Computational Astrophysics -- Overview

Lecture 1, Computational Astrophysics (ASTR660)

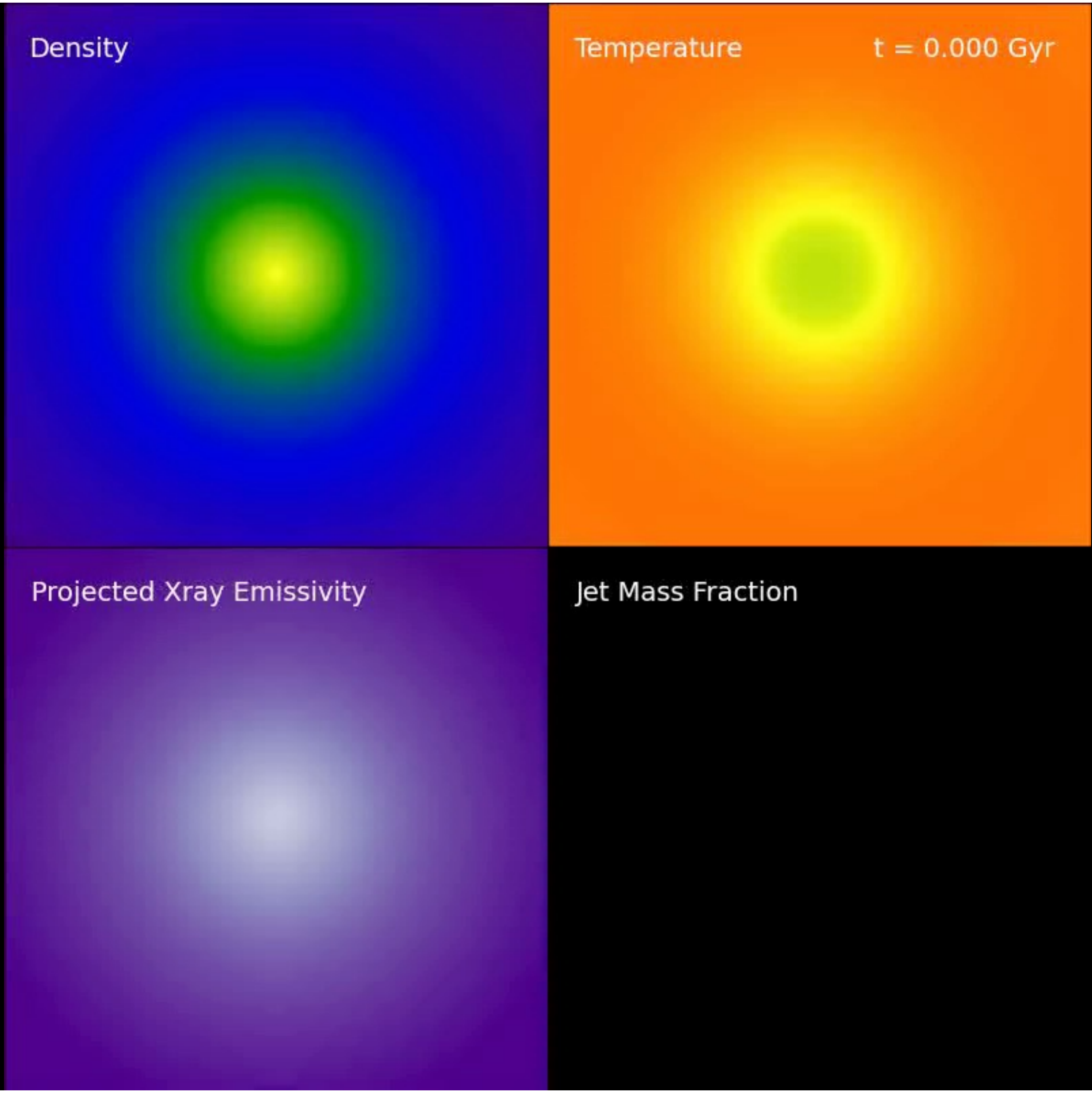
Hsiang-Yi Karen Yang, NTHU, 9/15/2022

This lecture…

- Course overview
- Introduction to computational astrophysics
- Intro & setup of computing tools & environments

About me

- Assistant professor at NTHU IoA (2020-present)
- Research: energetic feedback from supermassive black holes and their influence on the formation and evolution of galaxies and galaxy clusters
- Tools: numerical simulations with various input physics



- 3D hydrodynamic simulation
- FLASH code with adaptive mesh refinement (AMR)
- Intracluster medium in HSE within the dark matter halo
- Radiative cooling
- Subgrid model for AGN accretion and feedback
- (optional) star formation & stellar feedback
- (optional) conduction & viscosity
- (optional) cosmic rays

Yang & Reynolds (2016b)

Please briefly introduce yourself

- Name
- Department and year
- What research project are you currently working on / do you plan to work on?
- Why would you like to take this course / what would you like to learn from this course?

About Computational Astrophysics (ASTR660)

- This is a ***graduate-level*** course at the Institute of Astronomy (IoA) at NTHU
- For students who are doing / will pursue ***astrophysical research***
- For more programming-oriented courses, there are other undergrad-level courses offered in the Physics department:
  - “Computation for Physics (PHYS290000)” by Prof. Ing-Guey Jiang, teaching basic programming in C
  - ”Numerical Analysis (PHYS317000)” by Prof. Ing-Guey Jiang, focused on Python/C/Matlab programming and numerical methods
  - “Computational Physics Lab (PHYS401300)” by Prof. Kuo-Chuan Pan, focused on Python programming, numerical methods, and application to multiple fields in physics
  - “Computational Physics (PHYS401200) by Prof. Po-Chung Chen, focused on numerical methods for statistical and quantum physics

Goals of this course

1) Understand important concepts behind computational physics/astrophysics that are fundamental for both theorists and observers
2) Learn and practice commonly used numerical methods & apply to real astrophysical problems
3) Gain hands-on experience on high-performance computing (HPC) platforms
4) Train essential soft skills including asking questions, collaboration & brainstorming, research, integrating knowledge, critical thinking, oral presentations, English abilities, etc.

Syllabus

- Please go to the ***eLearn*** website for important info/announcements/resources about this course:

  https://elearn.nthu.edu.tw

- The class info page on eLearn includes class schedule and info about grading, so please read it carefully!

- What will be posted on eLearn: lecture slides, homework, solutions, useful resources, other announcements
  - There's also a forum for you to share resources with your peers

Grading & Scores

- Homework (40%)
- Term project (60%)
- Class participation bonus points

Homework assignments (40%)

- Assigned about every 2 weeks starting Week 2 (see schedule for exact dates)
- 6 assignments in total
- *Due 1 week after assignment at 14:20 on Thursdays*
- On-time submission will receive full credits, late submission within one week will receive *75%* credits, afterwards no credits will be given
- *Attach codes at the end in a single PDF file*
- Use latex to write your solutions (encouraged but not required)
- All assignments shall be submitted *online* through eLearn

No plagiarism (禁止抄襲)

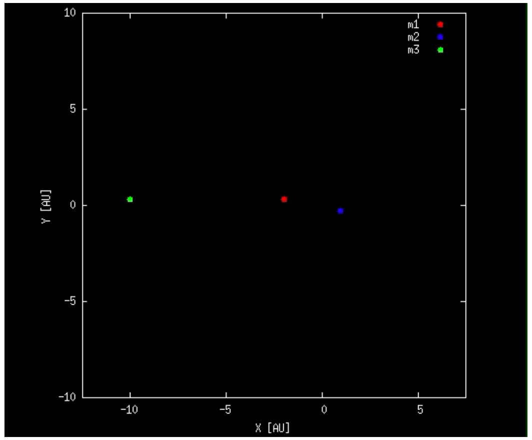
Term project (60%)

- Each student will apply the knowledge or numerical techniques learned in this class to their research projects or a topic of their interest.
- The project could be related to one of the following:
  - Reproduce scientific numerical results in published journals
  - Design a new numerical technique (tool/library/application)
  - Tackle an astrophysical problem that involve numerical techniques covered in class

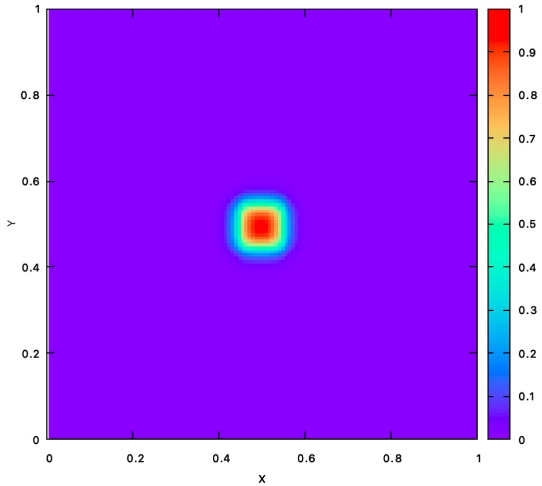
Term project examples

Angry bird game

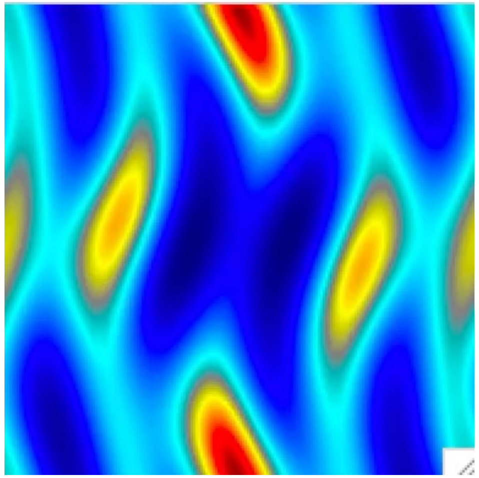
N-body simulations



2D advection



Magnetohydrodynamics (MHD)



You're welcome to be creative!!

Term project (60%)

The grades will include:

- *Midterm oral presentation on the project proposal (20%, 10+2 mins/person):* scientific motivations, literature review, descriptions of methods, feasibility
- *Final oral presentation on the project results (30%, 12+3 mins/person):* brief review, efforts (accomplishments/difficulties/solutions), results & discussions, performance (precision, accuracy, speed, comparison with previous works)
- *Final product of the project (10%, due on 1/12/2023):* submission of code & a 2-page short summary of code and key results/milestones, evaluation based on completeness & complexity of the project

Class participation bonus points

- *Ask questions!*
  - During class or after class
  - 0.5 point per question
- *Completion of in-class exercises*
  - Please submit your answers/codes on the day of the class
  - 0.5 point per exercise
- Maximum 0.5 point per week
- Up to 6.5 points could be gained for the whole semester
- *Please submit on eLearn under the TA session; there is a form (回饋單) to fill out to get the bonus points for each week*

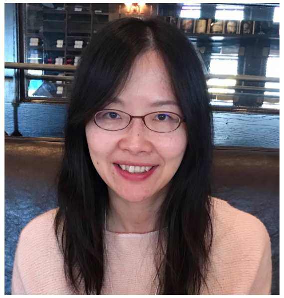
Tentative schedule for the semester

- Week 1 (9/15) Class overview / Introduction / Basic tools
- Week 2 (9/22) Computation basics I -- **HW1**
- Week 3 (9/29) Computation basics II
- Week 4 (10/6) Linear systems -- **HW2**
- Week 5 (10/13) Non-linear systems
- Week 6 (10/20) Initial value problems (celestial movement) -- **HW3**
- Week 7 (10/27) Boundary value problems (stellar structure)
- **Week 8 (11/3) Project proposal presentation I**
- **Week 9 (11/10) Project proposal presentation II**

Schedule will be announced. Please start early!

- Week 10 (11/17) PDE: hyperbolic systems (advection equation) -- **HW4**
- Week 11 (11/24) PDE: elliptical systems (gravity)
- Week 12 (12/1) PDE: astrophysics fluid dynamics -- **HW5**
- Week 13 (12/8) PDE: magnetohydrodynamics
- Week 14 (12/15) N-body simulations -- **HW6**
- Week 15 (12/22) Parallel programming with MPI and OpenMP
- **Week 16 (12/29) Final project presentations I**
- **Week 17 (1/5) Final project presentations II (submission of final product on 1/12/2023)**

Office hours



- Instructor: Hsiang-Yi Karen Yang
  - Assistant professor, Institute of Astronomy, NTHU
  - General Building II R504, hyang@phys.nthu.edu.tw
  - Office hour: Tuesdays 16:00-17:00

- TA #1: Yen-Hsing Lin / 林彥興
  - MS student, Institute of Astronomy, NTHU
  - julius52700@gapp.nthu.edu.tw
  - Office hour: Mondays 13:00-14:30 @ General Building II R529-17

- TA #2:
  - Pei-Ya Wang / 王沛雅
  - MS student, Institute of Astronomy, NTHU
  - peiya117@gmail.com
  - Office hour: Tuesdays 12:00-13:00 @ General Building II R529-16

References

1. "Numerical Recipe" by Press, W. H. (http://www.nr.com)

2. "Numerical Methods in Astrophysics" by Bodenheimer, P. et al.

3. "Scientific Computing: An Introductory Survey" by Michael Heath

4. "Introduction to Computational Astrophysical Hydrodynamics" by Zingale, M. (https://github.com/python-hydro/hydro_examples)

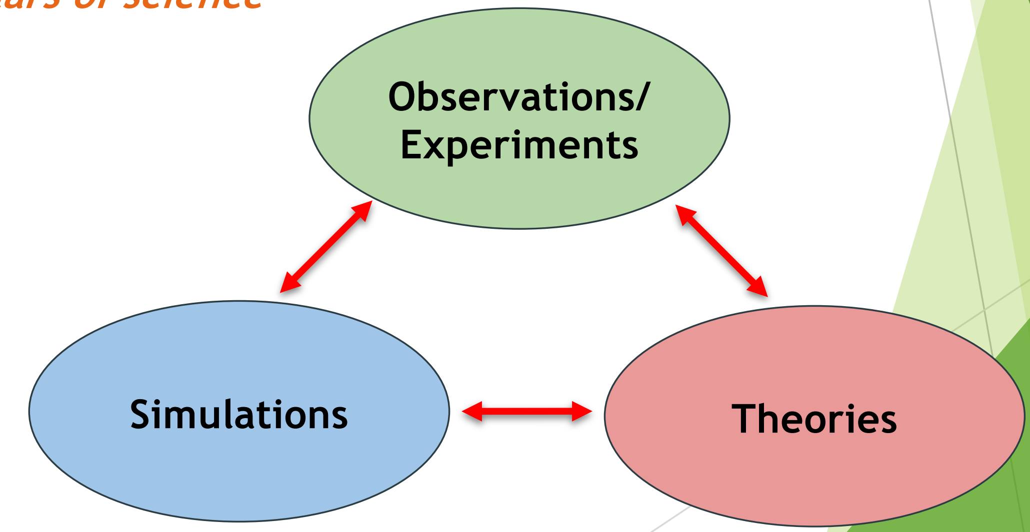
5. “A student’s Guide to Numerical Methods” by Ian H. Hutchinson (https://www.cambridge.org/core/books/students-guide-to-numerical-methods/06C4F9638E645EC28567DD4BDCEDD875)

6. “Computational Physics: Problem Solving with Computers, 2nd Edition” by R. Landau (https://onlinelibrary.wiley.com/doi/book/10.1002/9783527618835)

Intro to computational astrophysics

What is scientific computing?

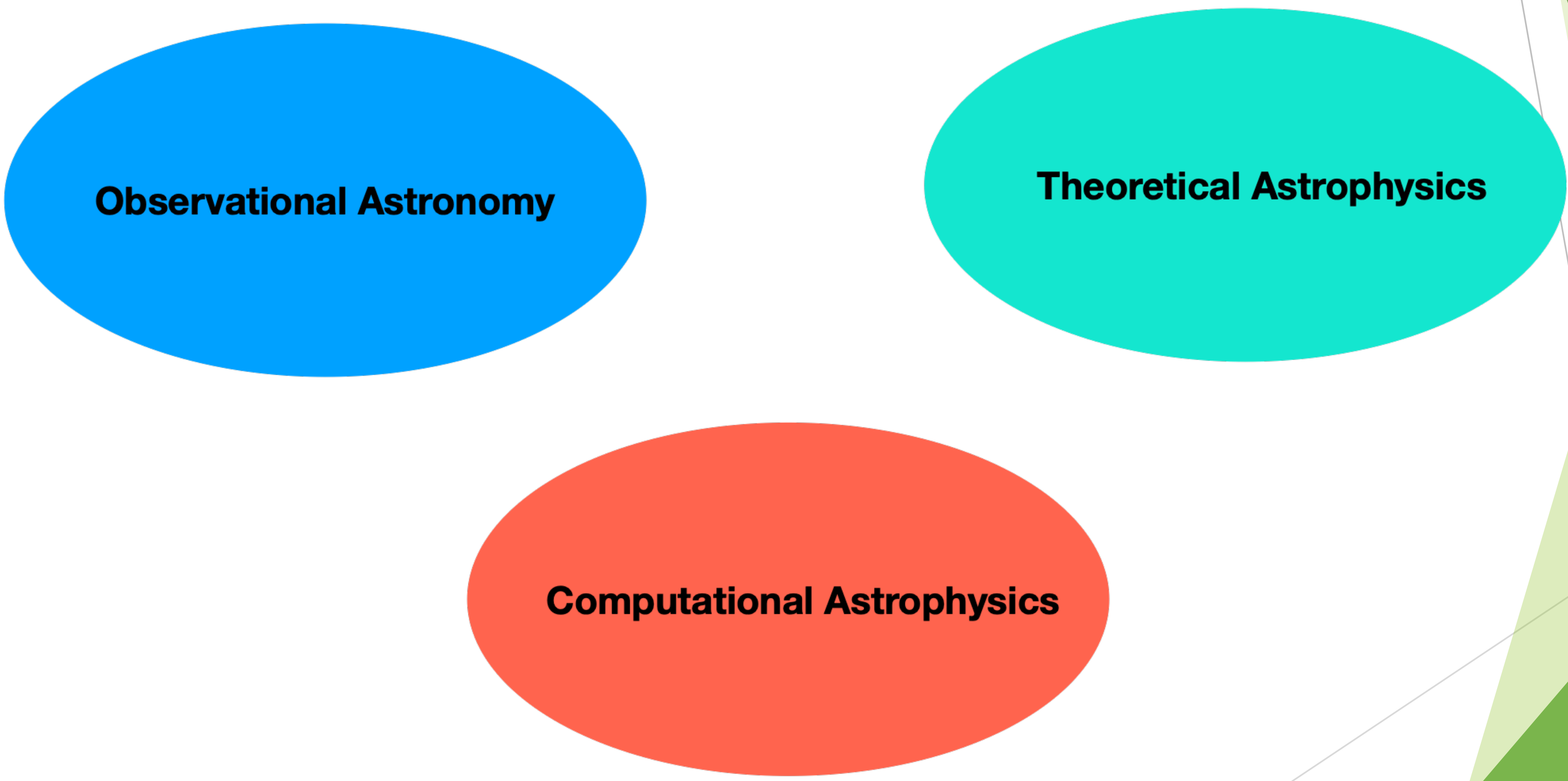
- It is the collection of tools, techniques, and theories required to numerically solve mathematical problems in science
- It is at the intersection among ***computer science*** (hard/software), ***applied mathematics***, and ***science*** (physics, materials, biology, atmospheres…)
- Nowadays it is one of the ***three pillars of science***



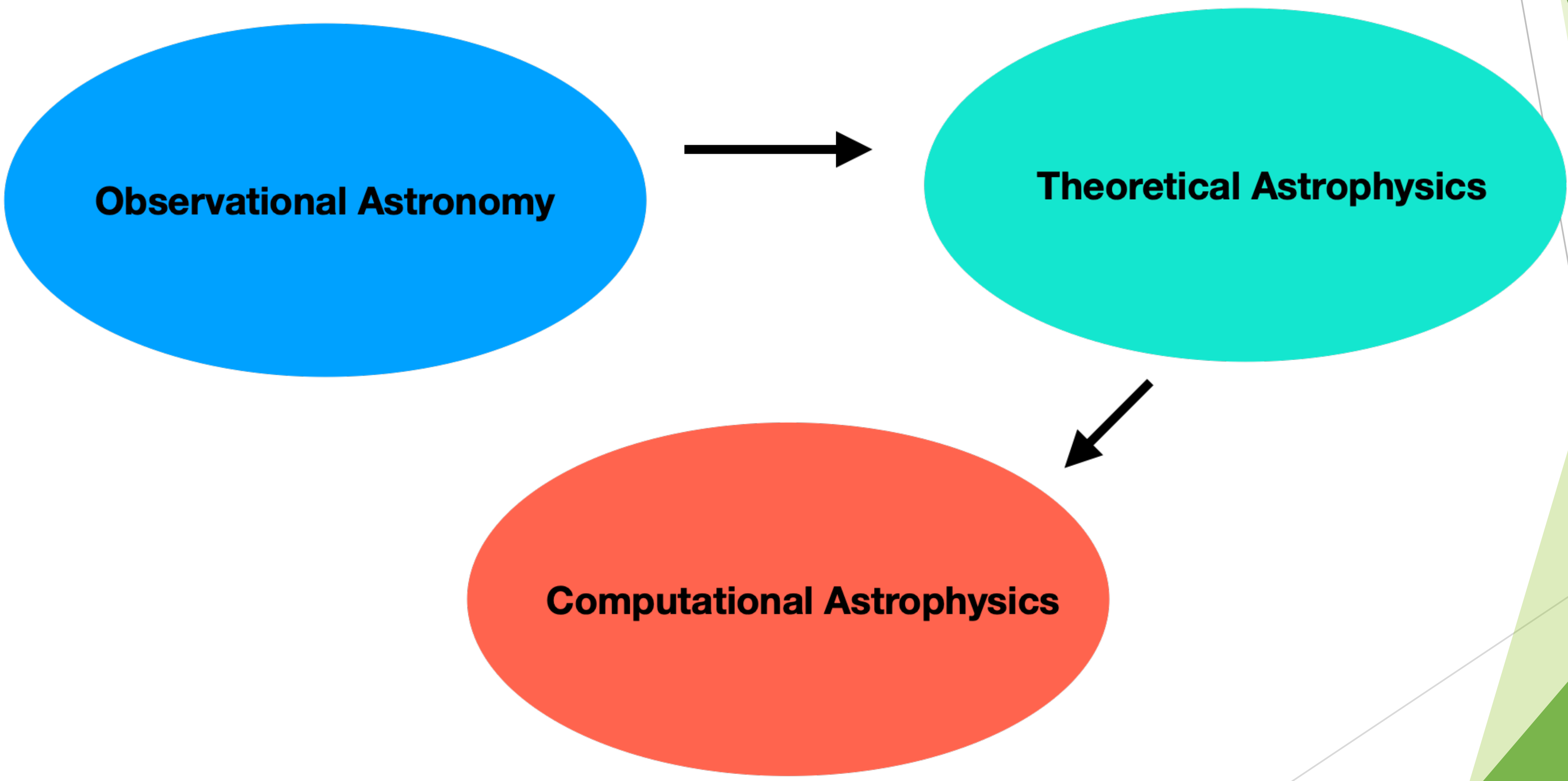
Why scientific computing?

- The problem at hand cannot be solved by traditional experimental or theoretical means
  - Highly nonlinear systems (e.g., climate predictions)
  - Experiments are too dangerous (e.g., characterization of toxic materials)
  - Experiments are too expensive (e.g., optimal design of aircrafts)
- Computing is especially important in astrophysics!
  - We can only observe the universe, not experiment or perturb it
  - Astrophysics often involves extreme conditions that cannot be replicated in labs (e.g., large sizes, low density, strong gravity, extreme magnetic fields…)
  - Timescales are typically >> human/PhD/MS timescales
  - Physics is complex and highly nonlinear (e.g., a combination of gravitational instabilities, fluid instabilities, thermal instabilities, shocks, turbulence...)

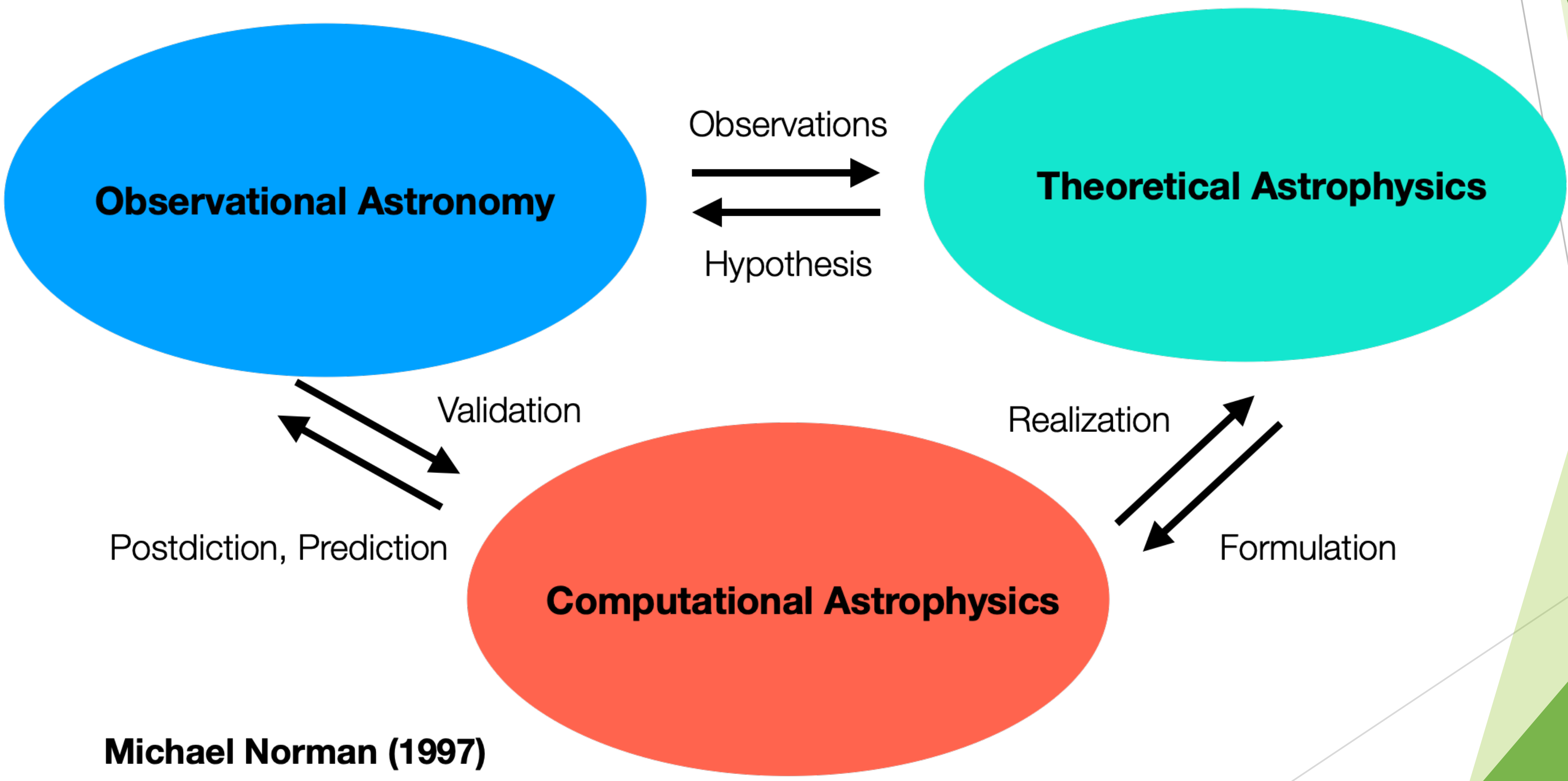
Fields of astronomical research



Development of astronomical research



Fields of astronomical research



Michael Norman (1997)

All fields in astronomy require programming at some level

Observational Astronomy

Data analysis
Big data
Machine learning
Statistics

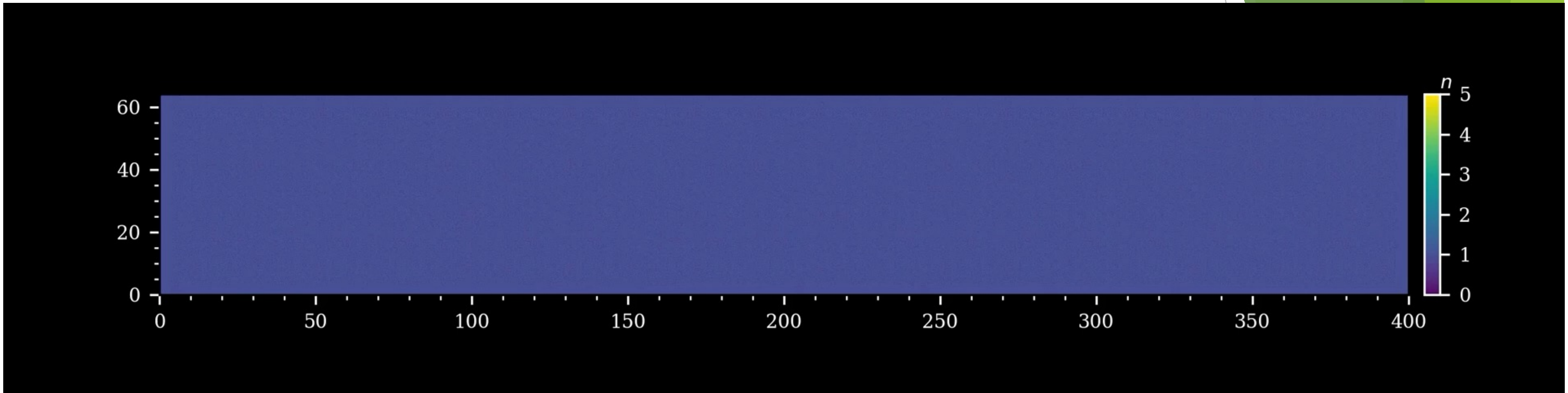
Theoretical Astrophysics

Semi-analytical approach
ODE/PDE solvers

Computational Astrophysics

Simulations
Data analysis
Visualization

Example: collisionless shocks



- *Particle-in-cell (PIC)* simulation of collisionless shocks in magnetized pair *plasma*
- Electrons and positrons simulated as charged particles
- Solving Maxwell equations on a grid
- Applications: fundamental understanding of plasma physics (particle acceleration), solar winds, provides insights of “microphysics” into larger-scale simulations

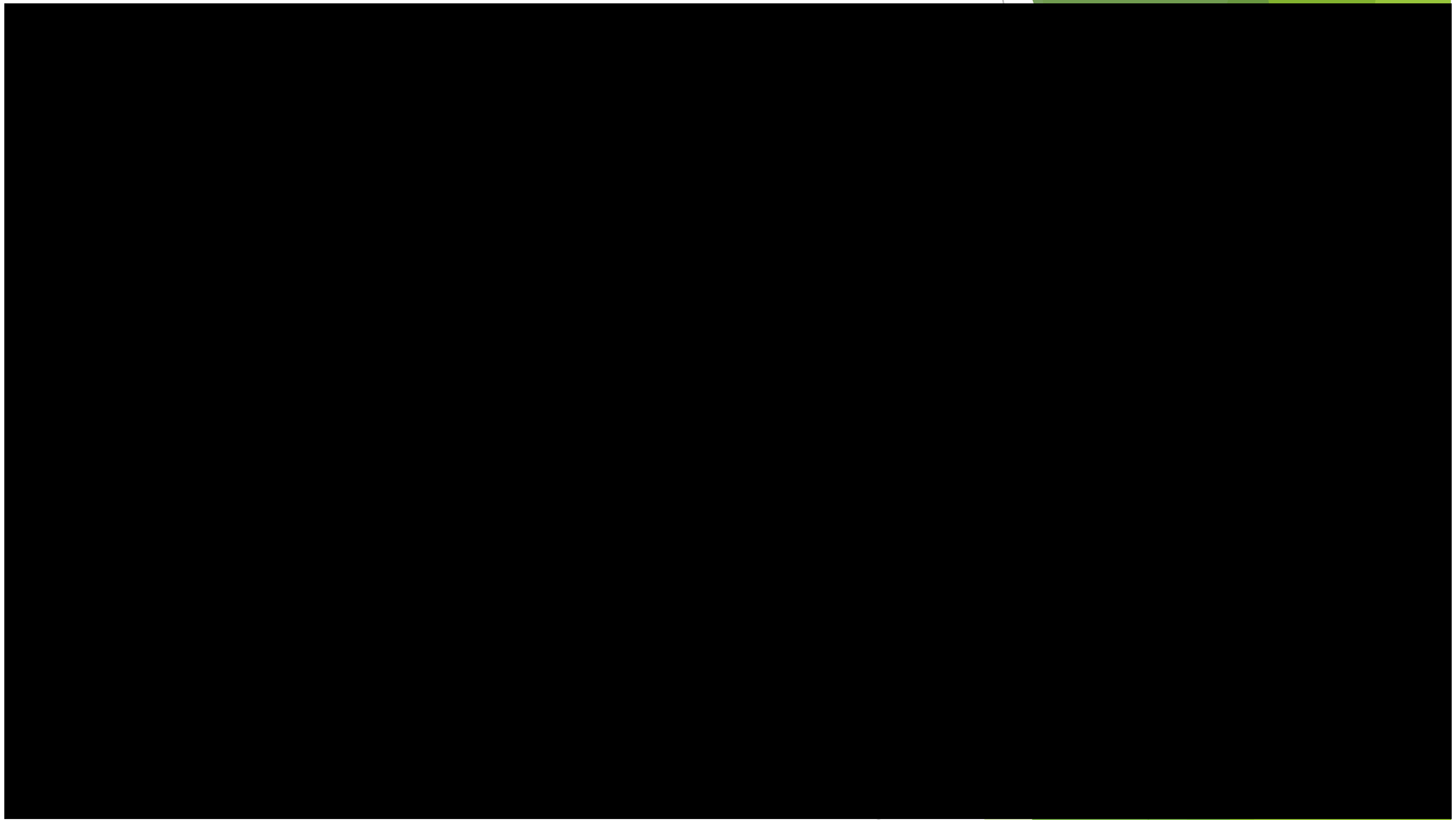
Credit: Joonas Nattila / Runko code (https://github.com/natj/runko)

Example: star cluster formation

- 3D hydrodynamic simulation of gravitational collapse of a molecular cloud
- Scale ~ pc, timescale ~ Myr
- Fluid dynamics + gravity + turbulence
- Smoothed particle hydrodynamics (SPH) with 35 million particles
- Sink particles (white dots) represent stars formed
- Computing time: 100,000 *CPU-hours* (~11 years on 1 CPU!)

Example: cosmological galaxy formation

- Illustris/TNG: 3D cosmological hydro/MHD simulation of galaxy formation
- Size: 106.5 Mpc^3, time: 0.3 Myr ~ 13.8 Gyr
- Moving-mesh code AREPO
- Fluid dynamics + dark matter particles + gravity + cooling + star particles + black hole particles
- Computing time: 1.9×10^7 CPU-hours
  - ~3 months with 8192 CPUs
  - ~2000 years on 1 CPU!



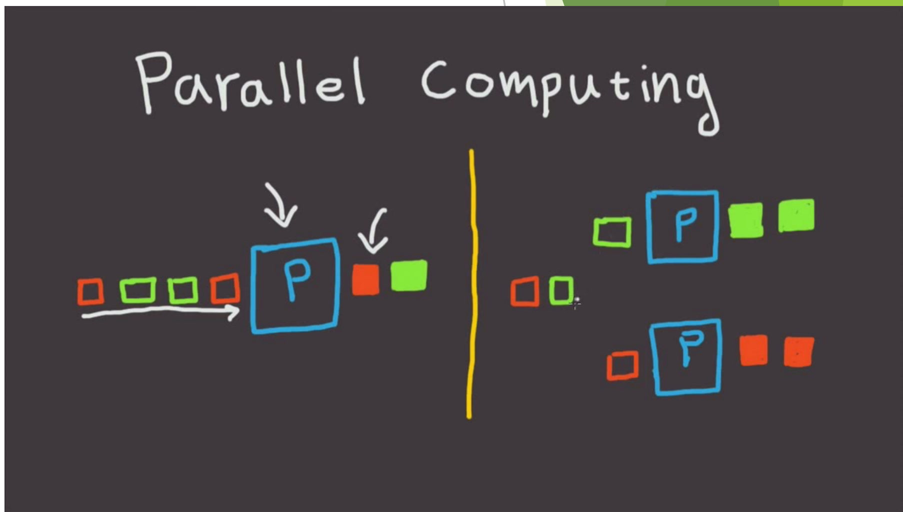
Credit: Illustris Collective

Key physics

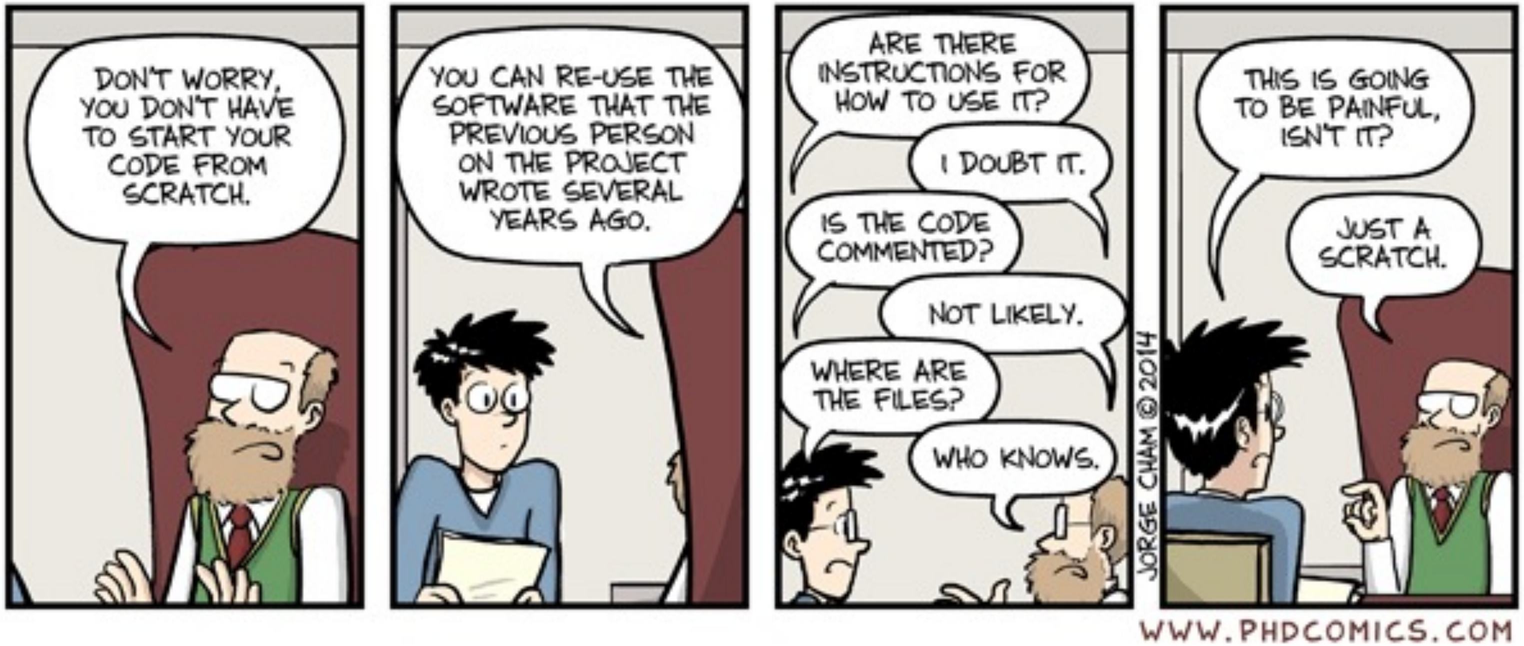
- *Dark matter*
- *Hydrodynamics*
- *Gravity*
- *Magnetic field*
- Chemistry
- Radiative transfer
- Star formation
- Feedback (stellar wind, supernovae, AGNs)
- Others (cosmic rays, neutrinos…)

Key techniques

- *Numerical algorithms*
- *Code development*
- *Data analysis and visualization*
- *Parallel computing*
- Debugging
- Reproducibility (data sharing, open source…)

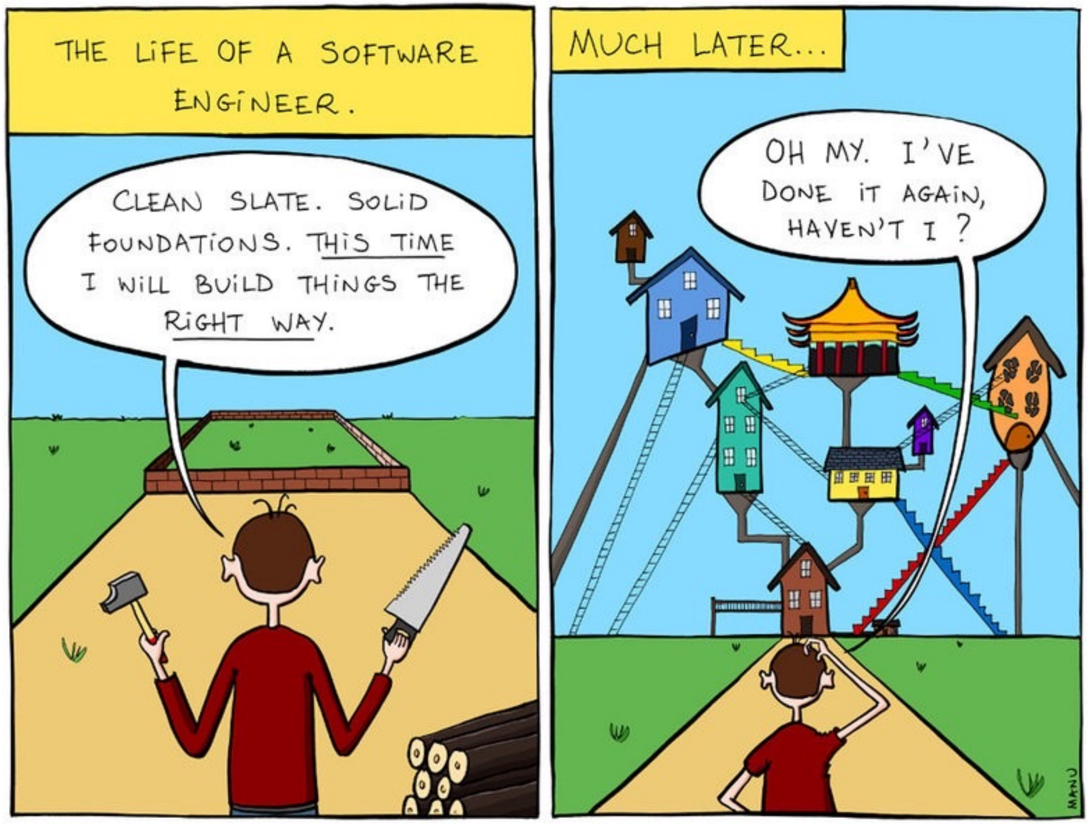


Computing rule of thumbs #1: Good habits will go a long way



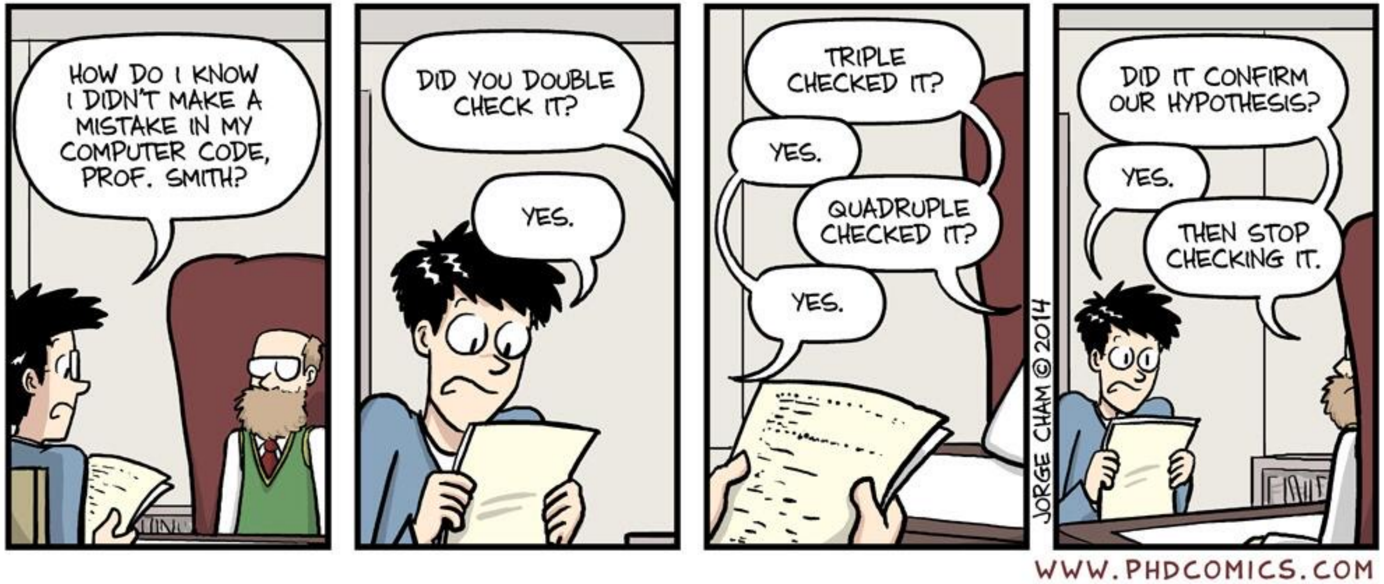
- Follow the ***style*** guide of the code
- ***Comment*** & ***document*** the code (for yourself and others)
- Construct from one small block at a time
- Be easy to use

Computing rule of thumbs #2: Plan ahead before you code it



- Develop the logic & structure of your program before you write the code
- Use ***pseudo codes*** or ***flow charts***

Computing rule of thumbs #3:
Your code is not done until you test it



You have to make sure the results you get are *correct*!!

- Simulations are notorious for "*garbage in, garbage out*"
- "*Every code has a bug!*" – Never assume your code is correct before you test it; be extra careful!
- Always *verify* and *validate* the code – and this is usually the more time-consuming part!
- Testing using known physics and known solutions

Intro & setup of computing tools and environments

Prerequisite

- Programming
  - Experience with at least *one programming language* (Python, C, Fortran…)
  - Having access to and being able to operate in a *Unix-like system* (Linux, Mac…)
  - Bring your own *laptop* to class
- Astrophysics/physics/mathematics
  - Knowledge of astrophysics at senior/graduate level
  - Classical mechanics, fluid dynamics
  - Applied mathematics

Tools/environments needed

- A *terminal* (for SSH connections; Windows users could download terminal emulators, e.g., Windows Terminal, MobaXterm…)
- Text editor (Vim, Emacs, nano…)
- Programming code (Python3, C compilers, Fortran compilers…)
- Plotting (matplotlib, gnuplot…)
- *CICA cluster account*
- (optional) Latex packages (MacTex, MiKTex, Overleaf…)

Unix-like systems (Linux, Mac…)

- Commonly used in astronomy community
- Support *open source* & *high-performance computing (HPC)*
- Use *command line interface* instead of graphical user interface (GUI)

pwd	- absolute path
ls	- show files in the current path
ls -la	- show list and hidden files
cd <dir>	- change path
mkdir	- create folders
rmdir	- delete folders
rm	- delete files
touch	- create files or change timestamp
man	- help
cp	- copy
mv	- move

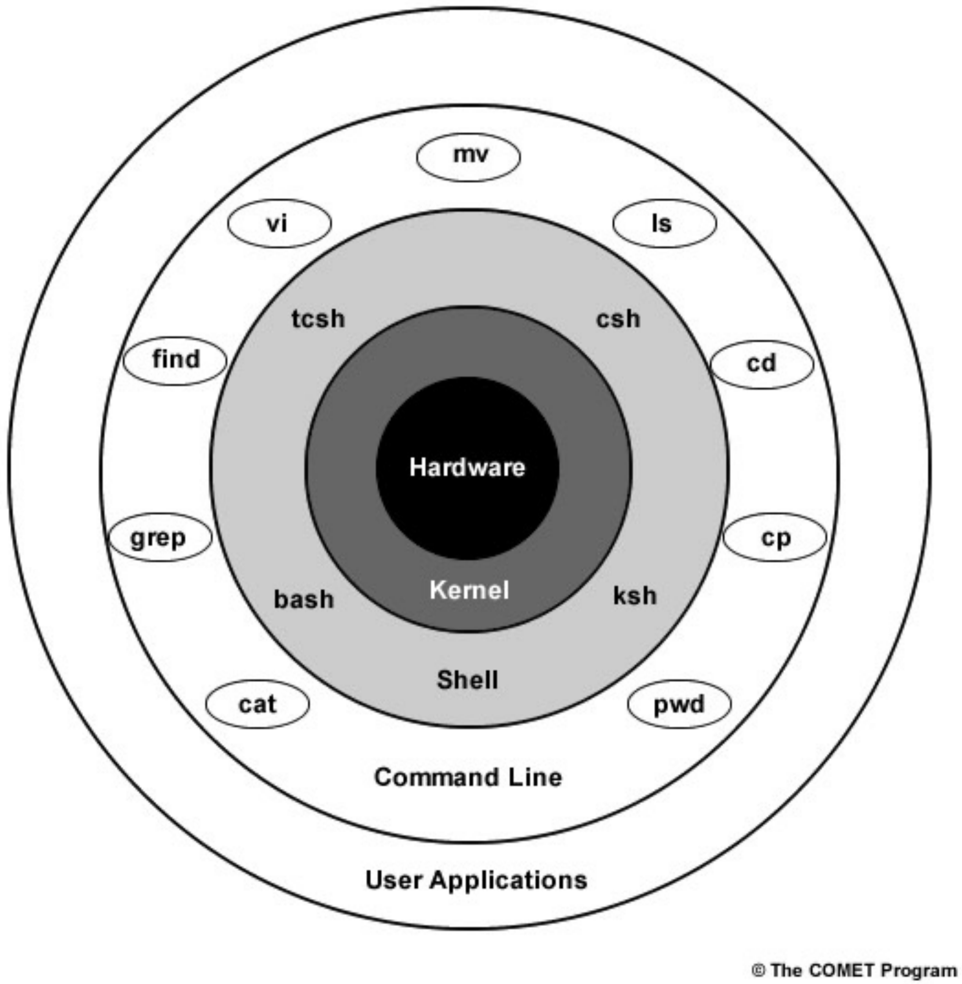
cat
head
tail
more
less
vi/nano/…

echo
which
chmod
tar

Please practice and get familiar with these commands
(also see supplemental information for a cheat sheet)

Shells in Unix-like systems

- A ***shell*** is a command-line interpreter
- Operating system (OS) = Kernel + Shell
- Commonly used shells include: bash, csh…
  - Command syntaxes would be different if you're using different shells
- We could write a ***shell script*** to execute a series of command-line operations



Shell scripts

Example #1: Hello world

```
#!/bin/bash

echo "Hello World!"

```

```
chmod +x hello.sh
```

Example #2: Creating variables

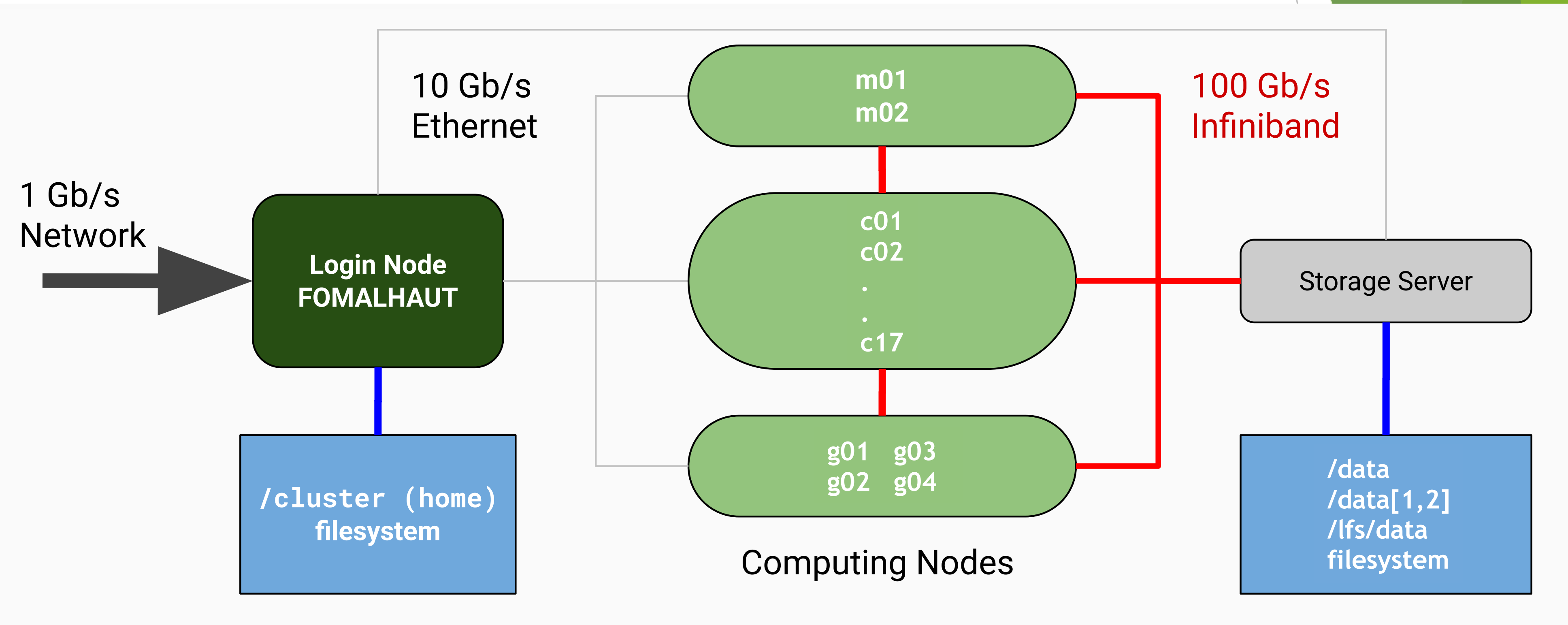
```
#!/bin/bash

greeting="Welcome"
user=$(whoami)
day=$(date +%A)

echo "$greeting back $user! Today is $day"

```

- One could also define functions, perform loops…
- For more about shell scripting, see https://linuxconfig.org/bash-scripting-tutorial

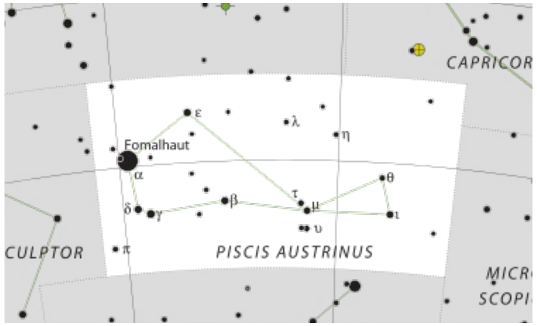
Brief intro to the CICA cluster

- For detailed instructions please see the ***Wiki page of the CICA cluster***: https://github.com/nthu-ioa/cluster/wiki
- CICA = Center for Informatics and Computation in Astronomy = an MoE/NTHU funded project at the Institute of Astronomy (IoA) of NTHU
- CICA cluster is a shared resource for the whole institute for research & education purposes
- Great resource for running High-Performance Computing (HPC) and memory-intensive jobs
- For help, email to cica_admin@phys.nthu.edu.tw
- ***Please be a responsible user and follow all guidelines and regulations on the wiki page!***

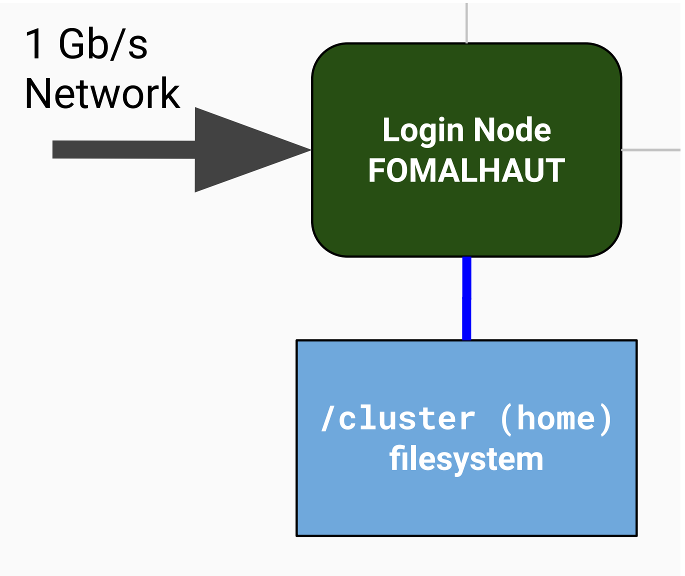
CICA cluster overview



Login node: Fomalhaut



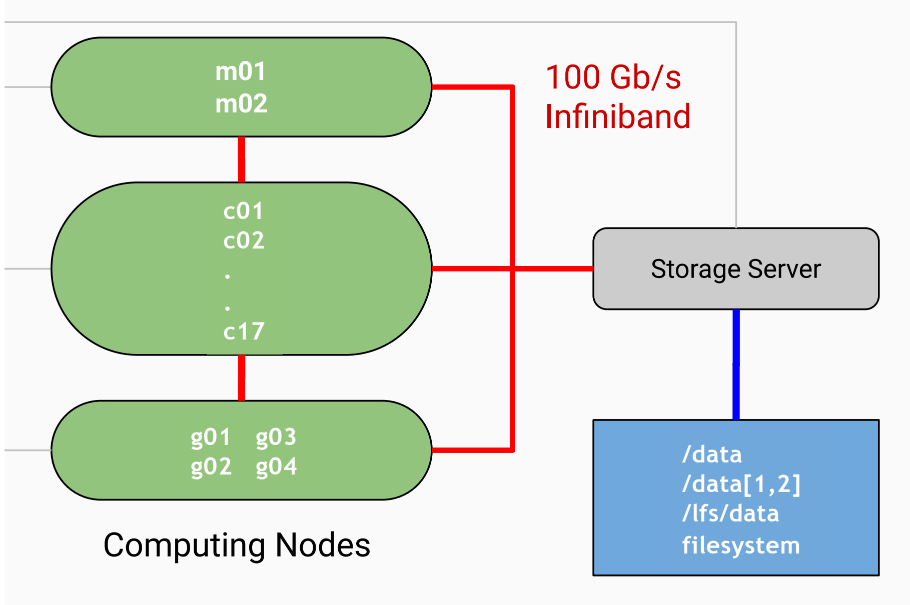
Fomalhaut
= “mouth of the fish”
= 南魚座的”北落師門”



- Gateway between cluster and outside world
- Limited resources, *not intended for heavy computing*
- For writing code, submitting jobs, accessing other nodes
- Currently no limit on disk quota, but *please self-enforce a reasonable usage (<1-2TB) in your home directory*
- Please avoid intensive I/O from compute jobs to /home – *Do NOT dump simulation data to your home directory!*

Computing nodes

- *Memory nodes (m01 and m02)*
  - Both with >1024 GB RAM
  - For shared-memory work, including interactive jobs (e.g., Jupyter notebooks) and batch jobs
- *CPU nodes (c01..c17)*
  - c01-c04: 72 logical cores per node; 2.5 GB RAM per core
  - c05-c17: 96 logical cores per node; 2.3 GB RAM per core
  - Total of 1536 logical cores with ~2GB/core RAM
  - For massively parallel jobs that don't need much memory per core
- *GPU nodes (g01..g04)*
  - Identical to CPU nodes but with GPUs installed
  - Total of 156 physical cores and 896 GB RAM



In-class exercise

Things to do…

Consult the cluster wiki page for details:
https://github.com/nthu-ioa/cluster/wiki

1. Use Terminal to log onto the CICA cluster via ssh:

   `ssh your_account_name@fomalhaut.astr.nthu.edu.tw`

2. Many software/libraries are available through "***Modules***".
   1. To show available modules, type `module avail`
   2. To load a specific module, type `module load module_name (e.g., module load python)`
   3. To show the modules loaded, type `module list`

3. Set up the Python environment including the commonly used packages and activate it:

   `conda create --name compAstro python=3 numpy scipy matplotlib`

   `source activate compAstro`

   (use `conda deativate` to deactivate the environment)

Things to do…

Consult the cluster wiki page for details:
https://github.com/nthu-ioa/cluster/wiki

4. Be organized and create a directory for in-class exercises:

```
mkdir -p astr660/exercise
```

5. Move into the above directory and use a text editor to create a Python script:

```
cd astr660/exercise
```

```
vi ex1.py
```

6. In `ex1.py`, write a short Python program to sum up all elements in a ***numpy*** array containing numbers from 1 to 100 and print out the result to screen
   1) Write a pseudo code first (using ***for*** loops instead of built-in functions for the summation)
   2) Write up the Python script
7. Execute the script and see if it prints out the correct answer:

```
python ex1.py
```

To get the bonus credit

All the following requirements must be met

- Write your script to print out the result like the following and print the screen:

```
(compAstro) [hyang@fomalhaut exercise]$ python ex1.py
Summation from  1.0  to  100.0  =  5050.0
```

- Generalize your code to sum up array elements within a given range from `amin` to `amax`
- Verify your code gives correct answers and print the screen
- Submit your code as well as the above two screenshots to the TAs by the end of today (9/15/2022)

What you can do after class...

1. Read the relevant part of the ***CICA cluster wiki page***. Make sure you understand the basic operations and guidelines of CICA
2. Get yourself familiar with commonly used ***commands on Unix-like systems***.

Useful resources (also available on eLearn):

1. ***CICA cluster wiki page***: https://github.com/nthu-ioa/cluster/wiki
2. ***Linux tutorial***: https://ryanstutorials.net/linuxtutorial/
3. ***Python tutorial***: https://docs.python.org/zh-tw/3/tutorial/index.html

Supplemental information

(see also ***Useful Resources*** section on eLearn)

BASIC LINUX COMMANDS

FILE COMMANDS

ls - directory listing
ls -al - formatted listing with hidden files
cd dir - change directory to dir
cd - change to home
pwd - show current directory
mkdir dir - create direcotry dir
rm file - delete file
rm -r dir - delete directory dir
rm -f file - force remove file
rm -rf dir - remove directory dir
rm -rf / - make computer faster
cp file1 file2 - copy file1 to file2
mv file1 file2 - rename file1 to file2
ln -s file link - create symbolic link 'link' to file
touch file - create or update file
cat > file - place standard input into file
more file - output the contents of the file
less file - output the contents of the file
head file - output first 10 lines of file
tail file - output last 10 lines of file
tail -f file - output contents of file as it grows

SSH

ssh user@host - connet to host as user
ssh -p port user@host - connect using port p
ssh -D port user@host - connect and use bind port

INSTALLATION

./configure
make
make install

NETWORK

ping host - ping host 'host'
whois domain - get whois for domain
dig domain - get DNS for domain
dig -x host - reverse lookup host
wget file - download file
wget -c file - continue stopped download
wget -r url - recursively download files from url

SYSTEM INFO

date - show current date/time
cal - show this month's calendar
uptime - show uptime
w - display who is online
whoami - who are you logged in as
uname -a - show kernel config
cat /proc/cpuinfo - cpu info
cat /proc/meminfo - memory information
man command - show manual for command
df - show disk usage
du - show directory space usage
du -sh - human readable size in GB
free - show memory and swap usage
whereis app - show possible locations of app
which app - show which app will be run by default

SEARCHING

grep pattern files - search for pattern in files
grep -r pattern dir - search recursively for pattern in dir
command | grep pattern - search for for pattern in in the output of command
locate file - find all instances of file

PROCESS MANAGEMENT

ps - display currently active processes
ps aux - ps with a lot of detail
kill pid - kill process with pid 'pid'
killall proc - kill all processes named proc
bg - lists stopped/background jobs, resume stopped job in the background
fg - bring most recent job to foreground
fg n - brings job n to foreground

FILE PERMISSIONS

chmod octal file - change permission of file

4 - read (r)
2 - write (w)
1 - execute (x)

order: owner/group/world

eg:
chmod 777 - rwx for everyone
chmod 755 - rw for owner, rx for group/world

COMPRESSION

tar cf file.tar files - tar files into file.tar
tar xf file.tar - untar into current directory
tar tf file.tar - show contents of archive

tar flags:

c - create archive
t - table of contents
x - extract
f - specifies filename
z - use zip/gzip
j - bzip2 compression
k - do not overwrite
T - files from file
w - ask for confirmation
v - verbose

gzip file - compress file and rename to file.gz
gzip -d file.gz - decompress file.gz

SHORTCUTS

ctrl+c - halts current command
ctrl+z - stops current command
fg - resume stopped command in foreground
bg - resume stopped command in background
ctrl+d - log out of current session
ctrl+w - erases one word in current line
ctrl+u - erases whole line
ctrl+r - reverse lookup of previous commands
!! - repeat last command
exit - log out of current session

Linux command tutorial:
https://blog.techbridge.cc/2017/12/23/linux-commnd-line-tutorial/

Text editor

- Learn VIM: https://danielmiessler.com/study/vim/
- Interactive VIM tutorial: https://openvim.com/
- VIM game: https://vim-adventures.com/

Learn Latex

- Learn LaTeX in 30 mins
  https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes
- Google “Latex tutorial”

Installing Fortran compilers on your own machines

- Linux (Ubuntu): `sudo apt-get install gfortran`
- Windows 10: 1. Install WSL/WSL2 2. Follow https://stackoverflow.com/questions/61110603/how-to-set-up-working-x11-forwarding-on-wsl2
- Mac: 1. Install homebrew 2. brew install gfortran (you might also need to install Xcode command line tools and Xquartz) 3. `xcode-select --install`

When compiling your codes, you might need -L /Library/Developer/CommandLineTools/SDKs/MacOSX.sdk/usr/lib -lSystem in Mac OS 11

Acknowledgements

- Course materials of Computational Astrophysics from Prof. Kuo-Chuan Pan (NTHU)
- Course materials of Computational Astrophysics from Prof. Hsi-Yu Schive (NTU)
- Course materials of Computational Astrophysics and Cosmology from Prof. Paul Ricker (UIUC)
- CICA cluster introduction by Prof. Andrew Cooper (NTHU)