Draft version July 2, 2024
Typeset using LaTeX default style in AASTeX631

Documentation for CHIPS: Complete History of Interaction-Powered Supernovae

CHIPS team: Yuki Takei,[1, 2, 3] Daichi Tsuna,[4, 2] Takatoshi Ko,[2, 5] and Toshikazu Shigeyama[2, 5]

[1] *Yukawa Institute for Theoretical Physics, Kyoto University, Kitashirakawa-Oiwake-cho, Sakyo-ku, Kyoto, Kyoto 606-8502, Japan*
[2] *Research Center for the Early Universe (RESCEU), Graduate School of Science, The University of Tokyo, Tokyo 113-0033, Japan*
[3] *Astrophysical Big Bang Laboratory, RIKEN, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan*
[4] *TAPIR, Mailcode 350-17, California Institute of Technology, Pasadena, CA 91125, USA*
[5] *Department of Astronomy, Graduate School of Science, The University of Tokyo, Tokyo, Japan*

ABSTRACT

This document introduces CHIPS (Complete History of Interaction-Powered Supernovae), an open-source radiation hydrodynamical code that can model various observables of interacting core-collapse supernovae. Given a model of the progenitor star and various parameters for the mass eruption and the subsequent SN, the code can calculate the circumstellar matter created upon eruption and bolometric light curves for the subsequent SNe, as well as the light curves in different optical bands.
This is a living document, which will be updated as new features are added to CHIPS. The current version (July 3, 2024) is for CHIPS-v2.1, released as of July 2024.

Contents

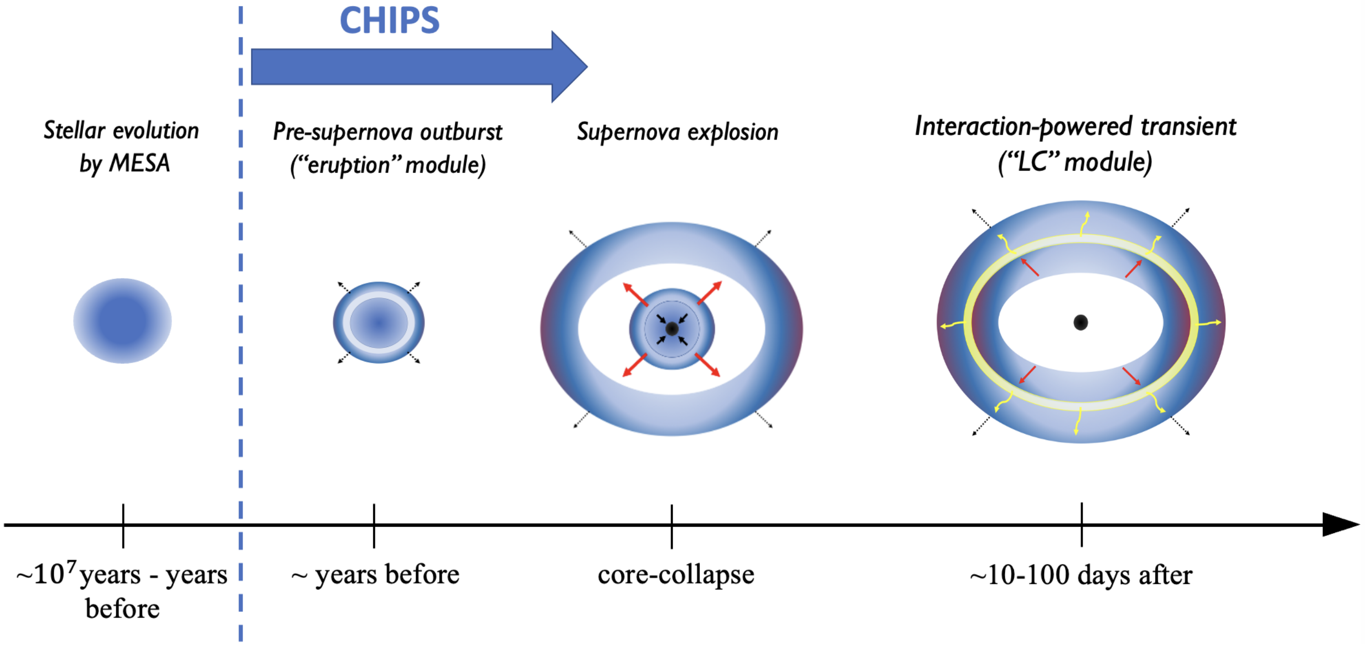
CHIPS
Stellar evolution by MESA
Pre-supernova outburst ("eruption" module)
Supernova explosion
Interaction-powered transient ("LC" module)
~10^7 years - years before
~ years before
core-collapse
~10-100 days after

Figure 1. Schematic overview of CHIPS. The code first simulates the outburst just prior to core-collapse that creates a dense CSM, and then simulates the interaction between the SN ejecta and the CSM, including light curves powered by the interaction.

1. WHAT IS CHIPS?

1.1. *Overview*

CHIPS[1] is an open-source tool for simulating the circumstellar matter (CSM) and light curves of interaction-powered transients. Coupling the MESA stellar evolution code and one-dimensional (radiation-)hydrodynamical codes implemented by the authors, the user can obtain the CSM created from pre-SN outburst, and light curves (LCs) of the subsequent interaction-powered SNe.

The schematic picture of our code is shown in Figure 1. CHIPS can generate a realistic CSM from a model-agnostic mass eruption calculation, which can serve as a reference for observers to compare with various observations of the CSM. It can also generate bolometric LCs from CSM interaction, which can be compared with observed LCs.

For our sample red supergiant (RSG) models, the calculation of mass eruption and LC typically takes respectively half a day and half an hour on modern CPUs. For the sample stripped star models, both the eruption and LC calculations typically take around an hour.

1.2. *Example Applications*

Here are some examples of how CHIPS has been used in the past:

- Model light curves of interacting SNe that have dense CSM close to the star, to extract the CSM parameters. Our code has been used to interpret light curves of some SN IIn (Takei et al. 2022), SN Ibn/Icn (Takei et al. 2024, Pellegrino et al. in prep.), and Type II SN 2023ixf (Hiramatsu et al. 2023).
- Simulate a realistic hydrodynamical profile of dense CSM originating from a mass outburst (Tsuna et al. 2021; Ko et al. 2022; Tsuna & Takei 2023), which can be used for other analyses such as modeling spectral signatures of these CSM (Ishii et al. 2024).
- Obtain "SN precursor" light curves from mass outbursts (Tsuna et al. 2023; Takei et al. 2024).
- Pass over our SN modeling outputs to other post-processing codes, to model multi-wavelength and multi-messenger emission from interacting SNe (under development).

[1] https://github.com/DTsuna/CHIPS

2. SETUP AND QUICKSTART

The requirement for running `CHIPS` are compilers for C and fortran (e.g. `gcc` and `gfortran` respectively), and `python3` with `numpy`, `scipy`, and `mesa_reader` [2] (modules for reading MESA stellar models) installed. If `matplotlib` is installed, one can also obtain some plots that are automatically generated by `CHIPS`.

To get started with `CHIPS`, you can clone the following repository and compile the scripts.

```
git clone https://github.com/DTsuna/CHIPS.git
make
```

For historical reasons, the code for mass eruption is in Fortran and that for the LC is in C. These codes are separately compiled in a single makefile. After compilation, the calculations for both the mass eruption and the LC are done by:

```
python3 run.py --argument1 value1 --argument2 value2 ...
```

The input parameters/files can be controlled by the arguments described in Sec 3 and Table 2. If one is interested in quickly knowing some examples for setting the arguments, skip to Sec 5.

3. INPUT

3.1. *Input files*

3.1.1. *MESA stellar models*

`CHIPS` requires a stellar model as initial condition of our calculations. The code accepts as inputs stellar models generated by MESA[3] (Paxton et al. 2011, 2013, 2015, 2018, 2019; Jermyn et al. 2023), which is a state-of-the-art open-source stellar evolution code widely used in the stellar physics and transients communities. Alternatively, one can use the MESA pre-SN models generated by us. Our sample models cover

- IIn progenitors: red supergiants of solar metallicity (Z=0.014) with ZAMS mass range 13-26 $M_\odot$, with 1 $M_\odot$ interval up to 20 $M_\odot$ and 2 $M_\odot$ interval from 20 to 26 $M_\odot$,
- Ibn progenitors: hydrogen-poor helium stars of solar metallicity with ZAMS mass range 15-29 $M_\odot$ with 1 $M_\odot$ interval, and
- Icn progenitors: hydrogen and helium poor stars of solar metallicity with ZAMS masses of 20, 22, 25, 29 $M_\odot$.

The pre-SN models are in zip files in the directories input/mesa_models_XXX. Once you un-zip these files, you will find files with naming showing the mass and metallicity at ZAMS. **For the light curve module, the currently available opacity tables support only solar metallicity (Z=0.014) input progenitor models for IIn/Ibn, and the above four models for Icn. Work is under way to enable more flexibility in the progenitors.**

If you wish to use your own stellar model, the columns listed in Table 1 are required in the MESA input files to run `CHIPS`. In most cases these parameters are saved in the files by default, but you can check those in the `profile_columns.list` file in your MESA working directory to see if everything is included.

3.1.2. *Opacity table (optional)*

For the opacity in the eruption calculations, a simple analytical formula in equation (6) of Kuriyama & Shigeyama (2020) is used by default. One can instead give custom (Rosseland-mean) opacity tables, like those made by e.g., OPAL (Iglesias & Rogers 1996) or AESOPUS (Marigo & Aringer 2009; Marigo et al. 2022), by adding the argument `--opacity-table` when executing `run.py`. There are two notes of caution: (i) The table must be rectangular with a format given as the sample in the `input/rosseland` directory, and (ii) When values outside the table is requested, the closest edge values are used. Some sample tables are given in the `input/rosseland` directory, which the user can adopt when doing the calculations with our sample progenitors.

[2] For installation of `mesa_reader`, please see https://github.com/wmwolf/py_mesa_reader for details.

[3] https://docs.mesastar.org

Table 1. Required parameters/columns when reading the input MESA files.

Name	Definition	Note
`photosphere_r`	photospheric radii ($R_\odot$)	For stellar wind component
`photosphere_L`	photospheric luminosity ($L_\odot$)	For stellar wind component
`Teff`	Effective temperature (K)	For stellar wind component
`star_mdot`	star's mass-loss rate ($M_\odot$ yr^{-1})	For stellar wind component
`star_mass`	star's mass ($M_\odot$)	
`he_core_mass`	Mass of helium core ($M_\odot$)	Required for "IIn" models
`c(co)_core_mass`	Mass of carbon-oxygen core ($M_\odot$)	Required for "Ibn" models
`si_core_mass`	Mass of silicon core ($M_\odot$)	Required for "Icn" models
`mass`	mass coordinate	
`logR`, `logRho`, `logP`, `logT`	radius, density, pressure, temperature profiles	
`h1`	hydrogen mass fraction	
`he3`, `he4`	^{3}He, ^{4}He mass fraction	
`c12`, `n14`, `o16`	^{12}C, ^{14}N, ^{16}O mass fraction	
`ne20`, `mg24`	^{20}Ne, ^{24}Mg mass fraction	
`si28`, `s32`	^{28}Si, ^{32}S mass fraction	

Note—The `c_core_mass` has been renamed to `co_core_mass` from the MESA r21.12.1 release; `CHIPS` accepts either of these two, and regards this as the CO core mass in the code.

3.2. *Other input parameters*

The other input parameters and the arguments are in Table 2. Every time when executing `run.py`, the main arguments used for the calculations are stored in a file `params/params_(DATE AND TIME).dat`.

Required parameters $f_{\rm inj}$ (`--finj`) and $t_{\rm inj}$ (`--tinj`) control the energy and timing of the energy injection that powers the pre-SN eruption. The parameter $f_{\rm inj}$ is dimensionless, normalized by the binding energy of the computational region. As a rule of thumb, a larger $f_{\rm inj}$ leads to heavier (and faster) CSM (see Takei et al. 2022, 2024), and $f_{\rm inj} \gtrsim 1$ leads to (near-)complete ejection of the envelope. The parameter $t_{\rm inj}$ is the interval in years between energy injection (that creates the CSM) and the final SN. A larger $t_{\rm inj}$ (for a fixed $f_{\rm inj}$) leads to a more extended, diluted CSM. The CGS values of the injected energy and $t_{\rm inj}$ can be found in the output file `EruptionFiles/parameter.txt`.

There are other optional arguments to control the pre-SN eruption. The eruption code is Lagrangian, and one can use the arguments `--eruption-innerMr` and `--eruption-Ncell` to control the inner mass coordinate and number of cells in the computational region (see Sec. 4.1 for details). Argument `--injection-duration` adjusts the duration of the energy injection (see Ko et al. 2022 for applications).

The LC module has the basic parameters $E_{\rm ej}$ (`--Eej`) and $M_{\rm Ni}$ (`--Mni`) that respectively specifies the ejecta kinetic energy and ^{56}Ni mass of the SN[4]. For SN IIn models, adding the argument `--calc-multiband` will enable ray-tracing calculations to obtain multi-band optical LCs (see Sec. 3.2.3 in Takei et al. 2022). One can also entirely skip the (more expensive) pre-SN eruption calculation and directly start the LC calculation, using the argument `--skip-eruption`. This requires an output file (`EruptionFiles/result99.txt`) already obtained by a previous eruption calculation.

4. CALCULATIONS AND OUTPUTS

Here we summarize the basic calculations done in our code, and various output files. For details of the calculations, we recommend checking Appendix A, and our methods paper (Takei et al. 2022, 2024). Figure 2 shows the file tree of the `CHIPS` code, summarizing the key inputs, outputs and the source codes.

[4] The ejecta mass $M_{\rm ej}$ is automatically set by CHIPS from the equation $M_{\rm ej} = M_* - M_{\rm CSM} - M_{\rm rem}$, where the remnant mass $M_{\rm rem}$ is obtained from an analytical fitting relation between $M_{\rm rem}$ and the CO core mass (Schneider et al. 2021). The remnant masses are typically $\approx 1.4\ M_\odot$ for most of our sample progenitors.

Table 2. Input parameters and arguments for running the `CHIPS` code, using the execution script `run.py`.

Name	Definition	Default
`stellar-model`	Path to input MESA model	(Required)
`finj`	Injected energy normalized by envelope binding energy	(Required)
`tinj`	Energy injection time prior to core-collapse, in years	(Required)
`Eej`	Explosion energy of the final SN, in ergs	$[10^{51}, 3\times10^{51}, 10^{52}]$
`Mni`	Mass of radioactive ^{56}Ni, in solar mass	0
`eruption-innerMr`	The innermost mass coordinate where the energy is injected.	Sec. 4.1.1, 4.1.2
`eruption-Ncell`	Number of cells in the eruption calculation	10000
`injection-duration`	Duration of energy injection for eruption calculation, in seconds	1000
`opacity-table`	Opacity table for eruption calculation (optional)	Analytical formula (App. A)
`analytical-CSM`	Replace CSM profile with analytical model (for IIn models)	False
`skip-eruption`	Skip eruption and start LC calculation using pre-computed CSM	False
`calc-multiband`	Obtain multi-band LC (only for IIn models)	False

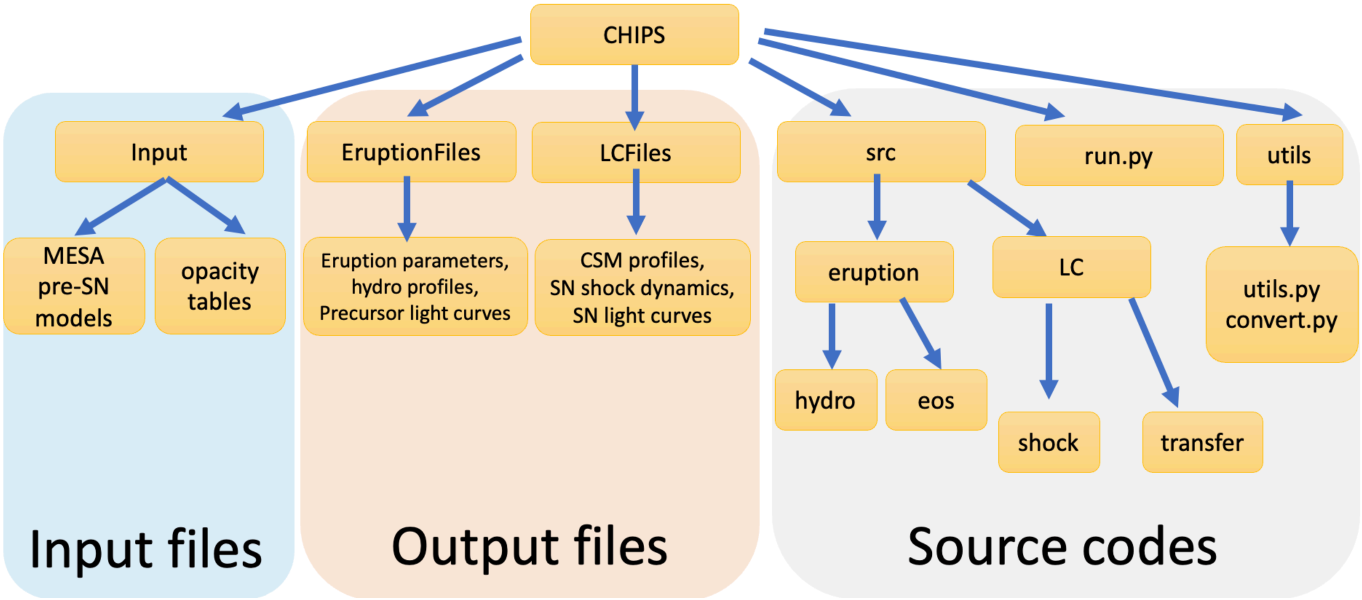

Figure 2. File tree for the `CHIPS` code. The module is composed of input files that include MESA models and opacity table, the source codes for both the mass eruption and light curve calculations as well as simple python scripts for pre/post-processing various files, and the directories for storing outputs from each of the calculations.

The calculations are done slightly differently for supergiants and stripped stars, due to the very different radii, dynamical times and compositions for these progenitors. The progenitors (`SNType` in the code) are conveniently labeled as “IIn”, “Ibn” and “Icn”, with the following simple criteria (Hachinger et al. 2012; Williamson et al. 2021):

- We label as “IIn” progenitors as those with hydrogen masses greater than 0.033 $M_\odot$
- “Ibn” progenitors as those with hydrogen mass less than this threshold but helium mass greater than 0.05 $M_\odot$,
- “Icn” progenitors with both hydrogen and helium masses lower than these thresholds.

The CSM profiles and the final light curve files share essentially the same format, while the detailed input/outputs of the intermediate files are somewhat different.

4.1. *Calculations and Outputs for pre-SN mass eruption*

The eruption part of the code, developed by Kuriyama & Shigeyama (2020), solves the response of the initially hydrostatic stellar envelope after a sudden energy injection. The location of the energy injection is decided by the input parameter `--eruption-innerMr`, with a default value being $0.2M_\odot$ outside the inner core of the star. The location of the "inner core" is defined from the core mass saved in the MESA file, specifically the He core mass for IIn progenitors, the CO core mass for Ibn progenitors, and the Si core mass for Icn progenitors.

4.1.1. *Red supergiant ("IIn") progenitor case*

The envelope eruption process occurs over roughly a dynamical timescale of the stellar envelope, defined in the code by the stellar radius R_* and mass of the envelope (computational region) $M_{\rm env}$ as

$$t_{\rm dyn} \equiv \sqrt{\frac{\pi^2 R_*^3}{4GM_{\rm env}}} \sim 170\,{\rm days} \left(\frac{R_*}{700\,R_\odot}\right)^{3/2} \left(\frac{M_{\rm env}}{10\,M_\odot}\right)^{-1/2} \text{(supergiants)}. \tag{1}$$

Afterwards, for computational speed the inner part of the star (defined at radii $< 10^{13}$ cm) is removed from the simulation. This inner region is typically still well within the star, and does not affect the dynamics of the final CSM.

The calculations can be sped up further by turning off radiation transfer at $t > 2t_{\rm dyn}$, by setting `continueTransfer` in `run.py` to False[5]. However this creates an artificial shocked region near the stellar surface due to the accretion shock by bound CSM that falls back. We thus recommend doing this with the argument `--analytical-CSM` in Table 2, which updates the profile to what would be expected when running rad-hydro calculations (see Fig 2 of Takei et al. 2022). However this is not guaranteed to work for every case, and may crash when e.g. `--tinj` is over several decades, when the accretion shock can engulf the bulk of the CSM. When flipping this switch, the user should be careful to check if the reconstructed CSM profile (`LCFiles/CSM.txt`) is successfully generated and looks appropriate (and that's why we don't add this as an flexible argument in Table 2).

The output files are stored in the `EruptionFiles/` directory. The final profile is saved as `atCCSN.txt`, and intermediate hydrodynamical profiles are saved as files `resultXX.txt` and `intermediateYYyr.txt`. The former files (XX$= 01, 02, \cdots, 99$) are saved at $t = 2t_{\rm dyn}[(\texttt{XX} - 1)/90]$ from the first 90 files and with interval of $(t_{\rm inj} - 2t_{\rm dyn})/9$ for the last 9 files, mainly to follow before and after the eruption. The files `intermediateYYyr.txt` (YY$= 01, 02, \cdots, t_{\rm inj}$) save the profiles at integer years after energy injection. If these files are available, for light curve modeling one can skip the eruption calculation using the code `after_eruption.py`. For example if one ran a calculation for $t_{\rm inj} > 10$ year, and then instead wants to calculate the light curves for $t_{\rm inj} = 10$ years, run

```
python after_eruption.py --Eej 1e51 --stellar-model (Path To MESA Model)
--profile-at-cc EruptionFiles/intermediate10yr.txt
```

4.1.2. *Stripped ("Ibn", "Icn") progenitor case*

The calculations for stripped progenitors are done similarly to the RSG case, except that the governing timescale is very different. The dynamical timescale is now

$$t_{\rm dyn} \sim 1800\,{\rm s} \left(\frac{R_*}{R_\odot}\right)^{3/2} \left(\frac{M_{\rm env}}{2\,M_\odot}\right)^{-1/2} \text{(stripped stars)}, \tag{2}$$

which is several orders of magnitude shorter than that of supergiants. This is also 3–5 orders of magnitude shorter than the timescale for $t_{\rm inj}$ we expect for the observed interacting SN Ibc ($\sim$ year), and it is computationally unfeasible to run the eruption code for many thousands of dynamical times.

Fortunately, after a few dynamical times we find that the internal energy in the ejected CSM becomes negligible compared to its kinetic energy, and the CSM simply evolves under gravity. Thus for stripped stars, we stop the simulations at $t = 2t_{\rm dyn}$, and approximately solve the trajectory of each computational cell independently under the equation of motion,

$$\frac{d^2 r}{dt^2} = -\frac{GM_r}{r^2}, \tag{3}$$

[5] We have done this in our Takei et al. (2022) paper as we had to run a large number of parameter sets.

where M_r denotes the enclosed mass of each cell. We integrate this for the unbound part of the CSM from $t = 2t_{\rm dyn}$ (saved as "result90.txt") to $t = t_{\rm inj}$, with a 4th-order Runge-Kutta solver in scipy (`integrate.rk45`). We then do a fit of the (unbound part of the) CSM density profile with the analytical formula (Tsuna & Takei 2023, eq 2), which also enables extending the CSM profile inwards to the bound part of the CSM that continues to the surface of the star.

For Ibn/Icn case, the code outputs the files `resultXX.txt` at time $t = 2t_{\rm dyn}[(\texttt{XX}-1)/99]$ from energy injection. The calculations only run until $\sim 2t_{\rm dyn} \sim 10^3$s, so the `intermediateYYyr.txt` files are not generated. If one would like to generate new CSM with different values of `--tinj` using a precomputed eruption calculation, they can run `run.py` with the new `tinj` and with the argument `--skip-eruption`. Then the code skips the eruption calculation and does the (much faster) Runge-Kutta calculation starting from `EruptionFiles/result99.txt`.

4.2. *Calculations for Interacting SN Light Curves (LCs)*

After the SN ejecta collides with the CSM, reverse and forward shocks form and propagates into the SN ejecta and the unshocked CSM respectively. The radiative flux emerging from the shocked region propagates to the observer, after being (partially) reprocessed by the unshocked CSM. The CHIPS code calculates the bolometric and multi-band LCs following the methodology first developed in Takei & Shigeyama (2020), and refined in Takei et al. (2022, 2024).

The CSM profiles obtained from the eruption calculation and the above post-processing steps are re-meshed into 1000 cells and saved into a file `LCFiles/CSM.txt`. The file contains the enclosed CSM mass, radius, velocity, density and hydrogen/helium mass fractions. We then generate Rosseland and Planck mean opacities for radiation transfer using the mass-weighted mean abundance of the CSM, which are saved in `LCFiles`.

The properties of the SN ejecta are then calculated. We assume a homologously expanding SN ejecta with a double power-law density profile (Matzner & McKee 1999). The code also incorporates the user-supplied explosion energy (`--Eej`) to calculate the density/velocity profile of the SN ejecta.

Using the ejecta and CSM profiles, the propagation of the forward and reverse shocks are solved. This is saved in `LCFiles/..._shock_output_...txt`, with labels of `SNType` (IIn/Ibn/Icn), `Mni`, `tinj` and `Eej`. This file is then passed over to a post-processing script, that solves radiation transfer of shock-generated emission in the unshocked CSM to estimate the luminosity at the observer. The bolometric LC is saved in `LCFiles/..._lightcurve_...txt`, again with labels of `SNType`, `Mni`, `tinj` and `Eej`. If the LC is successfully generated with a well-defined rise and fall, the code finally outputs the peak luminosity (erg/s) and rise/decay timescale (days) of the SN LC. The rise (decay) time is defined as the interval between the peak and when the luminosity first rises (decays) to 1% of the peak.

5. EXAMPLES

We show examples of the outputs of our code, for both hydrogen-rich (SN IIn) and hydrogen-poor (SN Ibn) progenitors. We present the input, and some figures we have plotted from the output files using matplotlib.

5.1. *Case for hydrogen-rich progenitor (Type IIn supernovae)*

Let us simulate a IIn-like interaction-powered SN of

- a RSG star with ZAMS (Zero-Age Main Sequence) mass 15 Msun and metallicity 0.014,
- which experiences mass eruption 30 years before core-collapse, with energy injection of 0.8 times the envelope's binding energy,
- and finally explodes with energy 10^{51} erg.

Then one can run the execution script `run.py` as follows:

```
python3 run.py --finj 0.8 --tinj 30 --stellar-model input/mesa_models_rsg/15Msun_Z0.014
    _preccsn.data --Eej 1e51 --analytical-CSM
```

The outputs are summarized in Figure 3.

- (Top-left): shock propagation and mass eruption at a few epochs from energy injection, plotted using `EruptionFiles/resultXX.txt`.
- (Top-right): CSM profile at a few epochs from eruption, plotted using `EruptionFiles/intermediateYYyr.txt`.

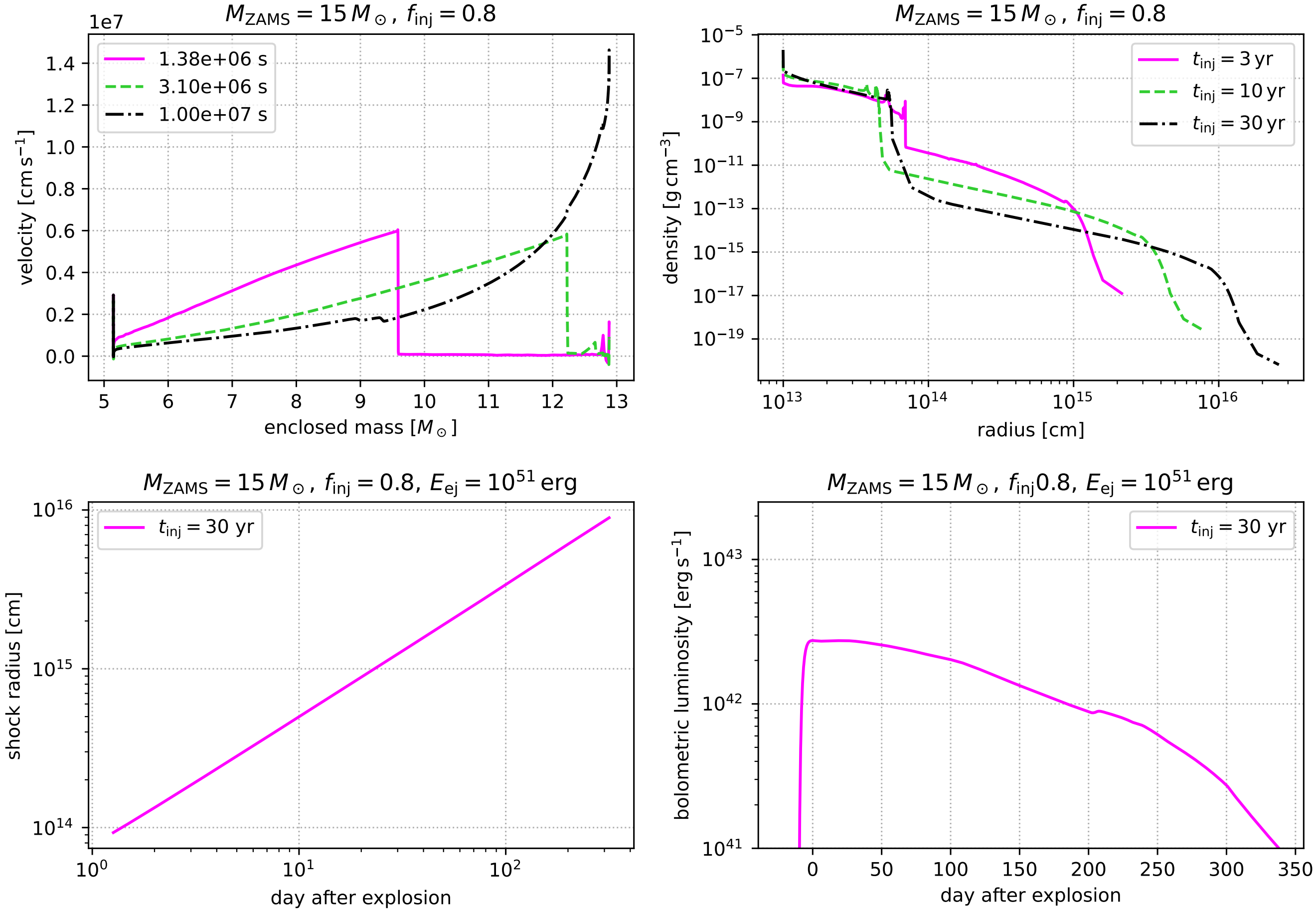

Figure 3. Sample outputs for the case of hydrogen-rich (IIn) progenitor.

- (Bottom-left): SN shock dynamics (shock radius as a function of time from SN explosion), plotted using `LCFiles/IIn_shock_output_...erg.txt`.
- (Bottom-right): SN light curves, plotted using `LCFiles/IIn_lightcurve_...erg.txt`.

5.2. *Case for stripped progenitor (Type Ibn/Icn supernovae)*

Here we model a SN Ibn by adopting a helium star progenitor,

- with ZAMS (Zero-Age Main Sequence) mass 20 Msun and metallicity 0.014,
- which experiences mass eruption 0.5 years before core-collapse, with energy injection of 0.5 times the envelope's binding energy,
- and finally explodes with energy 10^{51} erg.

we run the execution script `run.py` as follows:

```
python3 run.py --finj 0.5 --tinj 0.5 --stellar-model input/mesa_models_ibn/20Msun_Z0.014
    _preccsn.data --Eej 1e51 --injection-duration 10 --opacity-table input/rosseland/
    opacity_X0000Y0986Z0014.txt --eruption-Ncell 5000
```

Note that here in the eruption calculation, we have changed the duration of the energy injection to a value much shorter than the envelope's dynamical time ($\sim 10^3$ s), and used the opacity for helium-rich gas. The outputs are summarized in Figure 4, in a similar format as Figure 3.

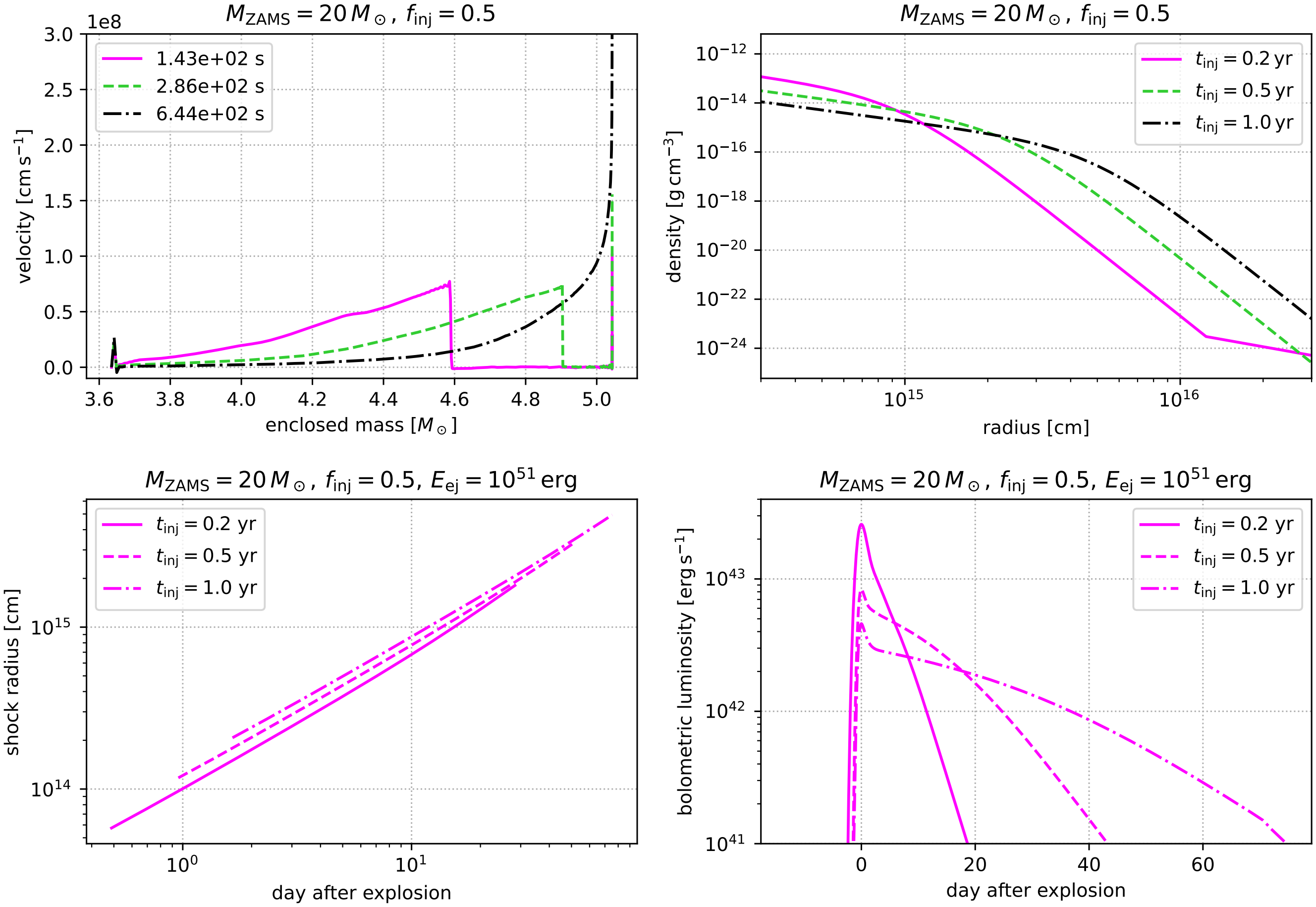

Figure 4. Sample outputs for the case of hydrogen-poor (Ibn) progenitor.

Acknowledgements

We thank Naoto Kuriyama (past developer) for the huge contributions for the eruption module. We also thank the users Ayako Ishii, Kazuma Kato, Craig Pellegrino for the feedback that helped continuously improve the code. The development of CHIPS project has been supported by JSPS KAKENHI grants 21J13957, 22K03671.

APPENDIX

A. GOVERNING EQUATIONS AND NUMERICAL DETAILS

A.1. *Mass Eruption Calculation*

The following (non-relativistic) hydrodynamical equations are solved in the stellar envelope, in Lagrangian coordinates:

$$\frac{\partial(1/\rho)}{\partial t} - \frac{\partial(4\pi r^2 v)}{\partial m} = 0, \tag{A1}$$

$$\frac{\partial v}{\partial t} + 4\pi r^2 \frac{\partial p}{\partial m} = g, \tag{A2}$$

$$\frac{\partial E}{\partial t} + \frac{\partial(4\pi r^2 v p)}{\partial m} = vg - \frac{\partial L}{\partial m}, \tag{A3}$$

where ρ is the mass density, t the time from energy injection, r the radius, v the radial velocity, m the enclosed mass, p the pressure, L the luminosity, E the total energy density of radiation, kinetic and internal energy, and $g = -Gm/r^2$ the gravitational acceleration.

These equations are solved using the piecewise parabolic method (Colella & Woodward 1984). In order to calculate L in each cell, we use the diffusion approximation with a flux limiter λ (Levermore & Pomraning 1981):

$$L = -\frac{16\pi^2 acr^4}{3\kappa_{\mathrm{R}}}\frac{\partial T^4}{\partial m}\lambda, \tag{A4}$$

$$\lambda = \frac{3(2+R)}{6+3R+R^2},\ \ R = \frac{\left|\frac{\partial aT^4}{\partial r}\right|}{\kappa_{\mathrm{R}}\rho aT^4}, \tag{A5}$$

where a is the radiation constant, c the speed of light, T the temperature, and κ_{R} is the Rosseland mean opacity. The energy conservation equation (A3) is integrated with respect to time in a partially implicit manner (Shigeyama & Nomoto 1990).

The opacity is by default calculated by the following analytical formula (with dimensional parameters in CGS units):

$$\kappa = \kappa_{\mathrm{molecular}} + \frac{1}{(\kappa_{\mathrm{H}^{-1}})^{-1} + (\kappa_{\mathrm{e}} + \kappa_{\mathrm{Kramers}})^{-1}}, \tag{A6}$$

$$\kappa_{\mathrm{molecular}} = 0.1Z, \tag{A7}$$

$$\kappa_{\mathrm{H}^{-1}} = 1.1 \times 10^{-25} Z^{0.5}\rho^{0.5}T^{7.7}, \tag{A8}$$

$$\kappa_{\mathrm{e}} = \frac{0.2\,(1+X)}{\left(1 + 2.7\times 10^{11}\frac{\rho}{T}\right)\left[1 + \left(\frac{T}{4.5\times 10^{8}}\right)^{0.86}\right]}, \tag{A9}$$

$$\kappa_{\mathrm{Kramers}} = 4\times 10^{25}\,(1+X)\,(Z+0.001)\,\frac{\rho}{T^{3.5}}, \tag{A10}$$

where $\kappa_{\mathrm{molecular}}$ denotes the molecular opacity, $\kappa_{\mathrm{H}^{-1}}$ the hydrogen opacity, κ_{e} the electron scattering opacity, and $\kappa_{\mathrm{Kramers}}$ Kramers' opacity for absorption regarding free-free, bound-free, and bound-bound transitions. Parameters X and Z are the mass fractions of hydrogen and elements heavier than helium. The user can also supply a manual opacity table as described in Section 3.1.2, which is recommended for hydrogen-free progenitors where this formula is not calibrated for.

The HELMHOLTZ equation of state (Timmes & Swesty 2000) is used to calculate thermodynamic quantities from the internal energy density and ρ, which are obtained from the time integration of the hydrodynamical equations. For the range not covered by this equation of state, the pressure is assumed to be the sum of ideal gas and radiation, i.e.,

$$p = \frac{\mathcal{R}}{\mu}\rho T + \frac{a}{3}T^4, \tag{A11}$$

where $\mathcal{R}$ is the gas constant, and μ the mean molecular weight obtained assuming the gas is fully ionized.

When solving the hydrodynamical equations, the boundary conditions are given as follows:

$$v_{\mathrm{inner}} = 0,\ r_{\mathrm{inner}} = \mathrm{constant},\ p_{\mathrm{outer}} = 0. \tag{A12}$$

Here the subscript inner (outer) indicates quantities at the base (outer edge) of the envelope.

The energy is injected at the inner boundary (either specified by the user, or by default 0.2 $M_\odot$ outside the inner core; Sec 4.1) at a constant rate as

$$\frac{\mathrm{d}E}{\mathrm{d}t} = \frac{E_{\mathrm{inj}}}{\Delta t_{\mathrm{inj}}} \equiv \frac{f_{\mathrm{inj}}E_{\mathrm{outer}}}{\Delta t_{\mathrm{inj}}}, \tag{A13}$$

where E_{inj} is the total injected energy, and Δt_{inj} is the energy injection timescale with a default value of 1000 seconds. The energy is equally injected in the innermost 10 cells of the computational region. The energy is characterized in this code by the dimensionless parameter f_{inj}, which is the ratio of E_{inj} to the initial total binding energy of the outer envelope E_{outer}. Values of $f_{\mathrm{inj}} \gtrsim 0.2$ generally lead to a fraction of the envelope being unbound (Ko et al. 2022), and $f_{\mathrm{inj}} \approx 1$ leads to most of the envelope being unbound.

The mass coordinates of the Lagrangian cells can either be uniform spacing, or logarithmic spacing with finer meshing outside with an optical depth of $0.1(\kappa/1\ \mathrm{cm}^2\ \mathrm{g}^{-1})$ in the outermost cell. For IIn progenitors, we use the uniform spacing. For Ibn/Icn stars with more compact radii, we use the logarithmic spacing to fully resolve the photosphere close to the surface.

A.2. *SN Ejecta-CSM Interaction Calculation*

A.2.1. *Parameter Setup*

For the SN ejecta, we assume homologously expanding ejecta with a double power-law density profile (Matzner & McKee 1999),

$$\rho_{\rm ej}(r,t) = \begin{cases} t^{-3}\left[r/(gt)\right]^{-n} & (r/t > v_t), \\ t^{-3}(v_t/g)^{-n}\left[r/(tv_t)\right]^{-\delta} & (r/t < v_t), \end{cases} \tag{A14}$$

with δ fixed to 1. The constants g and v_t are functions of the ejecta mass $M_{\rm ej}$ and energy $E_{\rm ej}$ as

$$g = \left\{\frac{1}{4\pi(n-\delta)}\frac{[2(5-\delta)(n-5)E_{\rm ej}]^{(n-3)/2}}{[(3-\delta)(n-3)M_{\rm ej}]^{(n-5)/2}}\right\}^{1/n}, \ v_t = \left[\frac{2(5-\delta)(n-5)E_{\rm ej}}{(3-\delta)(n-3)M_{\rm ej}}\right]^{1/2}. \tag{A15}$$

The value of the outer exponent n is determined by first obtaining a “global” polytropic index $N_{\rm pol}$ inside the envelope at core-collapse that best fits the relation $p = K\rho^{1+1/N_{\rm pol}}$, where constants K and $N_{\rm pol}$ are the fitting parameters. Then we use a formula connecting $N_{\rm pol}$ and n as (Matzner & McKee 1999; equation 25)

$$n = \frac{N_{\rm pol}+1+\beta N_{\rm pol}}{\beta N_{\rm pol}}, \tag{A16}$$

where $\beta \approx 0.19$ only weakly depends on $N_{\rm pol}$.

For the CSM profile, our default LC code models the density profile relying on an analytical fitting formula for the density profile of CSM created from mass eruption, obtained in Tsuna et al. (2021); Tsuna & Takei (2023) using a suite of eruption simulations,

$$\rho_{\rm CSM}(r) = \rho_* \left[\frac{(r/r_*)^{1.5/y}+(r/r_*)^{n_{\rm max}/y}}{2}\right]^{(-y)}, \tag{A17}$$

where r_*, ρ_*, y, $n_{\rm max}$ are the fitting parameters. When using this formula, the code fits the outer (marginally and highly unbound) region of the CSM obtained by the eruption calculation, and extrapolates the fit inwards to $r_{\rm in} = 2R_*$, where $r_{\rm in}$ denotes the innermost radius of the CSM. This choice of the inner boundary is rather arbitrary; in our case it is motivated from the assumption of homologous SN ejecta, which is valid only after the ejecta expands roughly twice its initial radius.

Outside the erupted material, the code adds by hand CSM composed of a stellar wind emitted by the progenitor before the eruptive mass loss. The wind is assumed to have a steady wind profile $\rho = \dot{M}/4\pi r^2 v_w$, that is continuous from the outermost cell of the erupted matter and extends out to 3×10^{16} cm. This prescription is purely for extending the computational region, and the resulting $\dot{M}/v_w$ is rather arbitrary and does not necessarily match the commonly seen empirical mass-loss formulae (which is also rather uncertain). Since the outermost cell is very low in density and is well optically thin, the details of this do not affect the major characteristics of the final LC.

A.2.2. *Supernova Shock Dynamics*

The code first calculates the shock structure as a function of radius r for each of the shocks, by the following equations at the shock rest frame that assumes steady state.

$$\frac{\partial(r^2\rho v)}{\partial r} = 0, \tag{A18}$$

$$v\frac{\partial v}{\partial r} + \frac{1}{\rho}\frac{\partial p}{\partial r} = 0, \tag{A19}$$

$$\frac{\partial}{\partial r}\left[r^2\left\{\rho v\left(\frac{1}{2}v^2 + e + \frac{p}{\rho}\right) + F\right\}\right] = 0. \tag{A20}$$

For simplicity we assume gas and radiation are in thermal equilibrium, and set the pressure p and specific internal energy e as a function of ρ and temperature T $(= T_{\rm g} = T_{\rm r})$ as

$$p = \frac{\rho}{\mu m_u}k_B T + \frac{1}{3}aT^4 \tag{A21}$$

$$\rho e = \frac{3}{2}\frac{\rho}{\mu m_u}k_B T + aT^4, \tag{A22}$$

where μ, m_u, k_B, a, $T_{\rm g}$, $T_{\rm r}$ are respectively the mean molecular weight, the atomic mass unit, the Boltzmann constant, the radiation constant, the gas temperature, and the radiation temperature. The radiative flux $F = L/4\pi r^2$ is calculated by flux-limited diffusion as in Equation (A5). We use the TOPS opacity table, which tabulates κ_R as a function of density and temperature.

To solve the hydrodynamical equations (A18)-(A20), we first set boundary conditions at the immediate downstream of each shock by the jump conditions, which can be written in the rest frame of the shock as

$$\rho_{\rm down}\upsilon_{\rm down} = \rho_{\rm up}\upsilon_{\rm up}, \tag{A23}$$

$$\rho_{\rm down}\upsilon_{\rm down}^2 + p_{\rm down} = \rho_{\rm up}\upsilon_{\rm up}^2 + p_{\rm up}, \tag{A24}$$

$$\frac{1}{2}\upsilon_{\rm down}^2 + e_{\rm down} + \frac{p_{\rm down}}{\rho_{\rm down}} = \frac{1}{2}\upsilon_{\rm up}^2 + e_{\rm up} + \frac{p_{\rm up}}{\rho_{\rm up}}, \tag{A25}$$

where the subscripts down (up) denote variables at the downstream (upstream) of the shock. The upstream density $\rho_{\rm up}$ for the forward (reverse) shock is given from the density profiles in the CSM (ejecta), and $p_{\rm up}, e_{\rm up}$ are assumed to be 0 for both shocks.

There are three unknown parameters in the equations: the velocities of the two shocks and the radiative flux at the forward shock. The radiative flux at the reverse shock front is set to only those from radioactive decay of ^{56}Ni/^{56}Co that escapes from the ejecta, since the reverse shock dissipates the kinetic energy into radiation much weaker than the forward shock. We derive these three values so that the velocity, pressure, and flux become continuous at the contact discontinuity by integrating the equations iteratively. Then we obtain the positions of the shocks at the next time step as $r_{\rm rs(fs)}(t + dt) = r_{\rm rs(fs)}(t) + u_{\rm rs(fs)}(t)dt$.

The assumption of a steady-state shock is violated for cases of very dense CSM when shock breakout occurs inside the CSM and much outside the star. In these cases, before shock breakout the photon diffusion is slower than the expansion of the shocked shell. In these cases, we thus set the initial time of our shock calculation $t_{\rm ini}$ at when the shock reaches a radii where diffusion and dynamical timescales are comparable,

$$\tau_{\rm CSM}(r > r_{\rm sh})\frac{r_{\rm sh}}{c} = \frac{r_{\rm sh}}{\upsilon_{\rm sh}}, \tag{A26}$$

where $r_{\rm sh}$, $\upsilon_{\rm sh}$, $\tau_{\rm CSM}$ are respectively the shock radius, shock velocity, and the optical depth of the CSM outside $r_{\rm sh}$. The mapping between $r_{\rm sh}$ and $t_{\rm ini}$ are done using a self-similar solution for shock propagation $(r_{\rm sh}(t), \upsilon_{\rm sh}(t))$, applicable before shock breakout when the shock is adiabatic (Chevalier 1982). The solution depends on the CSM density profile within the breakout radius, γ, and n. We assume a CSM profile of $\rho \propto r^{-1.5}$ and adiabatic index of $\gamma = 4/3$, and tabulate the dimensionless proportionality factor (A in Chevalier 1982) for various values of n. Using the n from our SN ejecta model, we obtain A through interpolation.

When we conduct radiative transfer, we include the contribution of photons generated in the shocked region before breakout and escapes the shocked region upon breakout. This contribution is added to the forward shock flux $F(r_{\rm fs})$ in the form (Arnett 1980; Smith & McCray 2007)

$$F_{\rm bo}(t) \propto \frac{E_{\rm int,\,bo}}{4\pi r_{\rm sh}(t)^2 t_{\rm ini}} \exp\left[-\frac{t}{t_{\rm ini}} - \frac{1}{2}\left(\frac{t}{t_{\rm ini}}\right)^2\right]. \tag{A27}$$

In the pre-breakout adiabatic phase (Chevalier 1982), the internal energy stored in the shocked region at time t is proportional to the ejecta's kinetic energy dissipated up to time t, whose proportionality factor can be obtained from the self-similar solution. We similarly tabulate this proportionality factor for various values of n (Chevalier 1982), again assuming $\rho \propto r^{-1.5}$ and $\gamma = 4/3$. For a given value of n, we obtain $E_{\rm int,\,bo}$ through interpolation.

A.2.3. *Light Curve Calculation*

By solving equations for the shock structure at each time step, we obtain the flux F at the forward shock as a function of time. To obtain the thermal structure inside the unshocked CSM with this flux and the LCs observed by a distant observer, `CHIPS` solves the following two-temperature radiative transfer and energy equations for $E(= aT_{\rm r}^4)$, and $U\,(= 3k_B T_{\rm g}/(2\mu m_{\rm u}))$,

$$\frac{\partial E}{\partial t} + \frac{\partial(r^2 F)}{r^2 \partial r} = \kappa_{\rm a}\rho c(aT_{\rm g}^4 - E), \tag{A28}$$

$$\left(\frac{\partial}{\partial t} + \upsilon\frac{\partial}{\partial r}\right)U + p_{\rm g}\upsilon\frac{\partial \rho^{-1}}{\partial r} = \kappa_{\rm a}c(E - aT_{\rm g}^4), \tag{A29}$$

where U denotes the specific internal energy of the gas, $p_{\rm g}$ the gas pressure, and $\kappa_{\rm a}$ the Planck mean absorption opacity.

The boundary condition for the radiative flux is obtained as follows. Integrating equation (A28) over the volume enclosed between $r = r_{\rm fs}$ and $r = r_{\rm out}$ yields

$$\int_{r_{\rm fs}}^{r_{\rm out}} \frac{\partial(4\pi r^2 E)}{\partial t} dr \simeq 4\pi r_{\rm fs}^2 F(r_{\rm fs}) - 4\pi r_{\rm out}^2 F(r_{\rm out}), \tag{A30}$$

where the source term is omitted. From this equation, we obtain the time derivative of the total radiation energy in this volume as

$$\begin{aligned} \frac{\partial}{\partial t}\int_{r_{\rm fs}}^{r_{\rm out}} 4\pi r^2 E dr &= \int_{r_{\rm fs}}^{r_{\rm out}} \frac{\partial(4\pi r^2 E)}{\partial t} dr - 4\pi r_{\rm fs}^2 E(r_{\rm fs}) u_{\rm fs} \\ &\simeq -4\pi r_{\rm out}^2 F(r_{\rm out}) + 4\pi r_{\rm fs}^2 (F(r_{\rm fs}) - E(r_{\rm fs}) u_{\rm fs}), \end{aligned} \tag{A31}$$

Since the last term $F(r_{\rm fs}) - E(r_{\rm fs}) u_{\rm fs}$ denotes the flux at the shock front, this should be equal to F at the forward shock (Otherwise the radiation energy would be lost or generated at the shock). Thus we obtain

$$F(r_{\rm fs}) = F + E(r_{\rm fs}) u_{\rm fs}, \tag{A32}$$

as the boundary condition.

REFERENCES

Arnett, W. D. 1980, ApJ, 237, 541, doi: 10.1086/157898

Chevalier, R. A. 1982, ApJ, 258, 790, doi: 10.1086/160126

Colella, P., & Woodward, P. R. 1984, Journal of Computational Physics, 54, 174, doi: 10.1016/0021-9991(84)90143-8

Hachinger, S., Mazzali, P. A., Taubenberger, S., et al. 2012, MNRAS, 422, 70, doi: 10.1111/j.1365-2966.2012.20464.x

Hiramatsu, D., Tsuna, D., Berger, E., et al. 2023, ApJL, 955, L8, doi: 10.3847/2041-8213/acf299

Iglesias, C. A., & Rogers, F. J. 1996, ApJ, 464, 943, doi: 10.1086/177381

Ishii, A. T., Takei, Y., Tsuna, D., Shigeyama, T., & Takahashi, K. 2024, ApJ, 961, 47, doi: 10.3847/1538-4357/ad072b

Jermyn, A. S., Bauer, E. B., Schwab, J., et al. 2023, ApJS, 265, 15, doi: 10.3847/1538-4365/acae8d

Ko, T., Tsuna, D., Takei, Y., & Shigeyama, T. 2022, ApJ, 930, 168, doi: 10.3847/1538-4357/ac67e1

Kuriyama, N., & Shigeyama, T. 2020, A&A, 635, A127, doi: 10.1051/0004-6361/201937226

Levermore, C. D., & Pomraning, G. C. 1981, ApJ, 248, 321, doi: 10.1086/159157

Marigo, P., & Aringer, B. 2009, A&A, 508, 1539, doi: 10.1051/0004-6361/200912598

Marigo, P., Aringer, B., Girardi, L., & Bressan, A. 2022, ApJ, 940, 129, doi: 10.3847/1538-4357/ac9b40

Matzner, C. D., & McKee, C. F. 1999, ApJ, 510, 379, doi: 10.1086/306571

Paxton, B., Bildsten, L., Dotter, A., et al. 2011, ApJS, 192, 3, doi: 10.1088/0067-0049/192/1/3

Paxton, B., Cantiello, M., Arras, P., et al. 2013, ApJS, 208, 4, doi: 10.1088/0067-0049/208/1/4

Paxton, B., Marchant, P., Schwab, J., et al. 2015, ApJS, 220, 15, doi: 10.1088/0067-0049/220/1/15

Paxton, B., Schwab, J., Bauer, E. B., et al. 2018, ApJS, 234, 34, doi: 10.3847/1538-4365/aaa5a8

Paxton, B., Smolec, R., Schwab, J., et al. 2019, ApJS, 243, 10, doi: 10.3847/1538-4365/ab2241

Schneider, F. R. N., Podsiadlowski, P., & Müller, B. 2021, A&A, 645, A5, doi: 10.1051/0004-6361/202039219

Shigeyama, T., & Nomoto, K. 1990, ApJ, 360, 242, doi: 10.1086/169114

Smith, N., & McCray, R. 2007, ApJL, 671, L17, doi: 10.1086/524681

Takei, Y., & Shigeyama, T. 2020, PASJ, 72, 67, doi: 10.1093/pasj/psaa050

Takei, Y., Tsuna, D., Ko, T., & Shigeyama, T. 2024, ApJ, 961, 67, doi: 10.3847/1538-4357/ad0da4

Takei, Y., Tsuna, D., Kuriyama, N., Ko, T., & Shigeyama, T. 2022, ApJ, 929, 177, doi: 10.3847/1538-4357/ac60fe

Timmes, F. X., & Swesty, F. D. 2000, ApJS, 126, 501, doi: 10.1086/313304

Tsuna, D., & Takei, Y. 2023, PASJ, 75, L19, doi: 10.1093/pasj/psad041

Tsuna, D., Takei, Y., Kuriyama, N., & Shigeyama, T. 2021, PASJ, 73, 1128, doi: 10.1093/pasj/psab063

Tsuna, D., Takei, Y., & Shigeyama, T. 2023, ApJ, 945, 104, doi: 10.3847/1538-4357/acbbc6

Williamson, M., Kerzendorf, W., & Modjaz, M. 2021, ApJ, 908, 150, doi: 10.3847/1538-4357/abd244