Math 228B: Project 3 Writeup

Hongli Zhao

March 31, 2020

1 Burger's Equation Riemann Problem Linear Upwind Method

We have the quasilinear Burger's equation:

$$u_t + uu_x = 0$$

1.1 (a) Find exact solution

Due to the presence of jump discontinuities, the exact solution is in the Rankine-Hugoniot weak sense.

1.1.1 initial condition $u_L < u_R$

Using the method of characteristics, define:

$$\frac{dx(t)}{dt} = u(x(t), t)$$

as our characteristic lines, by total derivative formula, solution u is constant along the characteristics:

$$\frac{du}{dt} = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x}\frac{dx}{dt} = u_t + uu_x = 0$$

thus the characteristic lines will be given by:

$$x(t) = u_0(x_0)t + x_0$$

then given the initial data:

$$u(x,0) = \begin{cases} u_L = 0, x < 0 \\ u_R = 1, x > 0 \end{cases}$$

we have that if $x_0 < 0$, $u_0(x_0) = 0$, and if $x_0 > 0$, $u_0(x_0) = 1$. Concisely:

$$x(t) = \begin{cases} x_0, x_0 < 0 \\ t + x_0, x_0 > 0 \end{cases}$$

In the $t - x$ plane, this is reflected as straight vertical lines on the left half of the plane, and straight lines of slope 1 on the right half of the plane. The region spanning $x = 0$ and $x = t$ is undefined. An infinite number of weak solutions can be defined. The two typical solutions are:

1. shock wave solution;

$$u(x,t) = \begin{cases} 0, x < st \\ 1, x > st \end{cases}$$

where s is the speed of propagation.

In the shock wave solution, by the Rankine-Hugoniot condition, we have:

$$s(u_L - u_R) = f(u_L) = f(u_R)$$

where f is the quasilinear term.

$$s = \frac{\frac{1}{2} \cdot 1^2 - \frac{1}{2} \cdot 0^2}{1 - 0} = \frac{1}{2}$$

Therefore the shock wave solution is:

$$u(x,t) = \begin{cases} 0, x < \frac{1}{2}t \\ 1, x > \frac{1}{2}t \end{cases}$$

Another physical solution is the rarefaction wave:

$$u(x,t) = \begin{cases} 0, x \leq 0 \\ \frac{x}{t}, 0 \leq x < t \\ 1, x > t \end{cases}$$

which corresponds to vanishing viscosity. We should use the rarefaction wave solution be our true exact solution, as it satisfies the entropy condition and is physical.

1.1.2 initial condition $u_L > u_R$

With the initial data:

$$u_0(t) = \begin{cases} u_L = 1 \\ u_R = 0 \end{cases}$$

The characteristic curves are switched (in terms of left and right half plane) compared to the previous case above. Therefore there exists a unique shock wave solution:

$$u(x,t) = \begin{cases} 1, x < \frac{1}{2}t \\ 0, x > \frac{1}{2}t \end{cases}$$

where the shock wave speed is determined already by the Rankine-Hugoniot condition in (a1).

1.2 (b) Numerical Solution

The final solutions at $t = 0.8$ are as follows, using the finest level of discretization ($N = 240$).

1.3 $u_L < u_R$

In the case $u_L < u_R$, with the CFL condition less than 1, the numerical solution has a rarefaction wave behavior.

It is also interesting to see that if we use a CFL condition of exactly 1, we get a shock wave solution (which is unphysical), we will be treating the rarefaction wave solution as our true solution in the following parts.

Figure 1: Numerical solution for $u_L < u_R$ at $T = 0.8$

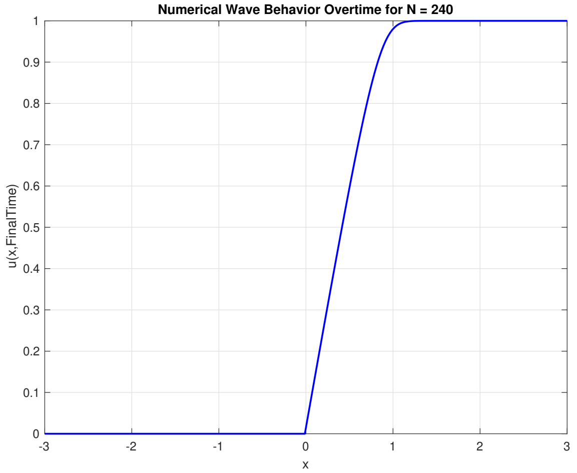
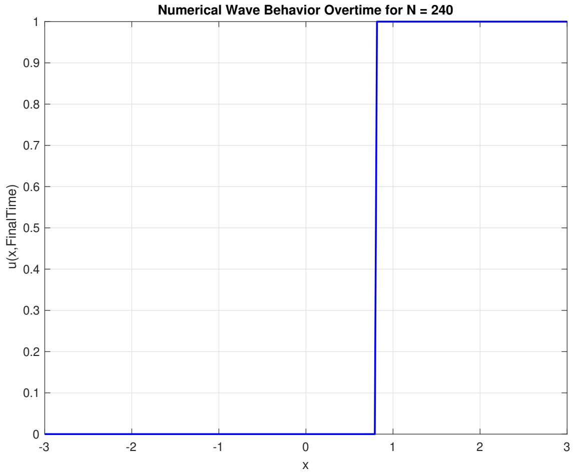


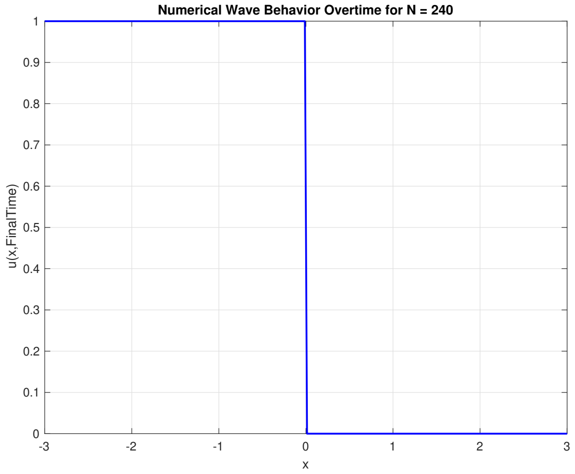
Figure 2: Numerical solution for $u_L < u_R$, CFL=1.0, Shock wave behavior at $T = 0.8$



1.4 $u_L > u_R$

The numerical solution for this case is a jump discontinuity solution.

Figure 3: Numerical solution for $u_L > u_R$ at $T = 0.8$



1.5 (c) The CFL Condition

We know for linear advection equation we have the form:

$$u_t + cu_x = 0$$

yielding the CFL condition:

$$\left| \frac{c\Delta x}{\Delta t} \right| \leq 1$$

via Von Neumann analysis.

In the nonlinear case, we can express the hyperbolic conservation law in quasilinear form:

$$u_t + u \cdot u_x = 0$$

for the Burger's equation and use:

$$\left| \frac{U_{max}\Delta x}{\Delta t} \right| \leq 1$$

as the CFL condition, where U_{max} would be the maximum value of the numerical solution U on the grid.

Since we always have $U_j^n \leq 1$, we can let $U_{max} = 1$, and thus our CFL condition is:

$$\left|\frac{\Delta t}{\Delta x}\right| \leq 1$$

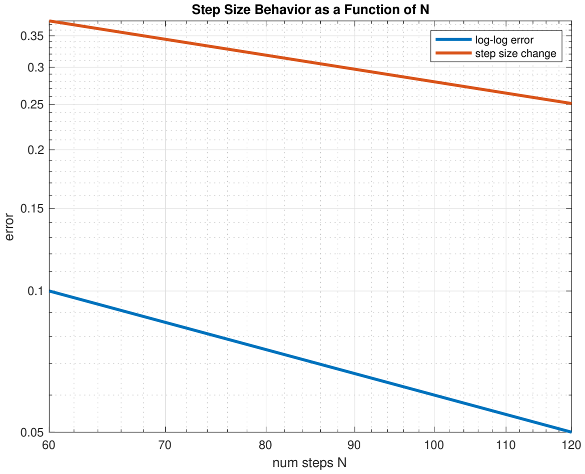
which in the next section of this problem we show that different CFL conditions will give rise to different numerical behaviors.

1.6 (d) Convergence of upwind method

1.7 $u_L < u_R$

The log-log error plot (as a function of number of grid points N) is presented below. Since we are using $N = 240$ as the base solution, the error will become 0 and log-log will be undefined at that point; therefore, we only plot $N = 60, 120$ and expect linear convergence. The error plot is overlaid with the $O(h)$ plot as a comparison.

Figure 4: Log-Log Error Behavior $u_L < u_R$

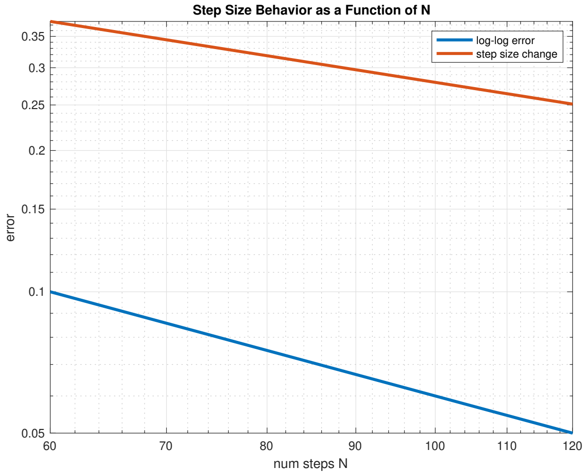


The linear upwind scheme is (slightly better than) first order convergent.

1.8 $u_L > u_R$

Similarly, the log-log plot is:

Figure 5: Log-Log Error Behavior $u_L > u_R$



1.9 (e)(f) Convergence

As shown in the error plot, the method does converge as we increase the number of steps for our discretization. However, the method is not in conservative form, we would expect that it does not capture the advection behavior fully, and therefore does not converge to the true exact solution, as the plot shown below demonstrates, comparing the numerical solution at $N = 240$ with the exact weak solution.

1.10 (g) Comparison with exact solution

The wave behavior is as follows, using $N = 240$. The linear upwind scheme fails to capture the advection and introduces error, especially in the jump discontinuity solution:

Figure 6: Numerical v. Exact Solution, $u_L < u_R$, at $T = 0.8$

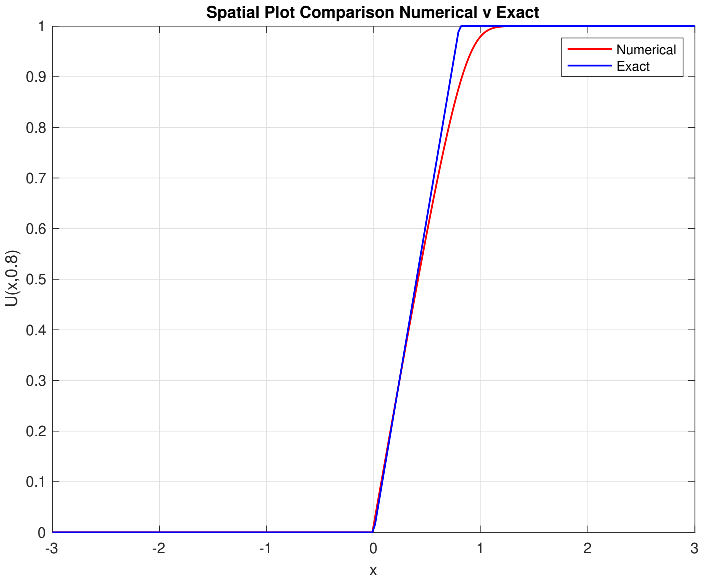
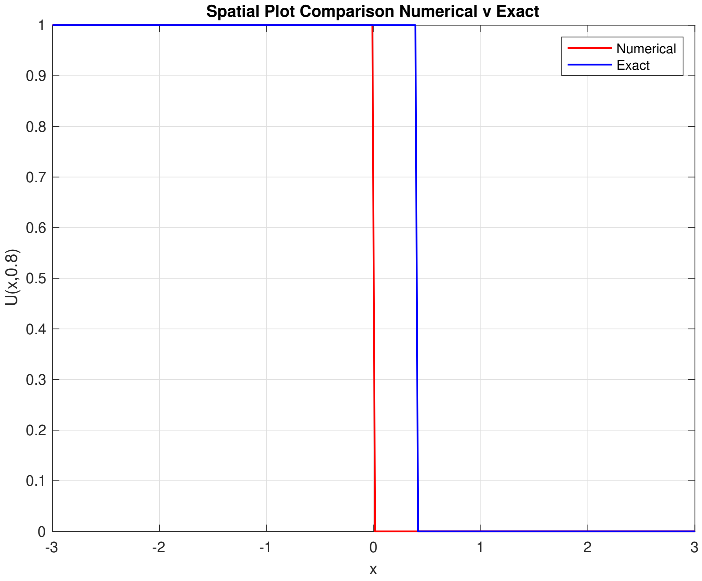


Figure 7: Numerical v. Exact Solution, $u_L > u_R$, at $T = 0.8$



2 Burger's Equation Riemann Problem Conservative Upwind Method

Derivation of the scheme is as follows.

Instead of discretizing the nonlinear term in x, we maintain the quasilinear form and choose to discretize the flux f directly:

$$u_t + (f(u))_x = 0$$

here the flux function for the Burger's equation is:

$$f(u) = \frac{1}{2}u^2$$

2.1 (a) Numerical Flux Function

The conservative form use numerical differencing with respect to the flux directly, and the form is given by:

$$U_j^{n+1} = U_j^n - \frac{\Delta t}{\Delta x}\big[F(U_{j-p}^n, \cdots, U_j^n, \cdots, U_{j+q}^n) - F(U_{j-p-1}, \cdots, U_{j+q-1})\big]$$

Given the first order upwind method:

$$U_j^{n+1} = U_j^n - \frac{\Delta t}{\Delta x}(f(U_j^n) - f(U_{j-1}^n))$$

discretizing the flux term, we plug in the flux function:

$$U_j^{n+1} = U_j^n - \frac{\Delta t}{\Delta x}(\frac{1}{2}(U_j^n)^2 - \frac{1}{2}(U_{j-1}^n)^2)$$

which is in the conservative form.

Thus our numerical flux function is the same as the flux:

$$F(U_j^n) = \frac{1}{2}(U_j^n)^2$$

2.2 (b) Convergence study

Similar to the case of linear upwind method, we plot log-log error behavior using $N = 60, 120$ (since we are using $N = 240$ as our base solution, the error at $N = 240$ will be 0 and log will be undefined).

The conservative form upwind method is still first-order accurate as we increase the number of grid points.

Figure 8: Log-Log Error Behavior, $u_L < u_R$

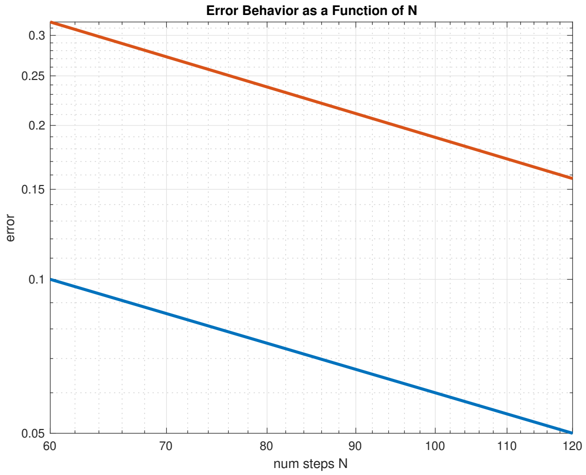
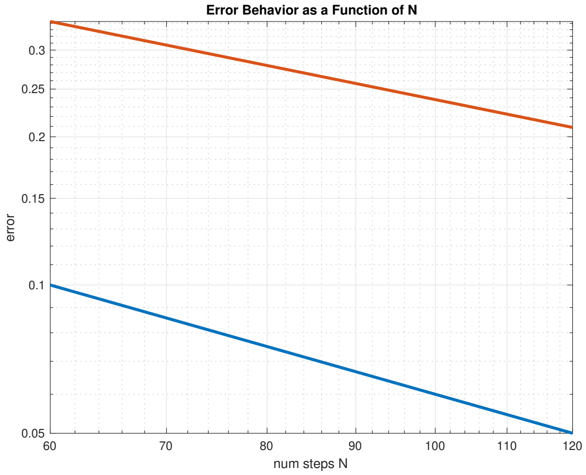


Figure 9: Log-Log Error Behavior, $u_L > u_R$



2.3 (c) Compare numerical solution with exact solution, superimposed plots

The solution plots are presented below, at $T = 0.8$ and with $N = 240$.

Although there are still errors present due to the method is only linearly accurate, we see that the conservative upwind method captures the advection of the jump discontinuity weak solution well.

Figure 10: Numerical v. Exact Solution, $u_L < u_R$, at $T = 0.8$

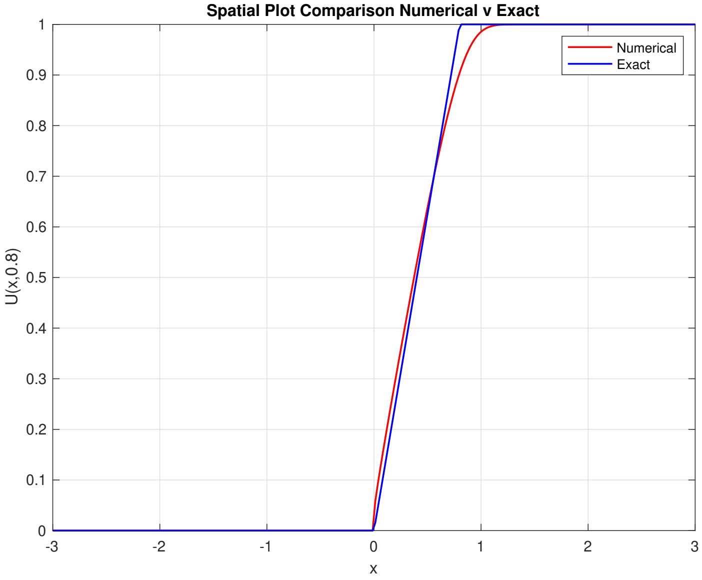
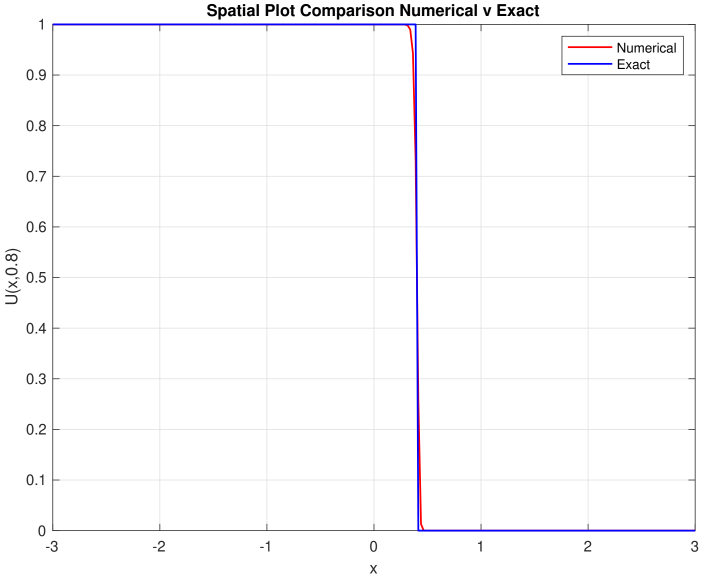


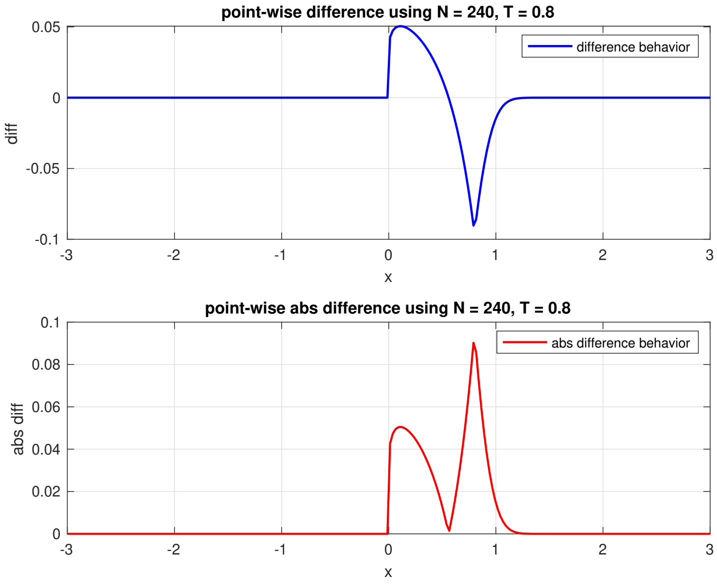
Figure 11: Numerical v. Exact Solution, $u_L > u_R$, at $T = 0.8$



2.4 (d) Point-wise differences between numerical and exact solution

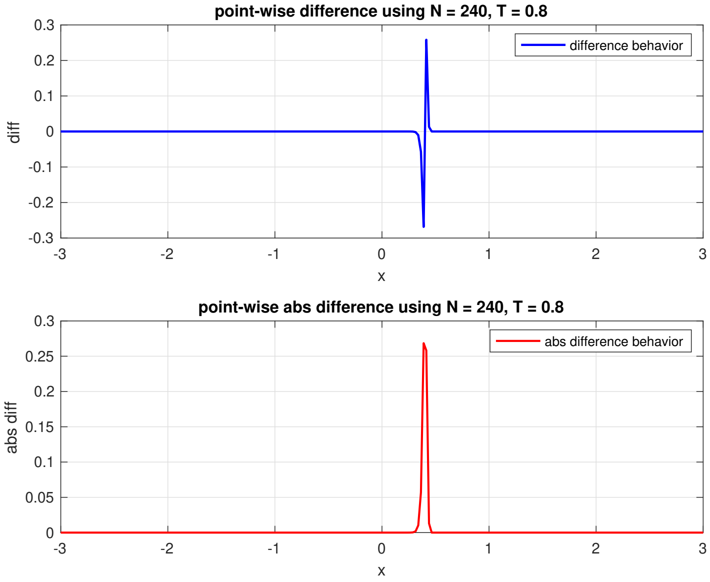
Absolute difference plot is also included. At time $T = 0.8$, using number of grid points $N = 240$:

Figure 12: Difference plot $u_L < u_R$, at $T = 0.8$



Looking at the difference plots and comparing them with the solution plots at $T = 0.8$ above, we see that errors tend to be large at the ”sharp turns” of both solutions, when the solution transitions from one boundary to another.

Figure 13: Difference plot $u_L > u_R$, at $T = 0.8$



3 Two Step Lax Wendroff with Strang Splitting

Here we are solving the hyperbolic system:

$$\begin{cases} \frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} = 0 \\ \frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}(\alpha \frac{Q^2}{A}) + \frac{A}{\rho} \cdot \frac{\partial p(A)}{\partial x} = -2 \cdot \frac{\alpha}{\alpha - 1} \nu \frac{Q}{A} \end{cases}$$

With initial data:

$$\begin{pmatrix} A(x,0) \\ Q(x,0) \end{pmatrix} = \begin{pmatrix} A_0 \\ 0 \end{pmatrix}$$

and $A_0 > 0$.

And boundary data:

$$\begin{cases} p(0,t) = p_0(t) \\ p(L,t) = p_L(t) \end{cases}$$

with pressure function:

$$p(A) = G_0(\sqrt{\frac{A}{A_0}} - 1)$$

and the derivative with respect to A (useful in caculating the Jacobian below), is:

$$p'(A) = (\frac{G_0}{2\sqrt{A_0}})\frac{1}{\sqrt{A}}$$

The Lax-Wendroff scheme with Strang splitting is given. Let $h = \Delta x$ and $k = \Delta t$:

$$U_{j-\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}(U_j^n + U_{j-1}^n) + \frac{1}{2}k\Big[- \frac{F(U_j^n) - F(U_{j-1}^n)}{h} + \frac{S(U_j^n) - S(U_{j-1}^n)}{2}\Big]$$

$$U_{j+\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}(U_{j+1}^n + U_j) + \frac{1}{2}k\Big[- \frac{F(U_{j+1}^n) - F(U_j^n)}{h} + \frac{S(U_{j+1}^n) - S(U_j^n)}{2}\Big]$$

$$U_j^{n+1} = U_j^n - \frac{k}{h}\Big[F(U_{j+\frac{1}{2}}^{n+\frac{1}{2}}) - F(U_{j-\frac{1}{2}}^{n+\frac{1}{2}})\Big] + \frac{1}{2}k\Big[S(U_{j+\frac{1}{2}}^{n+\frac{1}{2}}) - S(U_{j-\frac{1}{2}}^{n+\frac{1}{2}})\Big]$$

which is a 3-stencil.

3.1 treating the boundary data

At the boundaries, $x = 0$ and $x = L$, we have the pressure information given by:

$$\begin{cases} p(0,t) = p_0(t) \\ p(L,t) = p_L(t) \end{cases}$$

using the pressure formula we can derive:

$$\begin{cases} p_0(t) = G_0(\sqrt{\frac{A(0,t)}{A_0}} - 1) \\ p_L(t) = G_0(\sqrt{\frac{A(L,t)}{A_0}} - 1) \end{cases}$$

and we have the boundary data for A:

$$\begin{cases} A(0,t) = A_0\Big[\frac{1}{G_0}p_0(t) + 1\Big]^2 \\ A(L,t) = A_0\Big[\frac{1}{G_0}p_L(t) + 1\Big]^2 \end{cases}$$

At the boundaries, we would also need the data $Q(0,t), Q(L,t)$. To derive the data, we need to linearize the solution next to the boundaries, i.e. $U(1,t), U(L-h,t)$ and use "upwinding".

Linearization around a the solution $U_* = \begin{pmatrix} A_* \\ Q_* \end{pmatrix}$ would yield:

$$\begin{pmatrix} A \\ Q \end{pmatrix}_t + \begin{pmatrix} 0 & 1 \\ -\alpha \frac{Q_*^2}{A_*^2} + \frac{1}{\rho} A_* p'(A_*) & 2\alpha \frac{Q_*}{A_*} \end{pmatrix} \cdot \begin{pmatrix} A \\ Q \end{pmatrix}_x = S(A,Q) = \begin{pmatrix} 0 \\ f(A,Q) \end{pmatrix} \quad (1)$$

where f is given by:

$$f(A,Q) = -2\frac{\alpha}{\alpha - 1}\nu\frac{Q}{A}$$

Letting B to denote the Jacobian matrix in (1), the solution U_* would approximate the boundary data well as $\Delta x, \Delta t$ become small. We can use the linearization and solve a linear system in the Riemann invariants, name w_*, v_*, which are functions of Q, A (would like to recover w_0^{n+1}, v_N^{n+1}, from:

$$\begin{cases} w_t + \lambda_1^* w_x = s_1(w,v) \\ v_t + \lambda_2^* v_x = s_2(w,v) \end{cases}$$

where λ^* is given by:

$$\lambda_{1,2} = \frac{\alpha Q}{A} \pm \sqrt{\alpha(\alpha-1)(\frac{Q}{A})^2 + \frac{G_0}{2\rho}(\frac{A}{A_0})^{\frac{1}{2}}}$$

with $\lambda_1 < \lambda_2$.

The system arose from diagonalizing the linerized system matrix B above and finding the eigenvalues λ_1, λ_2. Then finding the left eigenvectors would yield a matrix $L = \begin{pmatrix} l_1^T \\ l_2^T \end{pmatrix}$ where l_1, l_2 are left eigenvectors. Then multiplying L on both sides of the linearized hyperbolic system would yield:

$$L\begin{pmatrix} A \\ Q \end{pmatrix}_t + \begin{pmatrix} \lambda_1^* & 0 \\ 0 & \lambda_2^* \end{pmatrix} \cdot L\begin{pmatrix} A \\ Q \end{pmatrix}_x = L\begin{pmatrix} 0 \\ f(A,Q) \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$$

then let $\begin{pmatrix} w \\ v \end{pmatrix} = L\begin{pmatrix} A \\ Q \end{pmatrix}$ gives us the system.

And use the upwind scheme to recover the boundary.

The upwind scheme is provided:

$$w_0^{n+1} = w_0^n - \frac{k}{h}\lambda_1^*(w_1^n - w_0^n) + \frac{k}{2}((s_1)_1^n + (s_1)_0^n)$$

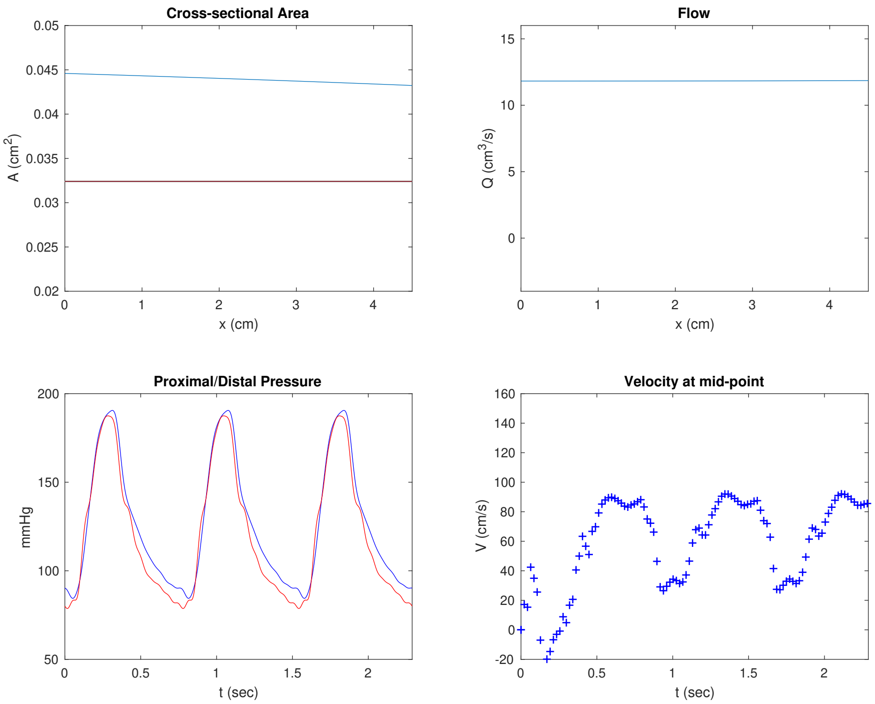
$$v_N^{n+1} = v_N^n - \frac{k}{h}\lambda_2^*(v_N^n - v_{N-1}^n) + \frac{k}{2}((s_2)_N^n + (s_2)_{N-1}^n)$$

3.2 Numerical results

First, plot the velocity at midpoint for both the normal and hyperaemia cases. The MATLAB code takes $\sim$ 10 minutes to run to fully complete the numerical solution.

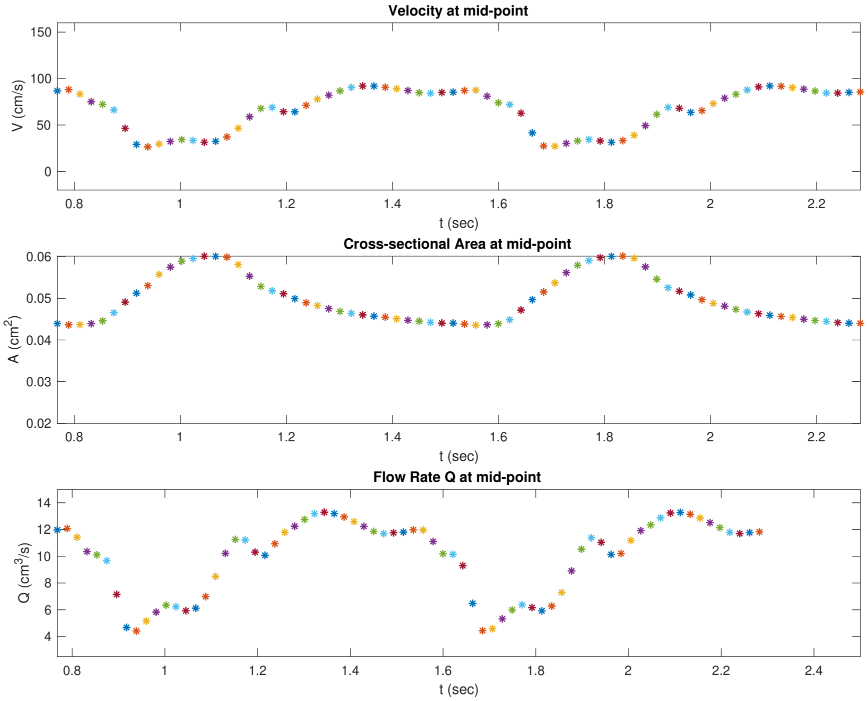
At final time $0.763s \times 3$ cycles $= 2.289s$, the plots are as follows:

Figure 14: Profile, normal artery at final time, 3 cycles



The midpoint velocity and cross-sectional area after stabilization is (1 cycle):

Figure 15: Velocity, Area and Flow profile, normal artery, 2 cycles



The artery profile during hyperaemia is presented below, we see that the scheme is able to capture increased blood flow and velocity.

Figure 16: Profile, hyperaemia artery at final time, 3 cycles

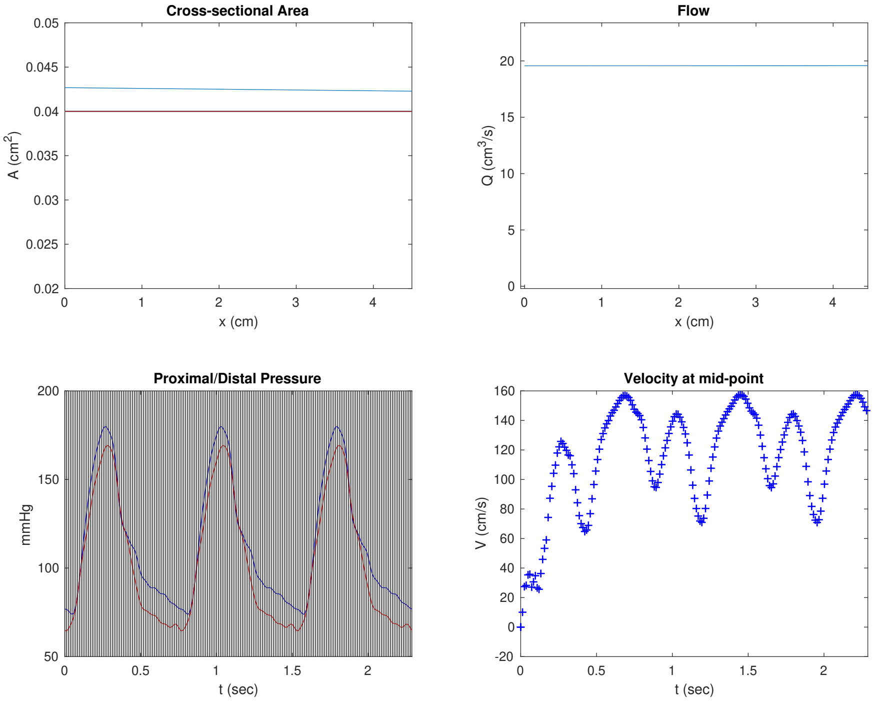
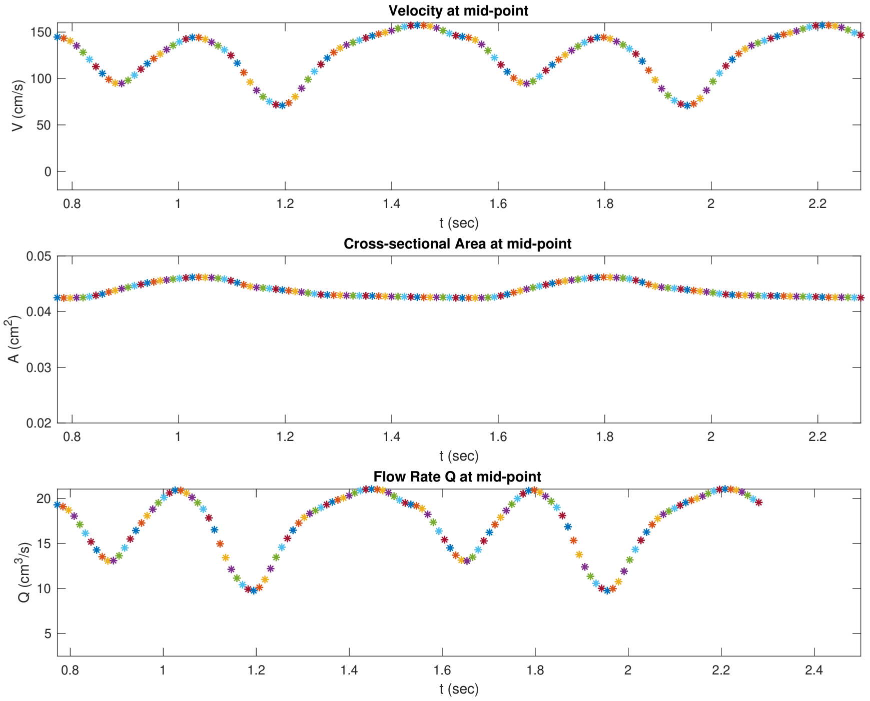


Figure 17: Velocity, Area and Flow profile, hyperaemia artery, 2 cycles



3.3 Reporting Maximum Mid-point Flow Rate

The outputs for the reporting routine is as follows, MATLAB code included in zip file:

```
=========*=========*=========*==========
normal artery
maxflow reporting ...
max flow= 13.2923

=========*=========*=========*==========
hyperaemia artery
maxflow reporting ...
max flow= 21.0546
```

Thus this also shows that the scheme is able to capture the increased flow rate.

3.4 Comment on plot behavior during first cycle $t \in [0, 0.8]$

The solution velocity seems to be unstable during the first simulation of cardiac cycle, with random values oscillating around the starting point. For the rest artery case, velocity was even initially negative, which is unphysical.

Inspecting the numerical cross-sectional area, A seems to exhibit oscillatory behavior during the first cycle and at times grows irregularly. As shown below (modified plotting routines to show what happened during $(0, 0.763)$):

Figure 18: Profile, normal artery, full 3 cycles

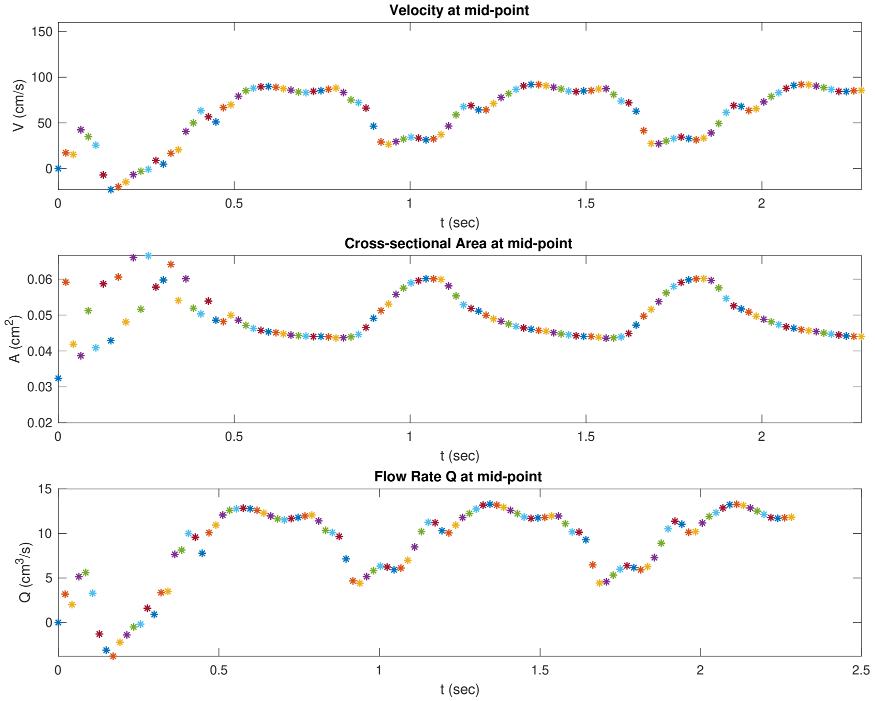
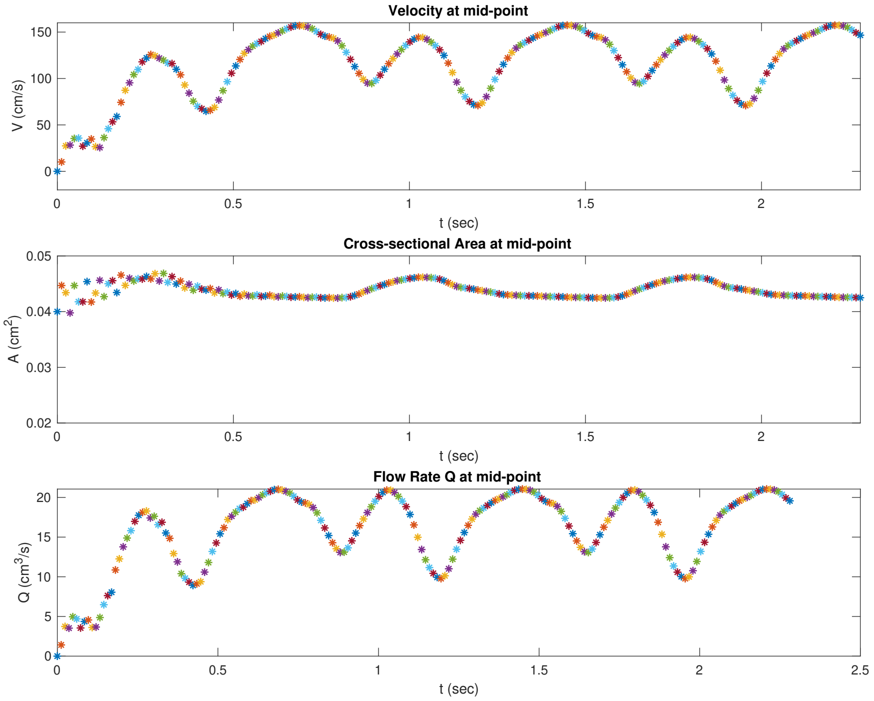


Figure 19: Profile, hyperaemia artery, full 3 cycles



As shown in the plots, our scheme introduces oscillations before it stabilizes.