2.7 Soliton Solutions of the KdV Equation

This project is concerned with the KdV equation, namely

$$u_t + uu_x + \delta^2 u_{xxx} = 0 \tag{1}$$

We are given that this equation possesses a single soliton solution $f(x-ct)$ where

$$f(x) = A\text{sech}^2\left(\frac{x-x_0}{\Delta}\right),\ \Delta^2 = \frac{12\delta^2}{A},\ c = \frac{A}{3} \tag{2}$$

Further it is assumed that $u(x,t)$ obeys the cyclic boundary condition, ie

$$u(x+1,t) = u(x,t) \tag{3}$$

so that only the region $0 \le x \le 1$ needs to be considered.

Question 1

First we intend to verify that the single soliton solution (2) is indeed a [non-periodic] solution of the KdV equation (1)

Theorem 1. *The single soliton solution (2) is indeed a [non-periodic] solution of the KdV equation (1)*

Proof. Let $u(x,t) = f(x-ct)$. Thus,

$$u_t = -cf'$$
$$u_x = f'$$
$$u_{xx} = f''$$
$$u_{xxx} = f'''$$

Then to satisfy the KdV, f must satisfy the ODE

$$\delta^2 f''' + ff' - cf' = 0$$

Consider f as defined in (2). Let $\xi = x - x_0$ for brevity. Then

$$f(x) = A\text{sech}^2\left(\frac{\xi}{\Delta}\right)$$

From now, for brevity, let $\text{sech}\left(\frac{\xi}{\Delta}\right) \equiv s$ and $\tanh\left(\frac{\xi}{\Delta}\right) \equiv t$, such that:

$$f(x) = As^2$$
$$f'(x) = -\frac{2A}{\Delta}s^2 t$$
$$f''(x) = \frac{2A}{\Delta^2}s^4 + \frac{4A}{\Delta^2}s^2 + 2$$
$$f'''(x) = \frac{16A}{\Delta^3}s^4 t - \frac{8A}{\Delta^3}s^2 t^3$$

Further noting that $\frac{\delta^2}{\Delta^3} = \frac{A}{12\Delta}$, we get that the left hand side of the above ODE is:

$$\begin{aligned}\text{LHS} &= \frac{16A\delta^2}{\Delta^2}s^4t - \frac{8A\delta^2}{\Delta^3}s^2t^3 - \frac{2A^2}{\Delta}s^4t + \frac{2cA}{\Delta}s^2t \\ &= \frac{A^2}{\Delta}s^2t\left(\frac{4}{3}s^2 - \frac{2}{3}t^2 - 2s^2 + \frac{2}{3}\right) \\ &= \frac{A^2}{\Delta}s^2t\left(\frac{2}{3}(1 - s^2 - t^2)\right) = 0 = \text{RHS}\end{aligned}$$

Thus the claim follows! ■

We shall define mass, M, and energy, E, of the motion as

$$M \equiv \int_0^1 u(x,t)\,dx \text{ and } E \equiv \int_0^1 \frac{1}{2}u^2(x,t)\,dx \tag{4}$$

, and shall prove another important result.

Theorem 2. *For a periodic solution to the KdV equation (1) that satisfied the conditions (2), M and E are independent of time.*

Proof. Firstly, note

$$\begin{aligned}\frac{dM}{dt} &= \int_0^1 u_t(x,t)\,dx \\ &= -\int_0^1 \left[uu_x + \delta^2 u_{xxx}\right]\,dx \\ &= -\left[\frac{1}{2}u^2 + \delta^2 u_{xx}\right]_0^1 \\ &= 0,\end{aligned}$$

where the second equality follows from the fact that u is a solution to (1), and the last equality follows from the periodic boundary conditions in (2). Thus mass M is independent of time.

Similarly, we notice:

$$\begin{aligned}\frac{dE}{dt} &= \int_0^1 uu_t\,dx \\ &= -\int_0^1 \left[u^2u_x + \delta^2 uu_{xxx}\right]\,dx \\ &= -\left[\frac{1}{3}u^3 - \delta^2 u_{xx}\left(uu_{xx} - \frac{1}{2}u_x^2\right)\right]_0^1 \\ &= 0,\end{aligned}$$

where the second equality follows from the fact that u is a solution to (1), the third equality follows by integrating by parts the previous line, and the last equality follows from the periodic boundary conditions in (2). Thus mass E is independent of time. ■

Question 2

We shall employ a leap-frog scheme, first employed by Zabusky and Kruskal, for solving the KdV, as detailed in the project brief. Let u_m^n be the solution at $x = hm, t = kn$ for $n, m = 0, 1, \cdots$, then the KdV equation is discretised to be

$$u_m^{n+1} = u_m^{n-1} - \frac{k}{3h}(u_{m+1}^n + u_m^n + u_{m-1}^n)(u_{m+1}^n - u_{m-1}^n) - \frac{k\delta^2}{h^3}(u_{m+2}^n - 2u_{m+1}^n + 2u_{m-1}^n - u_{m-2}^n) \tag{5}$$

We shall now prove something important about this scheme

Theorem 3. *The above numerical scheme has local truncation error of order $O(k^2+h^2)$*

Proof. Notice $k=\Delta t$ and $h=\Delta x$. Using Taylor expansion, we notice that the left hand side of (5) leads one to (letting $x=hm, t=kn$, and for brevity $u(x,t)\equiv u$):

$$u+u_tk+u_{tt}\frac{k^2}{2}+\frac{k^3}{6}u_{ttt}+O(k^4)$$

, and the right hand side leads one to:

$$u-u_tk+u_{tt}\frac{k^2}{2}-\frac{k^3}{6}u_{ttt}+O(k^4)-\frac{k}{3h}(3u+u_{xx}h^2+O(h^4))(2u_xh+\frac{1}{3}u_{xxx}h^3+O(h^5))-\frac{k\delta^2}{h^3}(2u_{xxx}h^3+\frac{1}{2}u_{xxxxx}h^5+O(h^7))$$

Comparing both sides, we get that the local truncation error, $e_{h,k}$ is such that

$$\begin{aligned}k\cdot e_{h,k}&=2k(u_t+uu_x+\delta^2u_{xxx})+\frac{k^3}{3}u_{ttt}+\frac{kh^2}{6}(2uu_{xxx}+4u_xu_{xx}+3\delta^2u_{xxxxx})+O(k^5+kh^4)\\&=\frac{k^3}{3}u_{ttt}+\frac{kh^2}{6}(2uu_{xxx}+4u_xu_{xx}+3\delta^2u_{xxxxx})+O(k^5+kh^4)\end{aligned}$$

Thus the local truncation error is of order $O(k^2+h^2)$. ∎

For $\delta=1$, we are given that the scheme is stable if

$$k\le\frac{h^3}{4+h^2|u_{\max}|}. \tag{6}$$

Consequently, we obtain the following claim:

Theorem 4. *The above numerical scheme, for general δ, is stable if*

$$k\le\frac{h^3}{4\delta^2+h^2|u_{max}|}. \tag{7}$$

Proof. Consider re-scaling the KdV equation (1), ie $t\to t'=\tau t, x\to x'=Lx$. This results in (1) being re-scaled as

$$\tau u_{t'}+Luu_{x'}+\delta^3L^3u_{x'x'x'}=0$$

Since we know the stability condition for (1) in the case of $\delta=1$, re-scaling KdV for general δ to resemble the case of $\delta=1$ seems to be the natural thing to do. As a result, we choose $L=\tau=\frac{1}{\delta}$. This results in

$$\Delta x'=L\Delta x=\frac{h}{\delta},$$

$$\Delta t'=\tau\Delta t=\frac{k}{\delta}.$$

This results in

$$\frac{k}{\delta}\le\frac{h^3/\delta^3}{4+\frac{h^2}{\delta^2}|u_{\max}|},$$

ie

$$k\le\frac{h^3}{4\delta^2+h^2|u_{\max}|},$$

thus the claim. ∎

We notice that we will need an alternative method for making the first step. For the initial time step, we use an uncentered scheme, ie

$$u_j^1=u_j^0-\frac{k}{6h}(u_{j+1}^0+u_j^0+u_{j-1}^0)(u_{j+1}^0-u_{j-1}^0)-\frac{\delta^2k}{2h^3}(u_{j+2}^0-2u_{j+1}^0+2u_{j-1}^0-u_{j-2}^0) \tag{8}$$

As one might note above, we use forward difference method for the initial time step in our scheme (this is the only difference when comparing this to the Zabusky-Kruskal scheme). This has an error of $O(k+h^2)$ however. But this is insignificant as if our initial time step is small enough, our error wouldn't change significantly (noting that this is only used for the first step). Furthermore, this still preserve the quadratic error in $x-$ space, which is ideal. Of course, using something like Crank-Nicholson is also an option, but since our initial leap-frog scheme

is explicit, it is ideal to use an explicit finite difference scheme for the first step as well. Furthermore, as in the Zabusky-Kruskal scheme, this scheme for the initial step still conserves mass for the initial time step, as if we say $M^j = \sum_{i=0}^{N-1} u_i^j$, then the above scheme clearly has $M^0 = M^1$. Furthermore, if we neglect $O(k^2)$ terms, this scheme still implies energy is almost conserved (by choosing $E^j = \sum_{i=0}^{N-1} \frac{1}{2}(u_i^j)^2$), as was the case for the Zabusky-Kruskal scheme.

Using this information, we implement the numerical scheme to solve the KdV equation with periodic boundary conditions in Python. The source code can be found in the program listings at the end of this report.

As as a check on the accuracy of the program, we calculate $u(x,t)$ for $t = 0.5, A = 2, x_0 = 0.25, \delta = 0.03$ and the initial data

$$u(x,0) = A\text{sech}^2\left(\frac{x - x_0}{\Delta}\right)$$

We have tabulated some of the numerical results in the table below. The domain is chosen from -4 to 4, which is enough to show the soliton pattern. Also we specify the values of h, k we used later. We note that the peak numerical value is 1.999886570588327, and the minimum numerical value is $-3.210783823236606e-05$.

x	$u(x,t)$
$-4.05.553304248963797e-06$	$5.553304248963797e-06$
...	...
-2.0000000000002203	$-8.274603759880045e-06$
...	...
0.0019999999995592432	$-2.1253974683254145e-05$
...	...
0.5819999999994954	1.9994710641403395
0.5839999999994951	1.999886570588327
0.5859999999994949	1.997340331699849
...	...
1.001999999999449	$9.236110179304313e-05$
...	...
3.997999999999119	$6.220319698627268e-06$

Table 1: Tabulated values of $u(x,t)$

We used $h = 0.002$ and $k = 0.5 * \frac{h^3}{4\delta^2+2h^2}$. Reasoning for this is to ensure the the numerical solution is, on average, as close as possible to the analytical solution, at least up to 3 significant figures (again, on average). This can be observed, for instance, by looking at mean error for different values of h (and similarly calculated values of k as above) in the plot below. Keen reader might notice that the above value of k is chosen such that the scheme is stable (as $u_{\max} = 2$).

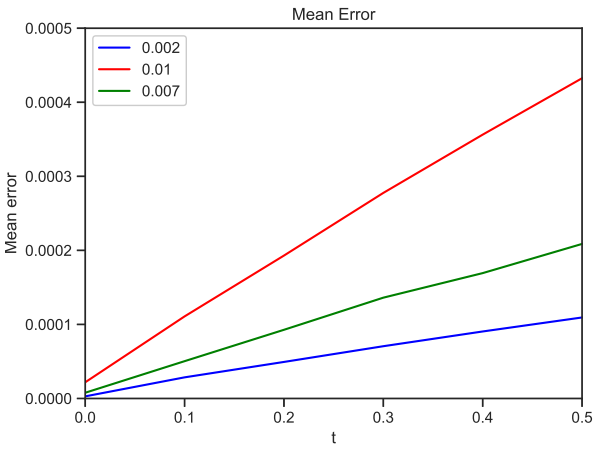

Figure 1: Plots of mean errors vs t

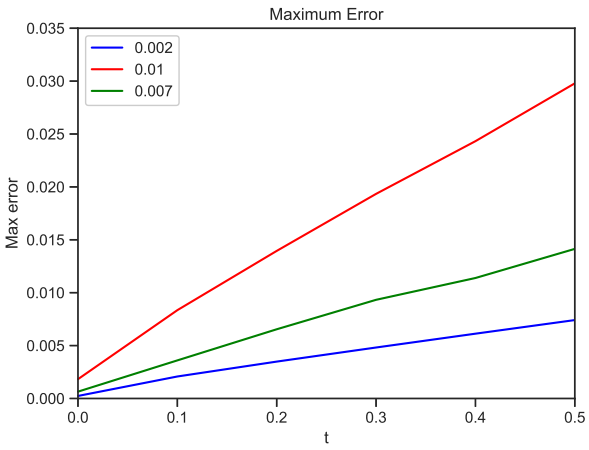

Figure 2: Plots of maximum errors vs t

We notice, further, that the mean error seems to increase as time t increases. This trend is followed by maximum error as well, and thus we choose $h = 0.002$. We could have chosen a smaller value, but in favour of ensuring we have a manageable number of iterations, and thus the program runs in reasonable time, we chose to stick with $h = 0.002$. In this case, we also note that at this value of h, the max error is less than 0.01, thus ensuring, in worst case, our numerical solution is correct up to 2 significant figures. We note that the mean error in our case is 0.00021667947602558365, and maximum error is 0.0074210428880802142.

We have plotted the numerical solution and the exact solution at $t = 0.5$ below.

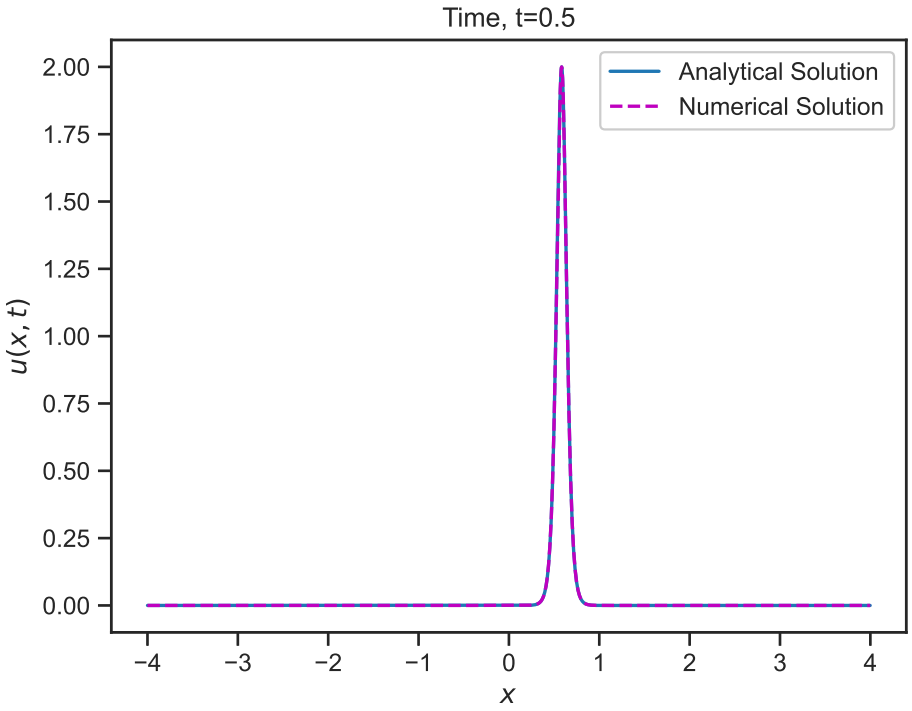

Figure 3: Plots of $u(x, t)$ vs x, at $t = 0.5, A = 3, \delta = 0.003, x_0 = 0.25$

We notice there are a few differences between the numerical and analytical solution, although fairly minor. One of them is that the propagation speed of the numerical solution is smaller than that of the analytical solution when plotted for the same values of t, albeit very slightly. This can be observed by noting that numerical solution is very slightly translated in the negative x direction compared to the analytical solution (or noting that the peak for the numerical solution occurs sooner than the analytical solution). As we shall realise later, this conclusion is in fact incorrect, and the numerical solution propagates faster than the numerical solution. The cause for this inaccuracy is due to numerical inaccuracies in our plots of the analytical solution, as the

steps used for the plots are not small enough for the plots to be accurate enough. An evidence for this is in the plot below.

Another thing we notice is that the numerical solution's peak is greater than the value of the exact solution at the same point, although again, very slightly (and the numerical solution peak is almost the value of the actual analytical solution peak of 2 up to at least 5 significant figures in the case of $h = 0.002$, as seen in Table 1). Also, we notice that the for the same values of t, the numerical solution peaks higher than the analytical solution(when plotted for the same values of t), as evidenced in plot below. Of course, however, analytically, the solution peaks at value of 2 which is greater than the numerically calculated peak of the exact solution (thus suggesting, again, inaccuracy in our plot). In fact, for the case of $h = 0.002$, we expect the peak to be at the value of 1.999886570588327, which is less than analytically expected 2, but highly accurate to the exact peak of 2.

Furthermore, there are points where the numerical solution is less than 0. This is again evidenced in the plot below. To observe this better (as the error is too small to be visible in plots by naked eye), we used $h = 0.007$

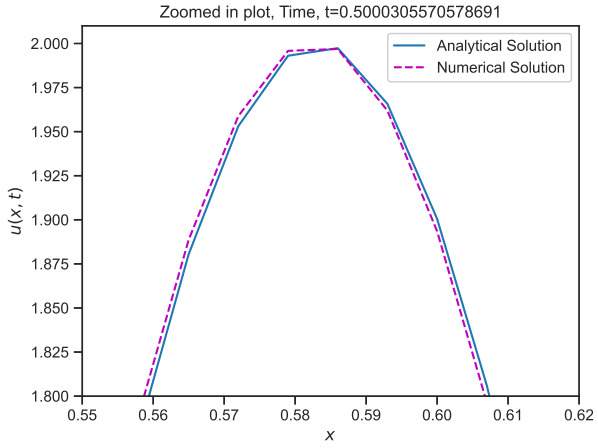

Figure 4: Near the peak of the numerical solution

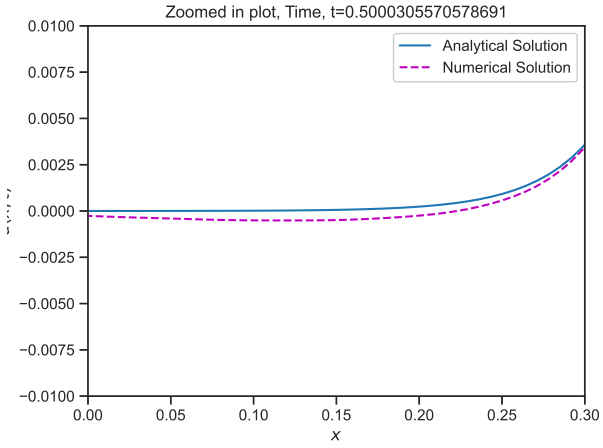

Figure 5: Near the bottom of the solution

Zoomed in plots of $u(x,t)$ vs x, at $t = 0.5, A = 3, \delta = 0.003, x_0 = 0.25$

As was noted earlier, the propagation speed of the numerical solution doesn't agree **exactly** with that of the analytical solution. However, for the case of $h = 0.002$, this difference between the two propagation speeds is not very large. In fact, analytically, we expect the peak at $t = 0.5$ to be at $x = 0.58333$. From the numerical solution (as illustrated in Table 1), the peak for the numerical solution occurs at $x = 0.5839999999994951$. Thus suggesting that the speed, till $t = 0.5$, numerical is $c_{num} = 0.66799999999$, which is greater than the analytically expected speed of propagation!

Question 3

We now aim to analyse the evolution of the initial data corresponding to the sum of two solitons, one with $A = 2, x_0 = 0.25, \delta = 0.03$, and the other with $A = 1, x_0 = 0.75, \delta = 0.03$. That is, our initial condition is

$$u(x,0) = 2\text{sech}^2\left(\frac{x-0.25}{\sqrt{12(0.03)^2/2}}\right) + \text{sech}^2\left(\frac{x-0.75}{\sqrt{12(0.03)^2}}\right)$$

For this question, to decrease the overall number of iterations requires, we use $h = 0.005$ and $k = 0.5 * \frac{h^3}{4\delta^2+2h^2}$. The value of k is chosen as we expect, due to mass conservation, for the maximum value of $u_{\max}$ to never cross 2. Furthermore, our domain of integration for this PDE is $[-8, 8]$ as this domain is large enough to illustrate the time evolution of the solitons sufficiently well. We illustrate our plot at different value of t below, to indicate the time evolution of our solution.

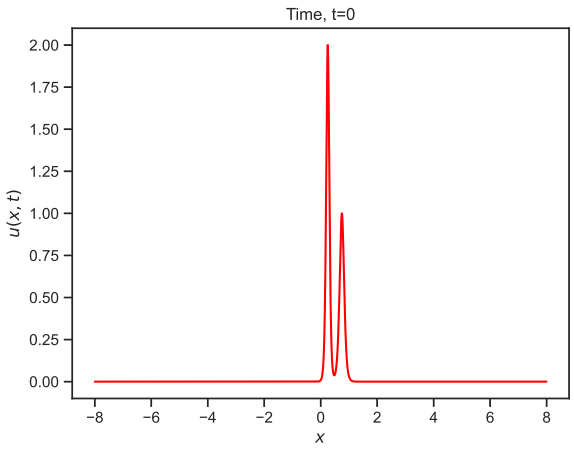

Figure 6: $t = 0$

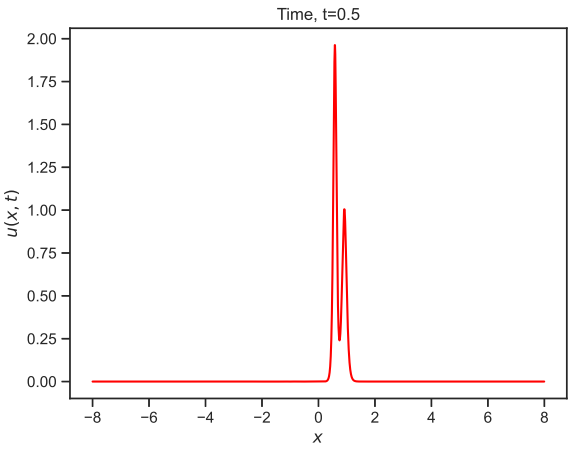

Figure 7: $t = 0.5$

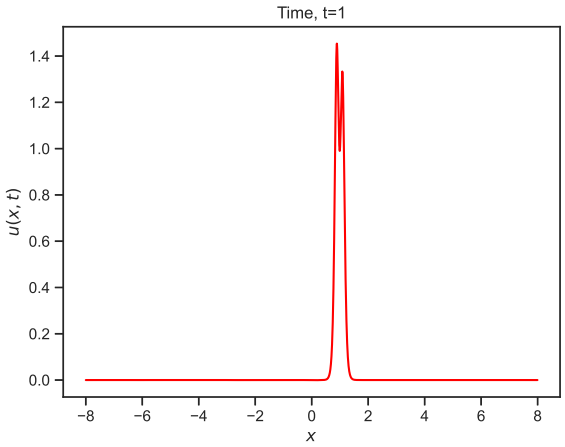

Figure 8: $t = 1$

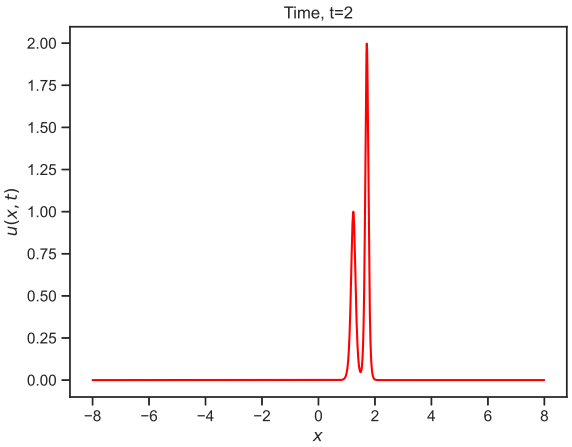

Figure 9: $t = 2$

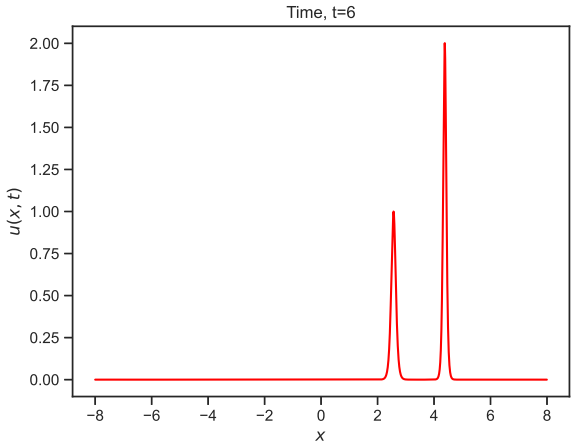

Figure 10: $t = 6$

Plots of $u(x, t)$ at different values of t

The time evolution plots above suggest elastic collision of solitons. Waves emerging from collision retain shape and speed of initial two waves (had they time evolved individually), but are phase shifted. The faster wave (the soliton with initial condition $A = 2$)is shifted forward in the combined initial condition solution, and the slower waves (the soliton with initial condition $A = 1$) is shifted backwards. This is illustrated below for time $t = 6$.

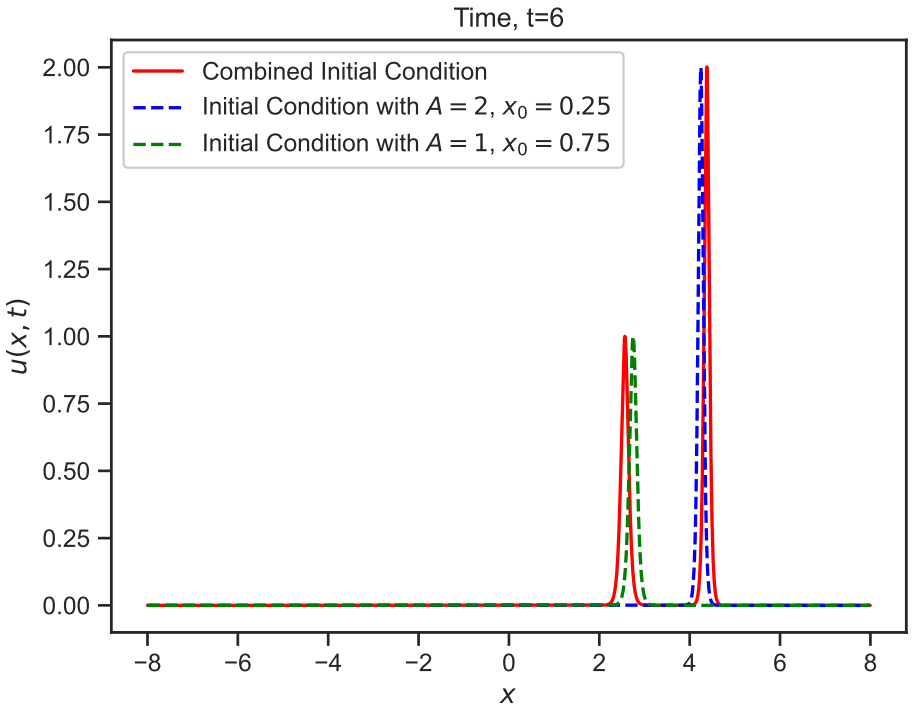

Figure 11: Plot at $t = 6$, with our solution overlaid with time evolved solutions with the initial conditions individually

Another observation is that the solution has double peak throughout the collision, and an exponentially diminishing tail as $|x| \to \infty$. Something interesting to notice is that the interaction is non-linear, as the solitons first bounce and then exchange shape and speeds. This illustrates the non-linearity of the PDE!

In addition, we use the trapezium rule to calculate the masses and energies associated with our solutions at different points in time. These are tabulated below.

t	M	E
0	0.5017848660422465	0.26544920710345604
0.5	0.5017848867163822	0.265449206495078
1.0	0.5017850158660467	0.26544920335787764
...	...	...
2.0	0.5017845378253456	0.2654492064808077
...	...	...
6.0	0.5017851294669069	0.26544920346284406
...	...	...
8.0	0.5017846229854733	0.26544920488647356

Table 2: Tabulated values of E and M at different points in time

The variation in time of mass M is observed to be small, and on the order of $O(10^{-7})$, thus suggesting that M is approximately constant during this time evolution, and up to 6 significant figures,

$$M \approx 0.501785.$$

Similarly, the variation in time of Energy E is observed to be small and on the order of $O(10^{-9})$, thus suggesting,

once again, that E is approximately constant during this time evolution, and up to 7 significant figures,

$$E \approx 0.2654492$$

If we evolved the initial conditions individually, calling them E_1, E_2 and M_1, M_2 respectively for initial condition with $A = 2$ or $A = 1$, and E, M for the combined initial condition. We tabulated the difference $M_1 + M_2 - M$ and $E_1 + E_2 - E$ at different times t below. We notice that roughly, we have that $M = M_1 + M_2$ (upto at least 6 significant figures) and $E \approx E_1 + E_2$ (up to 4 significant figures) (in fact we note that $E_1 + E_2 < E$ fairly consistently below). This behaviour further reaffirms our earlier observation of the collisions being elastic, as energy and mass both are (roughly) conserved(!).

t	$M_1 + M_2 - M$	$E_1 + E_2 - E$
0.0	0.0	-0.0002079914328393384
0.5	$2.2810198174738616e-10$	-0.0002079916979558496
1.0	$-8.347679769649119e-09$	-0.0002079910085527877
...	...	...
2.0	$-5.216014897779786e-08$	-0.0002079935844609703
...	...	...
6.0	$-3.847344430818467e-07$	-0.0002079881393480254
...	...	...
9.0	$-9.326868477543826e-08$	-0.0002079920360705811

Table 3: Tabulated values of $E_1 + E_2 - E$ and $M_1 + M_2 - M$ at different points in time

Question 4

We shall now consider the boundary condition

$$u(x, 0) = \sin(2\pi x)$$

In the case of $\delta = 0$, qualitatively, the solution gets steeper and steeper, eventually leading to $\left|\frac{\partial u}{\partial x}\right| \to \infty$ at $x = \frac{1}{2}$, that is time evolution indicated eventual formation of a shock. This is illustrated in figures below.

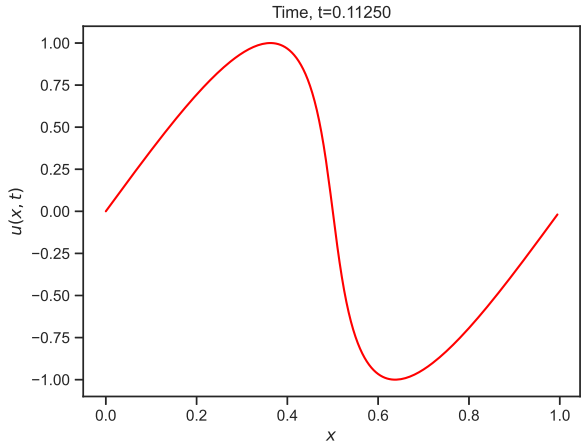

Figure 12: $t = 0.7/2\pi$

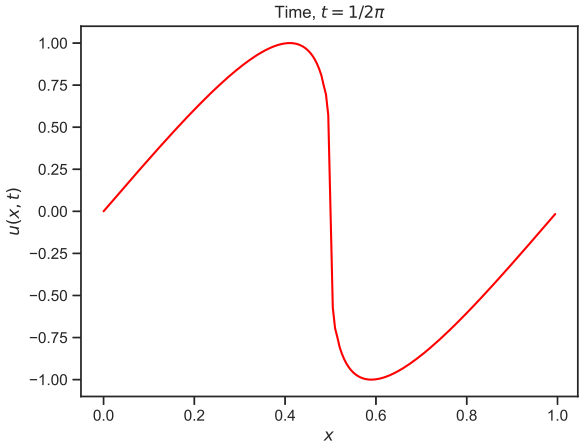

Figure 13: $t = 1/2\pi$, the time when shock forms

Plots for the above initial condition, at $\delta = 0$

This makes sense, as if we use method of characteristics with $\delta = 0$, we note that the solution has to satisfy $u = \sin(2\pi x - ut)$. However, then we notice that the characteristics eventually cross each other, leading to

formation of shocks. This happens at $t = t_* = \frac{1}{2\pi}$

We shall now analyse the KdV equation. Recall that the KdV equation we are considering is

$$u_t + uu_x + \delta^2 u_{xxx} = 0$$

Let L, T, U be the typical scales for x, t, u. Then the typical scale in the KdV are

$$u_t \sim \frac{U}{T}$$

$$uu_x \sim \frac{U^2}{L}$$

$$\delta^2 u_{xxx} \sim \frac{\delta^2 U}{L^3}$$

We notice that the non linear term dominates if $\frac{U^2}{L} \gg \frac{\delta^2 U}{L^3}$, ie $U \gg \frac{\delta^2}{L^2}$. If this is true, then we have non-linearity dominating, and the KdV is approximately $u_t + uu_x = 0$, ie the case of $\delta = 0$. However, time evolution using this equation, as seen above, leads to formation of shock due to steepening.

If the non-linear term is dominated by the third order derivative term, then we require $U \ll \frac{\delta^2}{L^2}$, in which case we have a linear PDE: $u_t + \delta^2 u_{xxx} = 0$. This behaviour changes, thus, depending on the size of $\frac{UL^2}{\delta^2}$. At first, non-linearity dominates, and solution tends towards a shock. If $\delta \neq 0$, however, then this shock is never formed, and the dispersive term dominates. Then we get well defined waves.

We further investigate, numerically, the case of $\delta = 0.03$, and observe that the above analysis holds true! This can be observed in our plots below.

We notice we approximately have a single peak till about $t \approx 0.10101$, after which we have formation of a second peak. Further, we also notice that till this time, the solution roughly follows the qualititative behaviour of the solution in the case of $\delta = 0$. After this time, we have formation of another peak and the solution steepens, becoming steepest at around $t \approx 0.335$. After this time which we get a train of waves formed as we get formation of a third peak (and an indication that in the KdV, the non-linear term is dominated by the dispersive term)! In this case, we have a train of 3 waves. We observe that in this train of waves the taller waves seem to be travelling faster than the shorter ones. This makes sense, as in shallow water, we would expect phase velocity $c_p \propto h$, where h is the height of the wave. We also notice that these waves, eventually, have non-linear (almost-elastic) collisions (as the faster waves are 'behind' the slower waves), and swap (as was the case in Question 3).

Till $t \approx 0.101$, thus, our solution predominantly satisfies the non-linear $u_t + uu_x = 0$. So we claim that numerical solution changes character at a certain time, t_B, which in this case is $t_\mathrm{B} \approx 0.10101$.

We further aim to investigate our numerical solution for different values of δ. We shall use two values of δ, particularly $\delta = 0.01$ and $\delta = 0.05$ as evidence in this project report.

For $\delta = 0.01$, the plots to indicate time evolution can be found below. We notice that the solution exhibits similar behaviour as earlier for the case of $\delta = 0.03$. However, interestingly the solution seems to indicate $t = t_\mathrm{B} \approx 0.12361$, as till this time, the solution has a single peak and the evolution of the solution, qualitatively, is very similar to that for the solution in the case of $\delta = 0$. Further, we notice that the solution forms multiple peaks, and continues to get steeper till $t \approx 0.29958$, when it is the steepest. After this time, more peaks are formed, and once again we get a train of waves. In this case, we have 9 peaks. Interestingly, the amplitudes seem to follow a roughly linear relation. Furthermore, the propagation speed of these individual waves in this train of waves follows the same relation as was expressed in the case of $\delta = 0.03$. Also, we notice that these train of waves take part in non-linear (almost-elastic) collisions (as the faster waves are 'behind' the slower waves), and swap (as was the case in Question 3).

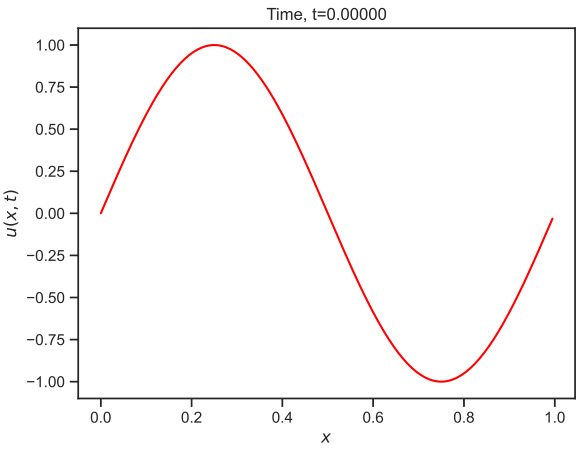

Figure 14: $t = 0.7/2\pi$

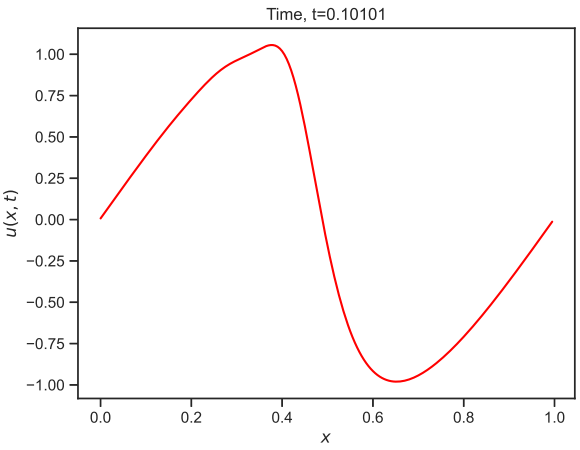

Figure 15: $t = 1/2\pi$, the time when shock forms

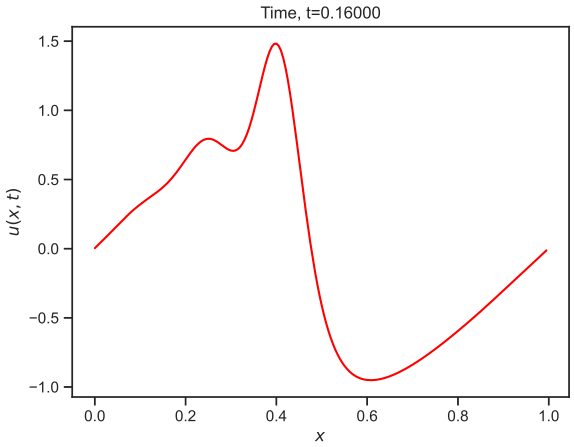

Figure 16: $t = 1/2\pi$, the time when shock forms

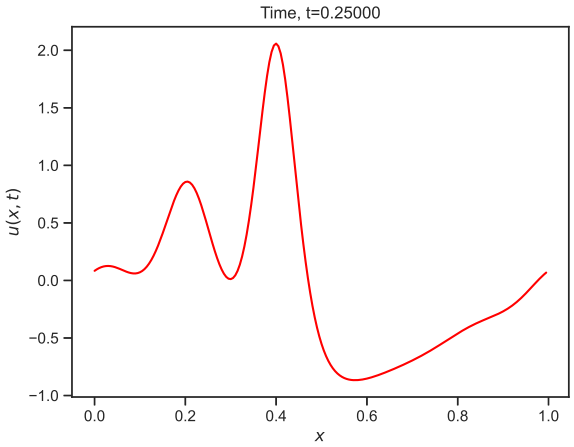

Figure 17: $t = 1/2\pi$, the time when shock forms

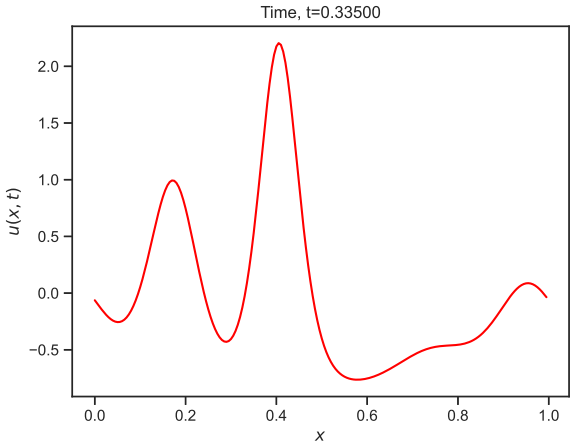

Figure 18: $t = 1/2\pi$, the time when shock forms

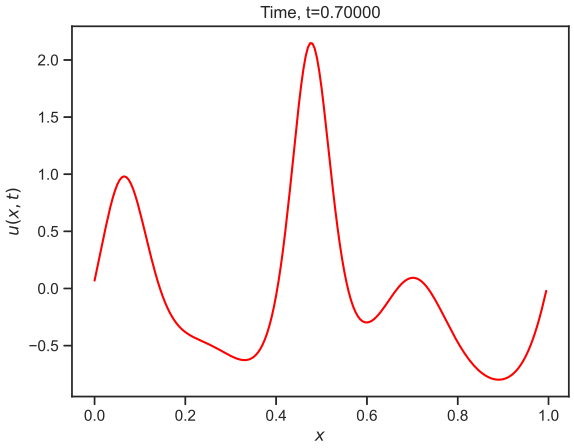

Figure 19: $t = 1/2\pi$, the time when shock forms

Plots for the above initial condition, at $\delta = 0.03$, at different times t

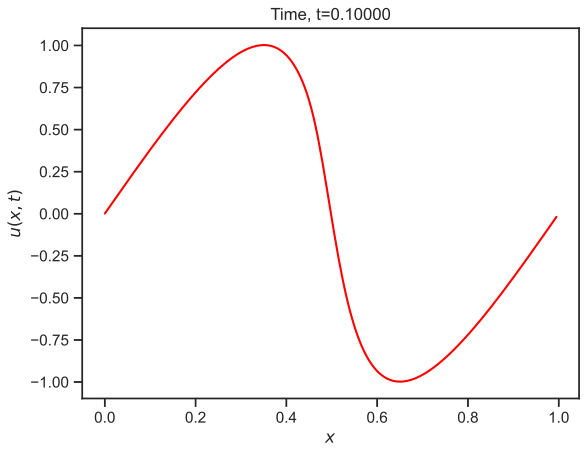

Figure 20: $t = 0.7/2\pi$

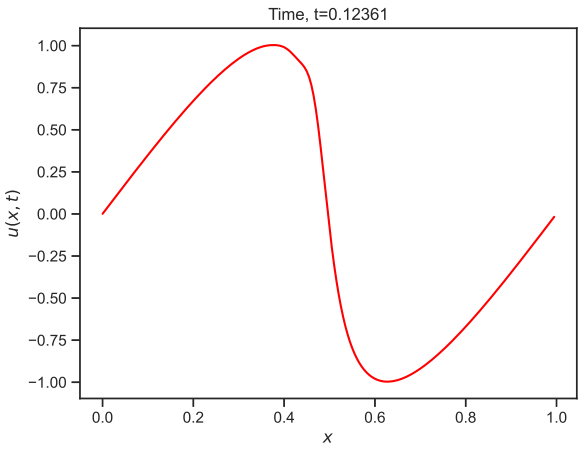

Figure 21: $t = 1/2\pi$, the time when shock forms

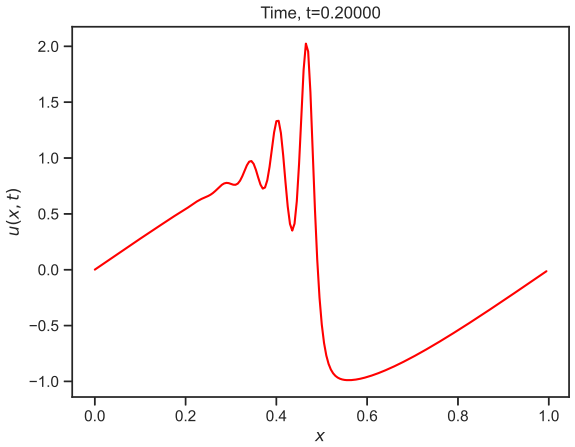

Figure 22: $t = 1/2\pi$, the time when shock forms

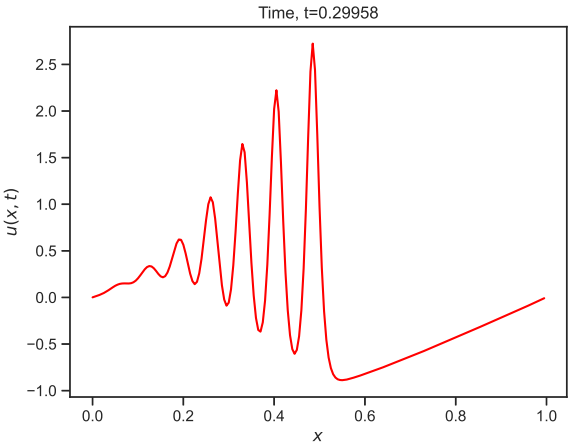

Figure 23: $t = 1/2\pi$, the time when shock forms

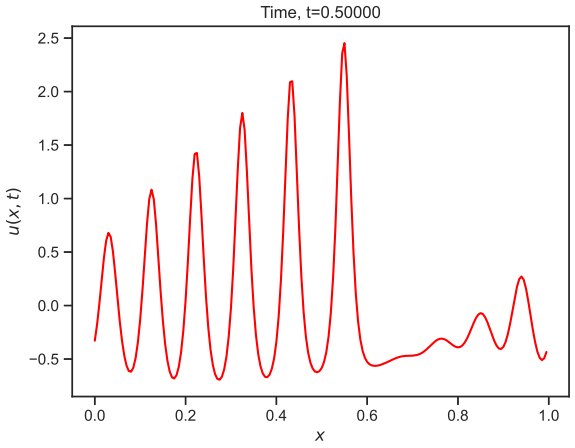

Figure 24: $t = 1/2\pi$, the time when shock forms

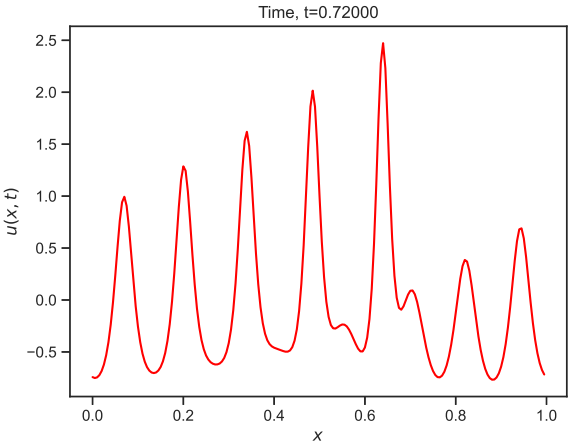

Figure 25: $t = 1/2\pi$, the time when shock forms

Plots for the above initial condition, at $\delta = 0.01$, at different times t

Lastly, we aim to analyse the solution for the case of $\delta = 0.05$. In this case, the plots to indicate time evolution can be found below. We notice that the solution exhibits, yet again, similar behaviour as earlier for the case of $\delta = 0.03$ and $\delta = 0.01$. However, interestingly the solution seems to indicate $t = t_{\mathrm{B}} \approx 0.0905$. Further, we notice that the solution forms multiple peaks, and continues to get steeper till $t \approx 0.351$, when it is the steepest. After this time, more peaks are formed, and once again we get a train of waves. In this case, we have only 2 peaks. Furthermore, the propagation speed of these individual waves in this train of waves follows the same relation as was expressed in the case of $\delta = 0.03$. Also, we notice that these train of waves take part

in non-linear (almost-elastic) collisions (as the faster waves are 'behind' the slower waves), and swap (as was the case in Question 3).

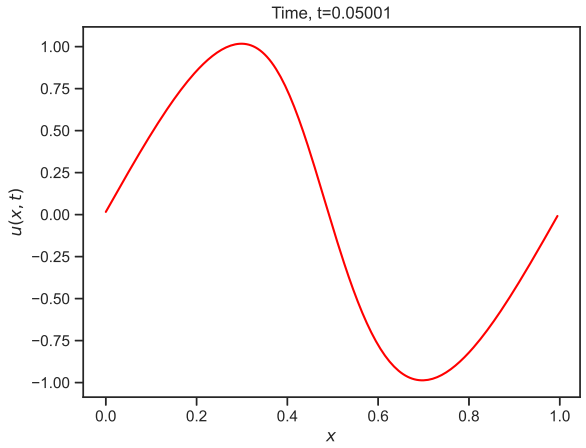

Figure 26: $t = 0.7/2\pi$

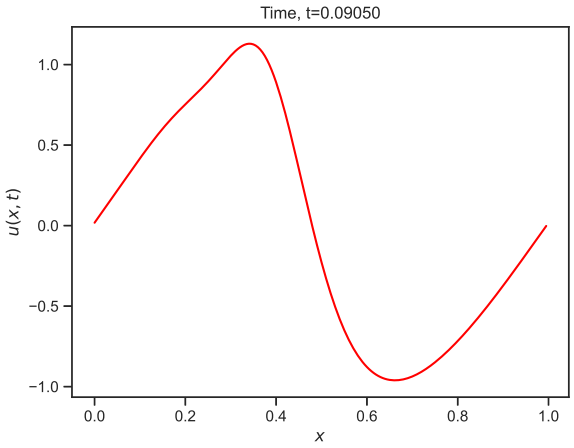

Figure 27: $t = 1/2\pi$, the time when shock forms

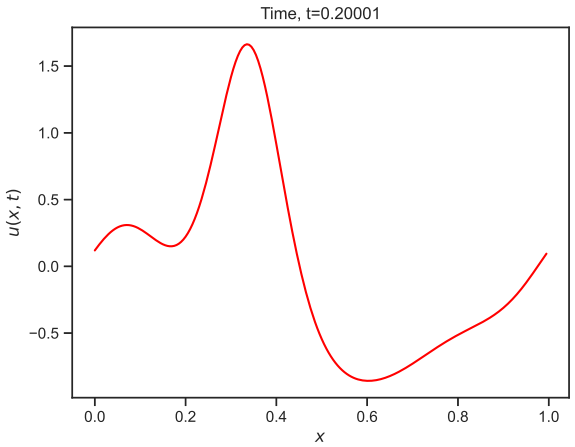

Figure 28: $t = 1/2\pi$, the time when shock forms

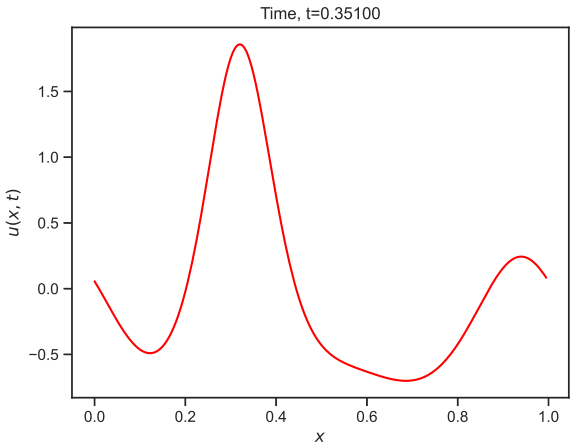

Figure 29: $t = 1/2\pi$, the time when shock forms

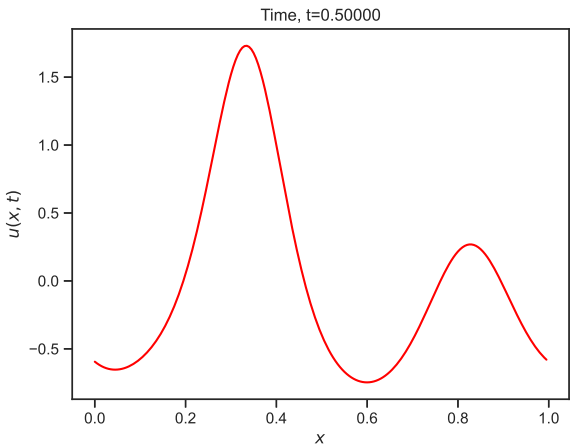

Figure 30: $t = 1/2\pi$, the time when shock forms

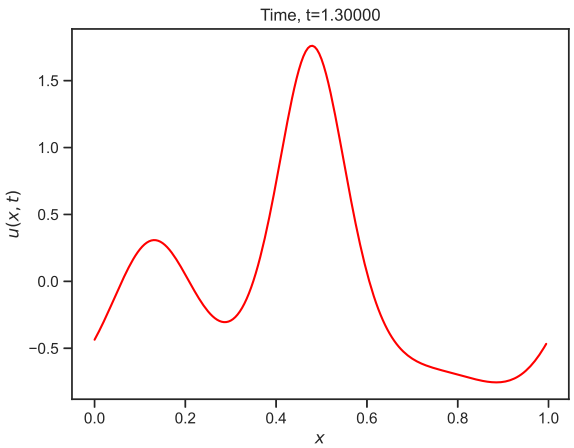

Figure 31: $t = 1/2\pi$, the time when shock forms

Plots for the above initial condition, at $\delta = 0.01$, at different times t

Overall, we notice that as δ becomes larger than $\delta = 0.03$, the number of waves in the train in the solution decrease, and as δ approaches 0 (and thus smaller than 0.03), the number of waves in the train of waves seem to dramatically increase.

Another thing we notice is that the point till which the numerical solution seems to follow the equation in $\delta = 0$, and thus maintaining the same character ends at a certain time t_{B} which is different for different values of δ. However, we note a general trend, wherein t_{B} seems to decrease as δ increases, and seems to increase as δ decreases below $\delta = 0.03$. Interestingly enough (and something that makes analytical sense), t_{B} seems to approach t_* as δ approaches 0.

Another thing we notice is that the point where the solution is steepest, after which (we think) the dispersive term dominates in the KdV equation also seems to follow a trend. We note a general trend, wherein this time seems to decrease as δ decreases, and seems to increase as δ increases above $\delta = 0.03$, which is opposite of the trend for t_{B}. But this makes sense, because as δ decreases and approaches 0, we expect the dispersive term to become more and more insignificant, and thus we expect it to take really long time for dispersive term to dominate. This indicates that we expect this time to tend to ∞ as we get closer and closer to $\delta = 0$.

Lastly, and rather trivially, we note that the maximum value $|u_{\max}|$ seems to increase as δ decreases below 0.03 (and thus as it gets closer to $\delta = 0$), and seems to decrease as δ increases above 0.03. This behaviour is once again justified by the fact that as $\delta = 0$, we expect formation of shocks, and so would expect steeper and steeper solutions as $\delta \downarrow 0$. And as δ increases, we expect the dispersive term to dominate, and thus would expect a solution that is flatter and less steep (and we would expect, eventually, $u_{\max} \downarrow 1$).

Appendix A: Program Listings

Listing 1: Program for Question 2

```
#---start of def---
def initial_cond(x,A,x0,delta,init_type):
    if(init_type==1):
        a = np.cosh((1/delta) *(1/np.sqrt(12)) * np.sqrt(A) * (x - x0))
        return A/(np.multiply(a,a))
    else:
        b=np.sin(2*np.pi*x)
        return b
#---end of def---
#---start of def---
def U1(u0, h, k, delta):
    up1 = np.hstack([u0[1:], u0[:1]])
    up2 = np.hstack([u0[2:], u0[:2]])
    u = np.array(u0)
    um1 = np.hstack([u0[-1:], u0[:-1]])
    um2 = np.hstack([u0[-2:], u0[:-2]])

    a = (up1 - um1) / (2 * h)
    b = (up2 - 2 * up1 + 2 * um1 - um2) / (2 * h * h * h)
    d = (up1 + u + um1) / 3

    sol=u0-k*np.multiply(a,d)-delta*delta*k*b
    return sol
#---end of def---
#---start of def---
def U2(u0, u1, h, k, delta):
    up1 = np.hstack([u1[1:], u1[:1]])
    up2 = np.hstack([u1[2:], u1[:2]])
    up3 = np.hstack([u1[3:], u1[:3]])
    up4 = np.hstack([u1[4:], u1[:4]])
    u = np.array(u1)
    um1 = np.hstack([u1[-1:], u1[:-1]])
    um2 = np.hstack([u1[-2:], u1[:-2]])
    um3 = np.hstack([u1[-3:], u1[:-3]])
    um4 = np.hstack([u1[-4:], u1[:-4]])

    a = (up1 - um1) / (2 * h)
    b = (up2 - 2 * up1 + 2 * um1 - um2) / (2 * h * h * h)
    d = (up1 + u + um1) / 3

    return u0 - 2 * k * np.multiply(d, a) - 2 * delta*delta * k * b
#---end of def---
#---start of def---
def solver(u0,u1,h,k,delta,steps):
    for i in range(steps-1):
        U = U2(u0,u1,h,k,delta)
        u0=u1
        u1=U
    return u1
#---end of def---
#---start of def---
def calc_k(h,delta,A):
    k=h*h*h/(4*delta*delta+h*h*A)
    return k
#---end of def---
#---start of def---
def analytical_solution(x,t,A,delta,x0):
```

```
    #a*(sech(sqrt(a/(delta*delta*12))*(x-c*t-x0)) .* sech(sqrt(a/(delta*
    delta*12))*(x-c*t-x0)) )
    c=A/3
    a = np.cosh((1/delta) *(1/np.sqrt(12)) * np.sqrt(A) * (x - x0 - c*t))
    return A/(np.multiply(a,a))
#---end of def---
#---start of def---
def visualise(u0,u1,h,k,delta,steps, ylim: tuple = (-1,1.5), xlim = (-8,8),
    anim_interval = 100):

    fig, ax = plt.subplots();
    line, = ax.plot([], [])
    frames_req=steps+1
    def init():
        line.set_data([], [])
    def animate(i):
        global u0
        global u1
        global U
        if(i==0):
            line.set_data(x, u0)
        if(i==1):
            line.set_data(x, u1)
        else:
            for j in range(1):
                U = U2(u0,u1,h,k,delta)
                u0 = u1
                u1 = U
        fig.suptitle("Time t="+str(i*k))
        line.set_data(x, U)

    anim = animation.FuncAnimation(fig, animate, init_func=init, frames=
    frames_req, interval = anim_interval, repeat=False)

    plt.close()

    return HTML(anim.to_jshtml())
#---end of def---
```

Listing 2: Program for Question 3

```
A1=2
x01=0.25
A2=1
x02=0.75
h=0.005
x=np.arange(-8,8,h)
N=int(np.ceil(16/h))
delta=0.03
k=min(0.5*calc_k(h,delta,A1), 0.5*calc_k(h,delta,A2))
initial_cond_param_1={'A':A1,'x0':x01,'delta':delta,'init_type':1}
initial_cond_param_2={'A':A2,'x0':x02,'delta':delta,'init_type':1}
solution_params_first_step = {'h':h, 'k':k, 'delta':delta}

print("h="+str(h)+"; k="+str(k))

t_list=[6]

for t in t_list:
    M=int(np.ceil(t/k))
    num_solution_params_final = {'h':h, 'k':k, 'delta':delta,'steps':M}

    u_prev = initial_cond(x, **initial_cond_param_1)+initial_cond(x, **
   initial_cond_param_2)
    u_curr = U1(u_prev, **solution_params_first_step)
    u_final_num=solver(u_prev,u_curr,**num_solution_params_final)

    u_prev_1 = initial_cond(x, **initial_cond_param_1)
    u_curr_1 = U1(u_prev_1, **solution_params_first_step)
    u_final_num_1=solver(u_prev_1,u_curr_1,**num_solution_params_final)

    u_prev_2 = initial_cond(x, **initial_cond_param_2)
    u_curr_2 = U1(u_prev_2, **solution_params_first_step)
    u_final_num_2=solver(u_prev_2,u_curr_2,**num_solution_params_final)

    plt.plot(x,u_final_num,'r',label='Combined Initial Condition')
    plt.plot(x,u_final_num_1,'b--',label='Initial Condition with $A=2$, $x_0
   =0.25$')
    plt.plot(x,u_final_num_2,'g--',label='Initial Condition with $A=1$, $x_0
   =0.75$')
    plt.title("Time, t="+str(t))
    plt.ylabel('$u(x,t)$')
    plt.xlabel('$x$')
    plt.legend()
    plt.savefig('figq4big.eps')
    plt.show()

    Mass=np.trapz(u_final_num, x)
    E=np.trapz(0.5*np.multiply(u_final_num,u_final_num),x)

    print("Mass, M = "+str(Mass))
    print("Energy, E = "+str(E))
```

Listing 3: Program for Question 4

```
A=2
x0=0.25
h=0.005
t=0.335
x=np.arange(0,1,h)
N=int(np.ceil(16/h))
delta=0.03
k=0.5*calc_k(h,delta,A)
M=int(np.ceil(t/k))

initial_cond_param={'A':A,'x0':x0,'delta':delta,'init_type':2}
solution_params_first_step = {'h':h, 'k':k, 'delta':delta}
num_solution_params_final = {'h':h, 'k':k, 'delta':delta,'steps':M}
analytical_solution_params_final={'A':A,'delta':delta,'x0':x0}

u_prev = initial_cond(x, **initial_cond_param)
u_curr = U1(u_prev, **solution_params_first_step)
u_final_num=solver(u_prev,u_curr,**num_solution_params_final)

print("h="+str(h)+"; k="+str(k))
print(max(u_final_num))
plt.plot(x,u_final_num,'r')
plt.title("Time, t="+str('%.5f'%(k*2+k*(M-2))))
plt.ylabel('$u(x,t)$')
plt.xlabel('$x$')
plt.savefig('figq5av.eps')
plt.show()
```