Research Article
Journal of Marine and Engineering Technology (JOINMET) 1(2), 48-58, 2021
Recieved: 09-Dec-2021 Accepted: 25-Dec-2021

Numerical Simulations of Convection-Diffusion Equation Using Higher-Order Finite Difference Techniques

1st Osman UNAL[1*], 2nd Nuri AKKAS[1]

[1] Maritime Vocational High School, Sakarya University of Applied Sciences, Sakarya 54050, Turkey, osmanunal@subu.edu.tr, nuriakkas@subu.edu.tr

ABSTRACT

Computational Fluid Dynamic (CFD) researchers encounter some numerical errors namely numerical dispersion and unphysical oscillation. First-order finite difference technique causes to large numerical dispersion. Therefore, CFD researchers prefer higher-order methods in order to decrease the numerical dispersion. There are some higher-order space discretization methods in literature such as Quick (Quadratic Upstream Interpolation for Convective Kinematics), TCDF (third order continuously differentiable function) flux limiter and Adaptive flux limiter. Although these higher-order space discretization methods diminish the numerical errors, the use of first-order time discretization method leads to huge numerical dispersion. On the other hand, second-order time discretization method significantly reduces numerical dispersion. First objective of this study is to propose combination of second-order accurate Crank-Nicolson time discretization method and TCDF spatial discretization techniques. Secondly, this study is to present all numerical simulation codes in Matlab environment to make other researchers' works easy.

Keywords: Finite Difference Technique, Higher-Order Methods, Numerical Simulation.

Yüksek Mertebeden Sonlu Fark Yöntemlerini Kullanarak Konveksiyon-Difüzyon Denklemin Sayısal Simülasyonları

ÖZ

Hesaplamalı Akışkanlar Dinamiği (HAD) araştırmacıları, sayısal dağılım ve fiziksel olmayan salınım gibi bazı sayısal hatalarla karşılaşırlar. Birinci mertebeden sonlu farklar yöntemi büyük sayısal dağılıma neden olur. Bu nedenle, HAD araştırmacıları sayısal dağılımı azaltmak için daha yüksek mertebeden yöntemleri tercih ederler. Literatürde Quick (Konvektif Kinematik için Kuadratik Yukarı Akış İnterpolasyonu), TCDF (üçüncü dereceden sürekli türevlenebilir fonksiyon) akı sınırlayıcı ve Uyarlamalı akı sınırlayıcı gibi bazı yüksek dereceli uzay ayrıklaştırma yöntemleri vardır. Bu yüksek dereceli uzay ayrıklaştırma yöntemleri sayısal hataları azaltsa da, birinci derece zaman ayrıklaştırma yönteminin kullanılması büyük sayısal dağılıma yol açar. Öte yandan, ikinci dereceden zaman ayrıklaştırma yöntemi sayısal dağılımı önemli ölçüde azaltır. Bu çalışmanın ilk amacı, ikinci dereceden doğruluktaki Crank-Nicolson zaman ayrıklaştırma yöntemi ile TCDF uzaysal ayrıklaştırma tekniklerinin bir kombinasyonunu önermektir. İkinci olarak, bu çalışma diğer araştırmacıların çalışmalarını kolaylaştırmak için tüm sayısal simülasyon kodlarını Matlab ortamında sunmaktır.

Anahtar Kelimeler: Sonlu Fark Yöntemi, Yüksek Dereceli Yöntemler, Sayısal Simülasyon.

*Corresponding Author's email: osmanunal@subu.edu.tr

1 Introduction

Numerical dispersions and unphysical oscillations are main problems of CFD researchers. Higher-order techniques are suggested to decrease numerical dispersion [1]. Third-order Quick method developed by Leonard [2] effectively reduced numerical errors. Zhang [3] improved the TCDF method in order to diminish numerical dispersion and optimize the number of Newton iterations using continuously differentiable function. Jiang [4] decreases the top limit of the limiter from 2 to 1.5 to decrease the number of Newton iterations. In 2020, Unal [5] developed a flexible flux limiter function that is a function of the Courant number instead of constant value. While the flexible flux limiter function adjusts the upper limit of the limiter function closer to 2 for low Courant number to increases accuracy of the numerical calculation, it approaches the upper limit of the limiter function to 1.15 at high courant values in order to suppress unphysical oscillation. This study proposes to use second-order time discretization method with previously developed space discretization techniques to reduce numerical dispersion effectively.

2 Higher-order Space Discretization Methods

Equation 1 shows unsteady convection-diffusion equation [6] that consists of diffusion term, convection term and accumulation term.

$$D\nabla^2 U - v.\nabla U = \frac{\partial U}{\partial t} \tag{1}$$

In equation 1, D refers to coefficient of physical dispersion and v implies to the velocity. Equation 2 shows one-dimensional form of unsteady convection-diffusion equation.

$$D\frac{\partial^2 U}{\partial x^2} - v\frac{\partial U}{\partial x} = \frac{\partial U}{\partial t} \tag{2}$$

Spatial discretization of one-dimensional form of unsteady convection-diffusion equation is indicated in equation 3.

$$D\frac{U_{i+1} - 2U_i + U_{i-1}}{\Delta x^2} - v\frac{U_{i+1/2} - U_{i-1/2}}{\Delta x} = \frac{\partial U}{\partial t} \tag{3}$$

In equation 3, central values such as U_i, U_{i+1} and U_{i-1} are known in numerical calculation. However, there is no any exact knowledge about face values like as U_{i+1} and U_{i-1}. Therefore, these face values must be predicted according to proper techniques.

$$U_{i+1/2} \approx U_i \tag{4}$$

Equation 4 shows single point upstream technique. Upstream technique [7] means that the right face value equals to the previous central block value.

$$U_{i+1/2} \approx \frac{U_{i+1} + U_i}{2} - \frac{U_{i+1} - 2U_i + U_{i-1}}{8} \tag{5}$$

Quick method indicated in equation 5 is good technique to decrease numerical dispersion. However, it leads to unphysical oscillation when physical dispersion coefficient approaches to zero [8]. Therefore, limiter functions are used to predict face values. Equation 6 indicates the limiter functions.

$$U_{i+1/2} \approx U_i + \frac{1}{2}\varphi(r)\left[U_i - U_{i-1}\right] \tag{6}$$

In equation 6, first right-hand side term is upstream term and second right-hand side term is anti-diffusive term. Equation 7 shows the concentration gradient ratio (r). It is used to determine limiter function (φ).

$$r = \frac{U_{i+1} - U_i}{U_i - U_{i-1}} \tag{7}$$

Unphysical oscillation is suppressed using the total variation diminishing (TVD) approach [9, 10]. Limiter must pass through grey shaded region in figure 1 to get second order accuracy. This condition is satisfied by TCDF and Adaptive 1.5 methods. The TCDF and Adaptive 1.5 methods are shown in Equations 8 and 9.

$$\varphi(r)=\begin{cases} r^3-2r^2+2r & if, \quad 0\le r<0.5 \\ 0.75r+0.25 & if, \quad 0.5\le r<2 \\ \dfrac{2r^2-2r-9/4}{r^2-r-1} & if, \quad 2\le r<+\infty \end{cases} \tag{8}$$

$$\varphi(r)=\begin{cases} r^3-2r^2+2r & if, \quad 0\le r<0.5 \\ -\dfrac{1.5}{15}\log\left(e^{\left(-\frac{1.5}{15}\right)(0.75r+0.25)}+e^{-15}\right) & if, \quad 0.5\le r \end{cases} \tag{9}$$

3 Combination of TCDF techniques and Crank-Nicolson Method

The upper limit of the TCDF method is fixed to 2 and the upper limit of the Adaptive 1.5 is fixed to 1.5. Adaptive 1.5 is better method than TCDF method to vanish unphysical oscillations. Figure 1 indicates TCDF and Adaptive 1.5 methods in TVD region.

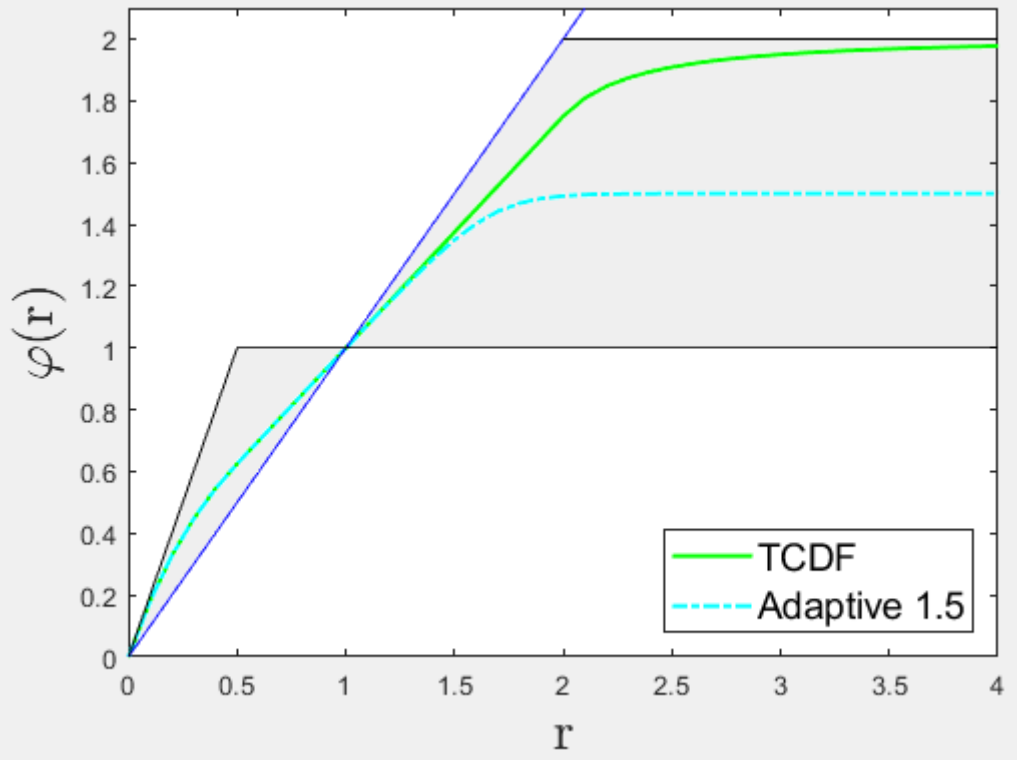

Figure 1. TVD region.

In this study, fully implicit method (equation 10) and semi-implicit method [11] (equation 11) will be used as time discretization methods. Semi-implicit method (Crank-Nicolson) decreases numerical dispersion significantly compared to first-order fully implicit method.

$$-\nu\frac{U_{i+1/2}^{n+1}-U_{i-1/2}^{n+1}}{\Delta x}=\frac{U_i^{n+1}-U_i^n}{\Delta t} \tag{10}$$

$$-\nu\frac{0.5\left(U_{i+1/2}^{n}-U_{i-1/2}^{n}\right)+0.5\left(U_{i+1/2}^{n+1}-U_{i-1/2}^{n+1}\right)}{\Delta x}=\frac{U_i^{n+1}-U_i^n}{\Delta t} \tag{11}$$

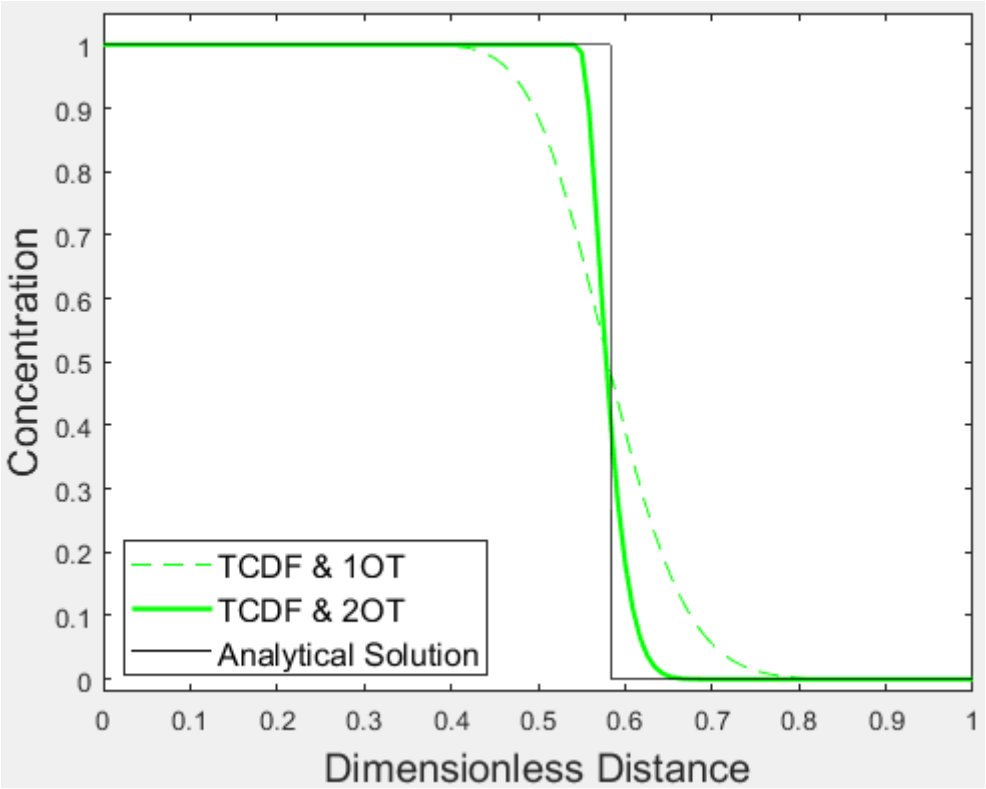

Figure 2. Application of Crank-Nicolson method to TCDF limiter (Nc=1).

Figure 2 shows first-order accurate fully implicit time discretization method for third-order accurate TCDF space discretization method. Figure 2 indicates that combination of TCDF technique and Crank-Nicolson method are closer to analytical solution than first-order time discretization method.

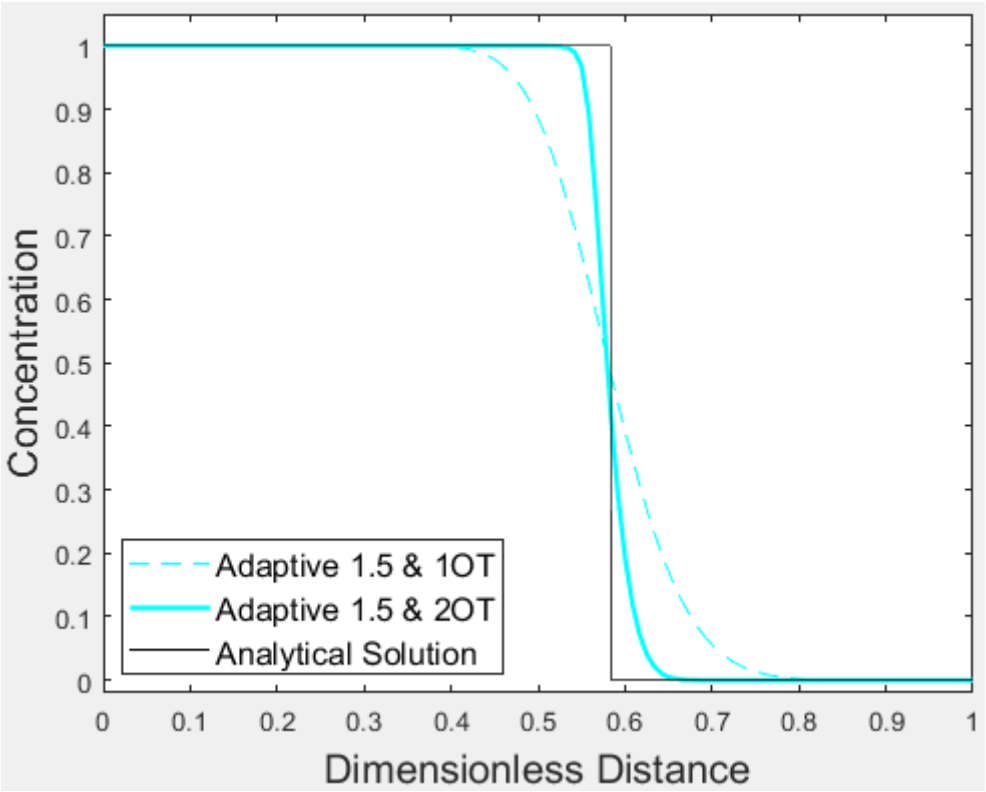

Figure 3. Application of Crank-Nicolson method to Adaptive 1.5 limiter (Nc=1).

Similarly, figure 3 indicates first-order accurate fully implicit time discretization method for third-order accurate Adaptive 1.5 space discretization method. Figure 3 implies that Crank-Nicolson method can be applied to Adaptive 1.5 methods. Crank-Nicolson method reduces nearly more than half of numerical dispersion for both schemes. Thus, this study proposes to combine Crank-Nicolson time discretization method and TCDF or Adaptive 1.5 techniques to minimize numerical dispersion.

4 Conclusions

In this study, higher-order finite difference schemes for unsteady convection-diffusion equation have been investigated to minimize numerical errors namely numerical dispersion. It has been observed that TCDF and Adaptive 1.5 space discretization methods are reduced the numerical dispersion. However, numerical errors are also due to time discretization methods. Higher-order time discretization methods are used to decrease numerical errors. This study proposed to combine second-order Crank-Nicolson time discretization method and TCDF or Adaptive 1.5 techniques to diminish numerical dispersion. It has been observed that combination of higher-order time and higher-order space discretization method minimizes numerical dispersion and the numerical results of this combination are quite close to analytical solution without any unphysical oscillation.

Nomenclature

U = concentration

D = physical dispersion

Nc = courant number

r = concentration gradient ratio or flux ratio

Δt = timestep

v = velocity

φ = limiter function

Subscripts

i = index for blocks in the x direction

$i-\frac{1}{2}$ = index for left face values

$i+\frac{1}{2}$ = index for right face values

Superscripts

n = old timestep

$n+1$ = current timestep

Authors' Contribution

Osman ÜNAL: To create ideas or hypotheses for research and/or article, to plan materials and methods to reach conclusions, to take responsibility for the logical explanation and presentation of findings, to take responsibility for literature review during research, to take responsibility for the creation of the entire article, to rework not only in terms of spelling and grammar, but also in terms of intellectual content before submitting the article.

Nuri AKKAŞ: To create ideas or hypotheses for research and/or article, to plan materials and methods to reach conclusions, to take responsibility for the logical explanation and presentation of findings, to take responsibility for literature review during research, to take responsibility for the creation of the entire article, to rework not only in terms of spelling and grammar, but also in terms of intellectual content before submitting the article.

References

[1] Wolcott, D., H. Kazemi, and R. Dean. A practical method for minimizing the grid orientation effect in reservoir simulation. in SPE annual technical conference and exhibition. 1996. Society of Petroleum Engineers.

[2] Leonard, B.P.J.C.m.i.a.m. and engineering, A stable and accurate convective modelling procedure based on quadratic upstream interpolation. 1979. 19(1): p. 59-98.

[3] Zhang, D., et al., A review on TVD schemes and a refined flux-limiter for steady-state calculations. 2015. 302: p. 114-154.

[4] Jiang, J. and R.M. Younis. An Efficient Fully-Implicit MFD-MUSCL Method Based on a Novel Multislope Limiting Procedure. in SPE Reservoir Simulation Conference. 2017. Society of Petroleum Engineers.

[5] ÜNAL, O., Simulation of Immiscible Displacement of Petroleum via Second and Third Order Finite Differencing Techniques. 2020.

[6] Peaceman, D.W., Fundamentals of numerical reservoir simulation. 2000: Elsevier.

[7] Ertekin, T., J.H. Abou-Kassem, and G.R. King, Basic applied reservoir simulation. 2001.

[8] Liu, J., High-resolution methods for enhanced oil recovery simulation. 1993, University of Texas at Austin.

[9] Harten, A.J.S.J.o.N.A., On a class of high resolution total-variation-stable finite-difference schemes. 1984. 21(1): p. 1-23.

[10] Sweby, P.K.J.S.j.o.n.a., High resolution schemes using flux limiters for hyperbolic conservation laws. 1984. 21(5): p. 995-1011.

[11] Crank, J. and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. in Mathematical Proceedings of the Cambridge Philosophical Society. 1947. Cambridge University Press.

Appendices

Appendix A

```
tic
clear all
clc
I=120+1;   % Number of i-direction points
t=7;    % Total simulation time %t=9.5;
W=1;    % W=1-->upstream   W=0.5-->mid-point           W=0-->downstream
Q=0.5;  % Q=1-->implicit   Q=0.5-->center-in-time(CN)  Q=0-->explicit
dx=0.1; % Space interval
dt=0.14;    % Time interval %dt=0.095;
vf=1;       % Velocity*(df/du)
L=vf*dt/dx; % Courant Number
D=0;     % Physical dispersion
K=D*dt/2/dx^2;  % Diffusion term coefficient
Dnum=vf*dx*((W-0.5)+L*(Q-0.5)); % Numerical dispersion
Dt=D+Dnum;
Dnum_Leonard=vf*dx*((dx/6)+L*(Q-0.5)); % Numerical dispersion of Leonard
Daa=vf*dx*(L*(Q-0.5));
Dt_Leonard=D+Dnum_Leonard;
U00=0.5; Ui0=0; % Initial Condition
U0n=1; UIn=0;   % Boundary Condition
% Concentration Convergence Criteria
Ucc=0.00001;
% NRI interval
dU=10^-8;
Up(1)=U00;
Up(2:I)=Ui0;
% Fuzzy
```

```
LF=readfis('LF');
n_plus=evalfis([1.05;3;L],LF);
for n=1:t/dt
Uv=Up;
it=1;
vUv=Ucc;
while vUv>=Ucc
iU=Uv+dU;
for ii=1:I
if ii==1
f(ii)=K*(Uv(ii+1)-2*U0n+U0n+Up(ii+1)-2*Up(ii)+Up(ii))-L*(Q*(U0n-U0n)+(1-Q)*(Up(ii)-Up(ii)))-
U0n+Up(ii);
elseif ii==2
f(ii)=K*(Uv(ii+1)-2*Uv(ii)+U0n+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)-U0n)+(1-Q)*(Up(ii)-
Up(ii-1)))-Uv(ii)+Up(ii);
elseif ii==3
rvi=(Uv(ii+1)-Uv(ii))/(Uv(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-Uv(ii-1));
rvi_1=(Uv(ii)-Uv(ii-1))/(Uv(ii-1)-U0n);
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-U0n);
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
f(ii)=K*(Uv(ii+1)-2*Uv(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
elseif 3<ii && ii<I-1
rvi=(Uv(ii+1)-Uv(ii))/(Uv(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-Uv(ii-1));
rvi_1=(Uv(ii)-Uv(ii-1))/(Uv(ii-1)-Uv(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-Uv(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
f(ii)=K*(Uv(ii+1)-2*Uv(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
elseif ii==I-1
rvi=(UIn-Uv(ii))/(Uv(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-Uv(ii-1));
rvi_1=(Uv(ii)-Uv(ii-1))/(Uv(ii-1)-Uv(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-Uv(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
f(ii)=K*(UIn-2*Uv(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
elseif ii==I
f(ii)=K*(UIn-2*UIn+Uv(ii-1)+Up(ii)-2*Up(ii)+Up(ii-1))-L*(Q*(UIn-Uv(ii-1))+(1-Q)*(Up(ii)-Up(ii-
1)))-UIn+Up(ii);
end
end
A=f';
B=zeros(I,I);
for ii=1:I
if ii==1
fiU(ii)=K*(iU(ii+1)-2*U0n+U0n+Up(ii+1)-2*Up(ii)+Up(ii))-L*(Q*(U0n-U0n)+(1-Q)*(Up(ii)-Up(ii)))-
U0n+Up(ii);
B(ii,ii+1)=(fiU(ii)-f(ii))/dU;
elseif ii==2
fiU(ii)=K*(Uv(ii+1)-2*iU(ii)+U0n+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(iU(ii)-U0n)+(1-Q)*(Up(ii)-
Up(ii-1)))-iU(ii)+Up(ii);
B(ii,ii)=(fiU(ii)-f(ii))/dU;
```

```
fiU(ii)=K*(iU(ii+1)-2*Uv(ii)+U0n+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)-U0n)+(1-Q)*(Up(ii)-
Up(ii-1)))-Uv(ii)+Up(ii);
B(ii,ii+1)=(fiU(ii)-f(ii))/dU;
elseif ii==3
rvi=(Uv(ii+1)-Uv(ii))/(Uv(ii)-iU(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-iU(ii-1));
rvi_1=(Uv(ii)-iU(ii-1))/(iU(ii-1)-U0n);
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(iU(ii-1)-U0n);
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(Uv(ii+1)-2*Uv(ii)+iU(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-iU(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
B(ii,ii-1)=(fiU(ii)-f(ii))/dU;
rvi=(Uv(ii+1)-iU(ii))/(iU(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(iU(ii)-Uv(ii-1));
rvi_1=(iU(ii)-Uv(ii-1))/(Uv(ii-1)-U0n);
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-U0n);
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(Uv(ii+1)-2*iU(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(iU(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-iU(ii)+Up(ii);
B(ii,ii)=(fiU(ii)-f(ii))/dU;
rvi=(iU(ii+1)-Uv(ii))/(Uv(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-Uv(ii-1));
rvi_1=(Uv(ii)-Uv(ii-1))/(Uv(ii-1)-U0n);
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-U0n);
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(iU(ii+1)-2*Uv(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
B(ii,ii+1)=(fiU(ii)-f(ii))/dU;
elseif 3<ii && ii<I-1
%for west-west
rvi=(Uv(ii+1)-Uv(ii))/(Uv(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-Uv(ii-1));

rvi_1=(Uv(ii)-Uv(ii-1))/(Uv(ii-1)-iU(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-iU(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(Uv(ii+1)-2*Uv(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
B(ii,ii-2)=(fiU(ii)-f(ii))/dU;
rvi=(Uv(ii+1)-Uv(ii))/(Uv(ii)-iU(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-iU(ii-1));
rvi_1=(Uv(ii)-iU(ii-1))/(iU(ii-1)-Uv(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(iU(ii-1)-Uv(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
```

```
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(Uv(ii+1)-2*Uv(ii)+iU(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-iU(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
B(ii,ii-1)=(fiU(ii)-f(ii))/dU;
rvi=(Uv(ii+1)-iU(ii))/(iU(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(iU(ii)-Uv(ii-1));
rvi_1=(iU(ii)-Uv(ii-1))/(Uv(ii-1)-Uv(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-Uv(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(Uv(ii+1)-2*iU(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(iU(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-iU(ii)+Up(ii);
B(ii,ii)=(fiU(ii)-f(ii))/dU;
rvi=(iU(ii+1)-Uv(ii))/(Uv(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-Uv(ii-1));
rvi_1=(Uv(ii)-Uv(ii-1))/(Uv(ii-1)-Uv(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-Uv(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(iU(ii+1)-2*Uv(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
B(ii,ii+1)=(fiU(ii)-f(ii))/dU;

elseif ii==I-1
%for west-west
rvi=(UIn-Uv(ii))/(Uv(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-Uv(ii-1));
rvi_1=(Uv(ii)-Uv(ii-1))/(Uv(ii-1)-iU(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-iU(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(UIn-2*Uv(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
B(ii,ii-2)=(fiU(ii)-f(ii))/dU;
rvi=(UIn-Uv(ii))/(Uv(ii)-iU(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(Uv(ii)-iU(ii-1));
rvi_1=(Uv(ii)-iU(ii-1))/(iU(ii-1)-Uv(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(iU(ii-1)-Uv(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(UIn-2*Uv(ii)+iU(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(Uv(ii)+ADTvi-iU(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-Uv(ii)+Up(ii);
B(ii,ii-1)=(fiU(ii)-f(ii))/dU;
rvi=(UIn-iU(ii))/(iU(ii)-Uv(ii-1));
lfvi=max(0,TVD(rvi,L,n_plus));
ADTvi=0.5*lfvi*(iU(ii)-Uv(ii-1));
rvi_1=(iU(ii)-Uv(ii-1))/(Uv(ii-1)-Uv(ii-2));
lfvi_1=max(0,TVD(rvi_1,L,n_plus));
ADTvi_1=0.5*lfvi_1*(Uv(ii-1)-Uv(ii-2));
rpi=(Up(ii+1)-Up(ii))/(Up(ii)-Up(ii-1));
lfpi=max(0,TVD(rpi,L,n_plus));
ADTpi=0.5*lfpi*(Up(ii)-Up(ii-1));
```

```
rpi_1=(Up(ii)-Up(ii-1))/(Up(ii-1)-Up(ii-2));
lfpi_1=max(0,TVD(rpi_1,L,n_plus));
ADTpi_1=0.5*lfpi_1*(Up(ii-1)-Up(ii-2));
fiU(ii)=K*(UIn-2*iU(ii)+Uv(ii-1)+Up(ii+1)-2*Up(ii)+Up(ii-1))-L*(Q*(iU(ii)+ADTvi-Uv(ii-1)-
ADTvi_1)+(1-Q)*(Up(ii)+ADTpi-Up(ii-1)-ADTpi_1))-iU(ii)+Up(ii);
B(ii,ii+1)=(fiU(ii)-f(ii))/dU;
elseif ii==I
%for west
fiU(ii)=K*(UIn-2*UIn+iU(ii-1)+Up(ii)-2*Up(ii)+Up(ii-1))-L*(Q*(UIn-iU(ii-1))+(1-Q)*(Up(ii)-
Up(ii-1)))-UIn+Up(ii);
B(ii,ii-1)=(fiU(ii)-f(ii))/dU;
end
end
B(:,1)=[]; B(:,end)=[];
v=sparse(B)\sparse(A);
vX=Uv(2:end-1)'-v;
Uv=[U0n vX' UIn];
vUv=max(abs(v));
it=it+1;
end % NR iteration.
Un=Uv;
Up=Un;
n_it(n)=it-1;
set(gcf, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1]);
plot(0:dx:(I-1)*dx,Un,'gv-','markerfacecolor','g');
hold on
hold off
Sum=0;
for ii=1:length(0:dx:(I-1)*dx)-1
Sum=Sum+abs(Un(ii+1)-Un(ii));
end
TV=Sum;
n_plus=evalfis([TV;it-1;L],LF);
%Dispersion Indicator
for ii=1:length(0:dx:(I-1)*dx)-1
sharp(ii)=(Un(ii)-Un(ii+1))/dx;
end
TVn(n)=TV;
m_sharp(n)=max(sharp);
pause(0.001)
end
max_TV=max(TVn)
T_it=sum(n_it)
max_sharp=mean(m_sharp)
toc
```

Appendix B

```
function y=TVD(x,Courant,n_plus)
SDO=38;
if SDO==1
%First Order Upstream
y=0;
elseif SDO==31
%Third Order Upwinding (TVD with Leonard-Cubic)
y=max(0,min([2,2*x,(2*x+1)/3]));
elseif SDO==32
%Third Order Upwinding (TVD with Leonard-Quick)
y=max(0,min([2,2*x,(3*x+1)/4]));
elseif SDO==33
%TCDF
if 0<=x && x<0.5
y=x^3-2*x^2+2*x;
elseif 0.5<=x && x<2
y=0.75*x+0.25;
elseif 2<=x
y=(2*x^2-2*x-9/4)/(x^2-x-1);
else
y=0;
end
%Adaptive(1.5) Flux Limiter
if 0<=x && x<0.5
y=x^3-2*x^2+2*x;
elseif 0.5<=x
y=1/(-15/1.5)*log(exp((-15/1.5)*(0.75*x+0.25))+exp(-15));
else
y=0;
```

```
end
elseif SDO==37
end
end
```