Numerical Investigation on the Evolution of oscillons

Majd Hamdan

Department of Physics, Middlebury College, Middlebury, VT

Project report for PHYS 0704

December 9, 2021

Abstract

The fundamental forces of Nature are nonlinear. Numerical simulation shows that real scalar field in a nonlinear theory give rise to long-lived localized lumps held together by their own interactions, known as oscillons. Understanding oscillons behavior could help in building more accurate models of the early universe. The goal of this project is to use numerical simulations to understand the formation and the lifetime of these lumps under given initial conditions. In particular, we attempt to reproduce some of the results reported in [1] and Fig 4.7 in [2]. Our results are in agreement with both sources.

Date Accepted: _____

Contents

I Introduction

In our Universe, there exist a variety of stable lumps that are kept together by the gravitational force. This is the case for planets, stars, solar systems and galaxies. A fundamental question in physics is how does matter form spatially localized lumps, and how do these lumps fall apart, evaporate, or radiate if at all. In nonlinear field theories, we can find lumps that are held together by their own interactions, with a rich array of properties. The traditional example of a dissipative solution is waves on surface of water. Throwing a stone into an unbound body of water creates waves that travel in all directions away from where the stone hit the surface. Energy is carried by the waves to a larger and larger area until it eventually dissipates. Imagine now touching water with your fingertip and slowly lifting your finger. This will bend the surface of water towards your fingertip because of the surface tension. Imagine further, contrary to reality, that this bending (disturbance) in the surface does not lead to propagating waves, but rather stays at its initial location and oscillate around the surface plane forever without dissipation. This is also a possible solution in Nonlinear field theory. While we can't observe such a solution on the surface of unbound body of water, there are other physical systems that posses non-dissipative solutions. Such non-dissipative solutions are called *solitons*. Solitons can either be time independent or periodic in time. In fact, the analogy of water just introduced is fundamental to the history of solitons, since they were first discovered in a water channel. In 1895, Korteweg and de Vries first identified hydrodynamical soliton when trying explain the following account of J. Scott Russell [3]:

> I was observing the motion of a boat which was rapidly drawn along a narrow channel by a pair of horses, when the boat suddenly stopped – not so the mass of water in the channel which it had put in motion; it accumulated round the prow of the vessel in a state of violent agitation, then suddenly leaving it behind, rolled forward with great velocity, assuming the form of a large solitary elevation, a rounded, smooth and well-defined heap of water, which continued its course along the channel apparently without change of form or diminution of speed. I followed it on horseback, and overtook it still rolling on at a rate of some eight or nine miles an hour, preserving its original figure some thirty feet long and a foot to a foot and a half in height. Its height gradually diminished, and after a chase of one or two miles I lost it in the windings of the channel. Such, in the month of August 1834, was my first chance interview with that singular and beautiful phenomenon which I have called the Wave of Translation.

The type of solutions of interest in in this paper are not entirely non-dissipative nor periodic, but they dissipate energy over a very long time-much longer than their period of oscillation. These types of solutions are described by many names, but throughout this paper we will call them *oscillons*. Here, we aim at studying the evolution of spherically symmetric scalar field configurations in symmetric

double-well potential to reproduce the results given by M. Gleiser in [1]. In particular, we examine the possibility that oscillons exist in a (3+1)-dimensional single real scalar field with self-interactions dictated by a double-well potential. Because the problem is essentially time dependent and nonlinear, we use a numerical approach, the Finite difference method. We begin by providing a brief history of oscillons and reasons why such solutions are relevant to modern physics. In the second part, we present the numerical method used to integrate the equation of motion, namely the Finite difference method. In the last part, we expand on the results of the simulation and describe the properties of oscillons predicted by our simulation.

II Background

II. (i) History and Motivation

The initial studies of oscillons were motivated by the investigation of whether elementary particles might be oscillating solutions of some nonlinear wave equation [4]. The interest in oscillons grew more quickly after 1994 [5], following the work of Marcelo Gleiser, who studied the formation of oscillons for symmetric and asymmetric double well potentials. Gleiser came to the study of oscillons by investigating cosmological phase transitions [1]. Since then, the existence of oscillons has also been shown in cosmic vacuum bubble collisions [6], bosonic sector of the standard model [7, 8], and supersymmetric theories [9]. The Big bang theory (BBT) predicts that a rapid space-time expansion should have taken place in the very short era just after the Big Bang. It also predicts that the early Universe was extremely hot and dense. However, in the model, the temperature of the very early Universe is nearly zero during inflation. This discrepancy implies that we need a dynamical mechanism to heat up the cold Universe after inflation. During this process the scalar field might be likely to form oscillons, which could provide a detectable peak in the spectrum of gravitational waves by the planned space-based gravitational wave observatories [10]. Therefore, understanding the existence and stability of oscillons, their lifetime, and their energy is of great important to verify current theories and provide insight onto what properties a theory should have to allow for their existence.

II. (ii) Equation of motion in Nonlinear scalar fields

Motions is most fundamentally described using the principle of action. The action, S, is typically defined as an integral of the *Lagrangian* over time along the path of the system under study:

$$S = \int_{t_1}^{t_2} L\, dt \tag{1}$$

In Newtonian mechanics, the Lagrangian is defined as the difference between the kinetic and potential energies, $L = T - V$. Using the principle of least action, given a set of particles with coordinates

$x_i(t)$, the *Euler's equation* gives an equation of motion for each particle:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}_i}\right) = \frac{\partial L}{\partial x_i} \tag{2}$$

Take for example the gravitational field, $V = -\frac{GMm}{r}$ and $L = m\dot{x}^2/2 - \frac{GMm}{r}$. Euler's equation gives: $m\ddot{x} = -\frac{GMm}{r^2}$. This is Newton's second law when only considering gravity. Gravitational field is linear because its potential is linear in the field itself ($\phi = V/m$). When V is polynomial in the ϕ, we have a nonlinear field.

The Lagrangian is written in terms of the *Lagrangian density* $\mathcal{L}$: $L = \int \mathcal{L}\, dv$. The equation of motion written in terms of $\mathcal{L}$ is now called the *Euler–Lagrange equation*:

$$\frac{\partial}{\partial x^\mu}\left[\frac{\partial \mathcal{L}}{\partial\left(\partial_\mu \phi\right)}\right] = \frac{\partial \mathcal{L}}{\partial \phi} \tag{3}$$

where ϕ is a single real scalar field, $\partial_\mu \phi = \frac{\partial}{\partial x^\mu}\phi$, and $x^\mu = (ct, x_0, x_1, \ldots, x_n)$. When V is polynomial in ϕ ($V \propto \phi^n$ for $n > 2$), it is said that the field is self-interacting, because the Euler-Lagrange equation is now nonlinear, implying self-interaction. This equation describes the motion of the field-no matter the number of dimensions used- at every point in the space-time. A solution to this equation is a description of the configuration allowed to exist by the physically possible motion of the scalar field ϕ. There are two requirements that are necessary for the formation of oscillons[11]:

1. The initial configuration of the field must be non-homogeneous(doesn't have the same magnitude and direction at any place)
2. The scalar potential $V(\phi)$ needs to contain attractive self-interactions so self-interactions can sustain the lump

Many models satisfy these conditions. In the present work, we focus on models involving only a single real (3+1)-dimensional scalar field with self-interactions dictated by a double-well potential. In particular, we will study the simplest case where the scalar field is spherically symmetric. The symmetric double-well potential (SDWP) will be considered. For SDWP, the potential is defined as:

$$V_S(\phi) = \frac{\lambda}{4}\left(\phi^2 - \frac{m^2}{\lambda}\right)^2 \tag{4}$$

where the subscripts S stand for the SDWP, m is the mass of the field, and λ is a dimensionless coupling constant. The SDWP is shown in Fig.1.

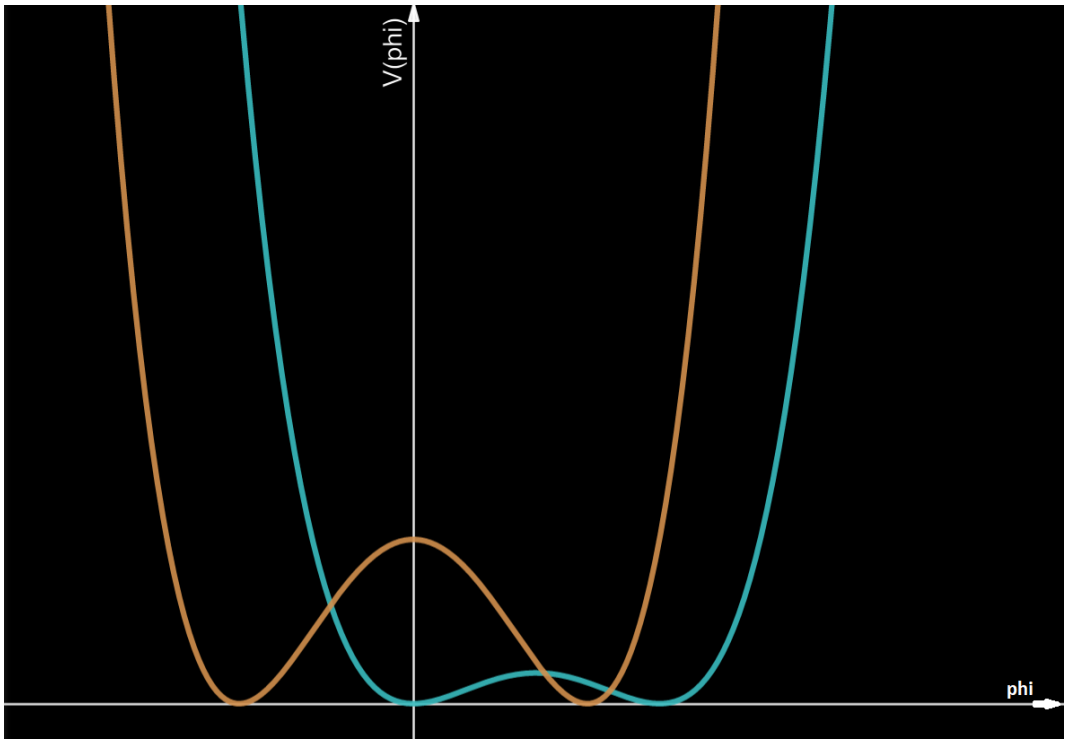

Fig. 1. Example of the SDWP (orange). Blue is an example of an asymmetric double well potential (ADWP)

III Method

III. (i) Formulation of the Problem [1]

Throughout this work, we use natural units unless explicitly otherwise stated ($\hbar = c = k_b = 1$). In order to understand how oscillons evolve in time, we must find their equation of motion. The action for a single (3+1) scalar field is given by[12]:

$$S[\phi] = \int d^4x \left[\frac{1}{2}\partial_\mu\phi\partial^\mu\phi - V(\phi)\right] \tag{5}$$

where $\mu = 0, 1, 2, 3$ and the Lagrangian is: $L = \frac{1}{2}\partial_\mu\phi\partial^\mu\phi - V(\phi)$. Then, the equation of motion reads:

$$\ddot{\phi} - \nabla^2\phi = -\frac{\partial V(\phi)}{\partial\phi}$$

We now consider spherical coordinates in the space dimension. For a configuration with spherical symmetry (ϕ is independent of θ and φ), the covariant derivative turns into an ordinary differential in terms of r, $\nabla^2\phi = \partial^2/\partial_r^2$. Let $\Phi = \frac{\sqrt{\lambda}}{m}\phi, \rho = mr$, and $\tau = mt$. Using Eq.(4), the equation of motion becomes:

$$\frac{\partial^2\Phi}{\partial\tau^2} - \frac{\partial^2\Phi}{\partial\rho^2} - \frac{2}{\rho}\frac{\partial\Phi}{\partial\rho} = \Phi - \Phi^3 \tag{6}$$

This is called the Klein-Gordon equation. the potential minima are at $\Phi = -1$ and $\Phi = 1$ for the SDWP.

Now that we have the equation of motion, more specification is required. From now one, we will call the spherically symmetric configurations we are studying bubbles. Let the bubble's initial radius be R_0. The main interest of this paper, is studying the lifetime of the oscillons. A reasonable measure of whether the oscillons have decayed is its energy level. The oscillons will start from initial enery E_0, radiate energy suddenly and decay [1]. A study of the rate of the energy radiation is the goal of this paper as it serves as a good indicator of the oscillon evolution. Energy is given by:

$$E\left[\phi_0\right] = \int d^3x \left[\frac{1}{2}\dot{\phi}_0^2 + \frac{1}{2}\left(\nabla\phi_0\right)^2 + V\left(\phi_0\right)\right] \tag{7}$$

When integrating over the whole space, this equation will give the total energy, which is constant. We can measure the energy of the oscillons by surrounding the initial configuration with a sphere of large radius, $R \gg R_0$, and measuring the energy inside it. We make the following definitions:

$$E_K(t) = 2\pi \int_0^{\epsilon R_0} dr r^2 \left(\frac{\partial\phi(r,t)}{\partial t}\right)^2 \tag{8}$$

$$E_S(t) = 2\pi \int_0^{\epsilon R_0} dr r^2 \left(\frac{\partial\phi(r,t)}{\partial r}\right)^2 \tag{9}$$

$$E_V(t) = 4\pi \int_0^{\epsilon R_0} dr r^2 V\left(\phi_0(r,t)\right) \tag{10}$$

where E_k is the kinetic energy, E_s is the surface energy, and E_v is the volume energy of the bubble and $R = \epsilon R_0$ where $\epsilon \geq 2$ is a dimensionless constant. The total energy within radius ϵR_0 is then:

$$E\left[\phi_0\right](t) = E_K(t) + E_S(t) + E_V(t) \tag{11}$$

When integrated over the whole space (ϵR_0 cover the whole volume of simulation), this equation can serve as a prove of the validity of our computation as it should stay constant when integrated over the whole space.

Eq.(6) must be solved to produce the bubble profile needed to find the energies in Eqs (8)-(10). Since Eq.(6) is a second-order equation in time two initial conditions at $\tau = 0$, and two boundary conditions in space must be specified. We impose the following conditions:

$$\begin{aligned}
\Phi(\rho,0) &= 2e^{-\rho^2/\rho_0^2} - 1, && \rho \in [0,L] \\
\frac{\partial\Phi(\rho,0)}{\partial\tau} &= 0, && \rho \in [0,L] \\
\frac{\partial\Phi(0,\tau)}{\partial\rho} &= 0, && \tau \in (0,T] \\
\Phi(L,\tau) &= -1, && \tau \in (0,T]
\end{aligned}$$

The first condition presents the initial profile of the bubble. In this work, we will investigate a Guassian bubble. This condition is generally of the form $(\Phi_c - \Phi_0)e^{-\rho/R_0^2} + \Phi_0$ where Φ_c is the value of the field at the bubble core and Φ_0 is one of the potential minima [2]. We take Φ_c as the other potential minima, $\Phi_c = 1$, giving the bubble a linear size of about $2R_0$. This explains the restriction placed on ϵ ($\epsilon \geq 2$) earlier. The second condition means that the bubble starts its evolution at rest. The third condition guarantees regularity at the center (mirror symmetric with respect to $\rho = 0$) The last condition simply forces the bubble to go back to Φ_0 at the boundary.

III. (ii) Algorithm

III. (ii).1 ρ Discretization

We start with the given equation and conditions:

$$\frac{\partial^2\Phi}{\partial\tau^2} - \frac{\partial^2\Phi}{\partial\rho^2} - \frac{2}{\rho}\frac{\partial\Phi}{\partial\rho} = \Phi - \Phi^3, \qquad \rho \in [0, L], \tau \in [0, T] \tag{12}$$

$$\Phi(\rho, 0) = 2e^{-\rho^2/\rho_0^2} - 1, \qquad \rho \in [0, L] \tag{13}$$

$$\frac{\partial\Phi(\rho, 0)}{\partial\tau} = 0, \qquad \rho \in [0, L] \tag{14}$$

$$\frac{\partial\Phi(0, \tau)}{\partial\rho} = 0, \qquad \tau \in [0, T] \tag{15}$$

$$\Phi(L, \tau) = -1, \qquad \tau \in [0, T] \tag{16}$$

To solve this PDE, the finite difference method is used to manipulate the $\frac{\partial^2}{\partial\rho^2}$ and $\frac{\partial}{\partial\rho}$ terms in order to reduce the partial differential equation to a set of ordinary differential equations. To do so, we introduce a spatial mesh in Ω with mesh points

$$\rho_0 = 0 < \rho_1 < \rho_2 < \cdots < \rho_N = L$$

we assume the distance between each mesh point is constant. We relax the condition that the PDE holds at all points in the space-time to the requirement that the PDE is fulfilled at the interior (not the boundary points) mesh points only. Thus the equation becomes:

$$\frac{\partial^2\Phi(\rho_i, \tau)}{\partial\tau^2} - \frac{\partial^2\Phi(\rho_i, \tau)}{\partial\rho^2} - \frac{2}{\rho_i}\frac{\partial\Phi(\rho_i, \tau)}{\partial\rho} = \Phi(\rho_i, \tau) - \Phi^3(\rho_i, \tau) \tag{17}$$

for $i = 1, \ldots, N-1$. At any point ρ_i we can approximate $\frac{\partial^2\Phi}{\partial\rho^2}$ by a finite difference:

$$\frac{\partial^2\Phi(\rho_i, \tau)}{\partial\rho^2} \approx \frac{\Phi(\rho_{i+1}, \tau) - 2\Phi(\rho_i, \tau) + \Phi(\rho_{i-1}, \tau)}{\Delta\rho^2} \tag{18}$$

and approximate $\frac{\partial\Phi}{\partial\rho}$ by a finite difference:

$$\frac{\partial\Phi(\rho_i, \tau)}{\partial\rho} \approx \frac{\Phi(\rho_{i+1}, \tau) - \Phi(\rho_{i-1}, \tau)}{2\Delta\rho} \tag{19}$$

We substitute Eq.(18) and Eq.(19) in Eq.(17) for every mesh point ρ_i to get:

$$\frac{\partial^2 \Phi_i}{\partial \tau^2} - \frac{\Phi\left(\rho_{i+1}, \tau\right) - 2\Phi(\rho_i, \tau) + \Phi(\rho_{i-1}, \tau)}{\Delta\rho^2} - \frac{\Phi(\rho_{i+1}, \tau) - \Phi(\rho_i, \tau)}{\Delta\rho} = \Phi(\rho_i, \tau) - \Phi^3(\rho_i, \tau), \quad (20)$$

for $i = 1, \ldots, N-1$. This is a system of ordinary differential equations in $N-1$ unknowns: $\Phi_1(t), \ldots, \Phi_{N-1}(t)$.

Now that we have the discretized PDE, we must also discretize the boundary conditions as well. Because we have N-1 ODEs, we need to provide initial conditions and boundary conditions for all of them. The initial condition $\Phi(\rho, 0) = 2e^{-\rho^2/\rho_0^2} - 1$ becomes: $\Phi_i(\rho_i, 0) = 2e^{-\rho_i^2/\rho_0^2} - 1$. Doing the same for the rest of the conditions gives:

$$\Phi_i(\rho_i, 0) = 2e^{-\rho_i^2/\rho_0^2} - 1 \quad (21)$$

$$\frac{\partial \Phi(\rho_i, 0)}{\partial \tau} = 0, \quad (22)$$

$$\frac{\partial \Phi(0, \tau)}{\partial \rho} = 0, \quad (23)$$

$$\Phi(\rho_N, \tau) = -1, \quad (24)$$

Given Eqs.(20)-(24), all we have to do now is plug these equation into an ODE solver. This will give us $\Phi_1(t), \ldots, \Phi_{N-1}(t)$ at every t. $\Phi(\rho_0, \tau)$ and $\Phi(\rho_N, \tau)$ are given by the boundary conditions. The algorithm was programmed in Python where we use scipy.integrate.odeint[1] as the ODE solver.

III. (ii).2 ρ and τ Discretization

Computing the solution using the prebuilt solver took a long time (about one hour for $N_\rho = 400$ and $\Delta\rho = 0.5$) which limited our ability to acquire accurate results by discretizing the mesh more and by increasing N_ρ. The bubble energy, while it followed the same trend describe by Gleiser in [1], showed many oscillations. To avoid using the solver and to double check our computation, we can also manually discretize τ. We introduce two uniform meshes, with constant mesh spacing $\Delta\tau$ and $\Delta\rho$:

$$\rho_i = i\Delta\rho, i = 0, \ldots, N_\rho, \quad \tau_n = n\Delta\tau, n = 0, \ldots, N_\tau$$

where $\Delta\rho = \rho_i - \rho_{i-1}, i = 1, \ldots, N_\rho$, and $\Delta\tau = \tau_n - \tau_{n-1}, n = 1, \ldots, N_\tau$. We start by discretizing Eqs. (12)-(16).

We relax the condition that the PDE holds at all points in the space-time domain $(0, L) \times (0, T)$ to the requirement that the PDE is fulfilled at the *interior* mesh points only. We get an equation like Eq.(17) but τ is replaced with τ_n. Then, using Eqs.(18) and (19), we replace the derivatives in ρ and τ

[1] https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.odeint.html

by central differences obtaining:

$$\frac{\Phi\left(\rho_i,\tau_{n+1}\right)-2\Phi\left(\rho_i,\tau_n\right)+\phi\left(\rho_i,\tau_{n-1}\right)}{\Delta\tau^2}-\frac{\Phi\left(\rho_{i+1},\tau_n\right)-2\Phi\left(\rho_i,\tau_n\right)+\phi\left(\rho_{i-1},\tau_n\right)}{\Delta\rho^2}$$
$$-\frac{2}{\rho_i}\frac{\Phi\left(\rho_{i+1},\tau_n\right)-\Phi\left(\rho_{i-1},\tau_n\right)}{2\Delta\rho}=\Phi(\rho_i,\tau_n)-\Phi^3(\rho_i,\tau_n) \tag{25}$$

A typical feature of Eq.(25) is that it involves Φ values from neighboring points only: $\Phi(\rho_i,\tau_{n\pm1})$, $\Phi(\rho_{i\pm1},\tau_n)$, and $\Phi(\rho_i,\tau_n)$. This is shown in Fig.2

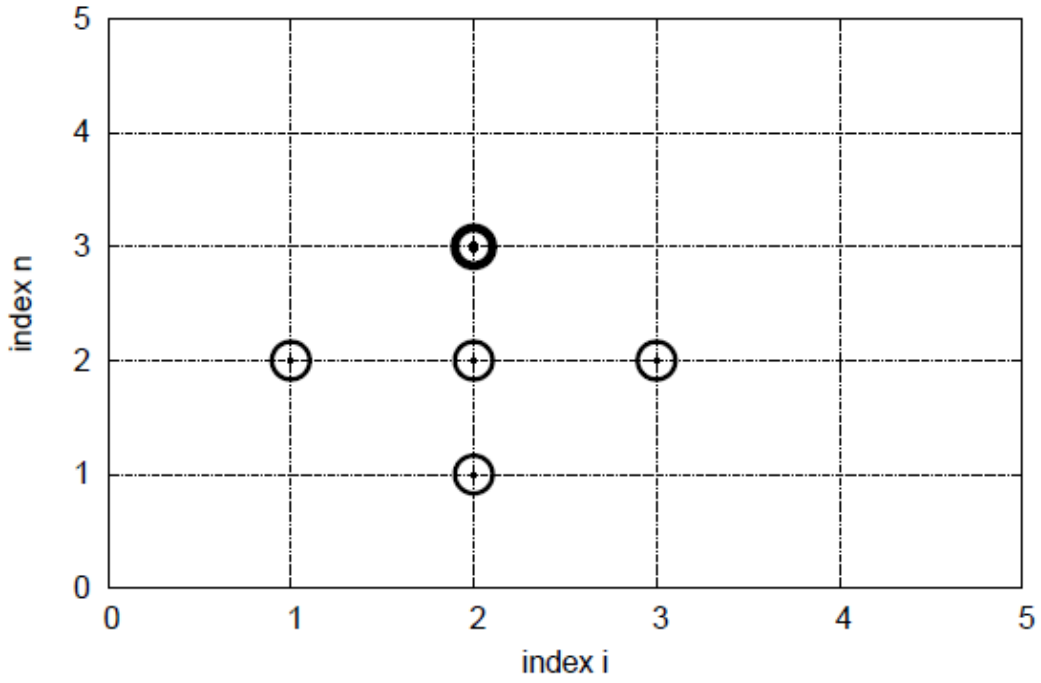

Fig. 2. Mesh in space and time. The circles show points connected in a finite difference equation, where the bolded circle represent $\Phi(\rho_i,\tau_{n+1})$. (from [13])

We assume that $\Phi(\rho_i,\tau_n)$, $\Phi(\rho_i,\tau_{n-1})$, $\phi(\rho_{i-1},\tau_n)$, and $\Phi(\rho_{i+1},\tau_n)$ are already known for $i=0,\ldots,N_\rho$. The only unknown is Eq.(25) is $\Phi(\rho_i,\tau_{n+1})$. By solving Eq.(25) for $\Phi(\rho_i,\tau_{n+1})$, we get:

$$\Phi\left(\rho_i,\tau_{n+1}\right)=\Delta\tau^2\left(\Phi(\rho_i,\tau_n)-\Phi^3(\rho_i,\tau_n)+\frac{2}{\rho_i}\frac{\Phi\left(\rho_{i+1},\tau_n\right)-\Phi\left(\rho_{i-1},\tau_n\right)}{2\Delta\rho}\right.$$
$$\left.+\frac{\Phi\left(\rho_{i+1},\tau_n\right)-2\Phi\left(\rho_i,\tau_n\right)+\phi\left(\rho_{i-1},\tau_n\right)}{\Delta\rho^2}\right)+2\Phi\left(\rho_i,\tau_n\right)-\Phi\left(\rho_i,\tau_{n-1}\right) \tag{26}$$

for $i=1,\ldots,N_\rho-1$ and $n=1,\ldots,N_\tau-1$. Therefore, to solve the PDE, we need to find $\Phi\left(\rho_i,\tau_{n+1}\right)$ for every $n=1,\ldots,N_\tau-1$ by applying Eq.(26) for every $i=1,\ldots,N_\rho-1$.

The first initial condition reads:

$$\Phi(\rho_i,0)=2e^{-\rho_i^2/\rho_0^2}-1$$

If n = 0, a problem with Eq.(26) arises since the formula for $\Phi(\rho_i, \tau_1)$ involves $\Phi(\rho_i, \tau_{-1})$. But if we replace the first derivative in the second initial condition by a finite difference approximation of the type:

$$\frac{\partial \Phi(\rho_i, 0)}{\partial \tau} = \frac{\Phi\left(\rho_i, \tau_1\right) - \Phi\left(\rho_i, \tau_0\right)}{\Delta \tau} = 0$$

we get:

$$\Phi\left(\rho_i, \tau_1\right) = \Phi\left(\rho_i, \tau_0\right) = 2e^{-\rho_i^2/\rho_0^2} - 1. \tag{27}$$

Since $\Phi(-1, \tau_n)$ is outside the mesh, we apply a centered difference of the type:

$$\frac{\partial \Phi(0, \tau_n)}{\partial \rho} = \frac{\Phi(1, \tau_n) - \Phi(0, \tau_n)}{\Delta \rho} = 0$$

on Eq.(15) obtaining:

$$\Phi(1, \tau_n) = \Phi(0, \tau_n) \tag{28}$$

Eq.(16) reads:

$$\Phi(\rho_N, \tau_n) = -1 \tag{29}$$

Now we have all we need to build the solution. We can summarize the computational algorithm by the following steps:

1. Compute $\Phi(\rho_i, 0)$ and $\Phi(\rho_i, 1)$ using Eq.(27) for $i = 0, \ldots, N_\rho$
2. Compute $\Phi(N_\rho, \tau_n)$ using Eq.(29) for $n = 0, \ldots, N_\tau$.
3. For each step level $n = 1, 2, \ldots, N_\tau - 1$, apply Eq.(26) to find $\Phi(\rho_i, \tau_{n+1})$ for $i = 1, \ldots, N_\rho - 1$ setting $\Phi(0, n+1) = \Phi(1, n+1)$ at every iteration by Eq.(28)

III. (iii) Energy Computation

Now that we have $\Phi(\rho_i, \tau_n)$ for every point on the mesh, we can verify our computation by finding the total energy and the bubble energy. If our computation is correct, the total energy must be conserved.

Given $\Phi(\rho, \tau)$ at every mesh point, we must discretize the energy expressions in Eqs.(8)-(10). This will give:

$$E_K(\tau_n) = 2\pi \sum_{\rho=0}^{\epsilon R_0} d\rho\, \rho_i^2 \left(\frac{\Phi\left(\rho_i, \tau_{n+1}\right) - \Phi\left(\rho_i, \tau_{n-1}\right)}{2\Delta \tau} \right)^2 \tag{30}$$

$$E_S(\tau_n) = 2\pi \sum_{\rho=0}^{\epsilon R_0} d\rho\, \rho_i^2 \left(\frac{\Phi\left(\rho_{i+1}, \tau_n\right) - \Phi\left(\rho_{i-1}, \tau_n\right)}{2\Delta \rho} \right)^2 \tag{31}$$

$$E_V(\tau_n) = 4\pi \sum_{\rho=0}^{\epsilon R_0} d\rho\, \rho_i^2 V\left(\Phi(\rho_i, \tau_n)\right) \tag{32}$$

$E_k(0) = 0$ by Eq.(27). We replace the central difference formula by $\frac{\Phi(\rho_i,N_\tau)-\Phi(\rho_i,N_\tau-1)}{\Delta\tau}$ in Eq.(30) to find $E_k(N_\tau)$. We perform the same replacement for $E_S(N_\tau)$ but in term of ρ. $E_S(0) = 0$ by Eq.(28).

IV Results and Discussion

The following figure displays the total energy computation performed for $R_0 = 4$, $N_\rho = 400$ and $\Delta\rho = 0.5$. The figure shows promising results. The total energy is conserved to 1 part in 8 and the total energy value is close to the total energy found by Gleiser in [1], about 180.

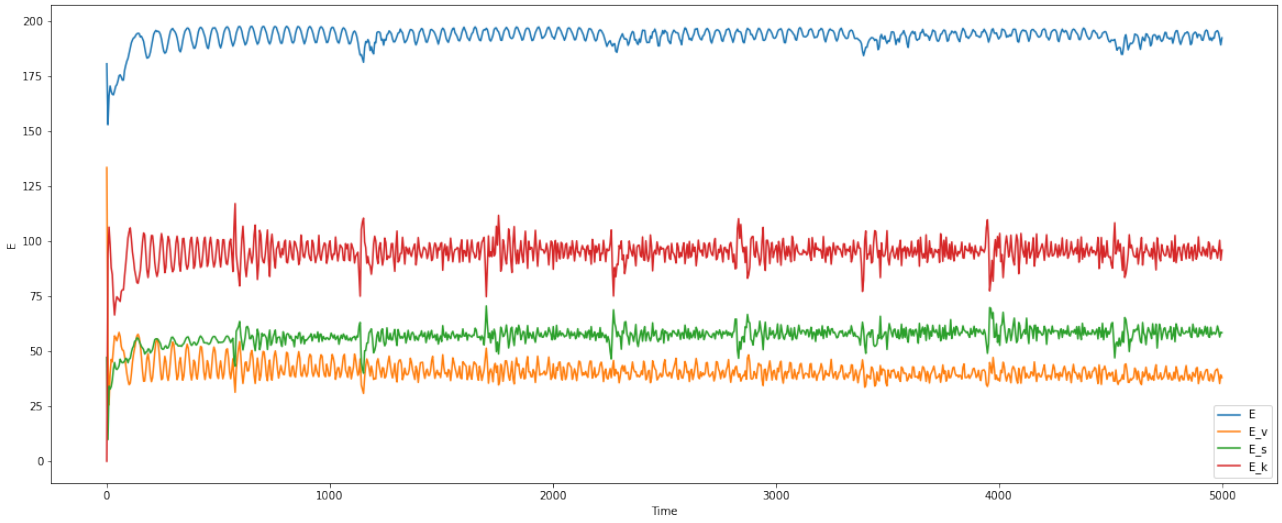

Fig. 3. Total Energy from computation (Method 1). Oscillations decrease in size with time.

Fig.4 displays the total energy of the bubble. This curve exhibit similar properties to Fig.5 from [1]. Notably, the oscillon starts with energy E_0, quickly shrinks by shedding energy, exists in a period of stability where practically no energy is radiated away (in our simulation its very short), then quickly radiates its remaining energy away. The behavior is shown for multiple points in time with a period of approximately $1000m^{-1}$, contrary to what is displayed in Fig.5. We are not certain what is causing the decay period to repeat. One possible reason for this behavior is that radiation can reflect of the lattice boundary and interfere with the bubble's evolution within the lattice. Here, we used a small static lattice. One obvious solution would be to use a very large static grid or to use a dynamically increasing lattice (the grid increases in size to avoid radiation reflection). However, any increase in the size or discretization of the lattice caused the algorithm to run out of memory. This pushed us to use method 2 as verification to method 1 and as a possible alternative that could handle a large lattice. Method 2 showed the same results in [1], thus it must be an error in the code or it must be that the integrator used is not designed to solve this type of ODEs. We did not use method 1 for the rest of the computation.

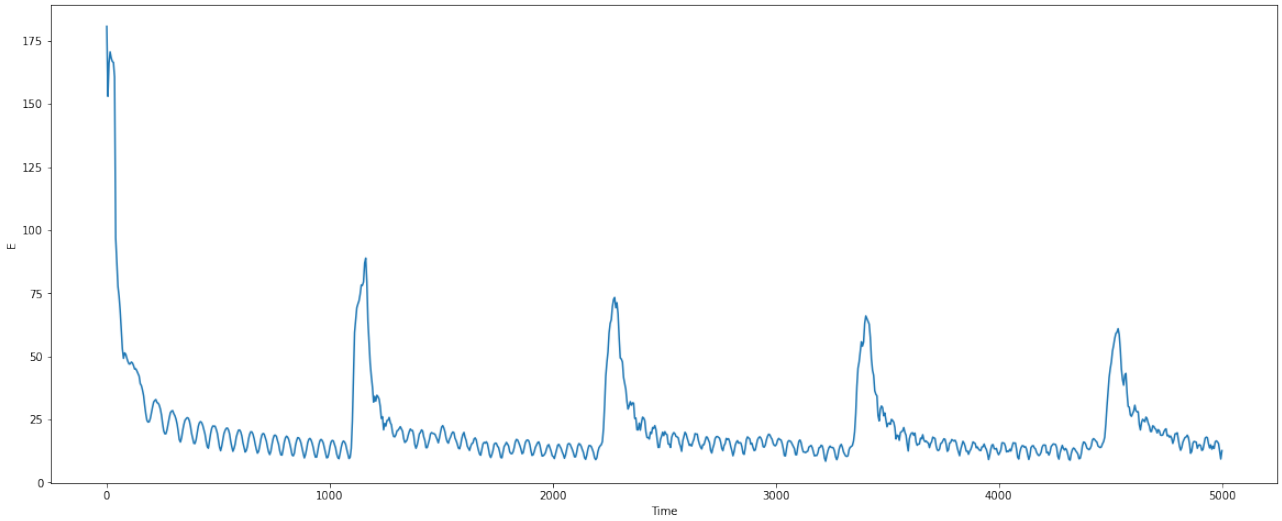

Fig. 4. Energy within a spherical shell surrounding initial configurations of $R_0 = 4$ as a function of time (Method 1)

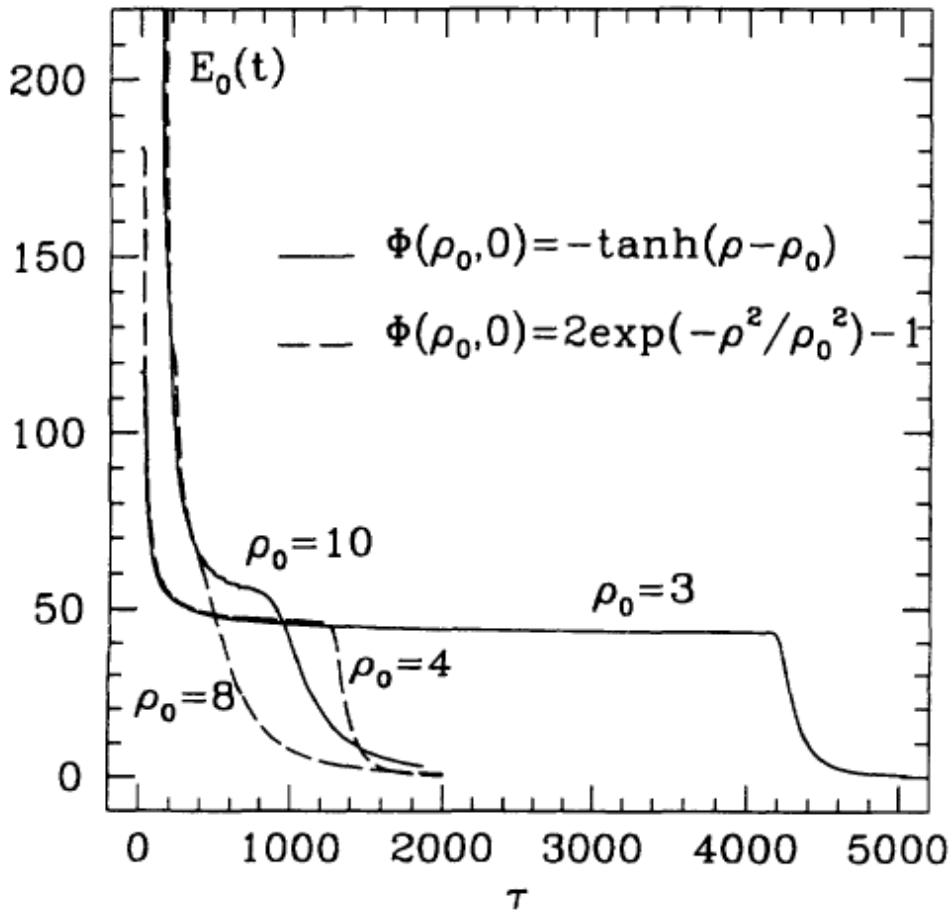

Fig. 5. Energy within a spherical shell surrounding several initial configurations as a function of time for the SDWP. Larger bubbles have shorter lifetimes.(From [1])

Now, we display the results of the computation using method 2. We used a mesh with $N_\rho = 2000$, $N_\tau = 5000$, $\Delta\tau = 0.1$, and $\Delta\rho = 0.2$ for most of the following results. This method was much faster than method 1. It performed the computation for a finer and larger mesh than the one used in method 1 in about 5 minutes. Fig.6 displays the total energy computation performed for $R_0 = 4$. Energy is conserved to approximately 1 part in 600. The total energy is about the same as the total energy found by Gleiser in [1].

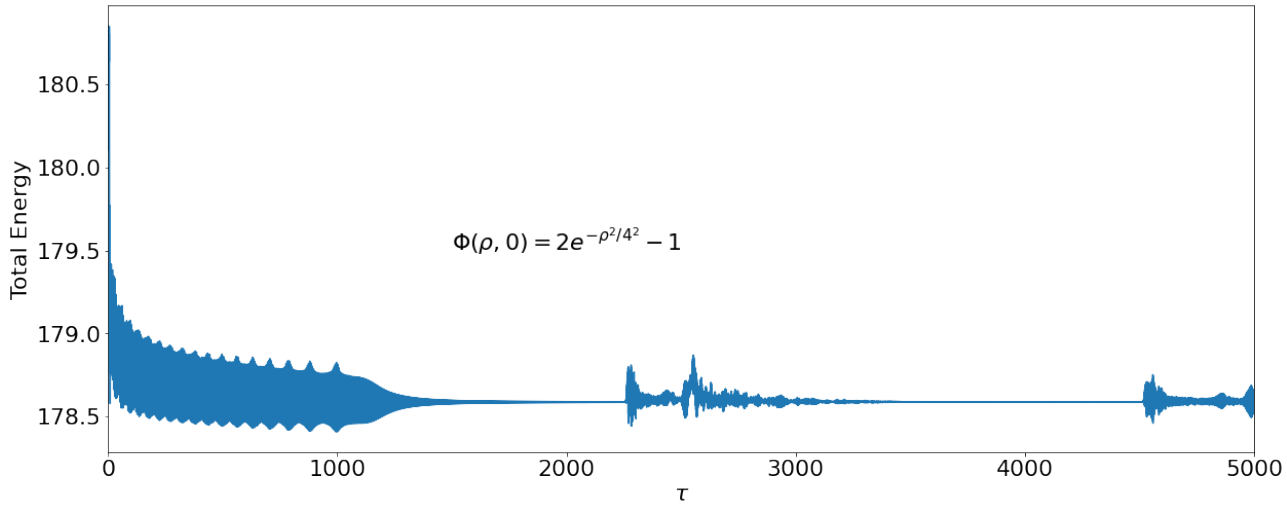

Fig. 6. Total energy as a function of time with $R_0 = 4$ (Method 2)

The total energy maximally deviate from 178.7 between 0 and $1200/m$- around the same time interval for which the oscillon with $R_0 = 4$ existed (see Fig.7). Using method 2, we computed the bubble energy for bubbles of several initial radii R_0. The results are displayed in Fig.7, and Fig.10 and Fig.11 in Appendix A.

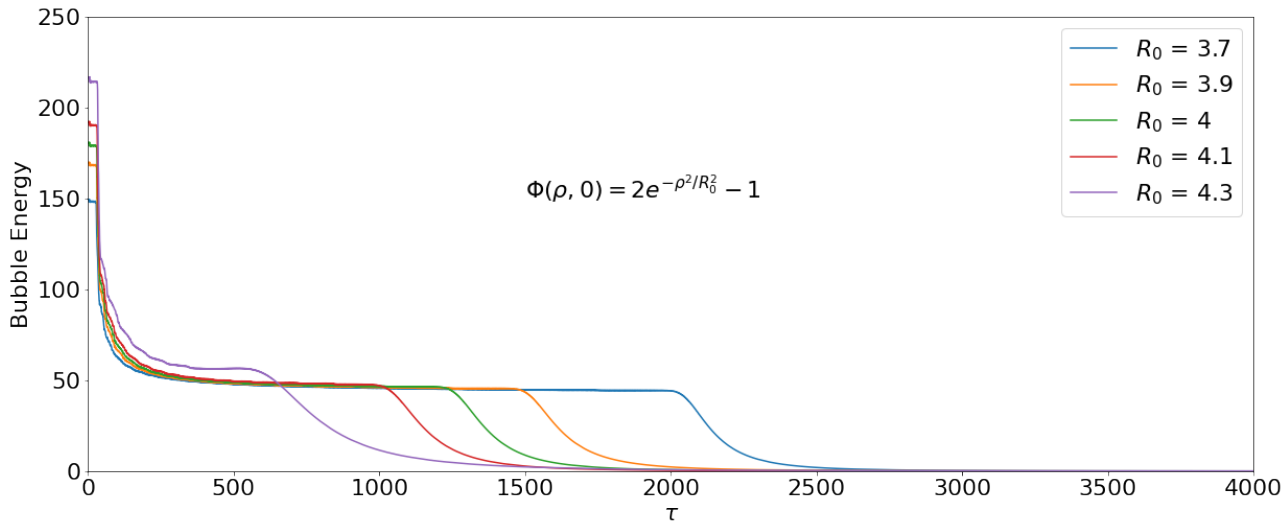

Fig. 7. Energy within a spherical shell surrounding initial configurations of $R_0 \in [3.7, 4.3]$ as a function of time (Method 2)

The results are in agreement with Fig.5. It is clear that bubbles energy is dependent on R_0. With further investigation, we found that bubbles with $R_0 < 2.4$ and $R > 4.3$ quickly disappear. The most interesting bubbles had $2.4 \leq R_0 \leq 4.3$. They settle into a long period of stability where no energy is radiated away. We observe that bubbles, regardless of their initial energy, have almost the same energy when the bubbles are in their stable stage. Gleiser named this stage *the oscillon stage* [1]. This stage can last more than $10^3\, m^{-1}$. It can even last for larger than $5 \times 10^3\, m^{-1}$ for $2.8 \leq R_0 \leq 3.1$ where we had to increase our mesh boundaries to find when they decay. Fig(7) shows the same evolution stages described by Gleiser in [1] with large enough radius. First, the bubble quickly sheds its initial energy and enters its oscillon stage with $E \approx= 50m/\lambda$. Second, the bubble stay in the oscillon stage for $\tau > 10^3 m^{-1}$ where the field oscillate between the two potential minima. Lastly, the oscillons quickly radiate its energy.

To show the relationship between the oscillon life time and the initial radius R_0, we performed method 2 for $R_0 = 1.9$ to $R_0 = 4.8$ with 0.1 step. We then plotted the lifetime of the oscillon against R_0. The results are displayed in the following figure:

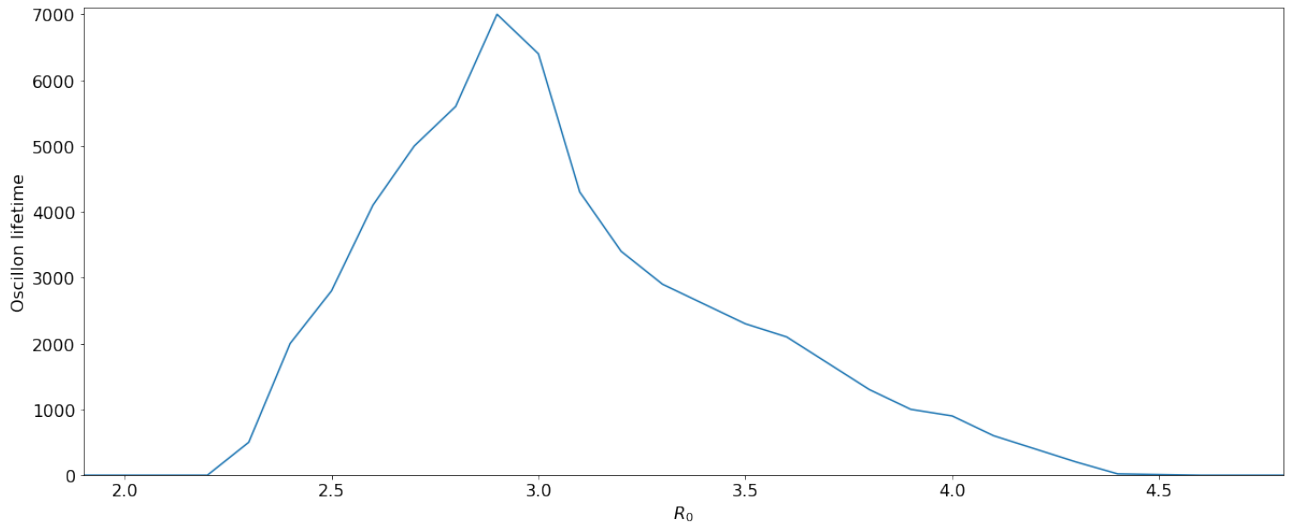

Fig. 8. Oscillon lifetime as a function of initial radius for $1.9 \leq R_0 \leq 4.7$ (Method 2)

The longest living oscillon, with $\tau \approx 7 \times 10^3$, comes from an initial bubble with a Guassian initial configuration with radius $R_0 = 2.8$. Fig.8 is in general agreement with Fig.9 from [2] except for the existence of resonance. From Fig.8, we arrive at the conclusion that for oscillons formation, not only the initial configuration of the field must be non-homogeneous and the scalar potential must be non-linear in ϕ, but also the bubble energy must be in the interval defined by E_{eff} must be in the interval $E_{R_0=2.4} \leq E_{eff} \leq E_{R_0=4.3}$.

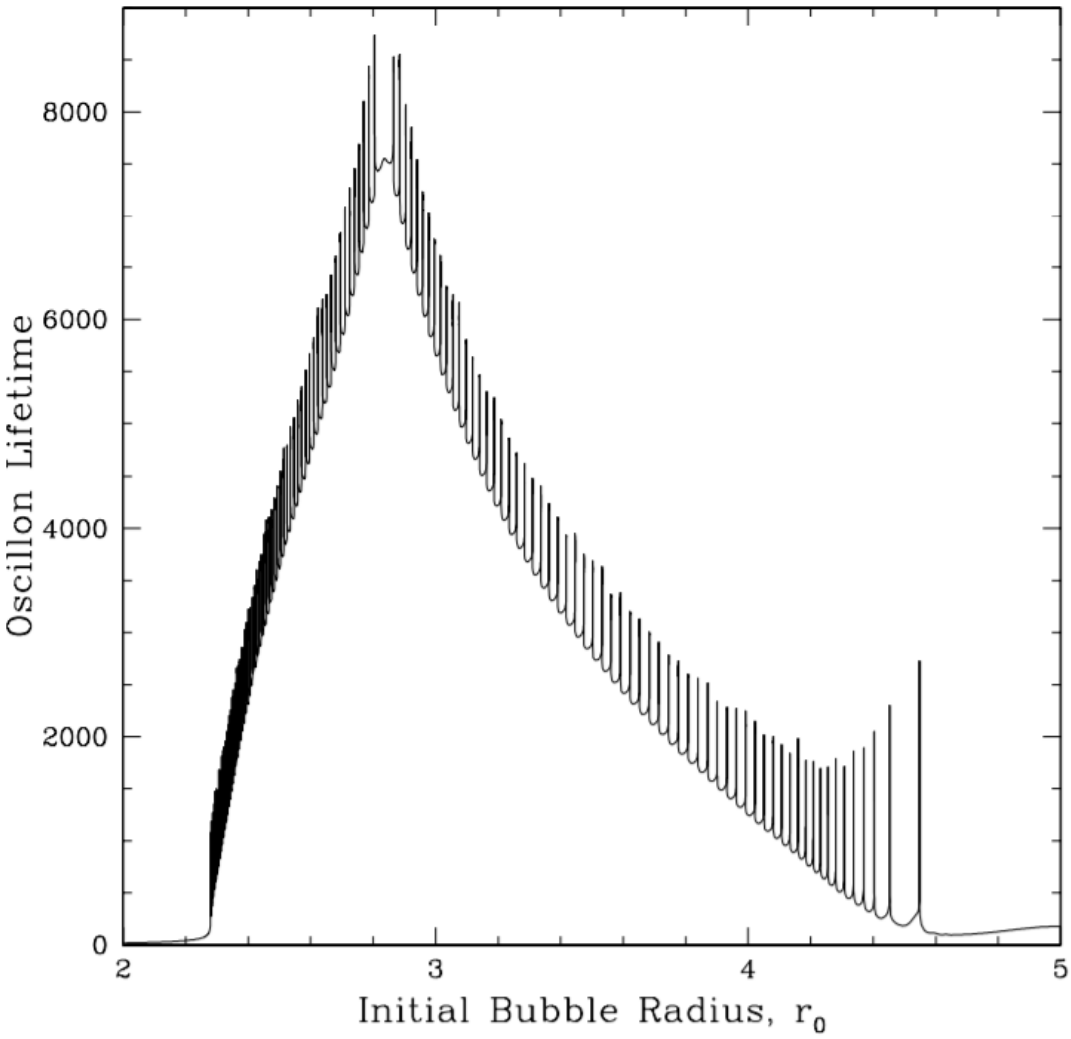

Fig. 9. Oscillon lifetime versus initial bubble radius for $2.0 \leq R_0 \leq 5.0$ from [2].

V Future work

There are many different directions we could take this research. One obvious route we could take is to address two main lingering questions from the numerical results:

1. Why no oscillons are possible for $R_0 > 4.3$ and $R_0 < 2.4$?
2. How does oscillons finally collapse?

To answer these questions we must investigate oscillons closely. We expect that a detailed analysis of the oscillons period, radius, and kinetic, surface, and volume energies will provide some insight into answering these questions. Moreover, we only investigated spherically symmetric bubbles in a (3+1)-dimensional single real scalar field. To understand the oscillons effects on modern theories, we have to investigate their evolution in more complex situations (no spherical symmetry, complex scalar field, collisions, more than one scalar field...)

Another route is to reduce the error and improve the speed of computations. A possible method to improve the speed of computation is *vectorization*. The computational algorithm for solving the PDE visits one mesh point at a time and evaluates a formula for the new value $\Phi(\rho_i, \tau_{n+1})$ at that point. This is implemented by using two loops, one for i and one for n. Such loops may run slowly in Python. Vectorization allows to perform operations on entire arrays instead of working with one element at a time. Operations on whole arrays are possible if the computations involving each element are independent of each other, so the operations can be performed simultaneously. This allow us to exploit parallel computing. This would be especially useful when discretizing both τ and ρ. To decrease error in the results, we need $\Delta\tau$ and $\Delta\rho$ to be as small as possible, but smaller deltas means more computation time. If we are able to perform the same computation faster, we would have more freedom in choosing $\Delta\tau$ and $\Delta\rho$. Vectorization could also allow for the use of large static grid, reducing the effect of reflected radiation on the evolution of oscillons.

In this work, we used the most default call to scipy.integrate.odeint without making use of it parameters. Odeint allows for the control of absolute and relative error tolerances. It would be interesting to investigate how these parameters would affect the computation speed and accuracy.

One more route we could take is to explore how different models would affect the lifetime of oscillons. A General definition of Lagrangian density is given as:

$$\mathcal{L}_M = -\sqrt{-g}\left[\frac{1}{2}\nabla^a\phi\nabla_a\phi + U(\phi)\right]$$

where $U(\phi)$ is the potential determining the self-interaction of the scalar field, ∇_a is the derivative operator belonging to the metric g_{ab}, and the determinant of the metric is g. The tensor g_{ab} describes the geometry of the space-time we are studying. In this paper, we studied the case where $g_{ab} = \eta_{ab}$ = diag(-1, 1, 1, 1). Other metrics would lead to different expressions for the equation of motion. It would be interesting to investigate what kind of properties and metrics allow for oscillons and how a change of metric would affect the properties of oscillons.

VI Conclusion

In this work, we used numerical simulations to investigate the properties of oscillons in a (3+1)-dimensional single real scalar field with self-interactions dictated by a double-well potential in an attempt to reproduce some of the results reported in [1] and Fig 4.7 in [2]. Our results largely agree with both sources. We found that there are 3 stages for the evolution of a Gaussian bubble with radius R_0 in the SDWP with an initial Gaussian configuration. The bubble will first quickly sheds its initial energy until it reaches $E \approx 50m/\lambda$, stabilize into an oscillon configuration for $\tau > 10^3 m^{-1}$, then finally, it quickly radiate its remaining energy. These bubble will only enter the oscillon stage if $2.4 \leq R_0 \leq 4.3$, otherwise, they will decay quickly. Another remarkable results is that all oscillons have the same

energy (around $50\lambda/m$) which is independent of the initial radius R_0. Through these results, we understand why oscillons are defined as *long lived, localized, time-dependent field configuration with nearly constant energy*.

There are many routes we can take this work. Most notably is the investigation of why no oscillons are possible for $R_0 > 4.3$ and $R_0 < 2.4$ and the investigation of the final collapse of the oscillon. A detailed analysis on the oscillons period, radius, kinetic, surface, and volume energy can offer some insight into answering these questions. We can reduce the error and improve the speed of our computations by vectorizing our code. Rather than visiting every mesh point at a time, we perform operations on entire arrays through vectorization. Vectorization will also allow for the use of large static grid,reducing the effect of reflected radiation on the evolution of oscillon.

Acknowledgements

We thank Professor Noah Graham and Professor Paul Hess of the Middlebury Physics department for the simulating discussions and support throughout this project. Thank you for making this project possible.

References

[1] M. Gleiser, "Pseudostable bubbles," *Phys. Rev. D*, vol. 49, pp. 2978–2981, Mar 1994.

[2] E. P. Honda, "Resonant dynamics within the nonlinear klein-gordon equation: Much ado about oscillons," 2000. https://arxiv.org/abs/hep-ph/0009104.

[3] M. Dunajski, *Solitons, Instantons, and Twistors*. Oxford University Press, 2015.

[4] G. H. Derrick, "Comments on nonlinear wave equations as models for elementary particles," *Journal of Mathematical Physics*, vol. 5, no. 9, pp. 1252–1254, 1964. https://doi.org/10.1063/1.1704233.

[5] G. Fodor, "A review on radiation of oscillons and oscillatons," *arXiv: High Energy Physics - Theory*, 2019. https://arxiv.org/abs/1911.03340v1.

[6] M. C. Johnson, H. V. Peiris, and L. Lehner, "Determining the outcome of cosmic bubble collisions in full general relativity," *Phys. Rev. D*, vol. 85, p. 083516, Apr 2012.

[7] N. Graham, "An electroweak oscillon," *Phys. Rev. Lett.*, vol. 98, p. 101801, Mar 2007.

[8] N. Graham, "Numerical simulation of an electroweak oscillon," *Phys. Rev. D*, vol. 76, p. 085017, Oct 2007.

[9] R. Correa, L. Ospedal, W. de Paula, and J. Helayël-Neto, "Supersymmetry and fermionic modes in an oscillon background," *Physics Letters B*, vol. 780, pp. 159–165, 2018.

[10] S. Antusch, F. Cefalà, and S. Orani, "Gravitational waves from oscillons after inflation," *Phys. Rev. Lett.*, vol. 118, p. 011303, Jan 2017.

[11] J. Ollé, O. Pujolàs, and F. Rompineve, "Oscillons and dark matter," *Journal of Cosmology and Astroparticle Physics*, vol. 2020, pp. 006–006, Feb 2020.

[12] P. Salmi, "Oscillons," 2000. https://hdl.handle.net/1887/13117.

[13] H. P. Langtangen, *Finite difference methods for wave motion*. Department of Informatics, University of Oslo, 2016. http://hplgit.github.io/num-methods-for-PDEs/doc/pub/wave/html/wave.html.

A Simulation Figures

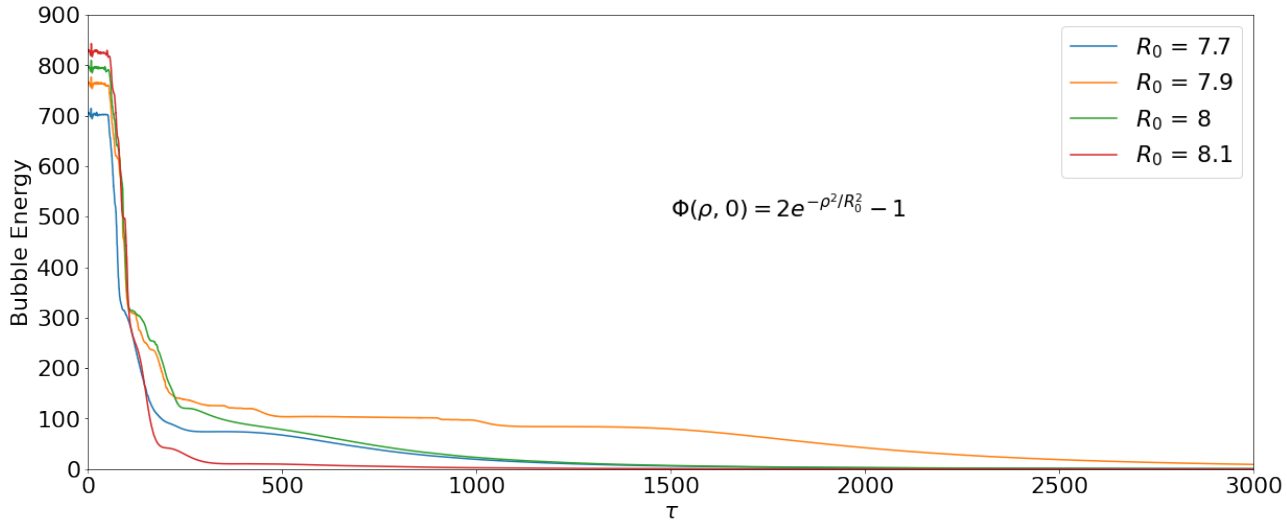

Fig. 10. Energy within a spherical shell surrounding initial configurations of $R_0 \in [7.7, 8.1]$ as a function of time (Method 2)

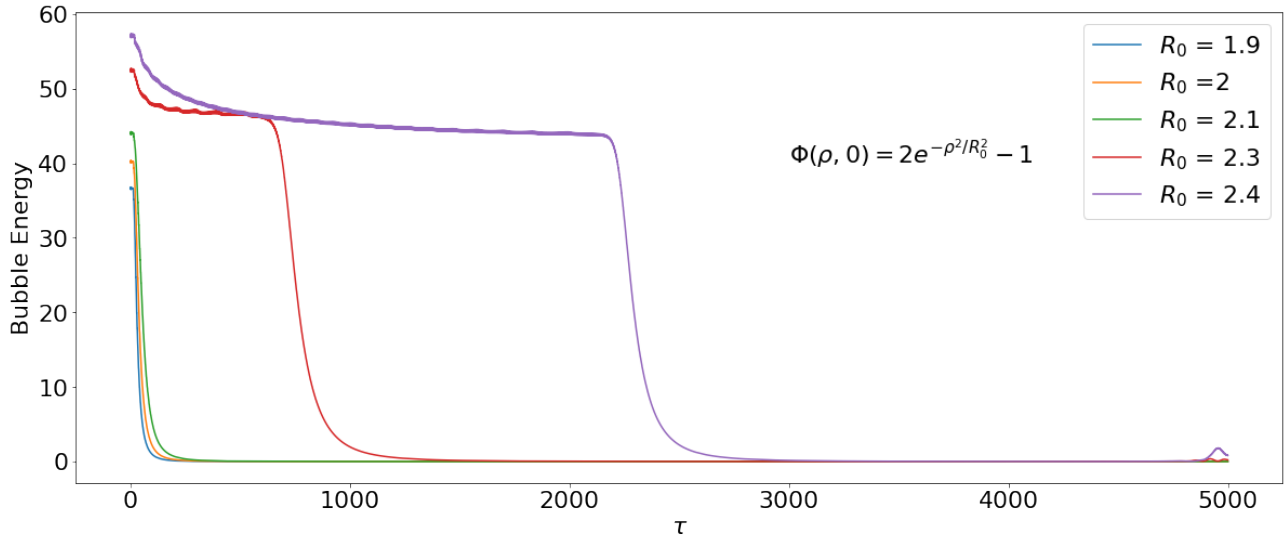

Fig. 11. Energy within a spherical shell surrounding initial configurations of $R_0 \in [1.9, 2.4]$ as a function of time (Method 2)

B Code

Method 1:

```

def phi_at_t_equal_zero(x):
```

```
    return 2*np.exp(-x**2/16) - 1#\rho_0 = 4,


# U is a list of all u_i. Its length is N
# u = [phi, dphi/dt]
# returns [dphi/dt, d^2phi/dt^2]
def dus(u,t):
    # first we provide dphi/dt
    rhs = []
    for i in range(0, N):
        rhs.append(u[i+N])
    # second we provie d^2phi/dt^2
    rhs.append(u[0] - u[0]**3 + (2/dx**2)*(u[1]-u[0]))
    for i in range(1, N-1):
        rhs.append(u[i] - u[i]**3 +(1/(dx**2))*(u[i+1]-2*u[i]+u[i-1]) + (2/x[
   i])*(1/(dx))*(u[i+1] - u[i]))
    rhs.append( (1/dx**2)*(-3 + u[N-2]) + (2/x[N-1])*(1/dx)*(-1-u[N-1]) )
    return rhs

global dx, L, x, N
L = 400
N = 800
N_t = 5000

x = np.linspace(0, L, N)
ts = np.linspace(0, N_t, 1000)
dx = x[1] - x[0]
dt = t[1] - t[0]

# initial conditions
U_0 = []#\rho_0 = 4, rho = 0
for i in range(0, N):
    U_0.append(phi_at_t_equal_zero(x[i]))
for i in range(0, N):
    U_0.append(0)

u = odeint(dus, U_0, ts)

```

Method 1 energy computation:

```
# total energy
epsilon_r_0 = N
E_k = []
for i in range(len(ts)):
```

```
    integral = 0
    for r in range(epsilon_r_0):
        integral += dx*(x[r]**2)*u[i][N+r]**2
    integral = integral*2*np.pi
    E_k.append(integral)

E_s = []
for i in range(len(ts)):
    integral = 0
    for r in range(epsilon_r_0):
        if r == 0 or r == N-1 :
            du_drho = 0
        else:
            du_drho = (u[i][r+1]-u[i][r])/dx
        integral += dx*((x[r] + 0.5*dx)**2)*du_drho**2
    integral = integral*2*np.pi
    E_s.append(integral)

E_v = []
for i in range(len(ts)):
    integral = 0
    for r in range(epsilon_r_0):
        integral += dx*(x[r]**2)*1/4*(u[i][r]**2 - 1)**2
    integral = integral*4*np.pi
    E_v.append(integral)

E = []
for i in range(len(ts)):
    E.append(E_v[i] + E_s[i] + E_k[i])

# bubble energy
epsilon_r_0 = 2*3*4 # rho_0 = 4
E_k = []
epsilon_r_0 = np.where(x >= epsilon_r_0)[0][0]
for i in range(len(ts)):
    integral = 0
    for r in range(epsilon_r_0):
        integral += dx*(x[r]**2)*u[i][N+r]**2
    integral = integral*2*np.pi
    E_k.append(integral)

E_s = []
for i in range(len(ts)):
    integral = 0
```

```
    for r in range(epsilon_r_0):
        if r == 0 or r == N-1 :
            du_drho = 0
        else:
            du_drho = (u[i][r+1]-u[i][r])/dx
        integral += dx*((x[r] + 0.5*dx)**2)*du_drho**2
    integral = integral*2*np.pi
    E_s.append(integral)

E_v = []
for i in range(len(ts)):
    integral = 0
    for r in range(epsilon_r_0):
        integral += dx*(x[r]**2)*1/4*(u[i][r]**2 - 1)**2
    integral = integral*4*np.pi
    E_v.append(integral)

E = []
for i in range(len(ts)):
    E.append(E_v[i] + E_s[i] + E_k[i])

```

Method 2: method 2 was written in Matlab as it proved to compute faster than python.

```
    rho_0 = 4;
    m = 1;
    lmda = 1;

    % simulation variables
    L_x = 2000;
    L_t = 5000;
    N_x = 10000;
    N_t = 50000;

    x = linspace(0, L_x, N_x);
    dx = x(2) - x(1);
    t = linspace(0, L_t, N_t);
    dt = t(2) - t(1);

    phi = zeros(N_x, N_t);

    % Step 1: compute phi(x_i, 0) and phi(x_i, 1) for i = 0,..., N_x
    for i = 1:N_x
        phi(i, 1) =  2*exp(-(x(i)/rho_0)^2)-1;
        phi(i, 2) = phi(i, 1);
```

```
    end

    % boundry condition
    for n =1:N_t
        phi(N_x, n) = -1;
    end

    for n = 2:N_t-1
        for i = 2: N_x-1
            phi(i, n+1) = (dt^2)*(phi(i,n) - phi(i,n)^3 ...
                + (2/x(i))*(phi(i+1,n)-phi(i-1,n))/(2*dx)+ ...
                (phi(i+1,n)-2*phi(i,n)+phi(i-1,n))/dx^2 ) ...
                + 2*phi(i,n)- phi(i,n-1);
            %           if i == 2
            %               phi(1, n+1) = phi(i, n+1);
            %           end
        end
        phi(1, n+1) = phi(2, n+1);
    end
```

Method 2 energy computation:

```
    % case variables
    r_0 = (2*3*rho_0)/dx +1;

    E_k = zeros(1, N_t);
    % n = 0 case
    E_k(1,1) = 0;
    for n = 2:N_t-1
        sum = 0;
        for i = 1:r_0
            sum = sum +  dx*(x(i)^2)*((phi(i, n+1)-phi(i, n-1))/(2*dt))^2;
        end
        sum = sum*2*pi;
        E_k(1, n) = sum;
    end

    % n = N_t case
    sum = 0;
    for i = 1:r_0
        sum = dx*(x(i)^2)*(phi(i,N_t) - phi(i,N_t-1)/dt)^2;
    end
    sum = sum*2*pi;
```

```
    E_k(1, N_t) = sum;


    E_s = zeros(1, N_t);
    for n = 1:N_t
        sum = 0;
        for i = 1:r_0
            if i == 1
                sum = sum +  dx*((x(i))^2)*((phi(i+1,n) - phi(i,n))/(dx))^2;
            elseif i == N_x
                sum = sum +  dx*((x(i))^2)*((phi(i,n)- phi(i-1,n))/(dx))^2;
            else
                sum = sum + dx*((x(i))^2)*((phi(i+1,n) - phi(i-1,n))/(2*dx))
^2;

            end
        end
        sum = sum*2*pi;
        E_s(1,n) = sum;
    end

    E_v = zeros(1, N_t);
    for n = 1:N_t
        sum = 0;
        for i = 1:r_0
            sum = sum+ dx*(x(i)^2)*1/4*(phi(i,n)^2 - 1)^2;
        end
        sum = sum*4*pi;
        E_v(1, n) = sum;
    end

    E = zeros(1, N_t);
    for n = 1:N_t
        E(1, n) = E_v(1, n) + E_s(1, n) + E_k(1,n);
    end
    writematrix(E,[num2str(rho_0),'_','.csv'])
```