INTERNATIONAL CENTRE FOR
NUMERICAL METHODS IN ENGINEERING

UNIVERSITAT POLITÈCNICA DE CATALUNYA

MASTER OF SCIENCE IN COMPUTATIONAL MECHANICS

Finite Element in Fluids

Homework 4B-Nonlinear Hyperbolic Problems

Eugenio José Muttio Zavala

March 20, 2019

Submitted To:
Prof. Antonio Huerta
Prof. Matteo Giacomini
Prof. Pablo Saez

1 BURGER'S EQUATION

PROBLEM STATEMENT

Consider a classical example of a nonlinear hyperbolic equation named as "Burger's Equation", where $f(u) = u^2/2$*, which in convective form is:*

$$\begin{cases} u_t + uu_x = 0 \\ u(x,0) = u_0(x) \end{cases} \tag{1.1}$$

The goal is to implement a nonlinear system of equations solver based in the Newton-Raphson formulation.

Solution:

The problem is composed by a nonlinear transport equation where the convection velocity is the solution u itself. Thus, the characteristics, satisfy the equation:

$$\frac{dx}{dt} = u(x,t) \tag{1.2}$$

The material derivative of the solution u is constant along the characteristics, which slope is constant, thus the characteristics are straight lines. This equation posses a unique solution as long the characteristics do not intersect. Non-intersecting characteristics are obtained for increasing continuous initial data, in contrast a decreasing initial data will lead to a discontinuos solution, in which the characteristics cross. The initial data to solve this problem is showed in the figure 1.1.

As can be seen in the initial data graph, it shows a decreasing line. In that sense, solving the nonlinear problem which lead to obtain a wrong solution. Then, the correct solution can be determined by using the strategy named *vanishing viscosity*. This approach objective is to obtain the inviscid Burger's equation solution by imposing a new term that has a viscous term, but as this term goes to zero then the solution is achieved. The new model of Burger's equation is:

$$u_t + uu_x = \epsilon u_{xx} \tag{1.3}$$

where is only valid when ϵ is small and u is smooth. In order to solve the problem numerically, first a FEM discretization is employed using a shape function approximation as:

$$u(x,t) \approx u_h(x,t) = \sum_j N_j(x)\, u_j(t) \tag{1.4}$$

$$v(x) = N_i(x) \tag{1.5}$$

Then, the next system of equations is arranged and should be solved at each time step:

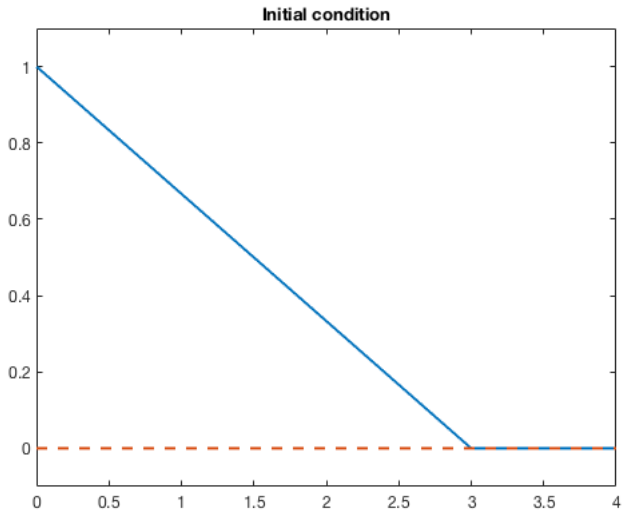


Figure 1.1: Initial data to solve inviscid Burger's equation.

$$\mathbf{M}\dot{\mathbf{U}} + \mathbf{C}(\mathbf{U})\mathbf{U} + \epsilon\mathbf{K}\mathbf{U} = 0 \tag{1.6}$$

This problem can be solved using different numerical strategies as computing a linear system using an explicit scheme like **Forward Euler**:

$$\mathbf{M}\frac{\mathbf{U}^{n+1} - \mathbf{U}^n}{\Delta t} + \mathbf{C}(\mathbf{U}^n)\mathbf{U}^n + \epsilon\mathbf{K}\mathbf{U}^n = 0 \tag{1.7}$$

$$\mathbf{M}\mathbf{U}^{n+1} = \left(\mathbf{M} - \Delta t\left(\mathbf{C}(\mathbf{U}^n) + \epsilon\mathbf{K}\right)\right)\mathbf{U}^n \tag{1.8}$$

The other option is to implement an stable scheme, in which there is a nonlinear system of equations that must be solved, like **Backward Euler**:

$$\mathbf{M}\frac{\mathbf{U}^{n+1} - \mathbf{U}^n}{\Delta t} + \mathbf{C}(\mathbf{U}^{n+1})\mathbf{U}^{n+1} + \epsilon\mathbf{K}\mathbf{U}^{n+1} = 0 \tag{1.9}$$

$$\left(\mathbf{M} + \Delta t\left(\mathbf{C}(\mathbf{U}^{n+1}) + \epsilon\mathbf{K}\right)\right)\mathbf{U}^{n+1} = \mathbf{M}\mathbf{U}^n \tag{1.10}$$

The original code contains an implicit nonlinear equations solver scheme known as *"Picard Method"*, where in each time it is needed to solve the equation 1.10 considering:

$$\mathbf{A}(\mathbf{U}^{n+1}) = \left(\mathbf{M} + \Delta t\left(\mathbf{C}(\mathbf{U}^{n+1}) + \epsilon\mathbf{K}\right)\right) \tag{1.11}$$

By using Picard method, it is needed an initial guess ${}^0\mathbf{U}^{n+1} = \mathbf{U}^n$ and then iterate (k) until convergence:

$$^{k+1}\mathbf{U}^{n+1} = \mathbf{A}^{-1}\left(^{k}\mathbf{U}^{n+1}\right)\left(\mathbf{M}\mathbf{U}^{n}\right) \tag{1.12}$$

In order to implement the Newton-Raphson method, at each time step it is needed to solve $\mathbf{f}(\mathbf{U}^{n+1}) = 0$, by considering this function as:

$$\mathbf{f}(\mathbf{U}) = (\mathbf{M} + \Delta t\mathbf{C}(\mathbf{U}) + \epsilon\Delta t\mathbf{K})\,\mathbf{U} - \mathbf{M}\mathbf{U}^{n} \tag{1.13}$$

It is needed an initial guess, that is the solution at the previous step as the Picard method, $^{0}\mathbf{U}^{n+1} = \mathbf{U}^{n}$ and then iterate (k) until convergence with:

$$^{k+1}\mathbf{U}^{n+1} = \left(^{k}\mathbf{U}^{n+1}\right) - \mathbf{J}^{-1}\left(^{k}\mathbf{U}^{n+1}\right)\mathbf{f}\left(^{k}\mathbf{U}^{n+1}\right) \tag{1.14}$$

Where $\mathbf{J} = \frac{d\mathbf{f}}{d\mathbf{U}}$ is the Jacobian matrix. The computation of the derivative of the function $\mathbf{f}$ has an special characteristic in the convective term, which is that this term depends in the solution. That means, the derivative of $\mathbf{C}(\mathbf{U})$ (residual function methodology) has the form:

$$\frac{d\mathbf{C}(\mathbf{U})}{d\mathbf{U}} = \frac{d\mathbf{C}(\mathbf{U})}{d\mathbf{U}}(\mathbf{C}(\mathbf{W})\mathbf{U} + \mathbf{C}(\mathbf{U})\mathbf{W}) = \mathbf{C}(\mathbf{W}) + \mathbf{C}(\mathbf{U})$$

Then considering $(\mathbf{U},\mathbf{W}) = (\mathbf{U}^{*},\mathbf{U}^{*})$:

$$\mathbf{C}(\mathbf{W}) + \mathbf{C}(\mathbf{U}) = \mathbf{C}(\mathbf{U}^{*}) + \mathbf{C}(\mathbf{U}^{*}) = 2\mathbf{C}(\mathbf{U}^{*})$$

And finally the Jacobian is:

$$\frac{d\mathbf{f}(\mathbf{U}^{*})}{d\mathbf{U}^{*}} = \mathbf{M} + 2\Delta t\mathbf{C}(\mathbf{U}^{*}) + \epsilon\Delta t\mathbf{K}$$

At this point, it has been shown all the ingredients to implement the Newton-Raphson method in Matlab in order to solve the Burger's nonlinear hyperbolic equation.

Code Implementation 1

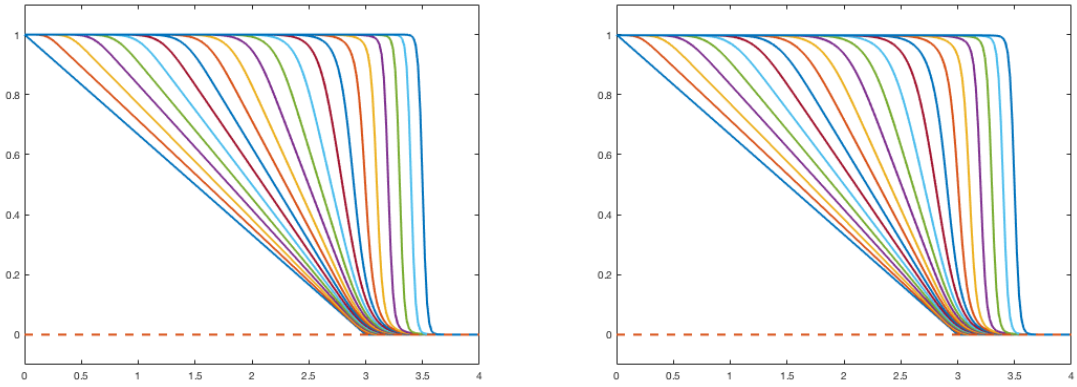
Considering the code given in class, an by duplicating the file *burgers_im.m* corresponding to the implicit solution by the *Picard Method*, it was easier just to modify some lines and include some others inside the *while loop* as:

```
while (error_U > 0.5e-5) && k < 20
        C = computeConvectionMatrix(X,T,U0);
        F = (M + At*C + At*E*K)*U0-M*U(:,n);
        J = M + 2*At*C + At*E*K;
        U1 = U0-J\F;
        error_U = norm(U1-U0)/norm(U1);
        U0 = U1; k = k+1;
    end
```

Now, it can be done the comparison between the methods (Explicit Euler, Implicit Picard and Implicit Newton-Raphson). The problem solved is the same as it was defined in the equation 1.1 with *final time equal to* **Tf** = **4sec**, but the time step and the viscosity imposed was changed to consider the different behavior of the methods, as is listed now:

1. $\Delta t = 0.005; \epsilon = 1e-2$
2. $\Delta t = 0.05; \epsilon = 1e-2$
3. $\Delta t = 0.1; \epsilon = 1e-2$
4. $\Delta t = 0.005; \epsilon = 1e-4$

The next graphics shows the results with different methods:



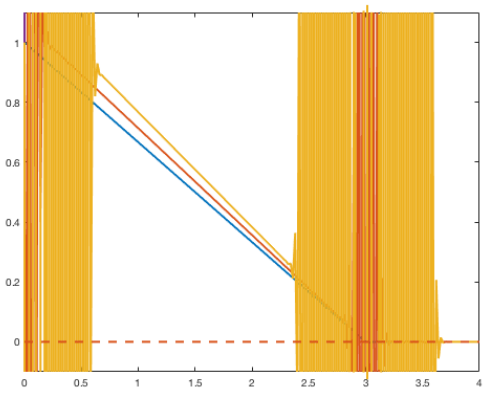
(a) Explicit Method

(b) Newton-Raphson Method

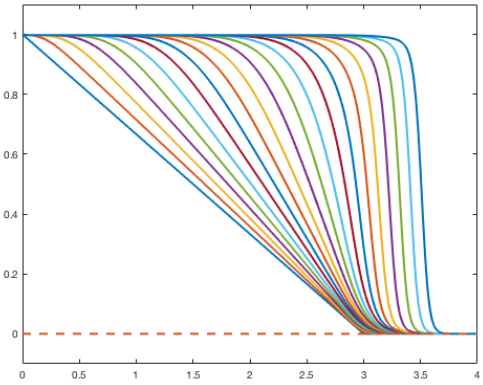
Figure 1.2: First Problem - $\Delta t = 0.005; \epsilon = 1e-2$

As it can be seen in the figure 1.2, both methods works perfectly (and also Picard method), there is no difference between them in the final time. This happens because of the chosen time step and also the viscosity value, that generates excellent results for any method. Consider now the graphs from figure 1.3 (second problem), when the time step is larger. Now, the explicit method blows up because its stability limits, which are not adequate to solve the problem. The same behavior occurs in the third problem using the explicit scheme, which the time step is bigger.

Moreover, consider the graphs from figure 1.4. The time step of this problem is adequate but the solutions are not what it is expected. This is because the viscosity chosen is tending to zero, and as it is explained before, the original inviscid Burger's equation generates discontinuities at the solution when the initial data is decreasing. That is the reason why this two iterative methods (Picard and N-R) are oscillating and diverging from the solution, which is more evident in N-R.

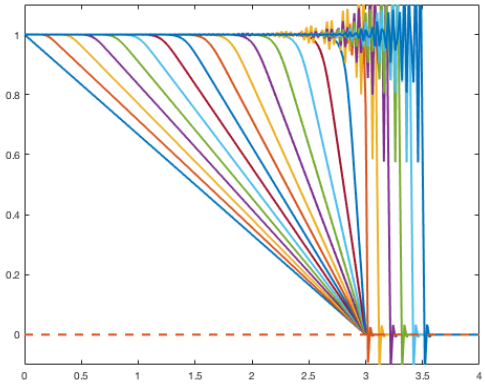


(a) Explicit Method

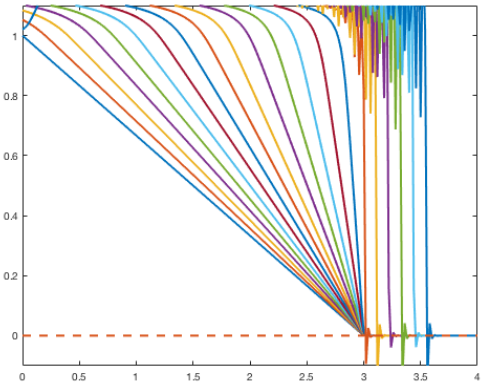


(b) Newton-Raphson Method

Figure 1.3: Second Problem - $\Delta t = 0.05$; $\epsilon = 1e-2$



(a) Picard Method



(b) Newton-Raphson Method

Figure 1.4: Fourth Problem - $\Delta t = 0.005$; $\epsilon = 1e-4$