PyUltraLight: A Pseudo-Spectral Code for the Dynamics of Ultra-Light Dark Matter

F. Irst,^{a,b,1} **S. Econd,**^{c} **T. Hird**^{a,2} **and Fourth**^{a,2}

^{a}One University,
  some-street, Country
^{b}Another University,
  different-address, Country
^{c}A School for Advanced Studies,
  some-location, Country

E-mail: first@one.univ, second@asas.edu, third@one.univ, fourth@one.univ

Abstract. PYULTRALIGHT is a Python based program that solves the Schrödinger-Poisson coupled differential equations, which govern the dynamics of ultra-light scalar field dark matter in the non-relativistic regime. PYULTRALIGHT utilises a symmetrised split-step pseudo-spectral Fourier algorithm to solve these differential equations, and demonstrates excellent energy conservation. Being Python-based, the program is particularly easy to set up, and the accompanying Jupyter notebook provides a simple interface in which to augment simulation parameters and create visualisations of output data. Using PYULTRALIGHT, we are able to reproduce many previously studied dynamical characteristics of ultra-light dark matter, even when running the simulations at relatively low spatial resolution from a standard user-PC.

---

[1]Corresponding author.
[2]Also at Some University.

Contents

1 Introduction

The true nature of dark matter (DM) is one of the most important outstanding problems in theoretical physics. WIMP models of dark matter are favoured by many, due in part to their consistency with supersymmetric extensions to the standard model of particle physics. However, with increasingly advanced direct-detection experiments failing to find evidence for WIMP candidates, the viable parameter space continues to shrink [1, 2]. Furthermore, the ΛCDM cosmologies consistent with WIMP models suffer from a number of widely-publicised small-scale problems [3]. While baryonic physics is proposed to offer solutions to a number of these problems, the perceived issues within ΛCDM are increasingly motivating novel research into alternative DM models, one such class of model being ultra-light dark matter (ULDM), also known as fuzzy dark matter (FDM), or BEC dark matter. ULDM models are well motivated by fundamental theories possessing approximate shift symmetries, and offer natural resolutions to the small-scale problems of ΛCDM without the need to invoke baryonic physics. This is essentially because below the de Broglie wavelength of the ULDM particle, the Heisenberg uncertainty principle suppresses gravitational collapse. Since the mass of the ULDM particle is postulated to be of order 10^{-22} eV, this amounts to suppression of gravitational collapse on kilo parsec scales [4]. A recent review of ULDM/FDM is given in [5].

The computational modelling of ULDM systems in the non-relativistic regime can be distilled into numerically solving the dynamical Schrödinger-Poisson (SP) system of coupled differential equations for given initial conditions. Approaches to this problem have been varied, and have given rise to modifications of existing cosmological simulation codes as well as the development of new codes specifically for the modelling of ULDM systems, though

public releases of the latter are lacking. One approach has been to use the Madelung fluid formulation of the SP system [6], in which there arises a quantum pressure term which can be treated numerically in a variety of ways. In [7], the cosmological code GADGET [8] has been modified in order to treat this quantum pressure using an effective particle-particle interaction. This modified code, AXION-GADGET, has been made publicly available [9]. In [10], a non-public extension of GADGET, P-GADGET3, has been modified to treat the quantum pressure term using smoothed-particle hydrodynamics (SPH) routines. The SPH approach has also been used in [11], while a particle-mesh approach was implemented in [12] to treat the quantum pressure term. Other cosmological simulation codes which have been recruited to solve ULDM dynamics (though not necessarily using the Madelung formulation) are NYX [13], GALACTICUS [14], AREPO [15], and GAMER [16, 17]. NYX was modified in [18] to study merging ULDM solitonic cores, GALACTICUS was modified in [19] to study the effects of tidal stripping and dynamical friction on ULDM halos, AREPO was modified in [20] to study the core-mass relationship and turbulence characteristics of ULDM halos, and GAMER was modified in [21] to perform a detailed study of structure formation in ULDM cosmologies.

While these modified cosmological simulation codes have produced promising results in the study of ULDM physics, at the time of writing only one such code, AXION-GADGET, has been made publicly available. The aim of this paper is to introduce PyUltraLIGHT, a stand-alone Python-based pseudo-spectral SP solver, and to demonstrate that this code is able to reproduce a number of key findings from the more complicated cosmological simulation codes without the need for high-performance computing environments. Consequently, we anticipate that as a publicly available resource, PyUltraLIGHT will serve as a valuable cross-check for more complex implementations and may serve as a basis for further development of such codes within the computational cosmology community.

PyUltraLIGHT makes use of a symmetrised-split-step pseudo-spectral Fourier algorithm (also known as leap-frog) to compute the dynamics of the SP system. We note that a similar methodology has been described in [22], though to the authors' knowledge this code has not been released publicly. The symmetrised-split-step method offers 2nd order accurate time integration steps and sub-percent level energy conservation while normalisation of the wavefunction is conserved to machine precision. As a pseudo-spectral code, linear differential operators are computed by direct multiplication in the Fourier domain, while non-linear terms are evaluated in position space. This means that PyUltraLIGHT is free from noise associated with finite-differencing methods of computing spatial derivatives, though there is a necessary computational cost associated with the Fourier and inverse Fourier transforms. These transforms are optimised in PyUltraLIGHT through the use of the pyFFTW pythonic wrapper around the C-based FFWTW subroutine library [23] [24]. While the code is currently implemented to run on a singe machine, the FFTW libraries offer full parallelisation, so there is no barrier to running the code in a cluster environment.

This paper is organised as follows. We first provide a short review of the physics of ULDM, including a derivation of the SP equations from an appropriate scalar-field Lagrangian. We then describe the details of the implementation of PyUltraLIGHT, before moving on to a testing and verification section, in which we reproduce a selection of results from a variety of recent ULDM simulations and discuss energy conservation within the code.

2 The Physics of ULDM: The Schrödinger-Poisson System

The existence of an extremely light scalar field, minimally coupled to gravity, is the central premise on which ULDM models are predicated. Within the ULDM framework, it is proposed that large-scale cosmological structure formation proceeds through the Bose-Einstein condensation of this scalar field. In this section we derive the (non-relativistic) dynamical equations of the single macroscopic condensate wavefunction, namely, the Schrödinger-Poisson equations. We begin by considering the following action functional for a scalar field, ϕ, minimally coupled to gravity and in the absence of self-interactions:

$$S = \int \frac{d^4x}{\hbar} \sqrt{-g} \left\{ \frac{1}{2} g^{\mu\nu} \partial_\mu \phi \partial_\nu \phi - \frac{m^2}{2\hbar^2} \phi^2 \right\}, \tag{2.1}$$

where we have taken $c = 1$ but retain factors of $\hbar$ at this stage. Applying the variational principle to this action yields the Euler-Lagrange equations

$$\frac{1}{\sqrt{-g}} \partial_\mu \left[\sqrt{-g} g^{\mu\nu} \partial_\nu \phi \right] - \frac{m^2}{\hbar^2} \phi = 0. \tag{2.2}$$

We evaluate equation 2.2 using linear perturbation theory, adopting the perturbed FRW metric in the Newtonian gauge:

$$ds^2 = -\left(1 + 2\Phi(\vec{r})\right)dt^2 + a(t)^2 \left(1 - 2\Phi(\vec{r})\right)d\vec{r}^2. \tag{2.3}$$

To linear order in $\Phi(\vec{r})$ we obtain

$$\ddot{\phi} - \frac{\left(1 + 4\Phi(\vec{r})\right)}{a(t)^2} \nabla^2 \phi + 3\left(1 + 2\Phi(\vec{r})\right)H\dot{\phi} + \left(1 + 2\Phi(\vec{r})\right)\frac{m^2}{\hbar^2}\phi = 0, \tag{2.4}$$

where $H = \dot{a}(t)/a(t)$. At late times, $H \ll m$ and it is sufficient to set $H = 0$ and $a(t) = 1$ in equation 2.4. In the non-relativistic limit, we may use the WKB approximation to obtain an ansatz solution to equation 2.4. We take

$$\phi = \frac{\hbar}{\sqrt{2}m} \left(\psi e^{-imt/\hbar} + \psi^* e^{imt/\hbar} \right), \tag{2.5}$$

where ψ is a slowly varying complex function so that terms involving $\ddot{\psi}$ may be neglected. We may also use $\nabla^2 \phi \ll m^2/\hbar^2 \phi$, such that our final result takes the form

$$i\hbar\dot{\psi} = -\frac{\hbar^2}{2m} \nabla^2 \psi + m\Phi\psi. \tag{2.6}$$

Thus, by making suitable approximations we have derived the Schrödinger equation for ψ, which we now interpret as the unique macroscopic wavefunction of a Bose-Einstein condensate. It follows that the particle density of the condensate is given by $|\psi|^2$, such that the mass density is simply $m|\psi|^2$. Using this, we may therefore write down the Poisson equation as

$$\nabla^2 \Phi = 4\pi G m |\psi|^2, \tag{2.7}$$

where G is Newton's gravitational constant. The coupled equations 2.6 and 2.7 together form the nonlinear Schrödinger-Poisson (SP) system which describes the dynamics of ULDM in the non-relativistic regime. It is this system of equations which PyUltraLight is designed to solve.

3 Algorithm and Implementation

Summary of this section TBW

3.1 Field Rescalings

It is helpful to recast the SP system (equations 2.6 and 2.7) in terms of adimensional quantities. In keeping with [25] and [22], we introduce length, time, and mass scales as follows:

$$\mathcal{L} = \left(\frac{8\pi\hbar^2}{3m^2 H_0^2 \Omega_{m_0}} \right)^{\frac{1}{4}} \approx 121 \left(\frac{10^{-23}\,\text{eV}}{m} \right) \text{kpc}, \tag{3.1}$$

$$\mathcal{T} = \left(\frac{8\pi}{3 H_0^2 \Omega_{m_0}} \right)^{\frac{1}{2}} \approx 75.5\,\text{Gyr}, \tag{3.2}$$

$$\mathcal{M} = \frac{1}{G} \left(\frac{8\pi}{3 H_0^2 \Omega_{m_0}} \right)^{-\frac{1}{4}} \left(\frac{\hbar}{m} \right)^{\frac{3}{2}} \approx 7 \times 10^7 \left(\frac{10^{-23}\,\text{eV}}{m} \right) \text{M}_\odot, \tag{3.3}$$

where m is the mass of the ultra-light scalar field, H_0 is the present-day Hubble parameter, G is Newton's gravitational constant and Ω_{m_0} is the present-day matter fraction of the energy density of the universe. We now recast equations 2.6 and 2.7 in terms of the following dimensionless quantities:

$$t' = \frac{t}{\mathcal{T}}, \quad \vec{x}' = \frac{\vec{x}}{\mathcal{L}}, \quad \Phi' = \frac{m\mathcal{T}}{\hbar}\Phi, \quad \psi' = \mathcal{T}\sqrt{mG}\psi. \tag{3.4}$$

Dropping the primes for notational convenience, we see that the coupled differential equations of the SP system are reduced to

$$i\dot{\psi}(\vec{x}, t) = -\frac{1}{2}\nabla^2 \psi(\vec{x}, t) + \Phi(\vec{x}, t)\psi(\vec{x}, t), \tag{3.5}$$

$$\nabla^2 \Phi(\vec{x}, t) = 4\pi|\psi(\vec{x}, t)|^2, \tag{3.6}$$

where it is understood that all quantities involved are dimensionless. We can recover any dimensionful quantities of interest through simple transformations using equations 3.1 to 3.3. For example, the integrated mass of the system, M_{tot}, is given by

$$M_{tot} = \mathcal{M} \int d^3 x |\psi|^2. \tag{3.7}$$

Likewise, the mass density at any point is given by

$$\rho = \frac{\mathcal{M}}{\mathcal{L}^3}|\psi|^2. \tag{3.8}$$

All computations performed by PyUltraLight are carried out using the dimensionless quantities defined in 3.4, while provisions are made within the code for initial conditions and outputs to be converted to and from the desired unit system specified by the user. Henceforth, we will often refer to the quantity $|\psi|^2$ simply as the density, where it is understood that this is in fact a dimensionless quantity related to the physical mass density through a constant of proportionality given by equation 3.8.

3.2 Dynamical Evolution

Dynamical evolution within PyUltraLight progresses via a symmetrised split-step Fourier process on an $N \times N \times N$ simulation grid with spatial periodic boundary conditions. To understand this method, we first consider the exact expression for the unitary time evolution of the wavefunction according to equation 3.5, namely

$$\psi(\vec{x}, t + h) = \mathcal{T} \exp \left[-i \int_t^{t+h} dt' \left\{ -\frac{1}{2} \nabla^2 + \Phi(\vec{x}, t') \right\} \right] \psi(\vec{x}, t), \tag{3.9}$$

where $\mathcal{T}$ is the time-ordering symbol. For sufficiently small timestep h, we can use the trapezoidal rule make the approximation

$$\int_t^{t+h} dt' \Phi(\vec{x}, t') \approx \frac{h}{2} \Big(\Phi(\vec{x}, t+h) + \Phi(\vec{x}, t) \Big). \tag{3.10}$$

We can therefore write the approximate form of equation 3.9 as

$$\psi(\vec{x}, t + h) \approx \exp \left[i \frac{h}{2} \Big(\nabla^2 - \Phi(\vec{x}, t+h) - \Phi(\vec{x}, t) \Big) \right] \psi(\vec{x}, t). \tag{3.11}$$

It is acknowledged that the exponential in equation 3.11 omits the time-ordering symbol, and is only equivalent to its time-ordered counterpart to order h^2.

Given that the linear differential operator in equation 3.11 acts naturally in Fourier space, while the nonlinear potential term is simplest to evaluate in position space, we are motivated to split the exponential so as to separate these terms. This will then allow us to employ a pseudo-spectral algorithm to compute the time evolution. Such a splitting of the exponential is valid when the timestep h is small, as the dispersive and nonlinear terms can be assumed to act independently. In this implementation, we split the exponential as follows:

$$\psi(\vec{x}, t+h) \approx \exp \left[-\frac{ih}{2} \Phi(\vec{x}, t+h) \right] \exp \left[\frac{ih}{2} \nabla^2 \right] \exp \left[-\frac{ih}{2} \Phi(\vec{x}, t) \right] \psi(\vec{x}, t). \tag{3.12}$$

This splitting can be understood thusly: first, a half timestep is taken in which only the nonlinear potential operator acts, followed by a full timestep in the linear term. The potential field is then updated, and a final half timestep in the nonlinear term is performed. We may use the Baker-Campbell-Hausdorff formula to express the product of exponentials in equation 3.12 as a single exponential. Keeping only terms to order h^2 we find:

$$\exp \left[i \frac{h}{2} \Big(\nabla^2 - \Phi(\vec{x}, t+h) - \Phi(\vec{x}, t) \Big) + \frac{h^2}{8} \Big[\nabla^2, \Phi(\vec{x}, t) \Big] - \frac{h^2}{8} \Big[\nabla^2, \Phi(\vec{x}, t+h) \Big] \right]. \tag{3.13}$$

Making use of the fact that $\Phi(\vec{x}, t + h) \approx \Phi(\vec{x}, t) + h \dot{\Phi}(\vec{x}, t)$, we see that the commutators in equation 3.13 cancel at $\mathcal{O}(h^2)$, such that this expression is equivalent to 3.11, with the dominant error term appearing at $\mathcal{O}(h^3)$.

Evaluation of equation 3.12 within PyUltraLight proceeds as follows: Initially, the nonlinear term acts in position space for one half-timestep. The result is then Fourier transformed, and a full timestep is taken in which the differential operator is applied in the Fourier domain. The potential field is then updated in accordance with equation 3.6. An inverse Fourier transform is then performed, and a final half timestep is taken in which the updated nonlinear term acts in position space to give the new ψ field configuration. This procedure

is known as the symmetrised split-step Fourier method, and is used widely in fields such as nonlinear fiber optics [26]. It can be represented schematically as follows:

$$\psi(\vec{x}, t+h) = \exp\left[-\frac{ih}{2}\Phi(\vec{x}, t+h)\right] \mathcal{F}^{-1} \exp\left[\frac{-ih}{2}k^2\right] \mathcal{F} \exp\left[-\frac{ih}{2}\Phi(\vec{x}, t)\right] \psi(\vec{x}, t), \quad (3.14)$$

where the order of operations is read from right to left, $\mathcal{F}$ and $\mathcal{F}^{-1}$ represent the discrete Fourier transform and its inverse, and k is the wavenumber in the Fourier domain. The potential field is updated following the inverse Fourier transform in equation 3.14, and this update procedure may be represented schematically as

$$\Phi(\vec{x}, t+h) = \mathcal{F}^{-1}\left(-\frac{1}{k^2}\right) \mathcal{F} \ 4\pi|\psi(\vec{x}, t_i)|^2, \quad (3.15)$$

where $\psi(\vec{x}, t_i)$ represents the field configuration at this intermediary point in the timestep. The final operation in equation 3.14 changes only the phase of the ψ field, and thus we could replace $\psi(\vec{x}, t_i)$ with $\psi(\vec{x}, t+h)$ in equation 3.15 with no change in meaning.

In the PYULTRALIGHT implementation, we make use of an additional simplification of the symmetrised split-step Fourier method. Namely, consecutive half-steps in the nonlinear term may be combined into a single full step in each case such that when iterated, only the first and last operations involve half steps. This can be represented schematically as

$$\psi(t+nh) = \exp\left[+\frac{ih}{2}\Phi\right] \left(\prod^n \exp\left[-\frac{ih}{2}\Phi\right] \exp\left[\frac{ih}{2}\nabla^2\right]\right) \exp\left[-\frac{ih}{2}\Phi\right] \psi(t), \quad (3.16)$$

where the Φ field is updated at each step in accordance with equation 3.15 and attention is drawn to the sign difference between the first and last operators.

The discrete Fourier transform (DFT) and its inverse are handled within PYULTRALIGHT using pyFFTW. pyFFTW is a pythonic wrapper around the C-based FFTW subroutine library, and allows for efficient evaluation of both real and complex DFTs [23] [24] [27]. Multithreading is supported by pyFFTW, and is implemented within PYULTRALIGHT by using the multiprocessing package to determine the number of online CPUs and feed this directly to the threads argument of the pyfftw.FFTW class.

In addition to the use of pyFFTW, performance within PYULTRALIGHT is further improved through the use of the NumExpr package, which provides routines for the efficient manipulation of array objects within the simulation using a vector-based virtual machine [28].

3.3 Initial Conditions: Soliton Profiles

The initialisation procedure within PYULTRALIGHT can be easily augmented to accommodate the requirements of the user. In the current implementation, the initial field configuration is built by loading a NumPy array file encoding a solitonic solution to the Schrödinger-Poisson system. Using this method, solitons may be placed at the desired locations within the simulation grid, with the corresponding mass, velocity, and phase parameters each specified by the user within the accompanying Jupyter notebook.

The solitonic solution to the SP system is a natural choice of initial field configuration; these stable, localised objects are ubiquitous in ULDM models [29].

In this section, we discuss the method used to generate the soliton profile. Following [22], we begin by assuming a spherically symmetric solution to the SP system:

$$\psi(\vec{x}, t) \to e^{i\beta t} f(r), \quad \Phi(\vec{x}, t) \to \varphi(r), \tag{3.17}$$

where $r = |\vec{x}|$. Introducing $\tilde{\varphi}(r) := \varphi(r) + \beta$, equations 3.5 and 3.6 are reduced to

$$0 = -\frac{1}{2} f''(r) - \frac{1}{r} f'(r) + \tilde{\varphi}(r) f(r) \tag{3.18}$$

$$0 = \tilde{\varphi}''(r) + \frac{2}{r} \tilde{\varphi}'(r) - 4\pi f(r)^2 \tag{3.19}$$

where primes denote derivatives with respect to r. To generate profiles for $\tilde{\varphi}(r)$ and $f(r)$ which satisfy these equations, we use a fourth-order Runge-Kutta algorithm, included as the auxiliary soliton_solution.py program. We make the arbitrary choice $f(r = 0)=1$, while the smoothness criterion demands that the first derivatives of $f(r)$ and $\tilde{\varphi}(r)$ vanish at the origin. We then search for solutions of $f(r)$ and $\varphi(r)$ which satisfy the boundary conditions $\lim(r \to \infty)\varphi(r) = 0$ and $\lim(r \to \infty)f(r) = 0$. To do this we use a shooting method, varying $\tilde{\varphi}(r = 0)$ until we obtain a solution of $f(r)$ which approaches zero at the maximal specified radius, r_m. The value of β is then calculated by assuming that $\varphi(r)$ goes as $-c/r$ at large radii, where c is a constant. Under this assumption, we can write

$$\tilde{\varphi}(r_m) = -\frac{c}{r_m} + \beta, \quad c = r_m^2 \tilde{\varphi}'(r_m). \tag{3.20}$$

We thus obtain the full solution $\psi(\vec{x}, t) = e^{i\beta t} f(r)$. Having initially chosen $f(r = 0) = 1$, we may then generalise to $f(r = 0) = \alpha$, where α is an arbitrary positive real number. It is easy to verify that if $e^{i\beta t} f(r)$ is a solution to the spherically symmetric SP system, then $g(r)$ is also a solution, where

$$g(r) = e^{i\alpha\beta t} \alpha f(\sqrt{\alpha}\, r). \tag{3.21}$$

Hence we obtain a family of spherically symmetric soliton solutions to the dimensionless SP system, for which the dimensionless soliton mass is proportional to $\sqrt{\alpha}$ and the full width at half maximum is proportional to $1/\sqrt{\alpha}$. Because the size of the soliton scales inversely with the mass, it is important to consider the maximum soliton mass which can be accurately modeled in a simulation with finite spatial resolution.

While the Schrödinger equation is not trivially form invariant under Galilean boosts, we can enforce Galilean covariance by augmenting the solutions through the addition of a velocity-dependent phase factor. Namely,

$$\psi(\vec{x}, t) = \alpha f\big(\sqrt{\alpha}|\vec{x} - \vec{v}t|\big) e^{i\big(\alpha\beta t + \vec{v}\cdot\vec{x} - \frac{1}{2}|\vec{v}|^2 t\big)}. \tag{3.22}$$

To construct the initial field configuration within PYULTRALIGHT, the NumPy array encoding the radial profile $f(r)$ for the $f(r) = 0) = 1$ case is loaded, where the spatial resolution of this profile must agree with the expected spatial resolution within the main code. Equation 3.22 is then used to augment the configuration such that the soliton is furnished with the position, mass, and velocity specified by the user in the accompanying Jupyter notebook. The user may also add an additional constant phase factor if desired.

While the current implementation of PYULTRALIGHT is designed such that the initial field configuration is built from solitons, it may be simply modified to take in alternative field configurations in future.

3.4 Timestep Constraints

When using finite-differencing methods to solve certain partial differential equations, the causality-preserving Courant-Friedrichs-Lewy (CFL) condition on the size of the simulation timestep is a necessary condition for convergence [30]. As such, this condition is often appears in numerical analysis of ULDM physics governed by the SP system, an example being [18]. However, because PyUltraLight does not use the method of finite differences to compute spatial derivatives, but rather computes them exactly via explicit multiplication in the Fourier domain, the CFL condition in its usual formulation is not applicable to this code.

In place of the CFL condition, the constraint on the timestep within PyUltraLight is set based on the fluid interpretation of the SP system. In dimensionless code units we can write

$$\psi \equiv \sqrt{\rho}e^{i\theta}, \quad \vec{v} \equiv \boldsymbol{\nabla}\theta. \tag{3.23}$$

From here we can define a 'maximum velocity' for the simulation grid. This is the velocity at which the fluid appears to move backward, which occurs when the phase change between successive grid points exceeds π. The timestep is then constrained such that a fluid travelling at this maximum grid velocity will traverse a distance of one grid space per timestep.

This proves to be a highly restrictive constraint on the timestep, and can be relaxed significantly at the user's discretion for simulations in which the expected velocities within the system will not approach this maximum grid velocity. Convergence testing on a case-by-case basis is encouraged to determine the timestep constraint necessary for a given degree of accuracy.

4 Testing and Verification

4.1 Interference Patterns During Soliton Collisions

Hierarchical structure formation within the ULDM framework proceeds through both gravitational overdensity collapse as well as through sequential binary mergers of solitonic cores to form individual virialised objects [25]. It is therefore important that ULDM simulations are able to accurately reproduce the expected characteristics of binary soliton interactions.

Soliton collisions can be classified into two categories depending on whether the total energy of the isolated binary system is positive or negative. In the case of positive energy, the solitons pass through each other, emerging largely undisturbed from their initial configurations following the interaction. During the collision, the wavefunctions describing the solitons are superposed, yielding distinctive interference patterns. In this section we demonstrate the ability of PyUltraLight to reproduce the interference patterns predicted by theoretical models during binary collisions.

Following [18], we consider the head-on collision of two solitions with mass ratio $\mu = 2$ and high relative velocity. Using equation 3.22, we can write the total wavefunction of the binary system in terms of dimensionless quantities along the collision axis as

$$\psi(x,t) = \alpha_1 f\left(\sqrt{\alpha_1}|x - x_1 - v_1 t|\right)e^{i\left(\alpha_1\beta t + v_1(x-x_1) - \frac{1}{2}v_1^2 t + \delta\right)}$$
$$+ \alpha_2 f\left(\sqrt{\alpha_2}|x - x_2 - v_2 t|\right)e^{i\left(\alpha_2\beta t + v_2(x-x_2) - \frac{1}{2}v_2^2 t\right)} \tag{4.1}$$

where x_1 and x_2 are the initial positions of the centres of the solitions, v_1 and v_2 are the velocities of the solitons, δ is a constant relative phase term, and $\alpha_1 = 1/4 \ \alpha_2$ parameterise the density profiles as discussed in section 3.3. Because of the symmetry of the configuration,

we take $v_1 = -v_2$ and $x_1 = -x_2$. We are interested in the density profile of the system at the time of maximal interference, where we expect the superposed wavefunction to possess pronounced features. Therefore, we choose the time at which the two components of the wavefunction are centred at the same location, such that $x_1 + v_1 t = -x_2 - v_2 t = 0$. This corresponds to a time $t_{max} = |x_1/v_1| = |x_2/v_2|$. The dimensionless density is then given by

$$|\psi(x, t_{max})|^2 = \alpha_1^2 \Bigg[f(\sqrt{\alpha_1}x)^2 + 16f(2\sqrt{\alpha_1}x)^2 +$$
$$8f(\sqrt{\alpha_1}x)f(2\sqrt{\alpha_1}x)\cos\left(-3\alpha_1\beta\left|\frac{x_1}{v_1}\right| + 2v_1 x + \delta\right)\Bigg] \qquad (4.2)$$

In Figure 1, we plot the dimensionless density profile at the time of maximal interference for two solitons with mass ratio $\mu = 2$, and phase difference $\delta = 0$. We see that the numerical result obtained using PyUltraLight closely matches the theoretical prediction of equation 4.2. Small disparities between the numerical and theoretical profiles may be attributed to the effect of gravitational contraction not included in the theoretical prediction of equation 4.2, and to a small offset in the true time of maximal interference. Neglecting these small deviations, we can verify that the code is well suited to faithfully reproduce the wave interference effects of the ULDM model.
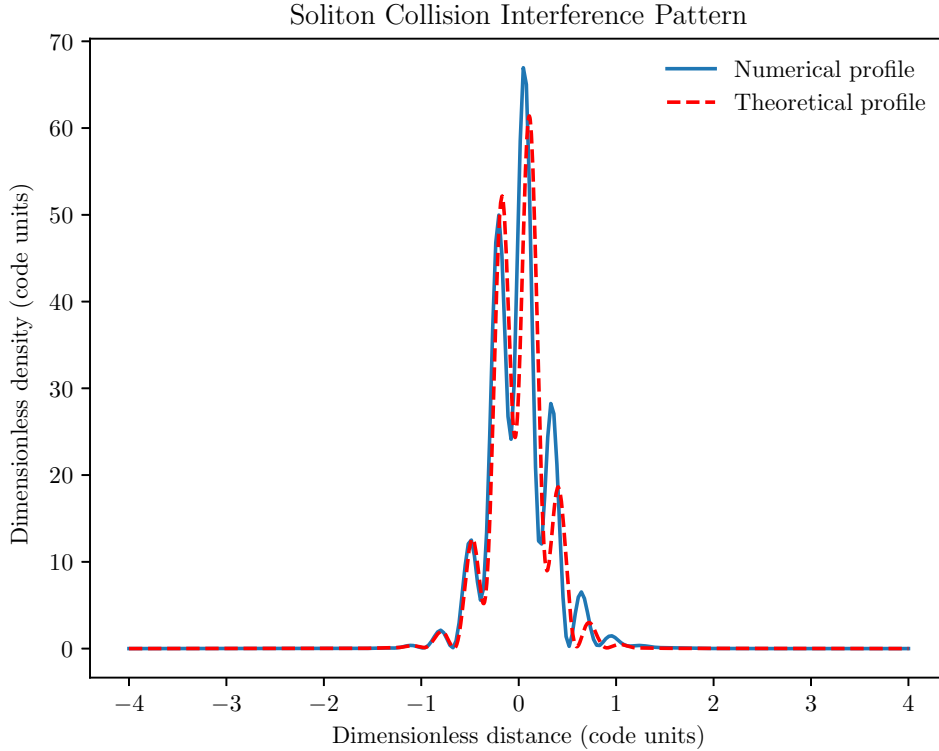
Figure 1: Comparison of theoretical and numerical density profiles at time of maximal interference for head-on collision of two solitons with mass ratio $\mu = 2$ and no relative phase difference. The solitons were chosen with dimensionless masses 5 and 10, with an initial separation of 4 code units and relative velocity of 20 code units. The simulation was run at 256^3 for a box of side-length 8 code units.

4.2 Effective Forces Due to Destructive Interference

As demonstrated in [22], the wavelike properties of ULDM can give rise to effective forces which can dramatically affect the dynamics of core collisions. These effective forces arise as a result of interference phenomena, and do not require the ULDM model to incorporate explicit local interactions. Figure 2 shows the results of two simulations using PyUltraLight in which two solitons of equal mass undergo a head-on collision. Contours of the density profile along the plane of symmetry are displayed. In the first simulation, there is no phase offset between the initial solitons, while in the second simulation the phases differ by π. We can see that the result of this phase shift is to create an effective repulsive force between the two solitons as they approach one another. Further discussion of this phenomenon and its possible observational consequences can be found in [22].
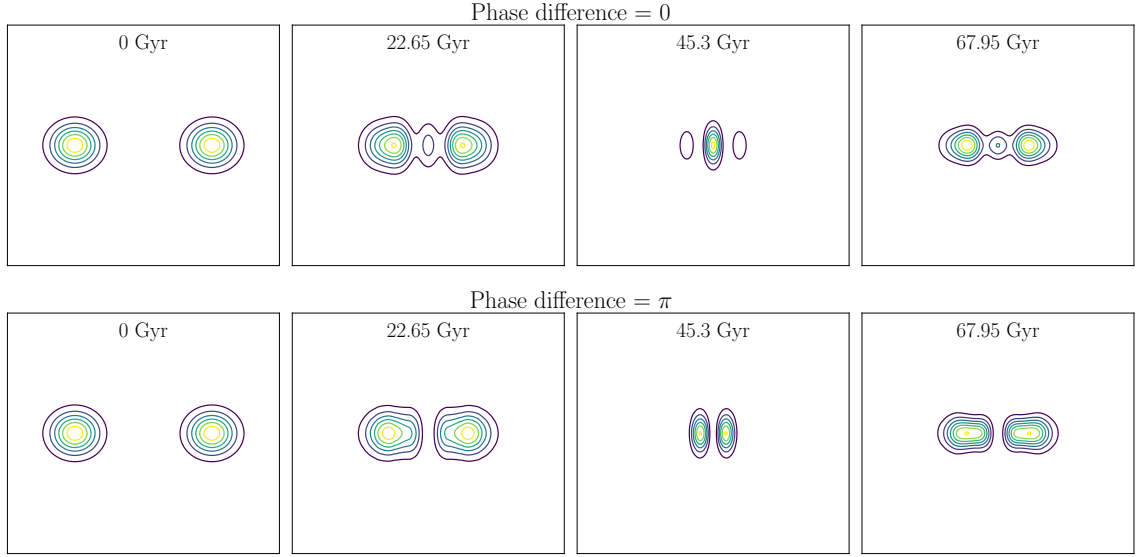

Figure 2: Head-on collisions between two solitons of mass 5, relative velocity 6, and initial separation 4 in code units. Time progresses from left to right across each row.

4.3 Tidal Disruption of Solitons Orbiting a Central Potential

PyUltraLight is designed such that a central potential may be included by way of a point-mass located at the centre of the simulation grid. While this setup is somewhat unphysical, it provides a starting point for a study of the stability of satellite dark matter halos orbiting a much larger object. Studies of this type can be used to make predictions regarding the lifespans of satellite galaxies around the Milky Way, offering a potential resolution of the so-called missing satellites problem [31].

An extensive study of the tidal disruption of ULDM solitonic cores orbiting a central potential has recently been undertaken in [32]. Here we reproduce just one of their results as a validity check for PyUltraLight. In particular, it is found in [32] that an initially spherical solitonic core without self-rotation will gradually spin up to form a tidally-locked irrotational Riemann -S ellipsoid when orbiting a host mass. Figure 3 shows the internal velocity field of a solitonic satellite after one complete revolution around a host mass, obtained using PyUltraLight on a 256^3 simulation grid. In this case the ratio of host to satellite mass was

≈ 55. It can be seen that the soliton has become elongated along the radial line connecting to the host, indicating that it is tidally locked. Meanwhile the internal velocity field can be seen to be that of an irrotational Riemann-S ellipsoid [33].
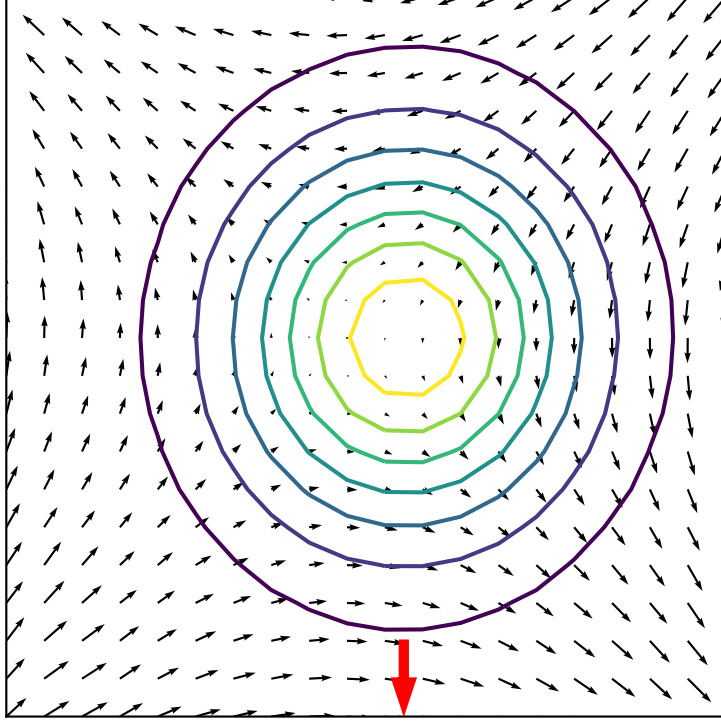
Figure 3: Solitonic core after one revolution of a central potential, demonstrating the internal velocity field of an irrotational Riemann-S ellipsoid (where the collective motion of the soliton has been subtracted from the velocity field). Deformation of the spherical density profile into that of an ellipsoid can be seen, with extension along the radial direction (red arrow).

4.4 Energy Conservation

In addition to the evolution of the condensate wavefunction ψ and the potential Φ, PyUltraLight also evaluates the energy of the SP system. An important achievement of the code is sub-percent level energy conservation, even at relatively low spatial resolution, for all dynamical scenarios tested. In this section we derive the expression for the total energy of the SP system and discuss its decomposition into individual constituents calculated separately within the code. We then present the numerical energy evolution of a variety of configurations.

We begin by defining a suitable action which yields the full SP system through its corresponding Euler-Lagrange equations. We find that variation of

$$S = \int dt \int_{\mathbb{R}^3} d^3x \ -\left\{\frac{1}{2}|\nabla\Phi|^2 + \Phi|\psi|^2 + \frac{1}{2}|\nabla\psi|^2 + \frac{i}{2}(\psi\dot{\psi}^* - \dot{\psi}\psi^*)\right\} \quad (4.3)$$

with respect to Φ, ψ^* and ψ yields equations 3.6, 3.5, and the conjugate of equation 3.5, respectively. The integrand of equation 4.3 is the Lagrangian density, $\mathcal{L}$, from which we can

derive the conserved energy in the usual way:

$$E_{tot} = \int_{\mathbb{R}^3} d^3x \left\{ \frac{\partial \mathcal{L}}{\partial \dot{\psi}} \dot{\psi} + \frac{\partial \mathcal{L}}{\partial \dot{\psi^*}} \dot{\psi^*} + \frac{\partial \mathcal{L}}{\partial \dot{\Phi}} \dot{\Phi} - \mathcal{L} \right\}. \tag{4.4}$$

Evaluating this expression, we obtain:

$$E_{tot} = \int_{\mathbb{R}^3} d^3x \left\{ \frac{1}{2}|\nabla\Phi|^2 + \Phi|\psi|^2 + \frac{1}{2}|\nabla\psi|^2 \right\} \tag{4.5}$$

$$= \int_{\mathbb{R}^3} d^3x \left\{ \frac{1}{2}\nabla(\Phi\nabla\Phi) - \frac{1}{2}\Phi\nabla^2\Phi + \Phi|\psi|^2 + \frac{1}{2}\nabla(\psi^*\nabla\psi) - \frac{1}{2}\psi^*\nabla^2\psi \right\} \tag{4.6}$$

$$= \int_{\mathbb{R}^3} d^3x \left\{ \frac{1}{2}\Phi|\psi|^2 - \frac{1}{2}\psi^*\nabla^2\psi \right\}. \tag{4.7}$$

where in the last step we have used Stokes' Theorem as well as the Poisson equation (3.6) to perform simplifications. Because we are working with the dimensionless quantities defined in equation 3.4, it is easy to see that this quantity is related to the physical energy through multiplication by a constant factor of $\mathcal{L}^5\mathcal{T}^{-4}G^{-1}$. It should be noted that equation 4.7 is not equivalent to the expectation value of the Schrödinger Hamiltonian, which is itself not a conserved quantity of the SP system and is given by

$$\langle \hat{H} \rangle = \int_{\mathbb{R}^3} d^3x \left\{ \Phi|\psi|^2 - \frac{1}{2}\psi^*\nabla^2\psi \right\}. \tag{4.8}$$

The two terms in the integral 4.7 are calculated separately within the code. It is easy to see that the first term represents the gravitational potential energy of the SP system. As discussed in [5], the second term may be decomposed further into two contributions which may be considered separately as kinetic and 'quantum' energies, but for the purposes of this implementation it is sufficient to consider only their combined contribution. Because PyUltraLight provides the option to include the central potential of a point mass located at the centre of the simulation grid, we have additional energy contributions when such a potential is included. Because this central potential does not arise from a ULDM object, we calculate the gravitational potential energy due to self-interaction separately from the gravitational potential energy due to the central potential.

Figures 4 and 5 demonstrate energy conservation for two possible simulation scenarios. In the first case, we examine the evolution of the energy for a single soliton undergoing significant tidal disruption whilst orbiting within a central potential. We see that as the soliton is disrupted, the kinetic energy increases, while the gravitational energy due to the central potential decreases, as expected. Meanwhile, the gravitational potential energy due to self-interaction gradually increases toward zero as the disruption continues. The sum of the individual energy components is conserved to 10^{-2} % in this case, where the simulation was run at 256^3. Higher grid resolutions result in even better energy conservation. Figure 5 demonstrates the evolution of the energy of a binary system of solitons in elliptical orbits around their common centre of mass. In this case the simulation ran for approximately three orbital periods at 256^3. At points of closest approach, it can be clearly seen that the kinetic energy increases as the solitons speed up, while the potential energy due to self-interaction decreases commensurately. In this scenario no central potential has been included, so the gravitational energy due to the central potential stays uniformly zero throughout the simulation. The small scale oscillations that can be observed are artifacts of the relatively

low spatial resolution, and smooth out as grid resolution is increased. In this case the total energy is conserved to 10^{-3} %.
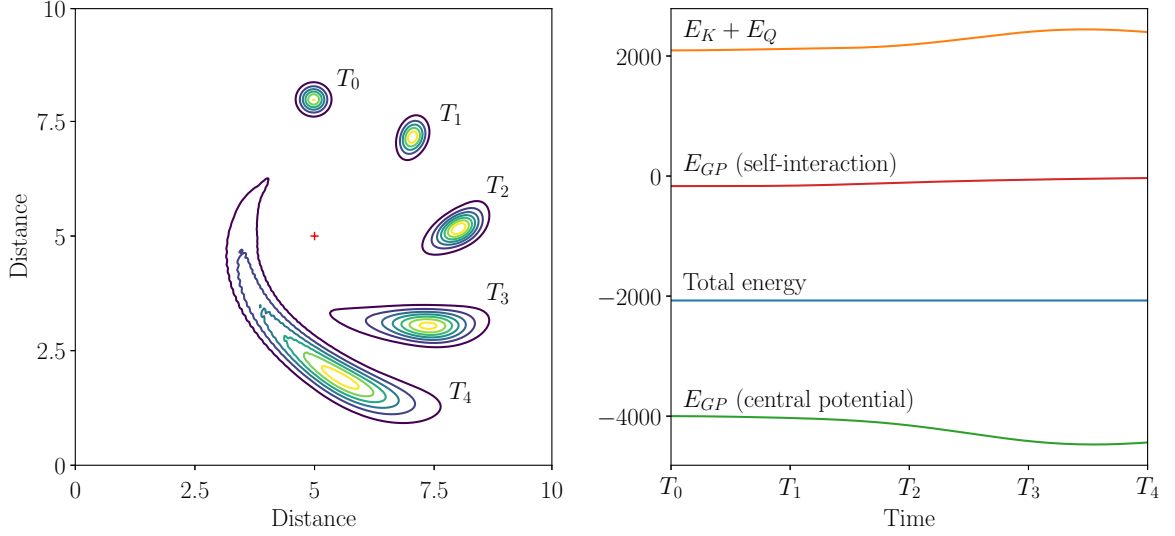
Figure 4: Left: Evolution of the density profile of a solitonic core at equally spaced times as it undergoes tidal disruption in a potential centred at the red cross. Right: Evolution of the energy of the system, where times are indicated in correspondence with the snapshots of the density profile. All quantities are in dimensionless code units.
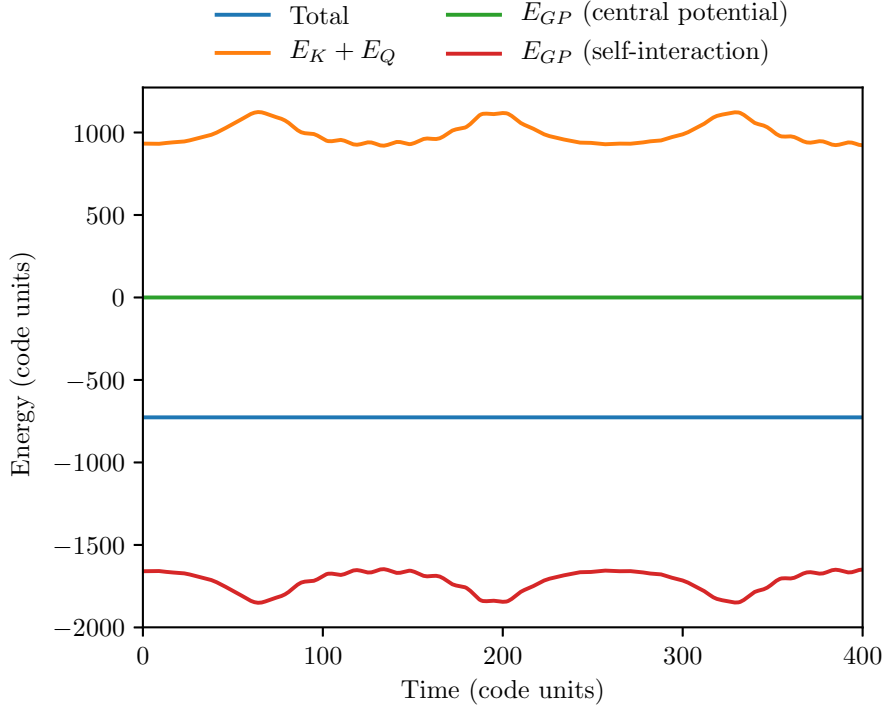
Figure 5: Evolution of the energy for a binary soliton system with each soliton in an elliptical orbit around the common centre of mass.

5 Discussion and Outlook

We have demonstrated that PyUltraLight is an accurate tool for the study of the dynamics of ultra-light dark matter governed by the Schrödinger-Poisson system of equations. The code makes use of a pseudo-spectral symmetrised split-step Fourier methodology, in which all spatial derivatives are treated via explicit multiplication in the Fourier domain, thereby avoiding difficulties associated with finite-differencing methods.

Energy conservation within PyUltraLight is excellent, at sub-percent level for simulations run at 128^3, and even better performance as resolution is increased. The code is able to reproduce complex phenomena resulting from the wave-like properties of ultra-light dark matter as predicted by theoretical models, such as the interference patterns arising during high-velocity collisions of solitonic cores, and the effective forces observed in cases where the colliding cores are out of phase. These phenomena can be clearly observed at relatively low spatial resolution, obviating the need for high-performance computing infrastructure to study the fundamental behaviour of ULDM systems. This makes PyUltraLight a useful tool for predicting the overall dynamics of a ULDM system in a computationally efficient manner prior to a rigorous examination using more advanced codes such as Nyx.

PyUltraLight is Python-based, and as such is particularly simple to understand and use. The accompanying Jupyter notebook allows for the efficient adjustment of simulation parameters, and offers a useful browser interface for quick visualisation of simulation results. Despite being Python-based, the code makes use of low-level language resources, namely the FFTW libraries through the use of the Pythonic pyFFTW wrapper.

While the current implementation of PyUltraLight is already a useful tool for simulating dynamical ULDM systems, there remains scope for substantial improvement. In particular, it is anticipated that future releases may incorporate adaptive mesh refinement capabilities and variable timestep functionality. Additionally, while the current implementation utilises a spatial grid of fixed size and does not provide a means by which to incorporate baryonic density contributions, future releases are likely to allow for cosmic expansion and a means by which to incorporate simplified models of baryonic matter. Further adaptations of the code may also be implemented to allow for greater flexibility in the choice of ULDM model, incorporating possibilities such as explicit self-interaction.

A Download and Licensing

To be written.

Acknowledgments

To be written.

References

[1] **PandaX-II Collaboration** Collaboration, A. Tan, M. Xiao, X. Cui, X. Chen, Y. Chen, D. Fang, C. Fu, K. Giboni, F. Giuliani, H. Gong, X. Guo, K. Han, S. Hu, X. Huang, X. Ji, Y. Ju, S. Lei, S. Li, X. Li, X. Li, H. Liang, Q. Lin, H. Liu, J. Liu, W. Lorenzon, Y. Ma, Y. Mao, K. Ni, X. Ren, M. Schubnell, M. Shen, F. Shi, H. Wang, J. Wang, M. Wang, Q. Wang, S. Wang, X. Wang, Z. Wang, S. Wu, X. Xiao, P. Xie, B. Yan, Y. Yang, J. Yue, X. Zeng, H. Zhang, H. Zhang, H. Zhang, T. Zhang, L. Zhao, J. Zhou, N. Zhou, and X. Zhou, *Dark*

matter results from first 98.7 days of data from the pandax-ii experiment, *Phys. Rev. Lett.* **117** (Sep, 2016) 121303.

[2] **LUX Collaboration** Collaboration, D. S. Akerib, S. Alsum, H. M. Araújo, X. Bai, A. J. Bailey, J. Balajthy, P. Beltrame, E. P. Bernard, A. Bernstein, T. P. Biesiadzinski, E. M. Boulton, R. Bramante, P. Brás, D. Byram, S. B. Cahn, M. C. Carmona-Benitez, C. Chan, A. A. Chiller, C. Chiller, A. Currie, J. E. Cutter, T. J. R. Davison, A. Dobi, J. E. Y. Dobson, E. Druszkiewicz, B. N. Edwards, C. H. Faham, S. Fiorucci, R. J. Gaitskell, V. M. Gehman, C. Ghag, K. R. Gibson, M. G. D. Gilchriese, C. R. Hall, M. Hanhardt, S. J. Haselschwardt, S. A. Hertel, D. P. Hogan, M. Horn, D. Q. Huang, C. M. Ignarra, M. Ihm, R. G. Jacobsen, W. Ji, K. Kamdin, K. Kazkaz, D. Khaitan, R. Knoche, N. A. Larsen, C. Lee, B. G. Lenardo, K. T. Lesko, A. Lindote, M. I. Lopes, A. Manalaysay, R. L. Mannino, M. F. Marzioni, D. N. McKinsey, D.-M. Mei, J. Mock, M. Moongweluwan, J. A. Morad, A. S. J. Murphy, C. Nehrkorn, H. N. Nelson, F. Neves, K. O'Sullivan, K. C. Oliver-Mallory, K. J. Palladino, E. K. Pease, P. Phelps, L. Reichhart, C. Rhyne, S. Shaw, T. A. Shutt, C. Silva, M. Solmaz, V. N. Solovov, P. Sorensen, S. Stephenson, T. J. Sumner, M. Szydagis, D. J. Taylor, W. C. Taylor, B. P. Tennyson, P. A. Terman, D. R. Tiedt, W. H. To, M. Tripathi, L. Tvrznikova, S. Uvarov, J. R. Verbus, R. C. Webb, J. T. White, T. J. Whitis, M. S. Witherell, F. L. H. Wolfs, J. Xu, K. Yazdani, S. K. Young, and C. Zhang, *Results from a search for dark matter in the complete lux exposure*, *Phys. Rev. Lett.* **118** (Jan, 2017) 021303.

[3] P. Bull, Y. Akrami, J. Adamek, T. Baker, E. Bellini, J. Beltrán Jiménez, E. Bentivegna, S. Camera, S. Clesse, J. H. Davis, E. Di Dio, J. Enander, A. Heavens, L. Heisenberg, B. Hu, C. Llinares, R. Maartens, E. Mörtsell, S. Nadathur, J. Noller, R. Pasechnik, M. S. Pawlowski, T. S. Pereira, M. Quartin, A. Ricciardone, S. Riemer-Sørensen, M. Rinaldi, J. Sakstein, I. D. Saltas, V. Salzano, I. Sawicki, A. R. Solomon, D. Spolyar, G. D. Starkman, D. Steer, I. Tereno, L. Verde, F. Villaescusa-Navarro, M. von Strauss, and H. A. Winther, *Beyond Λ CDM: Problems, solutions, and the road ahead*, *Physics of the Dark Universe* **12** (June, 2016) 56–99, [`arXiv:1512.0535`].

[4] W. Hu, R. Barkana, and A. Gruzinov, *Fuzzy cold dark matter: The wave properties of ultralight particles*, *Phys. Rev. Lett.* **85** (Aug, 2000) 1158–1161.

[5] L. Hui, J. P. Ostriker, S. Tremaine, and E. Witten, *Ultralight scalars as cosmological dark matter*, *Phys. Rev.* **D95** (2017), no. 4 043541, [`arXiv:1610.0829`].

[6] E. Madelung, *Eine anschauliche deutung der gleichung von schrödinger*, *Naturwissenschaften* **14** (Nov, 1926) 1004–1004.

[7] J. Zhang, Y.-L. Sming Tsai, J.-L. Kuo, K. Cheung, and M.-C. Chu, *Ultralight Axion Dark Matter and Its Impact on Dark Halo Structure in N-body Simulations*, **853** (Jan., 2018) 51, [`arXiv:1611.0089`].

[8] V. Springel, *The cosmological simulation code gadget-2*, *Monthly Notices of the Royal Astronomical Society* **364** (2005), no. 4 1105–1134, [`/oup/backfile/content_public/journal/mnras/364/4/10.1111_j.1365-2966.2005.09655.x/1/364-4-1105.pdf`].

[9] "https://github.com/liambx/axion-gadget."

[10] M. Nori and M. Baldi, *AX-GADGET: a new code for cosmological simulations of Fuzzy Dark Matter and Axion models*, *ArXiv e-prints* (Jan., 2018) [`arXiv:1801.0814`].

[11] P. Mocz and S. Succi, *Numerical solution of the nonlinear schrödinger equation using smoothed-particle hydrodynamics*, *Phys. Rev. E* **91** (May, 2015) 053304.

[12] J. Veltmaat and J. C. Niemeyer, *Cosmological particle-in-cell simulations with ultralight axion dark matter*, **94** (Dec., 2016) 123523, [`arXiv:1608.0080`].

[13] A. S. Almgren, J. B. Bell, M. J. Lijewski, Z. LukiÄĞ, and E. V. Andel, *Nyx: A massively parallel amr code for computational cosmology*, The Astrophysical Journal **765** (2013), no. 1 39.

[14] A. J. Benson, *Galacticus: A semi-analytic model of galaxy formation*, New Astronomy **17** (2012), no. 2 175 – 197.

[15] V. Springel, *E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh*, **401** (Jan., 2010) 791–851, [`arXiv:0901.4107`].

[16] H.-Y. Schive, Y.-C. Tsai, and T. Chiueh, *Gamer: A graphic processing unit accelerated adaptive-mesh-refinement code for astrophysics*, The Astrophysical Journal Supplement Series **186** (2010), no. 2 457.

[17] "https://github.com/gamer-project/gamer/wiki."

[18] B. Schwabe, J. C. Niemeyer, and J. F. Engels, *Simulations of solitonic core mergers in ultralight axion dark matter cosmologies*, Phys. Rev. D **94** (Aug, 2016) 043513.

[19] X. Du, C. Behrens, and J. C. Niemeyer, *Substructure of fuzzy dark matter haloes*, Monthly Notices of the Royal Astronomical Society **465** (2017), no. 1 941–951, [$/oup/backfile/content_public/journal/mnras/465/1/10.1093_mnras_tw2724/3/stw2724.pdf$].

[20] P. Mocz, M. Vogelsberger, V. H. Robles, J. Zavala, M. Boylan-Kolchin, A. Fialkov, and L. Hernquist, *Galaxy formation with becdm âĂŞ i. turbulence and relaxation of idealized haloes*, Monthly Notices of the Royal Astronomical Society **471** (2017), no. 4 4559–4570, [$/oup/backfile/content_public/journal/mnras/471/4/10.1093_mnras_tx1887/1/stx1887.pdf$].

[21] H.-Y. Schive, T. Chiueh, and T. Broadhurst, *Cosmic structure as the quantum interference of a coherent dark wave*, Nature Physics **10** (July, 2014) 496–499, [`arXiv:1406.6586`].

[22] A. Paredes and H. Michinel, *Interference of dark matter solitons and galactic offsets*, Physics of the Dark Universe **12** (2016) 50 – 55.

[23] "https://hgomersall.github.io/pyfftw/."

[24] M. Frigo and S. G. Johnson, "http://www.fftw.org/."

[25] H.-Y. Schive, M.-H. Liao, T.-P. Woo, S.-K. Wong, T. Chiueh, T. Broadhurst, and W.-Y. P. Hwang, *Understanding the core-halo relation of quantum wave dark matter from 3d simulations*, Phys. Rev. Lett. **113** (Dec, 2014) 261302.

[26] G. Agrawal, *Nonlinear Fiber Optics*. Optics and Photonics Series. Academic Press, 2013.

[27] M. Frigo and S. G. Johnson, *The design and implementation of FFTW3*, Proceedings of the IEEE **93** (2005), no. 2 216–231. Special issue on "Program Generation, Optimization, and Platform Adaptation".

[28] "http://numexpr.readthedocs.io/."

[29] D. J. E. Marsh and A.-R. Pop, *Axion dark matter, solitons and the cuspâĂŞcore problem*, Monthly Notices of the Royal Astronomical Society **451** (2015), no. 3 2479–2492, [$/oup/backfile/content_public/journal/mnras/451/3/10.1093/mnras/stv1050/2/stv1050.pdf$].

[30] M. Adeel Ajaib, *Numerical Methods and Causality in Physics*, ArXiv e-prints (Feb., 2013) [`arXiv:1302.5601`].

[31] D. H. Weinberg, J. S. Bullock, F. Governato, R. Kuzio de Naray, and A. H. G. Peter, *Cold dark matter: Controversies on small scales*, Proceedings of the National Academy of Science **112** (Oct., 2015) 12249–12255, [`arXiv:1306.0913`].

[32] X. Du, B. Schwabe, J. C. Niemeyer, and D. Bürger, *Tidal disruption of fuzzy dark matter subhalo cores*, **97** (Mar., 2018) 063507, [`arXiv:1801.0486`].

[33] T. RindlerâĂŘDaller and P. R. Shapiro, *Angular momentum and vortex formation in boseâĂŞeinsteinâĂŘcondensed cold dark matter haloes*, Monthly Notices of the Royal

Astronomical Society **422** no. 1 135–161,
[https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1365-2966.2012.20588.x].