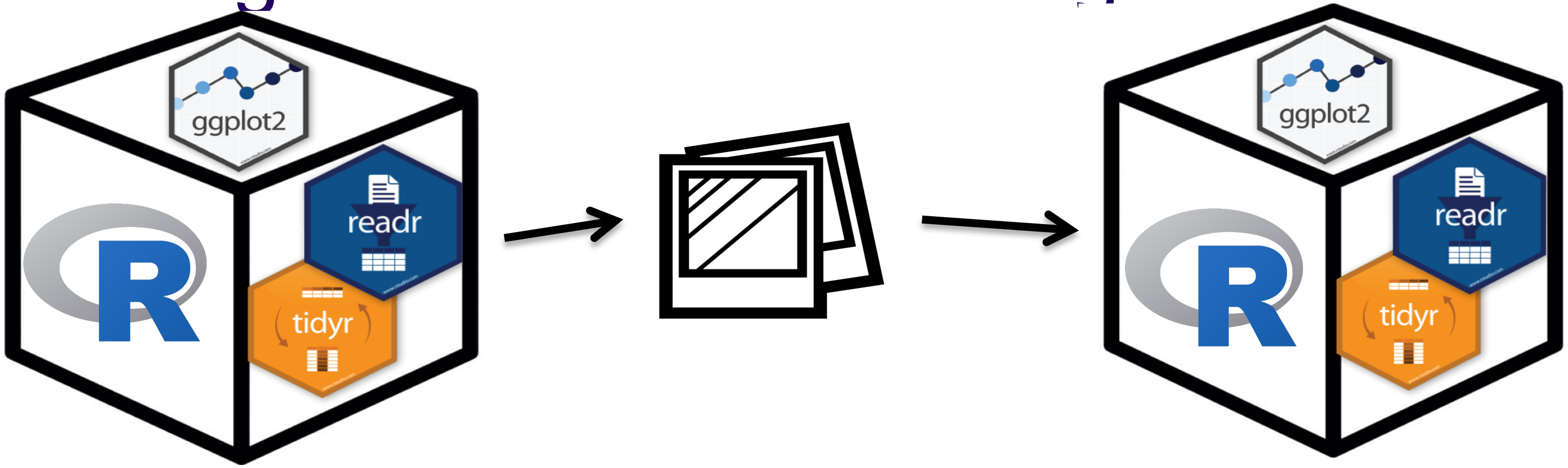
1. Compile Matlab code using the Matlab compiler (mcc)
   - requires a license
2. Prepare a copy of the Matlab runtime
   - download for free from Mathworks
3. Write a script that “installs” the runtime
   - The Matlab compiler actually writes most of this script for you
4. Use the runtime install to run the compiled Matlab code

CONTAINERS

Open Science Grid

Containers

- Containers are a tool for capturing an entire job “environment” (software, libraries, operating system) into an “image” that can be used again.



polaroid photos by Nick Bluth from the Noun Project

Open Science Grid

Using Containers in DHTC

- Two common container systems:

Docker

https://www.docker.com/

Singularity

http://singularity.lbl.gov/

- Requirements:
  - Underlying container system needs to be installed on the computers where your job runs
  - Permissions on that system allow the use of containers

Open Science Grid

Container Workflow

1. Create a container or find one online
   - DockerHub: https://hub.docker.com/
   - SingularityHub: https://singularity-hub.org/faq
2. Place container into public or private registry
3. Create a customized script/submit file that fetches/uses the container
   - Docker: Use HTCondor’s docker universe
   - Singularity: Wrapper script

Open Science Grid

Conclusion

To use any software in a DHTC system:

1. Create environment/software package
   - download pre-compiled code, compile your own, build your own, create/find a container
2. Write a script to set up the environment when the job runs
3. Account for all dependencies, files, and requirements in the submit file

Open Science Grid

Exercises

- Running Matlab Jobs
  - Exercise 1.6
- Running Python Jobs
  - Exercise 1.7: Pre-building Python and using that installation
  - Exercise 1.8: Writing a script that installs Python with every job
- Half of the room should start with Matlab, the other with Python

Open Science Grid

Questions?

- Now: Hands-on Exercises
  - 11:15am-12:15pm
- Next:
  - 12:15-1:15pm: Lunch
  - 1:15 onward: free time