Investigation of Dam Break Problem through solution of 1D-Shallow water equations using Fourth order Runge-Kutta Method, McCormack's Method and Gudonov's Method

Sagar Bhatt
Person Number: 50170651
Department of Mechanical and Aerospace Engineering,
University at Buffalo

1. INTRODUCTION

1.1 Problem:

We are interested in investigating the dam-break problem. Initially, the water on both sides of the dam is at rest and the dam is located at x = 50m with L = 100 m. The water depth upstream (x<50m) is H1 = 10 m whereas the water depth downstream (x>50) is H2 = 1 m. At t=0 the dam breaks the water upstream of the dam flows downstream. To this end we can solve the 1-D shallow-water equations:

$$\frac{\partial \vec{Q}}{\partial t} + \frac{\partial \vec{E}(\overrightarrow{Q})}{\partial x} = \vec{B}$$

$$\vec{Q} = \begin{pmatrix} H \\ m \end{pmatrix}, \vec{E}(\overrightarrow{Q}) = \begin{pmatrix} m \\ mu + \frac{1}{2} g H^2 \end{pmatrix}, \vec{B} = \begin{pmatrix} 0 \\ -gH \frac{\partial b}{\partial x} \end{pmatrix}$$

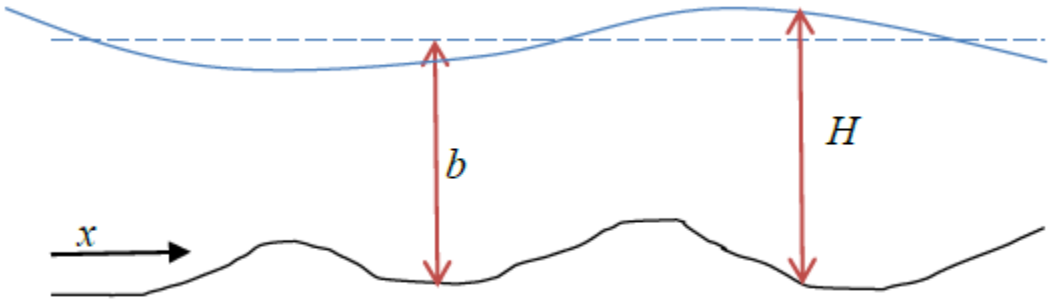

Figure 1:Problem Statement

H is the total depth, b is the elevation of the basin of the water depth if the surface is at rest, m=Hu is the
momentum, and g is the gravity (See Figure). Solve the equation with mesh size Δx=L/100 and compare it with a finer mesh solution. Solve the equations until t=4 s and compare your results against analytic/benchmark solutions (e.g., see Vincent et al. (2001) Journal of Hydraulic Engineering, Numerical modelling of bore propagation and run-up on sloping beaches using a MacCormack TVD scheme). Investigate the effect of Δt on the solution of each scheme. At least use one 2nd (or higher) order scheme, e.g., Lax-Wendroff, McCormack (40 bonus points if you use ENO 3rd order), one TVD scheme, e.g., R-K with TVD, FCT, Godunov with MUSCL, and one 1st order scheme, e.g., Godunov, to compute it. Compare the different methods. Do it component by component (as if it were scalar equation).

1.2Background Information:

1.2.1 McCormack's Method:

This is a multilevel method is applied to yield the following finite difference equations:

$$u_i^* = u_i^n - \frac{\Delta t}{\Delta x}(E_{i+1}^n - E_i^n)$$

And

$$u_i^{n+1} = \frac{1}{2}\left(u_i^n + u_i^* - \frac{\Delta t}{\Delta x}(E_i^n - E_{i-1}^n)\right)$$

The stability this method requires is $\left|u_{max}\frac{\Delta t}{\Delta x}\right| \leq 1$.

The solution from this method is much 'well behaved' compared to schemes like Lax-Wendroff. This is due to the splitting procedure and corresponding forward, backward differencing used to approximate spatial derivative. The solution degrades in method as courant number decreases from maximum allowable value of 1 and is best obtained when it is equal to 1.

1.2.2 Runge Kutta Method with TVD:

Here, a fourth order Runge Kutta method is used whose formulation is as follows:

$$u_i^{(1)} = u_i^n$$

$$u_i^{(2)} = u_i^n - \frac{\Delta t}{4}\left(\frac{\partial E}{\partial x}\right)_i^{(1)}$$

$$u_i^{(3)} = u_i^n - \frac{\Delta t}{3}\left(\frac{\partial E}{\partial x}\right)_i^{(2)}$$

$$u_i^{(4)} = u_i^n - \frac{\Delta t}{2}\left(\frac{\partial E}{\partial x}\right)_i^{(3)}$$

$$u_i^{n+1} = u_i^n - \Delta t\left(\frac{\partial E}{\partial x}\right)_i^{(4)}$$

This scheme develops some oscillations due to dispersion effects. To reduce these oscillations, a TVD flux limiter was added. This flux limiter has a damping effect that reduces the oscillations and gives a smooth output as result. To achieve this, the following step was added:

$$u_i^{n+1} = u_i^{n+1} - \frac{\Delta t}{2\Delta x}\left(\psi^n_{i+\frac{1}{2}} - \psi^n_{i-\frac{1}{2}}\right)$$

Where ψ is called Roe-Sweby Upwind TVD limiter and is given by:

$$\psi_{\,i+\frac{1}{2}} = \left[\frac{G_i}{2}\left(\left|\propto_{i+\frac{1}{2}}\right| + \frac{\Delta t}{\Delta x}\propto^2_{i+\frac{1}{2}}\right) - \left|\propto_{i+\frac{1}{2}}\right|\right](u_{i+1} - u_i)$$

$$\psi_{\,i-\frac{1}{2}} = \left[\frac{G_{i-1}}{2}\left(\left|\propto_{i-\frac{1}{2}}\right| + \frac{\Delta t}{\Delta x}\propto^2_{i-\frac{1}{2}}\right) - \left|\propto_{i-\frac{1}{2}}\right|\right](u_i - u_{i-1})$$

Here, G is given by:

$$G_i = \frac{\theta + |\theta|}{1+\theta}$$

Now,

$$\theta_{i+\frac{1}{2}} = \frac{u_{i+1+\sigma} - u_{i+\sigma}}{(u_{i+1} - u_i)}$$

And,

$$\theta_{i-\frac{1}{2}} = \frac{u_{i+\sigma} - u_{i-1+\sigma}}{(u_i - u_{i-1})}$$

Here, $\sigma = Sgn\left(\propto_{i+\frac{1}{2}}\right)$

And, $\propto_{i+\frac{1}{2}} = \begin{cases} u_i^n & if\ U^n_{i+1} - U^n_i = 0 \\ \frac{E^n_{i+1} - E^n_i}{U^n_{i+1} - U^n_i} & otherwise \end{cases}$

1.2.3 **Gudunov's Method with MUSCL:**

Gudunov's method is a finite volume method that solves the Riemann problems locally at each inter cell boundary. Monotonicity in this method results in a solution without

oscillations but being a low order scheme (first order), this method is highly diffusive. The formulation for this method is given by:

$$u^{n+1} = u^n - \frac{\Delta t}{\Delta x} E^n_{i+\frac{1}{2}} - E^n_{i-\frac{1}{2}}$$

Where, $E^n_{i+\frac{1}{2}}$ is the numerical flux defined by

$$E^n_{i+\frac{1}{2}} = E\left(u_i\ , u_{i+1}\right)$$

Hence,

$$E(Q_i, Q_{i+1}) = \frac{1}{2}\left(E\left(u_j\right) + E\left(u_{j+1}\right)\right) - |\propto|(u_{i+1} - u_i)$$

Where $\propto$ is max(eigen value).

2. METHOD OF SOLUTION:

The problem was defined by 1-D shallow water equations given b:

$$\frac{\partial \vec{Q}}{\partial t} + \frac{\partial \vec{E}(\overrightarrow{Q})}{\partial x} = \vec{B}$$

$$\vec{Q} = \binom{H}{m}, \vec{E}(\overrightarrow{Q}) = \begin{pmatrix} m \\ mu + \frac{1}{2} gH^2 \end{pmatrix}, \vec{B} = \begin{pmatrix} 0 \\ -gH \frac{\partial b}{\partial x} \end{pmatrix}$$

On RHS, we have $\vec{B} = \begin{pmatrix} 0 \\ -gH \frac{\partial b}{\partial x} \end{pmatrix}$. Here, b is the elevation of the basin of the water depth if the surface is at rest. Assuming, water basin does not change, $\frac{\partial b}{\partial x}$ = 0. Hence our equation takes the form,

$$\frac{\partial \vec{Q}}{\partial t} + \frac{\partial \vec{E}(\overrightarrow{Q})}{\partial x} = 0$$

Thus the equation takes form of hyperbolic equation where we can apply the given schemes in order to solve the problem.

2.2 Analytical Solution:

First the analytical solution was evaluated,to get an idea of what the solution looks like and what to expect from the results. For this purpose, the analytical solution was plotted from the expressions and methodology given in section 4.1 of ref [2]. The results from this formulation would serve as a means to validate our results. The code for this solution can be found in Appendix 1.

2.3 McCormack's Method:

In this method, we solve the given equations simultaneously using McCormack's method by first using H and Q in place of u and E and then Q and E in place of u and in the formulation for McCormack's method. This method is highly dispersive hence artificial dissipation was added in form of MatLab function 'smooth()'. The program for the method can be found in Appendix 2.

2.4 Runge Kutta Method with TVD:

In this method, standard fourth order Runge-Kutta method is applied on both equations of 1-D shallow water equations. Then a Roe-Sweby Upwind TVD Limiter is applied in order to dampen the oscillations. The limiter is evaluated by first finding out $\propto$, then using the $\propto$ to find out the signum function σ. Then we proceed to find the θ and G. This is used to find the limiters as follows:

$$\psi_{i+\frac{1}{2}} = \left[\frac{G_i}{2}\left(\left|\propto_{i+\frac{1}{2}}\right| + \frac{\Delta t}{\Delta x}\propto^2_{i+\frac{1}{2}}\right) - \left|\propto_{i+\frac{1}{2}}\right|\right](u_{i+1} - u_i)$$

$$\psi_{i-\frac{1}{2}} = \left[\frac{G_{i-1}}{2}\left(\left|\propto_{i-\frac{1}{2}}\right| + \frac{\Delta t}{\Delta x}\propto^2_{i-\frac{1}{2}}\right) - \left|\propto_{i-\frac{1}{2}}\right|\right](u_i - u_{i-1})$$

The program for the method can be found in Appendix 3.

2.5 Gudonov's Method with MUSCL:

We have used Gudonov's method for a piecewise solution of the Riemann problems. For this purpose, we have used Lax-Friedrichs flux in the method. The problem was evaluated at different grid sizes and time steps to evaluate the grid

sensitivity and dependence on Δt. The program for the method can be found in Appendix 4.

3. DISCUSSION OF RESULTS:

3.1 Analytical Solution:

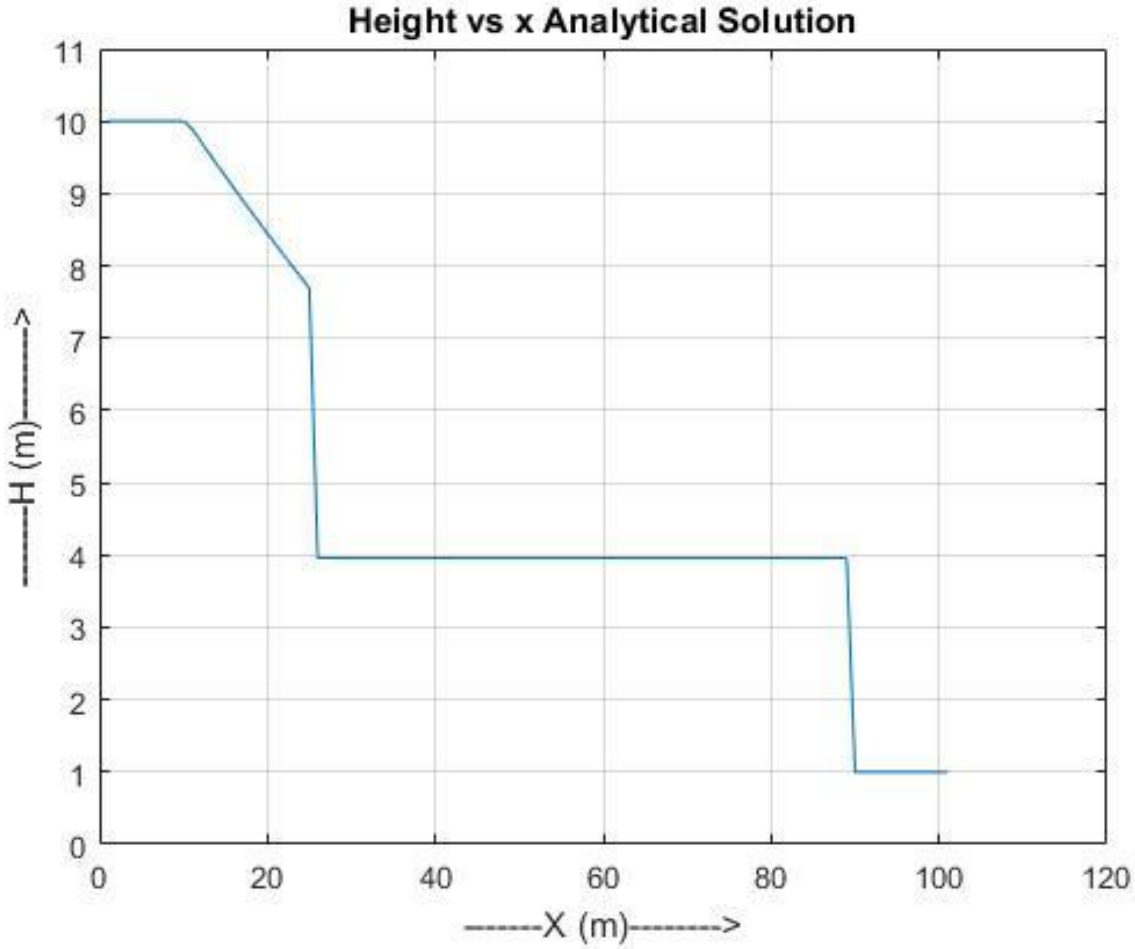

Figure 2 Analytical Solution

This figure represents the analytical solution of the problem as formulated in ref. [2]. This will serve as base result for our problem against which we will compare our solution.

3.2 McCormack's Method:

The following plot describes our solution using McCormack's scheme for a Δx=1 and a Δt=0.01:

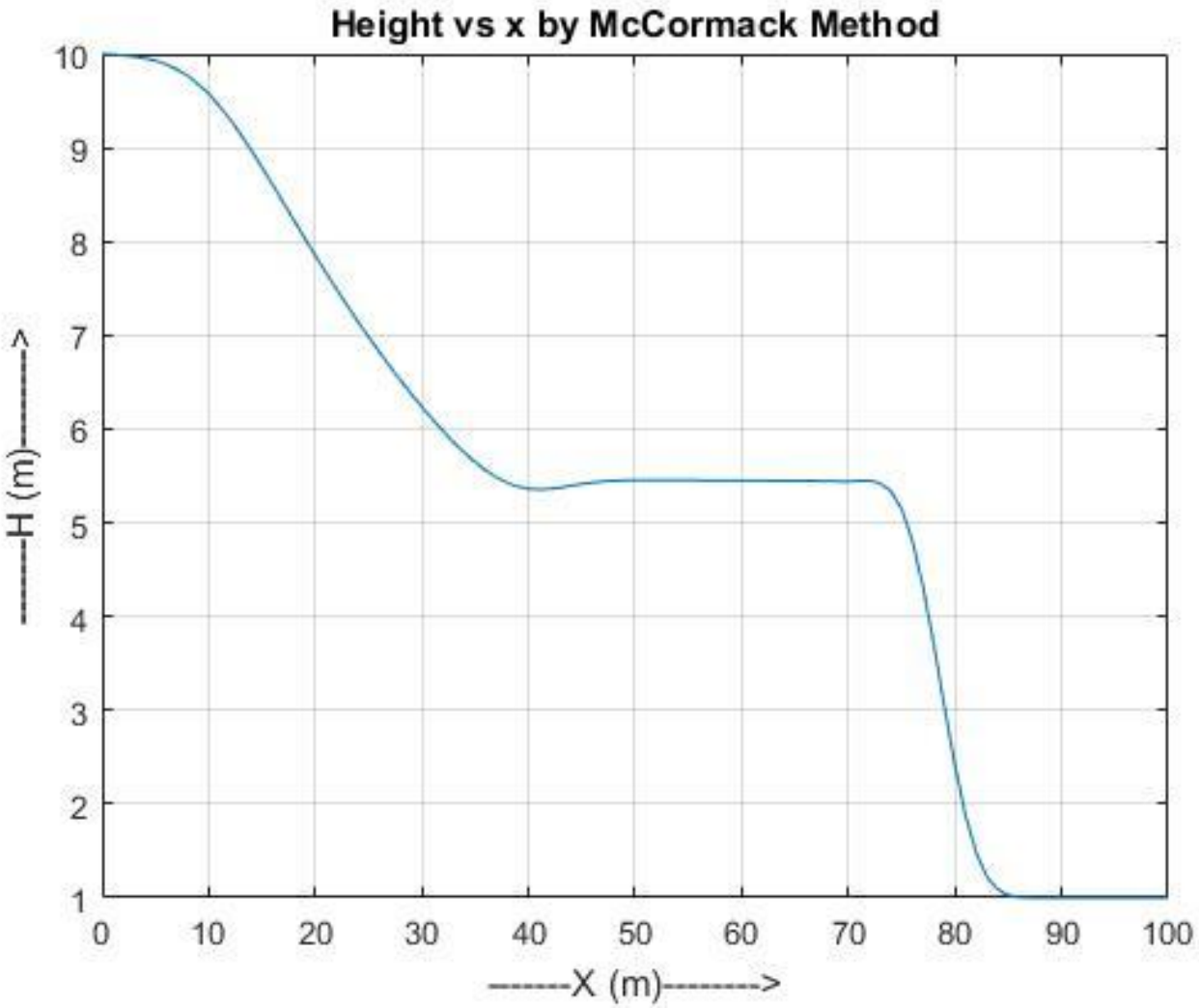

Figure 3 Solution from McCormack's Method for $\Delta x=1$ and $\Delta t=0.01$

As we can see from fig. 3 the method has some dispersion although barely noticeable after induction of artificial dissipation. The plot is consistent with the analytical solution shown in fig. 2.

The following plot describes our solution using McCormack's scheme for a $\Delta x=1$ and a $\Delta t=0.001$:

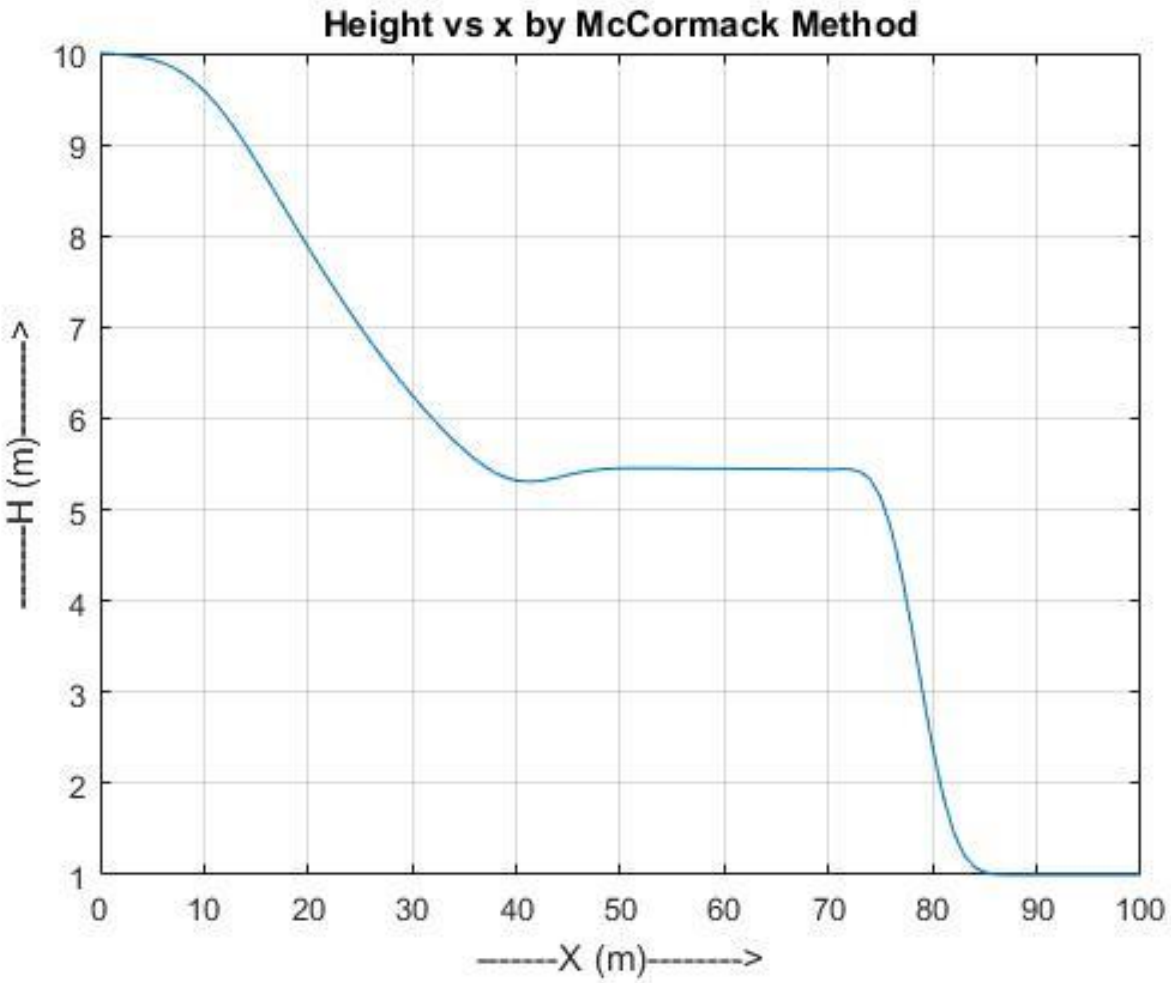

Figure 4 Solution from McCormack's method for $\Delta x=1$ and $\Delta t=0.001$

From fig. 4 we can see that as we reduced the time step, the errors in the plot shown in fig. 4 is more pronounced than fig. 2. This difference, although barely noticeable, shows that as Δt is reduced, the result is impacted negatively. This due to the fact that courant number is reducing. And we know that as courant number reduces, the result degenerates.

The following plot describes our solution using McCormack's scheme for a $\Delta x=0.5$ and a $\Delta t=0.01$:

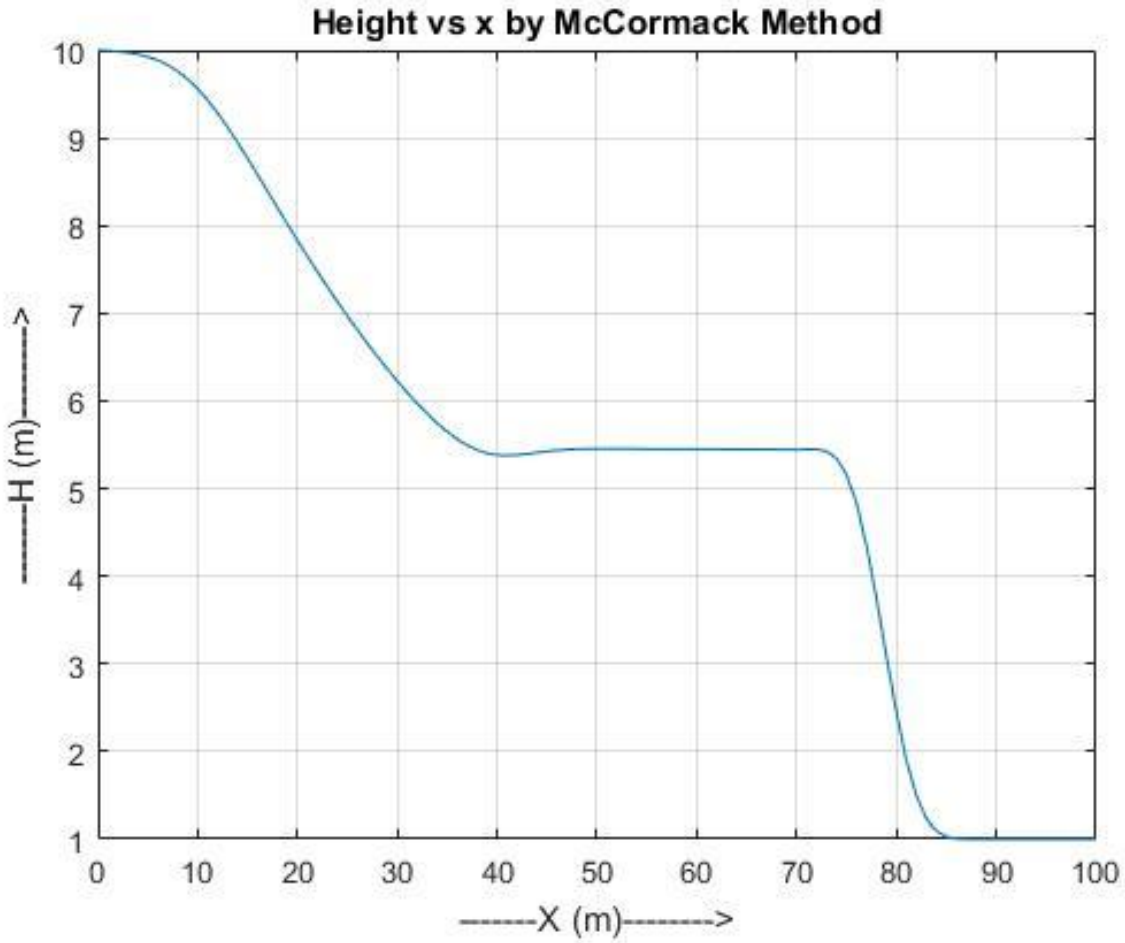

Figure 5 Solution from McCormack's method for $\Delta x=0.5$ and $\Delta t=0.01$

From fig. 5, we can see that as we reduce the grid size the accuracy of the method increases and the result starts resembling the analytical solution more accurately.

3.3**Runge Kutta Method with TVD:**

The following plot describes our solution using Runge-Kutta method for a $\Delta x=1$ and a $\Delta t=0.001$:

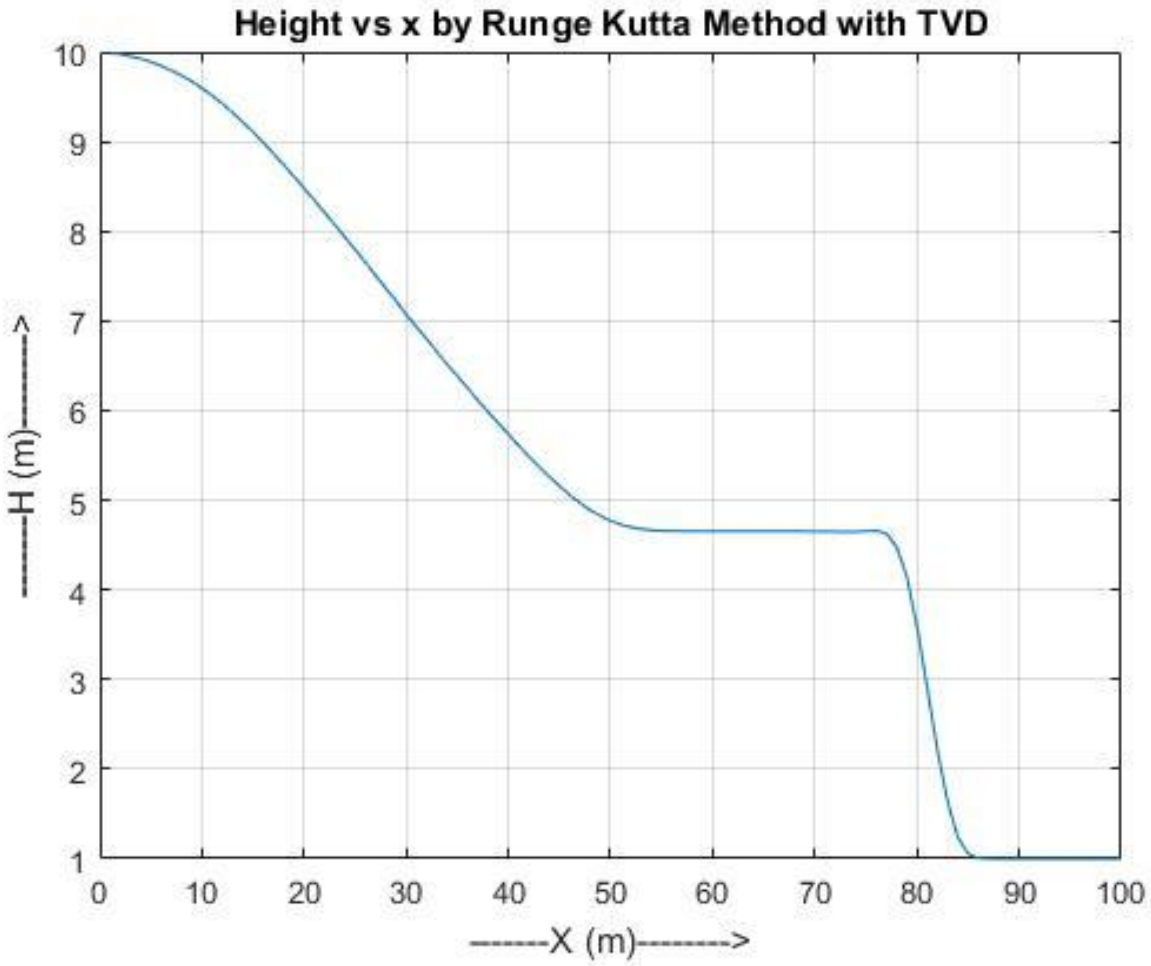

Figure 6 Solution from Runge-Kutta method for $\Delta x=1$ and $\Delta t=0.001$

From fig. 6, we can see that Runge-Kutta Method provides a solution which is consistent with the analytical solution shown in fig. 2. Als, we can see that adding the Roe-Sweby Upwind TVD Limiter to dampen any oscillations has almost completely removed any dispersion that mght be present.

The following plot describes our solution using Runge-Kutta method for a $\Delta x=1$ and a $\Delta t=0.0001$:

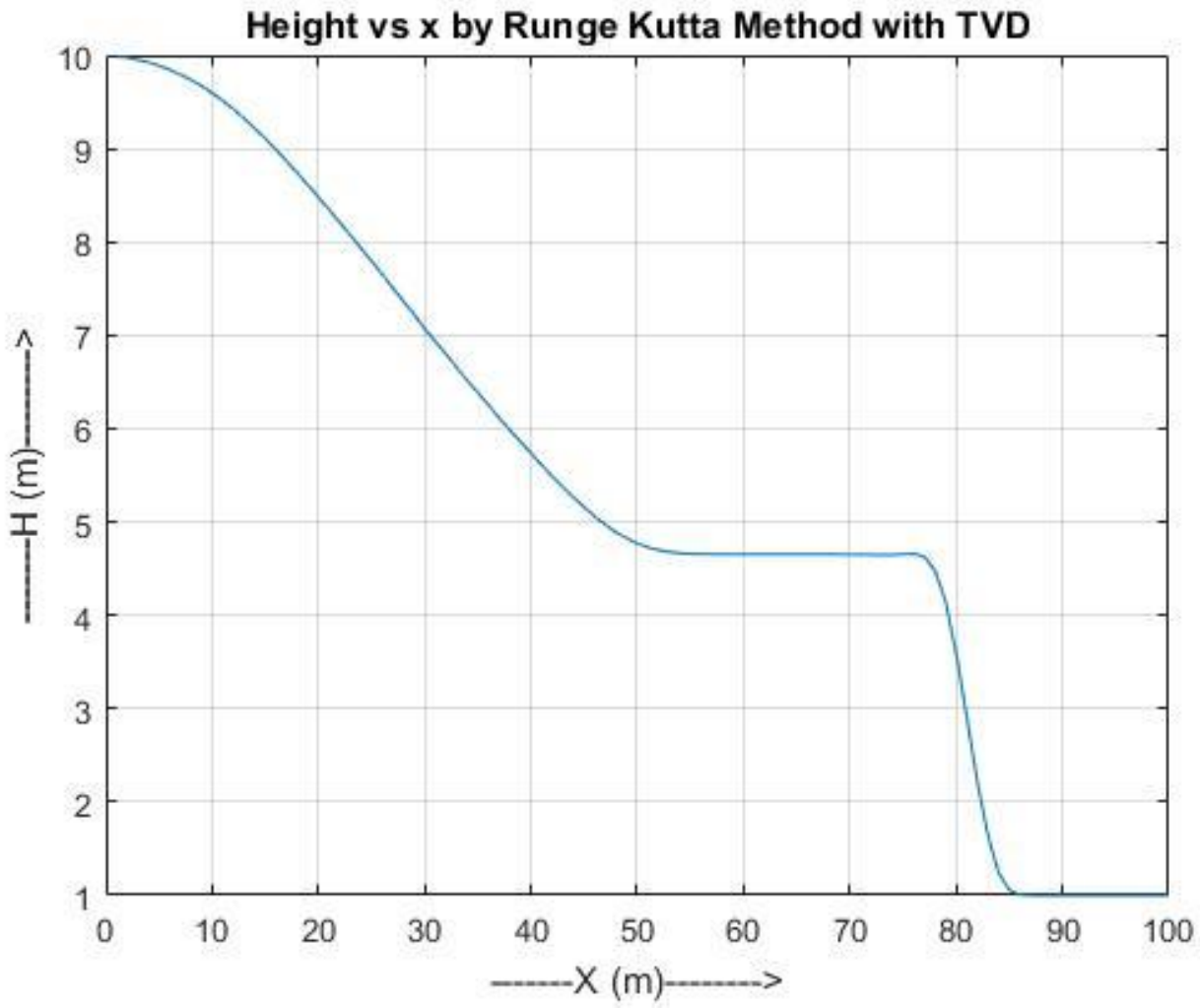

Figure 7 Solution from Runge-Kutta method for $\Delta x=1$ and $\Delta t=0.0001$

Fom fig. 7 wecan see that reducing time-step has hardly had any effect on the solution when we compare the results with fig. 6. This is due to the fact that Runge-Kutta method is unconditionally stable and the change in courant number does not affect results from Runge-Kutta method. Hence our solution is in agreement with already established theory.

The following plot describes our solution using Runge-Kutta method for a $\Delta x=0.8$ and a $\Delta t=0.001$:

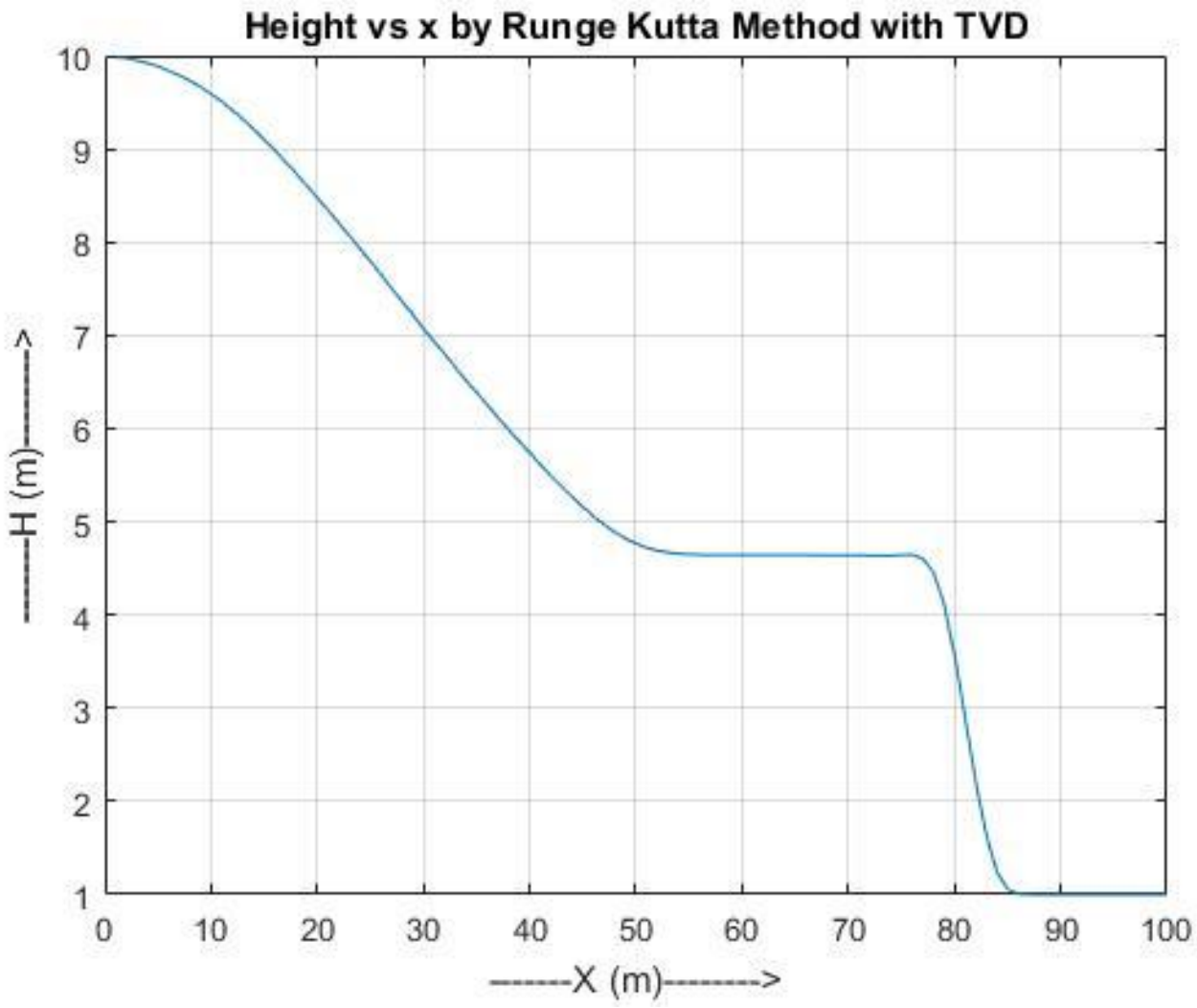

Figure 8: Solution from Runge-Kutta method for $\Delta x=0.8$ and $\Delta t=0.001$

From fig. 8 we can see that, as before, reducing the grid size has increased the accuracy of the scheme. Although barely noticeable, the result is much more refined than the results presented in fig. 6 and fig. 7.

3.4 **Gudunov's Method with MUSCL:**

The following plot describes our solution using Gudunov's method for a $\Delta x=1$ and a $\Delta t=0.1$:

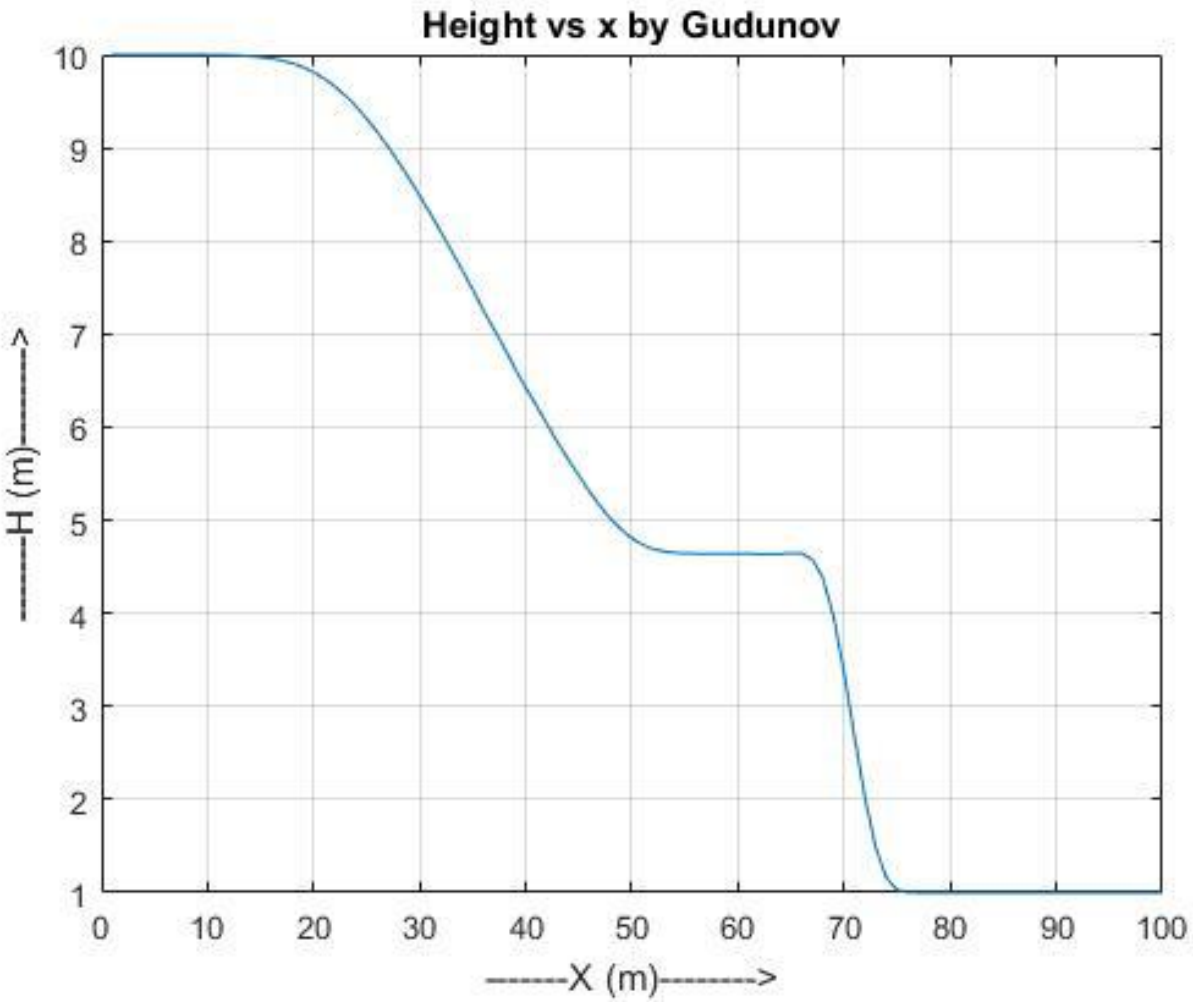

Figure 9: Solution from Gudunov's method for $\Delta x=1$ and $\Delta t=0.1$

The plot shown in fig. 9 shows that the results obtained through Gudunov's method is consistent with the results shown in fig 2 by analytical solution. We can also observe that being a lower order method, the plot differs slightly from the higher order methods presented previously.

The following plot describes our solution using Gudunov's method for a $\Delta x=1$ and a $\Delta t=0.01$:

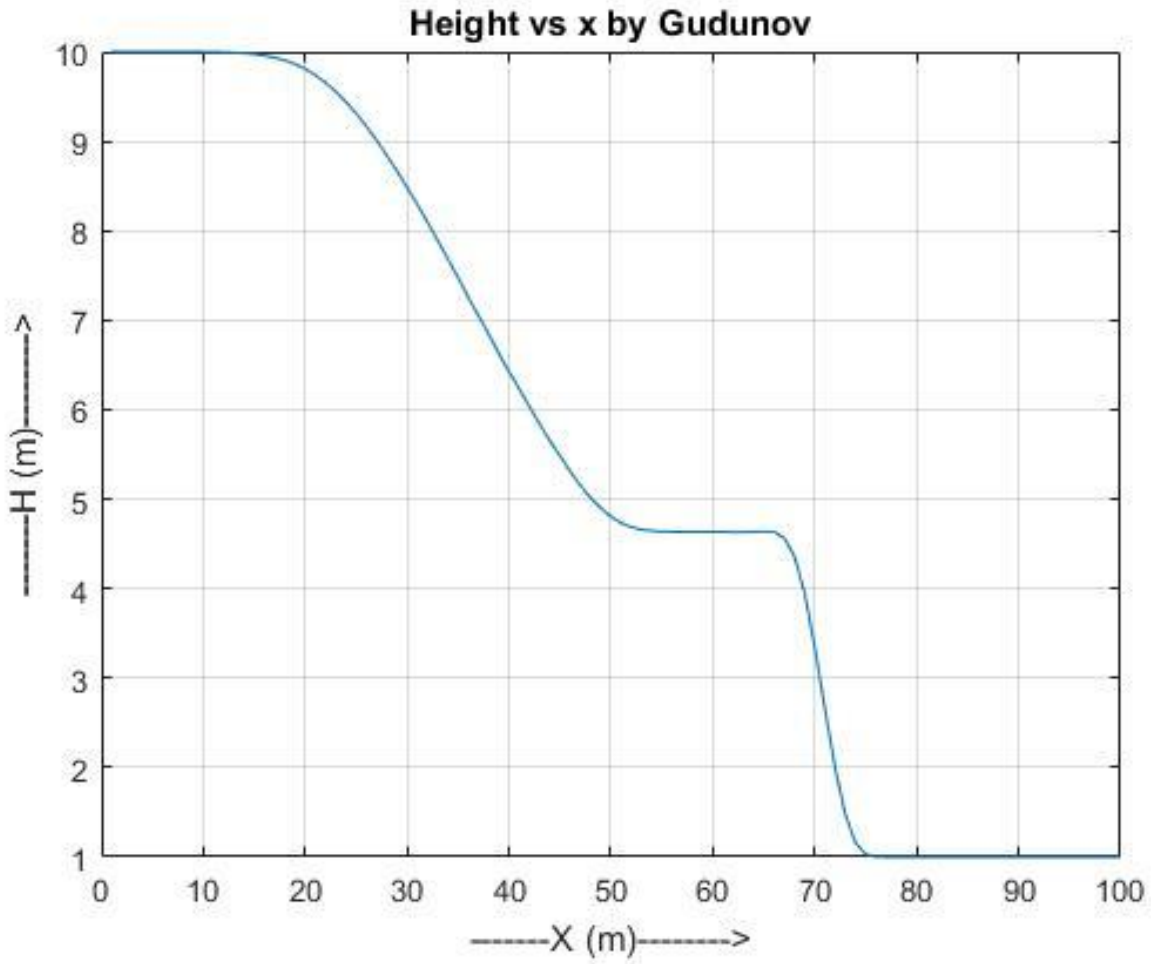

Figure 10: Solution for Gugdunov's method for $\Delta x=1$ and $\Delta t=0.01$

We observe here, as before, that reducing the time-step has very little effect on our result. The effect, however little, has negative effect on our result as it brings out the errors in our result in a much pronounced fashion. This is in agreement with above stated fact that as courant number reduces, the result degenerates.

The following plot describes our solution using Gudunov's method for a $\Delta x=0.5$ and a $\Delta t=0.1$:

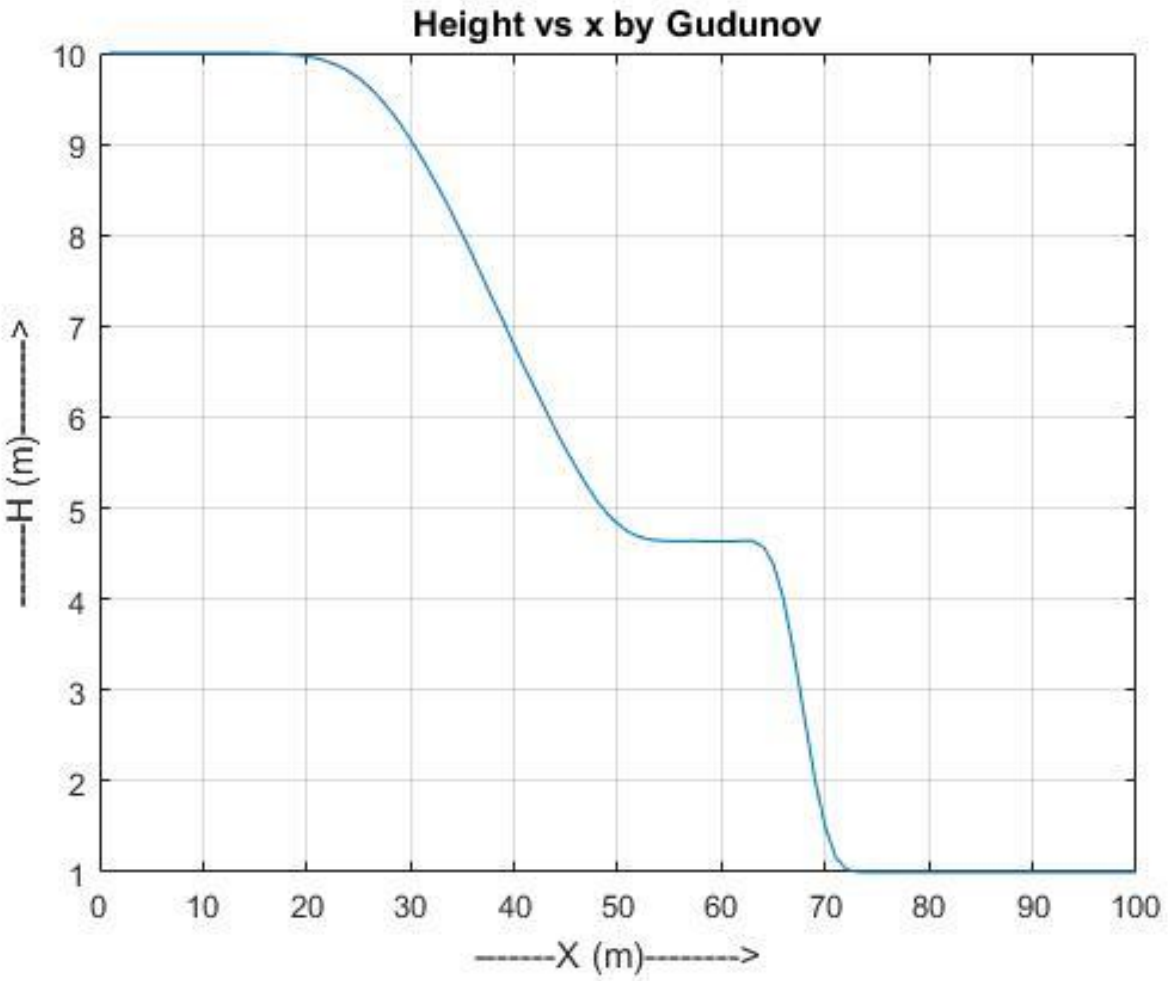

Figure 11: Solution for Gudunov's method for $\Delta x=0.5$ and $\Delta t=0.1$

From fig. 11, we can see that as the grid size is reduced, our result becomes more refined compared to fig 10 and fig. 11. This Result was expected and hence goes on to prove the validity of the result.

4 SUMMARY AND CONCLUSION

The Dam breaking problem was solved using the given 1-D Shallow-water equations. To solve the problem, McCormack's method, Gudunov's method and Runge Kutta method with TVD were used. The methods were discussed intensively and then implemented to solve the problem. The results were then compared with the analytical solution proposed by Vincent et al. (ref [2]). The results were then evaluated for grid sensitivity. After comparing the results, it was found that although the results were grid sensitive (as should be while implementing a finer grid), the effect was very minute and almost unnoticeable. This was expected as the methods were higher order scheme except Gudunov's method which is a low order scheme.

The effect of Δt on the results was also explored for all the methods. It was found that McCormack's method and Gudunov's method were very slightly sensitive to change in Δt. The result, as expected, degenerated with decrease in Δt. This was because decrease in Δt results in a reduced courant number, and we know that the best result is obtained at the maximum possible courant number and thereby it degenerates as courant number reduces.

The change in Δt did not, however, effect Runge-Kutta method as this method is highly stable and does not depend a lot on the courant number. Hence, a change in Δt does not affect the result much.

Finally, all the methods were consistent with the analytical solution given in Vincent, S., Caltagirone, J. P., & Bonneton, P. (2001). Numerical modelling of bore propagation and run-up on sloping beaches using a MacCormack TVD scheme. Journal of hydraulic research, 39(1), 41-49.

5 REFERENCES

1. Hoffmann, K.A. & Chiang S.T., 2000, Computational Fluid Dynamics Volume 1, Fourth Edition, Engineering Education Systems, Kansas, USA.

2. Vincent, S., Caltagirone, J. P., & Bonneton, P. (2001). Numerical modelling of bore propagation and run-up on sloping beaches using a MacCormack TVD scheme. Journal of hydraulic research, 39(1), 41-49.

3. Class notes for course MAE 542 by Dr. Iman Borazjani, Department of Mechanical and Aerospace Engineering, University at Buffalo, the State University at New York.

6. APPENDIX

Appendix 1:

```
%Analytical Solution

L = 100;
dx = 1;
g=9.81;
t = 4;
n=100;
H1=10;
H2 = 1;
Hg = 3.962;
x1=-sqrt(g*H1)*t;
x2= -sqrt(g*Hg)*t;
```

```
x3=g*t;

for i=1:n
    if (i-L/2)<=x1
        H(i) = H1;
        u(i) = 0;
    end
    if ((i-L/2)<=x2 & (i-L/2)>x1)
        p=(2*sqrt(g*H1)/3)-(i-L/2)/(3*t);
        H(i) = p^2/g;
    end
    if ((i-L/2)<=x3 & (i-L/2)>x2)
        H(i) = Hg;

    end

    if ((i-L/2)> x3)

        H(i) = H2;

    end
end
plot(H);
ylim([0 11]);
grid on;
xlabel('-------X (m)-------->');
ylabel('-------H (m)-------->');
title('Height vs x Analytical Solution');
```

Appendix 2:

```
%McCormack's Method

clear all;
clc;

n=100;
for i=1:n
   if i<=50
        H_old(i)=10;
        H(i)=10;
        H1(i)=0;
    else
        H_old(i)=1;
        H(i)=1;
        H1(i)=0;
    end
        v_old(i)=0;
        v(i)=0;
        v1(i)=0;
        Q(i)=0;
        Q1(i)=0;
        Q_old(i)=0;
        E_old(i)=0;
        E1(i)=0;
end
L=100;
```

```
dx=1;
dt=0.1;
k=(dt/dx);
g=9.81;

for t=0:dt:4
    for i=2:n-1
        Q(i)=(H(i)*v(i));
        Q_old(i)=(H_old(i)*v_old(i));
        E_old(i)=(H_old(i)*(v_old(i)^2))+(0.5*g*(H_old(i)^2));
        E1(i)=(H1(i)*(v1(i)^2))+(0.5*g*(H1(i)^2));

        H1(i)=H_old(i)-k*(Q_old(i+1)-Q_old(i));
        Q1(i)=Q_old(i)-k*(E_old(i+1)-E_old(i));
        H(i)=0.5*(H_old(i)+H1(i)-k*(Q1(i)-Q1(i-1)));
        Q(i)=0.5*(Q_old(i)+Q1(i)-k*(E1(i)-E1(i-1)));
    end
    H(1)=H(2);
    H(n)=H(n-1);
    Q(1)=Q(2);
    Q(n)=Q(n-1);
    H1(1)=H1(2);
    H1(n)=H1(n-1);
    Q1(1)=Q1(2);
    Q1(n)=Q1(n-1);
    v=Q./H;
    v_old=Q_old./H_old;
    H_old=H;
    Q_old=Q;

end


plot(H);
grid on;
xlabel('-------X (m)-------->');
ylabel('-------H (m)-------->');
title('Height vs x by McCormack Method');
```

Appendix 3:

```
% Runge-Kutta Method

clear all;
clc;
n=100;
for i=1:n
   if i<=50
        H_old(i)=10;
        H(i)=10;
        H1(i)=10;
        H2(i)=10;
        H3(i)=10;
        H4(i)=10;
    else
        H_old(i)=1;
        H(i)=1;
```

```
        H1(i)=1;
        H2(i)=1;
        H3(i)=1;
        H4(i)=1;
    end
        v_old(i)=0;
        v(i)=0;
        Q(i)=0;
        Q_old(i)=0;
        E_old(i)=0;
        Q1(i)=0;
        Q2(i)=0;
        Q3(i)=0;
        Q4(i)=0;
        E1(i)=0;
        E2(i)=0;
        E3(i)=0;
        E4(i)=0;
        a_old_H(i)=0;
        a_H(i)=0;
        sigma_old_H(i)=0;
        sigma_H(i)=0;
        sigma_old_Q(i)=0;
        sigma_Q(i)=0;
        theta_old_H(i)=0;
        theta_H(i)=0;
        theta_old_Q(i)=0;
        theta_Q(i)=0;
        G_old_H(i)=0;
        G_H(i)=0;
        G_old_Q=0;
        G_Q(i)=0;
        psi_old_H(i)=0;
        psi_H(i)=0;
        psi_old_Q(i)=0;
        psi_Q(i)=0;

end
L=100;
dx=1;
dt=0.001;
k=(dt/dx);
g=9.81;

for t=0:dt:4

    for i=2:n-2

        if H_old(i)-H_old(i-1)<=0.0005
            a_old_H(i)=H_old(i);
        else
            a_old_H(i)=((Q_old(i)-Q_old(i-1))/(H_old(i)-H_old(i-1)));
        end

       if H_old(i+1)-H_old(i)<=0.0005
            a_H(i)=H_old(i);
        else
            a_H(i)=((Q_old(i+1)-Q_old(i))/(H_old(i+1)-H_old(i)));
       end
```

```
        if Q_old(i)-Q_old(i-1)<=0.0005
             a_old_Q(i)=Q_old(i);
         else
             a_old_Q(i)=((E_old(i)-E_old(i-1))/(Q_old(i)-Q_old(i-1)));
        end

         if Q_old(i+1)-Q_old(i)<=0.0005
             a_Q(i)=Q_old(i);
         else
             a_Q(i)=((E_old(i+1)-E_old(i))/(Q_old(i+1)-Q_old(i)));
         end

             sigma_old_H(i)=sign(a_old_H(i));
             sigma_H(i)=sign(a_H(i));
             sigma_old_Q(i)=sign(a_old_Q(i));
             sigma_Q(i)=sign(a_Q(i));

         if H_old(i)-H_old(i-1)==0
             theta_old_H(i)=0;
         else
             theta_old_H(i)=(H_old(i+sigma_old_H(i))-H_old(i-
1+sigma_old_H(i)))/(H_old(i)-H_old(i-1));
         end
         if H_old(i+1)-H_old(i)==0
             theta_H(i)=0;
         else
             theta_H(i)=(H_old(i+1+sigma_H(i))-
H_old(i+sigma_H(i)))/(H_old(i+1)-H_old(i));
         end
         if Q_old(i)-Q_old(i-1)==0
             theta_old_Q(i)=0;
         else
             theta_old_Q(i)=(Q_old(i+sigma_old_Q(i))-Q_old(i-
1+sigma_old_Q(i)))/(Q_old(i)-Q_old(i-1));
         end
         if Q_old(i+1)-Q_old(i)==0
             theta_Q(i)=0;
         else
             theta_Q(i)=(Q_old(i+1+sigma_Q(i))-
Q_old(i+sigma_Q(i)))/(Q_old(i+1)-Q_old(i));
         end
         G_old_H(i)=(theta_old_H(i)+abs(theta_old_H(i)))/(1+theta_old_H(i));
         G_H(i)=(theta_H(i)+abs(theta_H(i)))/(1+theta_H(i));
         G_old_Q(i)=(theta_old_Q(i)+abs(theta_old_Q(i)))/(1+theta_old_Q(i));
         G_Q(i)=(theta_Q(i)+abs(theta_Q(i)))/(1+theta_Q(i));
         psi_old_H(i)=(0.5*G_old_H(i)*(abs(a_old_H(i))+k*(a_old_H(i)^2))-
abs(a_old_H(i)))*(H_old(i)-H_old(i-1));
         psi_H(i)=(0.5*G_H(i)*(abs(a_H(i))+k*(a_H(i)^2))-
abs(a_H(i)))*(H_old(i+1)-H_old(i));
         psi_old_Q(i)=(0.5*G_old_Q(i)*(abs(a_old_Q(i))+k*(a_old_Q(i)^2))-
abs(a_old_Q(i)))*(Q_old(i)-Q_old(i-1));
         psi_Q(i)=(0.5*G_Q(i)*(abs(a_Q(i))+k*(a_Q(i)^2))-
abs(a_Q(i)))*(Q_old(i+1)-Q_old(i));


    end

    for i=1:n
```

```
        Q(i)=(H(i)*v(i));
        Q_old(i)=(H_old(i)*v_old(i));
        E_old(i)=(H_old(i)*(v_old(i)^2))+(0.5*g*(H_old(i)^2));

        Q1(i)=(H1(i)*v_old(i));
        Q2(i)=(H2(i)*v_old(i));
        Q3(i)=(H3(i)*v_old(i));
        Q4(i)=(H4(i)*v_old(i));

        E1(i)=(H1(i)*(v_old(i)^2))+(0.5*g*(H1(i)^2));
        E2(i)=(H2(i)*(v_old(i)^2))+(0.5*g*(H2(i)^2));
        E3(i)=(H3(i)*(v_old(i)^2))+(0.5*g*(H3(i)^2));
        E4(i)=(H4(i)*(v_old(i)^2))+(0.5*g*(H4(i)^2));
    end
    for i=2:n-1
        H1(i)=H_old(i);
        H2(i)=H_old(i)-0.25*0.5*k*(Q1(i+1)-Q1(i-1));
        H3(i)=H_old(i)-0.33*0.5*k*(Q2(i+1)-Q2(i-1));
        H4(i)=H_old(i)-0.5*0.5*k*(Q3(i+1)-Q3(i-1));
        H(i)=H_old(i)-0.5*k*(Q4(i+1)-Q4(i-1));
        Q1(i)=Q_old(i);
        Q2(i)=Q_old(i)-0.25*0.5*k*(E1(i+1)-E1(i-1));
        Q3(i)=Q_old(i)-0.33*0.5*k*(E2(i+1)-E2(i-1));
        Q4(i)=Q_old(i)-0.5*0.5*k*(E3(i+1)-E3(i-1));
        Q(i)=Q_old(i)-0.5*k*(E4(i+1)-E4(i-1));
    end
    for i=1:n
        H(i)=H(i)-0.5*k*(psi_H(i)-psi_old_H(i));
        Q(i)=Q(i)-0.5*k*(psi_Q(i)-psi_old_Q(i));
    end
     H(1)=H(2);
    H(n)=H(n-1);
    Q(1)=Q(2);
    Q(n)=Q(n-1);

    v=Q./H;

    H_old=H;
    Q_old=Q;
    v_old=v;

end



 H=smooth(H);
 Q=smooth(Q);

plot(H);
grid on;
xlabel('-------X (m)-------->');
ylabel('-------H (m)-------->');
title('Height vs x by Runge Kutta Method with TVD');
```

Appendix 4:

```
% Gudunov's Method

clear all;
clc;

n=100;
for i=1:n
    if i<=50
        H_old(i)=10;
        H(i)=10;

    else
        H_old(i)=1;
        H(i)=1;

    end
    v_old(i)=0;
    v(i)=0;
    Q(i)=0;
    Q_old(i)=0;
    E_old(i)=0;
end
L=100;
dx=1;
dt=0.1;
k=(dt/dx);
g=9.81;


for t=0:dt:4

    for i=2:n-1
        Q(i)=(H(i)*v(i));
        Q_old(i)=(H_old(i)*v_old(i));
        E_old(i)=(H_old(i)*(v_old(i)^2))+(0.5*g*(H_old(i)^2));
        a(i)=max(abs(v_old(i)+sqrt(g*H_old(i))),abs(v_old(i-
1)+sqrt(g*H_old(i-1))));

        F1(i)=0.5*(Q_old(i)+Q_old(i-1))-0.5*(H_old(i+1)-H_old(i));
        F2(i)=0.5*(E_old(i)+E_old(i-1))-0.5*(Q_old(i+1)-Q_old(i));

    end

    for i=2:n-1
        H(i)=H_old(i)-k*(F1(i)-F1(i-1));
        Q(i)=Q_old(i)-k*(F2(i)-F2(i-1));
    end
    H=smooth(H);
    Q=smooth(Q);
    H(1)=H(2);
    H(n)=H(n-1);
    Q(1)=Q(2);
    Q(n)=Q(n-1);
```

```
    H_old=H;
    Q_old=Q;
    v_old=v;
    v=Q./H;

end

for i=1:n
    if H(i)>10
        H(i)=10;
    end
end

plot(H);

grid on;
xlabel('-------X (m)-------->');
ylabel('-------H (m)-------->');
title('Height vs x by Gudunov');
```