BWD AGWB

Generated by Doxygen 1.10.0

Chapter 1

Namespace Index

1.1 Package List

Here are the packages with brief descriptions (if available):

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 Create_z_at_age Namespace Reference

Functions

- None main ()
  Function to create list of z-values at given ages of the Universe.

4.1.1 Function Documentation

4.1.1.1 main()

```
None Create_z_at_age.main ( )
```

Function to create list of z-values at given ages of the Universe.

4.2 GWB Namespace Reference

Functions

- main ()
  Main function.

Variables

- action
- category
- size
- titlesize
- labelsize
- fontsize
- tuple s_in_Myr = (u.Myr).to(u.s)

4.2.1 Function Documentation

4.2.1.1 main()

```
GWB.main ( )
```

Main function.

The main functions sets the details of the simulation and runs the three main parts of the program.

4.2.2 Variable Documentation

4.2.2.1 action

```
GWB.action
```

4.2.2.2 category

```
GWB.category
```

4.2.2.3 fontsize

```
GWB.fontsize
```

4.2.2.4 labelsize

```
GWB.labelsize
```

4.2.2.5 s_in_Myr

```
tuple GWB.s_in_Myr = (u.Myr).to(u.s)
```

4.2.2.6 size

```
GWB.size
```

4.2.2.7 titlesize

```
GWB.titlesize
```

4.3 modules Namespace Reference

Namespaces

- namespace add_birth
- namespace add_bulk
- namespace add_merge
- namespace auxiliary
- namespace physics
- namespace RedshiftInterpolator
- namespace SFH
- namespace SimModel

4.3.1 Detailed Description

4.3.2 Modules

This package contains all the modules that are used in the project.

1. add_birth.py: function to add birth bins to the GWB
2. add_bulk.py: function to add bulk part of the GWB
3. add_merge.py: function to add merger bins to the GWB
4. auxiliary.py: auxiliary functions used in the project
5. physics.py: physics functions that are used in SeBa_pre_process.py
6. RedshiftInterpolator.py: class to interpolate redshifts
7. SFH.py: contains all the SFH functions
8. SimModel.py: class that stores all the important parameters of the simulation

4.4 modules.add_birth Namespace Reference

Functions

- None add_birth (sm.SimModel model, pd.DataFrame data, ri.RedshiftInterpolator z_interp, str tag)
  This routine adds the contribution of the 'birth bins' to the bulk GWB.

4.4.1 Function Documentation

4.4.1.1 add_birth()

```
None modules.add_birth.add_birth (
            sm.SimModel model,
            pd.DataFrame data,
            ri.RedshiftInterpolator z_interp,
            str tag )
```

This routine adds the contribution of the 'birth bins' to the bulk GWB.

Parameters

model	instance of SimModel, containing the necessary information for the run.
z_interp	instance of RedshiftInterpolator, used in the SFH calculations.
data	dataframe containing the binary population data.
tag	tag to add to the output files.

Returns

Saves a dataframe that contains the GWB at all freqyencies, and a dataframe that has the breakdown for the different redshift bins.

4.5 modules.add_bulk Namespace Reference

Functions

- None add_bulk (sm.SimModel model, pd.DataFrame data, ri.RedshiftInterpolator z_interp, str tag)

  This routine calculates the majority of the GWB, what is referred to in my thesis as the 'generic case'.

4.5.1 Function Documentation

4.5.1.1 add_bulk()

```
None modules.add_bulk.add_bulk (
            sm.SimModel model,
            pd.DataFrame data,
            ri.RedshiftInterpolator z_interp,
            str tag )
```

This routine calculates the majority of the GWB, what is referred to in my thesis as the 'generic case'.

Parameters

model	instance of SimModel, containing the necessary information for the run.
z_interp	instance of RedshiftInterpolator, used in the SFH calculations.
data	dataframe containing the binary population data.
tag	tag to add to the output files.

Returns

Saves a dataframe that contains the GWB at all freqyencies, and a dataframe that has the breakdown for the different redshift bins.

4.6 modules.add_merge Namespace Reference

Functions

- None add_merge (sm.SimModel model, pd.DataFrame data, ri.RedshiftInterpolator z_interp, str tag)

This routine adds the contribution of the 'merger bins' due to Kepler max to the bulk+birth GWB.

4.6.1 Function Documentation

4.6.1.1 add_merge()

```
None modules.add_merge.add_merge (
            sm.SimModel model,
            pd.DataFrame data,
            ri.RedshiftInterpolator z_interp,
            str tag )
```

This routine adds the contribution of the 'merger bins' due to Kepler max to the bulk+birth GWB.

Parameters

model	instance of SimModel, containing the necessary information for the run.
z_interp	instance of RedshiftInterpolator, used in the SFH calculations.
data	dataframe containing the binary population data.
tag	tag to add to the output files.

Returns

Saves a dataframe that contains the GWB at all freqyencies, and a dataframe that has the breakdown for the different redshift bins.

4.7 modules.auxiliary Namespace Reference

Functions

- tuple calc_parabola_vertex (float x1, float y1, float x2, float y2, float x3, float y3)

  Calculate the coefficients of a parabola given three points.
- float parabola (float x, float a, float b, float c)

  Calculate the value of a parabola given the coefficients.
- np.array get_bin_factors (np.array freqs, np.array bins)

  Determine bin factors that often recur in the calculation to store them.
- np.array get_width_z_shell_from_z (np.array z_vals)

  Returns the widths of the redshift shells in Mpc.
- np.array Omega (float Omega_ref, float f_ref, np.array freq)

  Create a f^{2/3} spectrum line.
- None make_Omega_plot_unnorm (np.array f, np.array Omega_sim, bool save=False, str save_name="void", bool show=False)

  Make a plot showing Omega for BWD.
- float tau_syst (float f_0, float f_1, float K)

  Calculates tau, the time it takes a binary with K to evolve from f_0 to f_1 (GW frequencies).
- float determine_upper_freq (float nu_low, float evolve_time, float K, bool DEBUG=False)

  Determines upper ORBITAL frequency for a binary with K, starting from nu_0, evolving over evolve_time.

Variables

- tuple s_in_Myr = (u.Myr).to(u.s)

4.7.1 Function Documentation

4.7.1.1 calc_parabola_vertex()

```
tuple modules.auxiliary.calc_parabola_vertex (
            float x1,
            float y1,
            float x2,
            float y2,
            float x3,
            float y3 )
```

Calculate the coefficients of a parabola given three points.

Parameters

x1,y1	x and y coordinates of the first point.
x2,y2	x and y coordinates of the second point.
x3,y3	x and y coordinates of the third point.

Returns

A, B, C: coefficients of the parabola.

4.7.1.2 determine_upper_freq()

```
float modules.auxiliary.determine_upper_freq (
            float nu_low,
            float evolve_time,
            float K,
            bool  DEBUG = False )
```

Determines upper ORBITAL frequency for a binary with K, starting from nu_0, evolving over evolve_time.

Parameters

nu_low	initial orbital frequency.
evolve_time	time it takes to evolve in Myr.
K	constant depending on the binary.

Returns

nu_upp: upper orbital frequency.

4.7.1.3 get_bin_factors()

```
np.array modules.auxiliary.get_bin_factors (
            np.array freqs,
            np.array bins )
```

Determine bin factors that often recur in the calculation to store them.

Parameters

freqs	central frequencies.
bins	frequency bin edges.

Returns

factors: factors to multiply the contributions with.

4.7.1.4 get_width_z_shell_from_z()

```
np.array modules.auxiliary.get_width_z_shell_from_z (
            np.array z_vals )
```

Returns the widths of the redshift shells in Mpc.

Parameters

z_vals	redshift values.

Returns

shells: shell widths in Mpc.

4.7.1.5 make_Omega_plot_unnorm()

```
None modules.auxiliary.make_Omega_plot_unnorm (
            np.array f,
            np.array Omega_sim,
            bool save = False,
            str save_name = "void",
            bool show = False )
```

Make a plot showing Omega for BWD.

Parameters

f	frequency array.
Omega_sim	Omega array.
save	save the figure.
save_name	name of the saved figure.
show	show the figure.

4.7.1.6 Omega()

```
np.array modules.auxiliary.Omega (
            float Omega_ref,
            float f_ref,
            np.array freq )
```

Create a f^{2/3} spectrum line.

Parameters

Omega_ref	reference Omega value.
f_ref	reference frequency.
freq	frequency array.

Returns

Omega: Omega array.

4.7.1.7 parabola()

```
float modules.auxiliary.parabola (
            float x,
            float a,
            float b,
            float c )
```

Calculate the value of a parabola given the coefficients.

Parameters

x	x value.
a,b,c	coefficients of the parabola.

Returns

y: y value.

4.7.1.8 tau_syst()

```
float modules.auxiliary.tau_syst (
            float f_0,
            float f_1,
            float K )
```

Calculates tau, the time it takes a binary with K to evolve from f_0 to f_1 (GW frequencies).

Parameters

f_0	initial frequency.
f_1	final frequency.
K	constant depending on the binary.

Returns

tau: time in Myr.

4.7.2 Variable Documentation

4.7.2.1 s_in_Myr

```
tuple modules.auxiliary.s_in_Myr = (u.Myr).to(u.s)
```

4.8 modules.physics Namespace Reference

Functions

- float chirp (float m1, float m2)

  Calculate the chirp mass in solar masses.
- float WD_radius (float m)

  Calculate the radius of a white dwarf of mass m.
- float a_min (float m1, float m2)

  Calculate minimum separation between two WDs of masses m1 and m2 (solar units).
- float Kepler (float m1, float m2)

  Calculate the orbital frequency of a binary with separation a_min and masses m1, m2.
- float K (float M)

  Calculate the factor K.
- np.array Period (float a, float m1, float m2)

  Calculate the orbital period of a binary system from Kepler's law.

4.8.1 Function Documentation

4.8.1.1 a_min()

```
float modules.physics.a_min (
            float m1,
            float m2 )
```

Calculate minimum separation between two WDs of masses m1 and m2 (solar units).

Parameters

m1	mass of the first WD in solar masses.
m2	mass of the second WD in solar masses.

Returns

The minimal separation in solar radii.

4.8.1.2 chirp()

```
float modules.physics.chirp (
            float m1,
            float m2 )
```

Calculate the chirp mass in solar masses.

Parameters

m1	mass of the first object in solar masses.
m2	mass of the second object in solar masses.

Returns

The chirp mass in solar masses.

4.8.1.3 K()

```
float modules.physics.K (
            float M )
```

Calculate the factor K.

Parameters

M	chirp mass in solar masses.

Returns

The factor K.

4.8.1.4 Kepler()

```
float modules.physics.Kepler (
            float m1,
            float m2 )
```

Calculate the orbital frequency of a binary with separation a_min and masses m1, m2.

Parameters

m1	mass of the first WD in solar masses.
m2	mass of the second WD in solar masses.

Returns

the orbital frequency in Hz.

4.8.1.5 Period()

```
np.array modules.physics.Period (
            float a,
            float m1,
            float m2 )
```

Calculate the orbital period of a binary system from Kepler's law.

Parameters

a	separation in solar radii.
m1	mass of the first WD in solar masses.
m2	mass of the second WD in solar masses.

Returns

The orbital periods in years.

4.8.1.6 WD_radius()

```
float modules.physics.WD_radius (
            float m )
```

Calculate the radius of a white dwarf of mass m.

Parameters

m	mass of the white dwarf in solar masses.

Eggleton 1986 fit to Nauenberg for high m and ZS for low m.

Returns

the radius in solar radii.

4.9 modules.RedshiftInterpolator Namespace Reference

Classes

- class RedshiftInterpolator

  This class is used to quickly determine the redshift at a given age of the Universe.

4.10 modules.SFH Namespace Reference

Functions

- representative_SFH (float age, ri.RedshiftInterpolator redshift_interpolator, float Delta_t=0., int SFH_num=1, float max_z=8.)

  Determines an appropriate value for the star formation rate at a given age.
- float SFH_MD (float z)

  Star formation history from [Madau, Dickinson 2014].
- float SFH2 (float z)

  Made up star formation history.
- float SFH3 (float z)

  Made up star formation history.
- float SFH4 (float z)

  Made up star formation history.

4.10.1 Function Documentation

4.10.1.1 representative_SFH()

```
modules.SFH.representative_SFH (
            float age,
            ri.RedshiftInterpolator redshift_interpolator,
            float  Delta_t = 0.,
            int  SFH_num = 1,
            float  max_z = 8.  )
```

Determines an appropriate value for the star formation rate at a given age.

The function looks for a representative value of the star formation rate given the age of the system, and takes into account an optional additional time delay.

Parameters

age	age of the system in Myr.
redshift_interpolator	RedshiftInterpolator object that interpolates the redshift at a given age.
Delta_t	time delay due to formation of binary or time required to reach the correct frequency bin, in Myr.
SFH_num	which star formation history to select. 1: Madau & Dickinson 2014, 2-4: made up, 5: constant 0.01.
max_z	maximum redshift.

Returns

SFR: star formation rate. Units: solar mass / yr / Mpc^3.

4.10.1.2 SFH2()

```
float modules.SFH.SFH2 (
            float z )
```

Made up star formation history.

Parameters

z	redshift.

Returns

SFR: star formation rate. Units: solar mass / yr / Mpc^3.

4.10.1.3 SFH3()

```
float modules.SFH.SFH3 (
            float z )
```

Made up star formation history.

Parameters

z	redshift.

Returns

SFR: star formation rate. Units: solar mass / yr / Mpc^3.

4.10.1.4 SFH4()

```
float modules.SFH.SFH4 (
            float z )
```

Made up star formation history.

Parameters

z	redshift.

Returns

SFR: star formation rate. Units: solar mass / yr / Mpc^3.

4.10.1.5 SFH_MD()

```
float modules.SFH.SFH_MD (
            float z )
```

Star formation history from [Madau, Dickinson 2014].

Parameters

z	redshift.

Returns

SFR: star formation rate. Units: solar mass / yr / Mpc^3.

4.11 modules.SimModel Namespace Reference

Classes

- class SimModel

  ! This class contains information about the run that needs to be shared over the different subroutines.

4.12 RedshiftInterpolator Namespace Reference

This module contains the class RedshiftInterpolator.

4.12.1 Detailed Description

This module contains the class RedshiftInterpolator.

The class RedshiftInterpolator is used to quickly determine the redshift at a given age of the Universe.

Author

Seppe Staelens

Date

2024-07-24

4.13 SeBa_pre_process Namespace Reference

Functions

- None main ()

    Function to calculate values from the SeBa output and save them in a dataframe.

4.13.1 Function Documentation

4.13.1.1 main()

```
None SeBa_pre_process.main ( )
```

Function to calculate values from the SeBa output and save them in a dataframe.

4.14 SimModel Namespace Reference

This module contains the class SimModel.

4.14.1 Detailed Description

This module contains the class SimModel.

The class SimModel contains information about the run that needs to be shared over the different subroutines.

Author

Seppe Staelens

Date

2024-07-24

Chapter 5

Class Documentation

5.1 modules.RedshiftInterpolator.RedshiftInterpolator Class Reference

This class is used to quickly determine the redshift at a given age of the Universe.

Public Member Functions

- None __init__ (self, str z_at_age_file)

    Initializes the RedshiftInterpolator object.
- float get_z_fast (self, float age)

    Quickly determine the redshift at a given age of the Universe.

Public Attributes

- interp_age

    The age of the Universe at which the redshift is determined.
- interp_z

    The redshift at the given age of the Universe.

5.1.1 Detailed Description

This class is used to quickly determine the redshift at a given age of the Universe.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 __init__()

```
None modules.RedshiftInterpolator.RedshiftInterpolator.__init__ (
             self,
            str z_at_age_file )
```

Initializes the RedshiftInterpolator object.

Parameters

z_at_age_file	file containing the redshift at a given age of the Universe.

5.1.3 Member Function Documentation

5.1.3.1 get_z_fast()

```
float modules.RedshiftInterpolator.RedshiftInterpolator.get_z_fast (
             self,
            float age )
```

Quickly determine the redshift at a given age of the Universe.

Parameters

age	age of the Universe in Myr.

Returns

redshift at the given age of the Universe.

5.1.4 Member Data Documentation

5.1.4.1 interp_age

```
modules.RedshiftInterpolator.RedshiftInterpolator.interp_age
```

The age of the Universe at which the redshift is determined.

5.1.4.2 interp_z

```
modules.RedshiftInterpolator.RedshiftInterpolator.interp_z
```

The redshift at the given age of the Universe.

The documentation for this class was generated from the following file:

- /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_Dwarf_AGWB/src/modules/RedshiftInterpolator.py

5.2 modules.SimModel.SimModel Class Reference

! This class contains information about the run that needs to be shared over the different subroutines.

Public Member Functions

- None __init__ (self, str INTEG_MODE, int z_interp, int N_freq=50, int N_int=20, float max_z=8, int SFH_num=1, float log_f_low=-5, float log_f_high=0)

  Initializes the SimModel object.
- None calculate_f_bins (self)

  Calculates the f bins and the bin factors.
- None calculate_z_bins (self)

  Calculates the z bins.
- None calculate_T_bins (self)

  Calculates the T bins.
- None calculate_cosmology_from_z (self)

  Calculations depending on the cosmology, starting from redshift bins.
- None calculate_cosmology_from_T (self, ri.RedshiftInterpolator z_interpolator)

  Calculations depending on the cosmology, starting from cosmic time bins.
- None set_mode (self, bool SAVE_FIG, bool DEBUG, bool TEST_FOR_ONE)

  Sets the mode of the simulation.

Public Attributes

- N_freq
- N_int
- max_z
- SFH_num
- log_f_low
- log_f_high
- INTEG_MODE
- f_plot

  The frequencies at which we will plot.
- f_bins

  The frequency bins.
- f_bin_factors

  The frequency bin factors that appear in the calculation.
- z_list

  The central values of the redshift bins.
- z_bins

  The redshift bins.
- T0
- T_range
- T_list
- T_bins
- dT
- z_widths

  The width of the redshift bins in Mpc.
- z_time_since_max_z

  The time since the maximum redshift
- ages

  The age of the universe at each redshift.
- SAVE_FIG
- DEBUG
- TEST_FOR_ONE

Static Public Attributes

- float light_speed = 0.30660139
  The speed of light in units of Mpc/Myr.

5.2.1 Detailed Description

! This class contains information about the run that needs to be shared over the different subroutines.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 __init__()

```
None modules.SimModel.SimModel.__init__ (
              self,
             str INTEG_MODE,
             int z_interp,
             int  N_freq = 50,
             int  N_int = 20,
             float  max_z = 8,
             int  SFH_num = 1,
             float  log_f_low = -5,
             float  log_f_high = 0 )
```

Initializes the SimModel object.

Parameters

N_freq	number of frequency bins.
N_int	number of integration bins (z or T).
max_z	maximum redshift.
SFH_num	which star formation history to select. 1: Madau & Dickinson 2014, 2-4: made up, 5: constant 0.01.
log_f_low	lower bound of the frequency bins in log10 space.
log_f_high	upper bound of the frequency bins in log10 space.

Returns

instance of SimModel, with frequency and redshift bins calculated, and cosmology set.

5.2.3 Member Function Documentation

5.2.3.1 calculate_cosmology_from_T()

```
None modules.SimModel.SimModel.calculate_cosmology_from_T (
              self,
             ri.RedshiftInterpolator z_interpolator )
```

Calculations depending on the cosmology, starting from cosmic time bins.

Calculates the redshifts, the time since the maximum redshift, and the ages of the universe at each time.

5.2.3.2 calculate_cosmology_from_z()

```
None modules.SimModel.SimModel.calculate_cosmology_from_z (
              self )
```

Calculations depending on the cosmology, starting from redshift bins.

Sets the widths of the z bins and the time since max z, as well as the age of the universe at each redshift.

5.2.3.3 calculate_f_bins()

```
None modules.SimModel.SimModel.calculate_f_bins (
              self )
```

Calculates the f bins and the bin factors.

5.2.3.4 calculate_T_bins()

```
None modules.SimModel.SimModel.calculate_T_bins (
              self )
```

Calculates the T bins.

5.2.3.5 calculate_z_bins()

```
None modules.SimModel.SimModel.calculate_z_bins (
              self )
```

Calculates the z bins.

5.2.3.6 set_mode()

```
None modules.SimModel.SimModel.set_mode (
              self,
             bool SAVE_FIG,
             bool DEBUG,
             bool TEST_FOR_ONE )
```

Sets the mode of the simulation.

Parameters

SAVE_FIG	whether to save the figures.
DEBUG	whether to print more output.
TEST_FOR_ONE	whether to test for only one system.
INT_MODE	whether to integrate over redshift or time.

5.2.4 Member Data Documentation

5.2.4.1 ages

`modules.SimModel.SimModel.ages`

The age of the universe at each redshift.

5.2.4.2 DEBUG

`modules.SimModel.SimModel.DEBUG`

5.2.4.3 dT

`modules.SimModel.SimModel.dT`

5.2.4.4 f_bin_factors

`modules.SimModel.SimModel.f_bin_factors`

The frequency bin factors that appear in the calculation.

5.2.4.5 f_bins

`modules.SimModel.SimModel.f_bins`

The frequency bins.

5.2.4.6 f_plot

`modules.SimModel.SimModel.f_plot`

The frequencies at which we will plot.

5.2.4.7 INTEG_MODE

`modules.SimModel.SimModel.INTEG_MODE`

5.2.4.8 light_speed

`float modules.SimModel.SimModel.light_speed = 0.30660139 [static]`

The speed of light in units of Mpc/Myr.

5.2.4.9 log_f_high

`modules.SimModel.SimModel.log_f_high`

5.2.4.10 log_f_low

`modules.SimModel.SimModel.log_f_low`

5.2.4.11 max_z

`modules.SimModel.SimModel.max_z`

5.2.4.12 N_freq

`modules.SimModel.SimModel.N_freq`

5.2.4.13 N_int

`modules.SimModel.SimModel.N_int`

5.2.4.14 SAVE_FIG

`modules.SimModel.SimModel.SAVE_FIG`

5.2.4.15 SFH_num

`modules.SimModel.SimModel.SFH_num`

5.2.4.16 T0

`modules.SimModel.SimModel.T0`

5.2.4.17 T_bins

`modules.SimModel.SimModel.T_bins`

5.2.4.18 T_list

`modules.SimModel.SimModel.T_list`

5.2.4.19 T_range

`modules.SimModel.SimModel.T_range`

5.2.4.20 TEST_FOR_ONE

`modules.SimModel.SimModel.TEST_FOR_ONE`

5.2.4.21 z_bins

`modules.SimModel.SimModel.z_bins`

The redshift bins.

5.2.4.22 z_list

`modules.SimModel.SimModel.z_list`

The central values of the redshift bins.

5.2.4.23 z_time_since_max_z

`modules.SimModel.SimModel.z_time_since_max_z`

The time since the maximum redshift

5.2.4.24 z_widths

`modules.SimModel.SimModel.z_widths`

The width of the redshift bins in Mpc.

The documentation for this class was generated from the following file:

- /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_Dwarf_AGWB/src/modules/SimModel.py

Chapter 6

File Documentation

6.1 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/Create_z_at_age.py File Reference

This program creates a list of redshift values at a list of ages of the Universe, that can be saved and used to interpolate in the main code.

Namespaces

- namespace Create_z_at_age

Functions

- None Create_z_at_age.main ()
  Function to create list of z-values at given ages of the Universe.

6.1.1 Detailed Description

This program creates a list of redshift values at a list of ages of the Universe, that can be saved and used to interpolate in the main code.

Date

2024-07-29

Author

Seppe Staelens

6.2 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/GWB.py File Reference

This program calculates the GWB based on the method described in my thesis, using uniform redshift bins.

Namespaces

- namespace GWB

Functions

- GWB.main ()

  Main function.

Variables

- GWB.action
- GWB.category
- GWB.size
- GWB.titlesize
- GWB.labelsize
- GWB.fontsize
- tuple GWB.s_in_Myr = (u.Myr).to(u.s)

6.2.1 Detailed Description

This program calculates the GWB based on the method described in my thesis, using uniform redshift bins.

Date

2024-07-26

The program calculates the GWB based on the method described in my thesis, using uniform redshift bins. It is divided into three main parts: the bulk part, the birth part, and the merger part. The bulk part calculates the majority of the GWB, what is referred to in my thesis as the 'generic case'. The birth part adds the contribution of the 'birth bins' to the bulk GWB. The merger part adds the contribution of the 'merger bins' due to Kepler max to the bulk GWB. The program saves a dataframe with all the essential information.

Author

Seppe Staelens

6.3 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/__init__.py File Reference

Namespaces

- namespace modules

6.4 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/add_birth.py File Reference

This file contains a routine that adds the contribution of the 'birth bins' to the bulk GWB.

Namespaces

- namespace modules
- namespace modules.add_birth

Functions

- None modules.add_birth.add_birth (sm.SimModel model, pd.DataFrame data, ri.RedshiftInterpolator z_↩ interp, str tag)

    This routine adds the contribution of the 'birth bins' to the bulk GWB.

6.4.1 Detailed Description

This file contains a routine that adds the contribution of the 'birth bins' to the bulk GWB.

Author

Seppe Staelens

Date

2024-07-24

6.5 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/add_bulk.py File Reference

This file contains a routine that calculates the majority of the GWB, what is referred to in my thesis as the 'generic case'.

Namespaces

- namespace modules
- namespace modules.add_bulk

Functions

- None modules.add_bulk.add_bulk (sm.SimModel model, pd.DataFrame data, ri.RedshiftInterpolator z_interp, str tag)

    This routine calculates the majority of the GWB, what is referred to in my thesis as the 'generic case'.

6.5.1 Detailed Description

This file contains a routine that calculates the majority of the GWB, what is referred to in my thesis as the 'generic case'.

Author

Seppe Staelens

Date

2024-07-24

6.6 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/add_merge.py File Reference

This file contains a routine that adds the contribution of the 'merger bins' due to Kepler max to the bulk+birth GWB.

Namespaces

- namespace modules
- namespace modules.add_merge

Functions

- None modules.add_merge.add_merge (sm.SimModel model, pd.DataFrame data, ri.RedshiftInterpolator z↩ _interp, str tag)

  This routine adds the contribution of the 'merger bins' due to Kepler max to the bulk+birth GWB.

6.6.1 Detailed Description

This file contains a routine that adds the contribution of the 'merger bins' due to Kepler max to the bulk+birth GWB.

Author

Seppe Staelens

Date

2024-07-24

6.7 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/auxiliary.py File Reference

This module contains auxiliary functions that are used in the main code.

Namespaces

- namespace modules
- namespace modules.auxiliary

Functions

- tuple modules.auxiliary.calc_parabola_vertex (float x1, float y1, float x2, float y2, float x3, float y3)

    Calculate the coefficients of a parabola given three points.
- float modules.auxiliary.parabola (float x, float a, float b, float c)

    Calculate the value of a parabola given the coefficients.
- np.array modules.auxiliary.get_bin_factors (np.array freqs, np.array bins)

    Determine bin factors that often recur in the calculation to store them.
- np.array modules.auxiliary.get_width_z_shell_from_z (np.array z_vals)

    Returns the widths of the redshift shells in Mpc.
- np.array modules.auxiliary.Omega (float Omega_ref, float f_ref, np.array freq)

    Create a f^{2/3} spectrum line.
- None modules.auxiliary.make_Omega_plot_unnorm (np.array f, np.array Omega_sim, bool save=False, str save_name="void", bool show=False)

    Make a plot showing Omega for BWD.
- float modules.auxiliary.tau_syst (float f_0, float f_1, float K)

    Calculates tau, the time it takes a binary with K to evolve from f_0 to f_1 (GW frequencies).
- float modules.auxiliary.determine_upper_freq (float nu_low, float evolve_time, float K, bool DEBUG=False)

    Determines upper ORBITAL frequency for a binary with K, starting from nu_0, evolving over evolve_time.

Variables

- tuple modules.auxiliary.s_in_Myr = (u.Myr).to(u.s)

6.7.1 Detailed Description

This module contains auxiliary functions that are used in the main code.

This module contains auxiliary physics functions that are used to pre-process the population synthesis data.

Author

Seppe Staelens

Date

2024-07-24

Author

Seppe Staelens

Date

2024-07-29

6.8 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/physics.py File Reference

Namespaces

- namespace modules
- namespace modules.physics

Functions

- float modules.physics.chirp (float m1, float m2)
    Calculate the chirp mass in solar masses.
- float modules.physics.WD_radius (float m)
    Calculate the radius of a white dwarf of mass m.
- float modules.physics.a_min (float m1, float m2)
    Calculate minimum separation between two WDs of masses m1 and m2 (solar units).
- float modules.physics.Kepler (float m1, float m2)
    Calculate the orbital frequency of a binary with separation a_min and masses m1, m2.
- float modules.physics.K (float M)
    Calculate the factor K.
- np.array modules.physics.Period (float a, float m1, float m2)
    Calculate the orbital period of a binary system from Kepler's law.

6.9 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/RedshiftInterpolator.py File Reference

Classes

- class modules.RedshiftInterpolator.RedshiftInterpolator
    This class is used to quickly determine the redshift at a given age of the Universe.

Namespaces

- namespace modules
- namespace modules.RedshiftInterpolator
- namespace RedshiftInterpolator
    This module contains the class RedshiftInterpolator.

6.10 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/SFH.py File Reference

This file contains the functions to determine the star formation rate.

Namespaces

- namespace modules
- namespace modules.SFH

Functions

- modules.SFH.representative_SFH (float age, ri.RedshiftInterpolator redshift_interpolator, float Delta_t=0., int SFH_num=1, float max_z=8.)

    Determines an appropriate value for the star formation rate at a given age.
- float modules.SFH.SFH_MD (float z)

    Star formation history from [Madau, Dickinson 2014].
- float modules.SFH.SFH2 (float z)

    Made up star formation history.
- float modules.SFH.SFH3 (float z)

    Made up star formation history.
- float modules.SFH.SFH4 (float z)

    Made up star formation history.

6.10.1 Detailed Description

This file contains the functions to determine the star formation rate.

Date

2024-07-24

The file contains the functions to determine the star formation rate. The function representative_SFH determines an appropriate value for the star formation rate at a given age. It allows for an optional additional time delay, due to a delay in formation of the binary, or if time is required to move to the correct frequency bin. The functions SFH_MD, SFH2, SFH3, and SFH4 are star formation histories that can be selected.

Author

Seppe Staelens

6.11 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/modules/SimModel.py File Reference

Classes

- class modules.SimModel.SimModel

    ! This class contains information about the run that needs to be shared over the different subroutines.

Namespaces

- namespace modules
- namespace modules.SimModel
- namespace SimModel

    This module contains the class SimModel.

6.12 /home/seppe/Documents/data/Papers/AnA.683.A139(2024)/White_↩ Dwarf_AGWB/src/SeBa_pre_process.py File Reference

This program takes the output of the SeBa population synthesis code and calculates other values from it. The results are saved in a dataframe that can be used in the main code.

Namespaces

- namespace SeBa_pre_process

Functions

- None SeBa_pre_process.main ()
  Function to calculate values from the SeBa output and save them in a dataframe.

6.12.1 Detailed Description

This program takes the output of the SeBa population synthesis code and calculates other values from it. The results are saved in a dataframe that can be used in the main code.

Date

2024-07-29

Author

Seppe Staelens

Index