Mathematics in Scientific Computing

Chi-Wang Shu

Division of Applied Mathematics

Brown University

Outline

- Introduction to scientific computing
- Solution of large linear systems
- More examples
- Concluding remarks

Introduction to scientific computing

- Scientific computing is a subject of applying mathematical tools to the design of efficient algorithms on computers
- Difference with computer science: emphasizing the role of mathematics to ensure accuracy, stability, and efficiency of numerical algorithms.

Example 1: Fast Fourier transform.

- The goal of the algorithm is very simple.

  We are given a uniform mesh over $[0, 2\pi]$:

$$0 = x_0 < x_1 < x_2 < \cdots < x_N = 2\pi$$

  where $x_{j+1} - x_j = h = \frac{2\pi}{N}$, and we assume for simplicity N is odd.

There is a one to one correspondence between the discrete Fourier coefficients

$$a_k : \qquad |k| \leq \frac{N-1}{2}$$

and the values of the function $f_N(x)$

$$f_N(x) = \sum_{|k| \leq (N-1)/2} a_k e^{ikx}$$

at the grid points x_j:

$$f_N(x_j) = \sum_{|k| \leq (N-1)/2} a_k e^{ikx_j}, \qquad j = 1, 2, \cdots, N.$$

- For example, if we are given

$$a_k : \qquad |k| \leq \frac{N-1}{2}$$

and we would like to compute $f_N(x_j)$, we could simply perform the summation

$$f_N(x_j) = \sum_{|k| \leq (N-1)/2} a_k e^{ikx_j},$$

for each $j = 1, 2, \cdots, N$. Clearly, the cost (number of multiplications, divisions, summations) is $O(N)$ for each j, hence the total cost is $O(N^2)$ for all $j = 1, 2, \cdots, N$.

- When N is very large (e.g. $N = 5000$ or more for signal analysis and image processing problems), there are two problems:
  - The computational cost to perform the Fourier transform is very large.
  - The accumulation of round-off errors is significant, so the computed results are not very accurate.

- The fast Fourier transform (FFT) is a mathematical reformulation of

$$f_N(x_j) = \sum_{|k| \le (N-1)/2} a_k e^{ikx_j},$$

by regrouping terms which can be repeatedly used, so that one can obtain the values of $f_N(x_j)$ for all $j = 1, 2, \cdots, N$ in $O(N \log N)$ cost. FFT is more efficient if N is a product of small primes, e.g. when $N = 2^n$. For modest N, even for N as large as 5000, $\log N$ is very small. The savings in computational cost is significant — a job which needs thousands of minutes can now be completed in a few minutes. Also, the round-off error accumulation is significantly reduced.

- We emphasize that FFT and the straightforward computation give mathematically identical results. There is no approximation involved here. Mathematics helps us in this case to get the same result much faster and more accurately.

Reference: J.W. Cooley and J.W. Tukey, *An algorithm for the machine calculation of complex Fourier series*, Mathematics of Computation, 19, 1965, 297–301.

Example 2: Fast multipole method.

This more recently developed fast method has the same spirit as the fast Fourier transform.

- The fast multipole method aims at certain types of dense systems of interactions.
- For example, we may have N particles located at x_j, $j = 1, 2, \cdots, N$ in R^3. Each pair of particles would interact with each other, the interaction strength could be inversely proportional to their distance $|x_i - x_j|$ or its square (e.g. gravity).

- Thus, to compute the movement of each particle, we would need to sum up the interaction of this particle with every other particle, resulting in $O(N)$ calculation. The total cost of computing the movement of all N particles is thus $O(N^2)$, which is very large.
- This time, the structure of the problem is less apparent than that of Fourier sum to group terms together to save computational cost. However, with a clever choice of grouping techniques, Greengard and Rokhlin were able to obtain the fast multipole method, which computes an approximation to the desired precision and achieves a computational cost of $O(N \log N)$.

- This technique has now been widely used in solving various integral equations and interaction systems. The discovery of the fast multipole method led to the AMS 2001 Leroy P. Steele Prize for a Seminal Contribution to Research shared by Greengard and Rokhlin.

Reference: L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, Journal of Computational Physics, 73, 1987, 325–348.

Solution of large linear systems

In principle, we know how to solve a linear system

$$Ax = b$$

by, e.g. the Cramer's rule. However, a direct implementation of Cramer's rule involves an operation count on the order of $O(n!)$ when A is an $n \times n$ matrix. This is of course not feasible for large n (e.g. $n = 1$ million which corresponds to a discretization of three dimensional PDE on a mesh of $100 \times 100 \times 100$ grid points).

Fortunately, we then have the method of Gauss elimination. For example, we look at the linear system

$$\begin{aligned} 2x_1 + x_2 &= 4 \\ x_1 - x_2 &= -1 \end{aligned}$$

If we multiply the second equation by 2, and subtract it from the first equation, the system becomes

$$\begin{aligned} 2x_1 + x_2 &= 4 \\ 3x_2 &= 6 \end{aligned}$$

The system is now upper-triangular, which is easy to solve by back substitution, to obtain the solution

$$x_1 = 1, \qquad x_2 = 2$$

It is easy to verify that this procedure has a cost of at most $O(n^3)$ to reach the solution (smaller for banded matrices), which is much lower than the $O(n!)$ cost of applying the Cramer's rule.

We can also show that, with suitable "pivoting", i.e. suitable interchanges of rows and columns of A (which do not change the solution), Gauss elimination can always be performed in a stable fashion whenever A is non-singular.

However, in many situations, even $O(n^3)$ is an excessive cost. Storage might also be a problem to store the full matrix A, when often there are many zeros in its entries.

Therefore, people have been studying various iterative methods to solve $Ax = b$. Iterative methods are particularly attractive for sparse but not narrowly banded matrices A, for which Gauss elimination is quite costly but Ax does not involve much cost to obtain for any given x.

The idea is to introduce iterative matrices B^n and vector c^n, and define the iteration as

$$x^{n+1} = B^n x^n + c^n.$$

We would like to require that, when $x^n \longrightarrow x$ for $n \longrightarrow \infty$, x is the solution of $Ax = b$. We would require B^n and c^n to be easy (cost effective) to compute, and we would require B^n to have similar sparseness as A so that the evaluation of $B^n x^n$ is cost effective.

Of course, it would make sense to use an iterative method only if the convergence is very fast, so that not that many iterations are needed to reach a solution very close to x.

In simpler situations, $B^n = B$ and $c^n = c$ which do not depend on n.

Over the years, there have been a lot of research on designing efficient iterative methods for solving linear systems.

When A has certain special properties, for example if A is symmetric and positive definite, iterative methods which can converge in $O(1)$ number of iterations (independent of n) can be designed (e.g. multigrid methods). Thus, for sparse matrices with a fixed number of entries per row (independent of n) the solution x can be obtained with $O(n)$ cost, which is clearly optimal.

The design and analysis of multigrid methods by Achi Brandt has been recognized both by the applied mathematics community and the computer science community with the Second SIAM/ACM Prize in Computational Science and Engineering (SIAM/ACM CSE Prize) in 2005.

When A is not symmetric and positive definite, there are various Krylov subspace iterative methods (based on the subspaces spanned by b, Ab, A^2b, etc.). However, iterative methods which can guarantee fast convergence are still lacking. This is an active research area.

More examples

We can clearly see the role of mathematics in scientific computing by looking at the work of a few SIAM/ACM CSE Prize recipients:

- The First SIAM/ACM CSE Prize was awarded to John B. Bell and Phillip Colella in 2003, for the development of mathematical methods and computer science tools for science and engineering.
- The Third SIAM/ACM CSE Prize was awarded to Chi-Wang Shu in 2007, for the development of numerical methods that have had a great impact on scientific computing.
- The Fourth SIAM/ACM CSE Prize was awarded to Cleve Moler in 2009, for his individual research in numerical analysis and the efficacy of his invention MATLAB.

- The fifth SIAM/ACM CSE Prize was awarded to J. Tinsley Oden in 2011, for his impact on the development of finite element methods.
- The sixth SIAM/ACM CSE Prize was awarded to Linda R. Petzold in 2013, for her contribution to numerical methods for ordinary differential equations and dynamical systems.
- The seventh SIAM/ACM CSE Prize was awarded to the PETSc Core Development Group (Satish Balay, Jed Brown, William Gropp, Matthew Knepley, Lois Curfman McInnes, Barry Smith, and Hong Zhang) in 2015, for their contribution to the development of Portable, Extensible Toolkit for Scientific Computation.
- The eighth SIAM/ACM CSE Prize was awarded to Thomas J. R. Hughes in 2017, for his pioneering work on finite element methods for partial differential equations.

High order numerical methods for hyperbolic equations

Hyperbolic conservation laws are PDEs of the form

$$u_t + f(u)_x + g(u)_y + h(u)_z = 0 \tag{1}$$

where $\xi_1 f'(u) + \xi_2 g'(u) + \xi_3 h'(u)$ is diagonalizable with real eigenvalues for any real $\xi = (\xi_1, \xi_2, \xi_3)$. The equation may have discontinuous solutions (shocks, contact discontinuities, etc.) even if the initial and boundary conditions are smooth.

Selected list of applications:

- Traffic flows
- Computational fluid dynamics, especially high speed flows
- Electro-magnetic waves, aeroacoustics
- Astrophysics
- Semiconductor device simulations
- Certain problems in computational biology
- . . .

For smooth solutions, high order accurate methods are very efficient. For example, for a periodic function $f(x)$, we assume the Fourier coefficients

$$\hat{f}(k) = \frac{1}{2}\int_{-1}^{1} f(x)e^{-ik\pi x}dx$$

are given for $-N \leq k \leq N$.

The partial Fourier sum is defined by

$$f_N(x) = \sum_{k=-N}^{N} \hat{f}(k) e^{ik\pi x}$$

and it converges very fast to $f(x)$:

$$\max_{-1 \le x \le 1} |f(x) - f_N(x)| \le C e^{-\alpha N}$$

with constants C and $\alpha > 0$ independent of N, provided the periodic function $f(x)$ is analytic.

If the function is less smooth we still have faster than algebraic convergence

$$\|f - f_N\| \leq \frac{C}{N^p} \|f\|_{H^p}$$

as long as f has its p-derivative in L^2.

Similar results exist for non-periodic problems using polynomial basis functions (Chebyshev and Legendre spectral methods).

If $f(x)$ is less smooth, the convergence rate degenerates. In particular, if $f(x)$ is discontinuous, then there is no convergence in the maximum norm:

- Away from the shock the convergence rate is $O(\frac{1}{N})$;
- Near the shock there are $O(1)$ oscillations.

This is called the Gibbs phenomenon.

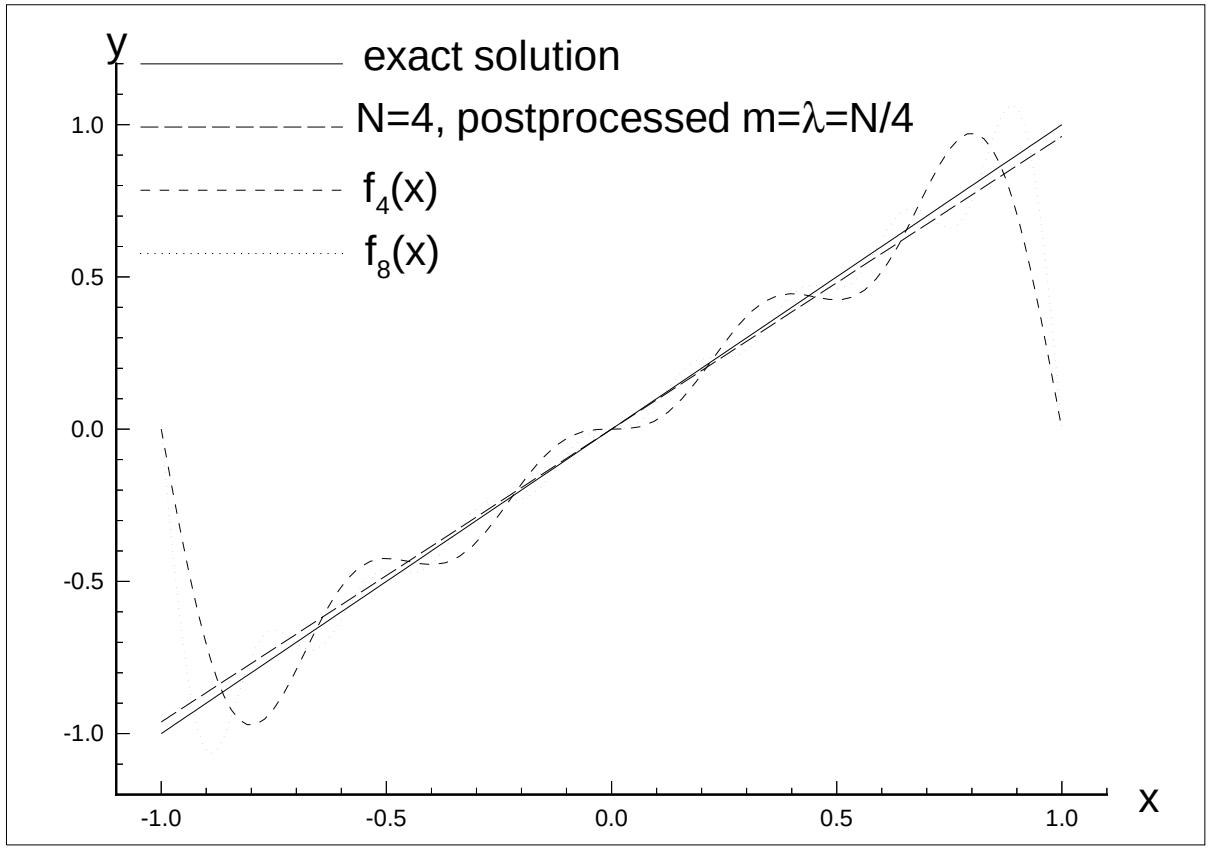


Figure 1: The function $f(x) = x$ (solid line); the Fourier partial sum $f_N(x)$ with $N = 4$ (short dashed line) and $N = 8$ (dotted line), and the approximation through the Gegenbauer procedure using $m = \lambda = N/4$ for $N = 4$ (long dashed line).

In the past 30 years, good progress has been made in designing high order numerical methods for solving hyperbolic equations even when the solutions are not smooth, or even discontinuous. These methods include (but are not restricted to):

- Spectral and other high order methods with suitable filters or post-processing;
- Weighted essentially non-oscillatory (WENO) finite difference and finite volume schemes;
- Discontinuous Galerkin (DG) finite element methods.

The state of the art high order, “high resolution” schemes, especially the WENO schemes and DG schemes (with suitable limiters), can achieve the following performance:

- high order accuracy convergence in smooth regions of the solution;
- sharp and essentially non-oscillatory shock transitions.

Additional Example 1: Simulation of astrophysical jet flows

To simulate the gas flows and shock wave patterns which are revealed by the Hubble Space Telescope images, one can implement theoretical models in a gas dynamics simulator. The two-dimensional model without radiative cooling is governed by the compressible Euler equations. The velocity of the gas flow is extremely high, and the Mach number could be hundreds or thousands.

An important property of the physical model of compressible Euler equations is that positivity-preserving property for density and pressure. That is, for the exact solution of the partial differential equations, if the initial condition has positive density and pressure, then the solution at later time also has positive density and pressure. The partial differential equations are ill-posed with negative density or pressure. That is, if either the density or pressure becomes negative, a small perturbation will be amplified without bound for later time, leading to the failure of numerical algorithms.

A big challenge for the computation of this astrophysical jet flow is, even for a state-of-the-art high order schemes, negative pressure could appear since the internal energy is very small compared to the huge kinetic energy (Ha, Gardner, Gelb and Shu, Journal of Scientific Computing 2005). One would then need to add additional numerical viscosity, hence lower the computational accuracy, in order to stablize the numerical algorithm.

In (Zhang and Shu, Journal of Computational Physics 2010), mathematical analysis leads to the design of suitable weak limiter, which maintains the originally designed high order accuracy of the discontinuous Galerkin methods, yet can guarantee the numerical solution to retain positive density and pressure.

First, we compute a Mach 80 (i.e. the Mach number of the jet inflow is 80 with respect to the soundspeed in the jet gas) problem without the radiative cooling.

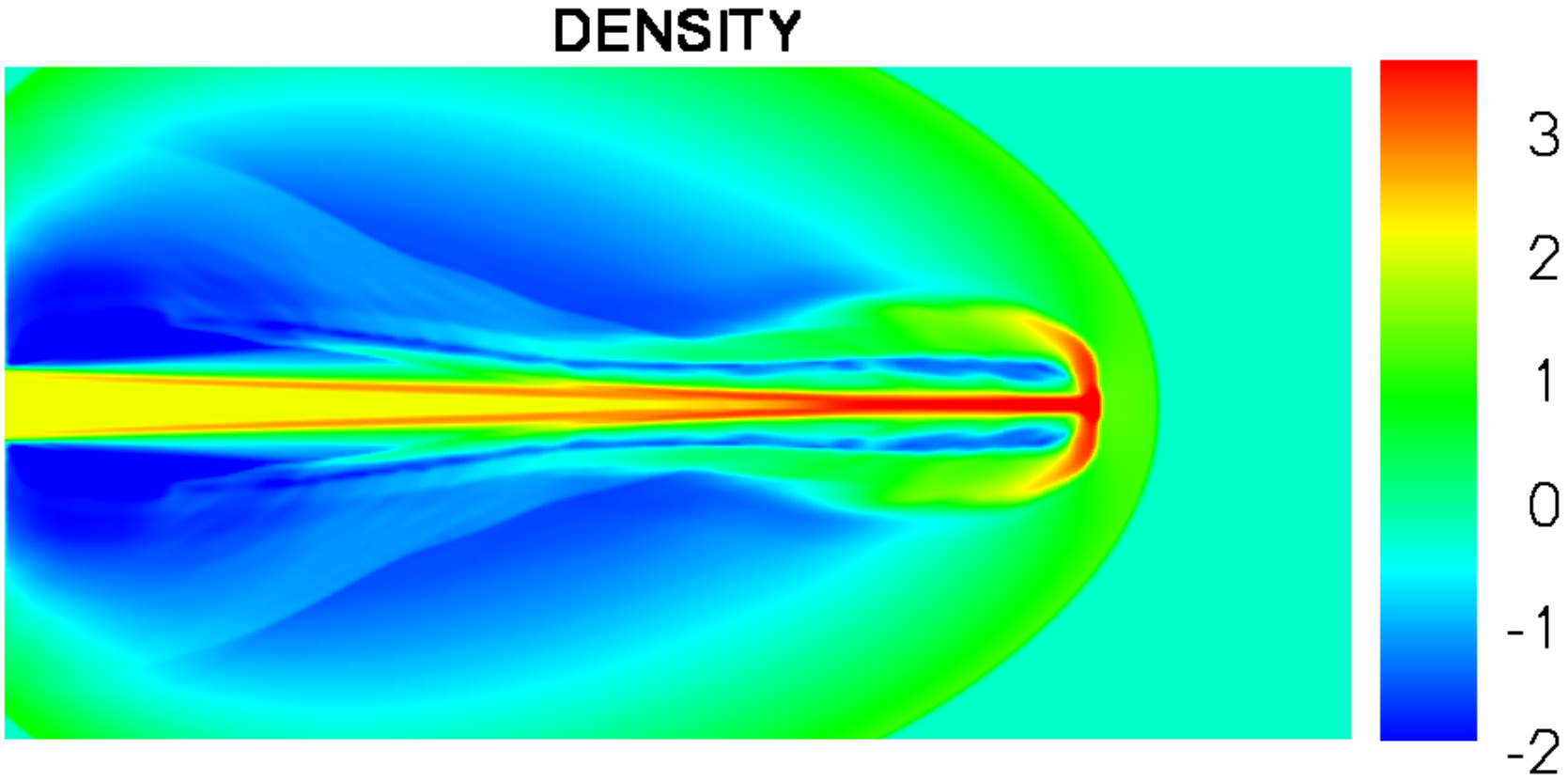


Figure 2: Simulation of Mach 80 jet without radiative cooling. Scales are logarithmic. Density.

Second, to demonstrate the robustness of our method, we compute a Mach 2000 problem. The domain is $[0, 1] \times [0, 0.5]$. The width of the jet is 0.1. The terminal time is 0.001. The speed of the jet is 800, which is around Mach 2100 with respect to the soundspeed in the jet gas.

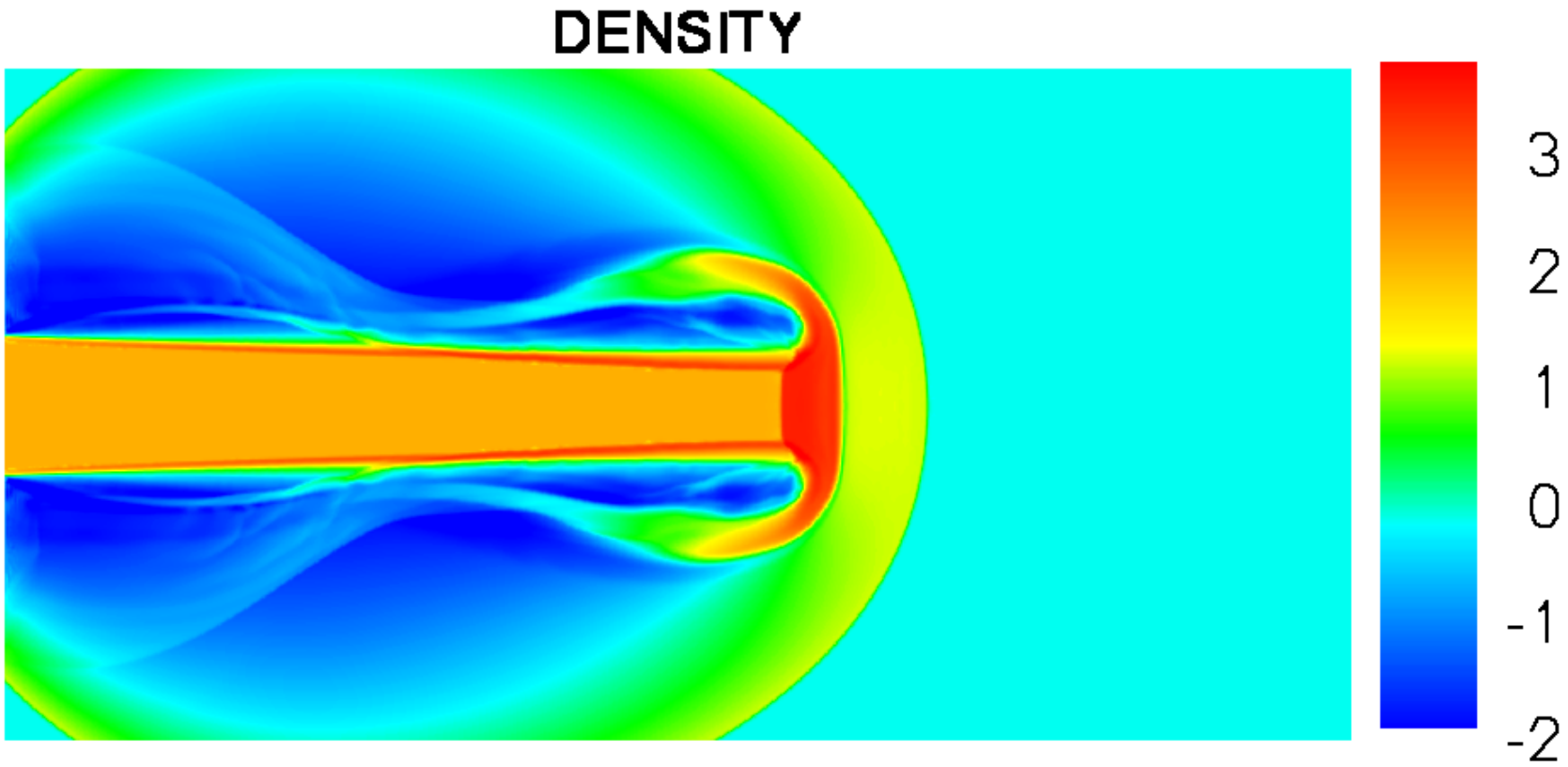


Figure 3: Simulation of Mach 2000 jet without radiative cooling. Scales are logarithmic. Density.

Lastly, we compute a Mach 80 (i.e. the Mach number of the jet inflow is 80 with respect to the soundspeed in the jet gas) problem with the radiative cooling to test the positivity-preserving property with the radiative cooling source term.

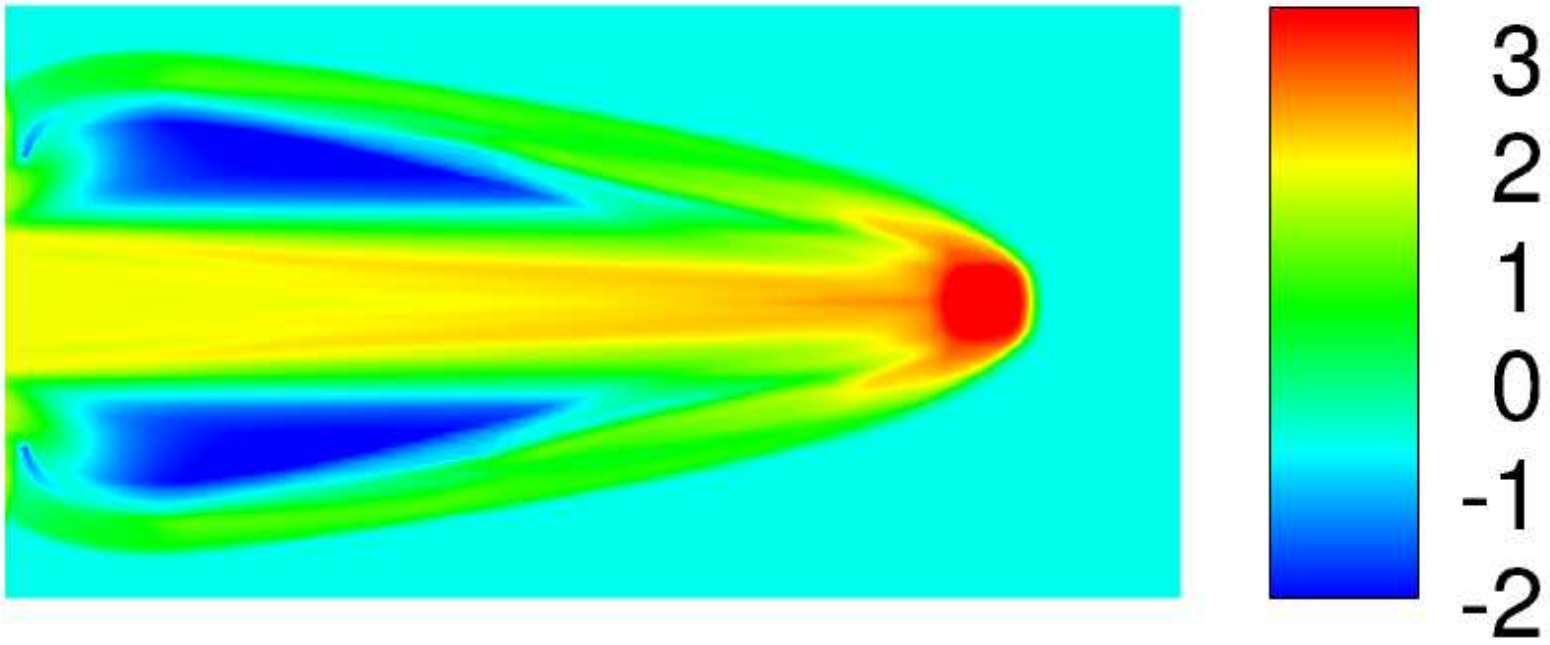
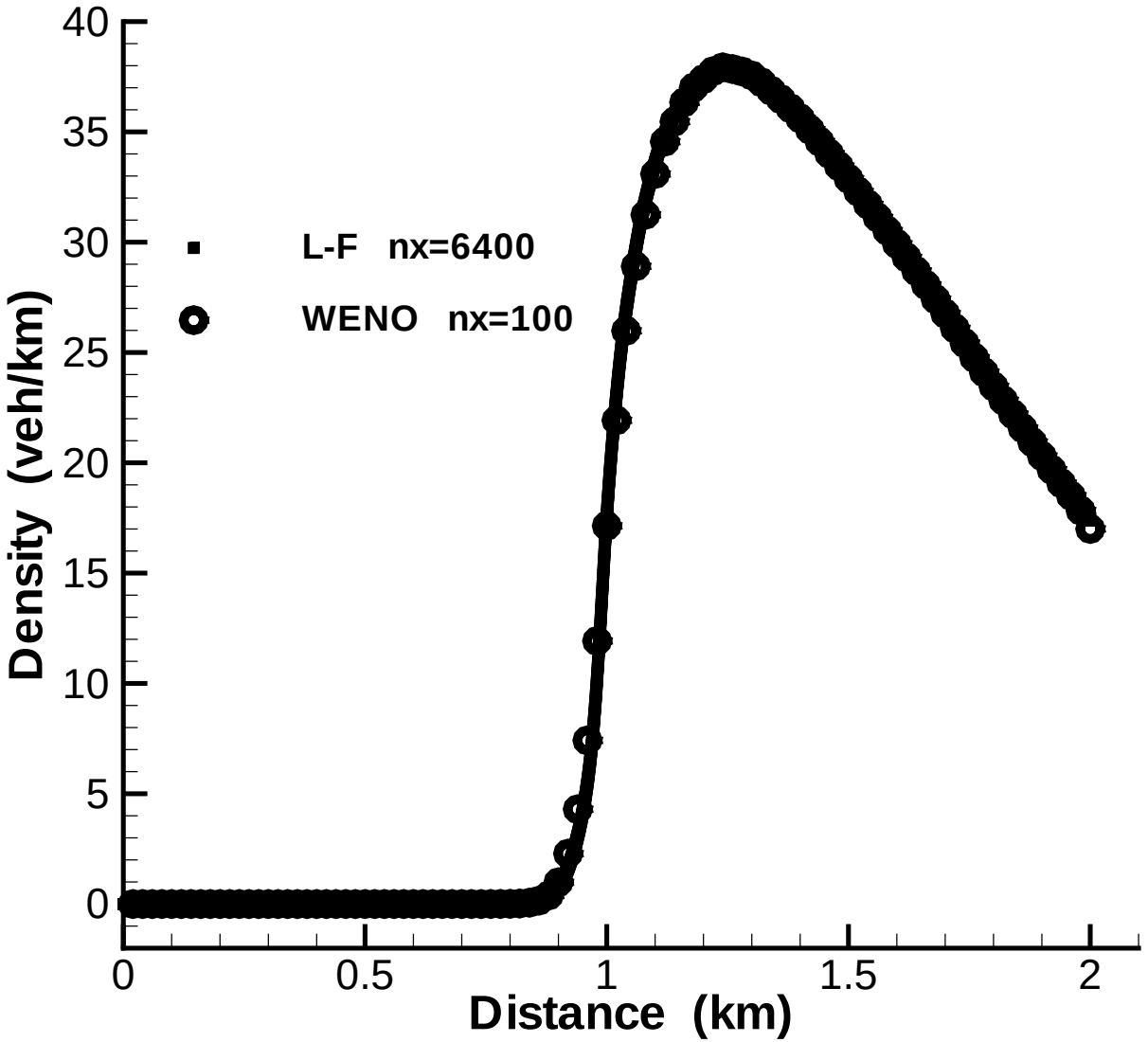


Figure 4: Simulation of Mach 80 jet with radiative cooling. The third order positivity-preserving RKDG scheme with the TVB limiter. Scales are logarithmic. Density.

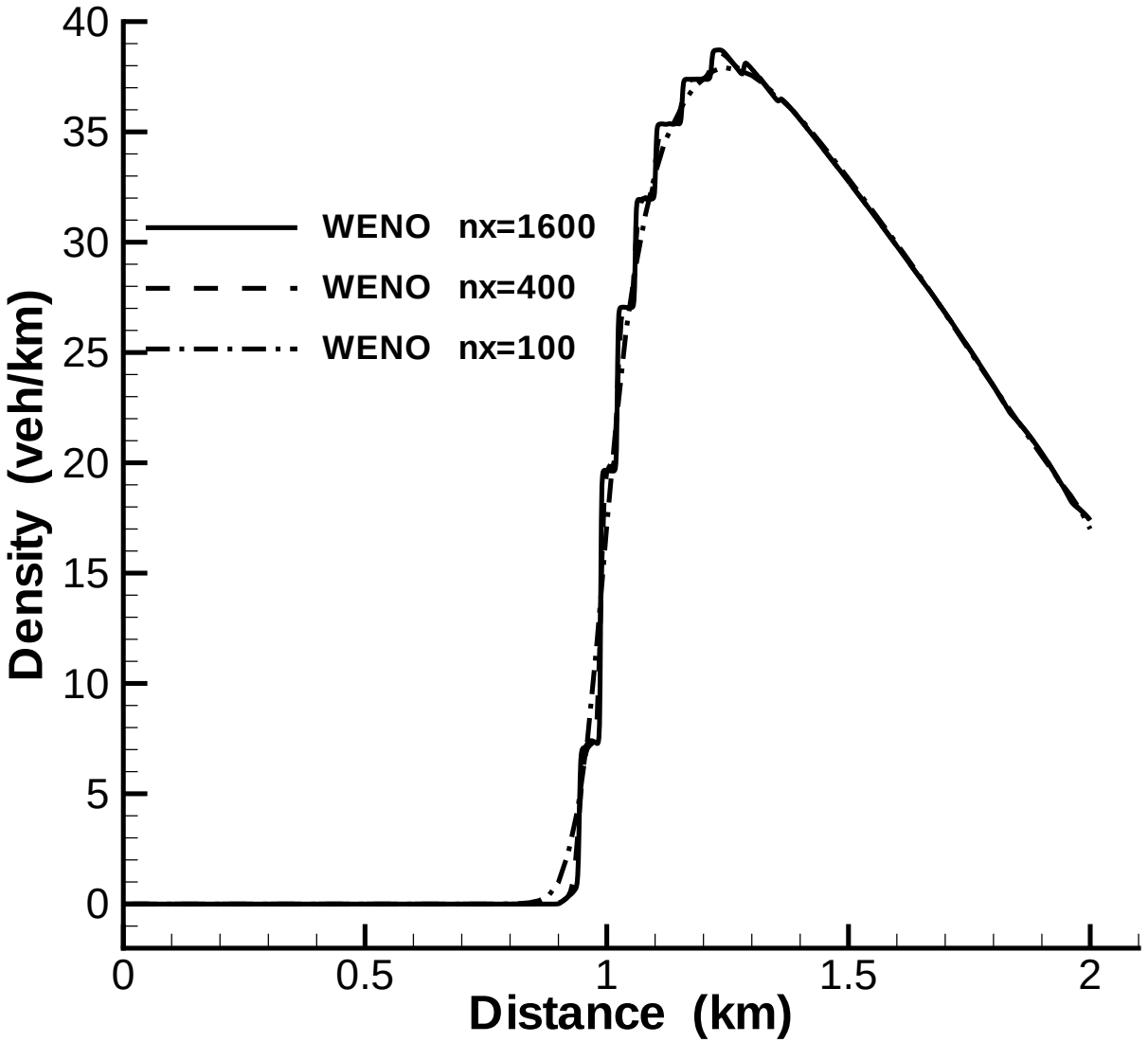
Additional Example 2: Simulation of a multi-class LWR traffic flow model

The mathematical model governing traffic flows with multiple vehicle types is a system of nonlinear hyperbolic partial differential equations, which admit solutions with discontinuities.

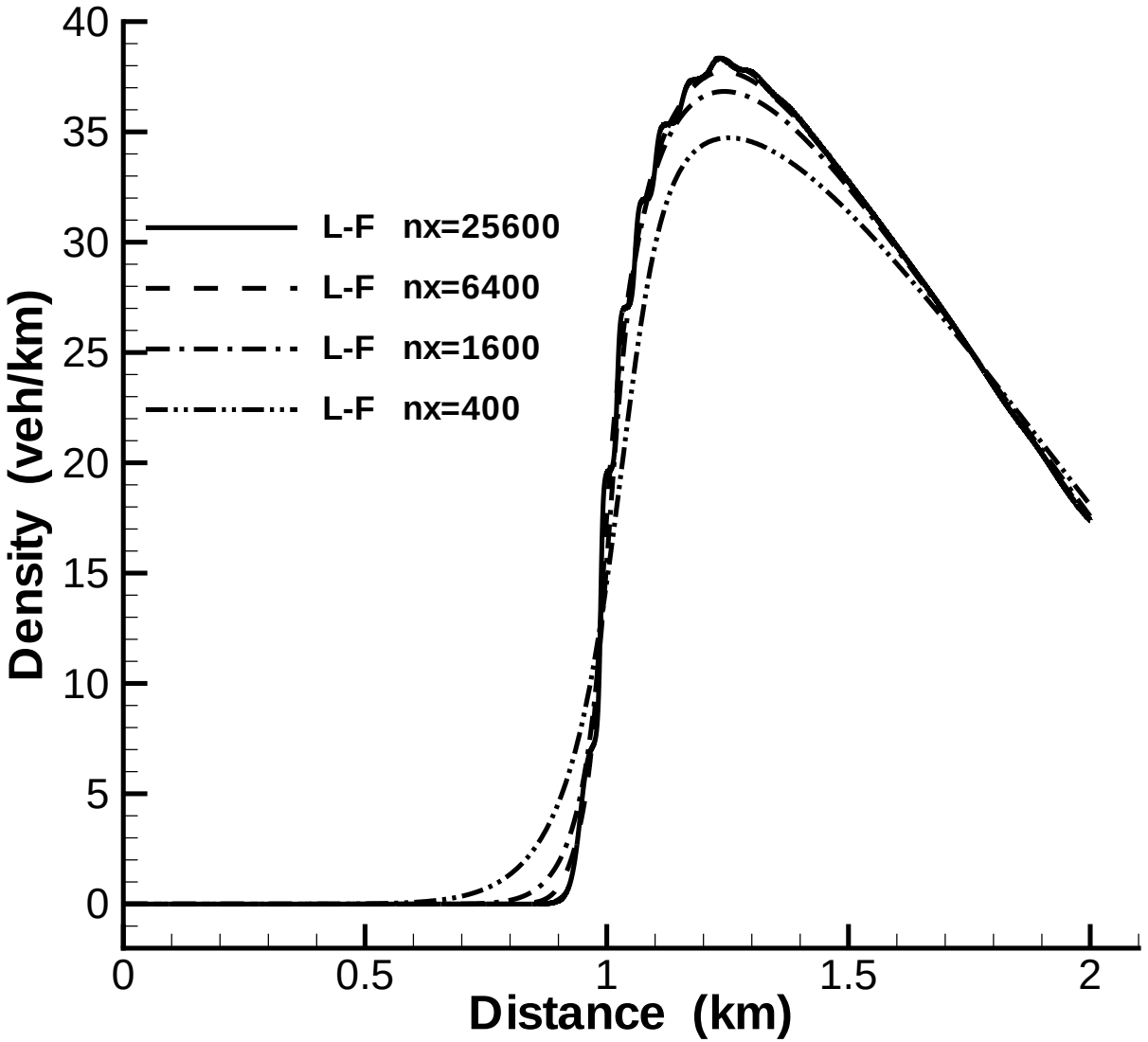
In Zhang, Shu, Wong and Wong, Journal of Computational Physics 2003, numerical simulation is performed for this traffic flow model. If one uses a first order accurate numerical algorithm with 6400 grid points (a very refined mesh in engineering computation), one obtain the following solution (which looks like the true solution, as it seems to be the numerically converged solution when one uses 400, 800, 1600, 3200 and 6400 grid points). A high order weighted essentially non-oscillatory (WENO) scheme using 100 grid points also gives similar results.


L-F nx=6400
WENO nx=100
Density (veh/km)
Distance (km)
0
5
10
15
20
25
30
35
40
0
0.5
1
1.5
2

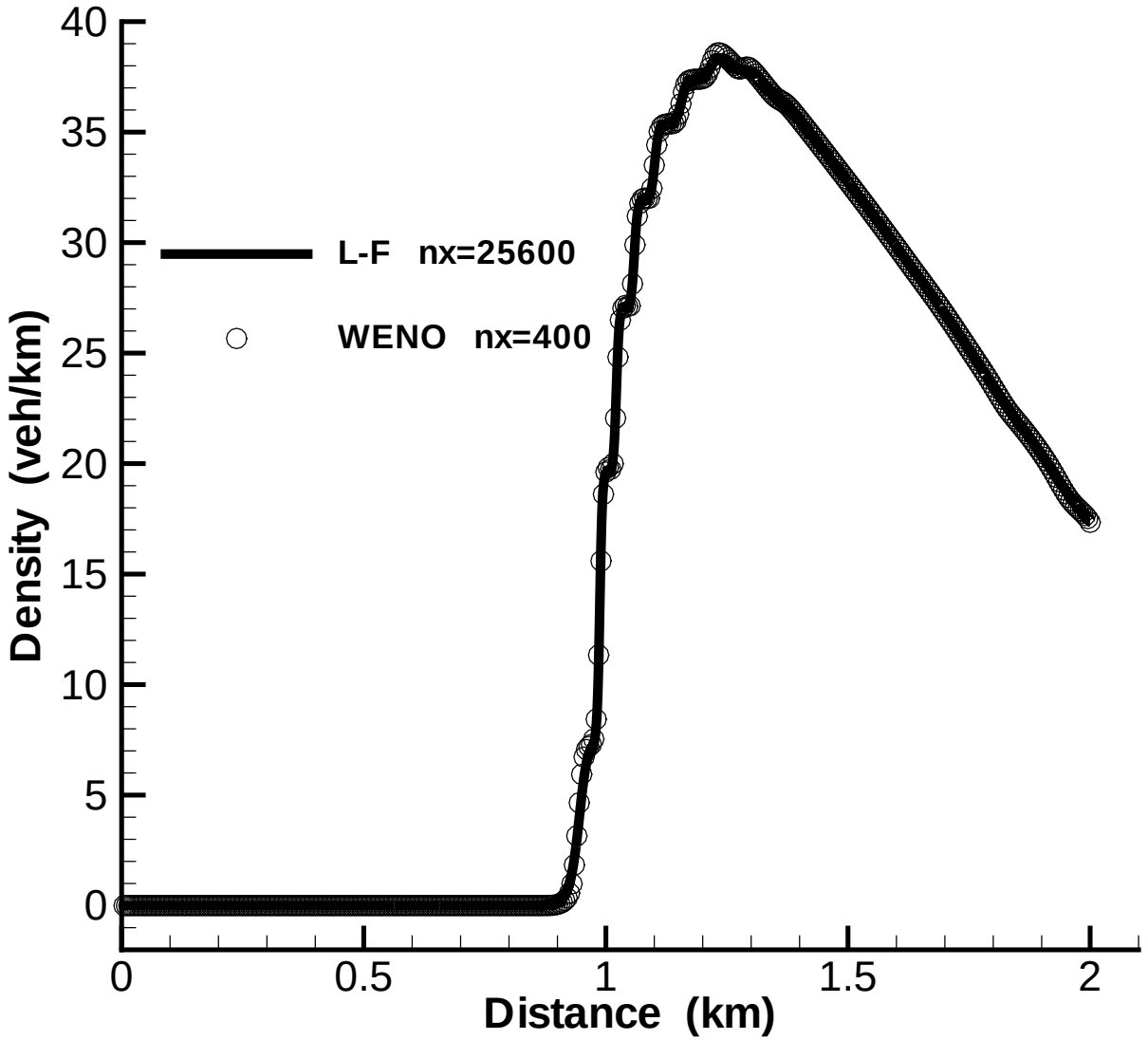
However, if one uses more grid points using the mathematically more accurate WENO schemes, one observes “staircase” solution structure which has been completely missed by the less accurate first order scheme with even 6400 points:


40
35
30
25
20
15
10
5
0
Density (veh/km)
WENO nx=1600
WENO nx=400
WENO nx=100
0
0.5
1
1.5
2
Distance (km)

Mathematically, the less accurate first order scheme should still converge to the correct solution with further mesh refinement, which is indeed the case when the computation is performed on extremely refined meshes reaching 25600 grid points:


L-F nx=25600
L-F nx=6400
L-F nx=1600
L-F nx=400
Density (veh/km)
Distance (km)
0
5
10
15
20
25
30
35
40
0
0.5
1
1.5
2

Finally, we can see that the solution of the high order WENO scheme with 400 grid points achieves comparable resolution as the first order scheme with 25600 grid points. Of course the high order scheme spends much less computational time:


L-F nx=25600
WENO nx=400
Density (veh/km)
Distance (km)
0
5
10
15
20
25
30
35
40
0
0.5
1
1.5
2

Concluding remarks

Mathematics can play an essential role in scientific computing:

- It can help the design of efficient algorithms.
- It can help to verify the reliability (stability and accuracy) of numerical algorithms.
- It has played a major role in the recent exploding development of machine learning and artificial intelligence.