Gravity Waves and Shocks in a Shallow Fluid

Exposition of the Problem A shallow, uniform, inviscid fluid under the influence of gravity can be described by the *shallow water equations*. Using these equations, we will simulate the evolution of a fluid in a long, narrow flume[1]. We will take the width of the flume to be w, the length L and the mean (resting) depth of the fluid will be H. The flow will be estimated at $N+1$ locations along the flume tank (see Figure 1). The other variables are as follows: $u(x,t)$ is the lengthwise flow speed, $h(x,t)$ and $\eta(x,t)$ are the total fluid depth and deviation from the mean depth, respectively and $\phi(x,t)$ is the concentration of a passive tracer (such as a colored dye, usually measured in grams per kilogram).

We will produce two simple models of this fluid: linear and non-linear. For the non-linear model, we will load the flow velocity u, the surface elevation of the fluid h and the concentration of the passive tracer ϕ from the data file `soln.mat`. The data files `u_lin.dat` and `h_lin.dat` store the flow variables u and h for the linear model. In Part 1, we will load these solutions for both the linear and non-linear model from the data files `soln.mat`, `u_lin.dat`, and `h_lin.dat` and initialize the model parameters. We will visualize these solutions by making several plots and animations. Once we have initialized and visualized the model, in Part 2 we will reproduce these results and make a new, linear prediction of the tracer ϕ, which is not provided in the data files. There are two existing functions (`nonlinear.m` and `gravity.m`) which we will call in order to reproduce the non-linear solutions from Part 1. We will also construct and call a function `transport.m` in order to make an estimate of the concentration of the passive tracer ϕ transported by the flow u, using the linear model. Finally, in Part 3 we will analyze the results of both the linear and non-linear models. We will construct another function `trapezoidal.m` in order to estimate integrals. Using this function, we will compute several integrals of the motion and examine the conservation laws of mass and energy, for both the linear and non-linear models. We will also examine the development of shocks in the fluid.

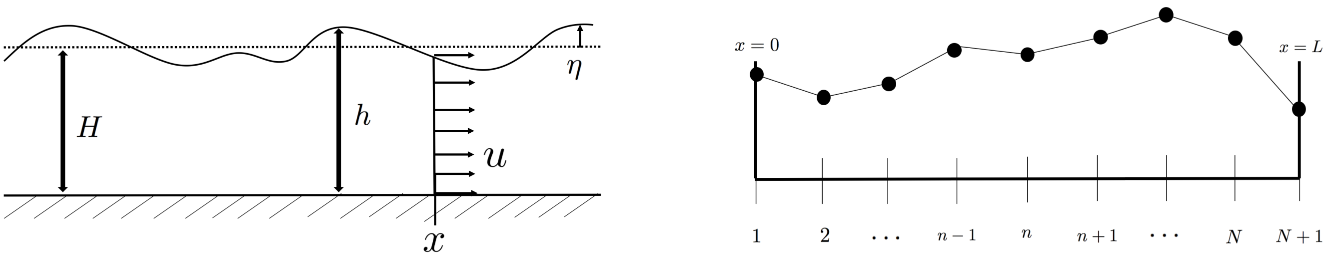

Figure 1: Left: Diagram of a shallow water system. The mean or resting depth is H and $\eta(x,t)$ represents transient deviations of the surface from H. The total depth of the fluid h is given by $h = H + \eta$. The flow speed $u(x,t)$ is independent of depth. Right: Schematic of the nodal grid for computing the numerical solution. There are $N+1$ evenly spaced nodes from $x = 0$ to $x = L$.

Figure 2: Coastal and Hydraulic Engineering Research Laboratory and the flume at SUNY Stony Brook.[1]

1. **Initializing the Model, Loading and Visualizing Solutions** Enter your directory `EAS230XXXXXXXX` and create a sub-directory `PP`. Enter the directory `PP` and save the files `u_lin.dat`, `h_lin.dat`, `soln.mat` as well as the functions `gravity.m` and `nonlinear.m`. Create a new script `PP.m` and define the following parameters and initial state ($m = 1$):

   Scalars Use the following variables to define the scalar quantities below: `L`, `g`, `H`, `rho`, `w`, `N`, `M`, `dx`, `dt`

$$L = 18 \text{ m} \qquad g = 9.81 \text{ m s}^{-2}$$

$$H = 0.75 \text{ m} \quad \rho = 1000 \text{ kg m}^{-3} \quad w = 1.5 \text{ m}$$

$$N = 128 \qquad M = 512$$

$$\Delta x = L/N \quad \Delta t = 0.9\Delta x/\sqrt{gH}$$

   Vectors / 1D Arrays Use the variables `x`, `theta` and `t` to define the following arrays:

$$x_n = 0, \Delta x, 2\Delta x, ..., (n-1)\Delta x, ..., (N-1)\Delta x, N\Delta x$$

$$\theta_n = 2\pi x_n / L$$

$$t_m = 0, \Delta t, 2\Delta t, ..., (m-1)\Delta t, ..., (M-2)\Delta t, (M-1)\Delta t$$

   Matrices / 2D Arrays Initialize variables `u`, `h`, `eta` and `phi` with zeros of size $(N+1) \times M$ and populate the first column using the following formulae:

$$u(x,0) = \tfrac{5}{11}\sqrt{gH}\,\sin\left(\tfrac{\theta}{2}\right)$$

$$h(x,0) = H\left(\tfrac{1}{3}\cos^{16}\left(\tfrac{\theta-\pi}{2}\right) + \tfrac{1}{5}\cos^{32}\left(\left(\theta - \tfrac{\pi}{2}\right)/4\right) + \left(1 - \tfrac{x}{L}\right) + \tfrac{691}{1818}\right)$$

$$\eta(x,0) = h(x,0) - H$$

$$\phi(x,0) = \tfrac{1}{100}\,\mathrm{e}^{-128\left(\frac{x}{L} - \frac{3}{4}\right)^2}$$

   (a) The ASCII data files `u_lin.dat` and `h_lin.dat` each contain one 129×512 array. These 2D arrays are the **linear** solution to the model. The rows store the data at the nodes and the columns store the data at different times. Load the ASCII data for the linear solution into your workspace.

   Create a figure with 3 subplots (1 column, 3 rows):

   - 1st row - initial condition of the linear solution $u(x,0)$ loaded from ASCII files.
   - 2nd row - initial condition of the linear solution $h(x,0)$ loaded from ASCII files.
   - 3rd row - user defined initial condition $\phi(x,0)$.

   Both the linear and non-linear model begin from the same initial state; this figure should be identical to the figure in Part 1b.

   (b) The MATLAB data file `soln.mat` contains three 129×512 arrays: `u_nlin`, `h_nlin` and `phi_nlin`. These 2D arrays are the **non-linear** solution to the model. The rows store the data at the nodes and the columns store the data at different times. Load the MATLAB data for the non-linear solution into your workspace.

   Create a figure with 3 subplots (1 column, 3 rows):

   - 1st row - initial condition of the non-linear solution $u(x,0)$ loaded from MATLAB file.
   - 2nd row - initial condition of the non-linear solution $h(x,0)$ loaded from MATLAB file.
   - 3rd row - initial condition of the non-linear solution $\phi(x,0)$ loaded from MATLAB file.

   Both the linear and non-linear model begin from the same initial state; this figure should be identical to the figure in Part 1a.

   (c) Create a figure with 10 subplots (5 rows, 2 columns). Plot the surface elevation h *vs.* x. The left column should be the linear solution; the right column should be the non-linear solution. The rows should plot the following times: row 1: $m = 1$, row 2: $m = 128$, row 3: $m = 256$, row 4: $m = 384$ and row 5: $m = 512$. Fully annotate all of the subplots accordingly.

(d) Visualize the solution using `getframe` to produce an animated figure (see below). Animate both the linear (ASCII data) and non-linear (MATLAB data) solutions. Lab 10 will be used to allow your TAs to help you visualize the solutions. **DO NOT INCLUDE** this code in your final submission.

```
figure
for m = 1:M
      %% plot commands
      MOV(m)=getframe(gcf);
end
```

2. **Creating the Model and Reproducing Solutions** The solutions for both the linear and non-linear system are given in the data files in Part 1 and you were required to plot and visualize them. You will now write MATLAB code to reproduce these solutions by calling the given functions (`nonlinear.m` and `gravity.m`) and creating and calling one of your own (`transport.m`). First, the main script `PP.m` must prompt the user to ascertain whether the linear or the non-linear solution is sought. Edit the script `PP.m` and prompt the user to select either the linear or non-linear solution. You must use a while-loop to force the user to enter either 0 for the linear solution or 1 for the non-linear solution. If the user selects the linear option, you must call the function `gravity.m` as well as create and call the function `transport.m` according to Section 2b. If the user selects the non-linear option, you must call the existing function `nonlinear.m` according to Figure 3. Edit `PP.m` in order to loop over time, given the first column of the variables in Part 1. Within the loop, call the functions according to the flow chart in Figure 3 and fill out the remaining 511 columns of the matrices `u`, `h`, `eta` and `phi`. (*N.B.* You should not alter the files `nonlinear.m` or `gravity.m`; altering these files may cause errors in your solution. For help using these functions, make the current working directory point to the directory where they are stored and type `help gravity` or `help nonlinear` at the command line.)

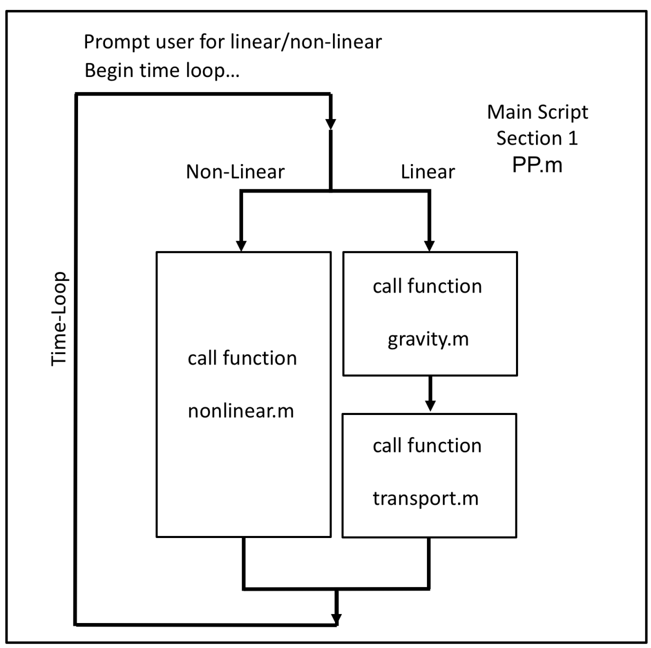

Figure 3: Flow chart for function calls in `PP.m` according to Sections 2a and 2b.

(a) **Constructing the Non-Linear Model** You must call the function `nonlinear.m` in order to produce a non-linear estimate of the variables h, η, u and ϕ at time-level $m+1$, given the values at the old time level m. The function should be called according to Figure 3. You will know your code works when it reproduces the results of Part 1.

(b) **Passive Tracer Transport** You must construct the function `transport.m` in order to estimate the transport of the passive tracer. The function should be constructed according to the equations below. Given the tracer values at the old time-level (ϕ^m), the tracer values at the new time-level (ϕ^{m+1}) are given by the following equations:

Row 1: left-boundary condition

$$-\phi_1^{m+1} + \phi_2^{m+1} = 0 \tag{1}$$

Rows 2 to N: interior nodes

$$\left[\frac{-\Delta t}{2\Delta x} u_n\right] \phi_{n-1}^{m+1} + \left[1 + \frac{\Delta t}{2\Delta x}\left(u_{n+1} - u_{n-1}\right)\right] \phi_n^{m+1} + \left[\frac{\Delta t}{2\Delta x} u_n\right] \phi_{n+1}^{m+1} = \phi_n^m \tag{2}$$

Row N+1: right-boundary condition

$$-\phi_N^{m+1} + \phi_{N+1}^{m+1} = 0 \tag{3}$$

where the advecting flow velocity u_n in Equation 2 is the old time-level (*i.e.*, $u_n = u_n^m$). You must create a function which takes as input the $N+1$ values of the tracer and flow speed at the time m (that is, ϕ_n^m and u_n^m), the number of cells N, as well as the time-step Δt and cell size Δx. The output of the function should be the $N+1$ values of the tracer at the new time $m+1$ (that is, ϕ_n^{m+1}). Within the function, the code must determine that the input variables are all of the correct size. If the input variables passed to the function are not the correct size, you must call the MATLAB function `error()`, which terminates the program. Once the input variables are tested for size, the function body must set up and solve the system $A\mathbf{x} = \mathbf{b}$ according to Equations 1 through 3.

INPUT ARGUMENTS:

- ϕ^m: $(N+1) \times 1$
- u^m: $(N+1) \times 1$
- N, Δt and Δx: 1×1

OUTPUT ARGUMENTS:

- ϕ^{m+1}: $(N+1) \times 1$

(c) **Constructing the Linear Model** You must now modify the code `PP.m` in order to call your new function `transport.m` and the given function `gravity.m` according to Figure 3 so that you can simulate both a linear and non-linear model of the tank. You will know your code works when it reproduces the results of Part 1. Create a figure which shows the tracer concentration as a function of x at the following times, on a single axis: $m = 1$, $m = 128$ and $m = 184$. Properly annotate your figure, including a legend.

3. **Analysis** Analyze the conservation properties of the linear and non-linear numerical solutions. Explore the development of shocks in a shallow fluid.

(a) **Conservation Laws** The following quantities are "conserved" in the sense that their values should not change with time, according to the original equations:

$$\text{Available Energy:} \qquad AE = APE + KE = \rho w \int_0^L \left(\frac{1}{2} h u^2 + \frac{1}{2} g \eta^2\right) dx \tag{4}$$

$$\text{Fluid Mass:} \qquad M_F = \rho w \int_0^L h dx \tag{5}$$

where the kinetic energy and available potential energy are given by

$$\text{Kinetic Energy:} \qquad KE = \rho w \int_0^L \left(\frac{1}{2} h u^2\right) dx$$

$$\text{Available Potential Energy:} \qquad APE = \rho w \int_0^L \left(\frac{1}{2} g \eta^2\right) dx$$

The integrals above can be estimated with the "Trapezoidal Rule":

$$\int_0^L f(x) dx \approx \sum_{n=1}^{N} \left(\frac{f_n + f_{n+1}}{2}\right) \Delta x \tag{6}$$

- Construct a function called `trapezoidal.m` which takes arrays of size $(N+1)\times 1$ and returns a single value which is the approximate integral of the array according to Equation 6. The input arguments should be the parameter N, the $(N+1)\times 1$ array itself, and the node increment Δx; the output argument should be the integral (*i.e.*, the left-hand-side of Equation 6). Within the function, the code must determine that the input variables are all of the correct size. If the input variables passed to the function are not the correct size, you must call the MATLAB function `error()`, which terminates the program.

  INPUT ARGUMENTS:
  - f_n: $(N+1)\times 1$
  - N and Δx: 1×1

  OUTPUT ARGUMENTS:
  - l.h.s., Eq. 6: 1×1

- Modify `PP.m` and call your function `trapezoidal.m` to compute the various integrals above. Create a figure showing the fluid mass and total available energy as a function of time for both the linear and non-linear solutions. In order for these quantities to appear on the same axes, plot them as a percentage of their values at $t = 0$. Fully annotate your plot. Comment on the conservation properties of our numerical methods. Which model (linear or non-linear) has better mass-conservation properties? Is the total available energy conserved in either the linear or the non-linear model? Which model (linear or non-linear) has better energy-conservation properties?
- Create another figure showing the total available energy, along with the kinetic energy and available potential energy as a function of time for both the linear and non-linear solutions. Use a mass-spring analogy (see Figure 4) to interpret/explain your results.

(b) **Non-linear Shock Development**

- Create a figure with 10 subplots (5 rows, 2 columns). Plot the surface elevation h *vs.* x. The left column should be the linear solution; the right column should be the non-linear solution. The rows should plot the following times: row 1: $m = 1$, row 2: $m = 128$, row 3: $m = 256$, row 4: $m = 384$ and row 5: $m = 512$. Fully annotate all of the subplots accordingly. This figure should look the same as in Part 1. In a fluid, a "shock" is characterized by discontinuous, or rapid changes in flow properties such as velocity, pressure, temperature, *etc.* Comment on the development of shocks in a linear *vs.* a non-linear shallow fluid. Does the linear model produce a shock wave in the fluid? What about the non-linear model?

References

[1] Farhadzadeh, A., SUNY Stonybrook, Dept. of Civil Engineering, Coastal and Hydraulic Engineering Research Laboratory
(http://www.stonybrook.edu/commcms/civileng/research/_wavecurrentflumespecification.php)

[2] Durran, D. R. (1991): *Numerical Methods for Wave Equations in Geophysical Fluid Dynamics*. Springer, NY

[3] Sainchera, S. and Banerjeea, J. (2015): Design of a numerical wave tank and wave flume for low steepness waves in deep and intermediate water. Proceedings of 8th International Conference on Asian and Pacific Coast.

[4] Dullemond, C. P., (2008): Lecture on Numerical Fluid Dynamics, Chapter 4
(http://www2.mpia-hd.mpg.de/~dullemon/lectures/fluiddynamics08/)

[5] Fishbane, P.M., Gasiorowicz, S., and S.T. Thornton, (1996) *Physics for Scientists and Engineers*, 2d ed. (Extended), Upper Saddle River, New Jersey: Prentice Hall.

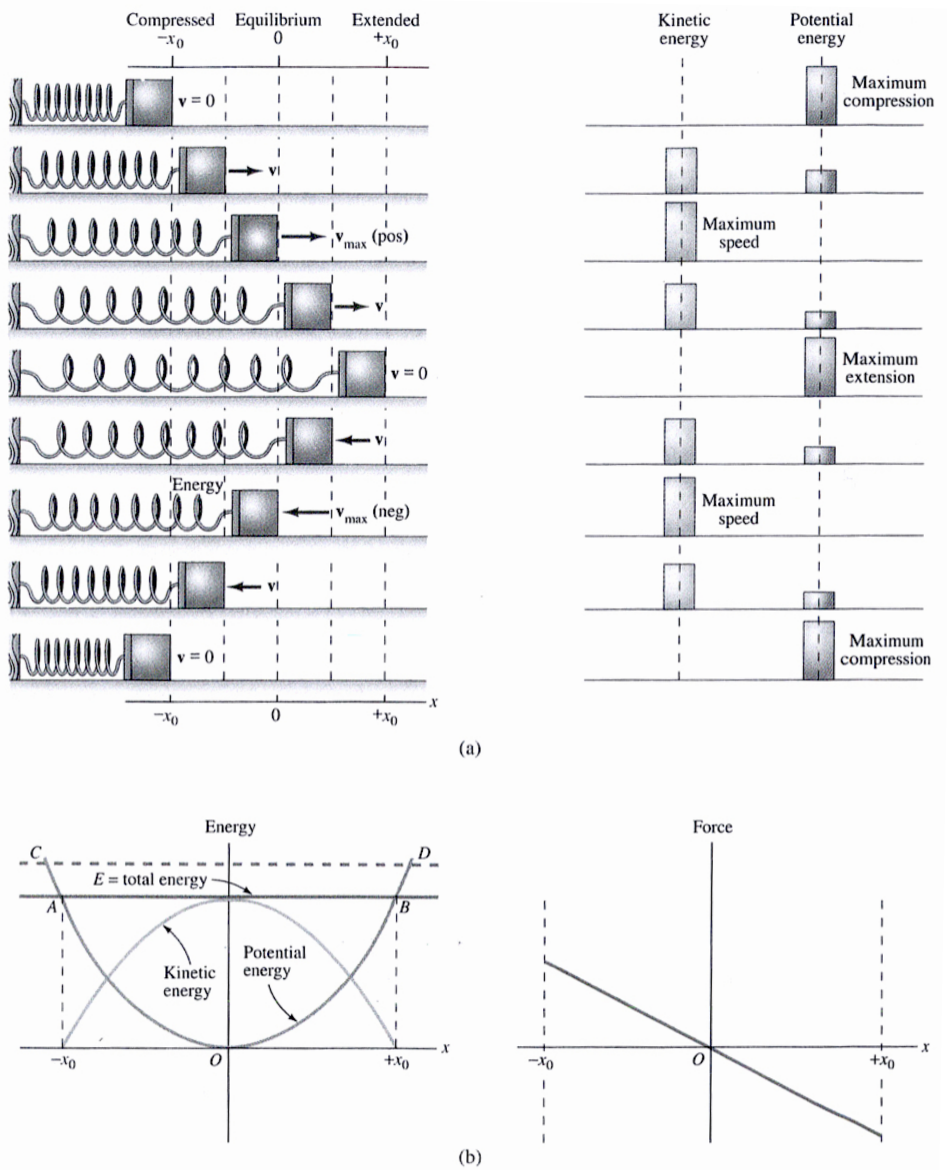

FIGURE 7–7 A mass on the end of a spring; equilibrium is $x = 0$. (a) At the top, the spring is compressed by an amount x_0. During one period (or cycle), the mass moves through $x = 0$ to an extension $x = x_0$ and back to $x = -x_0$. As the mass moves, the kinetic and potential energies associated with the mass-spring system change: One increases as the other decreases. (b) A plot of the force of a spring on a mass attached to its end, together with an energy diagram for the mass. The potential energy $U = \frac{1}{2}kx^2$ is drawn. The straight line represents the constant total energy, E; the kinetic energy, K, is the difference between E and U. The curve of K is determined purely by the conservation of energy. Points $+x_0$ and $-x_0$ are turning points for the motion, where the kinetic energy and therefore the speed are zero. For x outside the range $-x_0 < x < +x_0$, the kinetic energy would be negative, so the mass cannot go outside this range.

Figure 4: After Fishbane, *et al.*[5] Figure 7-7.