Solution to Korteweg-de Vries Equation

Ian William Hoppock

April 14, 2017

1 Code Documentation

The code is written in C, parallelised using OpenMP, and compiled with Autotools using Make. The file `korgi.c` is home to the `main` function, which initialises the program. Here, the user may choose different boundaries, step sizes and initial conditions.

The `main` function calls `RK4`, saved in `RKfour.c`, which initialises the Runge-Kutta algorithm. This in turn calls `rhs` to solve at each time step the right hand side of the equation, i.e., the spatial derivatives derived in Section 3.1. This function is saved in `spatial_finite_difference.c`. Other macros that are used throughout are saved in `vector.c` as well as `vector_add.c`, all with the header `linear_algebra.h`.

To compile, the user should enter into the shell (1) `autoreconf -i` (2) `./configure` (3) `make` (4) `./korgi`. If the user wishes to make use of the OpenMP parallel code, (4) should become `OMP_NUM_THREADS=<n> ./korgi`, where `<n>` should be replaced with the number of threads desired (sans the guillemet).

2 General Remarks

The Korteweg-de Vries (KdV) equation is named for Dutch mathematicians Diederik Korteweg (*1848; †1941) and Gustav de Vries (*1866; †1934). They are credited with rediscovering it in 1895. The earliest known discovery of the phenomenon that the equation defines was in 1834 with experiments by Scottish mathematician and engineer, educated at St Andrews, John Scott Russell FRSE FRS (*1808; †1882). He is the first researcher to describe a *soliton*, or a non-dispersive stable wave, first seen in the Union Canal running from Falkirk to Edinburgh. He wrote on the topic:

> 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 [the boat] had put in motion; [the mass of water] 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.

He was seeing the phenomenon shown in Figure 1. A vessel travelling in shallow water will produce solitons once it is moving at the maximum speed of a wave in that water $v = \sqrt{gh}$, where g is gravitational acceleration and h is the depth of the water. It is for this reason that tsunami waves travel exceptionally quickly across oceans, but John Scott Russell managed to keep up with the soliton on horseback.

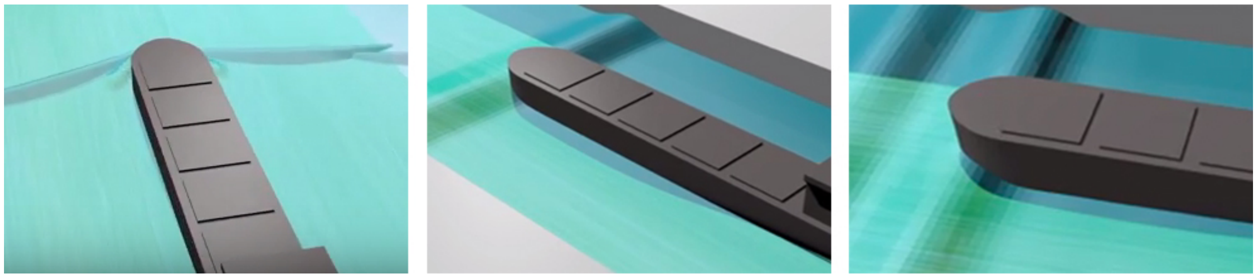

Figure 1: Left: A boat in shallow open water can travel like an ordinary boat, until it approaches the maximum speed of waves in water of that specific depth. At this point, the boat's wake will shift away from the boat and into a more forward propagation. Middle: Upon reaching a wave's maximum speed, this forward propagation, if contained in a channel, turns supercritical and moves ahead of the boat. This is a soliton. Right: The boat's wake will continue to produce solitons until it stops (cf. *Froude depth*).

It is precisely these solitons that the solution of KdV describes. These experiments gave way to the theoretical work of the English mathematician John William Strutt, Third Baron Rayleigh, The Lord Rayleigh OM PRS (*1842; †1919) and French mathematician Joseph Boussinesq (*1842; †1929). The equation found its way into a footnote in a paper by Boussinesq in 1877.

At last, in 1965 and 1967, the solution was found numerically, by Zabusky and Kruskal, and analytically, by Gardner, Greene, Kruskal, and Miura, respectively.

KdV models shallow waves and is famous due to the fact that it is a non-linear partial differential equation with an exact solution. It is written as

$$u_t - 6uu_x + u_{xxx} = 0,$$

where u is the wave amplitude function, and the coefficient 6 is traditional. It is obvious that it is not only non-linear by the middle term on the left hand side but also of order three by the third term on the left hand side. Taking each term on the left hand side in turn: the first is the vertical velocity of the wave at point (x, t); the second is the rate of change in the amplitude; and, the third represents dispersion.

The solitary wave solution of KdV, i.e., the *soliton*, conserves mass and energy because they retain their original shape during interactions with other waves with not but a phase shift.

3 Numerical Approximation

KdV is solved numerically by first discretising spatial terms and using finite differences. This reduces the problem down to a temporal ordinary differential equation. With appropriate (read: periodic) boundary conditions, the fourth order Runge-Kutta solver is employed. Lastly, initial conditions are considered.

3.1 Spatial Finite Difference

Let us rewrite KdV and group the derivative types

$$u_t = 6uu_x - u_{xxx}.$$

Using a uniform spatial discretisation $x_n = ndx$, we can approximate our spatial derivatives at the n^{th} step as

$$\begin{aligned}
\frac{\partial u_n}{\partial x} &\approx \frac{u_{n+1} - u_{n-1}}{2dx} \\
\frac{\partial^2 u_n}{\partial x^2} &\approx \frac{u_{n+1} + u_{n-1} - 2u_n}{dx^2} \\
\frac{\partial^3 u_n}{\partial x^3} &\approx \frac{u_{n+2} - 2u_{n+1} + 2u_{n-1} - u_n}{dx^3}.
\end{aligned}$$

Substituting these into KdV,

$$\frac{\partial u_n}{\partial t} = \frac{6u_n(u_{n+1} - u_{n-1})}{2dx} - \frac{u_{n+2} - 2u_{n+1} + 2u_{n-1} - u_n}{dx^3}.$$

3.2 Periodic Boundary Conditions

The one-dimensional equation is simulated with uniform spatial discretisation $dx = (x_f - x_0)/J$, where J is the user inputted number of points. If the user inputs a faster wave, he or she will want a larger domain and will need to increase x_f and J appropriately. As we do not have a defined function outside our domain, we employ periodic boundary conditions at the endpoints. These points are not defined on the interval, yet are used in the Runge-Kutta algorithm. From the equation in Section 3.1, we are clearly relying on out of bounds data, i.e., non-existent data. Thus, these *ghost points* are specifically defined in the `spatial_finite_difference.c` file. They are defined as follows:

$$\begin{aligned}
u_{J+2} &= u_2 \\
u_{J+1} &= u_1 \\
u_J &= u_0 \\
u_{J-1} &= u_{-1}.
\end{aligned}$$

Thus, it is clear these numbers are given concrete values based on the iterations, hence the periodicity of the boundary.

3.3 RK4

The fourth order Runge-Kutta method is the classic ordinary differential equation solver, and we employ it here to solve KdV. It discretises the temporal space to approximate the solution using four different intermediate values, resulting in an error of $O(dt^5)$. The algorithm approximates the value of the next time step of a differential equation using four intermediate approximate values. These values are, with time step dt,

$$\begin{aligned}
k_1 &= dt * f(x_n, t_n) \\
k_2 &= dt * f(x_n + k_1/2, t_n + dt/2) \\
k_3 &= dt * f(x_n + k_2/2, t_n + dt/2) \\
k_4 &= dt * f(x_n + k_3, t_n + dt).
\end{aligned}$$

It should be noted that there are many correct ways to write this algorithm; the authors prefer considering it with the dt term outside. The above combine to form the next position

$$x_{n+1} = x_n + (k_1 + 2k_2 + 2k_3 + k4)/6.$$

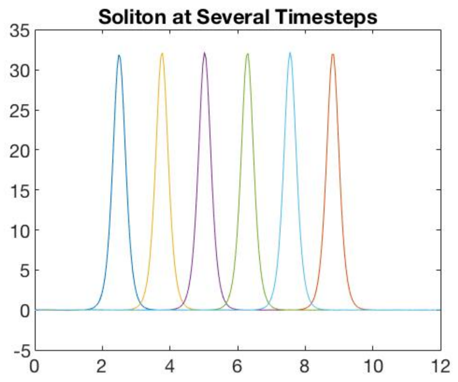

Figure 2: The position of one soliton at many time steps and superimposed. The position is plotted against the amplitude. Notice the same amplitudes and width of the soliton at the different time steps.

Briefly, k_1 extends the slope at the initial point to collect k_2, the slope at the midpoint. We then return to initial point and use k_2 to do a different approximation of slope at the midpoint, k_3. Finally, k_3 is extended all the way to the end position and collects the slope there, k_4. These are then given a pseudo-weighted-average with the midpoints given greater weight. Upon completion the x_{n+1} term becomes the x_n term and it may continue *ad nauseam*.

3.4 Initial Conditions

The exact soliton solution is given by

$$u(x,t) = A\,\mathrm{sech}^2(\gamma(x - vt)),$$

This comes from the canonical (read: contrived) initial condition for KdV,

$$u(x,0) = A\,\mathrm{sech}^2\,\gamma x,$$

where v is velocity, A is the amplitude, and γ is the wave number. These three terms are related via

$$v = 2A = 4\gamma^2.$$

In our code, we included a phase shift so as to make our calculations on the positive x-axis.

4 Results

We have coded a working fourth order Runge-Kutta algorithm for temporal derivatives and finite differences for spatial derivatives. We demonstrate that it indeed works in a variety of figures as well as a movie on the github repository called `two_solitons.avi`. In all simulations, 1,000 spatial points and 10,000 time steps were used.

Figure 2 shows one soliton, superimposed at several time steps. It is clear that there is no numerical addition or dissipation of energy due to the wave having the same amplitude and width at the different positions in time and space. This is true of every step in the simulation.

Figure 3 and its associated animation `two_solitons.avi` show that which is true of solitons and what was discussed in Section 3.4: Faster waves have a greater amplitude and wave number. Here, we have

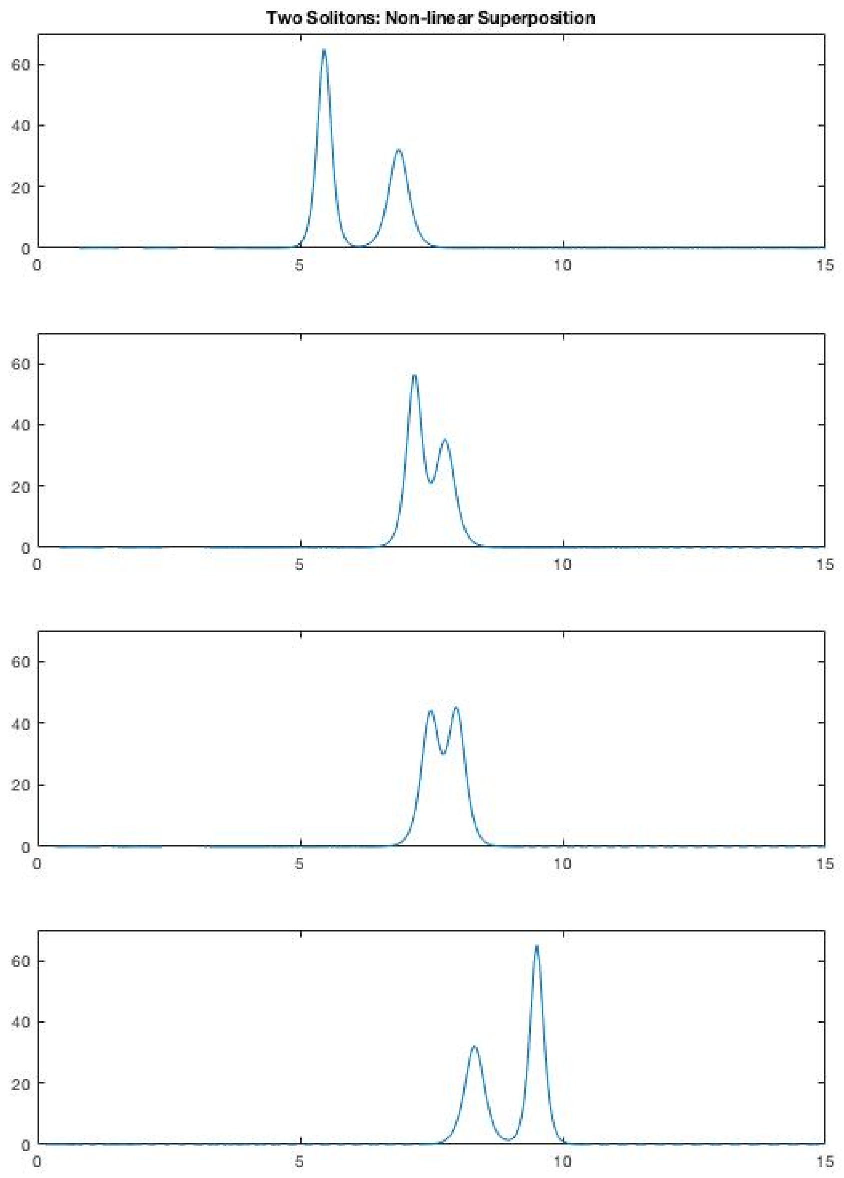

Figure 3: (Position vs. amplitude). The four panels show a two soliton system at various time steps. Superimposing would have been far too confusing; however, a solution to this confusion is represented in Figures 4 and 5. These are stills of the video in the github repository indeed show the greater amplitude wave moves faster and overtakes the smaller wave in a remarkable physical phenomenon of non-linear superposition.

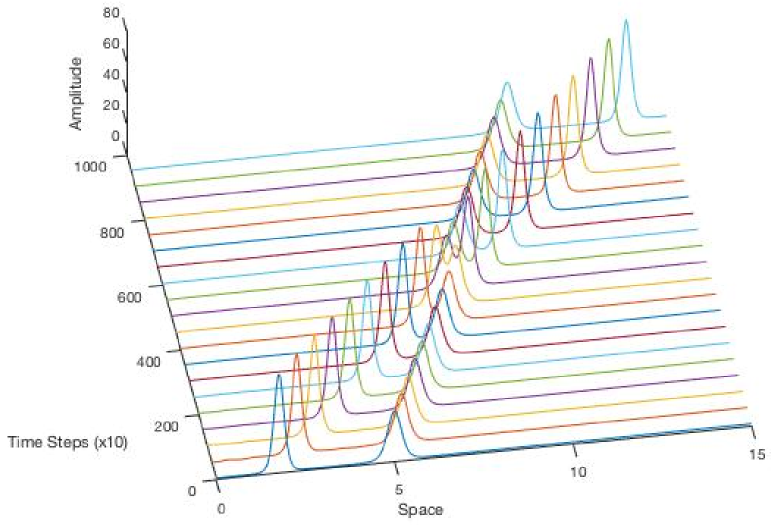

Figure 4: The standard graph plotted at many time steps. The phase shift is noticeable here. Indeed, if one is unable to see it, place a straight edge along the peaks of the shorter wave amplitude.

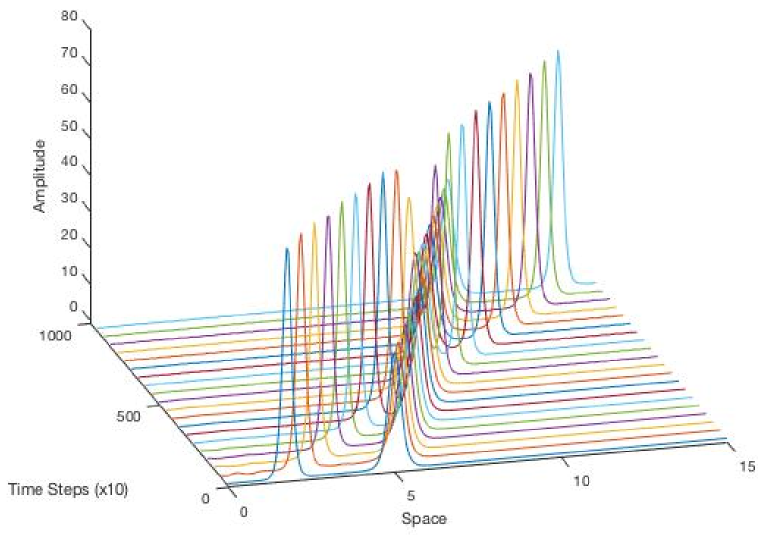

Figure 5: Another angle of the same picture above where the phase shift is even more present. One per five hundred time steps are shown.

simulated two waves, one with doubly velocity, in `korgi.c`. The initial condition is of the form

$$u(x,0) = 2A\,\mathrm{sech}^2(2\gamma x - 10) + A\,\mathrm{sech}^2(\gamma x - 20).$$

We see the remarkable simulated superposition of two solitons. Instead of experiencing linear superposition, they undergo dramatic swapping and preservation of energy, mass, amplitude, wave number. The only noticeable consequence is a reverse phase shift of the smaller wave. This can be seen by scrutinising Figures 4 and 5, which show the use of time as in plotting a thee-dimensional plot. I.e., amplitude is the z-axis, position is the x-axis, and time is the y-axis. Furthermore, we see the superposition is not additive, but, instead, we see the amplitude of the larger wave decrease during this event. The reader is encouraged to consider the video.