Draft 2: Solitons in the KdV equation

Aya Abdelhaq, Shadi Ali Ahmad, Koral Aykin

November 21, 2023

Abstract

A rough outline of our report draft on solitons is presented. We first discuss some analytic properties and solutions of the KdV equation, and then move on to the numerical methods that we will use in our code.

1 Exact solutions

The KdV equation is a partial differential equation that describes the evolution of water waves in terms of the height of the water (u) with respect to space (x) and time (t). The equation is given by:

$$\partial_t u + \epsilon u \partial_x u + \mu \partial_x^3 u = 0, \tag{1}$$

where $u(x,t)$ represents the height of the water, x is the direction of propagation, and t is time. The terms in the equation represent a nonlinear advection term ($\epsilon u \frac{\partial u}{\partial x}$) responsible for wave propagation and a dispersive term ($\mu \frac{\partial^3 u}{\partial x^3}$) responsible for wave spreading. Before trying to find numerical solutions to the above equation, we first consider a simpler but related problem, namely the advection equation for $u(x,t)$ given by

$$\partial_t u + \epsilon \partial_x u = 0, \tag{2}$$

were we have suppressed functional dependence and ϵ is a constant. We do so because it is easier to handle analytically and it allows us to build up intuition for the full equation.

It is straightforward to verify that *any* function of the form

$$u(x,t) = f(x - \epsilon t), \tag{3}$$

is a solution of Eq. (2). Plugging in this ansatz we have

$$\begin{aligned} \partial_t u + \epsilon \partial_x u &= \partial_t f(x-\epsilon t) + \epsilon \partial_x f(x - \epsilon t), \\ &= \frac{\partial f(x-\epsilon t)}{\partial(x-\epsilon t)} \frac{\partial(x-\epsilon t)}{\partial t} + \epsilon \frac{\partial f(x-\epsilon t)}{\partial(x-\epsilon t)} \frac{\partial(x-\epsilon t)}{\partial x}, \\ &= -\epsilon f' + \epsilon f' = 0. \end{aligned} \tag{4}$$

Here, the prime denotes differentiation relative to the variable $\xi := x - \epsilon t$.

The solution denotes an arbitrarily shaped wave-form for the height of the water that is being carried, without changing its shape, by a constant velocity ϵ. Suppose that we turn off dissipation in the Kortweg-de-Vries (KdV) equation (1) by setting $\mu = 0$, then the simplified equation takes a similar form to the advection equation

$$\partial_t u + \epsilon u \partial_x u = 0, \tag{5}$$

with the only difference being the dependence on the shape of u in the second term. So, we expect similar solutions that are carried with velocity ϵ in the non-dissipative regime, with the caveat that the velocity becomes dependent on the height of the wave itself precisely due to the non-linear dependence on u in the second term of the above equation.

Indeed, we may be more quantitative by characterizing these solutions. Let us plug in the ansatz in Eq. (3) in the KdW equation, with $\epsilon \to c$. By similar reasoning to the solution verification of the advection equation, we obtain

$$\begin{aligned}\partial_t u + \epsilon u \partial_x u + \mu \partial_x^3 u &= \partial_t f + \epsilon f \partial_x f + \mu \partial_x^3 u,\\ &= f' \frac{\partial(x-ct)}{\partial t} + \epsilon f f' \frac{\partial(x-ct)}{\partial x} + \mu f''' \frac{\partial(x-ct)}{\partial x},\\ &= -cf' + \epsilon f f' + \mu f''' = 0. \end{aligned}\tag{6}$$

The above equation characterizes the waveforms that are solutions to the KdW equation in the non-dissipative regime, and that are of the form $u = f(x-ct)$ for some constant c.

To get a feel for this equation, we may integrate this once relative to ξ to obtain

$$-cf + \frac{\epsilon}{2} f^2 + \mu f'' = C_1,$$

where C_1 is an integration constant. Rearranging gives

$$\mu f'' = C_1 + cf - \frac{\epsilon}{2} f^2.$$

As physicists, we will attempt to solve the simplest possible case, so we set both $C_1 = 0$, multiply throughout by f', and integrate once more relative to ξ to get

$$\frac{1}{2}(f')^2 = \frac{c}{2\mu} f^2 - \frac{\epsilon}{6\mu} f^3 = \frac{f^2}{\mu}\left(c - \frac{\epsilon}{3} f\right).$$

Taking the square root and integrating on both sides, we obtain

$$\int \frac{\sqrt{\mu} df}{f\sqrt{c - \frac{\epsilon}{3} f}} = \int d\xi.$$

We perform the integral on the left via a substitution

$$f = \frac{3c}{\epsilon} \operatorname{sech}^2(z),$$

under which the integration measure becomes

$$df = -\frac{6c}{\epsilon} \frac{\sinh(z)}{\cosh(z)^3} dz.$$

The square root in the denominator simply evaluates to a $\tanh z$, and there is a massive simplification yielding

$$-\frac{2}{\sqrt{c}} \sqrt{\mu} z = \xi - \xi_0,$$

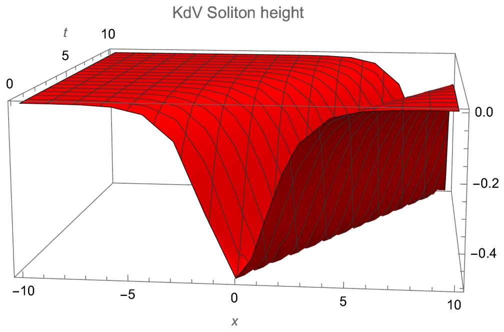
where ξ_0 is an integration constant. Solving for z and plugging into the expression for f, we obtain the general solution to the KdW equation as

$$f = \frac{3c}{\epsilon}\text{sech}^2\left[\frac{\sqrt{c\mu}}{2}(\xi - \xi_0)\right].$$

Plugging this back into the equation, we find that this will be a solution for any c, ϵ as long as $\mu = \pm 1$. For $\mu = 1$ and $\epsilon = -6$, we obtain

$$f = -\frac{c}{2}\text{sech}^2\left[\frac{\sqrt{c}}{2}(\xi - \xi_0)\right]. \tag{7}$$

We plot this in Fig. (1) for $c = 1$ and $\xi_0 = 0$.



2 Computational methods

The chosen computational approach is the finite difference technique; which is a numerical technique for approximating solutions to differential equations by discretizing the spatial and temporal domains. It involves replacing derivatives with discrete difference approximations, leading to a set of algebraic equations that can be solved iteratively. Consider a general partial differential equation (PDE) in one dimension:

$$F(u, u_x, u_{xx}, \ldots, u_t, u_{xt}, \ldots, x, t) = 0$$

Here, u is the dependent variable, x is the spatial coordinate, and t is time. The goal is to solve for $u(x, t)$.

2.1 Finite difference approach

Now, let's break down the finite difference approach:

- **Spatial Discretization:** Divide the spatial domain (x) into N grid points with a uniform spatial step size (Δx). Discrete spatial points: $x_i = i \cdot \Delta x$ for $i = 0, 1, \ldots, N$.
- **Temporal Discretization:** Discretize time (t) into M time steps with a uniform time step size (Δt). Discrete time points: $t_j = j \cdot \Delta t$ for $j = 0, 1, \ldots, M$.

- **Finite Difference Approximations** Express derivatives using central differences or other finite difference approximations. For example, the first derivative in space (u_x) is approximated as $\frac{u_{i+1}-u_{i-1}}{2\Delta x}$.

Now, substitute the approximations into the PDE to get a difference equation:

$$F(u_{i,j}, u_{i+1,j}, u_{i-1,j}, u_{i,j+1}, u_{i,j-1}, \Delta x, \Delta t) = 0$$

Solve this difference equation iteratively to update the solution over time and space. To discretize the KdV equation, the code employs first-order central difference approximations for the partial derivatives with respect to time ($\frac{\partial u}{\partial t}$) and space ($\frac{\partial u}{\partial x}$) which are give by:

$$\frac{\partial u}{\partial t} \approx \frac{u_{i,j+1} - u_{i,j}}{\Delta t}$$

$$\frac{\partial u}{\partial x} \approx \frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x}$$

This involves breaking down the continuous problem into discrete steps in time (j) and space (i). The code further expands $u(x,t)$ using a Taylor series in x around a given point x_i, considering neighboring points to formulate a set of equations. The heart of the simulation lies in the evolution scheme, where the values of u at different spatial points and times are interconnected. The evolution is dictated by the KdV equation, with a key formula

$$u(x_i, t_j) = \frac{1}{3}(u_{i-1,j} + u_{i,j} + u_{i+1,j})$$

consolidating the contributions from neighboring points. The soliton(u) function takes the initialized wave profile and evolves it over time using a finite difference scheme. It iterates over spatial points and time steps, updating the wave profile based on the KdV equation. The key computations involve solving a set of equations derived from a Taylor series expansion of the wave function.

2.1.1 Finite Difference Equations

In this section, we will derive finite difference equations for Eq. (1). Our goal is to determine the height of the soliton at points within the specified region for future times, given an initial condition. For that purpose, we start with Taylor expanding $u(x \pm \Delta x, t)$ and $u(x \pm 2\Delta x, t)$ up to third order in spatial variable x

$$u(x \pm \Delta x, t) = u(x,t) \pm \Delta x \frac{\partial u}{\partial x} + \frac{(\Delta x)^2}{2}\frac{\partial^2 u}{\partial x^2} \pm \frac{(\Delta x)^3}{6}\frac{\partial^3 u}{\partial x^3} \tag{8}$$

The Taylor expansion of $u(x \pm 2\Delta x, t)$ can be found by replacing Δx in the equation above with $2\Delta x$. After algebraic manipulation of those expansions, we can express the third order derivative of u as follows

$$\frac{\partial^3 u}{\partial x^3} = \frac{[u(x+2\Delta x, t) - u(x-2\Delta x, t)] - 2[u(x+\Delta x, t) - u(x-\Delta x, t)]}{2(\Delta x)^3} \tag{9}$$

By the same token, to approximate the first order derivative, we can use Taylor expansion up to first order and apply the same steps; resulting in

$$\frac{\partial u}{\partial x} = \frac{u(x+\Delta x, t) - u(x-\Delta x, t)}{2\Delta x} \tag{10}$$

The first order derivative of u with respect to t can be expressed as finite difference equation in the same manner

$$\frac{\partial u}{\partial t} = \frac{u(x,t+\Delta t) - u(x,t-\Delta t)}{2\Delta t} \tag{11}$$

Before proceeding further, it is reasonable to change our notation in a more compatible way with Python

$$\begin{aligned}\frac{\partial^3 u}{\partial x^3} &= \frac{u_{i+2,j} + 2u_{i-1,j} - 2u_{i+1,j} - u_{i-2,j}}{2(\Delta x)^3} \\ \frac{\partial u}{\partial x} &= \frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x} \\ \frac{\partial u}{\partial t} &= \frac{u_{i,j+1} - u_{i,j-1}}{2\Delta t}\end{aligned}$$

In this notation, Δx and Δt are discretization constants in space and time, respectively. We will return to how those constants need to be chosen in a way that eliminates instabilities.

Now, we combine the above results into an evolution scheme for Eq. (1). Notice that we do not have an finite difference expression for $u(x,t)$. It is given in the problem statement that $u(x,t)$ can be approximated as follows

$$u = \frac{u_{i+1,j} + u_{i,j} + u_{i-1,j}}{3} \tag{12}$$

Substituting the finite difference approximations into Eq. (1) results in

$$\begin{aligned}u_{i,j+1} = u_{i,j-1} &- \frac{\epsilon \Delta t}{3\Delta x}(u_{i+1,j} + u_{i,j} + u_{i-1,j})(u_{i+1,j} - u_{i-1,j}) \\ &- \frac{\mu \Delta t}{(\Delta x)^3}(u_{i+2,j} + 2u_{i-1,j} - 2u_{i+1,j} - u_{i-2,j})\end{aligned} \tag{13}$$

which gives the next time step u_{j+1} in terms of current and past time steps, u_j and u_{j-1} respectively.

It can be immediately noticed that to initiate the evolution, one needs to account for the time step before the initial time. This issue can remedied by replacing the central difference equation for the first derivative in time with that of forward difference, which is given by

$$\frac{\partial u}{\partial t} = \frac{u_{i,j+1} - u_{i,j}}{\Delta t} \tag{14}$$

and resulting in an evolution scheme

$$\begin{aligned}u_{i,j+1} = u_{i,j} &- \frac{\epsilon \Delta t}{6\Delta x}(u_{i+1,j} + u_{i,j} + u_{i-1,j})(u_{i+1,j} - u_{i-1,j}) \\ &- \frac{\mu \Delta t}{2(\Delta x)^3}(u_{i+2,j} + 2u_{i-1,j} - 2u_{i+1,j} - u_{i-2,j})\end{aligned} \tag{15}$$

which gives the next time step u_{j+1} in terms of only current time steps, u_j. Note that we use forward difference approximation only for initialization.

Another thing we need to pay attention before implementing the above results in Python is that those equations do not handle the boundary conditions. For example, to calculate $u_{N,2}$ and $u_{1,2}$, we need to know $u_{N+2,1}$ and $u_{-1,1}$. Instead, in the implementation, we assume fixed values of both endpoints and also zero derivative which asserts that $u_{N+1,1}$ is equal to $u_{N,1}$ according to our finite difference scheme. To illustrate how the boundary conditions are handled this way, we can consider the previous example. Now, we do not need to calculate $u_{N,2}$ and $u_{1,2}$ anymore since the both endpoints have fixed values, they are determined for all future times with the initial condition imposed. On the other hand, we still have to calculate $u_{N-1,2}$ and $u_{2,2}$ for which the values $u_{N+1,2}$ and $u_{0,2}$ are required prior to. Those values are obtained from the zero derivative condition which specifies the points one spatial step further from the endpoints.

2.1.2 Forward difference method interlude

2.2 Linear stability

Any discretization method used to solve a wave equation runs into a very natural problem. Both space and time are discretized into smallest indivisible units, Δt and Δx, and so there is a characteristic speed of propagation imposed by this discretization set by the velocity $v_d := \frac{\Delta x}{\Delta t}$. If we are interested in finding the shape of a traveling wave with velocity ξ, then the time step for the discretization cannot be bigger than the time it takes the wave to physically traverse the spatial distance between the two points on the grid. Thus, we have to make sure we upper bound the following quantity

$$\beta := \frac{\xi}{v_d} \leq \beta_m, \tag{16}$$

for some β_m which we will take to be as small as we can to ensure stability of the simulation.