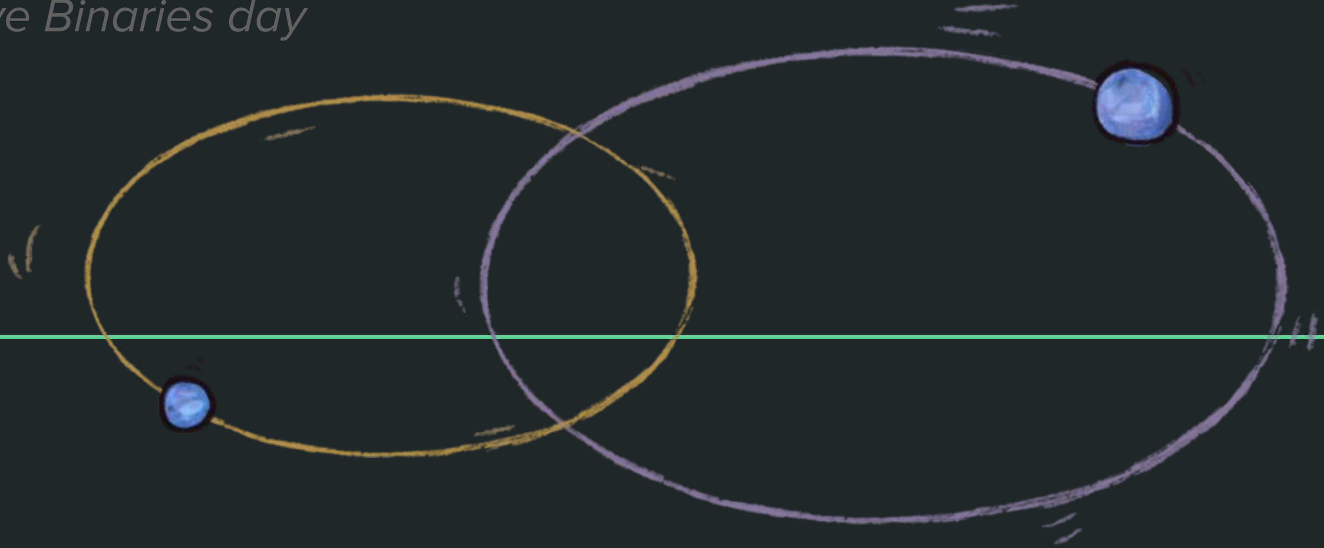
The binary module

20 June 2024

Massive Binaries day



Annachiara Picco

Mesa Down Under 2024

KU LEUVEN

The `star` module

$$\frac{\partial r}{\partial m} = \frac{1}{4\pi r^2 \rho}$$

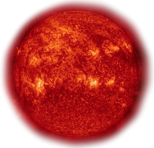
$$\frac{\partial P}{\partial m} = -\frac{Gm}{4\pi r^4}$$

$$\frac{\partial T}{\partial m} = -\frac{GmT}{4\pi r^4 P}\nabla$$

$$\frac{\partial l}{\partial m} = \varepsilon_{\mathrm{nuc}} - c_P \frac{\partial T}{\partial t} - \frac{\delta}{\rho}\frac{\partial P}{\partial t}$$

$$\frac{\partial X_i}{\partial t} = m_i \left[\sum_j r_{ji} - \sum_k r_{ik}\right] + \frac{\partial}{\partial m}\left[(4\pi \rho r^2) D_{\mathrm{mix}} \frac{\partial X_i}{\partial m}\right]$$

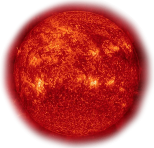
Star 1



The binary module

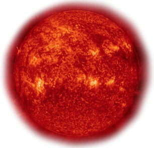
$$\frac{\partial r}{\partial m} = \frac{1}{4\pi r^2 \rho}$$
$$\frac{\partial P}{\partial m} = -\frac{Gm}{4\pi r^4}$$
$$\frac{\partial T}{\partial m} = -\frac{GmT}{4\pi r^4 P}\nabla$$
$$\frac{\partial l}{\partial m} = \varepsilon_{\mathsf{nuc}} - c_P \frac{\partial T}{\partial t} - \frac{\delta}{\rho}\frac{\partial P}{\partial t}$$
$$\frac{\partial X_i}{\partial t} = m_i \left[\sum_j r_{ji} - \sum_k r_{ik}\right] + \frac{\partial}{\partial m}\left[(4\pi \rho r^2) D_{\mathsf{mix}} \frac{\partial X_i}{\partial m}\right]$$

Star 1

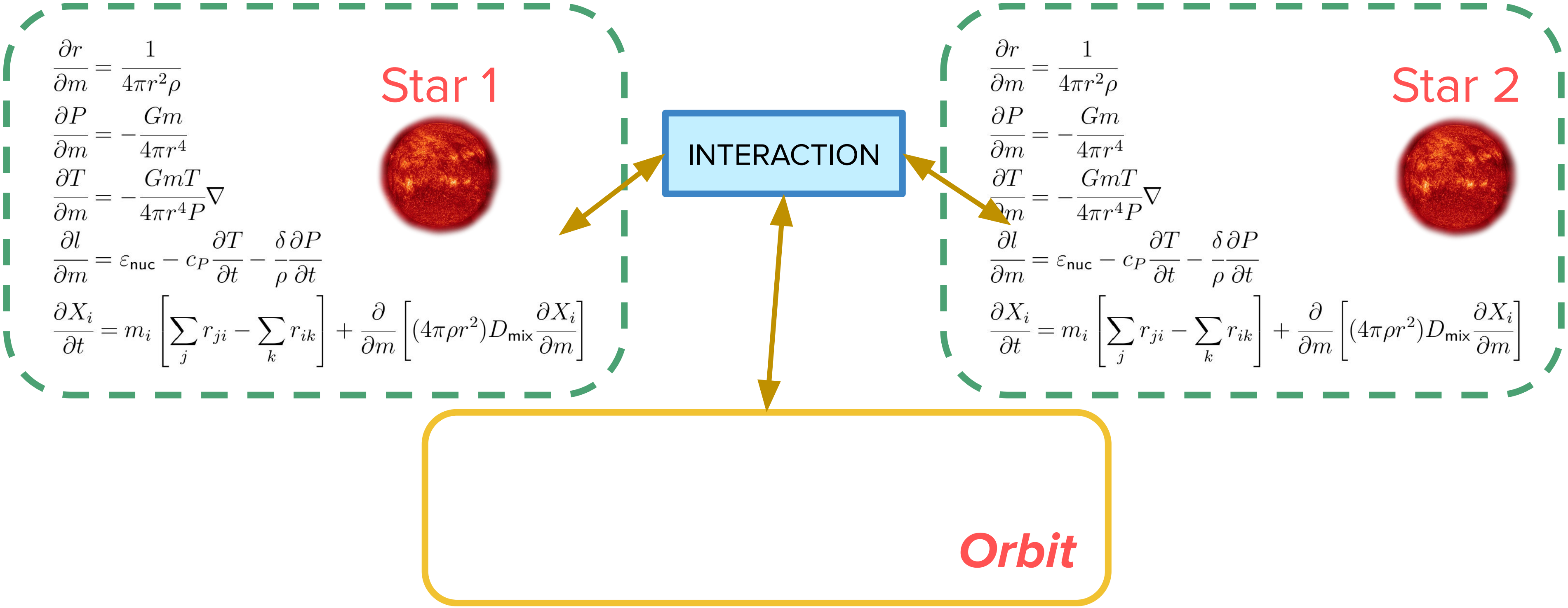


$$\frac{\partial r}{\partial m} = \frac{1}{4\pi r^2 \rho}$$
$$\frac{\partial P}{\partial m} = -\frac{Gm}{4\pi r^4}$$
$$\frac{\partial T}{\partial m} = -\frac{GmT}{4\pi r^4 P}\nabla$$
$$\frac{\partial l}{\partial m} = \varepsilon_{\mathsf{nuc}} - c_P \frac{\partial T}{\partial t} - \frac{\delta}{\rho}\frac{\partial P}{\partial t}$$
$$\frac{\partial X_i}{\partial t} = m_i \left[\sum_j r_{ji} - \sum_k r_{ik}\right] + \frac{\partial}{\partial m}\left[(4\pi \rho r^2) D_{\mathsf{mix}} \frac{\partial X_i}{\partial m}\right]$$

Star 2



The binary module



$$\frac{\partial r}{\partial m} = \frac{1}{4\pi r^2 \rho}$$

$$\frac{\partial P}{\partial m} = -\frac{Gm}{4\pi r^4}$$

$$\frac{\partial T}{\partial m} = -\frac{GmT}{4\pi r^4 P}\nabla$$

$$\frac{\partial l}{\partial m} = \varepsilon_{\text{nuc}} - c_P \frac{\partial T}{\partial t} - \frac{\delta}{\rho}\frac{\partial P}{\partial t}$$

$$\frac{\partial X_i}{\partial t} = m_i \left[\sum_j r_{ji} - \sum_k r_{ik}\right] + \frac{\partial}{\partial m}\left[(4\pi\rho r^2) D_{\text{mix}} \frac{\partial X_i}{\partial m}\right]$$

$$\frac{\partial r}{\partial m} = \frac{1}{4\pi r^2 \rho}$$

$$\frac{\partial P}{\partial m} = -\frac{Gm}{4\pi r^4}$$

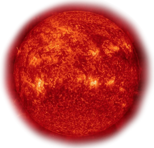
$$\frac{\partial T}{\partial m} = -\frac{GmT}{4\pi r^4 P}\nabla$$

$$\frac{\partial l}{\partial m} = \varepsilon_{\text{nuc}} - c_P \frac{\partial T}{\partial t} - \frac{\delta}{\rho}\frac{\partial P}{\partial t}$$

$$\frac{\partial X_i}{\partial t} = m_i \left[\sum_j r_{ji} - \sum_k r_{ik}\right] + \frac{\partial}{\partial m}\left[(4\pi\rho r^2) D_{\text{mix}} \frac{\partial X_i}{\partial m}\right]$$

The binary module

Star 1

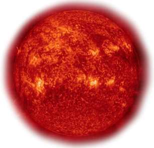


$$\frac{\partial r}{\partial m} = \frac{1}{4\pi r^2 \rho}$$
$$\frac{\partial P}{\partial m} = -\frac{Gm}{4\pi r^4}$$
$$\frac{\partial T}{\partial m} = -\frac{GmT}{4\pi r^4 P}\nabla$$
$$\frac{\partial l}{\partial m} = \varepsilon_{\rm nuc} - c_P \frac{\partial T}{\partial t} - \frac{\delta}{\rho}\frac{\partial P}{\partial t}$$
$$\frac{\partial X_i}{\partial t} = m_i \left[\sum_j r_{ji} - \sum_k r_{ik}\right] + \frac{\partial}{\partial m}\left[(4\pi\rho r^2) D_{\rm mix} \frac{\partial X_i}{\partial m}\right]$$

$$M_{1,\rm new} = M_{1,\rm old} + \Delta t \dot{M}_1$$
$$M_{2,\rm new} = M_{2,\rm old} + \Delta t \dot{M}_2$$
$$J_{\rm new} = J_{\rm old} + \Delta t \dot{J}$$

winds, MT

Star 2



$$\frac{\partial r}{\partial m} = \frac{1}{4\pi r^2 \rho}$$
$$\frac{\partial P}{\partial m} = -\frac{Gm}{4\pi r^4}$$
$$\frac{\partial T}{\partial m} = -\frac{GmT}{4\pi r^4 P}\nabla$$
$$\frac{\partial l}{\partial m} = \varepsilon_{\rm nuc} - c_P \frac{\partial T}{\partial t} - \frac{\delta}{\rho}\frac{\partial P}{\partial t}$$
$$\frac{\partial X_i}{\partial t} = m_i \left[\sum_j r_{ji} - \sum_k r_{ik}\right] + \frac{\partial}{\partial m}\left[(4\pi\rho r^2) D_{\rm mix} \frac{\partial X_i}{\partial m}\right]$$

$$\dot{J}_{\rm orb} = \dot{J}_{\rm ml} + \dot{J}_{\rm tides} + \dot{J}_{\rm GR} + \dot{J}_{\rm mb}$$

Orbit

The binary module

$$\frac{\partial r}{\partial m} = \frac{1}{4\pi r^2 \rho}$$

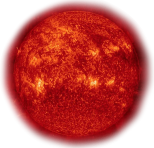
$$\frac{\partial P}{\partial m} = -\frac{Gm}{4\pi r^4}$$

$$\frac{\partial T}{\partial m} = -\frac{GmT}{4\pi r^4 P}\nabla$$

$$\frac{\partial l}{\partial m} = \varepsilon_{\text{nuc}} - c_P \frac{\partial T}{\partial t} - \frac{\delta}{\rho}\frac{\partial P}{\partial t}$$

$$\frac{\partial X_i}{\partial t} = m_i \left[\sum_j r_{ji} - \sum_k r_{ik}\right] + \frac{\partial}{\partial m}\left[(4\pi \rho r^2) D_{\text{mix}} \frac{\partial X_i}{\partial m}\right]$$

Star 1



$$M_{1,\text{new}} = M_{1,\text{old}} + \Delta t \dot{M}_1$$

$$M_{2,\text{new}} = M_{2,\text{old}} + \Delta t \dot{M}_2$$

$$J_{\text{new}} = J_{\text{old}} + \Delta t \dot{J}$$

Star 2
(point mass)

$$\frac{dM}{dt} = \ldots$$

$$\dot{J}_{\text{orb}} = \dot{J}_{\text{ml}} + \dot{J}_{\text{tides}} + \dot{J}_{\text{GR}} + \dot{J}_{\text{mb}}$$

Orbit

The binary module

The basic structure

```
# Start by copying the basic work folder into your preferred location

$ cp -r $MESA_DIR/binary/work template

$ cd template

# Display the content of the template folder with tree or ls -lh *

$ tree
```

```
.
├── clean
├── inlist
├── inlist1
├── inlist2
├── inlist_project
├── make
│   └── makefile
├── mk
├── re
├── rn
└── src
    ├── binary_run.f90
    ├── run_binary_extras.f90
    └── run_star_extras.f90

2 directories, 12 files
```

The binary module

The basic structure

```
# Start by copying the basic work folder into your preferred location

$ cp -r $MESA_DIR/binary/work template

$ cd template

# Display the content of the template folder with tree or ls -lh *

$ tree
```

```
.
├── clean
├── inlist
├── inlist1
├── inlist2
├── inlist_project
├── make
│   └── makefile
├── mk
├── re
├── rn
└── src
    ├── binary_run.f90
    ├── run_binary_extras.f90
    └── run_star_extras.f90

2 directories, 12 files
```

NEW **inlist_project,inlist1,inlist2**

The binary module

inlist_project

```
# Start by copying the basic work folder into your preferred location

$ cp -r $MESA_DIR/binary/work template

$ cd template

# Display the content of the template folder with tree or ls -lh *

$ tree

# Open inlist_project with your favorite editor
```

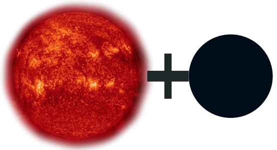
NEW **inlist_project**: contains controls for the binary, and the controls for the single stars are in `inlist1` and `inlist2`

inlist_project

```
&binary_job

  inlist_names(1) = 'inlist1'
  inlist_names(2) = 'inlist2'

  evolve_both_stars = .false.

/ ! end of binary_job namelist

&binary_controls

  m1 = 1.0d0  ! donor mass in Msun
  m2 = 1.4d0 ! companion mass in Msun
  initial_period_in_days = 2d0

  !transfer efficiency controls
  limit_retention_by_mdot_edd = .true.

  max_tries_to_achieve = 20

/ ! end of binary_controls namelist
```

The binary module

inlist_project



```
# Start by copying the basic work folder into your preferred location

$ cp -r $MESA_DIR/binary/work template

$ cd template

# Display the content of the template folder with tree or ls -lh *

$ tree

# Open inlist_project with your favorite editor
```

NEW **inlist_project**: contains controls for the binary, and the controls for the single stars are in `inlist1` and `inlist2`

inlist_project

```
&binary_job

  inlist_names(1) = 'inlist1'
  inlist_names(2) = 'inlist2'

  evolve_both_stars = .false.

/ ! end of binary_job namelist

&binary_controls

  m1 = 1.0d0  ! donor mass
  m2 = 1.4d0 ! companion mass in Msun
  initial_period_in_days = 2d0

  !transfer efficiency controls
  limit_retention_by_mdot_edd = .true.

  max_tries_to_achieve = 20

/ ! end of binary_controls namelist
```

No eccentricity

The binary module

inlist1

```
# Start by copying the basic work folder into your preferred location

$ cp -r $MESA_DIR/binary/work template

$ cd template

# Display the content of the template folder with tree or ls -lh *

$ tree

# Open inlist1 with your favorite editor
```

inlist1

```
&controls

      extra_terminal_output_file= 'log1'
      log_directory = 'LOGS1'

      ...

/ ! end of controls namelist
```

NEW **inlist1**: Nothing new here, but notice that the output folder is specified (and not hard coded, so you can choose it)

The binary module

Parameter libraries: $MESA_DIR/binary/defaults

$MESA_DIR/binary/defaults/**binary_job**.defaults

$MESA_DIR/binary/defaults/**binary_controls**.defaults

→ *Fortran namelists options*

The binary module

Parameter libraries: `$MESA_DIR/binary/defaults`

$MESA_DIR/binary/defaults/**binary_job**.defaults

$MESA_DIR/binary/defaults/**binary_controls**.defaults

→ *Fortran namelists options*

$MESA_DIR/binary/defaults/**pgbinary**.defaults

→ ***pgbinary*** 🤩

$MESA_DIR/binary/defaults/**binary_history_columns**.list

→ *Like in the star module*

The binary module

The basic structure

```
# Display the content of the template folder with tree or ls -lh *

$ tree
```

NEW **inlist_project**: contains controls for the binary, and the controls for the single stars are in inlist1 and inlist2

```
.
├── clean
├── inlist
├── inlist1
├── inlist2
├── inlist_project
├── make
│   └── makefile
├── mk
├── re
├── rn
└── src
    ├── binary_run.f90
    ├── run_binary_extras.f90
    └── run_star_extras.f90

2 directories, 12 files
```

The binary module

The basic structure

```
# Display the content of the template folder with tree or ls -lh *

$ tree
```

NEW **inlist_project**: contains controls for the binary, and the controls for the single stars are in `inlist1` and `inlist2`

NEW **run_binary_extras.f90**: similar functionality to `run_star_extras.f90`, you can include custom output, modified physics, termination conditions, ecc.

```
.
├── clean
├── inlist
├── inlist1
├── inlist2
├── inlist_project
├── make
│   └── makefile
├── mk
├── re
├── rn
└── src
    ├── binary_run.f90
    ├── run_binary_extras.f90
    └── run_star_extras.f90

2 directories, 12 files
```

The binary module

run_binary_extras.f90

```
# Open ./src/run_binary_extras.f90 with your favorite editor
```

```
subroutine data_for_extra_binary_history_columns(binary_id, n, names, vals, ierr)
        type (binary_info), pointer :: b
        integer, intent(in) :: binary_id
        integer, intent(in) :: n
        character (len=maxlen_binary_history_column_name) :: names(n)
        real(dp) :: vals(n)
        integer, intent(out) :: ierr
        ierr = 0
        call binary_ptr(binary_id, b, ierr)
        if (ierr /= 0) then
           write(*,*) 'failed in binary_ptr'
           return
        end if

end subroutine data_for_extra_binary_history_columns
```

NEW run_binary_extras.f90

The binary module

How to add more output columns?

1.
```
vals(1) = …
names(1) = …
```

2. Uncomment lines in a local copy of **binary_history_columns.list**

```
# Open ./src/run_binary_extras.f90 with your favorite editor
```

```
subroutine data_for_extra_binary_history_columns(binary_id, n, names, vals, ierr)
        type (binary_info), pointer :: b
        integer, intent(in) :: binary_id
        integer, intent(in) :: n
        character (len=maxlen_binary_history_column_name) :: names(n)
        real(dp) :: vals(n)
        integer, intent(out) :: ierr
        ierr = 0
        call binary_ptr(binary_id, b, ierr)
        if (ierr /= 0) then
           write(*,*) 'failed in binary_ptr'
           return
        end if

end subroutine data_for_extra_binary_history_columns
```

NEW **run_binary_extras.f90**

The binary module

run_binary_extras.f90

```
# Open ./src/run_binary_extras.f90 with your favorite editor
```

```fortran
subroutine data_for_extra_binary_history_columns(binary_id, n, names, vals, ierr)
      type (binary_info), pointer :: b
      integer, intent(in) :: binary_id
      integer, intent(in) :: n
      character (len=maxlen_binary_history_column_name) :: names(n)
      real(dp) :: vals(n)
      integer, intent(out) :: ierr
      ierr = 0
      call binary_ptr(binary_id, b, ierr)
      if (ierr /= 0) then
         write(*,*) 'failed in binary_ptr'
         return
      end if

end subroutine data_for_extra_binary_history_columns
```

NEW run_binary_extras.f90

The binary module

The binary_info type

Analog to the star_info type with information on a stellar model, there is a **binary_info** type b with information on the binary system (e.g. orbital period and masses).

Information contained within this type are in

$MESA_DIR/binary/public/binary_data.inc

```
!!! SOME EXAMPLES !!!

b% mtransfer_rate

! The star_info instances for each component
b% s1
b% s2

! Mass of each component in grams
b% m(1)
b% m(2)

! Analog to single star xtra array
b% xtra(:)
```

The `binary` module

Output

```
# Compile and run the template directory
$ ./mk
$ ./rn | tee out.txt
# Kill the run after ~50 models pressing ctrl+C
```

The binary module

Output

```
# Compile and run the template directory
$ ./mk
$ ./rn | tee out.txt
# Kill the run after ~50 models pressing ctrl+C
```

Photos are saved also for the binary! To restart:

```
$ ./re x050 | tee outre.txt
```

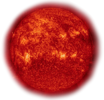
```
.
├── binary
├── binary_history.data
├── clean
├── inlist
├── inlist1
├── inlist2
├── inlist_project
├── log1
├── LOGS1
│   ├── history.data
│   ├── pgstar.dat
│   ├── profile1.data
│   ├── profile2.data
│   └── profiles.index
├── make
│   ├── …
│   └── run_star_extras.smod
├── mk
├── photos
│   ├── 1_x050
│   └── b_x050
├── re
├── rn
└── src
    ├── …
    └── run_star_extras.f90

4 directories, 31 files
```

Successful compilation

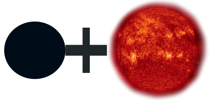
The binary module

Output

```
# Compile and run the template directory
$ ./mk
$ ./rn | tee out.txt
# Kill the run after ~50 models pressing ctrl+C
```



Output for the resolved star



Output for the binary

similar format as history.data

```
.
├── binary
├── binary_history.data
├── clean
├── inlist
├── inlist1
├── inlist2
├── inlist_project
├── log1
├── LOGS1
│   ├── history.data
│   ├── pgstar.dat
│   ├── profile1.data
│   ├── profile2.data
│   └── profiles.index
├── make
│   ├── …
│   └── run_star_extras.smod
├── mk
├── photos
│   ├── 1_x050
│   └── b_x050
├── re
├── rn
└── src
    ├── …
    └── run_star_extras.f90

4 directories, 31 files
```

The `binary` module

Terminal output

```
       step    lg_Tmax     Teff     lg_LH      lg_Lnuc     Mass      H_rich     H_cntr     N_cntr     Y_surf    eta_cntr   zones  retry
  lg_dt_yrs   lg_Tcntr     lg_R     lg_L3a     lg_Lneu     lg_Mdot   He_core    He_cntr    O_cntr     Z_surf    gam_cntr   iters
    age_yrs   lg_Dcntr     lg_L     lg_LZ      lg_Lphoto   lg_Dsurf  CO_core    C_cntr     Ne_cntr    Z_cntr    v_div_cs       dt_limit

         50   7.147732  5672.534  -0.102050  -0.102050   1.000000   1.000000   0.597770   0.005053   0.280000  -1.648242     787      0
 7.8132E+00   7.147732 -0.035889 -45.817583  -1.761971 -99.000000   0.000000   0.381663   0.009335   0.020000   0.093672       5
 1.3492E+09   1.984456 -0.101975 -15.954590 -99.000000  -6.736023   0.000000   0.000016   0.002085   0.020566  0.000E+00          b_jorb

binary_step      M1+M2      separ       Porb          e      M2/M1       pm_i    donor_i    dot_Mmt        eff       Jorb      dot_J    dot_Jmb
      lg_dt         M1         R1         P1      dot_e      vorb1        RL1    Rl_gap1     dot_M1   dot_Medd      spin1    dot_Jgr    dot_Jls
     age_yr         M2         R2         P2       Eorb      vorb2        RL2    Rl_gap2     dot_M2      L_acc      spin2    dot_Jml  rlo_iters

bin      50   2.400000   8.616442   1.891166  0.000E+00   1.400000          2          1  0.000E+00   1.000000  1.603E+52 -8.038E+33 -7.782E+33
   7.813209   1.000000   0.920686   0.000000  0.000E+00 134.463138   3.017402 -6.949E-01  0.000E+00  6.357E-08  0.000E+00 -2.558E+32  0.000E+00
 1.3492E+09   1.400000   0.000000   0.000000 -3.082E+47  96.045099   3.518571 -1.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00          1

save LOGS1/profile2.data for model 50
 save photos/b_x050, photos/1_x050 for model          50
```

For the case of both evolved stars: see later ;)

Exercise 1: Jdot equation

$$\dot{J}_{\rm orb} = \dot{J}_{\rm ml} + \dot{J}_{\rm tides} + \dot{J}_{\rm GR} + \dot{J}_{\rm mb}$$

Exercise 1: Jdot equation

I. *Angular momentum loss from gravitational radiation*

$$\dot{J}_{\rm gr} = \frac{32}{5c^5}\left(\frac{2\pi G}{P_{\rm orb}}\right)^{7/3}\frac{(M_1M_2)^2}{(M_1+M_2)^{2/3}}$$

Exercise 1: Jdot equation

I. *Angular momentum loss from gravitational radiation*

$$\dot{J}_{\rm gr} = \frac{32}{5c^5}\left(\frac{2\pi G}{P_{\rm orb}}\right)^{7/3}\frac{(M_1M_2)^2}{(M_1+M_2)^{2/3}}$$

Bigger for **more massive** binaries

Bigger for **closer** binaries

Exercise 1: Jdot equation

I. *Angular momentum loss from gravitational radiation*

$$\dot{J}_{\mathrm{gr}} = \frac{32}{5c^5}\left(\frac{2\pi G}{P_{\mathrm{orb}}}\right)^{7/3}\frac{(M_1 M_2)^2}{(M_1+M_2)^{2/3}}$$

```
$ cd $MESA_DIR/binary

$ grep -nri do_jdot_gr
```

Exercise 1: Jdot equation

I. Angular momentum loss from gravitational radiation

$$\dot{J}_{\rm gr} = \frac{32}{5c^5}\left(\frac{2\pi G}{P_{\rm orb}}\right)^{7/3}\frac{(M_1 M_2)^2}{(M_1+M_2)^{2/3}}$$

```
$ cd $MESA_DIR/binary

$ grep -nri do_jdot_gr
```

```
defaults/binary_controls.defaults:! do_jdot_gr
defaults/binary_controls.defaults:    do_jdot_gr = .true.
private/binary_ctrls_io.f90:         do_jdot_gr, &
private/binary_ctrls_io.f90:         b% do_jdot_gr = do_jdot_gr
private/binary_ctrls_io.f90:         do_jdot_gr = b% do_jdot_gr
private/binary_jdot.f90:         if (.not. b% do_jdot_gr) then
test_suite/jdot_ml_check/inlist_project:   do_jdot_gr = .false.
Binary file make/libbinary.a matches
Binary file make/binary_ctrls_io.o matches
public/binary_controls.inc:logical :: do_jdot_gr
```

Exercise 1: Jdot equation

I. *Angular momentum loss from gravitational radiation*

$$\dot{J}_{\rm gr} = \frac{32}{5c^5}\left(\frac{2\pi G}{P_{\rm orb}}\right)^{7/3}\frac{(M_1 M_2)^2}{(M_1+M_2)^{2/3}}$$

```
$ cd $MESA_DIR/binary

$ grep -nri do_jdot_gr

# Open the interesting file with your favorite text editor

$ less ./private/binary_jdot.f90
```

Exercise 1: Jdot equation

I. *Angular momentum loss from gravitational radiation*

$$\dot{J}_{\rm gr} = \frac{32}{5c^5}\left(\frac{2\pi G}{P_{\rm orb}}\right)^{7/3}\frac{(M_1 M_2)^2}{(M_1+M_2)^{2/3}}$$

./private/binary_jdot.f90

```
real(dp) function get_jdot(b)
         …
            ! calculate jdot from gravitational wave radiation
            if (.not. b% do_jdot_gr) then
                  b% jdot_gr = 0d0
            else if (.not. b% use_other_jdot_gr) then
                  call default_jdot_gr(b% binary_id, ierr)
         …
            end if
         …
end function get_jdot
```

Exercise 1: Jdot equation

I. *Angular momentum loss from gravitational radiation*

$$\dot{J}_{\rm gr} = \frac{32}{5c^5}\left(\frac{2\pi G}{P_{\rm orb}}\right)^{7/3}\frac{(M_1 M_2)^2}{(M_1+M_2)^{2/3}}$$

```
real(dp) function get_jdot(b)
      …
        ! calculate jdot from gravitational wav
        if (.not. b% do_jdot_gr) then
            b% jdot_gr = 0d0
        else if (.not. b% use_other_jdot_gr) the
            call default_jdot_gr(b% binary_id,
      …
        end if
      …
end function get_jdot
```

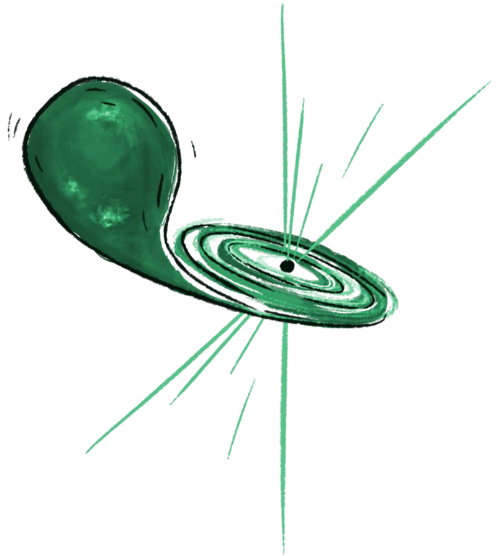
./private/binary_jdot.f90

```
      subroutine default_jdot_gr(binary_id, ierr)
       …
         bs4 = pow4(b% separation)
         clight5 = pow5(clight)
         cgrav3 =
standard_cgrav*standard_cgrav*standard_cgrav
         b% jdot_gr = -32d0 * cgrav3 * b% m(b% a_i) * b%
m(b% d_i) * (b% m(b% a_i) + b% m(b% d_i)) / &
             (5d0 * clight5 * bs4) * b% angular_momentum_j

      end subroutine default_jdot_gr
```

Exercise 1: Jdot equation

II. Eddington accretion limit

$$\dot{M}_{\rm Edd} \equiv \frac{4\pi G M_{\rm BH}}{\kappa c \eta}\,, \quad \eta \equiv 1 - \sqrt{1-(M_{\rm BH}/M_{\rm BH,0})^2}$$

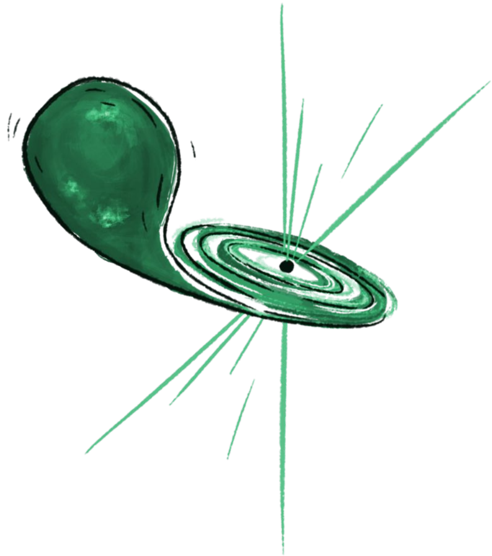


Exercise 1: Jdot equation

II. Eddington accretion limit

$$\dot{M}_{\rm Edd} \equiv \frac{4\pi G M_{\rm BH}}{\kappa c \eta}\,, \quad \eta \equiv 1 - \sqrt{1 - (M_{\rm BH}/M_{\rm BH,0})^2}$$

```
&binary_controls

  m1 = 1.0d0  ! donor mass in Msun
  m2 = 1.4d0 ! companion mass in Msun
  initial_period_in_days = 2d0

  !transfer efficiency controls
  limit_retention_by_mdot_edd = .true.

/ ! end of binary_controls namelist
```



Exercise 1: Jdot equation

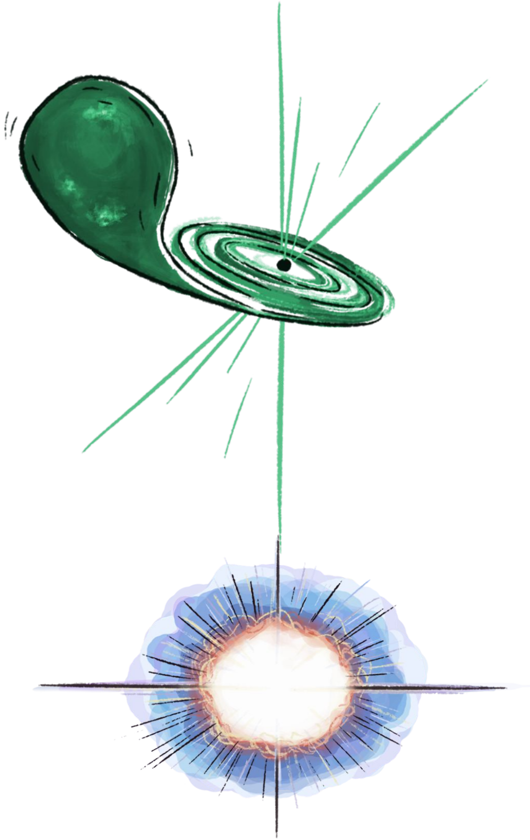
II. Eddington accretion limit

$$\dot{M}_{\rm Edd} \equiv \frac{4\pi G M_{\rm BH}}{\kappa c \eta}, \quad \eta \equiv 1 - \sqrt{1 - (M_{\rm BH}/M_{\rm BH,0})^2}$$

III. Mass loss

$$\dot{J}_{\rm ml} = ?$$

$$\dot{J}_{\rm orb} = \dot{J}_{\rm ml} + \dot{J}_{\rm tides} + \dot{J}_{\rm gr} + \dot{J}_{\rm mb}$$



Exercise 1: Jdot equation

II. Eddington accretion limit

$$\dot{M}_{\rm Edd} \equiv \frac{4\pi G M_{\rm BH}}{\kappa c \eta}, \quad \eta \equiv 1 - \sqrt{1 - (M_{\rm BH}/M_{\rm BH,0})^2}$$

III. Mass loss

$$\dot{J}_{\rm ml} = ?$$

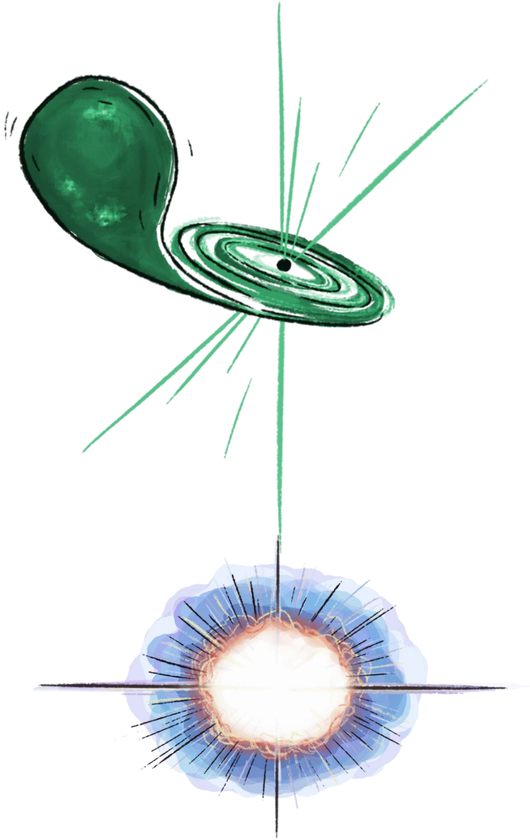
Leakages of momentum

Accretor

Winds

Donor

Circumbinary



Exercise 1: Try yourself!

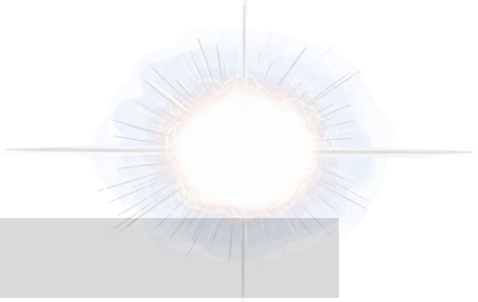
II. Eddington accretion limit

$$\dot{M}_{\mathrm{Edd}} \equiv \frac{4\pi G M_{\mathrm{BH}}}{\kappa c \eta}, \quad \eta \equiv 1 - \sqrt{1 - (M_{\mathrm{BH}}/M_{\mathrm{BH},0})^2}$$


```
$ grep -nri limit_retention_by_mdot_edd
```

III. Mass loss

$$\dot{J}_{\mathrm{ml}} = ?$$



```
$ grep -nri do_jdot_ml
```

Exercise 1: Results

II. Eddington accretion limit

$$\dot{M}_{\rm Edd} \equiv \frac{4\pi G M_{\rm BH}}{\kappa c \eta}, \quad \eta \equiv 1 - \sqrt{1 - (M_{\rm BH}/M_{\rm BH,0})^2}$$

```
subroutine eval_mdot_edd(binary_id, mdot_edd, mdot_edd_eta, ierr)
      …
        if (b% use_this_for_mdot_edd > 0) then
           mdot_edd = b% use_this_for_mdot_edd*(Msun/secyer)
        else
           ! eg., eq. (9) of Podsiadlowski, Rappaport & Han 2003, MNRAS, 341, 385
           if (.not. b% use_es_opacity_for_mdot_edd) then
              mdot_edd = pi4*standard_cgrav*b% m(b% a_i) &
                 /(clight*b% s_donor% opacity(1)*mdot_edd_eta)
            …
           end if
        end if
      …
end subroutine eval_mdot_edd
```

./private/binary_mdot.f90

Exe

II.

./private/binary_jdot.f90

```
subroutine default_jdot_ml(binary_id, ierr)
…
        !mass lost from vicinity of donor
        b% jdot_ml = (b% mdot_system_transfer(b% d_i) + b% mdot_system_wind(b% d_i))*&
            pow2(b% m(b% a_i)/(b% m(b% a_i)+b% m(b% d_i))*b% separation)*2*pi/b% period *&
            sqrt(1 - pow2(b% eccentricity))
        !mass lost from vicinity of accretor
        b% jdot_ml = b% jdot_ml + (b% mdot_system_transfer(b% a_i) + b% mdot_system_wind(b% a_i))*&
            pow2(b% m(b% d_i)/(b% m(b% a_i)+b% m(b% d_i))*b% separation)*2*pi/b% period *&
            sqrt(1 - pow2(b% eccentricity))
        !mass lost from circumbinary coplanar toroid
        b% jdot_ml = b% jdot_ml + b% mdot_system_cct * b% mass_transfer_gamma * &
            sqrt(standard_cgrav * (b% m(1) + b% m(2)) * b% separation)
    end subroutine default_jdot_ml
```

III. *Mass loss*

$$\dot{J}_{\rm ml} = \left[(\dot{M}_{1,\rm w} + \alpha\dot{M}_{\rm RLOF})M_2^2 + (\dot{M}_{2,\rm w} + \beta\dot{M}_{\rm RLOF})M_1^2\right] \times \frac{a^2}{(M_1+M_2)^2}\frac{2\pi}{P_{\rm orb}} + \gamma\delta\dot{M}_{\rm RLOF}\sqrt{G(M_1+M_2)a}$$

```
$ grep -nri do_jdot_ml
```

Exe

II.

```
subroutine default_jdot_ml(binary_id, ierr)
…
        !mass lost from vicinity of donor
        b% jdot_ml = (b% mdot_system_transfer(b% d_i) + b% mdot_system_wind(b% d_i))*&
            pow2(b% m(b% a_i)/(b% m(b% a_i)+b% m(b% d_i))*b% separation)*2*pi/b% period *&
            sqrt(1 - pow2(b% eccentricity))
        !mass lost from vicinity of accretor
        b% jdot_ml = b% jdot_ml + (b% mdot_system_transfer(b% a_i) + b% mdot_system_wind(b% a_i))*&
            pow2(b% m(b% d_i)/(b% m(b% a_i)+b% m(b% d_i))*b% separation)*2*pi/b% period *&
            sqrt(1 - pow2(b% eccentricity))
        !mass lost from circumbinary coplanar toroid
        b% jdot_ml = b% jdot_ml + b% mdot_system_cct * b% mass_transfer_gamma * &
            sqrt(standard_cgrav * (b% m(1) + b% m(2)) * b% separation)
      end subroutine default_jdot_ml
```

./private/binary_jdot.f90

III. Mass loss

$$\dot{J}_{\rm ml} = \left[(\dot{M}_{1,{\rm w}} + \alpha \dot{M}_{\rm RLOF}) M_2^2 + (\dot{M}_{2,{\rm w}} + \beta \dot{M}_{\rm RLOF}) M_1^2 \right] \times \frac{a^2}{(M_1+M_2)^2} \frac{2\pi}{P_{\rm orb}} + \gamma\delta \dot{M}_{\rm RLOF} \sqrt{G(M_1+M_2)a}$$

winds

```
$ grep -nri do_jdot_ml
```

./private/binary_jdot.f90

```
subroutine default_jdot_ml(binary_id, ierr)
…
         !mass lost from vicinity of donor
         b% jdot_ml = (b% mdot_system_transfer(b% d_i) + b% mdot_system_wind(b% d_i))*&
             pow2(b% m(b% a_i)/(b% m(b% a_i)+b% m(b% d_i))*b% separation)*2*pi/b% period *&
             sqrt(1 - pow2(b% eccentricity))
         !mass lost from vicinity of accretor
         b% jdot_ml = b% jdot_ml + (b% mdot_system_transfer(b% a_i) + b% mdot_system_wind(b% a_i))*&
             pow2(b% m(b% d_i)/(b% m(b% a_i)+b% m(b% d_i))*b% separation)*2*pi/b% period *&
             sqrt(1 - pow2(b% eccentricity))
         !mass lost from circumbinary coplanar toroid
         b% jdot_ml = b% jdot_ml + b% mdot_system_cct * b% mass_transfer_gamma * &
             sqrt(standard_cgrav * (b% m(1) + b% m(2)) * b% separation)
      end subroutine default_jdot_ml
```

Exe

II.

III. *Mass loss*

$$\dot{J}_{\rm ml} = \left[(\dot{M}_{1,\rm w} + \alpha\dot{M}_{\rm RLOF})M_2^2 + (\dot{M}_{2,\rm w} + \beta\dot{M}_{\rm RLOF})M_1^2\right] \times \frac{a^2}{(M_1+M_2)^2}\frac{2\pi}{P_{\rm orb}} + \gamma\delta\dot{M}_{\rm RLOF}\sqrt{G(M_1+M_2)a}$$

$$\epsilon = 1 - \beta - \alpha - \delta$$

MT efficiency

Minilabs of today!

```
$ grep -nri do_jdot_ml
```

In general:

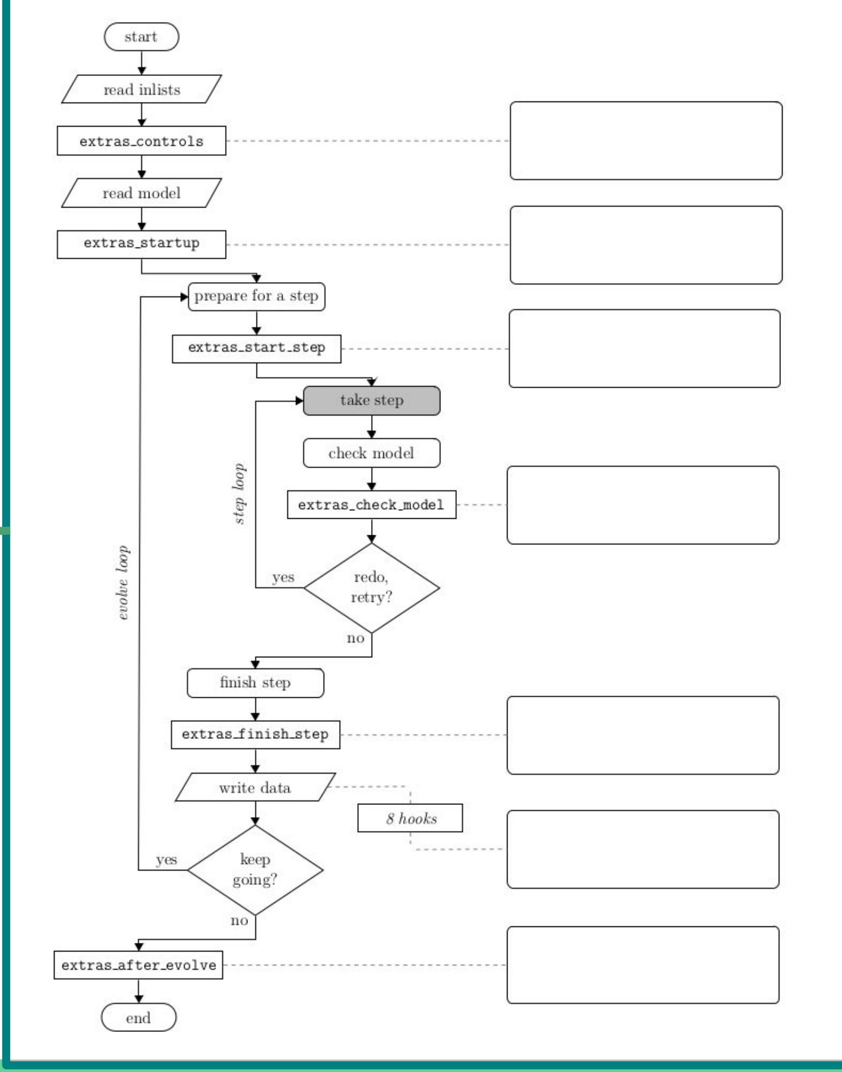
BLACK BOX ≠ MESA

😱 Don't be scared and get to know the code

—> you might help finding bugs!!
—> you might want to expand with your favorite piece of physics!!

Exercise 2: The `binary` flow

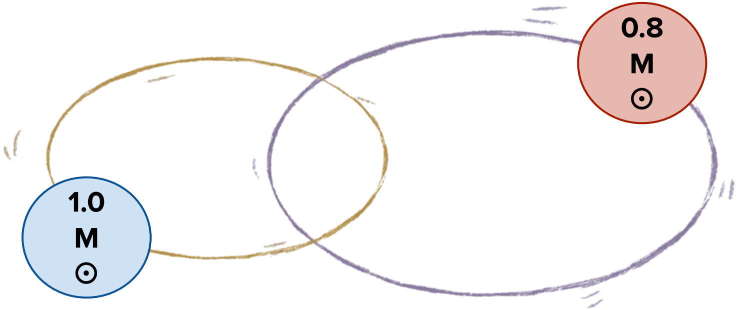
`star` module flow



Exercise 2: The `binary` flow

1.

```
# Copy the test suite folder somewhere
$ cp -r $MESA_DIR/binary/test_suite/evolve_both_stars .
$ cd evolve_both_stars
# Open the inlist_project with your favorite editor and inspect
```



$$P_{\rm orb} = 0.5 \text{ days}$$

Exercise 2: The `binary` flow

I.

```
# Copy the test suite folder somewhere
$ cp -r $MESA_DIR/binary/test_suite/evolve_both_stars .
$ cd evolve_both_stars
# Open the inlist_project with your favorite editor and inspect
```

II. Modify the `inlist1` for the donor star to have `max_model_number = 2`

./inlist1

```
&controls
…
    max_model_number = 2
/ ! end of controls namelist
```

Exercise 2: The `binary` flow

III. Print a sentence inside every routine in `run_binary_extras.f90` whose name starts with `extras_binary_` (but leave out `extras_binary_controls`):

./src/run_binary_extras.f90

```
integer function extras_binary_startup(binary_id,restart,ierr)
        …
        write(*,*) "BINARY - extras_binary_startup"
    end function extras_binary_startup
```

IV. Do the same in `run_star_extras.f90` for those whose name starts with `extras_` (but leave out `extras_controls`), specifying also the mass of the star with `s% m(1)/Msun`:

./src/run_star_extras.f90

```
    subroutine extras_startup(id, restart, ierr)
        …
        write(*,*) "STAR - extras_startup, STAR mass=", s% m(1)/Msun, " Msun"
    end subroutine extras_startup
```

Exercise 2: The `binary` flow

V. **CAVEAT** in `run_star_extras.f90`: Make sure to load the pointer s in the `extras_check_model` routine, otherwise you can't access s% m(1)

./src/run_star_extras.f90

```
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
        write(*,*) "STAR - extras_check_model, STAR mass=", s% m(1)/Msun, " Msun"

    end function extras_check_model
```

Exercise 2: Try!

I.

```
# Copy the test suite folder somewhere
$ cp -r $MESA_DIR/binary/test_suite/evolve_both_stars .
$ cd evolve_both_stars
# Open the inlist_project with your favorite editor and inspect
```

II. Modify the `inlist1` for the donor star to have `max_model_number = 2`

III. Print a sentence inside every routine in run_binary_extras.f90 whose name starts with extras_binary_ (but leave out extras_binary_controls)

IV. Do the same in run_star_extras.f90 for those whose name starts with extras_ (but leave out extras_controls), specifying also the mass of the star with `s% m(1)/Msun`

V. **CAVEAT** in `run_star_extras.f90`: Make sure to load the pointer s in the `extras_check_model` routine, otherwise you can't access `s% m(1)`

Exercise 2: Result pt.1

```
run
DATE: 2024-06-13
TIME: 17:41:56
...
STAR - extras_startup, STAR mass=   1.0000000000000000       Msun
STAR - extras_startup, STAR mass=  0.80000000000000004       Msun
BINARY - extras_binary_startup
...
STAR - extras_start_step, STAR mass=   1.0000000000000000       Msun
STAR - extras_start_step, STAR mass=  0.80000000000000004       Msun
BINARY - extras_binary_start_step


STAR - extras_check_model, STAR mass=   1.0000000000000000       Msun
STAR - extras_check_model, STAR mass=  0.80000000000000004       Msun
BINARY - extras_binary_check_model
...
```

Exercise 2: Result pt.1

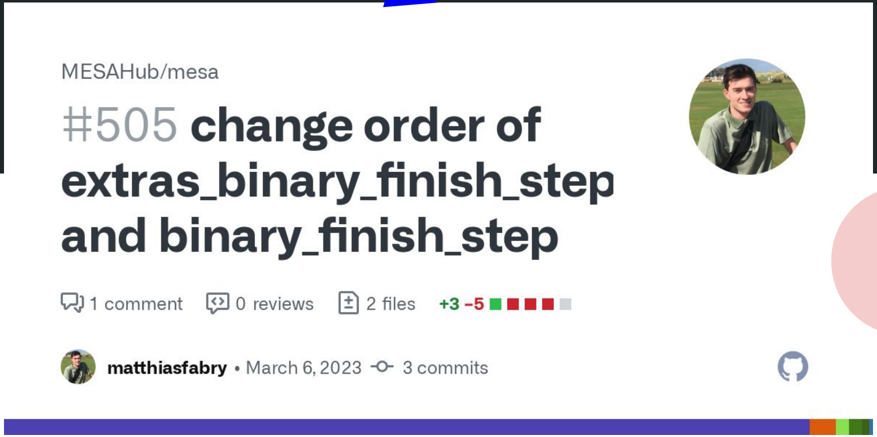
```
run
DATE: 2024-06-13
TIME: 17:41:56
...
STAR - extras_startup, STAR mass=   1.0000000000000000       Msun
STAR - extras_startup, STAR mass=  0.80000000000000004       Msun
BINARY - extras_binary_startup
...
STAR - extras_start_step, STAR mass=   1.0000000000000000       Msun
STAR - extras_start_step, STAR mass=  0.80000000000000004       Msun
BINARY - extras_binary_start_step
⭐MT⭐
STAR - extras_check_model, STAR mass=   1.0000000000000000       Msun
STAR - extras_check_model, STAR mass=  0.80000000000000004       Msun
BINARY - extras_binary_check_model
...
```

Exercise 2: Result pt.2

```
BINARY - extras_binary_finish_step
STAR - extras_finish_step, STAR mass=   1.0000000000000000       Msun
STAR - extras_finish_step, STAR mass=  0.80000000000000004       Msun
...
BINARY - extras_binary_after_evolve
STAR - extras_after_evolve, STAR mass=   1.0000000000000000       Msun
STAR - extras_after_evolve, STAR mass=  0.80000000000000004       Msun
...
DATE: 2024-06-13
TIME: 17:42:05
finished
```

Exercise 2: Result pt.2

```
BINARY - extras_binary_finish_step
STAR - extras_finish_step, STAR mass=   1.0000000000000000       Msun
STAR - extras_finish_step, STAR mass=  0.80000000000000004       Msun
...
BINARY - extras_binary_after_evolve
STAR - extras_after_evolve, STAR mass=   1.0000000000000000       Msun
STAR - extras_after_evolve, STAR mass=  0.80000000000000004       Msun
...
DATE: 2024-06-13
TIME: 17:42:05
finished
```



The code evolves :)

Exercise 2: Result pt.2

```
BINARY - extras_binary_finish_step
STAR - extras_finish_step, STAR mass=   1.0000000000000000      Msun
STAR - extras_finish_step, STAR mass=  0.80000000000000004      Msun
...
BINARY - extras_binary_after_evolve
STAR - extras_after_evolve, STAR mass=   1.0000000000000000      Msun
STAR - extras_after_evolve, STAR mass=  0.80000000000000004      Msun
...
DATE: 2024-06-13
TIME: 17:42:05
finished
```

What about `data_for_extra*_history_columns` and `data_for_extra*_profile_columns`?

Exercise 2: Result pt.2

```
BINARY - extras_binary_finish_step
STAR - extras_finish_step, STAR mass=   1
STAR - extras_finish_step, STAR mass=  0.
...
BINARY - extras_binary_after_evolve
STAR - extras_after_evolve, STAR mass=
STAR - extras_after_evolve, STAR mass=
...
DATE: 2024-06-13
TIME: 17:42:05
finished
```

```fortran
    ! try extras
    if (associated(b% how_many_extra_binary_history_columns) .and. &
       associated(b% data_for_extra_binary_history_columns)) then
       num_extra_cols = b% how_many_extra_binary_history_columns(b % binary_id)
       if (num_extra_cols > 0) then
          allocate(&
             extra_col_names(num_extra_cols), &
             extra_col_vals(num_extra_cols), stat = ierr)
          call b% data_for_extra_binary_history_columns( &
             b % binary_id, num_extra_cols, extra_col_names, extra_col_vals,
ierr)
          do i = 1, num_extra_cols
             if (extra_col_names(i) == name) then
                val = extra_col_vals(i)
                get1_binary_hist_value = .true.
                exit
             end if
          end do
          deallocate(extra_col_names, extra_col_vals)
          if (get1_binary_hist_value) return
       end if
    end if
```

binary/private/run_binary_support.f90

What about data_for_extra*_history_columns and data_for_extra*_profile_columns?

Exercise 2: Result pt.2

```
BINARY - extras_binary_finish_step
STAR - extras_finish_step, STAR mass=   1.0000000000000000       Msun
STAR - extras_finish_step, STAR mass=  0.80000000000000004       Msun
...
BINARY - extras_binary_after_evolve
STAR - extras_after_evolve, STAR mass=   1.0000000000000000       Msun
STAR - extras_after_evolve, STAR mass=  0.80000000000000004       Msun
...
DATE: 2024-06-13
TIME: 17:42:05
finished
```

When you build your `run_*extras.f90` with custom routine (my_jdot?), do this exercise again :)

Exercise 2: Home 🙃

You can try to complete your `binary` flow diagram

I.e., print a sentence also in `data_for_extra*_history_columns` and `data_for_extra*_profile_columns`

🚨 You will have to save custom quantities in the `vals` and names arrays have fun with binaries ⭐⭐