Numerical Methods in Fluid Mechanics
Rahul Krishna
Summer term 2024

Institut für Strömungsmechanik und Umweltphysik im Bauwesen

Assignment 4

Flux Limiters - Finite volume schemes for 1D advection equation

Date of issue: 20.06.2024

Date of submission: 03.07.2024

Requirements

- working groups of 2-3 people
- all codes and plots have to be submitted via e-mail to krishna@hydromech.uni-hannover.de
- the present document has to be signed digitally
- codes have to have headers stating the date and names and matriculation numbers of all participants
- tasks marked with a * are optional
- add comments to your codes (at least one comment for each individual information)
- all plots have to have labeled axes and legends
- always use the given variables and functions' names

Declaration of Independence

I hereby declare that the present assignment was worked on independently without help from a third party. No codes or other results were taken from other homework.

Surname	First name	Matriculation number	Signiture
Teesteegen	Marie	10061302	M Teesteegen
Duvvuru	LOKesh	10063226	Duvvuru
Toumi	Younes Abdelfatil	10064473	Toumi

1 Problem: Advective transport

The propagation of a concentration pulse in a current with constant flow velocity shall be calculated numerically in 1D. For that purpose, the flux limiter method shall be implemented. The transport is described by the transient advection equation:

$$\frac{\partial c}{\partial t} + v\frac{\partial c}{\partial x} = 0 \tag{1}$$

The domain has periodic boundary conditions. The initial condition is given by the function `init` from Assignment 3.

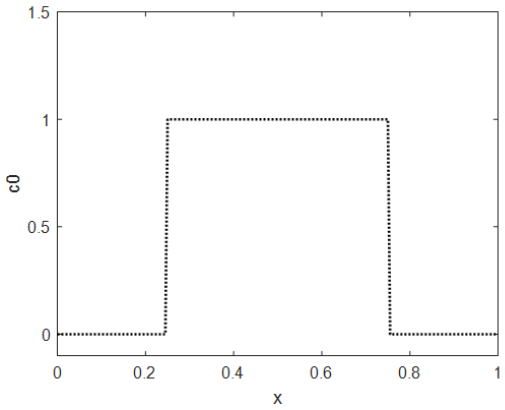


Figure 1: Initial condition from `init`

2 Flux limiter for advective problems

The mathematician Sergei K. Godunov proved a theorem that stated: A linear numerical scheme for solving partial differential equations which does not create new extrema (monotonic scheme) can be at most of first order.

Because of that it is not possible to simply increase the order of the method as this induces e.g. oscillations at sharp fronts. Nonlinear methods like flux limiters have been developed to be anyhow able to use higher order schemes and get more accurate solutions.

The jumps for the flux correction are chosen using limiter functions to prevent the scheme from creating new extrema. The most common limiter are Minmod, Superbee and MC limiter.

2.1 Deviation of an explicit finite volume scheme with flux limitations

A scheme for propagation of the solution reads

$$C_i^{n+1} = C_i^n + \frac{\Delta t}{\Delta x}\left(F_{i-1,i}^n - F_{i,i+1}^n\right), \tag{2}$$

where $F_{i-1,i}^n$ is an approximation of the flux over the interface between volume $i-1$ and i calculated from the approximate solution at time t^n. The first order upwind scheme uses the fluxes

$$F_{i-1,i}^n = \begin{cases} vC_{i-1}^n & \text{if } v > 0 \\ vC_i^n & \text{if } v < 0 \end{cases} \tag{3}$$

For higher order schemes, flux corrections are introduced. For the case $v > 0$, the flux approximation is

$$F_{i-1,i}^n = vC_{i-1}^n + \frac{1}{2}v\left(1 - \frac{v\Delta t}{\Delta x}\right)\Delta C_{i-1,i}^n. \tag{4}$$

The according scheme for higher order propagation of the solution using explicit time integration reads:

$$C_i^{n+1} = C_i^n - v\frac{\Delta t}{\Delta x}\left(C_i^n - C_{i-1}^n\right) - \frac{1}{2}\frac{v\Delta t}{\Delta x}\left(1 - v\frac{\Delta t}{\Delta x}\right)\left(\Delta C_{i,i+1}^n - \Delta C_{i-1,i}^n\right). \tag{5}$$

Depending on the choice of the jump value $\Delta C^n_{i-1,i}$, different higher order schemes are obtained.

In matrix-vector notation, this equation reads

$$\vec{C}^{n+1} = (\mathbf{I} - Cr \cdot \mathbf{P}_b)\, \vec{C}^n - \frac{1}{2} Cr\,(1 - Cr)\left(\Delta\vec{C}^n - \Delta\vec{C}^n_{(-1)}\right) \tag{6}$$

with the Courant number $Cr = v\Delta t/\Delta x$.

A limited version of the flux is introduced with the more general formulation

$$F^n_{i,i+1} = vC^n_i + \frac{1}{2} v \left(1 - \frac{v\Delta t}{\Delta x}\right) \delta C^n_{i,i+1}. \tag{7}$$

This leads to the propagation scheme

$$C^{n+1}_i = C^n_i - v\frac{\Delta t}{\Delta x}\left(C^n_i - C^n_{i-1}\right) - \frac{1}{2}\frac{v\Delta t}{\Delta x}\left(1 - v\frac{\Delta t}{\Delta x}\right)\left(\delta C^n_{i,i+1} - \delta C^n_{i-1,i}\right). \tag{8}$$

The flux correction term has now the jump value

$$\delta C^n_{i,i+1} = \Phi(\Theta^n_{i,i+1})(C^n_{i+1} - C^n_i) \tag{9}$$

with the definition in case $v > 0$

$$\Theta^n_{i,i+1} = \frac{C^n_i - C^n_{i-1}}{C^n_{i+1} - C^n_i}. \tag{10}$$

In matrix-vector notation, the scheme reads now

$$\vec{C}^{n+1} = (\mathbf{I} - Cr \cdot \mathbf{P}_b)\, \vec{C}^n - \frac{1}{2} Cr\,(1 - Cr)\left(\delta\vec{C}^n - \delta\vec{C}^n_{(-1)}\right). \tag{11}$$

The known schemes without limited fluxes (not total variation diminishing) are obtained with the functions

$$\begin{array}{cc} \Phi = 0 & \text{1st order upwinding scheme} \\ \Phi = 1 & \text{Lax Wendroff scheme} \\ \Phi = \Theta & \text{Beam warming scheme} \\ \Phi = \frac{1}{2}(1 + \Theta) & \text{Fromm scheme.} \end{array} \tag{12}$$

2.2 Calculation of the slope

The higher order schemes (Lax Wendroff, Beam warming, Fromm) have the drawback of producing new extrema. To avoid that limiter functions are introduced. The most common limiter functions are Minmod, Superbee and the MC Limiter.

Minmod limiter

The Minmod limiter calculates the limiting function for the jump $C^n_{i+1} - C^n_i$ at the interface between volume i and $i + 1$ as

$$\Phi^n_{\text{minmod } i,i+1} = \text{minmod}\left(\Theta^n_{i,i+1}, 1\right). \tag{13}$$

The function `minmod` is defined as

$$c = \text{minmod}\,(a,\ b) = \begin{cases} a & if\ \ |a| \le |b|\ \ and\ a \cdot b > 0 \\ b & if\ \ |a| > |b|\ \ and\ a \cdot b > 0 \\ 0 & if\ a \cdot b < 0 \end{cases}. \tag{14}$$

This leads to the limited jump function

$$\delta C^n_{\text{minmod } i,i+1} = \Phi^n_{\text{minmod } i,i+1}(C^n_{i+1} - C^n_i) = \text{minmod}\left(C^n_i - C^n_{i-1}, C^n_{i+1} - C^n_i\right). \tag{15}$$

In matrix-vector-form, the `minmod` function is evaluated line by line.

$$\delta C^n_{\text{minmod } i,i+1} = \text{minmod}\,(\mathbf{P}_b \overrightarrow{c}^n,\ \mathbf{P}_f \overrightarrow{c}^n) \tag{16}$$

Superbee limiter

The Superbee limiter is defined as

$$\Phi^n_{\text{superbee } i,i+1} = \max\left(0, min(2\Theta^n_{i,i+1}, 1), min(\Theta^n_{i,i+1}, 2)\right). \tag{17}$$

This leads to the limited jump function

$$\delta C^n_{\text{superbee } i,i+1} = \Phi^n_{\text{superbee } i,i+1}(C^n_{i+1} - C^n_i) = \text{maxmod}\,(A, B) \tag{18}$$

with

$$A = \text{minmod}(2(C^n_i - C^n_{i-1}), C^n_{i+1} - C^n_i) \tag{19}$$

$$B = \text{minmod}((C^n_i - C^n_{i-1}), 2(C^n_{i+1} - C^n_i). \tag{20}$$

The function `maxmod` is defined as

$$\text{c} = \text{maxmod}\,(a,\ b) = \begin{cases} a & if\ \ |a| \ge |b| \\ b & if\ \ |a| < |b| \end{cases}. \tag{21}$$

MC limiter

The MC limiter is defined as

$$\Phi^n_{\text{MC } i,i+1} = \max\left(0, min(2\Theta^n_{i,i+1}, 2, \frac{1 + \Theta^n_{i,i+1}}{2})\right). \tag{22}$$

This leads to the limited jump function

$$\delta C^n_{\text{MC } i,i+1} = \Phi^n_{\text{MC } i,i+1}(C^n_{i+1} - C^n_i) = \text{minmod}\left((C^n_{i+1} - C^n_{i-1}),\ 2(C^n_{i+1} - C^n_i),\ 2(C^n_i - C^n_{i-1})\right). \tag{23}$$

3 Tasks

See Below ↓

3.1 Flux Limiters

3.1.1 Minmod limiter

1. In addition to the jump $\delta\vec{C}^n$, the jump $\delta\vec{C}^n_{(-1)}$ is needed in eq. (11). Write down the equation for calculating $\delta\vec{C}^n_{(-1)}$ (which is $\delta C^n_{\text{minmod } i-1,i}$) corresponding to equation (15).

2. What are the entries of the matrices $\mathbf{O}_f$ and $\mathbf{O}_b$ if the jump $\delta\vec{C}^n_{(-1)}$ is given by

$$\overrightarrow{\delta C}^n_{(-1)} = \text{minmod}\left(\mathbf{O}_f \overrightarrow{c}^n,\ \mathbf{O}_b \overrightarrow{c}^n\right) \tag{24}$$

analogously to equation (16)?

$$\mathbf{O}_f = \begin{bmatrix} & & \\ & & \\ & & \\ & & \end{bmatrix} \qquad \mathbf{O}_b = \begin{bmatrix} & & \\ & & \\ & & \\ & & \end{bmatrix} \tag{25}$$

3.1.2 Superbee limiter

3. Write down equations (19) and (20) for $\delta C^n_{i,i+1}$ in matrix-vector-form.

4. Write down equations (19) and (20) for $\delta C^n_{i-1,i}$ in matrix-vector-form. Use matrices $\mathbf{O}_f$ and/or $\mathbf{O}_b$.

3.1.3 *MC limiter

5. *Write down equation (23) for $\delta C^n_{i,i+1}$ in matrix-vector-form.

6. *Write down equation (23) for $\delta C^n_{i-1,i}$ in matrix-vector-form. Use matrices $\mathbf{O}_f$ and/or $\mathbf{O}_b$. Define matrix $\mathbf{O}_c$ in analogy to matrix $\mathbf{P}_c$.

3.2 Implementation in Matlab

7. Create the function `A = pentadiagcyc(n, u2, u1, h, o1, o2)`. This function creates a matrix A of size $n \times n$. The value h is on the main diagonal, $u1$ on the lower first diagonal, $u2$ on the lower second diagonal, $o1$ on the upper first diagonal and $o2$ on the upper second diagonal. Furthermore, the values of the lower diagonals are in the upper right corner. The values of the upper diagonals are in lower left corner. All remaining entries are zero. If n is lower than five, an error message shall be displayed and the return value is $A = 0$.

   An example:

```
>> A = pentadiagcyc(7, 1, 2, 3, 4, 5)
A =
     3     4     5     0     0     1     2
     2     3     4     5     0     0     1
     1     2     3     4     5     0     0
     0     1     2     3     4     5     0
     0     0     1     2     3     4     5
     5     0     0     1     2     3     4
     4     5     0     0     1     2     3
```

8. Implement the function `c = maxmod(a, b)` defined in equation (21). Valid input values are two scalars a and b or two vectors a and b of the same length. The return value is a scalar or vector c calculated by line-by-line application of the function. Tip: `maxmod([1:5], [-3 3 3 3 -3])` `-> ans = -3 3 3 4 5`.

9. Implement the function `c = minmod(a, b)` defined in equation (14). Valid input values are two scalars a and b or two vectors a and b of the same length. The return value is a scalar or vector c calculated by line-by-line application of the function. Tip: `minmod([1:5], [-3 3 3 3 -3])` `-> ans = 0 2 3 3 0`.

10. Write the script `ueb6.m` based on the implementation for Assignment 2, that calculates the numerical solution to equation (1) on a domain with periodic boundary conditions using slope limiter. Implement the Superbee limiter. The size of the matrix n, the positions of the nodes $\overrightarrow{x}$ and the initial condition $\overrightarrow{c}$ are given by the known `init(2)` function. Use your functions `tridiagcyc` and `pentadiagcyc` for generating the matrices.

11. * Use negative flow velocities. What happens? For what range of the CFL number do you get a stable solution?

12. Set $CFL = 0.5$ and plot the solution after twenty time steps for a first oder scheme for time and space discretizations, a second order scheme for time and space discretizations, the analytical solution (known from Exercise 2) and a scheme using flux limiter. Write down which schemes you have used and compare your results with the statement of the Godunov-theorem (see introduction).

13. Implement the two missing limiter (at least the Minmod limiter). How does the Superbee limiter modify the Gaussian curve (`init(1)`)? Which limiter gives the best results? (The implementation of the MC limiter is optional*.)

Assignment n°4

25 June 2024 11:14

1. Equation for calculating $\delta \vec{C}^n_{(-1)}$:

$\delta \vec{C}^n_{(-1)}$ corresponds to $\delta C^n_{\text{minmod } i-1,i}$ which means :

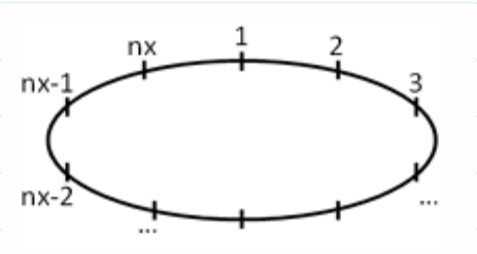
$$\delta \vec{C}^n_{\text{minmod } i-1,i} = \oint^n_{\text{minmod } i-1,i} \left(C^n_i - C^n_{i-1} \right) = \text{minmod} \left(\theta^n_{i-1,i}, 1 \right) \left(C^n_i - C^n_{i-1} \right)$$

$$\Rightarrow \quad \text{minmod} \left(C^n_{i-1} - C^n_{i-2}, C^n_i - C^n_{i-1} \right) \qquad \text{where } \theta^n_{i-1,i} = \frac{C^n_{i-1} - C^n_{i-2}}{C^n_i - C^n_{i-1}}$$

$$\boxed{\delta \vec{C}^n_{(-1)} = \text{minmod} \left(C^n_{i-1} - C^n_{i-2}, C^n_i - C^n_{i-1} \right)}$$

2. Entries of O_f and O_b for $\delta \vec{C}^n_{(-1)}$

Having a periodic BC:



- for $i = 1$:

  $\text{minmod} \left(C^n_{nx} - C^n_{nx-1}, C^n_1 - C^n_{nx} \right)$

- for $i = 2$:

  $\text{minmod} \left(C^n_1 - C^n_{nx}, C^n_2 - C^n_1 \right)$

- for $i = 3$:

  $\text{minmod} \left(C^n_2 - C^n_1, C^n_3 - C^n_2 \right)$

And so on... until $i = nx$

- for $i = nx$:

  $\text{minmod} \left(C^n_{nx-1} - C^n_{nx-2}, C^n_{nx} - C^n_{nx-1} \right)$

Then we get: $\delta C^n_{(-1)} = \text{minmod}(Q_b \vec{C}_n, Q_g \vec{C}_n)$

We can now vectorize and build the Q_g and Q_b matrix: Pouring $\vec{C}^n$:

$$Q_b = \begin{bmatrix} 0 & \cdots & 0 & -1 & 1 \\ 1 & & & 0 & -1 \\ -1 & & & & 0 \\ 0 & -1 & \ddots & & \vdots \\ \vdots & \ddots & & & \\ 0 & \cdots & 0 & -1 & 1 & 0 \end{bmatrix} \qquad \vec{C}^n = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_{nx-2} \\ C_{nx-1} \\ C_{nx} \end{bmatrix}$$

$$Q_g = \begin{bmatrix} 1 & 0 & \cdots & 0 & -1 \\ -1 & & & & 0 \\ 0 & & \ddots & & \vdots \\ \vdots & \ddots & & & \\ & & & 0 & \\ 0 & \cdots & 0 & -1 & 1 \end{bmatrix}$$

3. Writing Equations 19 & 20 for $\delta \vec{C}^n_{i,i+1}$:

The equations 19 & 20 are given by:

$$A = \text{minmod}(2(C^n_i - C^n_{i-1}), C^n_{i+1} - C^n_i) \quad \ldots\ldots (19)$$
$$B = \text{minmod}((C^n_i - C^n_{i-1}), 2(C^n_{i+1} - C^n_i). \quad \ldots\ldots (20)$$

By following same procedure as question 2. we express for each index $i = 1, 2, \ldots nx$:

- for $i = 1$:

A_1. minmod $\left(2(C^n_1 - C^n_{nx}),\ C^n_2 - C^n_1\right)$

B_1. minmod $\left(C^n_1 - C^n_{nx},\ 2(C^n_2 - C^n_1)\right)$

- for $i = 2$:

A_2. minmod $\left(2(C^n_2 - C^n_1),\ C^n_3 - C^n_2\right)$

B_2. minmod $\left(C^n_2 - C^n_1,\ 2(C^n_3 - C^n_2)\right)$

And so on... until $i = nx$

- for $i = nx$:

$$A_{nx} = \text{minmod}\left(2(C_{nx} - C_{nx-1}),\ C_1 - C_{nx}\right)$$

$$B_{nx} = \text{minmod}\left(C_{nx} - C_{nx-1},\ 2(C_1 - C_{nx})\right)$$

We can now vectorize :

$$A = \text{minmod}\left(2 P_b \vec{C}^n,\ P_g \vec{C}^n\right)$$

$$B = \text{minmod}\left(P_b \vec{C}^n,\ 2 P_g \vec{C}^n\right)$$

Stacking $\vec{C}^n$:

$$\vec{C}^n = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_{nx-2} \\ C_{nx-1} \\ C_{nx} \end{bmatrix}$$

where P_b and P_g where already given:

$$P_b = \begin{bmatrix} 1 & 0 & \cdots & 0 & -1 \\ -1 & & & & 0 \\ 0 & \ddots & & & \vdots \\ \vdots & & \ddots & & 0 \\ 0 & \cdots & 0 & -1 & 1 \end{bmatrix}$$

$$P_g = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 \\ 0 & & \ddots & & \vdots \\ \vdots & & & \ddots & 0 \\ 0 & & & & 1 \\ 1 & 0 & \cdots & 0 & -1 \end{bmatrix}$$

4. Writing Equations 19 & 20 for $\delta \vec{C}^n_{i-1,i}$:

Now the superbee limiter between $i-1$ and i is defined by:

$$\delta C^n_{\text{superbee } i-1,i} = \phi^n_{\text{superbee } i-1,i}\left(C^n_i - C^n_{i-1}\right) \qquad \text{where } \theta^n_{i-1,i} = \frac{C^n_{i-1} - C^n_{i-2}}{C^n_i - C^n_{i-1}}$$

$$= \max\left(0,\ \min\left(2\theta^n_{i-1,i},\ 1\right),\ \min\left(\theta^n_{i-1,i},\ 2\right)\right)\left(C^n_i - C^n_{i-1}\right)$$

by replacing, we find

$$\delta C^n_{\text{superbee } i-1,i} = \text{maxmod}\left(\text{minmod}\left(2(C^n_{i-1} - C^n_{i-2}), C^n_i - C^n_{i-1}\right), \text{minmod}\left(C^n_{i-1} - C^n_{i-2}, 2(C^n_i - C^n_{i-1})\right)\right)$$

by expressing it in terms of $A_{(-1)}$ and $B_{(-1)}$ we find :

$$A_{(-1)} = \text{minmod}(2(C^n_{i-1} - C^n_{i-2}), C^n_i - C^n_{i-1})$$
$$B_{(-1)} = \text{minmod}((C^n_{i-1} - C^n_{i-2}), 2(C^n_i - C^n_{i-1}).$$

We can now vectorize :

$$A_{(-1)} = \text{minmod}\left(2O_b \vec{C}^n, O_f \vec{C}^n\right)$$
$$B_{(-1)} = \text{minmod}\left(O_b \vec{C}^n, 2O_f \vec{C}^n\right)$$

5. Writing Equation(23) for $\delta \vec{C}^n_{i,i+1}$ MC :

MC limiter is given by equation (23)

$$\delta C^n_{\text{MC } i,i+1} = \Phi^n_{\text{MC } i,i+1}(C^n_{i+1} - C^n_i) = \text{minmod}\left(\frac{(C^n_{i+1} - C^n_{i-1})}{2}, 2(C^n_{i+1} - C^n_i), 2(C^n_i - C^n_{i-1})\right). \quad (23)$$

By following same procedure as question 2. we express for each index $i = 1, 2,$

- for $i = 1$:

$$\delta C^n_{MC_{1,2}} = \text{minmod}\left(\frac{C^n_2 - C^n_{nx}}{2}, 2(C^n_2 - C^n_1), 2(C^n_1 - C^n_{nx})\right)$$

- for $i = 2$:

$$\delta C^n_{MC_{2,3}} = \text{minmod}\left(\frac{C^n_3 - C^n_1}{2}, 2(C^n_3 - C^n_2), 2(C^n_2 - C^n_1)\right)$$

And so on ... until $i = nx$

- for $i = nx$:

$$\delta C^n_{MC_{nx,1}} = \text{minmod}\left(\frac{C^n_1 - C^n_{nx-1}}{2}, 2(C^n_1 - C^n_{nx}), 2(C^n_{nx} - C^n_{nx-1})\right)$$

We can now vectorize:

$$\delta C^n_{MC} = \text{minmod}\left(P_c \vec{C}^n,\ 2P_f \vec{C}^n,\ 2P_b \vec{C}^n \right)$$

where P_f and P_b are same as previously and P_c given by:

$$P_c = \frac{1}{2}\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & -1 \\ -1 & & \ddots & & & 0 \\ 0 & \ddots & & \ddots & & \vdots \\ \vdots & & \ddots & & & 0 \\ & & & & & 1 \\ 1 & 0 & \cdots & 0 & -1 & 0 \end{bmatrix}$$

6. Writing Equation(23) for $\delta \vec{C}^n_{i-1,i}$ MC:

Now the MC limiter between $i-1$ and i is defined by:

$$\delta C^n_{MC\ i-1,i} = \bar{\phi}^n_{i-1,i}\left(C^n_i - C^n_{i-1}\right)$$

$$= \max\left(0,\ \min\left(2\theta^n_{i-1,i},\ 2,\ \frac{1+\theta^n_{i-1,i}}{2}\right)\right) \qquad \text{where } \theta^n_{i-1,i} = \frac{C^n_{i-1} - C^n_{i-2}}{C^n_i - C^n_{i-1}}$$

replacing, we find

$$\delta C^n_{MC\ i-1,i} = \text{minmod}\left(2(C^n_{i-1} - C^n_{i-2}), 2(C^n_i - C^n_{i-1}), \frac{(C^n_{i-1} - C^n_{i-2}) + (C^n_i - C^n_{i-1})}{2(C^n_i - C^n_{i-1})^n} \cdot (C^n_i - C^n_{i-1})\right)$$

$$\delta C^n_{MC\ i-1,i} = \text{minmod}\left(2(C^n_{i-1} - C^n_{i-2}), 2(C^n_i - C^n_{i-1}), \frac{(C^n_i - C^n_{i-2})}{2}\right)$$

$$\delta C^n_{MC\ i-1,i} = \text{minmod}\left(\frac{C^n_i - C^n_{i-2}}{2}, 2(C^n_i - C^n_{i-1}), 2(C^n_{i-1} - C^n_{i-2})\right)$$

We can now vectorize:

$$\delta C^n_{MC\ (-1)} = \text{minmod}\left(Q_c \vec{C}^n,\ 2Q_f \vec{C}^n,\ 2Q_b \vec{C}^n\right)$$

where O_f and O_b are same as previously and O_c given by:

$$O_c = \frac{1}{2}\begin{bmatrix} 1 & 0 & \cdots & 0 & -1 & 0 \\ 0 & & & & 0 & -1 \\ -1 & & & & & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & \ddots & & & & 0 \\ 0 & \cdots & 0 & -1 & 0 & 1 \end{bmatrix}$$

11. Setting negative velocities

We setting negative flow velocity the solution becomes very unstable.
The values of CFL for which we have stability are:

$$CFL \leqslant 1$$

12. Comparing and Godunov Theorem:

Used schemes:

- first order $\phi = 0$
- Lax Wendroff $\phi = 1$
- Superbee, Minmod, MC flux limiter

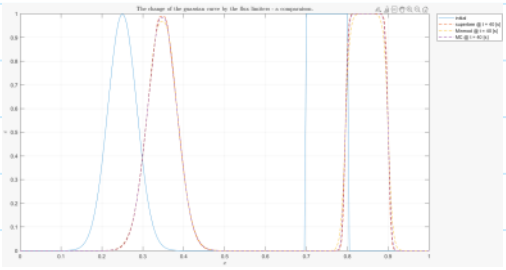
When using the second order scheme of Lax wendroff (corresponding to ϕ 1) new extremas appears as shown in fig transient_compare.png, this being the case only for second order schemes, whereas for the remaining 1st order, no new extremas appeared.
These new extremas can be dealt with using flux limiters.

⇩

Godunov Theorem

13. How does superbee change Gauss ?:

The superbee changes the gaussian curve by diffusing it (see fig gaussian_superbee)
By comparing the three flux limiters, the superbee method seems to be the best compared to the remaining ones, the followed by MC and finally the minmod.
The reason why superbee is better can be shown in the figure, where this last presents the least diffusion.


The change of the gaussian curve by the flux limiters : a comparaison.
initial
superbee @ t = 40 [s]
Minmod @ t = 40 [s]
MC @ t = 40 [s]
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
x