Modelling low Mach number stellar hydrodynamics with MAESTROeX

A. Harpole[1], D. Fan[2], M. P. Katz[3], A. J. Nonaka[2], D. E. Willcox[2], and M. Zingale[1]

[1]Department of Physics and Astronomy, Stony Brook University, Stony Brook, NY 11794-3800 USA

[2]Center for Computational Sciences and Engineering, Lawrence Berkeley National Lab, Berkeley, CA 94720 USA

[3]NVIDIA Corporation, 2788 San Tomas Expressway, Santa Clara, CA, 95050 USA

E-mail: alice.harpole@stonybrook.edu

Abstract. Modelling long-time convective flows in the interiors of stars is extremely challenging using conventional compressible hydrodynamics codes due to the acoustic timestep limitation. Many of these flows are in the low Mach number regime, which allows us to exploit the relationship between acoustic and advective time scales to develop a more computationally efficient approach. MAESTROeX is an open source low Mach number stellar hydrodynamics code that allows much larger timesteps to be taken, therefore enabling systems to be modelled for much longer periods of time. This is particularly important for the problem of convection in the cores of rotating massive stars prior to core collapse. To fully capture the dynamics, it is necessary to model these systems in three dimensions at high resolution over many rotational periods. We present an overview of MAESTROeX's current capabilities, describe ongoing work to incorporate the effects of rotation and discuss how we are optimising the code to run on GPUs.

1. Introduction

For many flows in astrophysical systems, the magnitude of the fluid velocity is much less than the soundspeed. Consequently, the Mach number, the ratio of the fluid velocity to the sound speed, is much less than 1: $\mathrm{Ma} = |\boldsymbol{U}|/c_s \ll 1$. Such low Mach number flows are challenging to model with standard compressible schemes using explicit timestepping methods, where the maximum size of the timestep is determined by the Courant-Friedrich-Lewy (CFL) condition. This states that for a grid with cell spacing Δx, the timestep $\Delta t < \Delta x / \max(|\boldsymbol{U}| + c_s)$. For low Mach number flows, this condition is dominated by the contributions of the sound speed, with the result that many fine timesteps are required when using high spatial resolution. Long timescale, high resolution simulations therefore become extremely computationally expensive.

This restriction can be lifted by using sound-proof methods. These methods can involve modifying the fluid equations, modifying the computational algorithm and/or modifying the flow parameters in order to allow much larger timesteps to be used. In MAESTROeX, we use the *low Mach number approximation*, a limit of the compressible Euler equations which effectively filters out sound waves. An overview of this shall be given in section 2, with further details of the MAESTROeX code and our development workflow given in section 3 and section 4.

An example of an astrophysical low Mach number flow is convection within the interiors of massive stars. Prior to core collapse, the interiors of massive stars consist of a sequence of convective burning shells, separated by inert, non-convective shells. Moving from the outermost layers inwards, these shells consist of heavier and heavier elements, with the core burning elements up to ^{56}Fe. An accurate description of the structure of such stars in the minutes before core collapse is important for supernova modelling. The composition and structure of the star prior to collapse provides the initial data for these models, so this needs to be accurate if the resulting supernova models are to be trusted. Modelling this convection is challenging using conventional compressible schemes because the domain size is very large (i.e. a significant fraction of the interior of the entire star), high resolution is required in order to properly capture the turbulent mixing that occurs at shell boundaries, and long time periods are required (multiple convective turnover times). In section 5, we shall describe ongoing work to model this problem in MAESTROeX, in particular describing how we are adding rotation to our existing scheme.

The latest generation of supercomputers coming online are relying more and more on GPU architectures in order to increase performance whilst minimizing power requirements. For HPC codes to best exploit the most powerful machines, it is therefore becoming increasingly necessary to port these codes so that they can run on GPUs. In section 6, we describe our work to port MAESTROeX to run on GPUs and demonstrate its performance.

2. Low Mach number hydrodynamics

Sound-proof methods for modelling low Mach number flows can take a variety of forms. One technique is to use preconditioners in order to reduce the stiffness of the equations, allowing larger timesteps to be used [1, 2]. Another technique is to modify the flow parameters, artificially boosting the speed of the flow without changing its behaviour so that the system evolves faster and fewer timesteps are required [3, 4]. Fully implicit time integration codes, such as those used by [5–7], are no longer restricted by the CFL condition and so can use arbitrarily large timesteps.

The technique that we use in MAESTROeX is to modify the fluid equations themselves so as to filter out the soundwaves. This is a similar approach to the incompressible [8] and anelastic [9–11] approximations, however the approximation that we use (called the low Mach number approximation [12–14] and the generalised pseudo-incompressible approximation [15]) allows for background stratification and large density and temperature perturbations due to heating and changes in composition. This is achieved by decomposing the pressure into a one-dimensional hydrostatic base state, $p_0 = p_0(r,t)$, and a dynamic pressure perturbation, $\pi = \pi(\boldsymbol{x},t)$, such that the full pressure is given by $p(\boldsymbol{x},t) = p_0(r,t) + \pi(\boldsymbol{x},t)$. In the low Mach number regime, asymptotic analysis shows that $|\pi|/p_0 = O(\mathrm{Ma}^2)$.

The low Mach number fluid equations are given by

$$\frac{\partial(\rho X_k)}{\partial t} = -\nabla\cdot(\rho X_k \boldsymbol{U}) + \rho\dot{\omega}_k, \tag{1}$$

$$\frac{\partial \boldsymbol{U}}{\partial t} = -\boldsymbol{U}\cdot\nabla\boldsymbol{U} - \frac{\beta_0}{\rho}\nabla\left(\frac{\pi}{\beta_0}\right) - \frac{\rho-\rho_0}{\rho}g\boldsymbol{e}_r, \tag{2}$$

$$\frac{\partial(\rho h)}{\partial t} = -\nabla\cdot(\rho h \boldsymbol{U}) + \frac{\mathrm{D}p_0}{\mathrm{D}t} + \rho H_{\mathrm{nuc}}, \tag{3}$$

where ρ, $\boldsymbol{U}$ and h are the mass density, fluid velocity and specific enthalpy, X_k and $\dot{\omega}_k$ are the species mass fraction and production rate of species k, and H_{nuc} is the energy release per time per unit mass. MAESTROeX defines the base state pressure p_0 to be consistent with the one-dimensional base state density, $\rho_0 = \rho_0(r,t)$, which represents the lateral average and is in hydrostatic equilibrium with p_0:

$$\nabla p_0 = -\rho_0 g\boldsymbol{e}_r, \tag{4}$$

where $g = g(r,t)$ is the magnitude of the gravitational acceleration, and $\boldsymbol{e}_r$ is the radial unit vector. The background stratification is captured by introducing a buoyancy-like term, β_0. It is defined as

$$\beta_0(r,t) = \rho_0(0,t) \exp\left(\int_0^r dr' \frac{1}{\overline{\Gamma}_1 p_0} \frac{\partial p_0}{\partial r'}\right), \tag{5}$$

where $\overline{\Gamma}_1$ is the lateral average of the first adiabatic exponent, $\Gamma_1 \equiv d(\ln p)/d(\ln \rho)|_s$, and s is the entropy. The equations are closed by casting the equation of state (EoS) as a velocity divergence constraint. This is done by taking the Lagrangian derivative of the EoS for pressure as a function of the thermodynamic variables, substituting in the equations of motion for mass and energy, and requiring that the pressure is described by a function of r and t based on the condition of hydrostatic equilibrium. Details of this derivation can be found in [13, 16]. This constraint is given by

$$\nabla \cdot (\beta_0 \boldsymbol{U}) = \beta_0 \left(S - \frac{1}{\overline{\Gamma}_1 p_0} \frac{\partial p_0}{\partial t}\right). \tag{6}$$

Here, S is an expansion term which describes local compressibility effects due to changes in composition and heating from reactions:

$$S = -\sigma \sum_k \xi_k \dot{\omega}_k + \frac{1}{\rho p_\rho} \sum_k p_{X_k} \dot{\omega}_k + \sigma H_{\text{nuc}}, \tag{7}$$

where we define the following thermodynamic quantities as

$$p_{X_k} \equiv \left.\frac{\partial p}{\partial X_k}\right|_{\rho,T,X_{j,j\neq k}}, \qquad \xi_k \equiv \left.\frac{\partial h}{\partial X_k}\right|_{p,T,X_{j,j\neq k}}, \qquad p_\rho \equiv \left.\frac{\partial p}{\partial \rho}\right|_{T,X_k},$$
$$\sigma \equiv \frac{p_T}{\rho c_p p_\rho}, \qquad p_T \equiv \left.\frac{\partial p}{\partial T}\right|_{\rho,X_k} \quad \text{and} \quad c_p \equiv \left.\frac{\partial h}{\partial T}\right|_{p,X_k}.$$

3. MAESTROeX

Prior to MAESTROeX, we developed the low Mach number code Maestro. Like MAESTROeX, Maestro is a block-structured adaptive mesh refinement (AMR) code for modelling low Mach number astrophysical codes. Its development is described in a series of papers: [13, 14, 16–18]. In Maestro, the system of low Mach number equations is solved using an explicit Godunov approach for the advection, a stiff ODE solver for the reactions (VODE [19]), and multigrid linear solvers for the pressure projection steps. Strang splitting [20] is used to integrate the thermodynamic variables, a second order projection method to integrate the velocity subject to the divergence constraint, and a velocity splitting scheme to hydrodynamically evolve the base state. The original Maestro code was developed using the Fortran 90 interface of the BoxLib software framework [21]; MAESTROeX instead uses the C++/Fortran 90 AMReX framework [22].

Maestro has been used to model a number of astrophysical systems, including convection in white dwarfs prior to type Ia supernovae [23–27], convection in massive stars [28] and type I X-ray bursts [29–31].

The numerical algorithm implemented in MAESTROeX improves upon the original Maestro algorithm in a number of ways. The temporal integration method has been greatly simplified without compromising the second order accuracy, and a new spherical base state mapping has been implemented in order to reduce mapping errors between spherical and Cartesian grids. Utilising the AMReX framework has allowed us to implement MPI+OpenMP (with tiling [21]) parallelism, which has been shown to scale well to over 10,000 MPI processes.

Further details of the algorithm implemented in MAESTROeX and its performance can be found in [32].

4. AMReX-Astro development workflow

MAESTROeX is part of the AMReX-Astro suite of open source adaptive mesh refinement hydrodynamics codes for astrophysical flows. Other codes in this family include Castro, an astrophysical radiation hydrodynamics simulation code [33], and Nyx, an N-body hydrodynamics cosmological simulation code [34]. All three codes are developed using the AMReX software framework, and share much in common in terms of their development, structure and numerical methods. Castro and MAESTROeX share a common set of microphysics solvers provided by the Starkiller Microphysics library[1]. Developers of the different codes work closely together (in fact many of the developers work on more than one of the codes), with their shared structure and framework meaning that new features developed in one code can easily be replicated in the others. A prime example of this is the GPU porting capability that has recently been implemented in Castro, and is currently in the process of being implemented in MAESTROeX.

All codes in the AMReX-Astro suite are open source, with all development carried out in public repositories hosted on GitHub[2]. We believe that having both the codes and the development process completely open promotes good scientific practices, as it means that results from our simulations are reproducible and the codes used to produce them can be examined and improved by the community. Others can use the codes for their own scientific investigations, and it is possible to adapt all or parts of the codes to suit new problems. Using version control allows us to keep a record of the codes' development process, helping us to track down bugs and means that new and existing developers can learn from previous mistakes. The development branches of the codes are tested nightly using a test suite of problems, checking that any new additions to the code have not significantly changed any of the solutions or slowed down the performance.

New versions of the codes are released on the first of each month. In the case of Castro, these versions are then archived on Zenodo[3]. This also provides us with DOIs (digital object identifiers), further enhancing the reproducibility of our results and ensuring the sustainability of the codes used to produce them. We intend to extend this archiving procedure to the other AMReX-Astro codes in the near future.

5. Rotation

MAESTROeX is currently able to model spherically symmetric systems, however it has no support for rotation which breaks the spherical symmetry. We wish to model convection in the interiors of massive stars, and it is known that these stars often have non-negligible rotational frequencies and that this rotation can have have significant effect on mixing at convective shell boundaries. We are therefore currently exploring several possible ways of implementing rotation in MAESTROeX.

As described in [23], to incorporate rotation in our equation set, we add the Coriolis and/or centrifugal terms to the velocity evolution equation:

$$\frac{\partial \boldsymbol{U}}{\partial t} = -\boldsymbol{U}\cdot\nabla\boldsymbol{U} - \frac{\beta_0}{\rho}\nabla\left(\frac{\pi}{\beta_0}\right) - \frac{\rho-\rho_0}{\rho}g\boldsymbol{e}_r - 2\boldsymbol{\Omega}\times\boldsymbol{U} - \boldsymbol{\Omega}\times(\boldsymbol{\Omega}\times\boldsymbol{r}), \tag{8}$$

where $\boldsymbol{\Omega}$ is the angular velocity, $\boldsymbol{F}_{\text{Coriolis}} = -2\boldsymbol{\Omega}\times\boldsymbol{U}$ is the Coriolis force and $\boldsymbol{F}_{\text{centrifugal}} = -\boldsymbol{\Omega}\times(\boldsymbol{\Omega}\times\boldsymbol{r})$ the centrifugal force. Note that for slowly rotating systems it is a reasonable approximation to ignore the centrifugal force.

Another way that rotation could be incorporated would be to introduce a new 'rotational pressure term', p_1, in order to balance the centrifugal potential:

$$\nabla p_0 + \nabla p_1 = \rho\nabla\phi + \frac{1}{2}\rho\nabla\Omega^2 r^2 = \rho\nabla\Phi_{\text{eff}}, \tag{9}$$

[1] https://github.com/starkiller-astro/Microphysics

[2] https://github.com/AMReX-Astro

[3] https://zenodo.org/

where $\phi = \phi(r)$ is the gravitational potential (so $\nabla\phi(r) = -g\boldsymbol{e}_r$), Ω is the rotational frequency, and Φ_{eff} is the new effective gravitational potential.

In MAESTROeX, we currently write the one-dimensional base state as a function of the radial coordinate, r. For a spherical system, this therefore assumes the base state to be spherically symmetric. However, a star that is rotating at a significant rate will no longer be spherical: it will instead become oblique, bulging around the equator. In order to capture the star's deformation without sacrificing the base state, we can rewrite the base state pressure as a function of the effective potential: $p_0(r) \rightarrow p_0(\Phi_{\text{eff}})$. We can then rewrite the other base state quantities and the system of low Mach number equations as functions of the new coordinates $(\Phi_{\text{eff}}, \boldsymbol{x}, t)$.

The massive stars that we are interested in modelling are believed to be relatively slow rotators, with the rotation rate in the core prior to collapse no more than a few percent of the Kepler frequency [35], so it is likely that one of the simpler approaches outlined above should be sufficient. However, for modelling e.g. type I X-ray bursts, it may be that a more sophisticated approach is needed. Type I X-ray bursts are produced by thermonuclear deflagrations in the liquid surface layers of fast rotating neutron stars. These stars typically rotate at frequencies of $\sim$300–600 Hz, a significant fraction of the stars' breakup velocity. This produces non-negligible deformation of the star about the equator, and may cause surface flows with a significant dependence on the meridional angle. To properly capture the effects of rotation in such a system, it may therefore be necessary to use a two-dimensional base state, with the base state pressure a function of both the radius and the meridional angle $p_0(r) \rightarrow p_0(r, \theta)$.

6. GPUs

The latest generation of supercomputers to come online (e.g. OLCF Summit, soon NERSC Perlmutter and OLCF Frontier) rely heavily on GPUs in order to achieve high performance while keeping energy consumption to a minimum (GPUs consume considerably less power per flop than CPUs). This trend is set to continue: supercomputers are moving to new architectures in order to increase performance rather than simply adding more and more CPUs. In order for HPC codes to be able to exploit these latest machines, it is therefore becoming increasingly necessary to port codes to run on GPUs.

We have begun porting MAESTROeX to GPUs, leveraging the GPU capabilities built into AMReX and following the lessons learned porting Castro to GPUs. In the AMReX framework, routines can be ported to GPUs by simply inserting a few macros into the code. The boilerplate code required to compile this for GPU is then generated by a custom preprocessor prior to compilation. CUDA managed memory is used to take care of data transfers between the CPU and GPU, so no explicit copies of data to/from the CPU and GPU are required, significantly reducing the extra code required to port the code. One of the key emphases of the AMReX approach to GPUs is to maintain performance portability: porting code to GPUs should not come at the expense of the code's performance on CPU-only machines. Because GPU-specific code is created by the preprocessor prior to compilation, it is not necessary to maintain separate versions of the code for different architectures. By changing the compiler flags, the same bit of code can be compiled for serial, MPI-only, OpenMP-only, MPI+OpenMP, GPU-only or MPI+GPU. In order to achieve the best performance on the GPU, we require some additional optimisation of the computational kernels. However, we find that these optimisations often improve the performance when the code is run on the CPU only as well. More details of the approach used to port our codes to GPU are to appear in a forthcoming AMReX-Astro GPU paper.

So far we have ported most of the source terms and hydrodynamics in MAESTROeX to run on GPUs. GPU support for the linear solvers is becoming increasingly available from the AMReX team. We have found that one of the challenges moving forwards is dealing with the one-dimensional base state. Currently, each MPI process holds a copy of the entire base state.

As the problem size and/or resolution gets larger, the size of the base state grows. Data transfers between the CPU and GPU are typically a slow process, so as the base state grows in size, the cost of these transfers becomes more significant and reduces the performance overall. As we are still in the process of porting our code to GPUs, there still remain some functions operating on the base state which run on the CPU. Every time one of these is called, if the necessary base state data is currently on the GPU, it is copied back to the CPU (and then back again next time it is required for a GPU-based calculation).

To alleviate this problem, we are looking at ways to reduce both the number and size of these data transfers. For example, in some routines we map the base state to the three-dimensional grid – for these, we can move this operation to the CPU so that the GPU does not require the entire 1d base state to be copied over. It may be necessary for us to develop a way to split up the base state, so that subgrids only receive the sections of the base state necessary for their calculations (rather than the base state for the entire problem domain).

As mentioned, our work porting MAESTROeX to run on GPUs is still ongoing, however in Figure 1 we show the speedup we have achieved so far for a number of individual functions that have been offloaded to GPU. The plot shows the average execution time of each function, recorded for the three-dimensional `reacting_bubble` problem on the OLCF Summit machine. This problem uses a simple reaction network modelling $^{12}\mathrm{C}+^{12}\mathrm{C} \rightarrow {}^{24}\mathrm{Mg}$ that can be integrated on the GPUs using the CUDA Fortran port of VODE [19] described in [36]. Both tests were run on a single node, consisting of two IBM POWER9 processors (together providing 42 physical CPU cores) and six NVIDIA Volta GPUs, and compiled using the PGI compiler version 19.4. The CPU test used 42 MPI processes, each with 4 OpenMP threads, and the GPU test used 6 MPI processes, each process associated with a single GPU. Note that the plot's x-axis scale is logarithmic. As can be seen, large speedups in excess of 30-40$\times$ were achieved for several of the functions when run on the GPU. Once we have finished porting the remaining routines to GPU and further optimised our code, we hope to achieve this sort of speedup for the entire timestep.

7. Summary

MAESTROeX is an open source code for modelling low Mach number astrophysical flows. It uses the low Mach number approximation, which allows it to use a much larger timestep than conventional compressible schemes by filtering out the soundwaves. Unlike other sound-proof methods, we still retain background stratification and large density and temperature perturbations due to local compositional changes and heating, both of which are particularly important for modelling atmospheres and burning.

MAESTROeX is a new and improved version of our previous code Maestro. It is based on AMReX (rather than FBoxLib), which allows us to exploit its powerful new solvers and make use of ongoing improvements. MAESTROeX is part of the AMReX-Astro suite of open source adaptive mesh refinement astrophysical hydrodynamics codes (which also includes Castro and Nyx). New features developed in other codes in the family can therefore be easily implemented in MAESTROeX. As development is open, results from our simulations can be reproduced, and monthly versioning ensures software sustainability.

We're currently working on implementing rotation in MAESTROeX. For modelling the interiors of massive stars (which rotate relatively slowly), we may be able to get away with neglecting the centrifugal force and simply add the Coriolis force as a source term. However, this will not be sufficient for faster rotating systems (e.g. for modelling Type I X-ray bursts on the surfaces of millisecond pulsars), so we are also exploring other methods including modelling the centrifugal force as an effective pressure term, rewriting the base state as a function of the effective gravitational potential and using a two-dimensional base state which is also a function of the meridional angle.

We're currently in the process of porting MAESTROeX to run on GPUs, using our experience

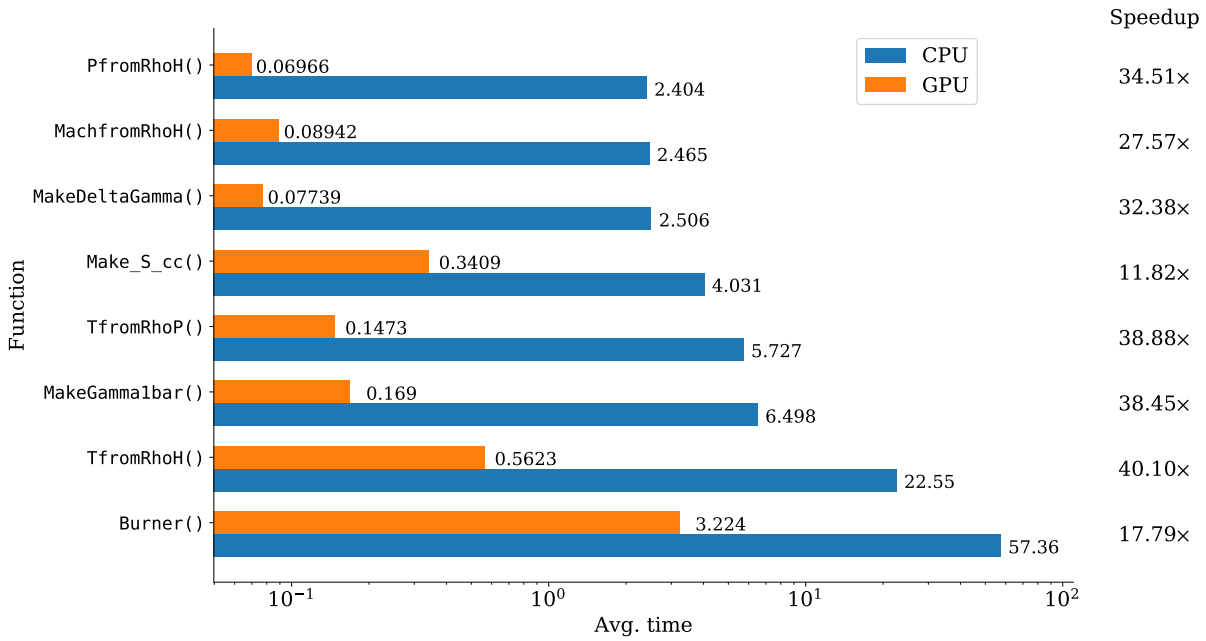

Figure 1. Comparison of the performance of functions on the CPU vs the GPU. The plot shows the average execution time for several functions, recorded for a run of the three-dimensional `reacting_bubble` problem on Summit. Both tests were performed using a single node, with the CPU test using 42 MPI processes, each with 4 OpenMP threads, and the GPU test using 6 MPI processes, each process associated with a single GPU. Note that the x-axis scale is logarithmic. On the right, we have listed the speedup for each function when run on GPUs.

doing the same for Castro. This will enable us exploit the latest supercomputer architectures. So far, we've ported source terms and are in the process of porting the hydro.

Work has recently begun on implementing the new time integration strategy, spectral deferred corrections (SDC) that has recently been implemented in Castro [37]. This method eliminates the coupling error between source terms and hydrodynamics incurred in operator splitting techniques. It is hoped that this shall be particularly useful for our simulations of rotating massive stars, where we have found energetic reactions in the star's core to be challenging to model using our current methods.

Acknowledgments

Figure 1 was generated using `Jupyter` [38] and `matplotlib` [39]. The work at Stony Brook was supported by the SciDAC program DOE grant de-sc0017955 and DOE/Office of Nuclear Physics grant DE-FG02-87ER40317. The work at LBNL was supported by the DOE Office of Advanced Scientific Computing Research under Contract No, DE-AC02-05CH11231. An award of computer time was provided by the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. This research has made use of NASA's Astrophysics Data System Bibliographic Services.

References

[1] Miczek F, Röpke F K and Edelmann P V 2015 *Astronomy & Astrophysics* **576** A50

[2] Barsukow W, Edelmann P V, Klingenberg C, Miczek F and Röpke F K 2017 *Journal of Scientific Computing* **72** 623–646

[3] Rempel M 2005 *The Astrophysical Journal* **622** 1320

[4] Hotta H, Rempel M, Yokoyama T, Iida Y and Fan Y 2012 *Astronomy & Astrophysics* **539** A30

[5] Viallet M, Baraffe I and Walder R 2011 *Astronomy & Astrophysics* **531** A86

[6] Viallet M, Goffrey T, Baraffe I, Folini D, Geroux C, Popov M, Pratt J and Walder R 2016 *Astronomy & Astrophysics* **586** A153

[7] Goffrey T, Pratt J, Viallet M, Baraffe I, Popov M, Walder R, Folini D, Geroux C and Constantino T 2017 *Astronomy & Astrophysics* **600** A7

[8] Boussinesq J 1903

[9] Ogura Y and Phillips N A 1962 *Journal of the atmospheric sciences* **19** 173–179

[10] Gough D 1969 *Journal of the atmospheric sciences* **26** 448–456

[11] Durran D R 1989 *Journal of the atmospheric sciences* **46** 1453–1461

[12] Day M S and Bell J B 2000 *Combustion Theory and Modelling* **4** 535–556

[13] Almgren A S, Bell J B, Rendleman C A and Zingale M 2006 *The Astrophysical Journal* **637** 922

[14] Nonaka A, Almgren A, Bell J, Lijewski M, Malone C and Zingale M 2010 *The Astrophysical Journal Supplement Series* **188** 358

[15] Vasil G M, Lecoanet D, Brown B P, Wood T S and Zweibel E G 2013 *The Astrophysical Journal* **773** 169

[16] Almgren A S, Bell J B, Rendleman C A and Zingale M 2006 *The Astrophysical Journal* **637** 922

[17] Almgren A, Bell J, Nonaka A and Zingale M 2008 *The Astrophysical Journal* **684** 449

[18] Zingale M, Almgren A, Bell J, Nonaka A and Woosley S 2009 *The Astrophysical Journal* **704** 196

[19] Brown P N, Byrne G D and Hindmarsh A C 1989 *SIAM journal on scientific and statistical computing* **10** 1038–1051

[20] Strang G 1968 *SIAM Journal on Numerical Analysis* **5** 506–517

[21] Zhang W, Almgren A, Day M, Nguyen T, Shalf J and Unat D 2016 *SIAM Journal on Scientific Computing* **38** S156–S172

[22] Zhang W *et al.* 2019 *Journal of Open Source Software* **4** 1370

[23] Zingale M, Nonaka A, Almgren A, Bell J, Malone C and Woosley S 2011 *The Astrophysical Journal* **740** 8

[24] Nonaka A, Aspden A, Zingale M, Almgren A, Bell J and Woosley S 2011 *The Astrophysical Journal* **745** 73

[25] Malone C, Nonaka A, Woosley S, Almgren A, Bell J, Dong S and Zingale M 2014 *The Astrophysical Journal* **782** 11

[26] Zingale M, Nonaka A, Almgren A, Bell J, Malone C and Orvedahl R 2013 *The Astrophysical Journal* **764** 97

[27] Jacobs A M, Zingale M, Nonaka A, Almgren A S and Bell J B 2016 *The Astrophysical Journal* **827** 84

[28] Gilet C, Almgren A, Bell J, Nonaka A, Woosley S and Zingale M 2013 *The Astrophysical Journal* **773** 137

[29] Malone C, Nonaka A, Almgren A, Bell J and Zingale M 2011 *The Astrophysical Journal* **728** 118

[30] Malone C, Zingale M, Nonaka A, Almgren A and Bell J 2014 *The Astrophysical Journal* **788** 115

[31] Zingale M, Malone C M, Nonaka A, Almgren A S and Bell J B 2015 *The Astrophysical Journal* **807** 60

[32] Fan D, Nonaka A, Almgren A S, Harpole A and Zingale M 2019 *arXiv preprint arXiv:1908.03634*

[33] Almgren A, Beckner V, Bell J, Day M, Howell L, Joggerst C, Lijewski M, Nonaka A, Singer M and Zingale M 2010 *The Astrophysical Journal* **715** 1221

[34] Almgren A S, Bell J B, Lijewski M J, Lukić Z and Van Andel E 2013 *The Astrophysical Journal* **765** 39

[35] Heger A, Woosley S, Langer N and Spruit H 2004 *Symposium-International Astronomical Union* vol 215 (Cambridge University Press) pp 591–600

[36] Zingale M, Almgren A S, Sazo M G B, Beckner V E, Bell J B, Friesen B, Jacobs A M, Katz M P, Malone C M, Nonaka A J, Willcox D E and Zhang W 2018 *Journal of Physics: Conference Series* **1031** 012024

[37] Zingale M, Katz M P, Bell J B, Minion M L, Nonaka A J and Zhang W 2019 *arXiv e-prints* arXiv:1908.03661 (*Preprint* `1908.03661`)

[38] Kluyver T, Ragan-Kelley B, Pérez F, Granger B E, Bussonnier M, Frederic J, Kelley K, Hamrick J B, Grout J, Corlay S *et al.* 2016 *ELPUB* pp 87–90

[39] Hunter J D 2007 *Computing in science & engineering* **9** 90