

Solitons and the KdV Equation

Arshia Akhtarkavan

PHYS305 - Computational Physics

December 2025

Overview

- 1 Motivation & Problem Setup
- 2 Deriving the ODE and Setting up the Numerical Integration
- 3 Numerical Results
- 4 Traveling-Wave Solution $\phi(x, t)$
- 5 Convergence and Self-Convergence

What is a Soliton?

A soliton is a nonlinear wave with [1]:

- permanent shape,
- spatial localization,
- elastic collisions (only a phase shift).

Meaning that it is a wave packet that keeps its shape, speed, and size as it travels.

One popular solution for solitons is given by the Korteweg-de Vries (KdV) equation:

$$\phi_t - 6\phi\phi_x + \phi_{xxx} = 0. \quad (1)$$

$$\frac{\partial \phi}{\partial t} - 6\phi \frac{\partial \phi}{\partial x} + \frac{\partial^3 \phi}{\partial x^3} = 0 \quad (2)$$

Traveling-Wave Ansatz and the reduction to an ODE

We can start by looking for solutions of the form:

$$\phi(x, t) = f(x - ct) = f(X), \quad X = x - ct. \quad (3)$$

Then:

$$\frac{\partial X}{\partial x} = 1, \quad \frac{\partial X}{\partial t} = -c. \quad (4)$$

Finally by using the chain rule:

$$\phi_x = f'(X), \quad \phi_t = -c f'(X), \quad \phi_{xxx} = f'''(X). \quad (5)$$

Plugging this equation into the KdV equation:

$$\phi_t - 6\phi \phi_x + \phi_{xxx} = 0, \implies -cf'(X) - 6f(X)f'(X) + f'''(X) = 0. \quad (6)$$

Thus we get the third order ODE:

$$f'''(X) = (c + 6f(X))f'(X). \quad (7)$$

Constant of Motion

Notice that by integrating the ODE, we get

$$f''(X) = cf(X) + 3f(X)^2 + C, \quad (8)$$

which can be rewritten as the constant of motion:

$$C = f''(X) - (c + 3f(X))f(X). \quad (9)$$

Using the initial conditions that we are assuming, alongside the simplification that $c = 1$ throughout this problem, we get

$$f(0) = -\frac{1}{2}, \quad f'(0) = 0, \quad f''(0) = \frac{1}{4}, \implies C = 0 \quad (10)$$

This is great, as the constant of motion can be used to determine the accuracy of our numerical solutions later on.

Reduction to a First-Order System

To numerically integrate this, we can rewrite this third-order ODE as a system of three first order ODEs: Define:

$$u_1 = f, \quad u_2 = f', \quad u_3 = f''. \quad (11)$$

Then the system becomes:

$$\begin{cases} u'_1 = u_2, \rightarrow u_1(0) = -\frac{1}{2}, \\ u'_2 = u_3, \rightarrow u_2(0) = 0, \\ u'_3 = (1 + 6u_1) u_2. \rightarrow u_3(0) = \frac{1}{4}. \end{cases} \quad (12)$$

Now we can use the 4th Order Runge-Kutta(RK4) method to numerically integrate this system over a given interval.

Fourth-Order Runge–Kutta (RK4) Method

We have the system

$$\vec{u}'(X) = \begin{bmatrix} u_2 \\ u_3 \\ (1 + 6u_1)u_2 \end{bmatrix}, \quad \vec{u}(0) = \begin{bmatrix} -\frac{1}{2} \\ 0 \\ \frac{1}{4} \end{bmatrix}, \quad (13)$$

we use RK4 with step size h for a system of ODEs, where we have:

$$\vec{k}_1 = \vec{F}(X_n, \vec{u}_n), \quad \vec{k}_2 = \vec{F}\left(X_n + \frac{h}{2}, \vec{u}_n + \frac{h}{2}\vec{k}_1\right), \quad (14)$$

$$\vec{k}_3 = \vec{F}\left(X_n + \frac{h}{2}, \vec{u}_n + \frac{h}{2}\vec{k}_2\right), \quad \vec{k}_4 = \vec{F}\left(X_n + h, \vec{u}_n + h\vec{k}_3\right). \quad (15)$$

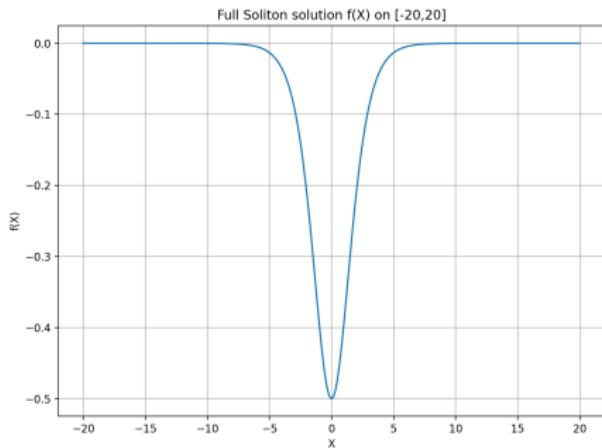
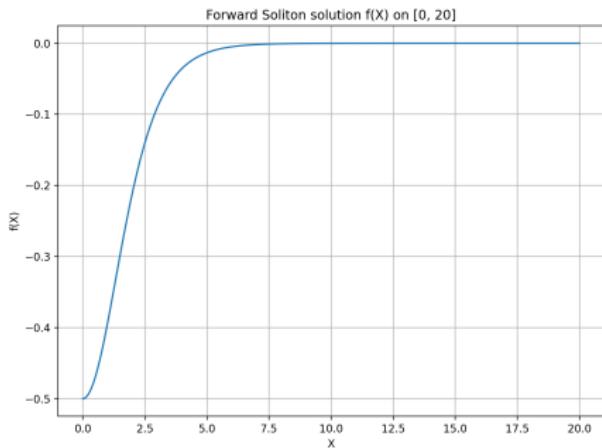
Where we can get \vec{u}_{n+1} by:

$$\vec{u}_{n+1} = \vec{u}_n + \frac{h}{6} \left(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4 \right). \quad (16)$$

RK4 has global error $\mathcal{O}(h^4)$, which we can verify through the convergence test later on.

Numerical Integration of the ODE

Using the described RK4 scheme, we can integrate the system on the interval $X \in [-20, 20]$ with the previously mentioned initial conditions. Observe that the ODE is symmetric, meaning $f(X) = f(-X)$, so we only have to compute $X > 0$ and then reflect the results to get the full interval of solutions.



Convergence Test: Constant of Motion

To test the accuracy of these numerical solutions, use the constant of motion $C(X)$:

$$C(X) = u_3 - (1 + 3u_1) u_1. \quad (17)$$

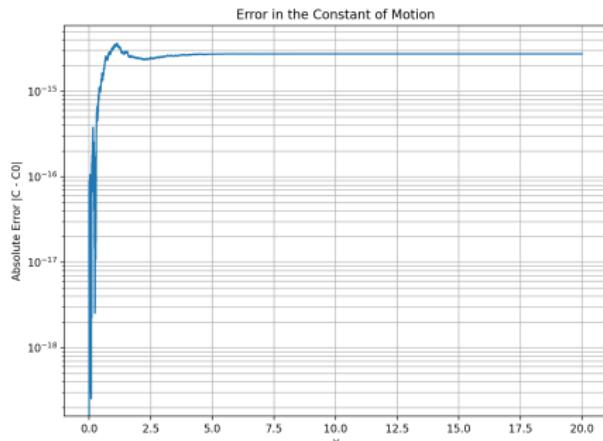
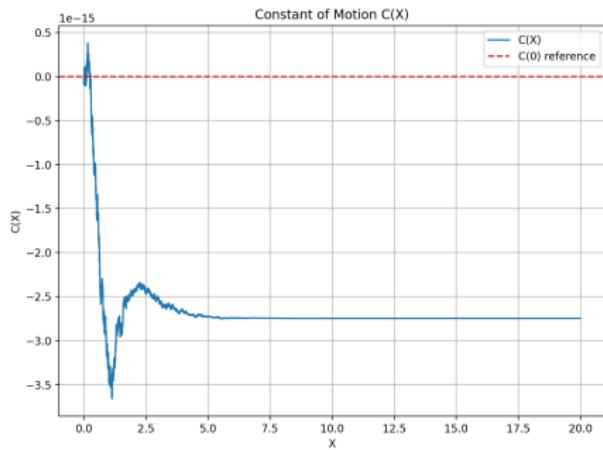
where analytically we know that $C(X) = 0 \quad \forall X$. Calculating the absolute error (Since $C(0) = 0$, we can't do relative error since we'd get a division by 0):

$$\delta C = |C(5) - C(0)| = |C(5)|, \quad (18)$$

for decreasing step sizes h .

$C(X)$ and error in the Constant of Motion

We can create the plots:



So overall, pretty good!

Creating the Traveling Wave

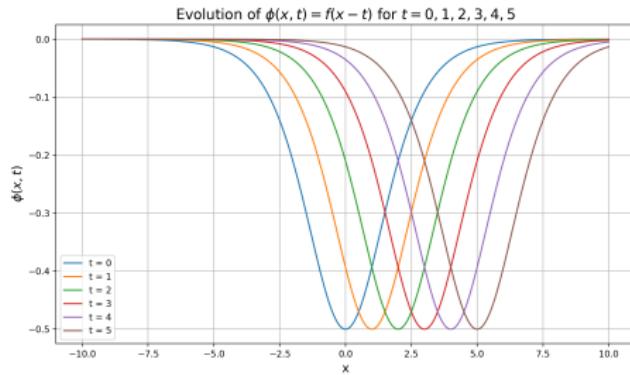
Recall that

$$\phi(x, t) = f(x - t) \quad (19)$$

Using the numerical solution $f(X)$ and a third-order Lagrange interpolation, we evaluate

$$\phi(x, t_k) = f(x - t_k) \quad (20)$$

for $x \in [-10, 10]$ and $t_k = 0, 1, 2, 3, 4, 5$. This is exactly what a soliton should look like as it travels to the right ($+x$ direction)!



Convergence Test: Constant of Motion

To test the order of accuracy, we compute $C(X)$ at $X = 5$ for different step sizes h :

$$C(5; h) = u_3(5; h) - (1 + 3u_1(5; h)) u_1(5; h). \quad (21)$$

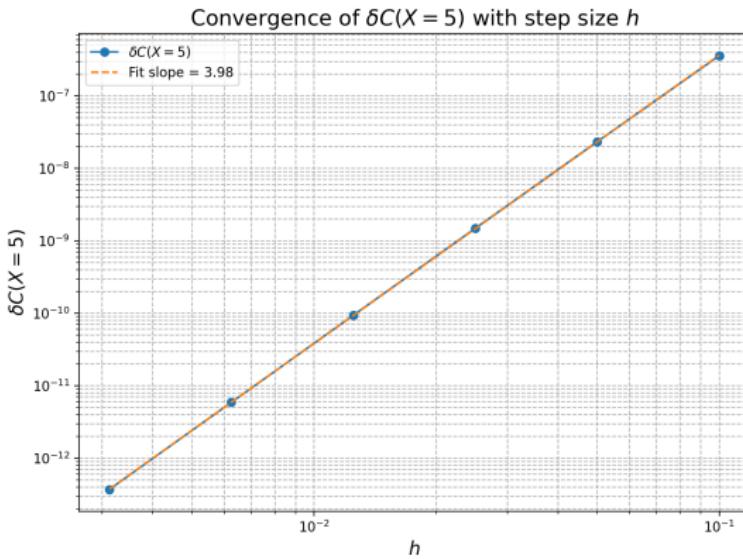
Since analytically $C(X) = C(5) = 0$, we can define the absolute error as

$$\delta C(h) = |C(5; h)|. \quad (22)$$

And so we can see how δC decreases as $h \rightarrow 0$.

Convergence of $\delta C(X = 5)$

Drawing a log-log plot:



Where δC approximately is $\propto h^4$ from the slope of the fitted line.
Matches the expected error order of RK4.

Self Convergence of $f(5)$

Since in this case the exact solution is unknown, we can test self convergence. Let $f(h)$ be the numerical value of $f(5)$ at step size h , and let h_2 and h_3 be the two finest resolutions (N is the largest for these cases). Then similar to how it was done in the last homework, we form the ratio

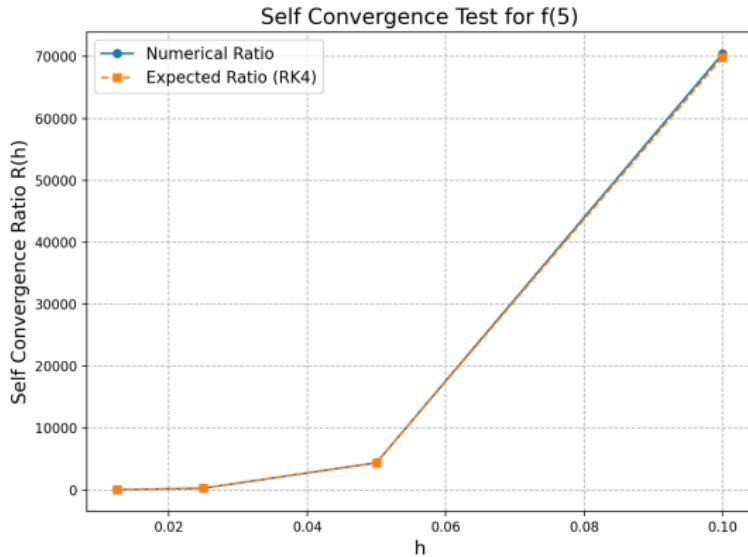
$$R_{\text{num}}(h) = \frac{f(h) - f(h_3)}{f(h_2) - f(h_3)} \quad (23)$$

And

$$R_{\text{expected}}(h) = \frac{(h/h_3)^n - 1}{2^n - 1}. \quad (24)$$

Where for the RK4 method, n is 4.

Self Convergence of $f(5)$ - Continued



The numerical ratio $R_{\text{num}}(h)$ and the theoretical ratio $R_{\text{expected}}(h)$ are very close here!

Richardson Extrapolation for $f(5)$

Finally by using the Richardson extrapolation, we can determine the error at any time (say $f(5)$). So

$$f_{h_2} \text{ for } N = 800, \quad f_{h_3} \text{ for } N = 1600. \quad (25)$$

The 4th-order Richardson estimate is

$$f_R = \frac{2^4 f_{h_3} - f_{h_2}}{2^4 - 1}. \quad (26)$$

Numerically:

$$f_{h_2} \approx -1.3296113376 \times 10^{-2}, \quad (27)$$

$$f_{h_3} \approx -1.3296113344 \times 10^{-2}, \quad (28)$$

$$f_R \approx -1.3296113342 \times 10^{-2}. \quad (29)$$

The estimated error in the finest-resolution value is $|f_R - f_{h_3}| \sim 10^{-12}$.

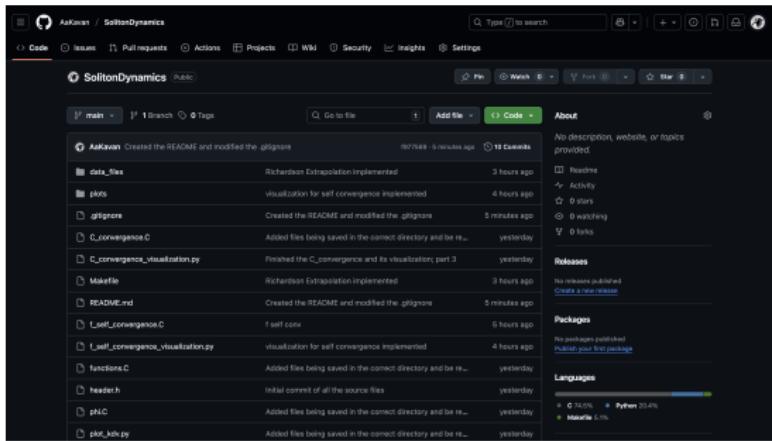
Conclusions

- Successfully solved the KdV ODE using the RK4 method
- Recreated the traveling wave solution for $t = 0, 1, 2, 3, 4, 5$
- Convergence test of the constant of motion showed $\delta C \propto h^4$, which matches the expected RK4 accuracy.
- Self-convergence test for $f(5)$ also agreed with this 4-th order convergence
- Found the error to the solutions for $f(X = 5)$ using Richardson extrapolation; the solutions were very accurate!

Conclusions - continued

The code worked extremely accurately!

All of the code for this final project is publicly available on GitHub at
<https://github.com/AaKavan/SolitonDynamics>



The screenshot shows the GitHub repository page for 'SolitonDynamics'. The repository is public and has 1 branch and 1 tag. The commit history is listed below:

- AnKavan Created the README and modified the gllignore 10/7/2023 5 minutes ago
- data_file Richardson Extrapolation implemented 3 hours ago
- piks visualization for self convergence implemented 4 hours ago
- gllignore Created the README and modified the gllignore 5 minutes ago
- C_convergence.C Added files being saved in the correct directory and be re... yesterday
- C_convergence_visualization.py Finished the C_convergence and its visualization; part 3 yesterday
- Makefile Richardson Extrapolation implemented 3 hours ago
- README.md Created the README and modified the gllignore 5 minutes ago
- L_self_Convergence.C f self conv 6 hours ago
- L_self_Convergence_visualization.py visualization for self convergence implemented 4 hours ago
- Functions.C Added files being saved in the correct directory and be re... yesterday
- header.h Initial comment of all the source-files yesterday
- phi.C Added files being saved in the correct directory and be re... yesterday
- phiK_Mat.h Add files being saved in the correct directory and be re... yesterday

On the right side of the repository page, there are sections for About, Releases, Packages, and Languages. The 'About' section notes 'No description, website, or topics provided.' The 'Languages' section shows Python 20.4% and Matlab 1.6%.

References



Philip G Drazin and Robin Stanley Johnson.
Solitons: an introduction, volume 2.
Cambridge university press, 1989.