

PY410 / 505
Computational Physics 1

Salvatore Rappoccio

Technical Lectures

- We will have several technical lectures to get you up to speed with programming and technical skills
 - Using UNIX/LINUX command line
 - Compiling, version control
 - Data representations, flow of control, object oriented programming, simple algorithms

Executing code

- We will have two environments officially supported:
 - Docker image for class software (Ubuntu based):
 - <https://hub.docker.com/r/srappoccio/compphys>
 - Vidia (for those who register at UB)
 - <https://vidia.ccr.buffalo.edu>
- Software for this class:
 - <https://github.com/rappoccio/CompPhys>
- If you want to use your laptop environment directly, you can but CAVEAT EMPTOR.

I assist students with laptop issues on best-effort basis

VIDIA : For those registered at UB

- Registering: <https://vidia.ccr.buffalo.edu>



VIDIA

DISCOVER
COMMUNITY
ABOUT
USER SUPPORT

LOGIN / SIGN UP

Virtual Infrastructure for Data Intensive Analysis

Data-intensive computing and analysis tool for SUNY
students and faculty

LEARN MORE ABOUT VIDIA



VIDIA : Logging in the first time

Salvatore Rocco Rappoccio Dashboard

Dashboard Introduction

Welcome to your customizable dashboard page!

To get started, click the "Add Modules" button towards the top of this page. You will then be presented with a list of modules you may add to your page. You may also, at that time, remove any unwanted modules or rearrange the current modules by drag-and-drop!

My Sessions

No active sessions found.

Storage (manage)

34% of 0.9GB

My Tools

Recent Favorites All Tools

Jupyter

pshekhar_project Jupyter Notebook

Workspace

These are your most recently used tools.

Activity 278

Dashboard

Profile

Groups 2

Account

Contributions

Usage

Collections 1

Messages 1

Projects

Citations



Scroll down

VIDIA : Logging in the first time

Jupyter notebooks

Tips and Documentation

VIDIA Tips

- [VIDIA Knowledge Base](#)
- [Using VIDIA](#)
- [HUBzero user documentation](#)

Tools and Documentation

- [Jupyter Launch the tool for docs access](#)
- [R and RStudio](#)
- [RapidMiner](#)
- [PSPP](#)

File Access

- [Upload and download your files](#)
- [File access with UBBBox](#)

Linux workspace

Tool Development

- [Tool Development Pipeline](#)
- [Workspace tool \(requires group membership\)](#)

VIDIA tool developer resources

- [VIDIA Tool Development Workflow](#)
- [Publish Jupyter Notebooks](#)
- [Calling Submit](#)
- [Tool GUIs with Rapture](#)

HUBzero tool developer resources

- [Using the Workspace](#)
- [Invoke scripts](#)
- [HUBbub 2014 Tool Development Training](#)

More information

- [VIDIA Dev Resources](#)
- [HUBzero Dev Documentation](#)

VIDIA : After you have used Workspace or Jupyter

- Workspaces and Jupyter

The screenshot displays the VIDIA dashboard for user Salvatore Rocco Rappoccio. The top navigation bar includes 'VIDIA', 'Discover', 'Community', 'About', 'Support', and a user profile section with the name 'Salvatore Rocco Rappoccio' and email 'srrappoc@buffalo.edu'. Below this is a search bar and a help icon. The main dashboard area is titled 'Salvatore Rocco Rappoccio Dashboard' and features an 'Add Modules' button. The dashboard is organized into three main panels:

- Dashboard Introduction:** A welcome message and instructions on how to use the dashboard, including the 'Add Modules' button.
- My Sessions:** A section showing 'No active sessions found.' and a storage usage indicator: 'Storage (manage) 5% of 0.9GB'.
- My Tools:** A section showing 'Recent', 'Favorites', and 'All Tools' tabs. Under the 'Recent' tab, 'Workspace' and 'Jupyter' are listed as recently used tools. Below the list, it says 'These are your most recently used tools.'

Two red callout boxes with arrows point to the 'Workspace' and 'Jupyter' entries in the 'My Tools' panel. The first callout box is labeled 'Linux workspace' and points to the 'Workspace' entry. The second callout box is labeled 'Jupyter notebooks' and points to the 'Jupyter' entry.

VIDIA : Workspace tool

The image shows two overlapping panels from the VIDIA documentation. The top panel, titled 'Tips and Documentation', contains the following sections:

- VIDIA Tips**
 - [VIDIA Knowledge Base](#)
 - [Using VIDIA](#)
 - [HUBzero user documentation](#)
- Tools and Documentation**
 - [Jupyter Launch the tool for docs access](#)
 - [R and RStudio](#)
 - [RapidMiner](#)
 - [PSPP](#)
- File Access**
 - [Upload and download your files](#)
 - [File access with UBBBox](#)

The bottom panel, titled 'Tool Development', contains the following sections:

- [Tool Development Pipeline](#)
- [Workspace tool \(requires group membership\)](#)
- VIDIA tool developer resources**
 - [VIDIA Tool Development Workflow](#)
 - [Publish Jupyter Notebooks](#)
 - [Calling Submit](#)
 - [Tool GUIs with Rapture](#)
- HUBzero tool developer resources**
 - [Using the Workspace](#)
 - [Invoke scripts](#)
 - [HUBbub 2014 Tool Development Training](#)
- More information**
 - [VIDIA Dev Resources](#)
 - [HUBzero Dev Documentation](#)

Linux workspace



VIDIA

VIDIA

Discover

Community

About

Support



Salvatore Rocco Rappoccio
srappoc@buffalo.edu

Resources



Tags

Search



Collect

Workspace

Development workspace

Launch Tool

Version 1.3 - published on 24 Oct 2013

This tool is closed source.

[View All Supporting Documents](#)

44 users, detailed usage

Share: [f](#) [t](#) [w](#) ...

0 Citation(s)

0 questions (Ask a question)

0 wish(es) (New Wish)

Launch Tool (hard to see)

VIDIA

VIDIA

Discover

Community

About

Support

Salvatore Rocco Rappoccio
srrappoc@buffalo.edu

Search

?

Exit

Pause

Workspace



terminate



Keep for later

Color xterm

```
srrappoc@vidia:~$
```

Access to menu (including Firefox)

Linux shell

HUB! #1 #2 #3 xterm

12:03 Aug 22

Storage (manage)

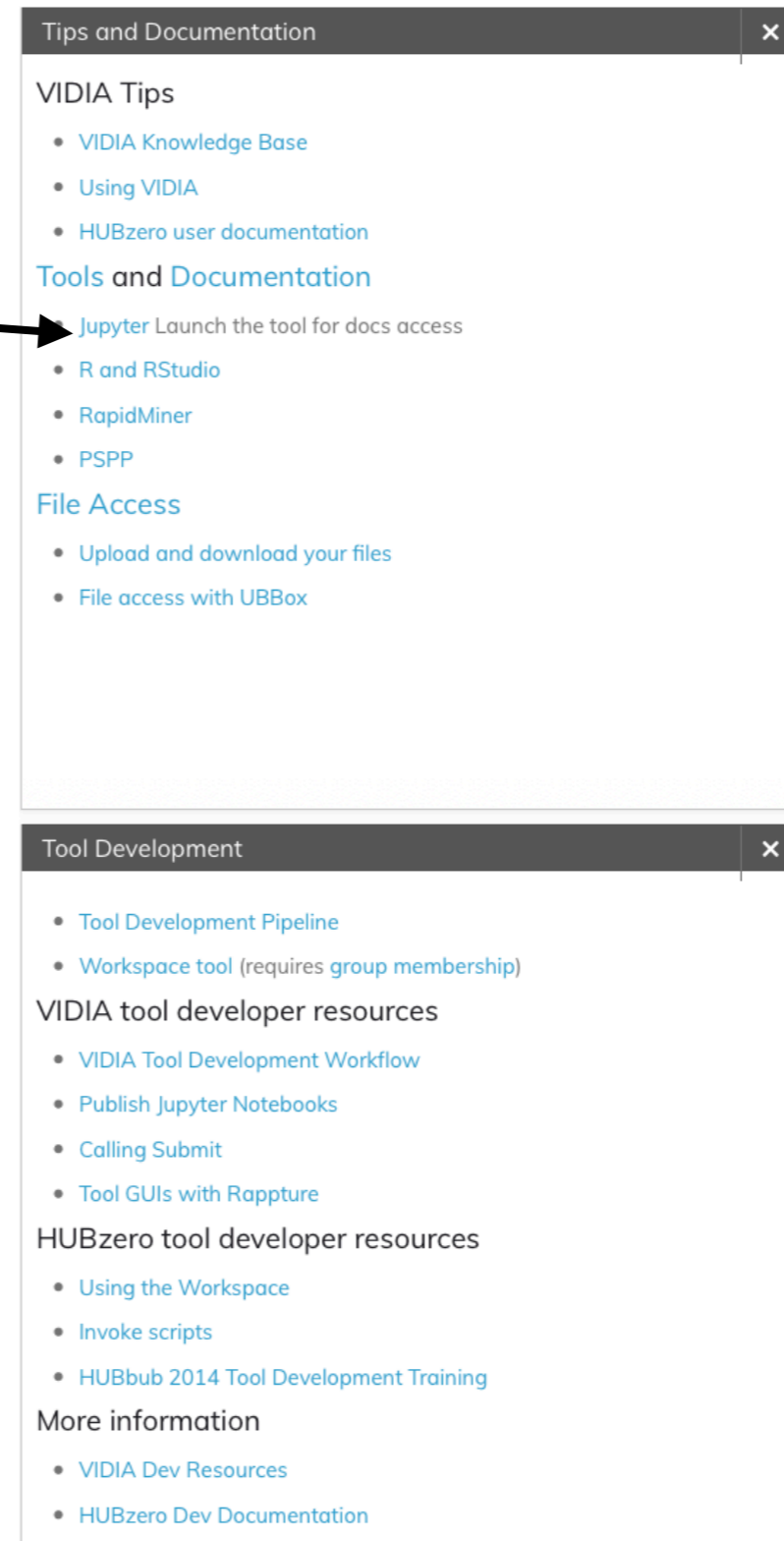
5% of 0.9GB



800 x 600

VIDIA : Jupyter

Jupyter notebooks



Tips and Documentation

VIDIA Tips

- [VIDIA Knowledge Base](#)
- [Using VIDIA](#)
- [HUBzero user documentation](#)

Tools and Documentation

- [Jupyter](#) Launch the tool for docs access
- [R and RStudio](#)
- [RapidMiner](#)
- [PSPP](#)

File Access

- [Upload and download your files](#)
- [File access with UBBBox](#)

Tool Development

- [Tool Development Pipeline](#)
- [Workspace tool](#) (requires [group membership](#))

VIDIA tool developer resources

- [VIDIA Tool Development Workflow](#)
- [Publish Jupyter Notebooks](#)
- [Calling Submit](#)
- [Tool GUIs with Rapture](#)

HUBzero tool developer resources

- [Using the Workspace](#)
- [Invoke scripts](#)
- [HUBbub 2014 Tool Development Training](#)

More information

- [VIDIA Dev Resources](#)
- [HUBzero Dev Documentation](#)

Jupyter

- Can also use Jupyter notebooks
- Also can be installed on your own machines if you want
- Nice little package for code and documentation at once

VIDIA  jupyter Index (read only)

 [Click Here to go to your Home Directory](#)

READ THIS: IMPORTANT CHANGES

You Are Running Jupyter 5.1

Jupyter

jupyter

Quit

Files **Running**

Select items to perform actions on them.

Upload **New** ↕

Name ↓ Last Modified File size

New → Python3

Gives you a notebook:

The screenshot shows the Jupyter notebook interface. At the top left, there is a 'VIDIA' logo and the 'jupyter' logo. The current notebook is titled 'Untitled1' and has a status bar indicating 'Last Checkpoint: a few seconds ago (unsaved changes)'. On the top right, there is a 'Python' logo and a 'Terminate Session' button. Below the title bar is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. To the right of the menu bar, there is a 'Trusted' indicator, a pencil icon, and 'Python3' with a dropdown arrow. Below the menu bar is a toolbar with various icons for file operations (save, new, copy, paste, undo, redo), execution (run, stop, refresh), and view toggling (code, dashboard, help). The main area of the notebook shows a code cell with the prompt 'In []:' followed by an empty input field.

Jupyter

jupyter Untitled1 Last Checkpoint: 3 minutes ago (unsaved changes)



File Edit View Insert Cell Kernel Widgets Help

Python2

Code CellToolbar Dashboard View: </> [grid icon] [refresh icon]

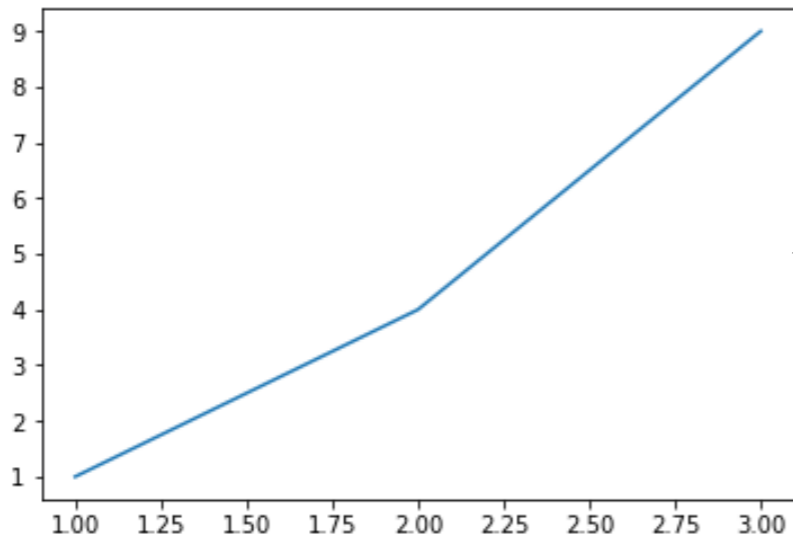
```
In [3]: import matplotlib
import matplotlib.pyplot as plt
import array

x = array.array('f', [1,2,3])
y = array.array('f', [1,4,9])

plt.plot(x,y)

plt.show()
```

Type python



Shift+Enter: It executes



In []:

Jupyter

jupyter Untitled1 Last Checkpoint: 37 minutes ago (unsaved changes)



File Edit View Insert Cell Kernel Widgets Help

Python2

Markdown CellToolbar Dashboard View: </>

```
In [3]: import matplotlib
import matplotlib.pyplot as plt
import array

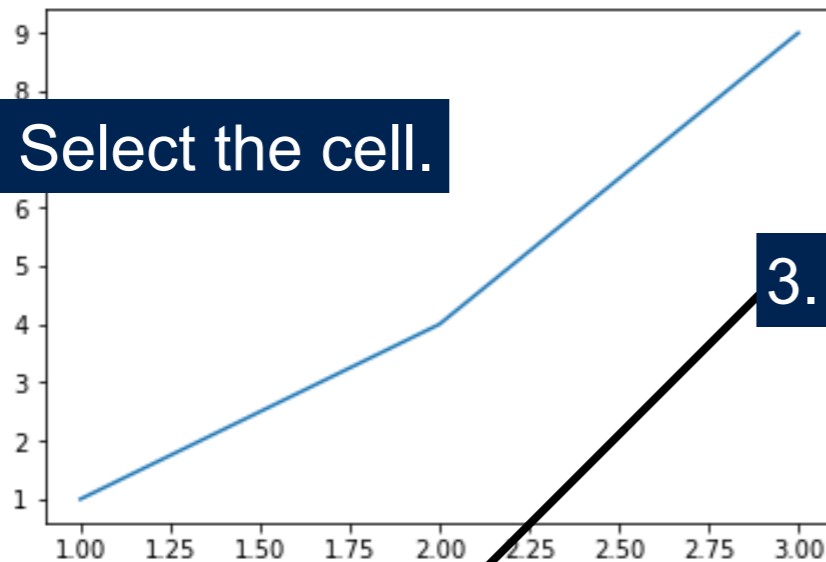
x = array.array('f', [1,2,3])
y = array.array('f', [1,4,9])

plt.plot(x,y)

plt.show()
```

2. Change cell to "Markdown"

1. Select the cell.



3. LaTeX goes in "\$ (stuff here) \$"

This is a markdown cell.

You can use LaTeX like this : $y = x^2$. Note of caution: it uses Mathjax, so not everything is supported.

Jupyter

jupyter Untitled1 Last Checkpoint: 37 minutes ago (unsaved changes)



File Edit View Insert Cell Kernel Widgets Help

Python2

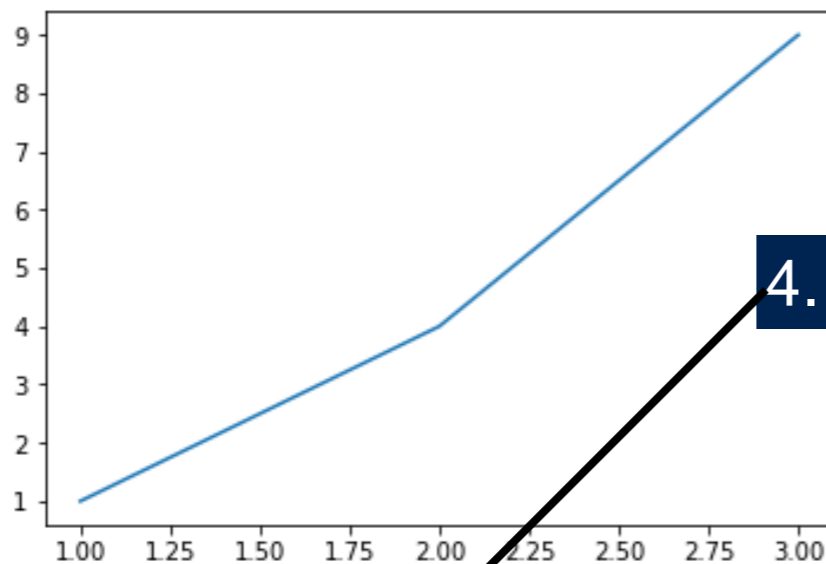
Markdown CellToolbar Dashboard View: </>

```
In [3]: import matplotlib
import matplotlib.pyplot as plt
import array

x = array.array('f', [1,2,3])
y = array.array('f', [1,4,9])

plt.plot(x,y)

plt.show()
```



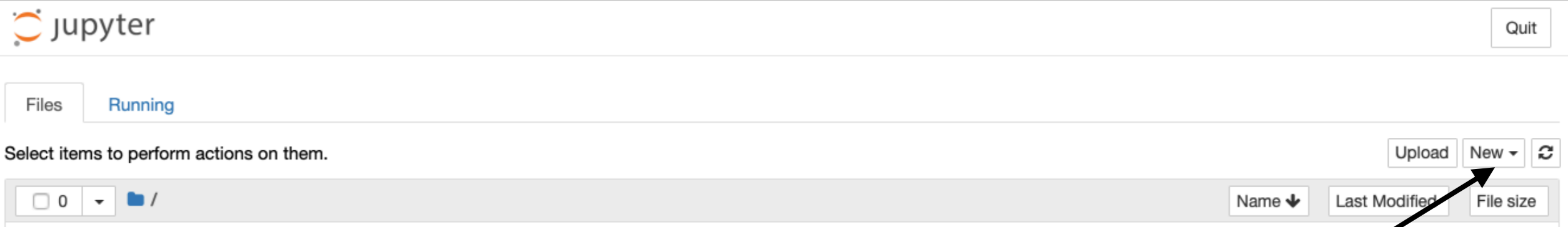
4. Shift+Click, and you get the markdown

This is a markdown cell.

You can use TeX like this : $y = x^2$. Note of caution: it uses Mathjax, so not everything is supported.

Jupyter

- But wait, it gets better! If you want, you can use the Jupyter terminal directly instead of the linux shell.



New —> Terminal

Gives you:

jupyter

```
srrappoc@vidia:~$
```

Jupyter

- But wait, it gets better! If you want, you can use the Jupyter terminal directly instead of the linux shell.
- Can get the code for the class directly!



```
srrappoc@vidia:~$ git clone https://github.com/rappoccio/CompPhys.git
Cloning into 'CompPhys'...
remote: Enumerating objects: 56, done.
remote: Counting objects: 100% (56/56), done.
remote: Compressing objects: 100% (50/50), done.
remote: Total 56 (delta 3), reused 56 (delta 3), pack-reused 0
Unpacking objects: 100% (56/56), done.
Checking connectivity... done.
srrappoc@vidia:~$ █
```


Jupyter

jupyter Quit

Files Running

Select items to perform actions on them. Upload New ▾ ↻

<input type="checkbox"/> 0 ▾	📁 /	Name ▾	Last Modified	File size
<input type="checkbox"/>	📁 CompPhys		a minute ago	
<input type="checkbox"/>	📁 data		7 months ago	
<input type="checkbox"/>	📁 Desktop		2 years ago	
<input type="checkbox"/>	📁 Downloads		4 months ago	
<input type="checkbox"/>	📁 notebooks		9 months ago	

Click on "CompPhys"

Next page is this:

jupyter Quit

Files Running

Select items to perform actions on them. Upload New ▾ ↻

<input type="checkbox"/> 0 ▾	📁 / CompPhys	Name ▾	Last Modified	File size
	📁 ..		seconds ago	
<input type="checkbox"/>	📁 JupyterExamples		6 minutes ago	19

Click to "JupyterExamples", then open the file example_jupyter.ipynb

Jupyter

VIDIA jupyter example_jupyter (autosaved) Python3 Trusted Terminate Session

File Edit View Insert Cell Kernel Widgets Help

Code Dashboard View: </>

Example using Jupyter

We will generate linear data with Gaussian noise and fit it to a straight line.

First, we import the libraries.

```
In [1]: import numpy as np
import matplotlib.pyplot as plt
```

Next, we create some simple x and y data from a linear model with some small Gaussian noise.

```
In [2]: N=100
x = np.linspace(0,10,101)
y = np.random.normal(scale=1,loc=x)

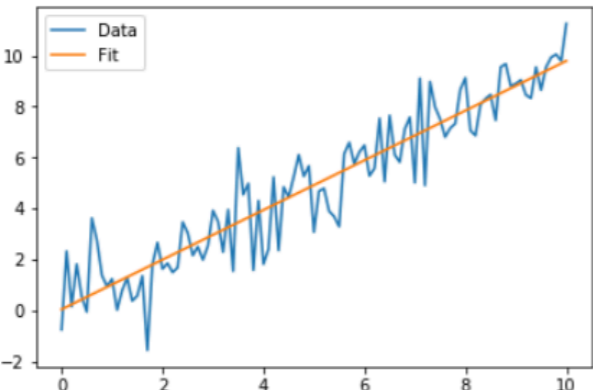
p, residuals, _, _, _ = np.polyfit(x, y, 1, full=True)
chisq_dof = residuals / (len(x) - 2)

print ('Fit results: y = %3.2f x + %3.2f' % (p[0], p[1]) )
```

Fit results: y = 0.98 x + 0.04

Finally, we plot the results.

```
In [3]: yfit = p[0] * x + p[1]
plt.plot(x,y, label='Data')
plt.plot(x,yfit, label='Fit')
plt.legend()
plt.show()
```

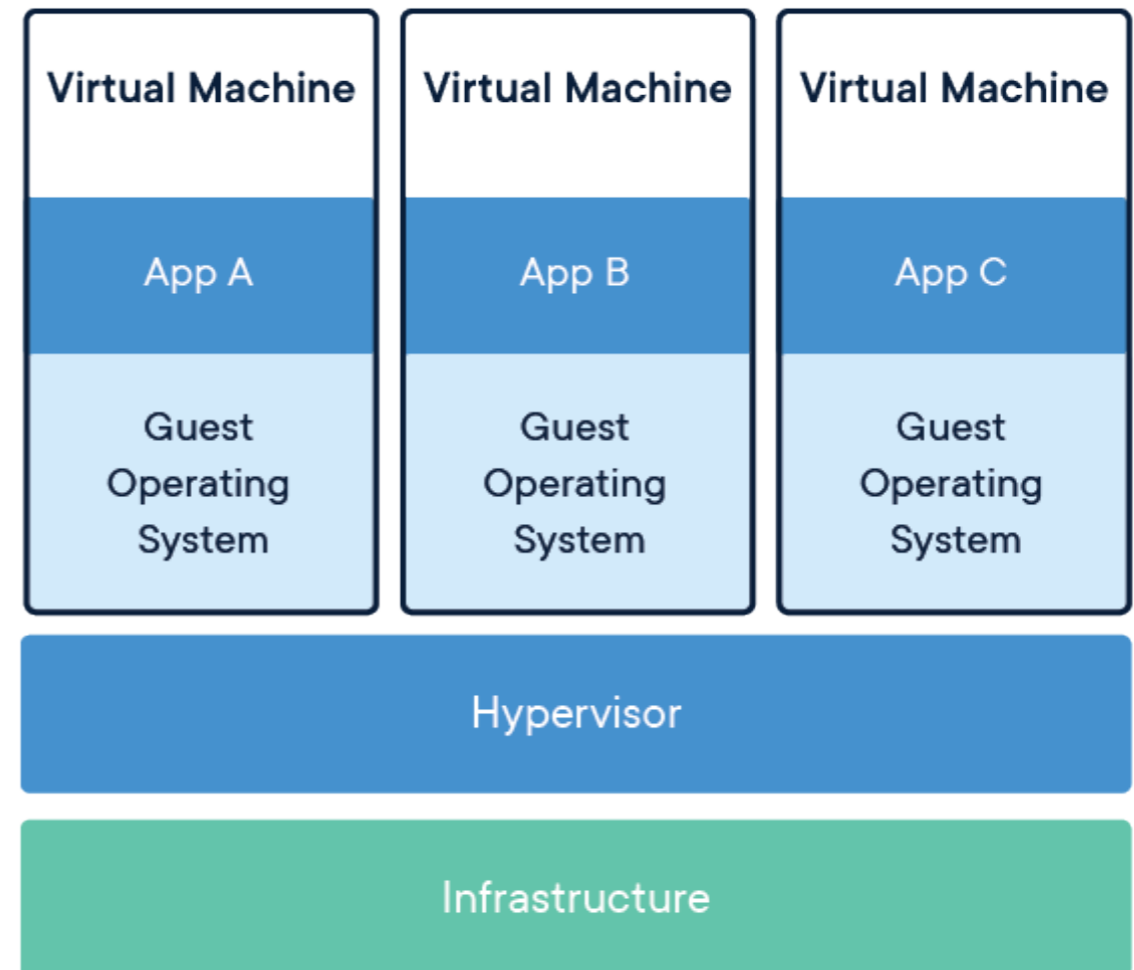
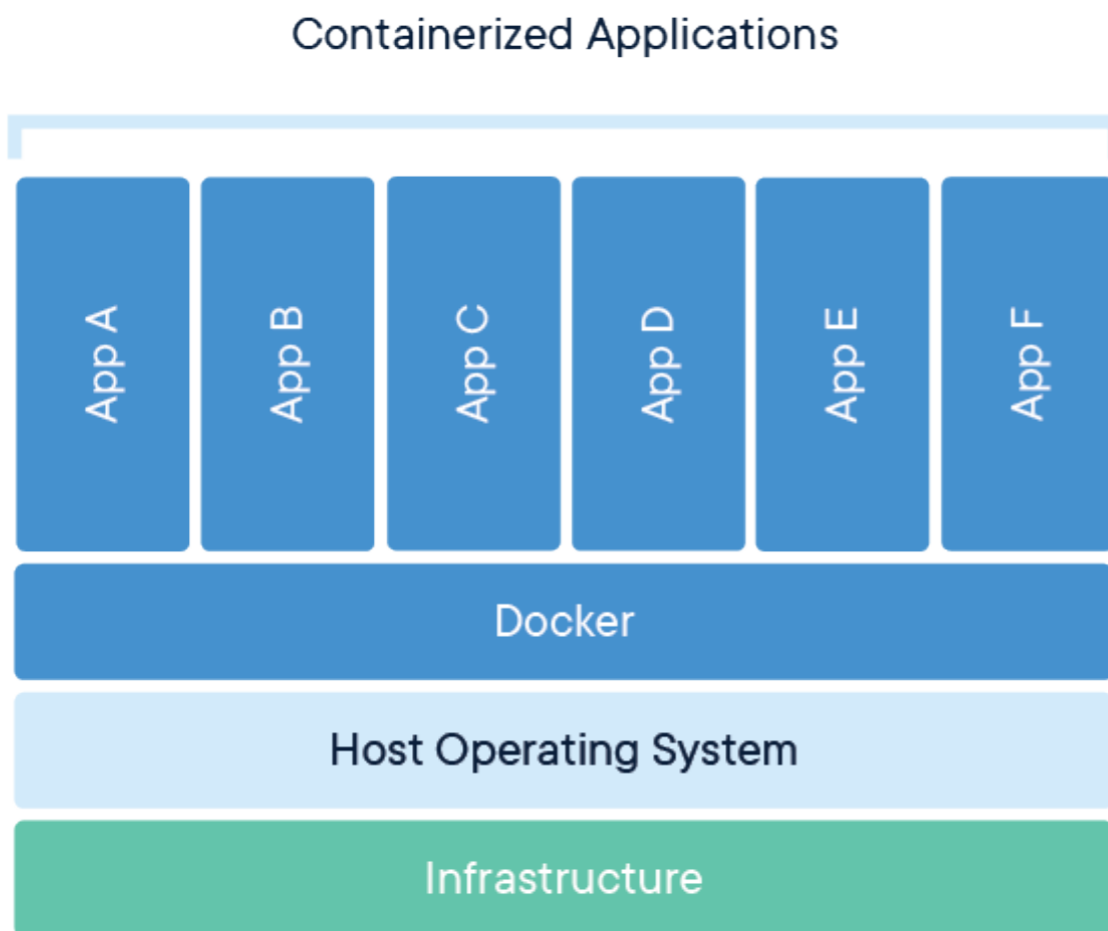


And ta-da! We did a jupyter.

To run:
Kernel:
Restart and Run All

Containers

- Can also make use of containers
 - Virtual “machine” but for applications



Containers

- Why bother?
 - So you don't have to spend time configuring software. It just works.
- Instructions:
 - Get docker : <https://docs.docker.com/install/>
 - (shortcuts: <https://download.docker.com>)
 - Follow installation instructions (OS dependent)
 - Once it is running, open your “Terminal” app in your laptop / host machine, and execute these:

```
mkdir results
git clone https://github.com/rappoccio/CompPhys.git
cd CompPhys
./runDocker.sh srappoccio/compphys:latest 1
```

Containers

- You will get a command line just like a linux computer (running Ubuntu 19)

```
% ./runDocker.sh srappoccio/compphys:latest 1  
192.168.1.231 being added to access control list  
compphys@55d39ad05e02:~$
```

- (Your IP and hash details will be different)

Containers

- There is also a jupyter mode you can run on your own machine:

```
./runDocker.sh srappoccio/compphys:latest 0
```

Last changed to “zero”

```
192.168.1.231 being added to access control list
[I 16:20:12.031 NotebookApp] Writing notebook server cookie secret to /results/.local/share/jupyter/runtime/notebook_cookie_secret
[I 16:20:12.446 NotebookApp] Loading IPython parallel extension
[I 16:20:12.447 NotebookApp] Serving notebooks from local directory: /results
[I 16:20:12.447 NotebookApp] The Jupyter Notebook is running at:
[I 16:20:12.448 NotebookApp] http://(6209865d7dfc or 127.0.0.1):8888/?
token=d1935c176ba256ed464e8f7bdb2e76fa516bb1044c8c7128
[I 16:20:12.448 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[W 16:20:12.466 NotebookApp] No web browser found: could not locate runnable browser.
[C 16:20:12.466 NotebookApp]
```

To access the notebook, open this file in a browser:

```
file:///results/.local/share/jupyter/runtime/nbserver-1-open.html
```

Or copy and paste one of these URLs:

```
http://(6209865d7dfc or 127.0.0.1):8888/?token=d1935c176ba256ed464e8f7bdb2e76fa516bb1044c8c7128
```

This is your token

Containers

- Now you're running jupyter locally:



The screenshot displays the top portion of the JupyterLab web interface. At the top left is the Jupyter logo and the text "jupyter". To the right are "Quit" and "Logout" buttons. Below this is a navigation bar with three tabs: "Files" (selected), "Running", and "IPython Clusters". Underneath the tabs, the text "Select items to perform actions on them." is visible. To the right of this text are "Upload", "New" (with a dropdown arrow), and a refresh icon. At the bottom of the interface, there is a file manager bar showing a selection of 0 items, a folder icon, and a slash. To the right of the file manager are three columns: "Name" (with a dropdown arrow), "Last Modified", and "File size".

UNIX/LINUX

- Operating systems (like Windows or Mac OS)
- We will be using LINUX (an offshoot of UNIX)
 - Invented by Linus Torvalds in 1991
 - Mac OS X is built upon LINUX
- EdX course from the creator here :
 - <https://www.edx.org/course/introduction-linux-linuxfoundationx-lfs101x-0#!>
- Linux is completely open source : you can modify it at your will and debuggers are also free
- Predominantly command line tools
- Cheat sheet on UBLearns (or indico for FNAL people)

Linux Shells

- We will be using an interface to Linux called a “shell”
- It is a command-line interpreter : you type, it executes
- Two major options are “bash” (as in, smash) and “csh” (like “sea shell”, modern version is “tcsh”, “tea sea shell”)
- For the most part, few differences with respect to this class, you can use either one.
- Major difference : environment variables syntax
 - bash: `export X=value`
 - tcsh: `setenv X value`
- That’s about it for the purposes of this class

Linux Command Line

- Basics : Listing directory contents : “ls”, like “list”

```
$ ls  
$
```

- Empty right now

- Basics : Echoing content : “echo” :

```
$ echo "Why am I here?"  
Why am I here?
```

- Basics : Saving what you type: the “pipe” command : “>”

```
$ echo "Who am I?" > stuff.txt  
$ echo "Why am I here?" > stuffagain.txt
```

- No output... what happened? “ls” again:

```
$ ls  
stuff.txt stuffagain.txt
```

- Aha, there it is. What’s in it?

Linux Command Line

- Reading files : “more”, “less”, and “cat”:
 - “cat” : concatenates files together (if just one, reads it)

```
$ cat stuff.txt  
Who am I?
```

- “less” : reads a file and paginates it

```
$ less stuff.txt  
Who am I?  
~
```

- “more” : reads multiple files separately

```
$ more stuff.txt  
Who am I?
```


Linux Command Line

– “cat” will concatenate:

```
$ cat stuff.txt stuffagain.txt  
Who am I?  
Why am I here?
```

– But “more” does more:

```
$ more stuff.txt stuffagain.txt  
:::::::::::::  
stuff.txt  
:::::::::::::  
Who am I?  
:::::::::::::  
stuffagain.txt  
:::::::::::::  
Why am I here?  
$
```

Linux Command Line

- What if you want to copy or move files?
- Copy : “cp” :

```
$ cp stuff.txt stuff1.txt
```

```
$ ls
```

```
stuff1.txt  stuffagain.txt  stuff.txt
```

```
$ cat stuff1.txt
```

```
Who am I?
```

- File “stuff1.txt” is a copy of “stuff.txt”
- They are duplicates. “stuff.txt” is retained.

- Move : “mv” :

```
$ ls
```

```
stuff2.txt  stuffagain.txt  stuff.txt
```

```
$ cat stuff2.txt
```

```
Who am I?
```

- File “stuff2.txt” is there, but “stuff1.txt” is gone
- Also known as “renaming”!

Linux Command Line

- Removing files : “rm”

```
$ rm stuff2.txt
```

```
rm: remove regular file `stuff2.txt'? y
```

```
$ ls
```

```
stuffagain.txt  stuff.txt
```

- Gone forever. Really, really, really gone. No “trash bin”. No “restore”. No “but professor, can’t I undo that?” No. You cannot. Bits are gone.
- Backups in LINUX are VERY IMPORTANT TO HAVE

–Example:

```
$ cp stuff.txt stuff_backup_27jan2017.txt
```

–OR! Use version control (more on that later)

Linux Command Line

- What if I hate typing files one at a time?
 - Then Linux is the best operating system for you
- Huge number of shortcuts in LINUX :
 - tab completion : If you start typing and press “TAB”, it will list the possible completions, or if there is only one, complete it for you
 - “up” key : reproduces previous line in Terminal
 - aliases : can define your own shortcuts
 - wildcards!

Hands on!

- Open terminal, go to “CompPhys/LinuxOverview”
- Look at “commandline1.sh” file:

```
% more commandline1.sh
echo "Always look on the bright side of life" > bright.txt
echo "If life is jolly rotten, there's something you've
forgotten" > forgotten.txt
cat bright.txt forgotten.txt > song.txt
cat song.txt=
```

- Execute:

```
% bash commandline1.sh
```

- Output:

```
Always look on the bright side of life
If life is jolly rotten, there's something you've forgotten
```

- Type “ls”:

```
bright.txt commandline1.sh forgotten.txt song.txt
```

- Demonstrates: more, ls, and cat commands, files, and piping

Linux Command Line

- **Wildcards** : <http://www.tldp.org/LDP/GNU-Linux-Tools-Summary/html/x11655.htm>

– **?** : single character

```
$ ls ?right.txt  
bright.txt
```

– ***** : any number of characters

```
$ ls *.txt  
bright.txt  forgotten.txt  song.txt  stuffagain.txt  
stuff_backup_27jan2017.txt  stuff.txt
```

```
$ ls s*.txt  
song.txt  stuffagain.txt  stuff_backup_27jan2017.txt  stuff.txt
```

Linux Command Line

- Wildcards:

- [] : specifies a range

```
$ echo "song1" > song1.txt
$ echo "song2" > song2.txt
$ echo "song3" > song3.txt
$ cat song[1-2].txt
song1
song2
```

- {} : name or wildcard

```
$ echo "song4" > whoops.ugh
$ cat {song*.txt,whoops.ugh}
song1
song2
song3
```

```
Always look on the bright side of life
```

```
If life is jolly rotten, there's something you've forgotten
```

```
song4
```


Linux Command Line

- Command line options :
 - What if you like “rm”, but hate to confirm each removal?
 - “rm” has OPTIONS you can add
- Options usually come in two forms :
 - Single dash, single letter : “ls -r”
 - Two dashes, many letters: “ls —help”
 - Most commands come with “—help”, so you can read the options you have

Linux Command Line

- Example : “ls”:

```
$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort.

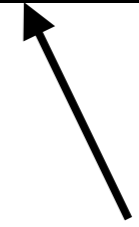
Mandatory arguments to long options are mandatory for short options too.
-a, --all          do not ignore entries starting with .
-A, --almost-all do not list implied . and ..
--author          with -l, print the author of each file
-b, --escape      print octal escapes for nongraphic characters
--block-size=SIZE use SIZE-byte blocks. See SIZE format below
```

- Common options
 - “-l” : long listing
 - “-a” : list all
 - “-t” : list by time
 - “-r” reverse

Linux Security

- What the heck is all this stuff?

```
-rw----- 1 srrappoc phyfac 6 Jan 27 11:14 whoops.ugh
```



Filename

Linux Security

- What the heck is all this stuff?

```
-rw----- 1 srrappoc phyfac 6 Jan 27 11:14 whoops.ugh
```

Time/date
of last mod.



Filename



Linux Security

- What the heck is all this stuff?

```
-rw----- 1 srrappoc phyfac 6 Jan 27 11:14 whoops.ugh
```

↑
“group”

↑
Time/date
of last mod.

↑
Filename

Linux Security

- What the heck is all this stuff?

```
-rw----- 1 srrappoc phyfac 6 Jan 27 11:14 whoops.ugh
```

↑ "permissions" ↑ "owner" ↑ "group" ↙ Time/date of last mod. ↘ Filename

Linux Security

- Levels of security :
 - All : everyone
 - Group : a group of users
 - Owner : the owner of the file
- Access levels :
 - Read : Can read from file
 - Write : Can write to file
 - Execute : Can execute if it is a program
- Can set different access levels for different levels
 - Most common example: “Owner” can read+write, “Group” can read, “All” can do nothing

Linux Security

- You can change the permissions, the owners, or the groups of files:
 - chown (change owner or group) :
 - usage : chown user:group file
 - change owner of file to “user” and group to “group”
 - chmod (change permission)
 - usage examples :
 - chmod a+x file (give execute permission to “all”)
 - chmod g+w file (give write permission to “group”)
 - chmod a+wrwx file (give read/write/execute permission to anyone)

Linux Directories

- To make directories, use “mkdir”:

```
$ mkdir testdir  
$ ls  
bright.txt  cmdline1.sh  forgotten.txt  song.txt  testdir
```

–To remove them, “rmdir” (must be empty)

- You can move files from one directory to another with “mv” just like before:

```
$ mv song.txt testdir/
```

- And you can use “ls” to look at the directory:

```
$ ls testdir/  
song.txt
```

Linux Directories

- The directory you are “in” is called the “current directory”
- To change your current directory, use “cd”:

```
$ cd testdir  
/nsm/home/srrappoc/testdir  
$ ls  
song.txt
```

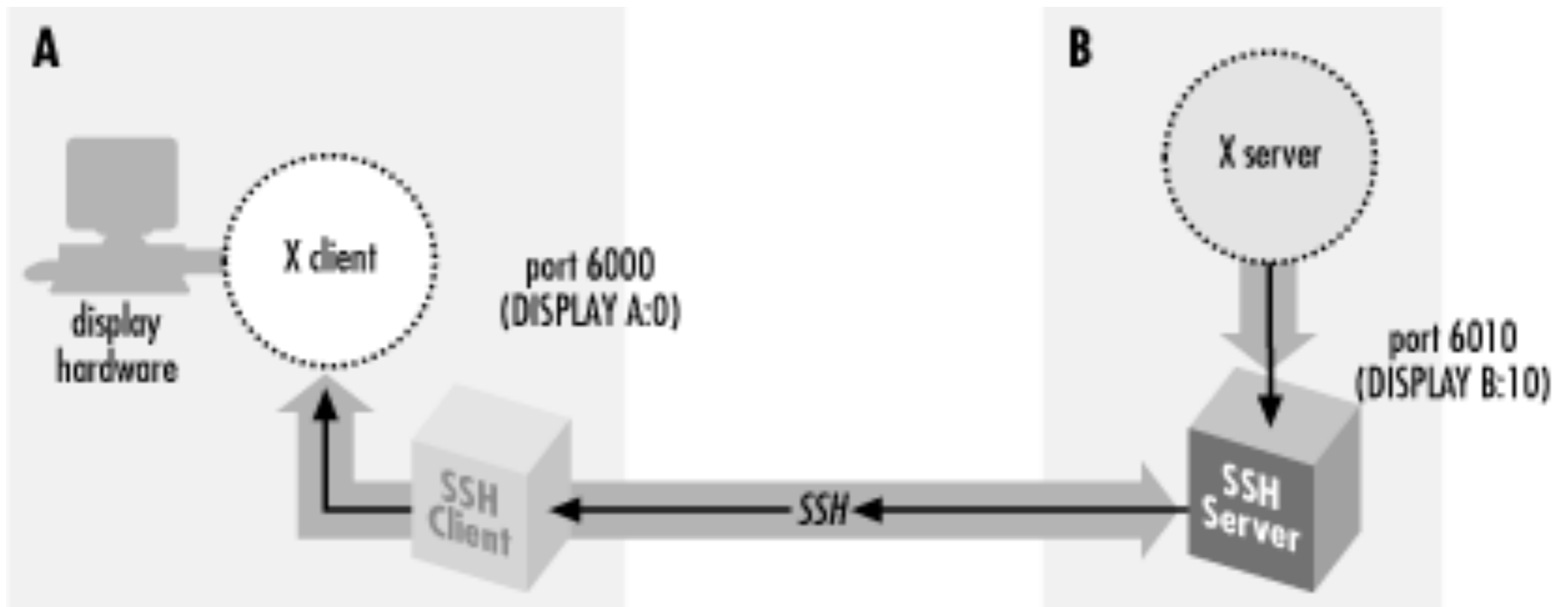
- To go to the directory “up” the chain, do “cd ..”:

```
$ cd ..
```

- There are shortcuts for “cd” :
 - “cd -” go to previous directory
 - “cd” (no arguments) go to home directory
- You can also print the working directory (pwd)

Editors

- AKA : why is it so f*ing hard to type stuff in Linux?
 - It isn't, you're just not thinking about it right :)
 - If you're sitting at the same terminal you're executing commands on, your server is the same as your client
 - If you're using ssh, you are trying to send a ton of graphical information over the net to type



Editors

- Linux comes with the “X11” graphics package : that’s how everything works
 - This is also the basis for Mac OS’s GUI
- Many editors are available, and many have “inline text” and “X11” modes
- Now we come to one of the greatest scientific controversies of all time:

THE EDITOR WARS

Editors

VIM

usable in just about any environment.

does one thing, well.



EMACS

flexible, customizable, and packed with every feature known to man.



NANO

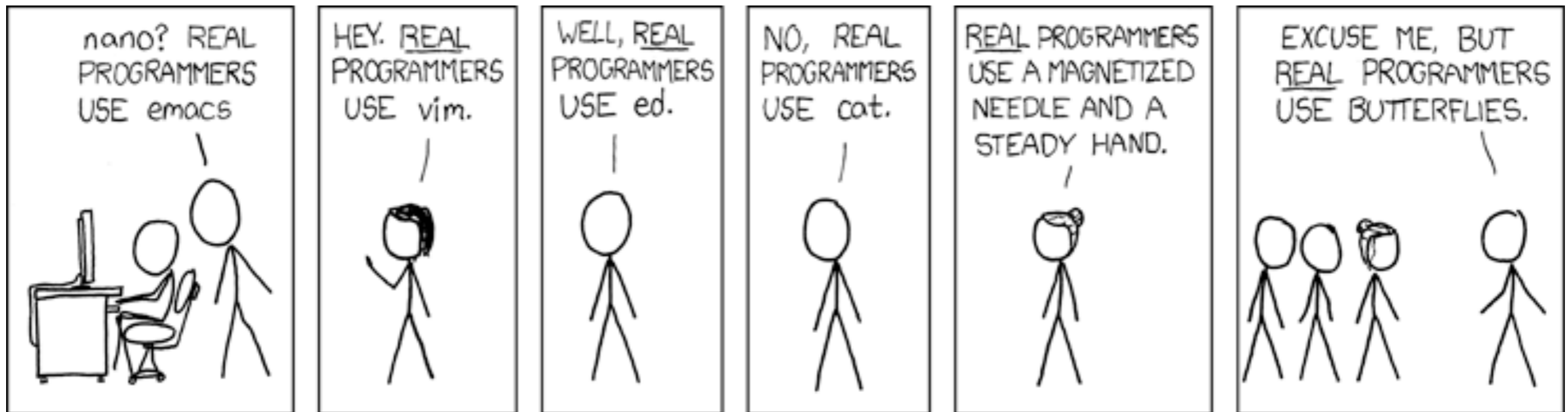
mostly used by people who do not know what they are doing; or psychopaths.



© 2015 CURTIS LASSAM - CUBE-DRONE.COM

<http://cube-drone.com/comics/c/holy-war>

Editors

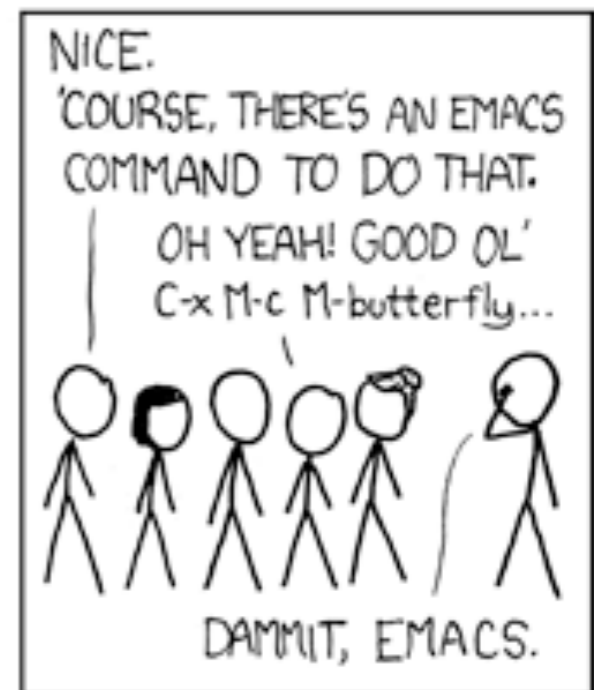
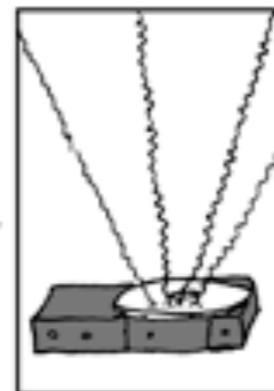
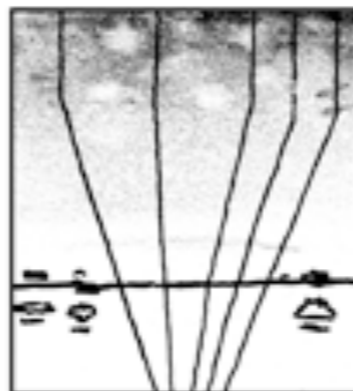


THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.



THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,

WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.



Editors

- I am personally firmly in the “Emacs” zone of the world
- If you are a vi or ed user, you can leave now
 - Kidding!!! *coughsortofcough*
- I can teach you the basics of emacs, and you should learn the rest on your own. You’re scientists, that’s how it goes. It’s like learning to walk, or ride a bike, or hazing or something.

Editors

- emacs :
 - X mode : emacs filename.txt &
 - Edit away as you want.
 - Terminal mode : emacs -nw filename.txt
 - “-nw” is “no window”
 - Tutorial : <https://www.gnu.org/software/emacs/tour/>
- vi / vim:
 - vi filename.txt
 - I suck at vi, so just go here:
 - <https://scotch.io/tutorials/getting-started-with-vim-an-interactive-guide>

Editors

- If you're terrified of learning how to use these, just use "nano". The interface is straightforward, but has extremely few features.
- Some of you young whipper snappers will want to use a more modern editor like sublime or xcode
 - If you know out how to use them, go for it, but I won't help you! :)

Editing: VIDIA

- Emacs and vi are installed in vidia
- You can also just use jupyter directly and type there
- Example:

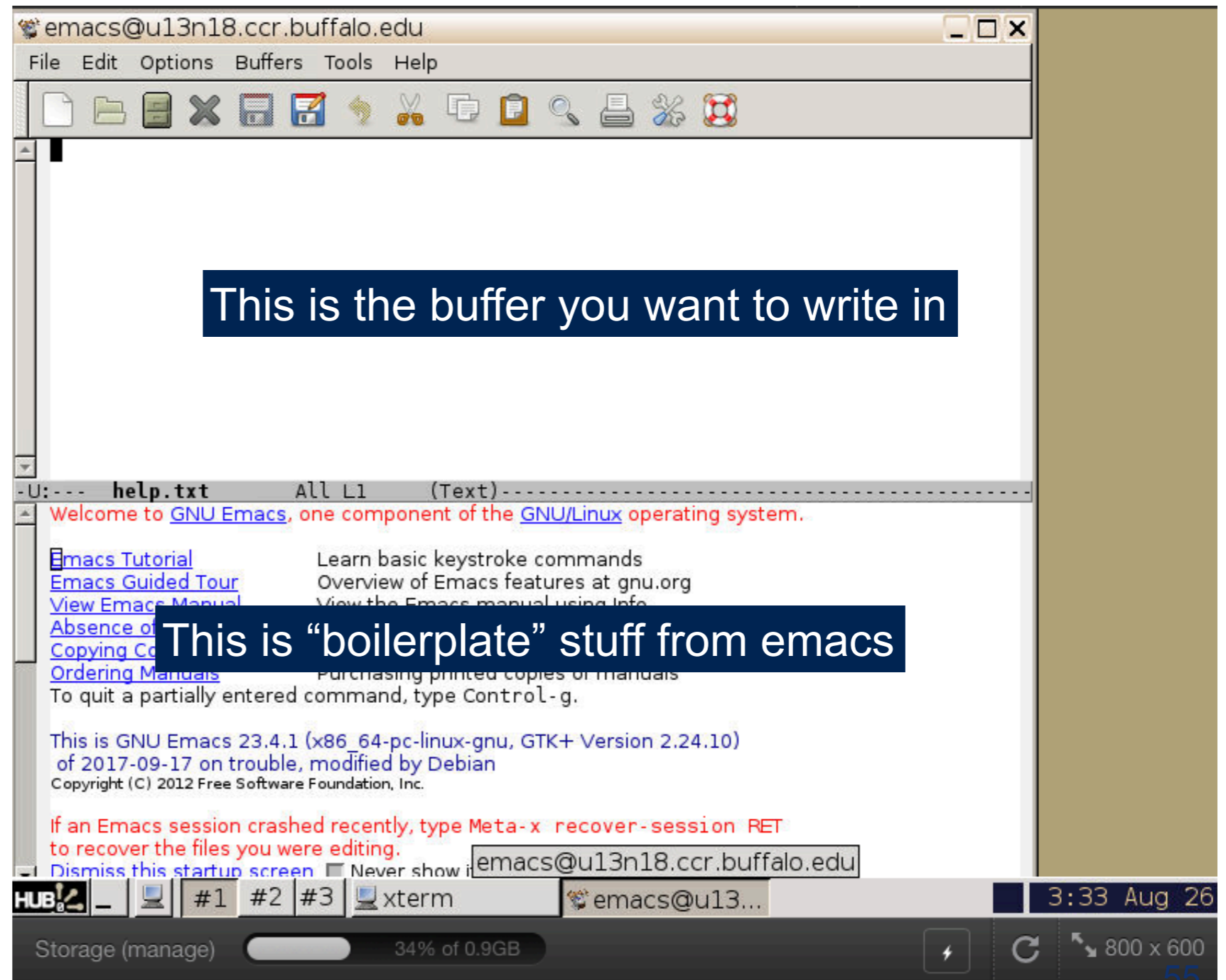
```
uxterm
srrappoc@vidia:~$ emacs help.txt &
```

File name is "help.txt"
"&" means operate in the background
(so you can type in terminal again!)

HUB #1 #2 #3 uxterm 3:32 Aug 26
Storage (manage) 34% of 0.9GB 800 x 600

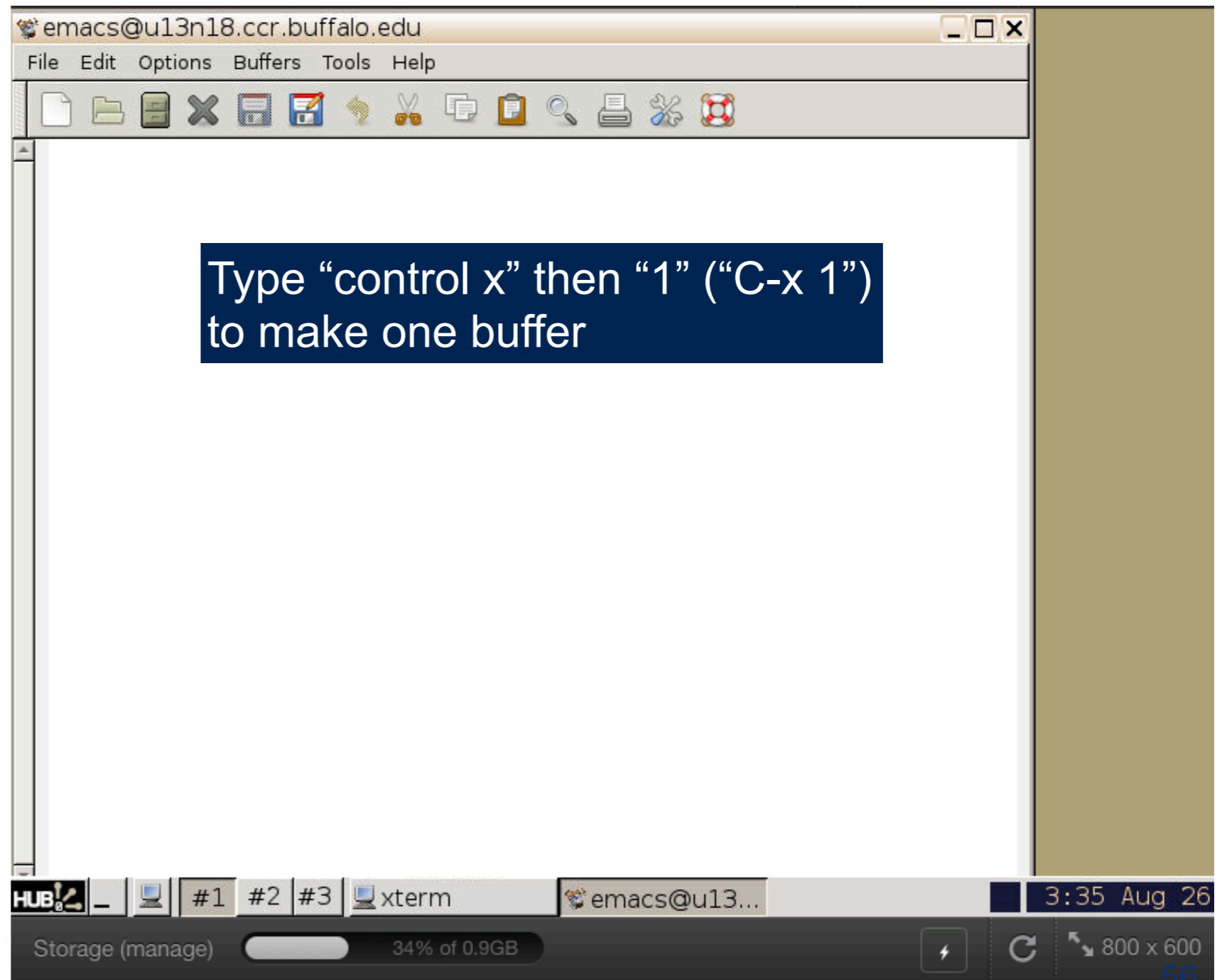
Editing: VIDIA

- Emacs and vi are installed in vidia
- You can also just use jupyter directly and type there
- Example:



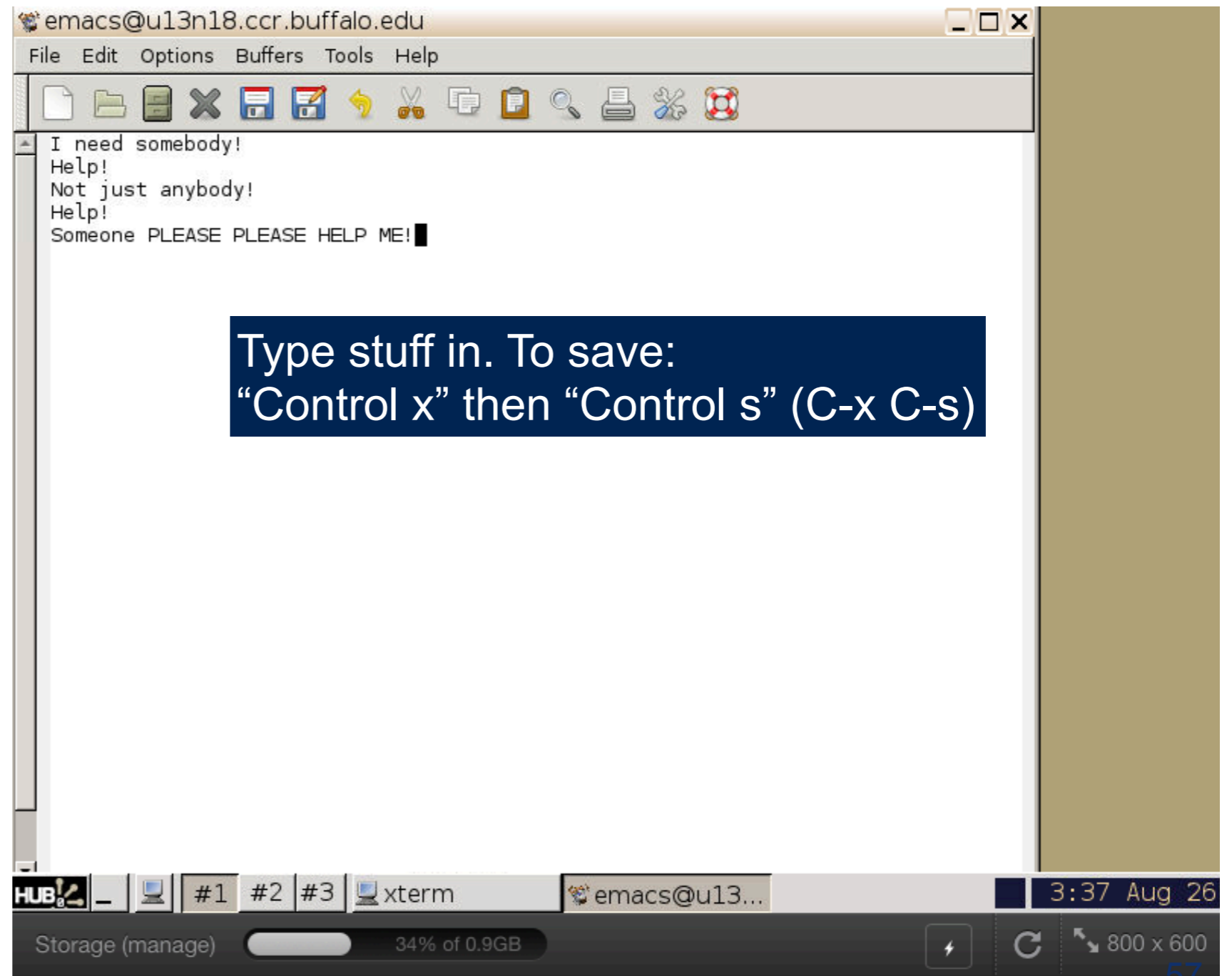
Editing: VIDIA

- Emacs and vi are installed in vidia
- You can also just use jupyter directly and type there
- Example:



Editing: VIDIA

- Emacs and vi are installed in vidia
- You can also just use jupyter directly and type there
- Example:

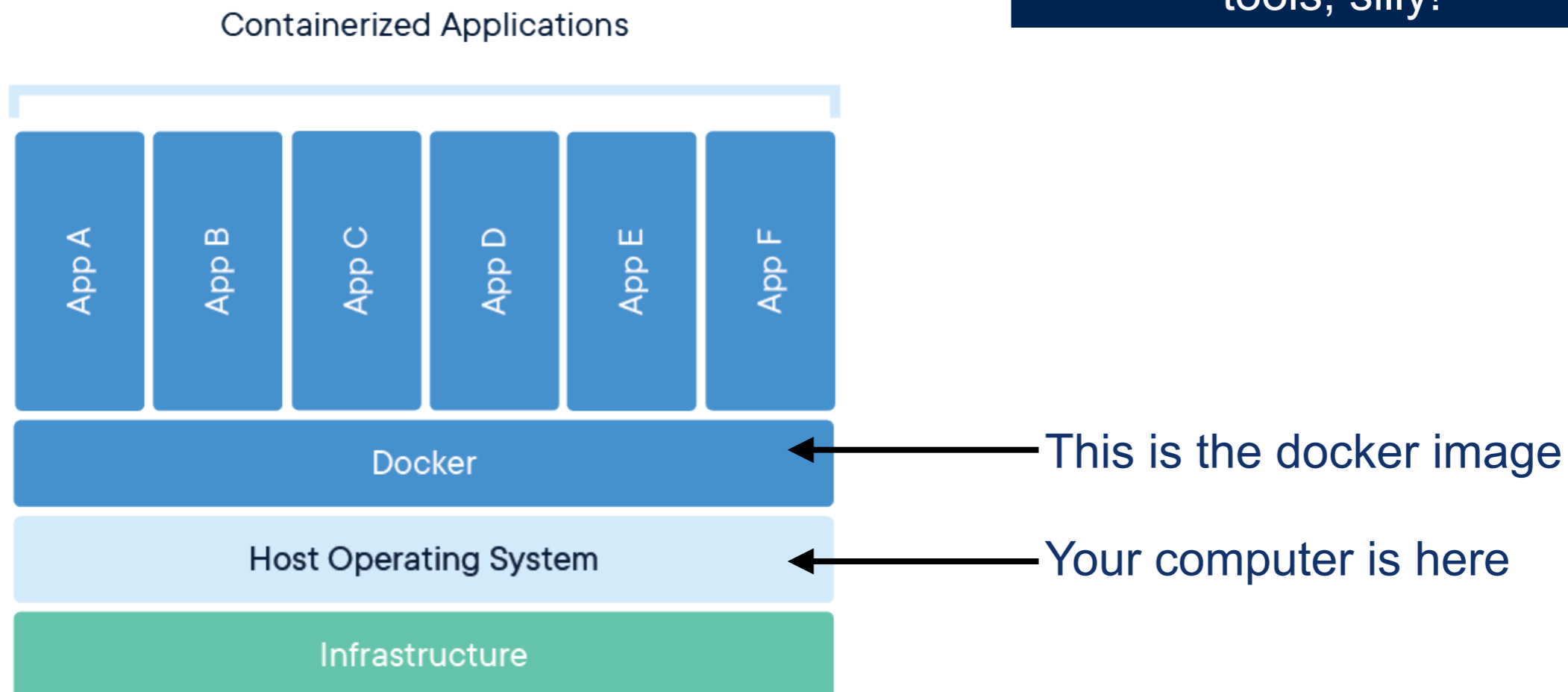


Editing: Docker

- There is no emacs in our docker image
- There is no vi in our docker image
- There is no editor at all

- WTH Sal?!?!?

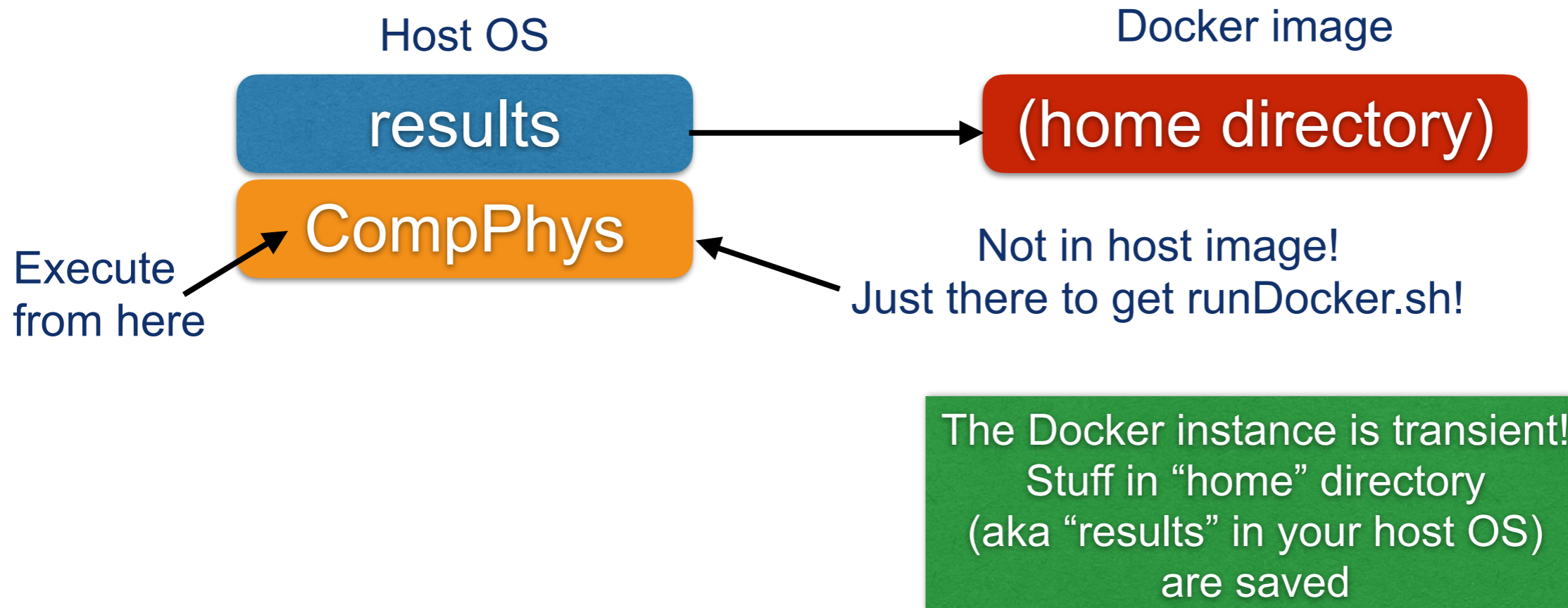
Use the host operating system's editing tools, silly!



Editing: Docker

- Remember the instructions? That little first line to make a “results” directory?

```
mkdir results
git clone https://github.com/rappoccio/CompPhys.git
cd CompPhys
./runDocker.sh srappoccio/compphys:latest 1
```



Docker setup

```
[galadriel:% mkdir results
[galadriel:% git clone https://github.com/rappoccio/CompPhys.git
Cloning into 'CompPhys'...
remote: Enumerating objects: 196, done.
remote: Counting objects: 100% (196/196), done.
remote: Compressing objects: 100% (146/146), done.
remote: Total 196 (delta 66), reused 172 (delta 43), pack-reused 0
Receiving objects: 100% (196/196), 535.99 KiB | 9.75 MiB/s, done.
Resolving deltas: 100% (66/66), done.
[galadriel:% ls
CompPhys          results
[galadriel:% ls *
CompPhys:
AdvCpp            JupyterExamples  ReviewPython      runDocker.sh
ArrayProgramming LinuxOverview    SwigExamples
DataAnalysis     README.md        Vectorization
Dockerfile       ReviewCpp        install_software.sh

results:
galadriel:% █
```

Docker setup

You type:

```
[galadriel:% mkdir results
[galadriel:% git clone https://github.com/rappoccio/CompPhys.git
Cloning into 'CompPhys'...
remote: Enumerating objects: 196, done.
remote: Counting objects: 100% (196/196), done.
remote: Compressing objects: 100% (146/146), done.
remote: Total 196 (delta 66), reused 172 (delta 43), pack-reused 0
Receiving objects: 100% (196/196), 535.99 KiB | 9.75 MiB/s, done.
Resolving deltas: 100% (66/66), done.
[galadriel:% ls
CompPhys          results
[galadriel:% ls *
CompPhys:
AdvCpp            JupyterExamples  ReviewPython      runDocker.sh
ArrayProgramming LinuxOverview    SwigExamples
DataAnalysis     README.md        Vectorization
Dockerfile       ReviewCpp        install_software.sh

results:
galadriel:%
```

Docker setup

Software downloads:

```
[galadriel:~]$ git clone https://github.com/rappoccio/CompPhys.git
Cloning into 'CompPhys'...
remote: Enumerating objects: 196, done.
remote: Counting objects: 100% (196/196), done.
remote: Compressing objects: 100% (146/146), done.
remote: Total 196 (delta 66), reused 172 (delta 43), pack-reused 0
Receiving objects: 100% (196/196), 535.99 KiB | 9.75 MiB/s, done.
Resolving deltas: 100% (66/66), done.
[galadriel:~]$ ls
CompPhys      results
[galadriel:~]$ ls *
CompPhys:
AdvCpp      JupyterExamples  ReviewPython      runDocker.sh
ArrayProgramming  LinuxOverview    SwigExamples
DataAnalysis  README.md        Vectorization
Dockerfile   ReviewCpp        install_software.sh

results:
galadriel:~$
```

Docker setup

```
[galadriel:% mkdir results
[galadriel:% git clone https://github.com/rappoccio/CompPhys.git
Cloning into 'CompPhys'...
remote: Enumerating objects: 196, done.
remote: Counting objects: 100% (196/196), done.
remote: Compressing objects: 100% (146/146), done.
remote: Total 196 (delta 66), reused 172 (delta 43), pack-reused 0
, 535.99 KiB | 9.75 MiB/s, done.
```

This is what is in the directories:

```
galadriel:% ls
CompPhys      results
[galadriel:% ls *
CompPhys:
AdvCpp        JupyterExamples  ReviewPython      runDocker.sh
ArrayProgramming  LinuxOverview    SwigExamples
DataAnalysis    README.md        Vectorization
Dockerfile      ReviewCpp        install_software.sh

results:
galadriel:%
```

Docker setup

- Execute the docker command, and the home directory has nothing in it
 - It's the “results” directory so that makes sense:

```
[galadriel:% ./runDocker.sh srappoccio/compphys:latest 1  
10.84.26.222 being added to access control list  
[compphys@0a8875f9d03e:~$ ls  
compphys@0a8875f9d03e:~$
```


Docker setup

- Now if I want to put something there I can:

```
[compphys@0a8875f9d03e:~$ echo "I get by with a little help from my friends" > CockersVersionisBetterSNS.txt
[compphys@0a8875f9d03e:~$ ls
CockersVersionisBetterSNS.txt
[compphys@0a8875f9d03e:~$ more CockersVersionisBetterSNS.txt
I get by with a little help from my friends
compphys@0a8875f9d03e:~$
```

- It is in the HOME directory on the docker image.
- AND it is in the “results” directory on your host OS:
 - It is NOT in the directory you executed from

```
[compphys@0a8875f9d03e:~$ exit
exit
[lsgaladriel:% ls
AdvCpp          Dockerfile      README.md      SwigExamples   runDocker.sh
ArrayProgramming JupyterExamples ReviewCpp      Vectorization
DataAnalysis    LinuxOverview   ReviewPython   install_software.sh
[galadriel:% ls ../results/
CockersVersionisBetterSNS.txt
[galadriel:% more ../results/CockersVersionisBetterSNS.txt
I get by with a little help from my friends
galadriel:%
```

It is NOT here.

It is in “../results”

Editing in Docker

- Now, if you edit the files in the “results” folder they appear in the “home” folder of the docker image:

```
galadriel:% emacs ../results/CockersVersionisBetterSNS.txt
```

```
What would you do if I sang out of tune?  
I get by with a little help from my friends
```

```
-uu-:---F1 CockersVersionisBetterSNS.txt All L1 (Text)---
```


Editing in Docker

- Now, if you edit the files in the “results” folder they appear in the “home” folder of the docker image:

```
galadriel:% ./runDocker.sh srappoccio/compphys:latest 1  
10.84.26.222 being added to access control list  
compphys@2bc56cef647a:~$ ls  
CockersVersionisBetterSNS.txt  CockersVersionisBetterSNS.txt~  
compphys@2bc56cef647a:~$ more CockersVersionisBetterSNS.txt  
What would you do if I sang out of tune?  
I get by with a little help from my friends  
compphys@2bc56cef647a:~$
```

ALL NEW DOCKER IMAGE!

“home” folder (“results” from host)
now has the updates!

Editing in Docker

- So you just edit in whatever form you want in the “results” folder and they will appear in the docker image in the “home” directory
- You will therefore put your assignments, etc, in the “results” folder and execute them from the “CompPhys” folder!