An Introduction to

Modules for Experiments in Stellar Astrophysics

Meridith Joyce
Assistant Professor

Physical Science 109

(307) 766-6150

mjoyce8@uwyo.edu

A/Professor Meridith Joyce
Department of Physics and Astronomy
& School of Computing
University of Wyoming, USA
Marie Curie, **Konkoly** Observatory: ‘22–‘24
MESA Developers: 2019 – present

What is MESA?

MESA, after astropy, is the most widely used, open-source software project in astronomy and astrophysics

MESA, after astropy, is the most widely used, open-source software project in astronomy and astrophysics

MESA and astropy were named in the Astro 2020 Decadal Survey as critical instruments for the future of astronomy in the next decade

MESA is an incredibly versatile software instrument

MESA is an incredibly versatile software instrument

MESA is a team of developers and thousands of users worldwide

MESA is an incredibly versatile software instrument

MESA is a team of developers and thousands of users worldwide

MESA is a model for open-source and community-driven science

A Brief History of Time(steps)

A Brief History of Time(steps)

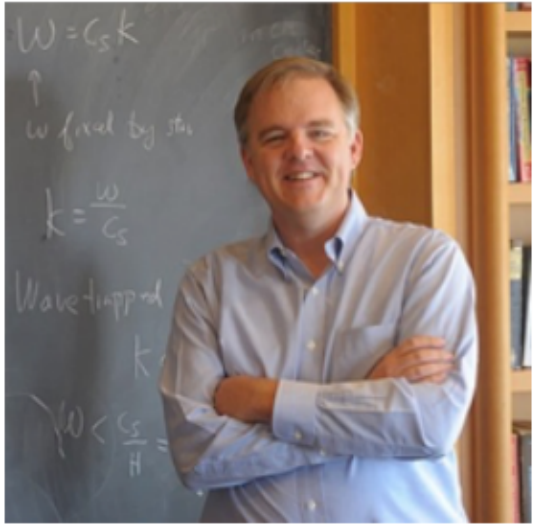
Our story
begins
with two
characters...

A Brief History of Time(steps)

Bill Paxton

Our story begins with two characters...

Lars Bildsten



UC SANTA BARBARA
Kavli Institute for Theoretical Physics

Bill
Paxton

- Senior Fellow

KITP, Kohn Hall
University of California
Santa Barbara, CA 93106

Bill Paxton

Biographical Sketch

My academic and career background is in Computer Science. While I was in graduate school, I worked with **Doug Engelbart** at **SRI** where he and his group were busy inventing personal computing (check out the 40th Anniversary Celebration of the **"Mother of All Demos"**). As soon as I got my "union card" from Stanford (Ph.D., 1977), I went to work at the Xerox Palo Alto Research Center (**PARC**) where they were creating new technologies like Ethernet, networked personal computers, bitmap displays, graphical user-interfaces, and laser printers. Two of my colleagues at PARC, **Chuck Geschke** and **John Warnock**, eventually left to form **Adobe Systems**. I joined them soon after (1983), in time to help build the original PostScript and later become one of the Adobe recipients of the **ACM's Software System Award** (1989) for PostScript's design and implementation (among other things, I did the Type 1 font algorithms starting from some of John's great ideas).

Thanks to Adobe I've been "retired" since 1990, and now I'm having fun being an unofficial scholar at the University of California, Santa Barbara (**UCSB**). The people at the Kavli Institute for Theoretical Physics (**KITP**), **Lars Bildsten** in particular, have been most welcoming and tolerant of my eccentricities. The stellar evolution program **EZ** was created as part of a project with Lars. I'm now working on **MESA**, an open source set of modules for software experiments in stellar astrophysics.

Since a physicist can do "astro-physics", I imagine a Computer Scientist can do "astro-computing" or perhaps it could be called "computational-astro-physics". That seems to be a good description of what I'm up to these days, and I'm having a great time! But if I do happen to have a bad day, I can always turn to Calvin for inspiration.

A Brief History of Time(steps)

2011 – Instrument paper I Paxton et al.

2013 – Instrument paper II Paxton et al.

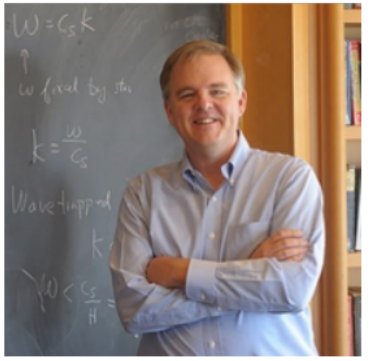
2015 – Instrument paper III Paxton et al.

2018 – Instrument paper IV Paxton et al.

2019 – Instrument paper V Paxton et al.

2023 – Instrument paper VI Jermyn et al.

Bill Paxton



Lars Bildsten

ADS library containing these papers (+errata):
https://ui.adsabs.harvard.edu/user/libraries/vT_uYj92TP6KMn4QWYBcVQ

Beatrice M. Tinsley Prize

The Tinsley Prize recognizes an outstanding research contribution to astronomy or astrophysics, of an exceptionally creative or innovative character. The prize is normally awarded every two years.

2021 – Bill Paxton

For his inspired work on providing, maintaining, and supporting the use of open-source stellar-evolution codes that have seeped into the foundation of research and education efforts.

We have **6 instrument papers**, and they should each be cited separately if you use MESA in a paper

Citing MESA

You should cite all of the available MESA instrument papers at the time of the MESA version being used, as MESA is sum of this work. Currently, that is:

```
Modules for Experiments in Stellar Astrophysics
\citep[MESA][]{Paxton2011, Paxton2013, Paxton2015, Paxton2018, Paxton2019, Jermyn2023}.
```

A bibtex file containing these references is available here:

https://docs.mesastar.org/en/release-r24.03.1/using_mesa/best_practices.html

Also!! cite the works corresponding to significant infrastructure that has been shared with the project

MESA critically rests on the hard work of many researchers who have generated the input microphysics data that underpins the `eos`, `kap`, `net`, and `neu` modules. We therefore encourage users to briefly summarize these, including appropriate citations.

```
The MESA EOS is a blend of the OPAL \citep{Rogers2002}, SCVH
\citep{Saumon1995}, FreeEOS \citep{Irwin2004}, HELM \citep{Timmes2000},
PC \citep{Potekhin2010}, and Skye \citep{Jermyn2021} EOSes.

Radiative opacities are primarily from OPAL \citep{Iglesias1993,
Iglesias1996}, with low-temperature data from \citet{Ferguson2005}
and the high-temperature, Compton-scattering dominated regime by
\citet{Poutanen2017}.  Electron conduction opacities are from
\citet{Cassisi2007} and \citet{Blouin2020}.

Nuclear reaction rates are from JINA REACLIB \citep{Cyburt2010}, NACRE \citep{Angulo1999} and
additional tabulated weak reaction rates \citet{Fuller1985, Oda1994,
Langanke2000}.  Screening is included via the prescription of \citet{Chugunov2007}.
Thermal neutrino loss rates are from \citet{Itoh1996}.
```

A bibtex file containing these references is available here:

https://docs.mesastar.org/en/release-r24.03.1/using_mesa/best_practices.html

Josiah Schwab

Adam Jermyn

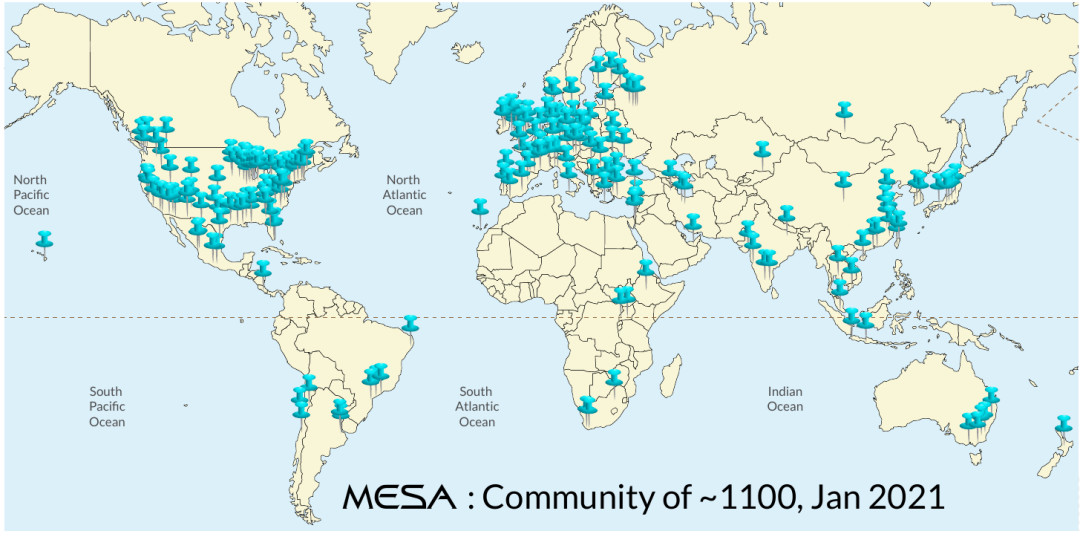
Meridith Joyce

Evan Bauer

Earl Bellinger

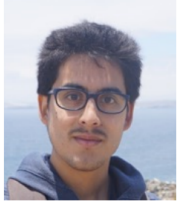
Anne Thoul

Radek Smolec



Bill Wolf

Rob Farmer

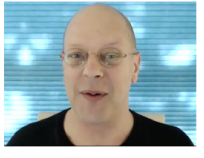


Pablo Marchant

Warrick Ball

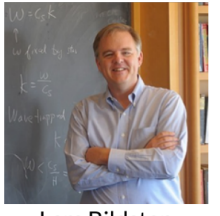
Aaron Dotter

Rich Townsend



Frank Timmes

Bill Paxton



Lars Bildsten

Matteo Cantiello

Josiah Schwab

Adam Jermyn

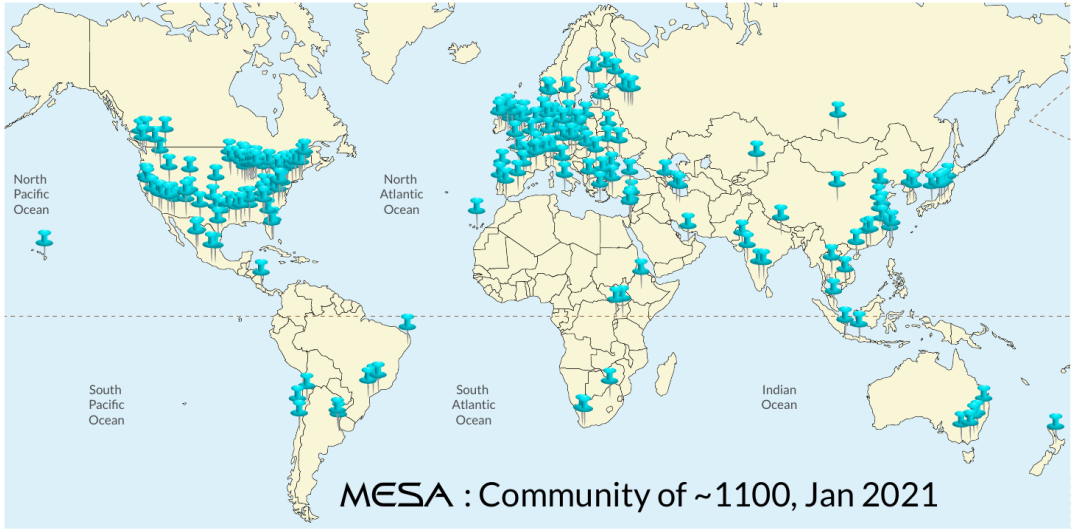
Meridith Joyce

Evan Bauer

Earl Bellinger

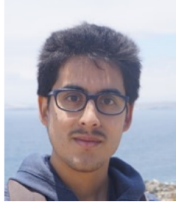
Anne Thoul

Radek Smolec



Bill Wolf

Rob Farmer



Pablo Marchant

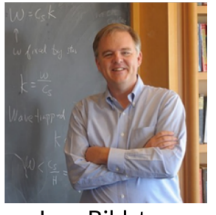
Warrick Ball

Aaron Dotter

Rich Townsend

Frank Timmes

Bill Paxton



Lars Bildsten

Matteo Cantiello

Joey Mombarg

Ebraheem Farag

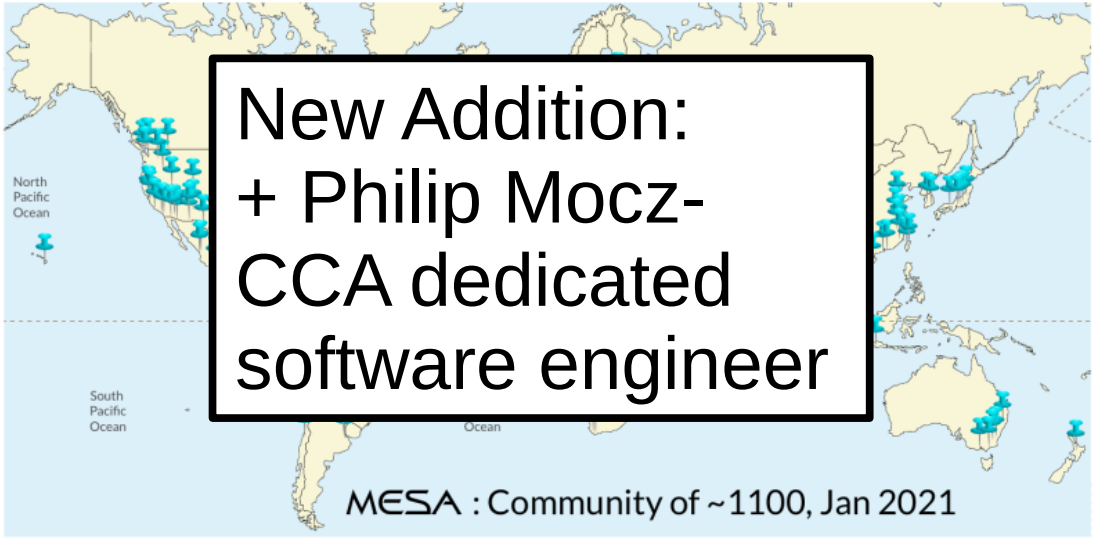
Meridith Joyce

Evan Bauer

Earl Bellinger

Anne Thoul

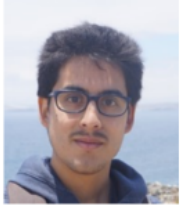
Radek Smolec



New Addition:
+ Philip Mocz-
CCA dedicated
software engineer

Bill Wolf

Matthias Fabry



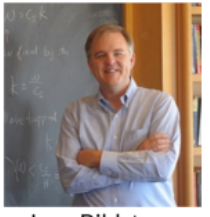
Pablo Marchant

Warrick Ball

Philip Mocz

Rich Townsend

Frank Timmes



Lars Bildsten

Matteo Cantiello

You can check out the past and current developers at:

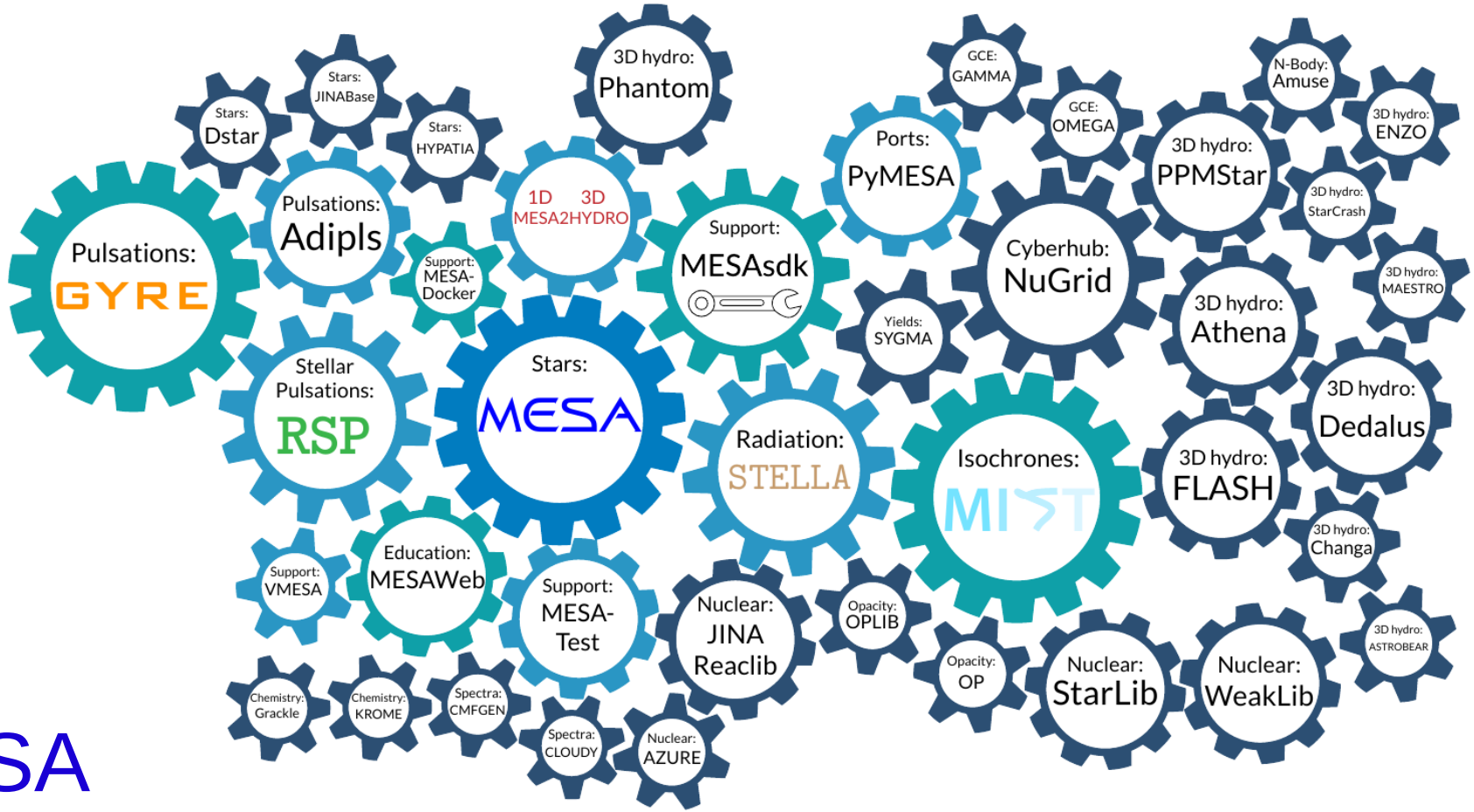
https://docs.mesastar.org/en/release-r24.03.1/about.html#the-mesa-team

MESA Summer Schools

Historically hosted at the University of California, Santa Barbara (11 years)...

Last year: MESA@Konkoly

Gaia LIGO SDSS Hubble JWST LSST TESS LCOGT NuSTAR



MESA ecosystem

Laboratory Astrophysics

MESA Open knowledge philosophy

MESA Open knowledge philosophy

Open Source

The code is freely available, but that’s only one piece of the picture. In order for a product to be truly open source, it also has to be usable

MESA Open knowledge philosophy

Open Source

The code is freely available, but that's only one piece of the picture. In order for a product to be truly open source, it also has to be usable

MESA's documentation is thorough, and extensive tutorials and pedagogical materials are freely available along with the software itself. The team is highly responsive to user questions

MESA Open knowledge philosophy

Open Knowledge

MESA Open knowledge philosophy

Open Knowledge

MESA “best practices” encourage the sharing of your parameter control files (inlists) as well as analysis scripts, numerical data, code for making figures, etc. The aspiration is *complete reproducibility* of science that uses MESA.

MESA Open knowledge philosophy

Open Knowledge

MESA “best practices” encourage the sharing of your parameter control files (inlists) as well as analysis scripts, numerical data, code for making figures, etc. The aspiration is *complete reproducibility* of science that uses MESA.

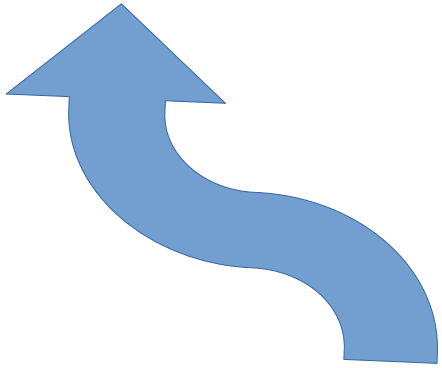
We can’t personally enforce this, but we hope the value of this level of scientific accountability is obvious. This is the way the world is heading, and grant agencies are adapting their funding priorities accordingly (e.g., open source/data sharing requirements in the EU)

Core Resources

Zenodo repository:
https://zenodo.org/records/10783349

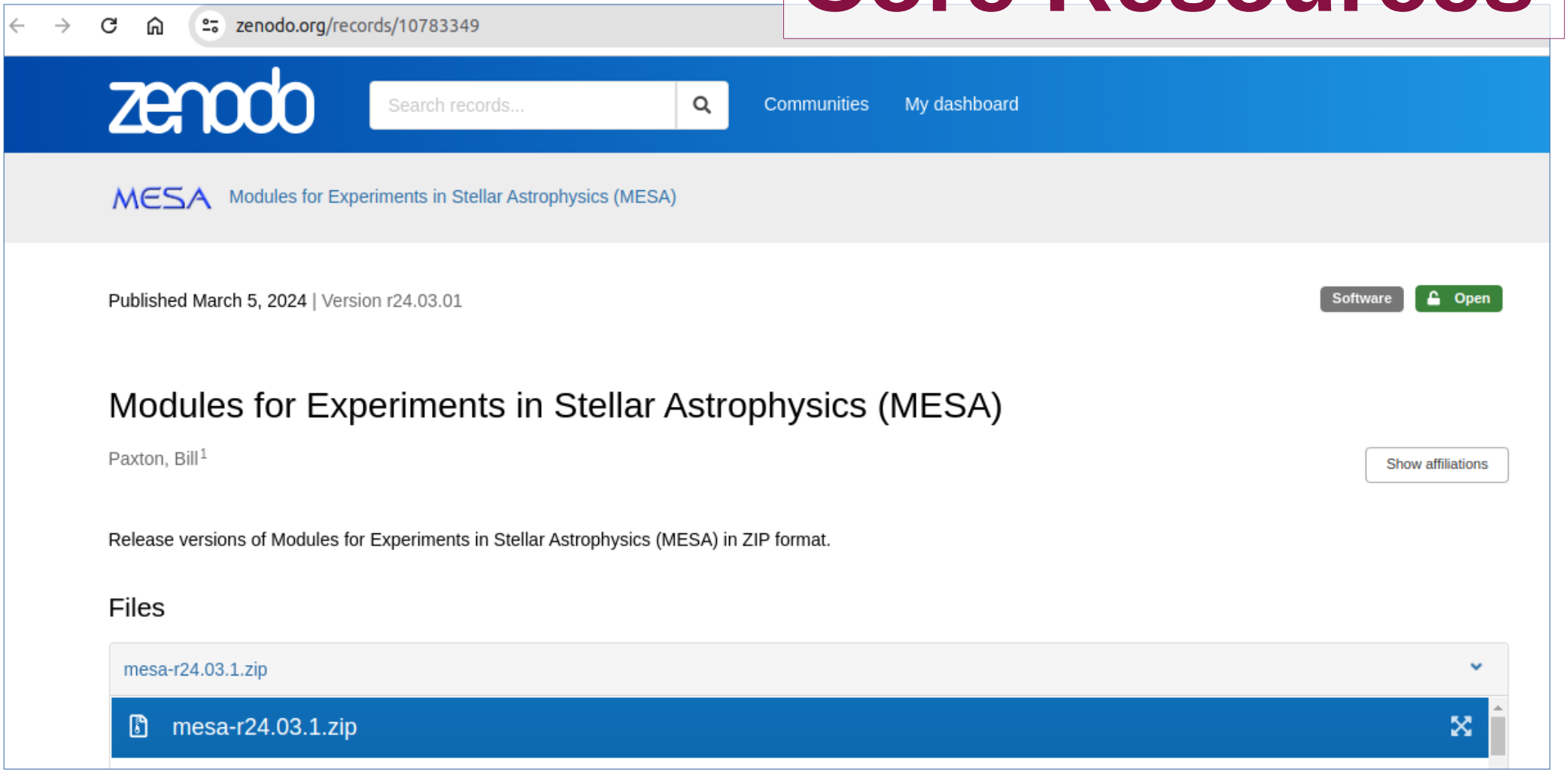
Core Resources

Zenodo repository:
https://zenodo.org/records/10783349



This is where you should download the code from!

Core Resources



Core Resources

Zenodo repository:
https://zenodo.org/records/10783349

Github repository:
https://github.com/MESAHub/mesa

Software Development Kit (SDK):
http://user.astro.wisc.edu/~townsend/static.php?ref=mesasdk

The code itself:
Mesa-r24.03.1/star/test_suite
Mesa-r24.03.1/star/defaults/*.list ; *.defaults

The web-hosted documentation:
https://docs.mesastar.org/en/release-r24.03.1/

Core Resources

inlists used in academic papers:
https://cococubed.com/mesa_market/inlists.html

Past MESA Summer School lectures and labs, including solutions:
http://cococubed.asu.edu/mesa_market/education.html (2011-2022)
https://mesahub.github.io/summer-school-2023/agenda/ (2023)

Mesa-users email list:
https://lists.mesastar.org/mailman/listinfo/mesa-users

py_mesa_reader by Bill Wolf:
https://github.com/wmwolf/py_mesa_reader

What can MESA do?

What can MESA do?

MESA is BROAD, not necessarily DEEP

What can MESA do?

MESA is BROAD, not necessarily DEEP

➡ **Depth comes from contributors improving small components of MESA relevant to their expertise**

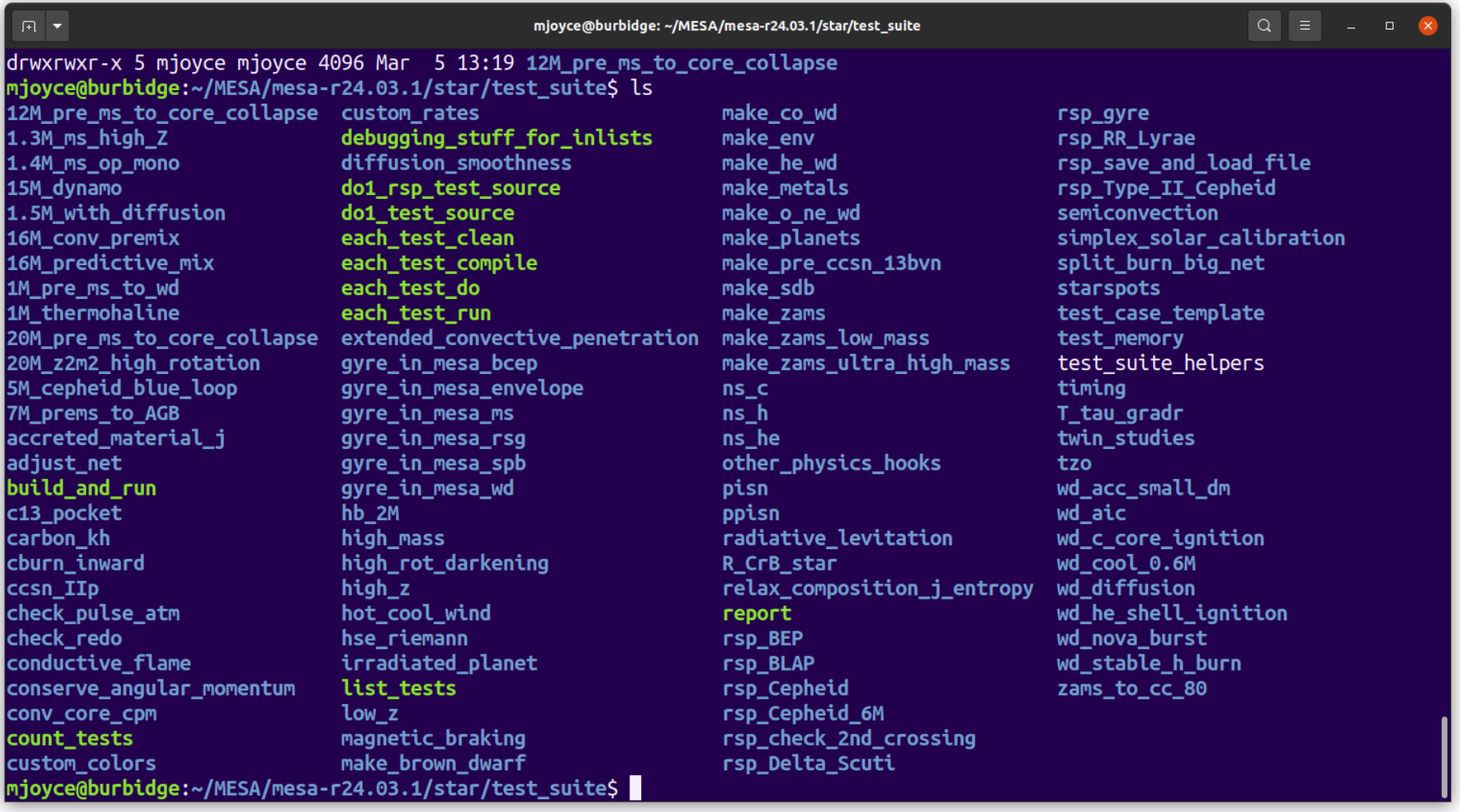
What can MESA do?

A tour of the test_suite

After installation, navigate to:

mesa-r24.03.1/star/test_suite/

A tour of the test_suite



Using MESA

Environment Variables

Environment variables
To run MESA, you must set *environment variables* that point your system to the correct paths for dependent packages

The variables are:

MESA_DIR – the location of your top-level MESA directory

MESASDK_ROOT – the location of the SDK

OMP_NUM_THREADS – the number of threads (= 2x number of cores) you choose to devote to MESA calculations.

Note that most laptops have between 2 and 8 threads available, or 1 to 4 cores. Setting **OMP_NUM_THREADS** above this will result in overextending the capabilities of your system, causing the calculations to run very slowly or crash

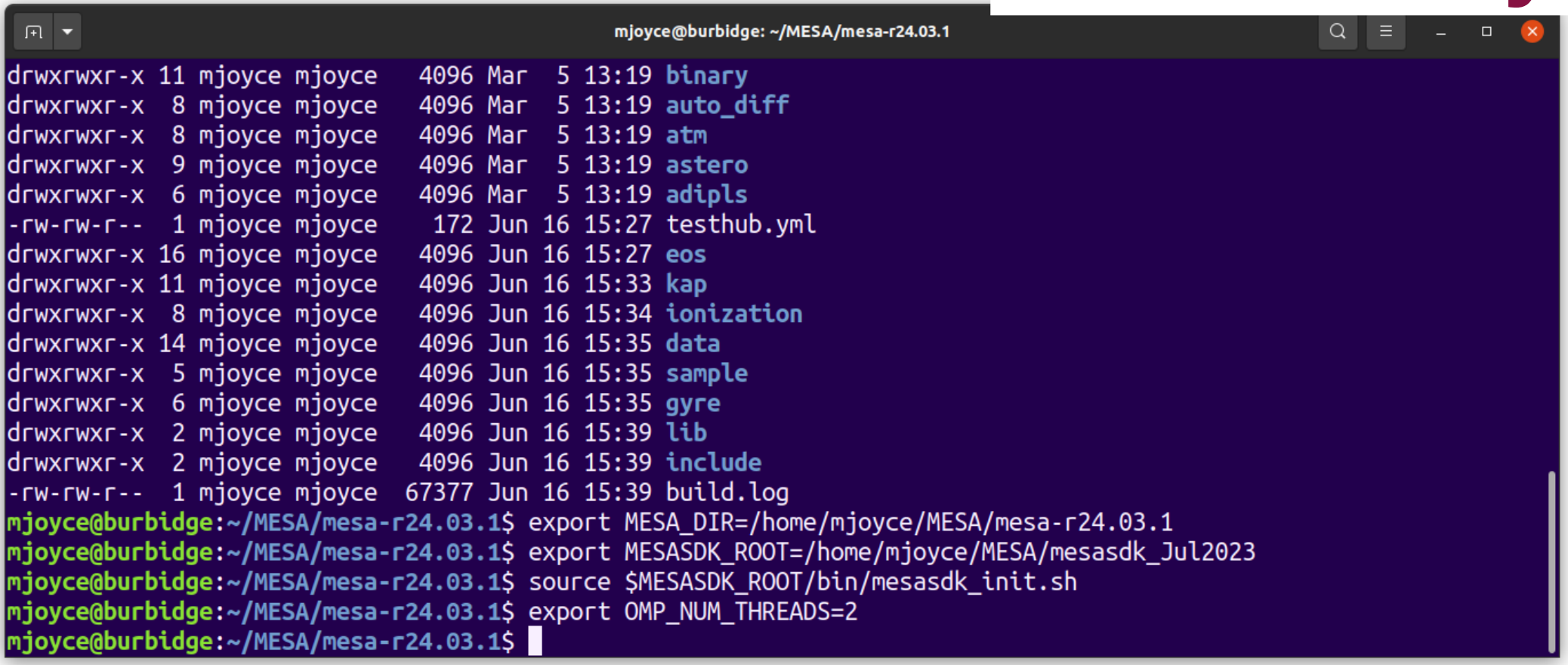
Environment Variables

The syntax for setting these variables differs between MacOS and Linux and depends on your shell environment (ex. I use Linux and bash shell)

Visit
https://www.youtube.com/watch?v=NmaLHFxpALg&ab_channel=FrankTimmes
for video guides on how to set up these variables for other systems

Other video installation guides also available at my website:
https://www.meridithjoyce.com/talks.html

Setting environment variables: the bad way



Setting environment variables the good way: .bashrc file

Since **.bashrc** (or equivalent) is read automatically each time you open a new terminal window, assigning environment variables in **.bashrc** (***or the equivalent profile of your local system***) means you do not need to assign them manually

Setting environment variables the good way: .bashrc file

.bashrc

```
1  # ~/.bashrc: executed by bash(1) for non-login shells.
2  # see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
3  # for examples
4
5  # If not running interactively, don't do anything
6  case $- in
7      *i*) ;;
8        *) return;;
9  esac
10
11 ## for screenshots only
12 export MESA_DIR=/home/mjoyce/MESA/mesa-r24031
13 export MESASDK_ROOT=/home/mjoyce/MESA/mesasdk
14 source $MESASDK_ROOT/bin/mesasdk_init.sh
15 export OMP_NUM_THREADS=2
```

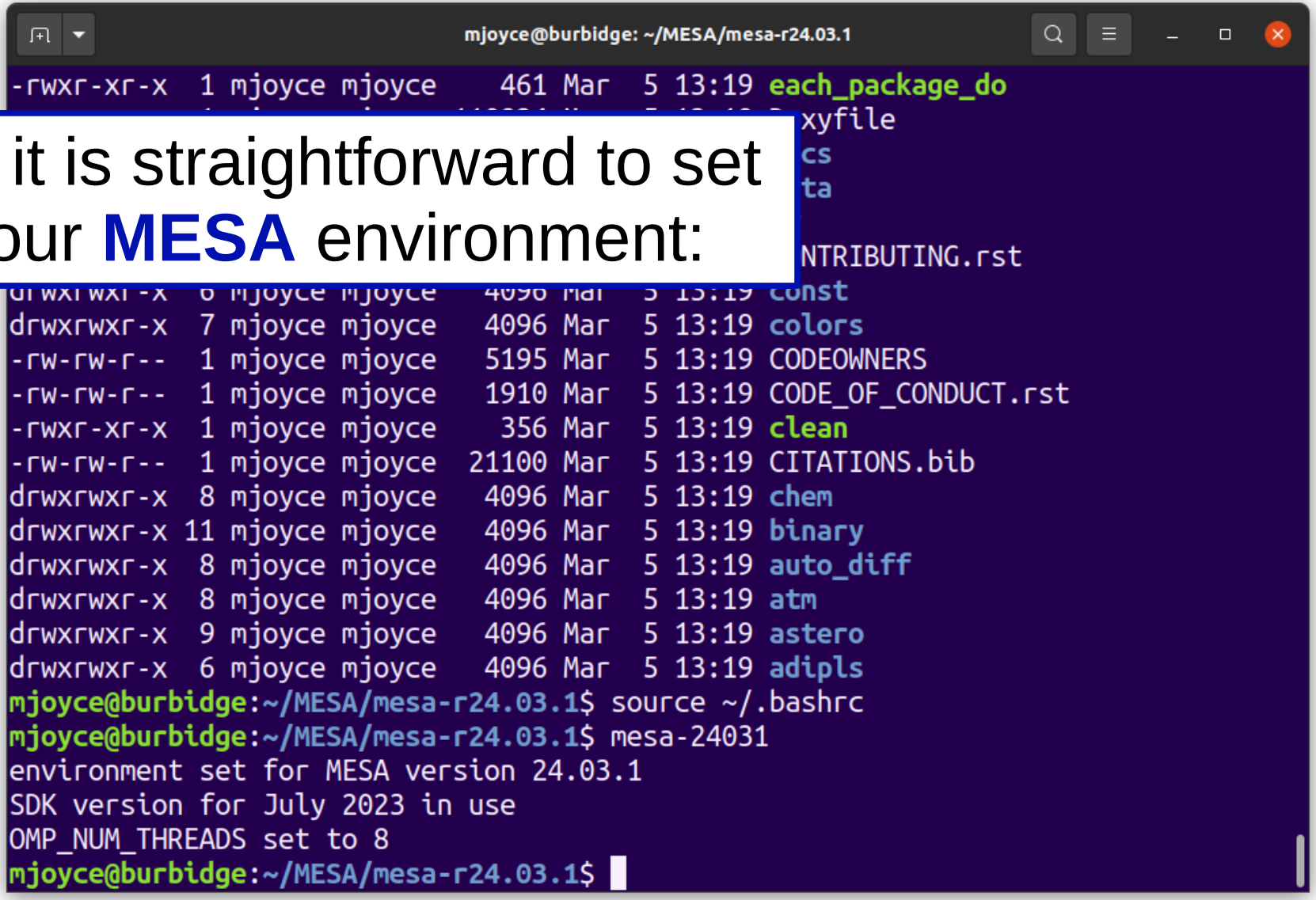
Best way: functions in .bashrc

```
function mesa-24031 {
export MESA_DIR=/home/mjoyce/MESA/mesa-r24.03.1
export MESASDK_ROOT=/home/mjoyce/MESA/mesasdk_Jul2023
source $MESASDK_ROOT/bin/mesasdk_init.sh
export OMP_NUM_THREADS=8
echo "environment set for MESA version 24.03.1"
echo "SDK version 23.7.3 in use"
echo "OMP_NUM_THREADS set to 8"
}
```

Because then you can toggle between MESA versions, if you ever have reason to do that...

```
371
372  function mesa-15140 {
373  export MESA_DIR=/home/mjoyce/MESA/mesa-r15140
374  export MESASDK_ROOT=/home/mjoyce/MESA/mesasdk_15140
375  source $MESASDK_ROOT/bin/mesasdk_init.sh
376  export OMP_NUM_THREADS=8
377  echo "environment set for MESA version 15140"
378  }
379
380
381  function mesa-22051 {
382  export MESA_DIR=/home/mjoyce/MESA/mesa-r22051
383  export MESASDK_ROOT=/home/mjoyce/MESA/mesasdk_15140
384  source $MESASDK_ROOT/bin/mesasdk_init.sh
385  export OMP_NUM_THREADS=8
386  echo "environment set for MESA version 22.05.1"
387  }
388
389
390  function mesa-23051 {
391  export MESA_DIR=/home/mjoyce/MESA/mesa-r23051
392  export MESASDK_ROOT=/home/mjoyce/MESA/mesasdk
393  source $MESASDK_ROOT/bin/mesasdk_init.sh
394  export OMP_NUM_THREADS=2
395  echo "environment set for MESA version 23.05.1"
396  }
397
```

Now it is straightforward to set up your **MESA** environment:



Microlab 0:
Check and set environment variables

(~3 minutes)

Microlab 0:
Check and set environment variables

(~3 minutes)

Show of thumbs when complete!

Setup and output

Inlists – Fortran namelists that contain **value** definitions for all of the **parameters** of your run ex) **history_filename = ‘history_my_run.data’**

Parameter libraries- *all of the possible values for your parameters can be found in the module defaults files*

Mesa-r24.03.1/star/defaults/ contains

- controls.defaults
- star_job.defaults
- pgstar.defaults
- history_columns.list
- profile_columns.list

Opacity defaults can be found in mesa-r24.03.1/kap/defaults/kap.defaults

EOS defaults in mesa-r24.03.1/eos/defaults/eos.defaults

Setup and output

Executable

star or **binary**; this is the program that is built by the compiler and runs your simulation

Scripts

clean, mk, rn, re – these are shell scripts that build and manipulate your program

Setup and output

By default, MESA keeps track of the full stellar structure of your model across evolutionary time

Output is stored in the LOGS/ directory

history.data traces evolutionary quantities

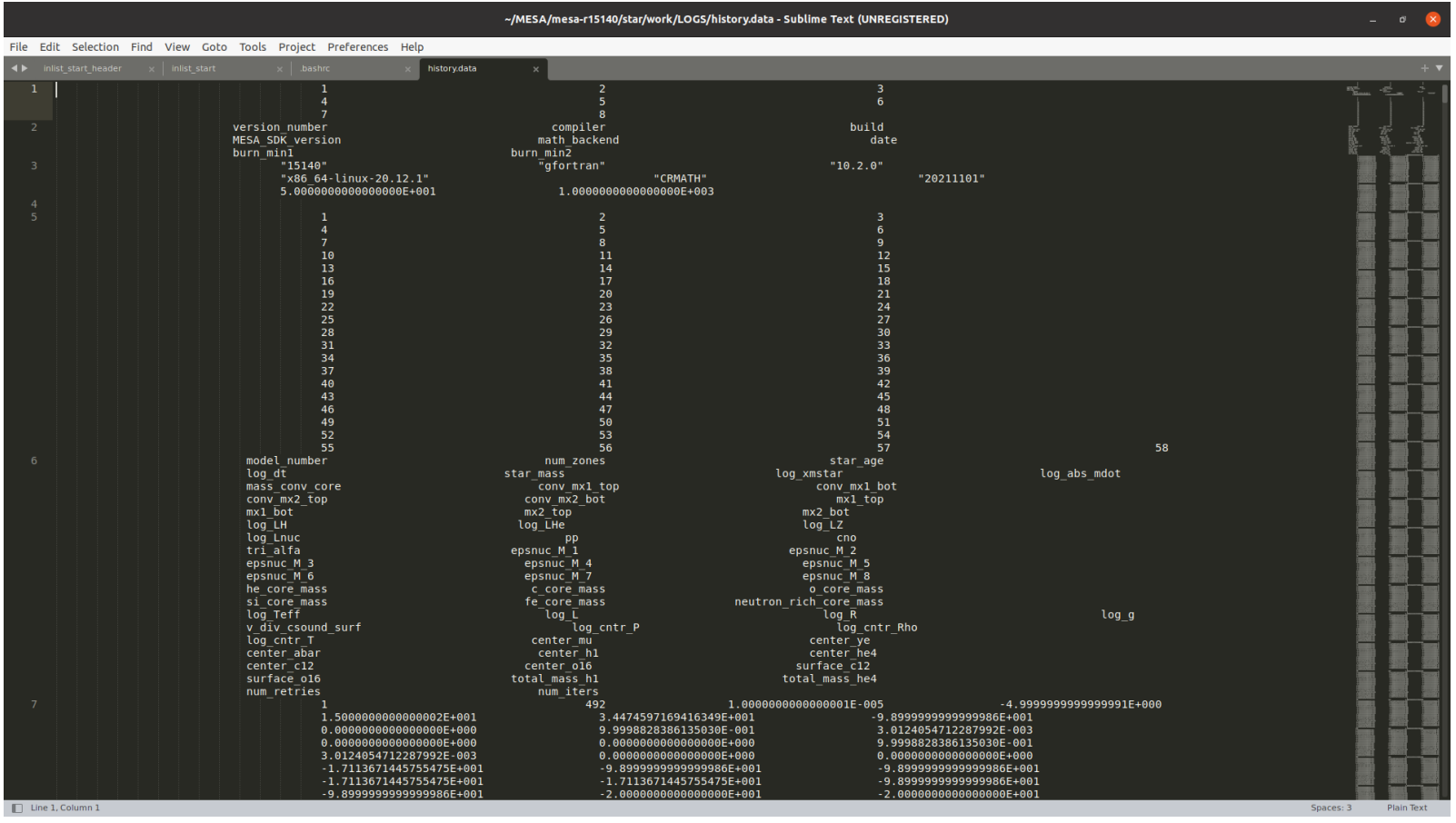
profileX.data gives you the structural model at some time step *dt.*
you can adjust the frequency of these outputs in the inlists

profiles.index provides a mapping between the integer in the profile output names and the model number from the evolutionary run (in cases where a profile is not generated at every time step)

You can also store binary snapshots of the models: **photos**

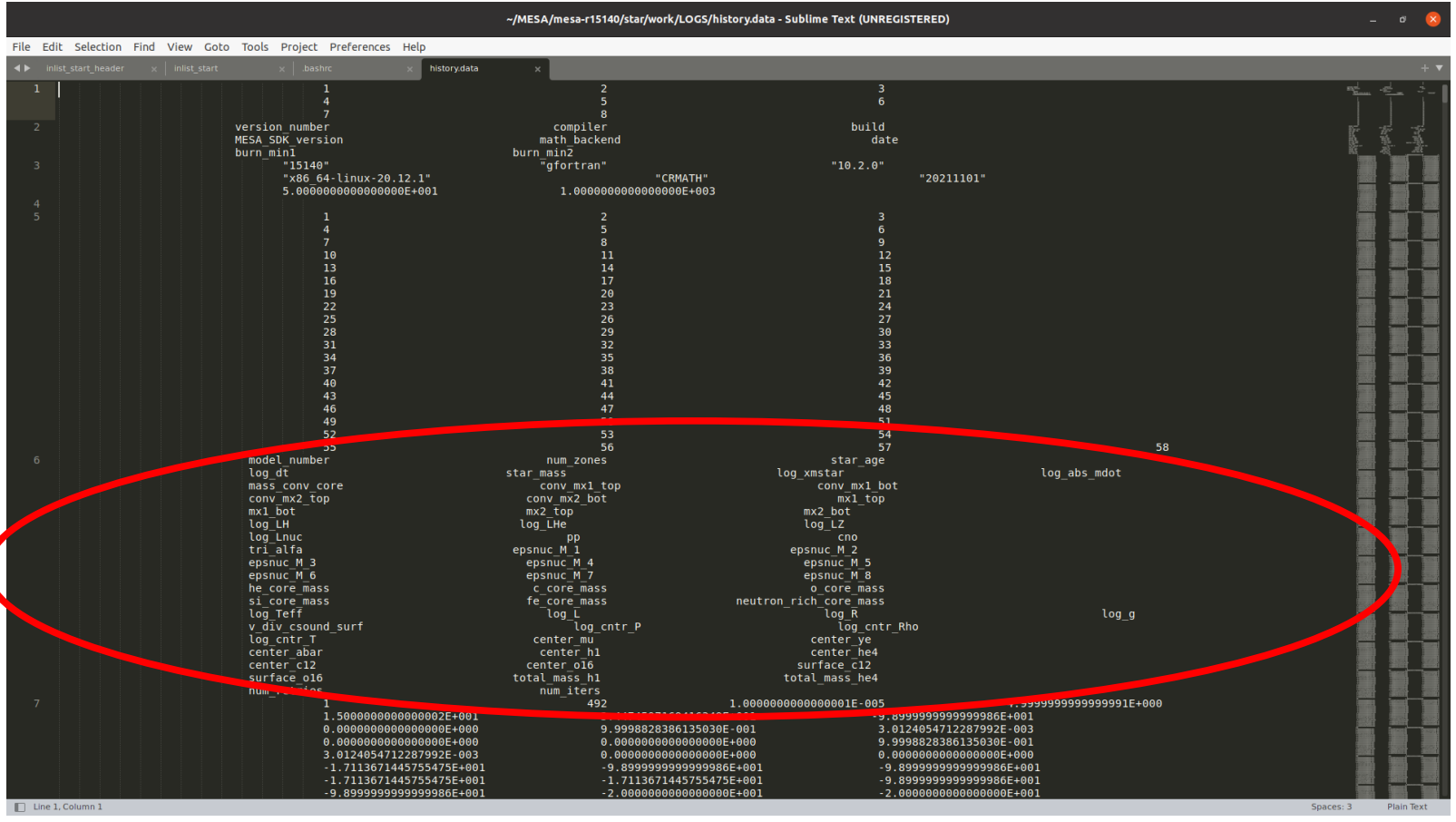
Setup and output

History output should look something like this:



Setup and output

History output should look something like this:



Microlab 1:

1) Explore your inlists
2) Run MESA out-of-the-box

(~10 minutes)

Microlab 1: Exploring MESA & inlists (10 min)

(1) What is the initial mass of the model that runs when you launch a simulation in mesa-r24.03.1/star/work/ ?

(2) What causes this model to stop?

(3) Does MESA create a starting model for this simulation, or does it load an existing model?

(4) Try to run this model using **./clean; ./mk; ./rn**
Ask a TA for help if you cannot! What do these scripts do?

(5) How hot is the star at step number 35 of this run?

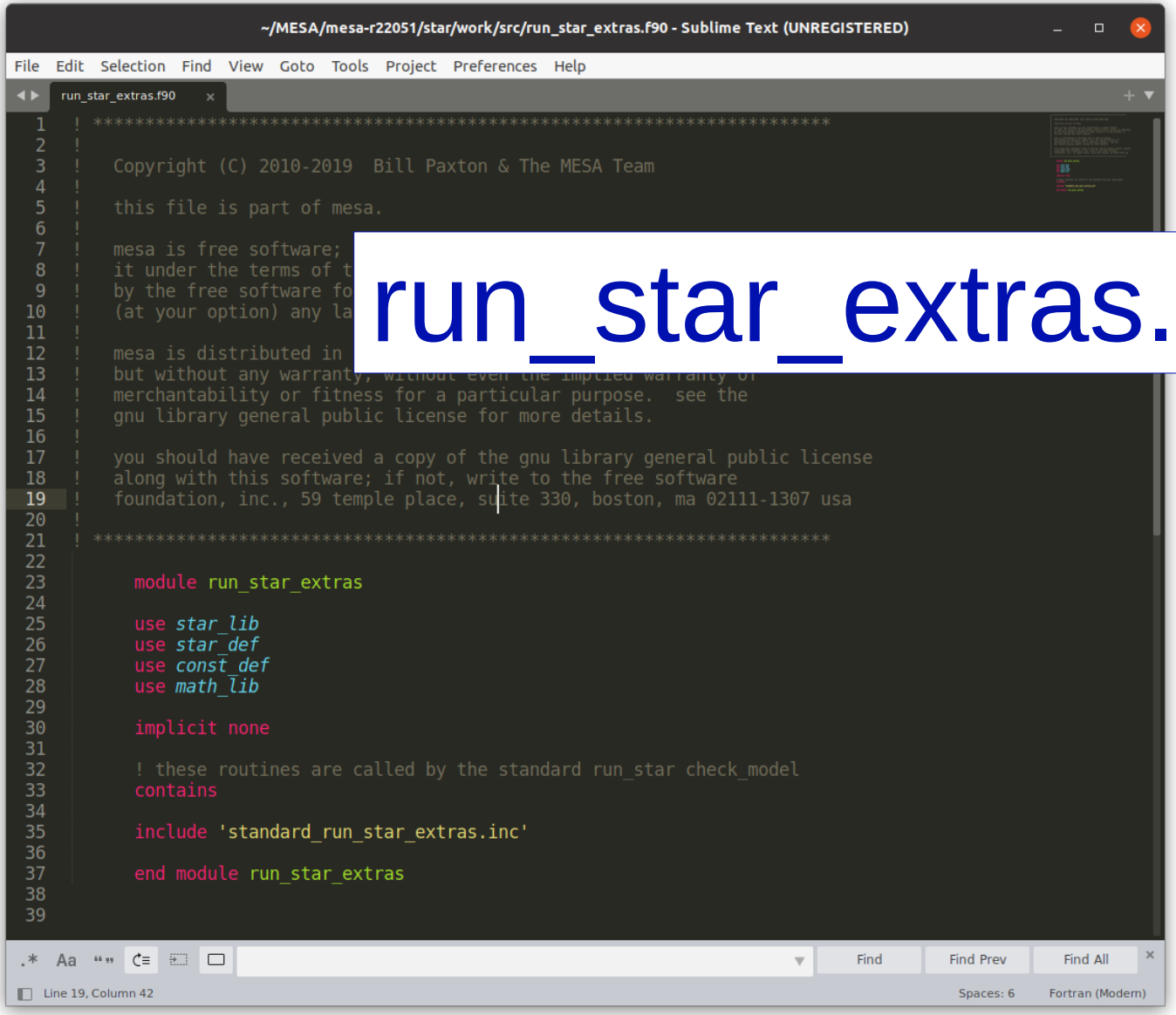
Using run_star_extras.f90

Using run_star_extras.f90, we can introduce our own

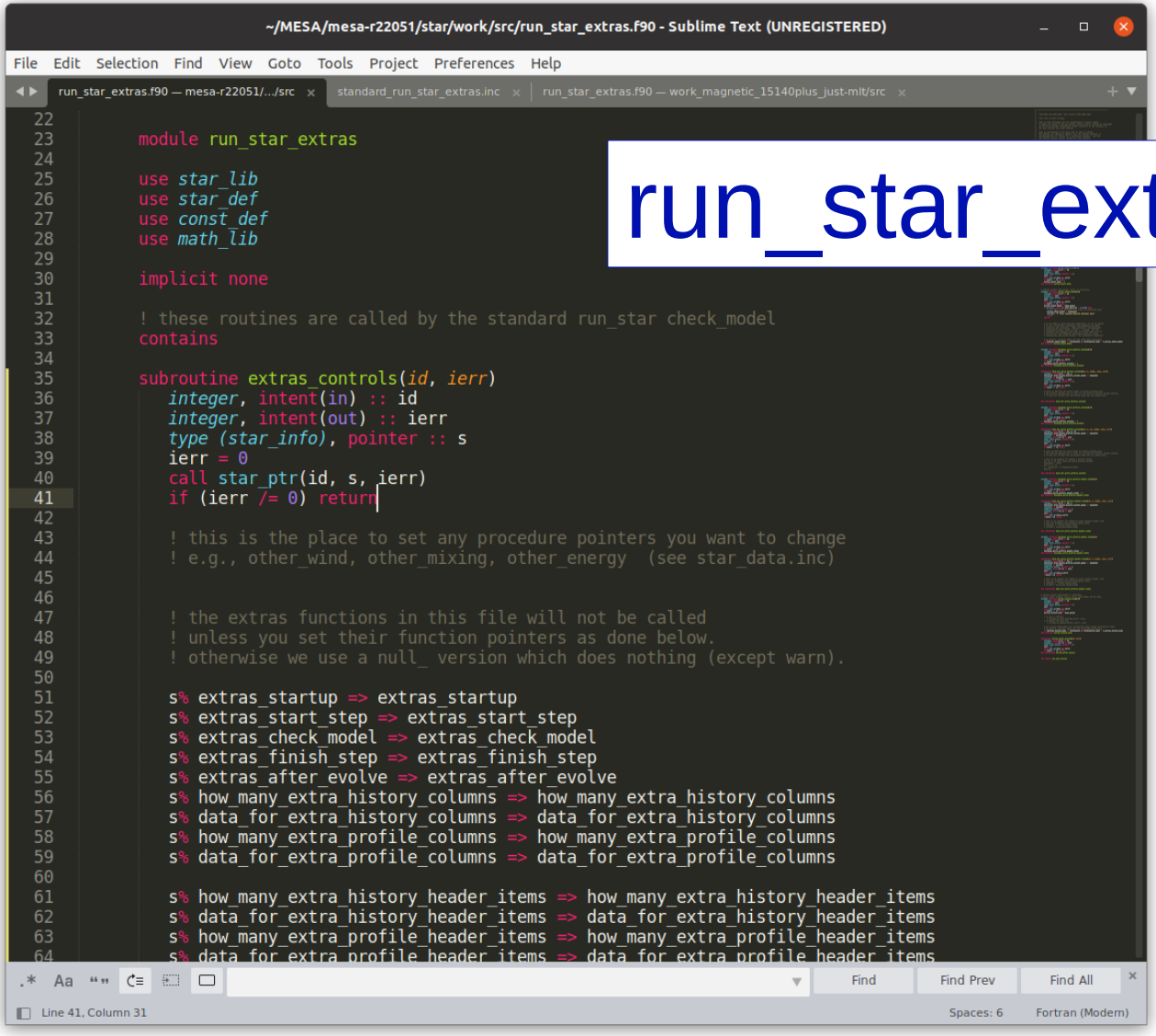
-**project-specific physics,** or

-additional functionality

without compromising the entire **MESA** source code base



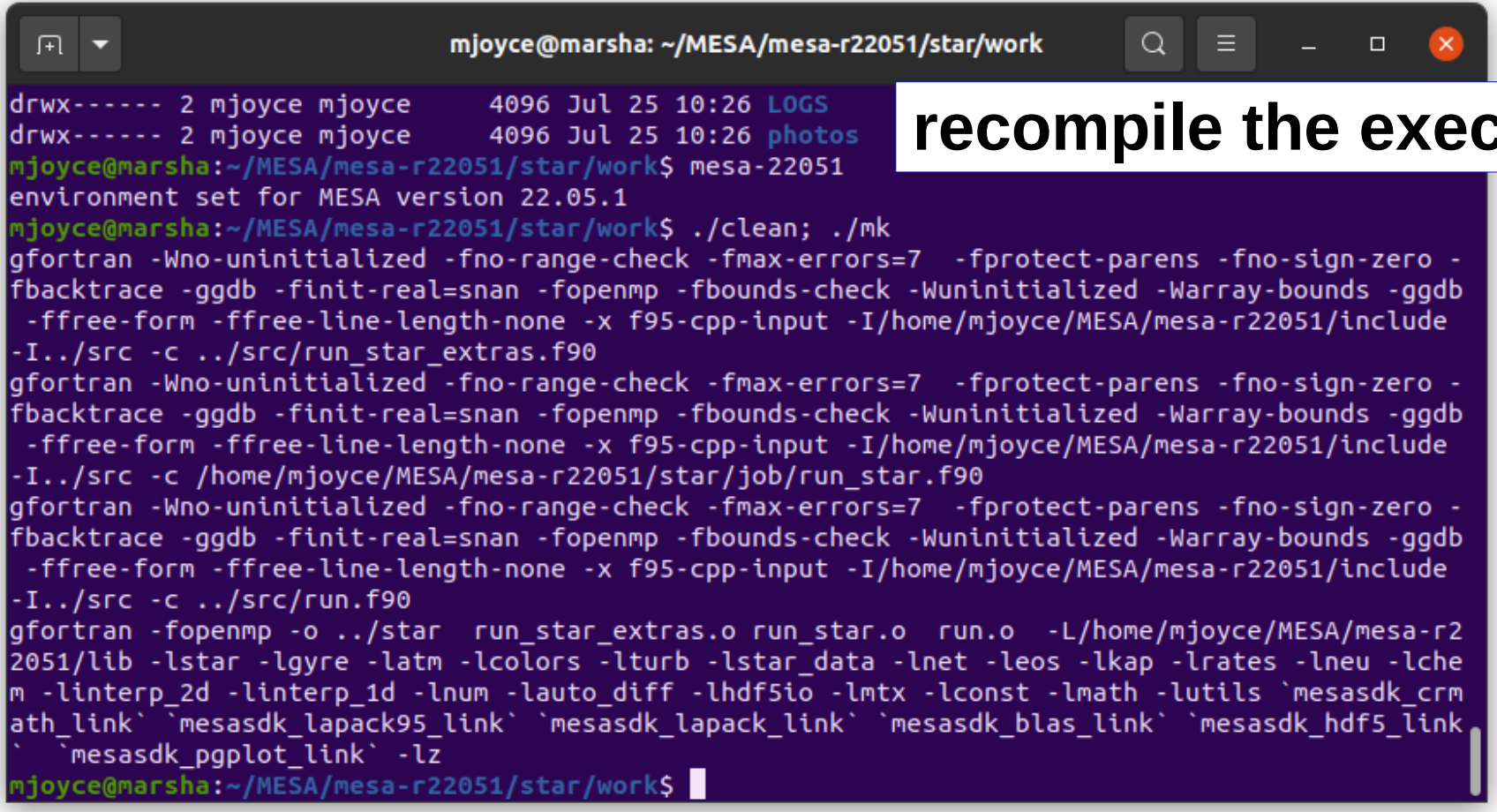
run_star_extras.f90: default



run_star_extras.f90: include

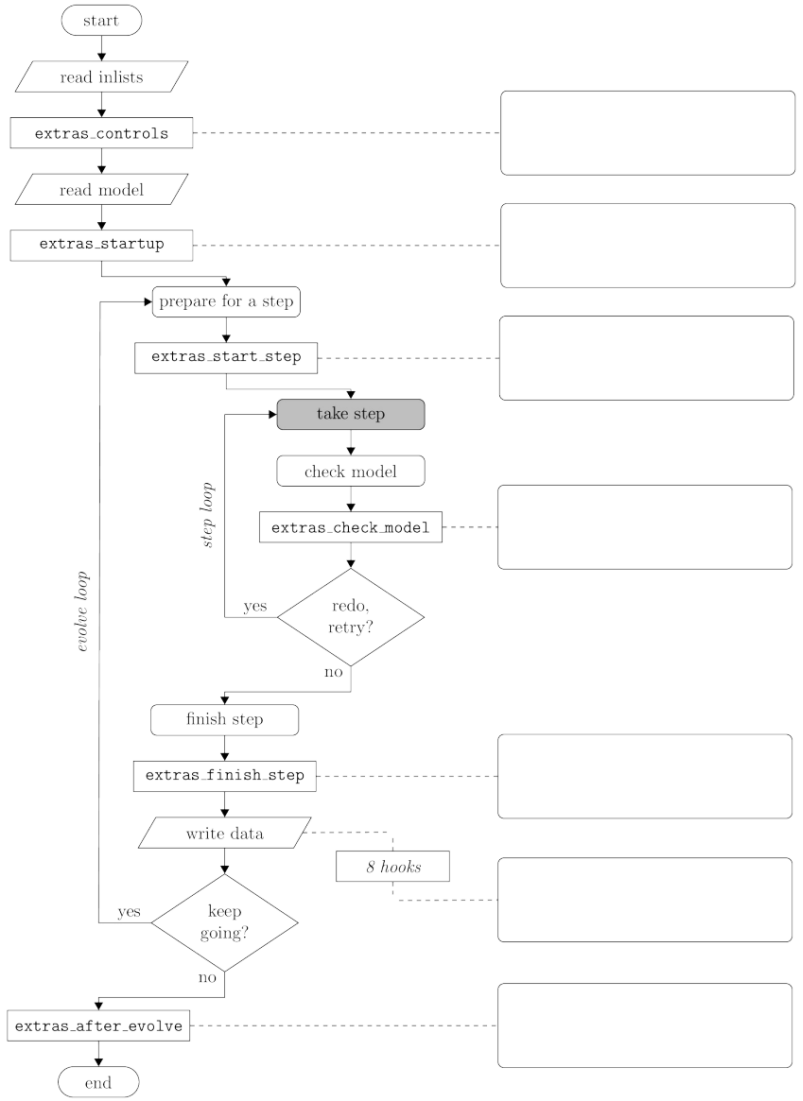
Using run_star_extras.f90

Every time you modify **run_star_extras**, you must



recompile the executable!!

Code Organization



Go to Sunny Wong's Lab 3 for a high-resolution version of this chart!

https://courtcraw.github.io/mesadu_wdbinaries/lab3.html

Code Organization

There are some actions you will want to compute *once per evolutionary time step* (**evolve loop**)

there are others you may want to compute *once per Newton solver iteration* (**step loop**)

where one evolve step contains several solver iterations

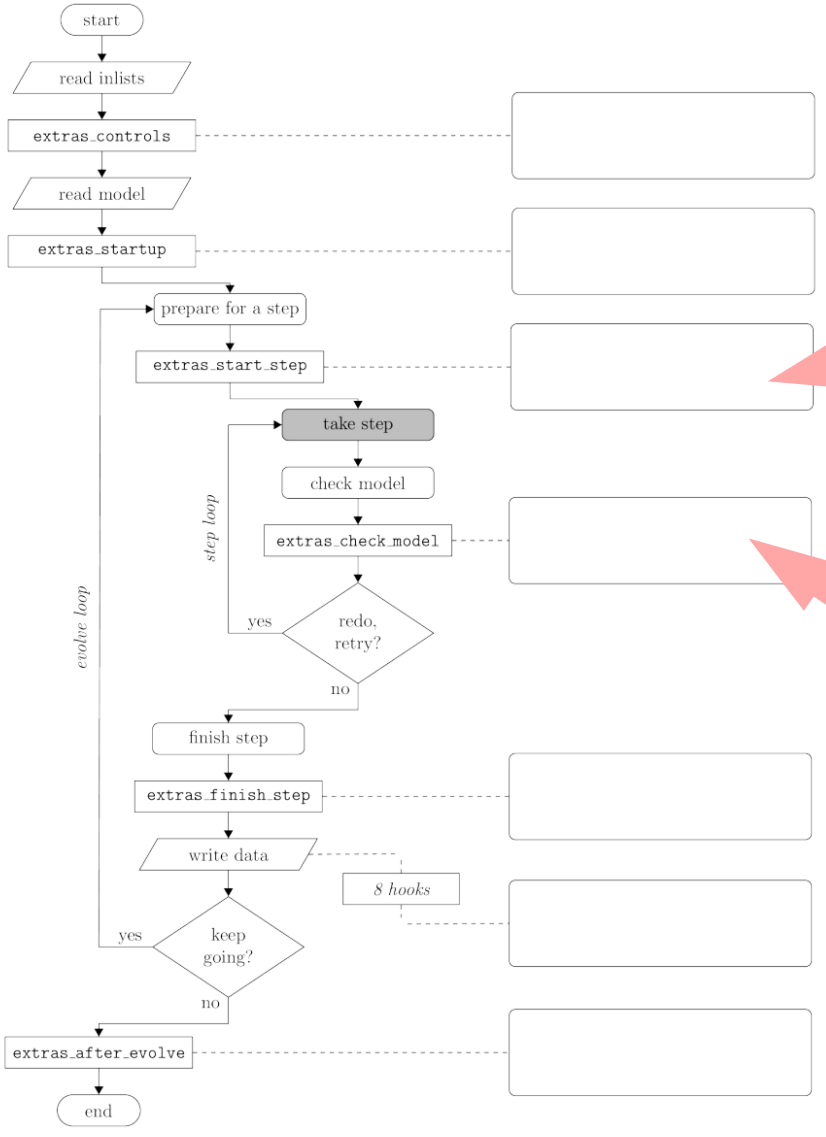
Code Organization

There are some actions you will want to compute *once per evolutionary time step* (**evolve loop**)

there are others you may want to compute *once per Newton solver iteration* (**step loop**)

where one evolve step contains several solver iterations

Example: Checking whether your model has satisfied some global physical property (i.e., reaching a certain radius) can take place once per evolutionary step




```
integer function extras_start_step(id)
   integer, intent(in) :: id
   integer :: ierr
   type (star_info), pointer :: s
   ierr = 0
   call star_ptr(id, s, ierr)
   if (ierr /= 0) return
   extras_start_step = 0
end function extras_start_step
```

```
! returns either keep_going, retry, or terminate.
integer function extras_check_model(id)
   integer, intent(in) :: id
   integer :: ierr
   type (star_info), pointer :: s
   ierr = 0
   call star_ptr(id, s, ierr)
   if (ierr /= 0) return
   extras_check_model = keep_going
   if (.false. .and. s% star_mass_h1 < 0.35d0) then
      ! stop when star hydrogen mass drops to specified level
      extras_check_model = terminate
      write(*, *) 'have reached desired hydrogen mass'
      return
   end if

   if (extras_check_model == terminate) s% termination_code = t_extras_check_model
end function extras_check_model
```

Suppose we want MESA to stop when the star reaches a certain luminosity

-When during the **step** should this condition be checked?

-How often should this condition be checked?

-In which **subroutine** should we check this condition?

Getting involved in the MESA project

Getting involved in the MESA project

users-list engagement: asking and answering questions

Feature requests

Feature development and sharing your code

Large contributions

Becoming a **MESA developer**

“MESA doesn’t work!”

“MESA sucks at X!”

“I want MESA to do something it does not currently do!”

Raise an issue on github!

Raise an issue on github!

https://github.com/MESAHub/mesa/issues

[Website]

Becoming a **MESA** developer

- Membership to the MESA developers team is done by nomination
- Any MESA developer can nominate a new member. The existing members of the team have two weeks to approve the nomination or not
- Typically, nomination is discussed with a candidate before the formal nomination process

Becoming a **MESA** developer

The MESA Team

The missions of the MESA Team are:

- **Stewardship**: supporting contributors, maintaining the access and updates, seeking enabling funding, supporting MESA Summer Schools that allow for continued engagement, documenting MESA development in the refereed literature, and sustaining advanced development.
- **Interface with the User Community**: answering questions from users, developing or accepting new code in an integrated fashion, supporting MESA workshops and events, maintaining a user registry, and identifying new MESA Team members from those most active and engaged in the intelligent use of MESA.
- **Enable Scientific Research and Education**: promoting MESA and its goals, e.g., through scientific contributions at relevant conferences, identifying science opportunities that match MESA capabilities, facilitating and encouraging appropriate scientific collaborative

Becoming a **MESA** developer

Supporting the **MESA** project is voluntary service work, but it is a prestigious project that opens a lot of opportunities

Writing code—whether developing MESA directly, its support tools, or programs that integrate with it—is a core component of MESA development

But it is not the only way to contribute!


Joey Mombarg


Ebraheem Farag


Meridith Joyce


Evan Bauer


Earl Bellinger


Anne Thoul


Radek Smolec

What we see


Bill Wolf


Matthias Fabry

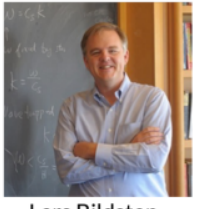

Pablo Marchant


Warrick Ball


Philip Mocz


Rich Townsend


Frank Timmes


Lars Bildsten


Matteo Cantiello


Evan Bauer


Evan Bauer

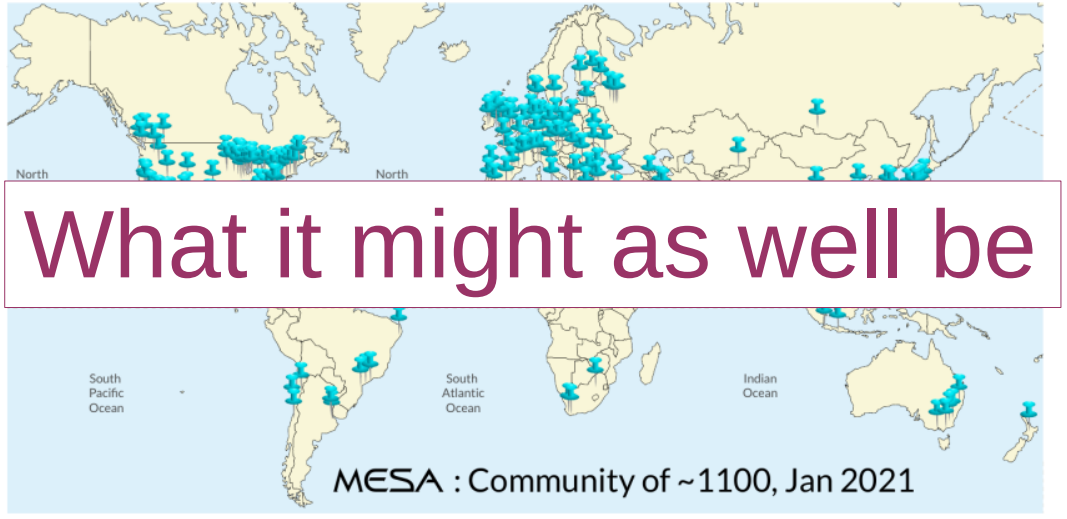

Meridith Joyce


Evan Bauer


Evan Bauer


Meridith Joyce


Evan Bauer



What it might as well be


Evan Bauer


Evan Bauer


Evan Bauer


Evan Bauer


Evan Bauer


Evan Bauer


Evan Bauer


Evan Bauer

Becoming a **MESA** developer

As developers inevitably move on (or advance to professor, at which point they no longer have time to do anything), it is important to bring outstanding young astrophysicists into the team to keep MESA relevant and ensure that it continues to serve the needs of the research community

Becoming a **MESA** developer

As developers inevitably move on (or advance to professor, at which point they no longer have time to do anything), it is important to bring outstanding young astrophysicists into the team to keep MESA relevant and ensure that it continues to serve the needs of the research community

We are especially interested in **recruiting women**. If you would like to know more about what it means to be a **MESA developer**, please talk to me at this workshop!

UNIVERSITY OF WYOMING | WyoWeb | Giving | Athletics | Alumni | +Application Login | Search the UW site...

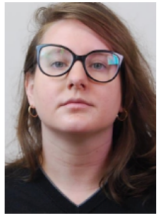
PROGRAMS & MAJORS | ADMISSIONS | COST & FINANCIAL AID | CURRENT STUDENTS | MORE | APPLY TO UW

UW Homepage > School of Computing > People > Faculty > Meridith Joyce

UNIVERSITY OF WYOMING | SCHOOL OF COMPUTING

MENU

- RESEARCH SCIENTISTS
- ELLEN AIKENS | ASSISTANT PROFESSOR
- MARGO BERENDSEN | GIS ANALYST
- SEAN FIELD | ASSISTANT PROFESSOR
- MERIDITH JOYCE | ASSISTANT PROFESSOR



Meridith Joyce

Assistant Professor | Physics | Astronomy | Mathematics | Statistics | School of Computing

Room 4083, Engineering

Laramie, WY
Phone: 307-766-5299
Email: mjoyce8@uwyo.edu

Talk to me about PhD openings at **Wyo** or how to apply to postdocs and fellowships in the **United States** and **Europe**!