Numerical Relativity 2023-2024

Homework 1 & 2

Malvina Bellotti
867169
Email: m.bellotti10@campus.unimib.it
Github with exercises: github.com/malvibellotti/numerical_rel

July 14, 2025

Contents

Chapter 1

Homework 1

1.1 Advection Equation

The 1D advection equation is given by:

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0 \tag{1.1}$$

where a is the constant advection velocity.

I studied the temporal evolution of two different initial conditions: a Gaussian profile and a step function.

Gaussian initial condition:

$$u(x, t = 0) = \exp[-(x - x_0)^2], \quad x_0 = 5 \tag{1.2}$$

Step function initial condition:

$$u(x, 0) = \begin{cases} 0, & x < 4 \text{ or } x > 6 \\ 1, & 4 \leq x \leq 6 \end{cases} \tag{1.3}$$

For the numerical approximation, I used a domain $x \in [0, 10]$ with periodic boundary conditions, $J = 101$ grid points, and Courant factor $c_f = a \cdot \Delta t/\Delta x = 0.5$. I implemented and compared four different finite difference schemes to approximate the solution evolution:

- FTCS (Forward Time, Centered Space):

$$u_j^{n+1} = u_j^n - \frac{c_f}{2}(u_{j+1}^n - u_{j-1}^n) \tag{1.4}$$

- Lax-Friedrichs:

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - \frac{c_f}{2}(u_{j+1}^n - u_{j-1}^n) \tag{1.5}$$

- Leapfrog:

$$u_j^{n+1} = u_j^{n-1} - c_f(u_{j+1}^n - u_{j-1}^n) \tag{1.6}$$

- Lax-Wendroff:

$$u_j^{n+1} = u_j^n - \frac{c_f}{2}(u_{j+1}^n - u_{j-1}^n) + \frac{c_f^2}{2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \tag{1.7}$$

With:

J	number of grid points
c_f	Courant factor, $c_f = \frac{a \cdot \Delta t}{\Delta x}$
dx	spatial grid spacing, $dx = \frac{L}{J-1}$ where $L = 10$
dt	time step size, calculated given cf and dx
u_j^n	numerical solution at grid point j and time step n

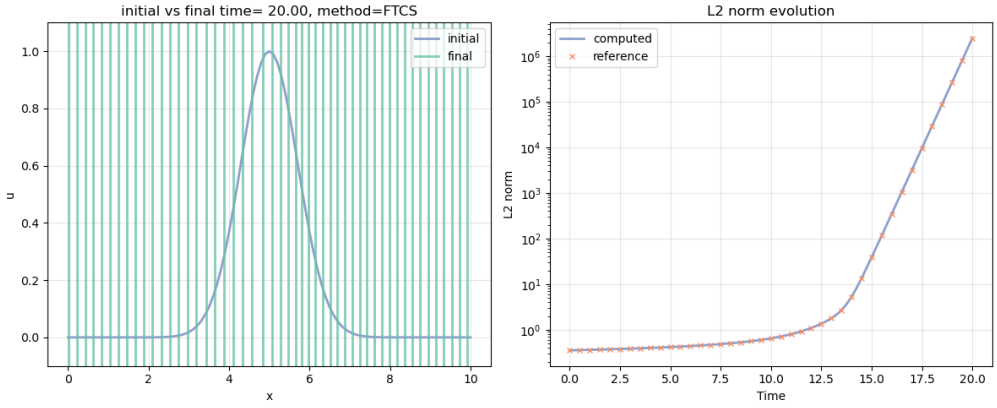
An important check for numerical schemes is the tracking of the L2norm evolution, since it should be a conserved quantity, and therefore should remain constant during the integration.

The L2 norm of a discrete solution is defined as:

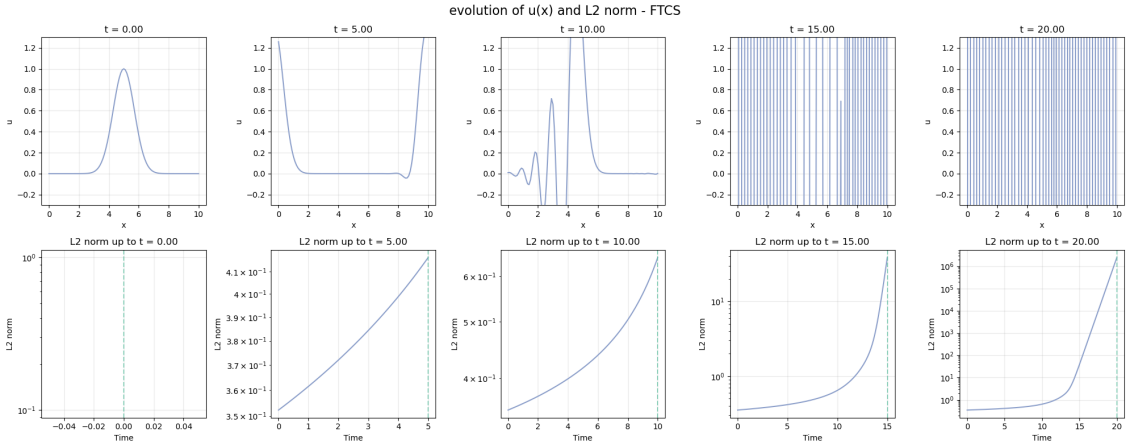
$$\|u^n\|_2 = \sqrt{\sum_{j=1}^{J} (u_j^n)^2} \tag{1.8}$$

1.1.1 Gaussian profile

FTCS



(a) Comparison between initial u(x) profile vs final u(x) profile, at time t = 20. On the right L2norm evolution compared with stored data.



(b) FTCS time evolution, showing growing oscillations.

The FTCS scheme shows unstable behavior for the advection equation. Oscillations that grow exponentially in time can be observed and are also represented in the increase of the L2 norm. The Courant factor was fixed at $c_f = 0.5$.

To analyze the stability of the FTCS (Forward-Time Central-Space) scheme for the 1D advection equation, we can apply the Von Neumann stability test.

The FTCS scheme is:

$$u_j^{n+1} = u_j^n - \frac{a\Delta t}{\Delta x}\left(u_{j+1}^n - u_{j-1}^n\right), \tag{1.9}$$

Assuming $u_j^n = \xi^n e^{ikx_j}$, and substituting it we get:

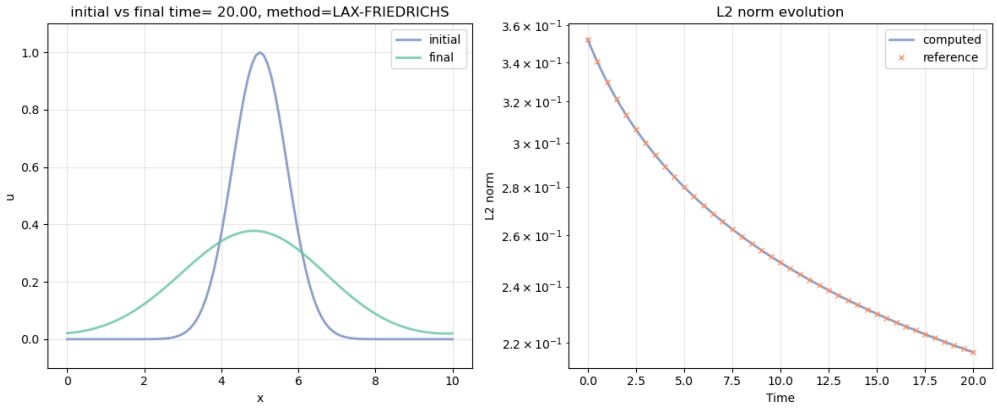
$$\xi = 1 - i\frac{a\Delta t}{\Delta x}\sin\left(\frac{k\Delta x}{2}\right).$$

and therefore:

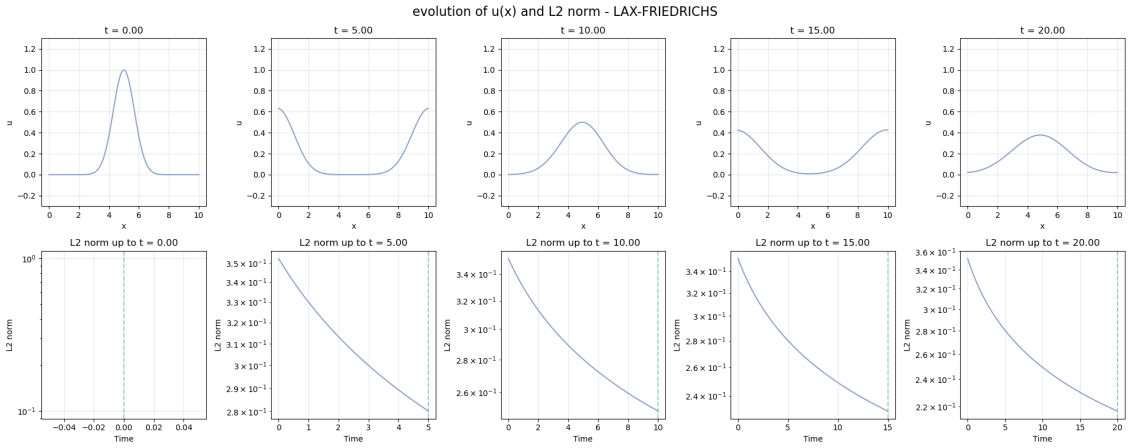
$$|\xi|^2 = 1 + \left[\frac{a\Delta t}{\Delta x}\sin\left(\frac{k\Delta x}{2}\right)\right]^2 > 1 \tag{1.10}$$

The oscillations are, in fact, amplified.

Lax-Friedrichs



(a) Comparison between initial u(x) profile vs final u(x) profile, at time t = 20. On the right L2norm evolution compared with stored data.



(b) Lax-Friedrichs time evolution: stable but dissipative behavior.

Stable for $c_f \leq 1$, but highly dissipative. The gaussian initial condition, instead of maintaining its shape, decreases in amplitude over time.

This behavior can be understood doing a Taylor expansion of the numerical scheme, which shows that the Lax-Friedrichs method approximates the advection equation with an additional dissipative term (numerical viscosity), which s responsible for the damping of the solution:

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = \frac{1}{2}\frac{|a|}{C_f}\Delta x \frac{\partial^2 u}{\partial x^2} + o(\Delta x^2). \tag{1.11}$$

The dissipative term is proportional to Δx, this means the scheme is consistent and convergent as the error vanishes linearly when $\Delta x \to 0$.

We can apply the Von Neumann stability test to recover the condition for which the scheme is stable.

The Lax-Friedrichs scheme is:

$$u_j^{n+1} = \frac{1}{2}\left(u_{j+1}^n + u_{j-1}^n\right) - \frac{a\Delta t}{2\Delta x}\left(u_{j+1}^n - u_{j-1}^n\right), \tag{1.12}$$

Assuming $u_j^n = \xi^n e^{ikx_j}$ and substituting into the scheme, we get:

$$\xi^{n+1}e^{ikx_j} = \frac{1}{2}\xi^n\left(e^{ikx_{j+1}} + e^{ikx_{j-1}}\right) - \frac{a\Delta t}{2\Delta x}\xi^n\left(e^{ikx_{j+1}} - e^{ikx_{j-1}}\right).$$

Using the identities

$$\cos(k\Delta x) = \frac{e^{ik\Delta x} + e^{-ik\Delta x}}{2}, \quad \sin(k\Delta x) = \frac{e^{ik\Delta x} - e^{-ik\Delta x}}{2i},$$

we find:

$$\xi = \cos(k\Delta x) - i\frac{a\Delta t}{\Delta x}\sin(k\Delta x).$$

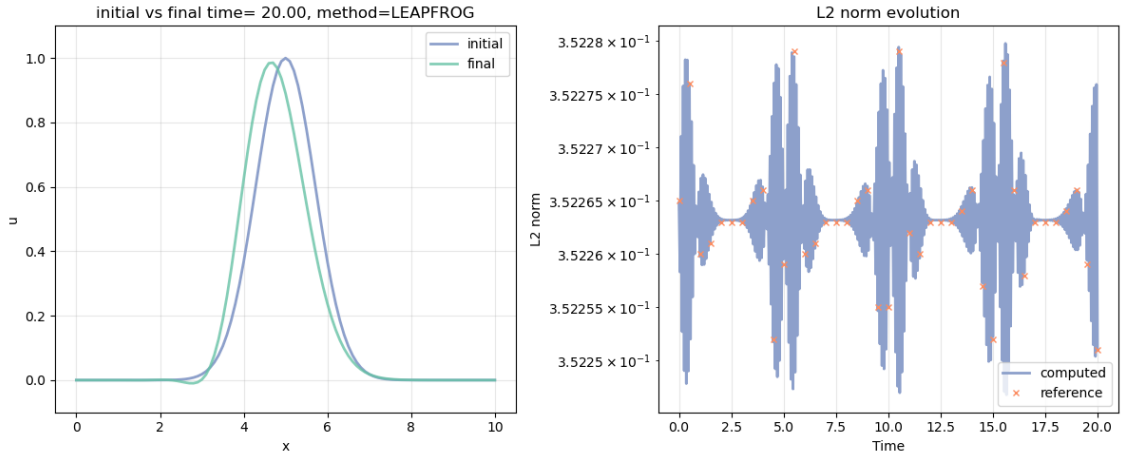
Therefore:

$$|\xi|^2 = \cos^2(k\Delta x) + \left(\frac{a\Delta t}{\Delta x}\right)^2 \sin^2(k\Delta x). \tag{1.13}$$

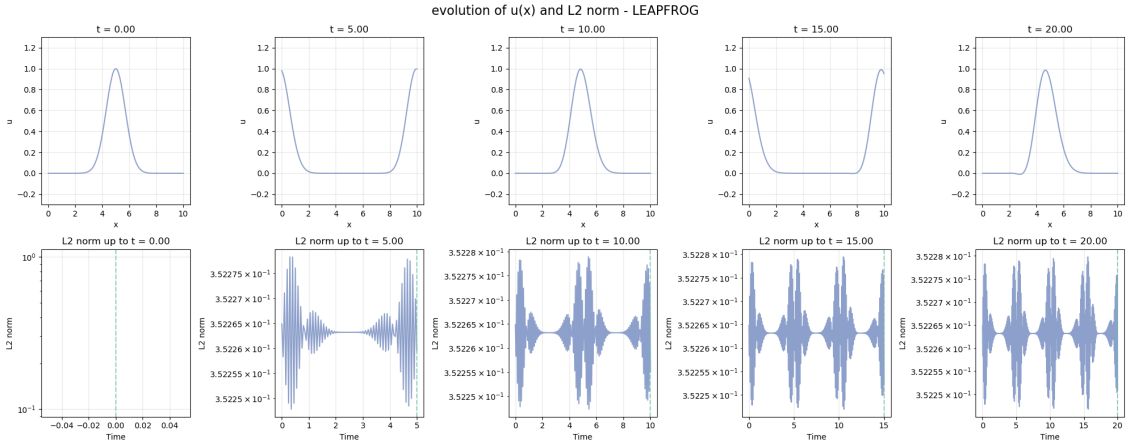
which leads to the Courant–Friedrichs–Lewy (CFL) condition:

$$|\xi|^2 \leq 1 \quad if \quad c_F = \frac{a\Delta t}{\Delta x} \leq 1. \tag{1.14}$$

Leapfrog



(a) Comparison between initial u(x) profile vs final u(x) profile. On the right L2norm evolution compared with stored data.

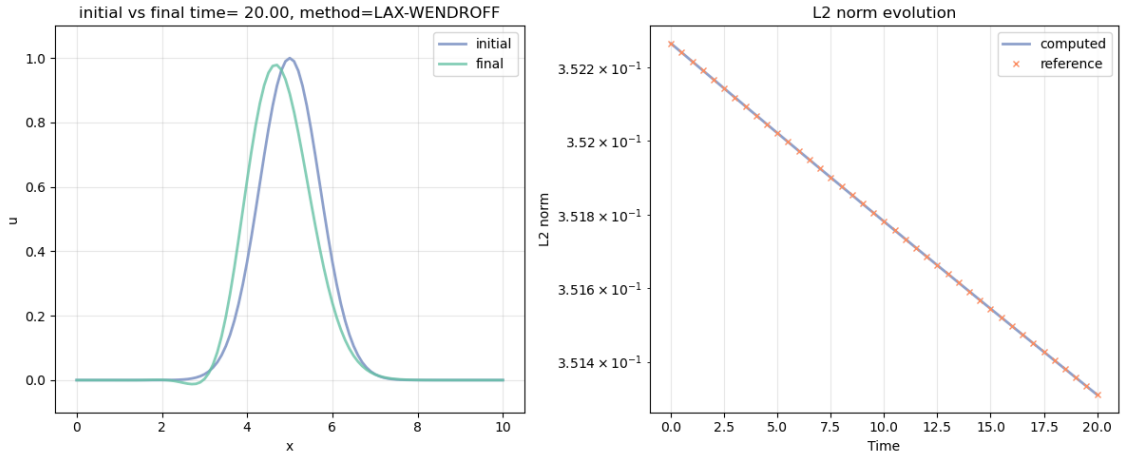


(b) Leapfrog temporal evolution showing the scheme's non-dissipative nature.

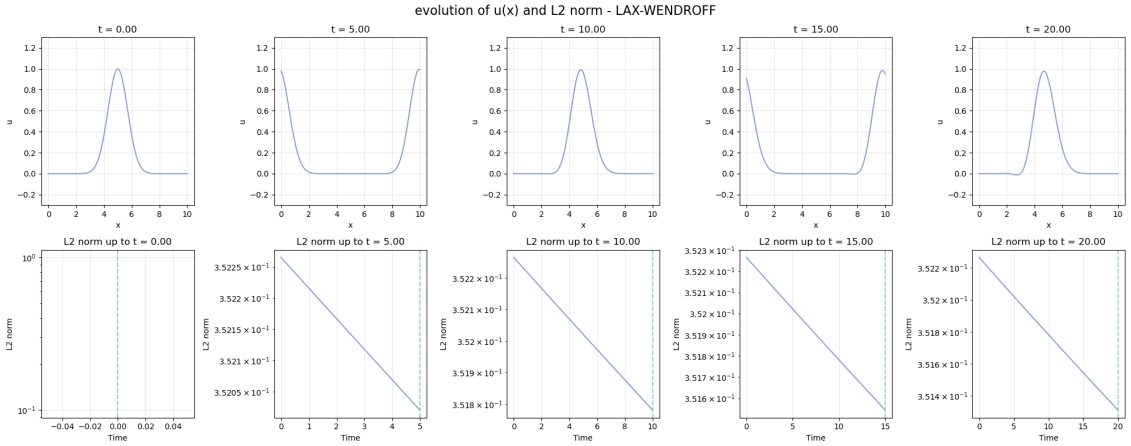
Leapfrog is a second order method that requires the computation of three time steps.

This method is stable for $c_f \leq 1$ and non-dissipative, in fact initial and final profiles are very similar. The L2-norm remains almost constant, with small oscillations.

Lax-Wendroff



(a) Comparison between initial u(x) profile vs final u(x) profile. On the right L2norm evolution compared with stored data.



(b) Lax-Wendroff temporal evolution showing minimal amplitude decay but a slight change in the L2norm that grows over time.

Lax-Wendroff is a combination of the Leapfrog and Lax-Friedrichs schemes, it is a second order scheme but requires the computation of only two time steps.

Looking at the Taylor expansion we can recognise both a dispersive term (3rd order) and a dissipative term (4th order). The dispersive term is therefore dominant.

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = -\frac{a\Delta x^2}{6}\left(1 - \frac{a^2\Delta x^2}{\Delta t^2}\right)\frac{\partial^3 u}{\partial x^3} + \frac{a^2\Delta t\Delta x^2}{24}\frac{\partial^4 u}{\partial x^4}\left(1 - \frac{a^2\Delta x^2}{\Delta t^2}\right) + o(\Delta x^4). \tag{1.15}$$

Comparison between methods

Below I show a performance comparison between the four schemes I used (FTCS, Lax-Friedrichs, Lax-Wendroff, and Leapfrog). To do this, I fixed the Courant factor to $c_F = 0.5$ and used $J = 101$ spatial grid points. The initial condition is a Gaussian profile.

The top-left panel shows the final solutions compared to the initial profile (dashed line). FTCS is unstable, while Lax-Friedrichs shows strong diffusion. Lax-Wendroff and Leapfrog are both more accurate.

The top-right panel plots the evolution of the L_2 norm over time. FTCS diverges exponentially, confirming its instability. The other methods remain stable, but with different behaviors: Leapfrog conserves the norm within 10^{-5}, while Lax-Friedrichs dissipates it over time.

The bottom-left panel zooms in on the L_2 norm evolution to highlight differences between the stable methods. The bottom-right panel shows the absolute error $|L_2(t) - L_2(0)|$, where Leapfrog achieves the highest accuracy, followed by Lax-Wendroff.

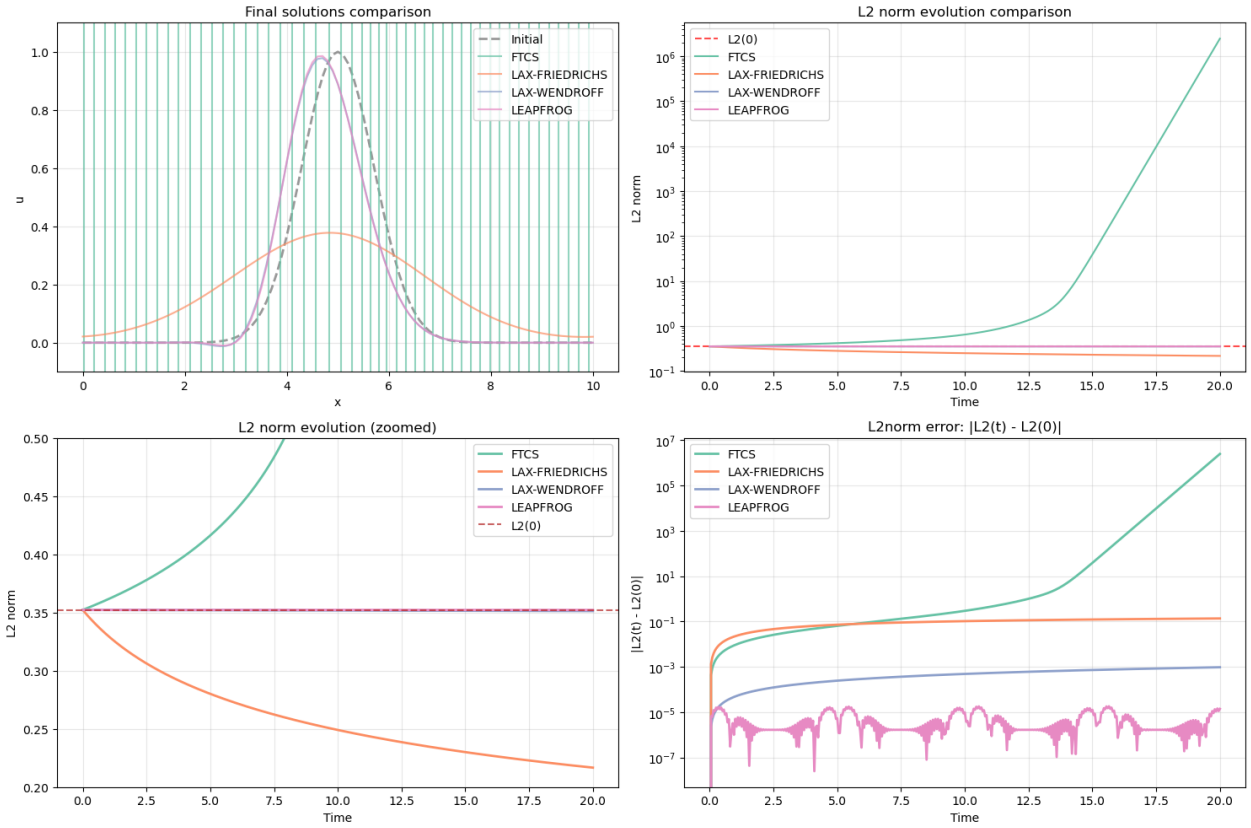


Figure 1.5: Performance comparison of four numerical schemes for the advection equation with a gaussian profile as initial condition. Top left: Final solutions. Top right: L_2 norm evolution. Bottom left: Zoom on L_2 norm. Bottom right: Absolute L_2 norm error.

Changing Courant factor and resolution

In this section I show the results of changing the Courant factor and the number of point in the spatial grid:

- $c_F = [0.1, 0.5, 0.9, 1.0, 1.1]$
- $J = [51, 101, 201, 401]$

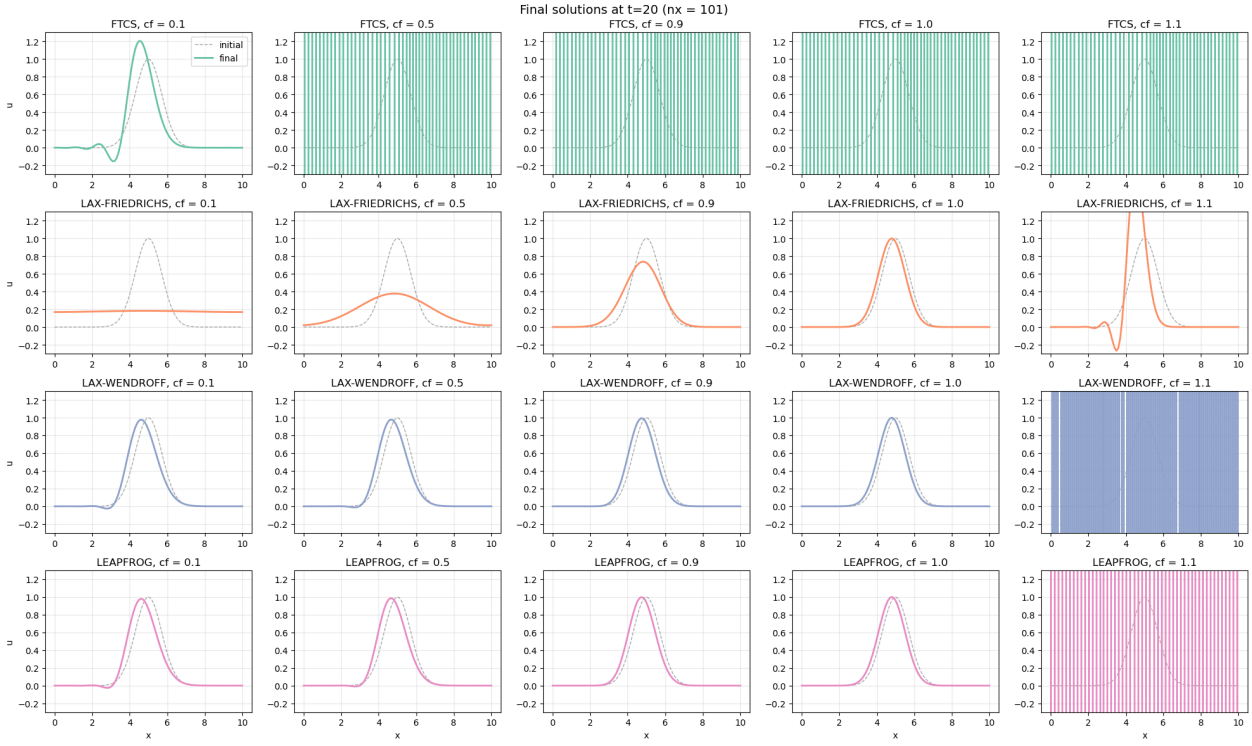


Figure 1.6: Impact of varying Courant factor (c_F) in the four numerical schemes.

Different CFL values affect solution accuracy and stability: lower c_F increase dissipation in schemes like Lax-Friedrichs, while higher values improve accuracy.

FTCS methos is instable, regardless of the Courant factor, its intabilities grow with the increase of c_F .

For all the schemes, when $c_f > 1$, violating the CFL condition (eq 1.14), we observe instabilities.

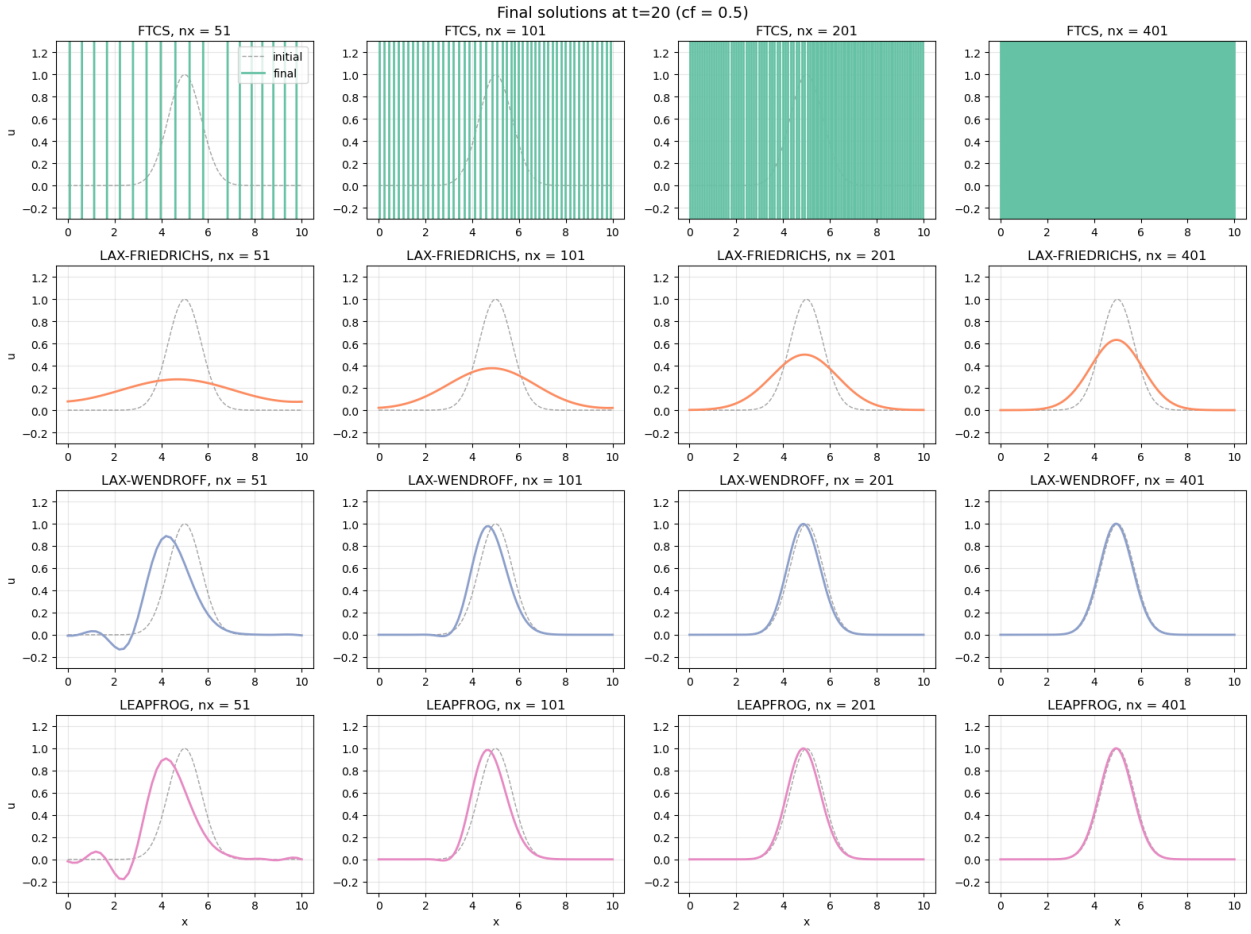
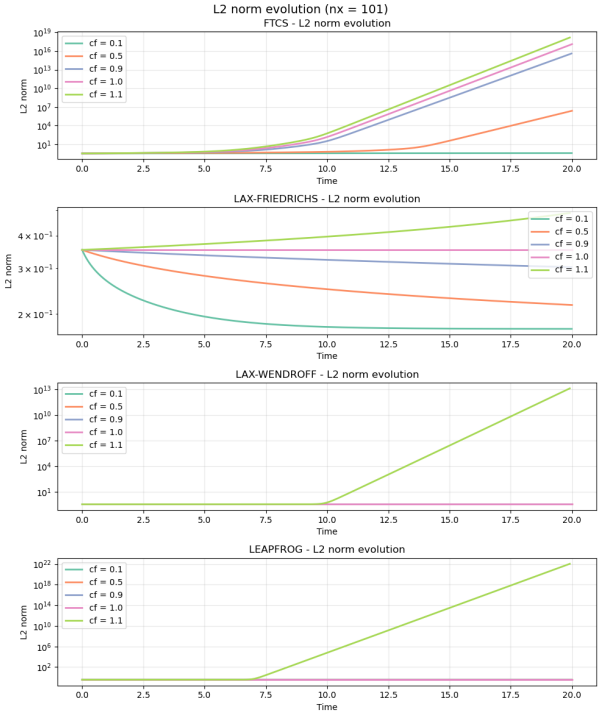


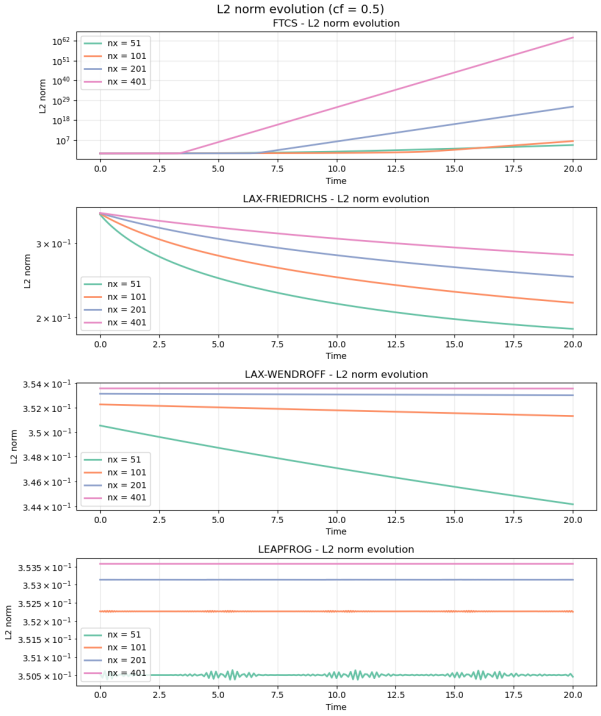
Figure 1.7: Impact of varying grid resolution (number of points in the spatial grid: J) in the four numerical schemes. Kepping a fixed c_F.

As the grid resolution increases Lax-Friedrichs shows gradual improvement and reduced numerical diffusion (error goes as Δx), Lax-Wendroff reaches convergence faster (second order scheme: error goes as Δx^2) .

Stable schemes tend to converge to the exact solution.



(a) L2 norm evolution for all the schemes changing Courant factor.



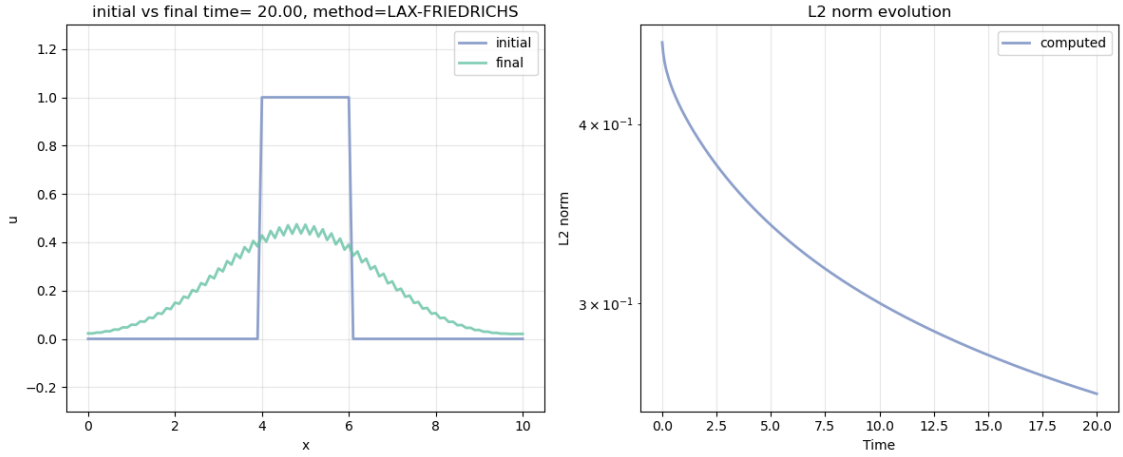
(b) L2 norm evolution for all the schemes changing grid resolution.

Figure 1.8: Behaviour of the L2norms for all the schemes as I changed c_F and the number of points J.

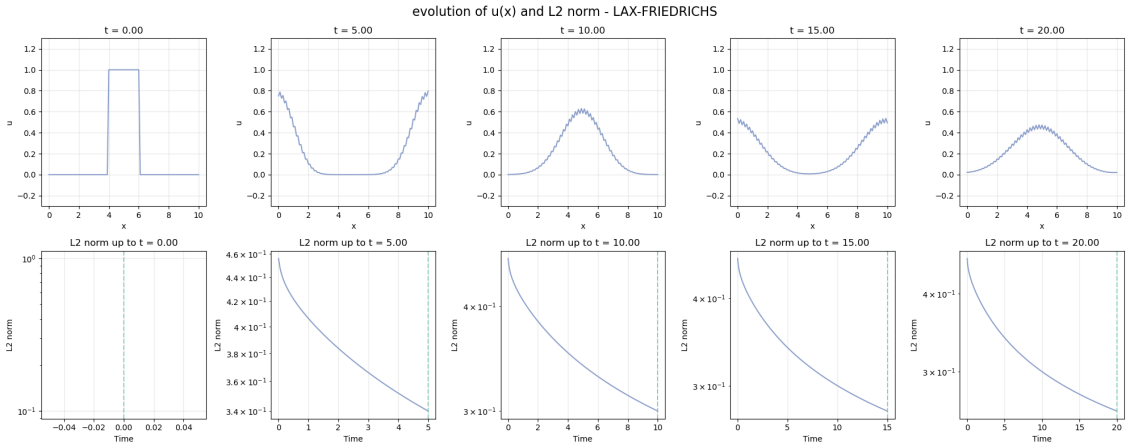
1.1.2 Step function

I then changed the initial condition to the step function profile, so that we can analyze the finite difference schemes' behavior near discontinuities.

Lax-Friedrichs



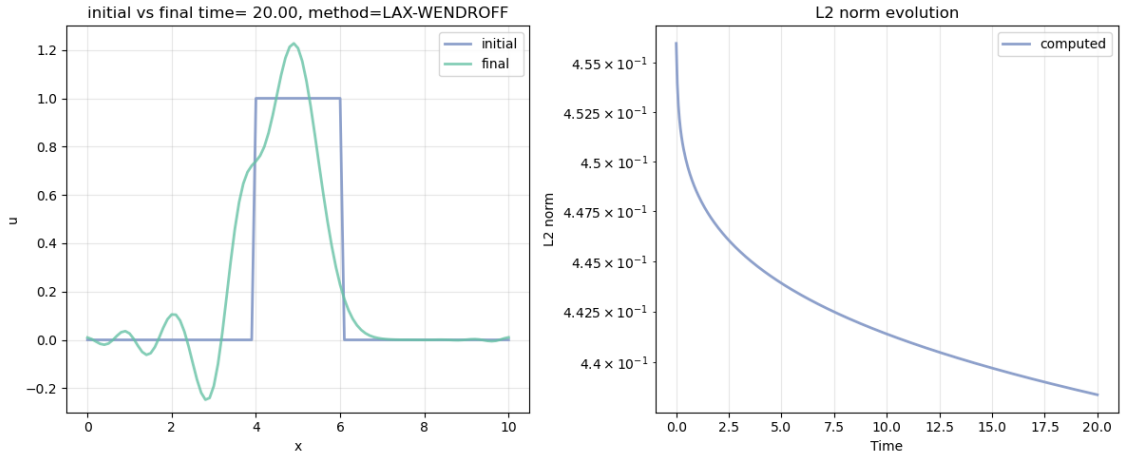
(a) Comparison between initial u(x) profile vs final u(x) profile. On the right L2norm evolution.



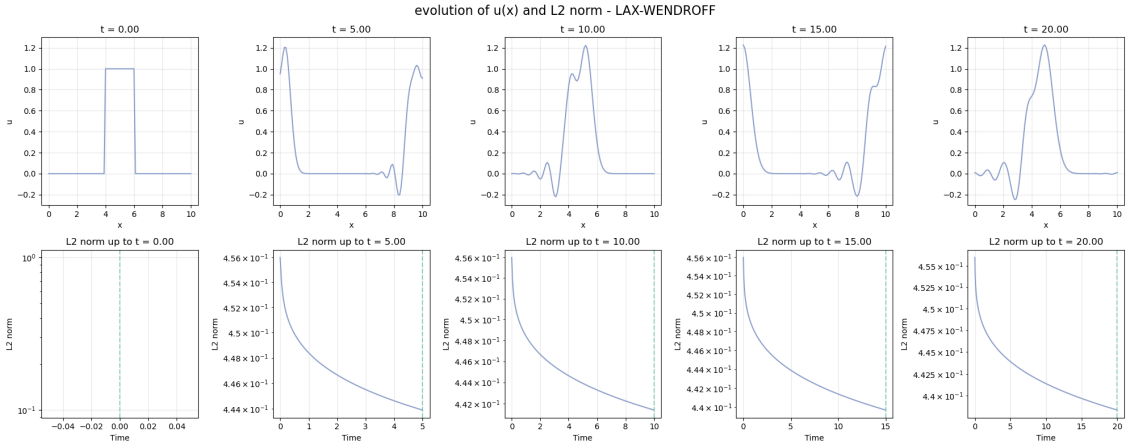
(b) Lax-Friedrichs scheme time evolution, showing smoothing of discontinuity.

We can observe a heavy smoothing of the discontinuity's sharp edges because of the scheme's dissipative nature.

Lax-Wendroff



(a) Lax-Wendroff comparison between initial u(x) profile vs final u(x) profile. On the right L2norm evolution.



(b) Lax-Wendroff temporal evolution showing the formation of oscillations near the discontinuities.

Unlike Lax-Friedrichs, the Lax-Wendroff scheme preserves better the sharpness of the discontinuity, but introduces oscillations near it.

Comparison between methods

I tested the four numerical schemes (FTCS, Lax-Friedrichs, Lax-Wendroff, and Leapfrog) on a discontinuous initial profile: a step function. The setup is the same as before, with $c_F = 0.5$ and $J = 101$.

The top right panel shows the final profile. FTCS diverges rapidly due to its known instability. Lax-Friedrichs remains stable but heavily smooths the discontinuity. Lax-Wendroff and Leapfrog both preserve the step better, though they show oscillations around the discontinuity.

The top left plot shows the evolution of the L_2 norm. FTCS grows exponentially, Lax-Friedrichs shows a decay due to numerical diffusion, Leapfrog and Lax-Wendroff maintain the norm almost constant.

The bottom left plot zooms in on the L_2 norm for the three stable schemes. The diffusive nature of Lax-Friedrichs is evident.

The bottom right panel shows the absolute error in L_2 norm relative to the initial value. Leapfrog and Lax-Wendroff are the most accurate of the four.

Comparison between methods

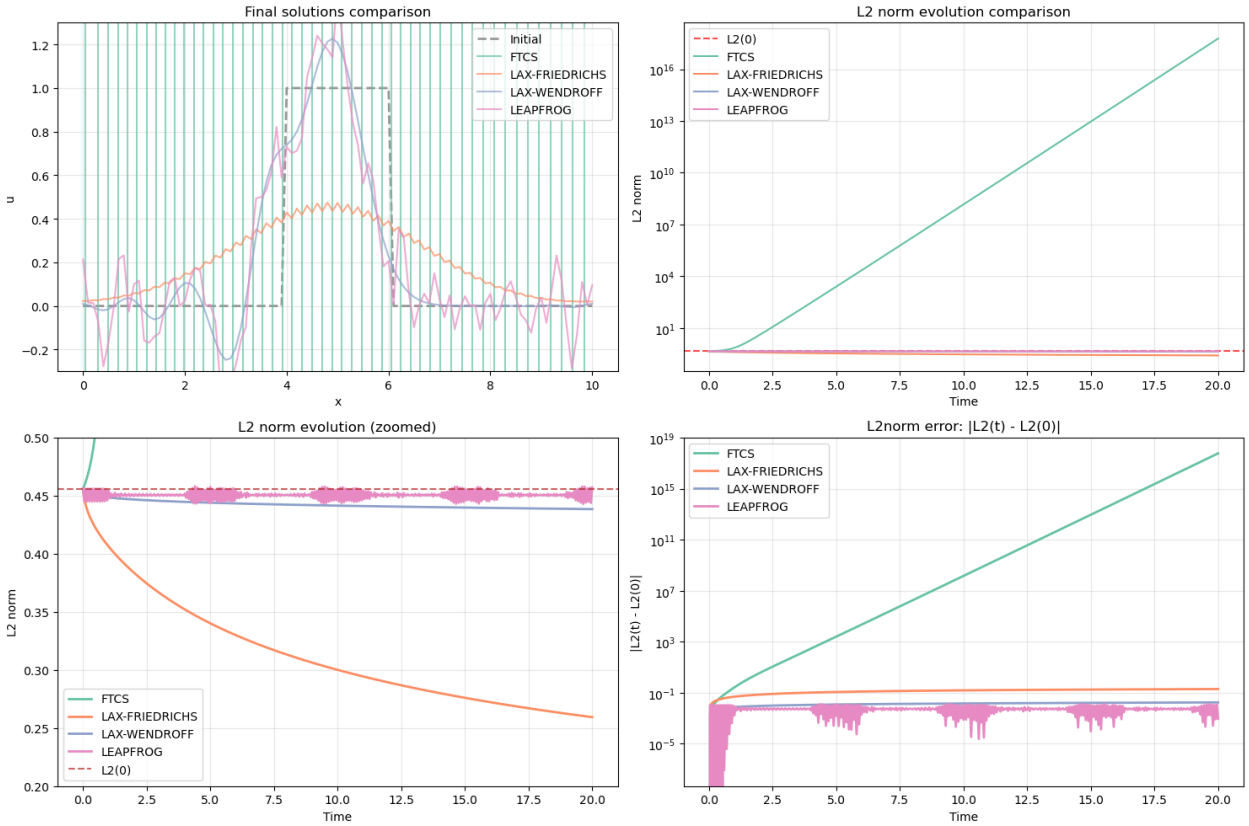


Figure 1.11: Performance comparison of four numerical schemes for the advection equation with a step-function initial condition. Top left: Final solutions. Top right: L_2 norm evolution. Botoom left: Zoom on L_2 norm. Bottom right: Absolute L_2 norm error.

Changing Courant factor and resolution

In this section I show the results of changing the Courant factor and the number of point in the spatial grid:

- $c_F = [0.1, 0.5, 0.9, 1.0, 1.1]$
- $J = [51, 101, 201, 401]$

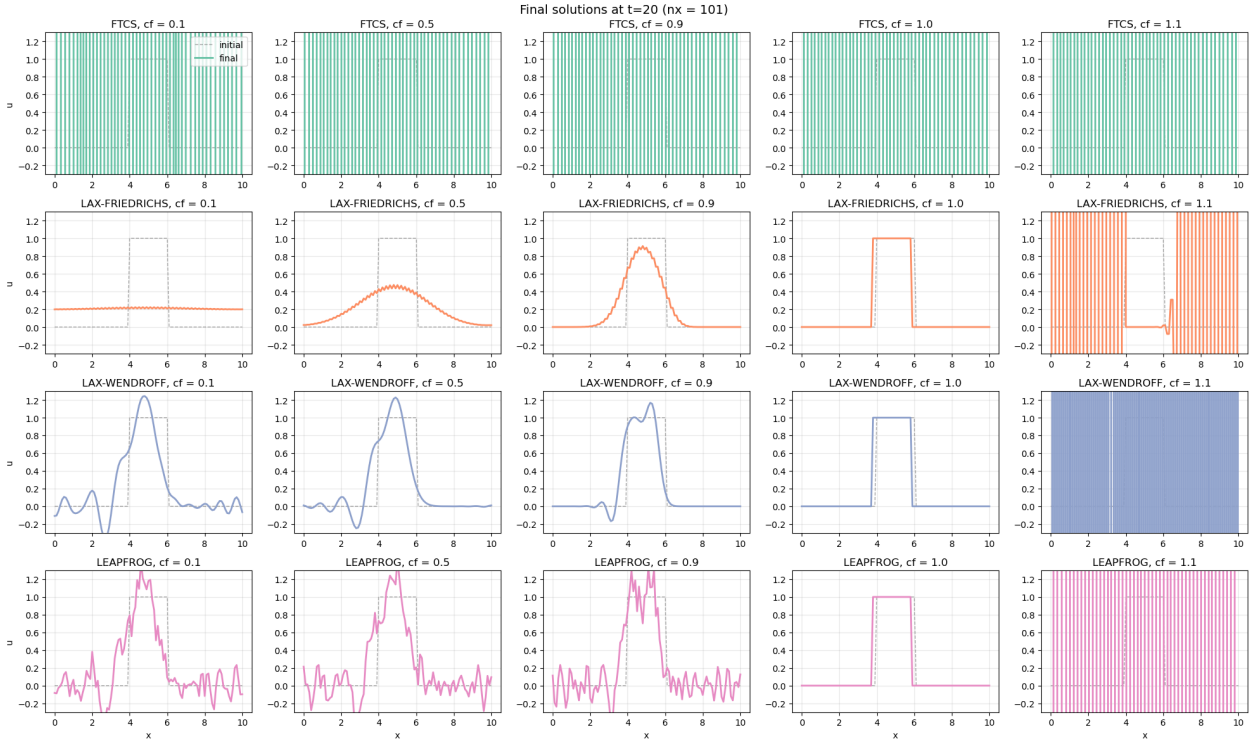


Figure 1.12: Impact of varying Courant factor (c_F) in the four numerical schemes, with step function as initial condition.

Reducing the Courant factor leads, in general, to stronger smoothing of discontinuities.

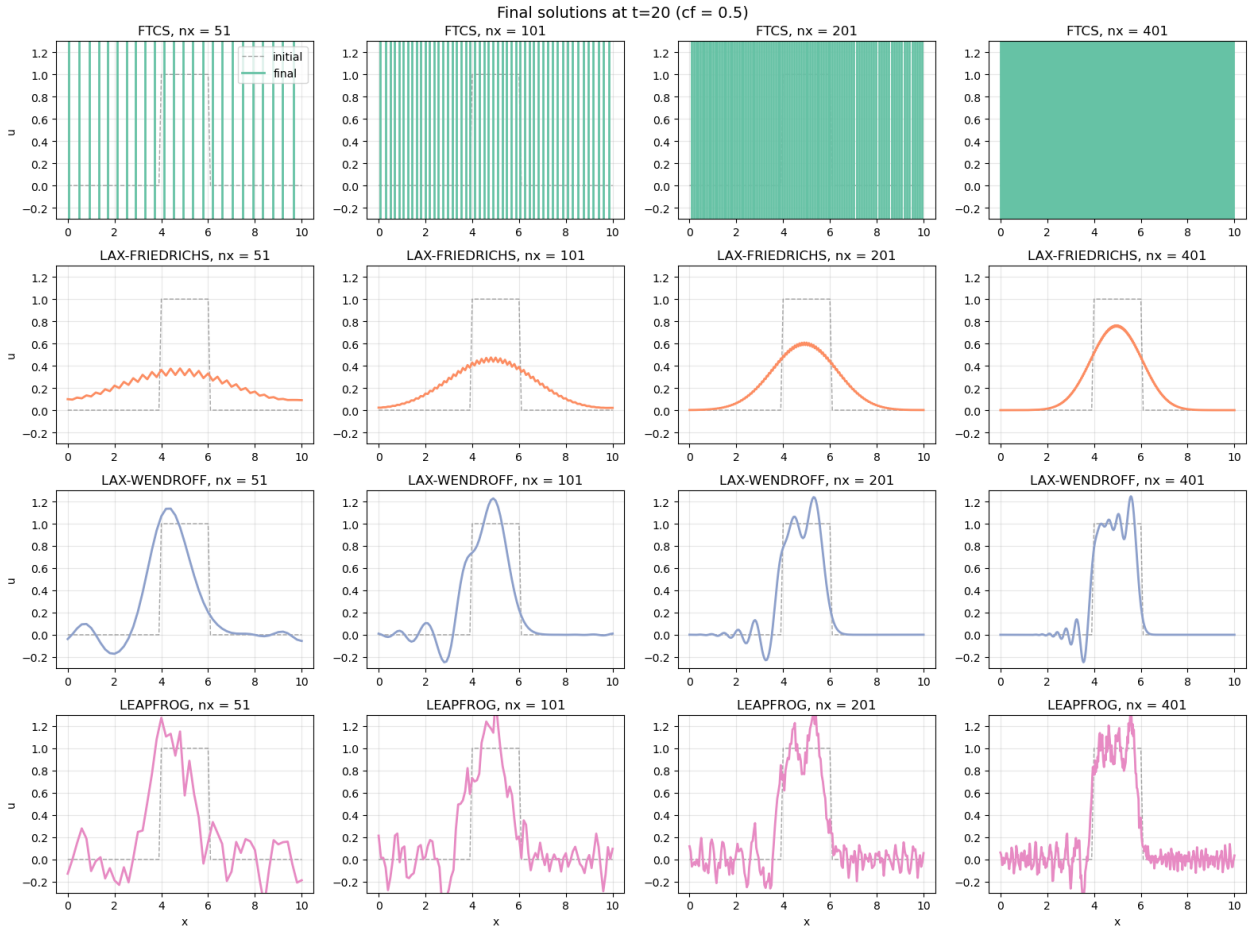
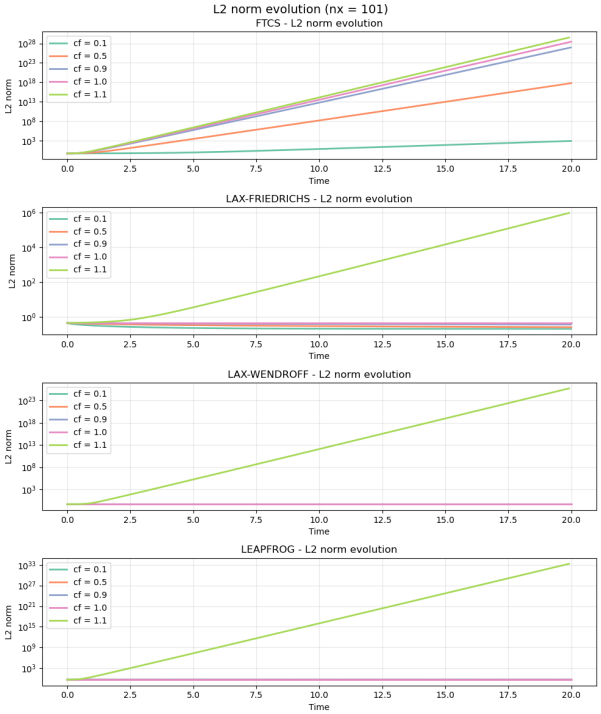
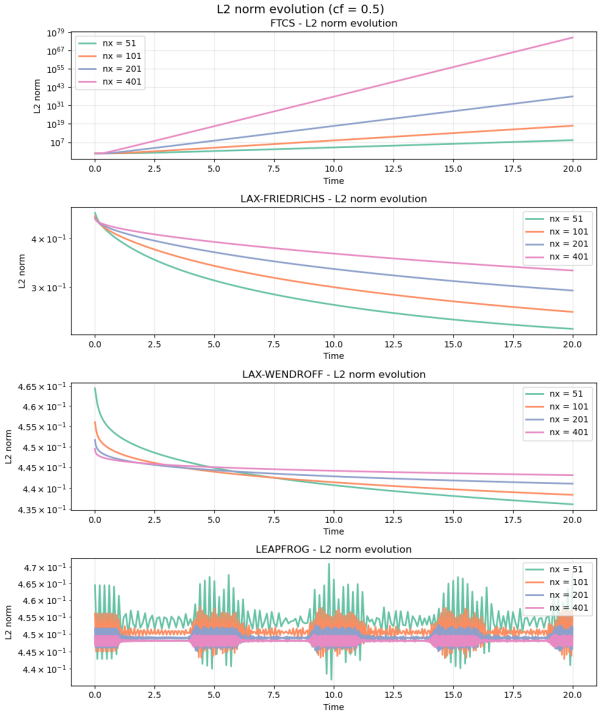


Figure 1.13: Impact of varying grid resolution ((number of point in the spatial grid: J) in the four numerical schemes.

Increasing the grid resolution improves the sharpness of the solution. In Lax-Friedrichs diffusive scheme the smoothing remains but the overall profile gets closer to the expected result. Oscillations in Lax-Wendroff become more localized. At low resolution, all schemes struggle to capture the discontinuity.



(a) L2 norm evolution for all the schemes changing Courant factor.



(b) L2 norm evolution for all the schemes changing grid resolution.

Figure 1.14: Behaviour of the L2norms for all the schemes as I changed c_F and J.

Neither scheme handles discontinuities perfectly. This motivates the need for more sophisticated schemes like Total Variation Diminishing Methods.

1.2 Burgers' Equation

The one-dimensional Burgers equation is a nonlinear hyperbolic conservation law, written as:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0. \tag{1.16}$$

Or, equivalently, in conservative form:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(f(u)\right) = 0, \tag{1.17}$$

with the flux $f = \frac{u^2}{2}$.

Here $u(x,t)$ is both the variable and the wave speed, which, unlike the advection case, will not be constant for every point of the profile. This leads to nonlinear behavior and the formation of a shock wave and a rarefaction wave.

I studied the time evolution of a gaussian initial profile using two numerical methods. The chosen initial condition is:

$$u(x, t=0) = 10\exp[-(x-x_0)^2], \quad x_0 = 5 \tag{1.18}$$

The simulation domain is $x \in [0, 10]$, with $J = 101$ grid points and periodic boundary conditions. The spatial resolution is $dx = L/(J-1)$, with $L = 10$. The time step dt is computed based on a fixed Courant factor $c_f = 0.5$ as:

$$dt = \frac{c_f \cdot dx}{\max(u)}.$$

I implemented and compared two finite difference upwind schemes, one in conservative form and one in non-conservative form, both with periodic boundary conditions:

- Upwind – Conservative form:

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}\left[f(u_j^n) - f(u_{j-1}^n)\right], \quad f(u) = \frac{1}{2}u^2. \tag{1.19}$$

- Upwind – Non-conservative form:

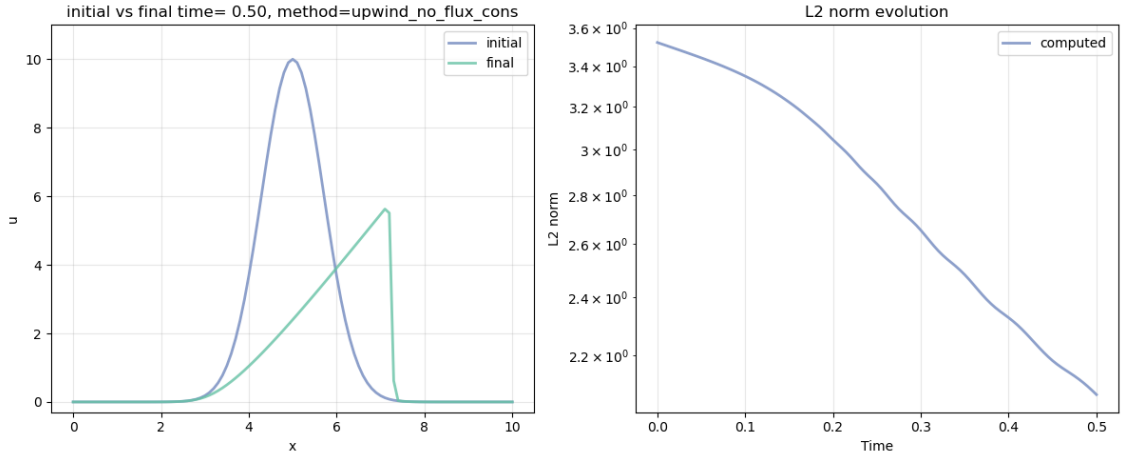
$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x} \cdot u_j^n \left(u_j^n - u_{j-1}^n\right). \tag{1.20}$$

To monitor the evolution, I also computed the L^2-norm of the solution at each time step:

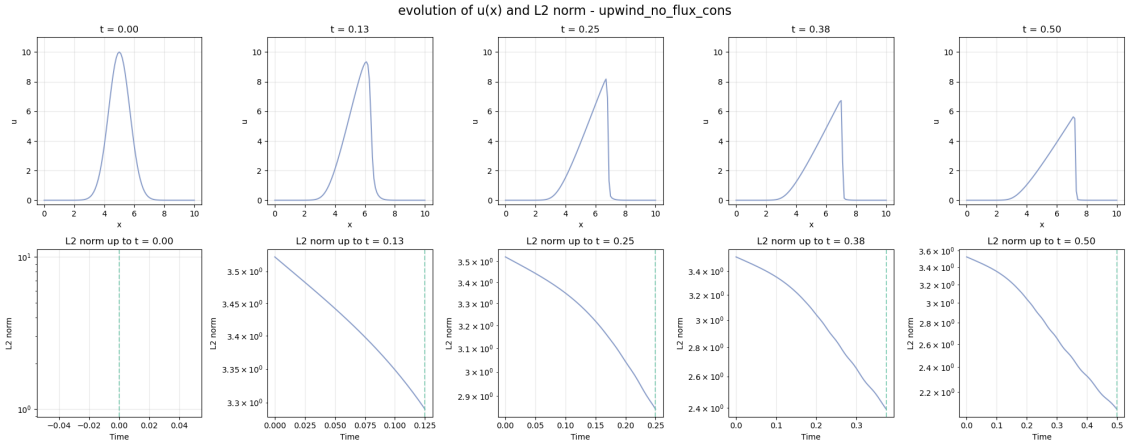
$$\|u\|_2 = \sqrt{\frac{1}{J}\sum_j u_j^2}.$$

1.2.1 Upwind (non flux conservative)

The gaussian profile steepens and eventually forms a shock.



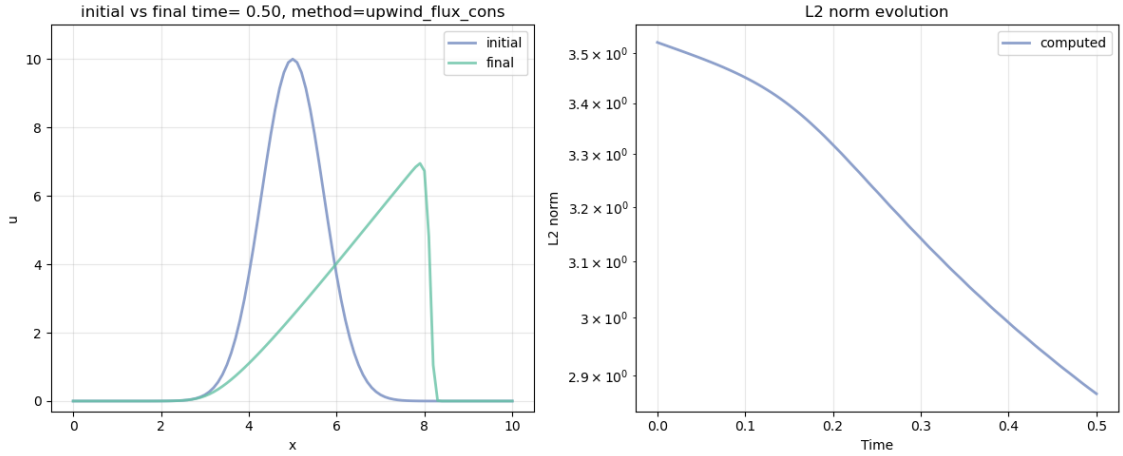
(a) Comparison between initial u(x) profile vs final u(x) profile, at time t = 0,4. On the right L2norm evolution in time.



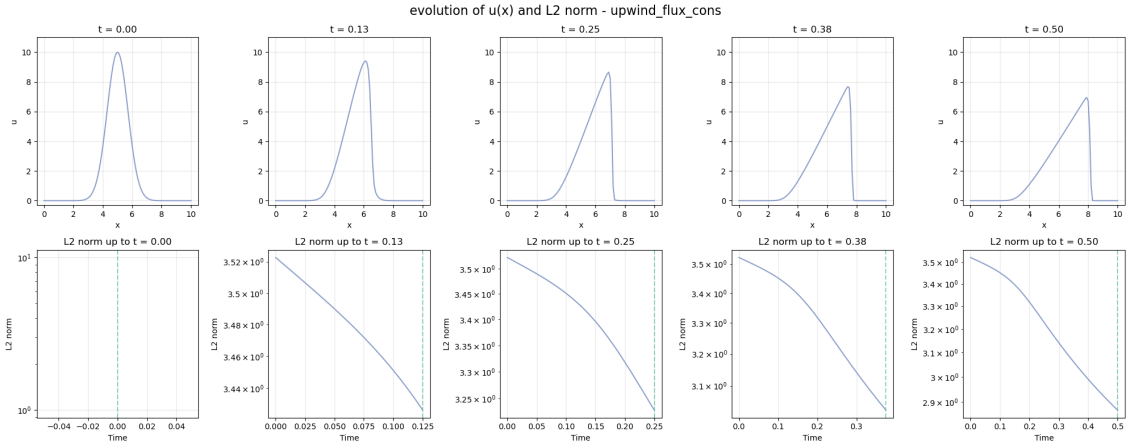
(b) Upwind, non flux conservative evolution at 5 different times.

Figure 1.15

1.2.2 Upwind (flux conservative)



(a) Comparison between initial u(x) profile vs final u(x) profile, at time t = 0,4. On the right L2norm evolution in time.



(b) Upwind, flux conservative evolution at 5 different times.

Figure 1.16

1.2.3 Comparison between methods

We can notice how the two methods, even if they produce a similar profile, have the peak in two different positions: in particular the peak position for the non-conservative scheme is incorrect.

In fact the Lax–Wendroff theorem states that a conservative numerical scheme, if convergent, will approximate the correct solution of a conservation law. This is not guaranteed for non-conservative schemes, which may converge to the wrong solution, as we also expect from the Hou–LeFloch theorem, which states: "Non conservative schemes do not converge to the exact solution if a shock wave is present".

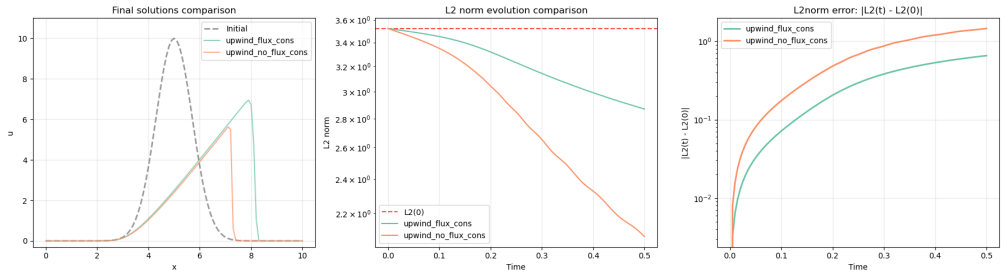


Figure 1.17: Comparison of upwind schemes for the Burgers equation, with Gaussian initial conditions (dashed line).

1.2.4 Changing Courant factor and resolution

In the following plots, I explore how changing the Courant factor c_F and the snumber of point in the spatial grid J affects the numerical solution of the Burgers equation:

- $c_F = [0.1, 0.5, 0.9, 1.0, 1.1]$
- $J = [51, 101, 201, 401]$

From a qualitative point of view, there is no dramatic difference in the shape of the solution when varying these parameters: the upwind conservative scheme remains stable and accurate, higher resolution refines the gradients.

Some smaller effects can be noticed looking at the L^2 norm.

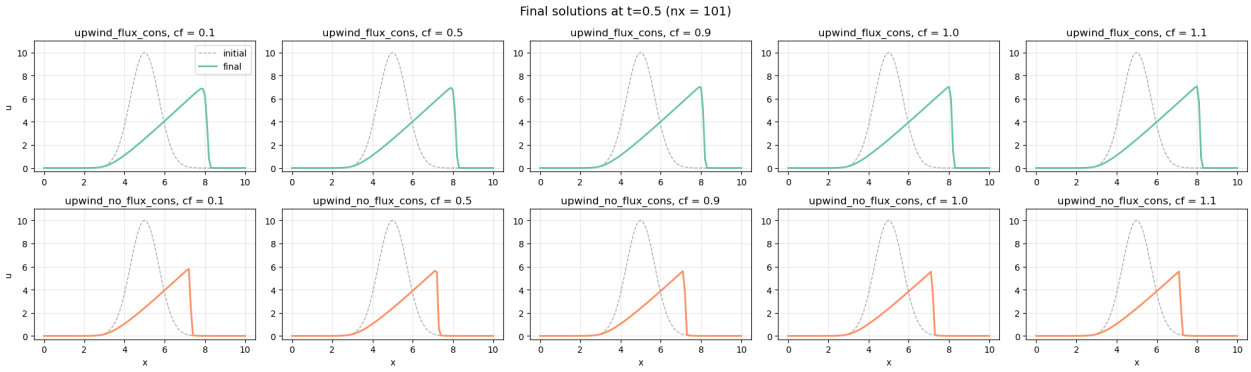


Figure 1.18: Effect of varying the Courant factor c_F on the conservative upwind scheme.

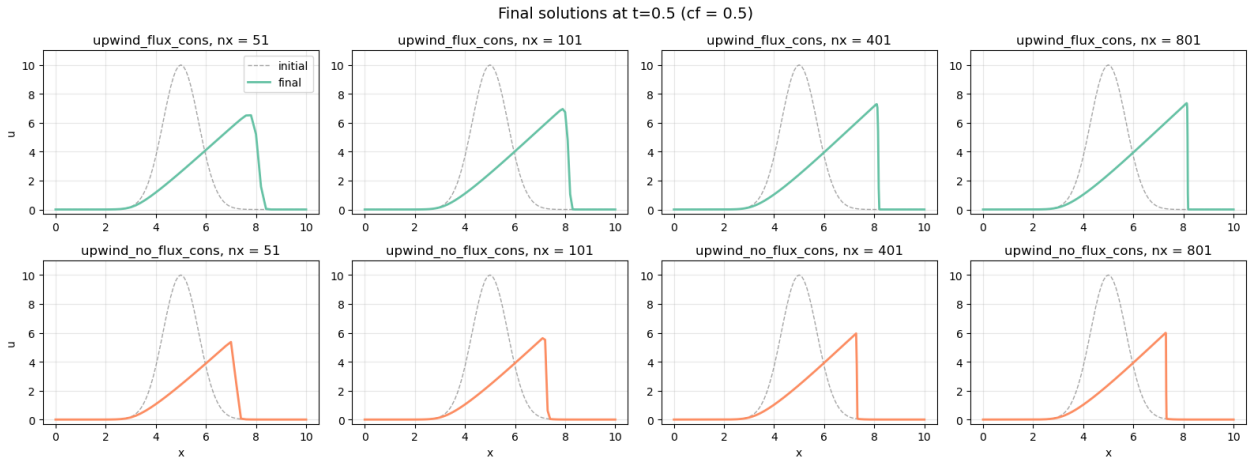
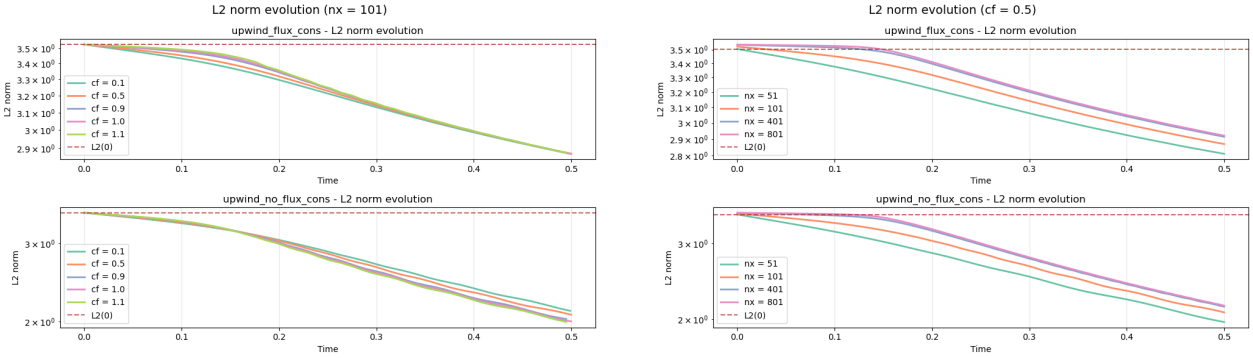


Figure 1.19: Effect of varying the number of spatial grid points J. Increasing resolution refines the gradients near the shock.



(a) L^2 norm evolution for different c_F.

(b) L^2 norm evolution for different resolutions J.

Minor variations appear in the L^2 norm over time when changing the Courant factor, while increasing resolution (right plot) leads to slightly improved conservation.

Chapter 2

Homework 2

In this second part of the report I used the Einstein Toolkit, an open-source numerical relativity framework, to perform simulations involving relativistic hydrodynamics. The simulations were done using the GRHydro module [2], which solves the General-Relativistic Magnetohydrodynamic Equations(GRMHD) [1].

2.1 Sod Shock Tube Problem

The Sod shock tube is a one-dimensional Riemann problem with an initial discontinuity separating two fluids.

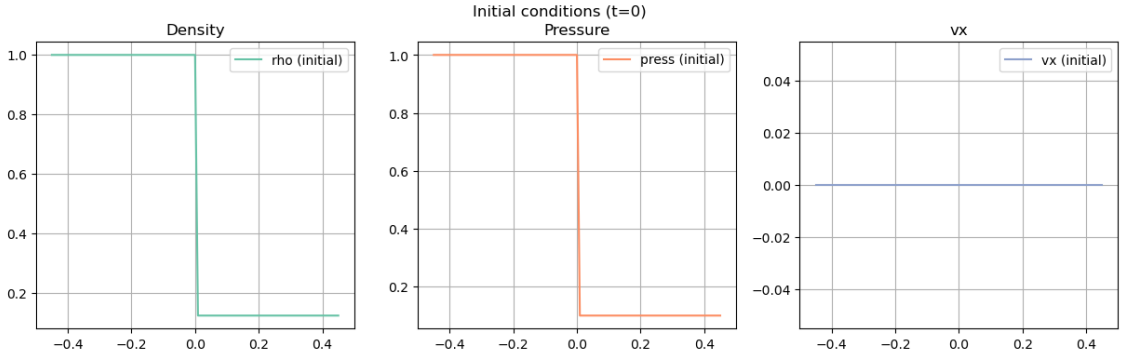


Figure 2.1: Initial conditions for the Sod shock tube: a discontinuity separates high-density, high-pressure fluid (left) from low-density, low-pressure fluid (right).

I solved the Sod problem using the Einstein Toolkit at five different spatial resolutions.

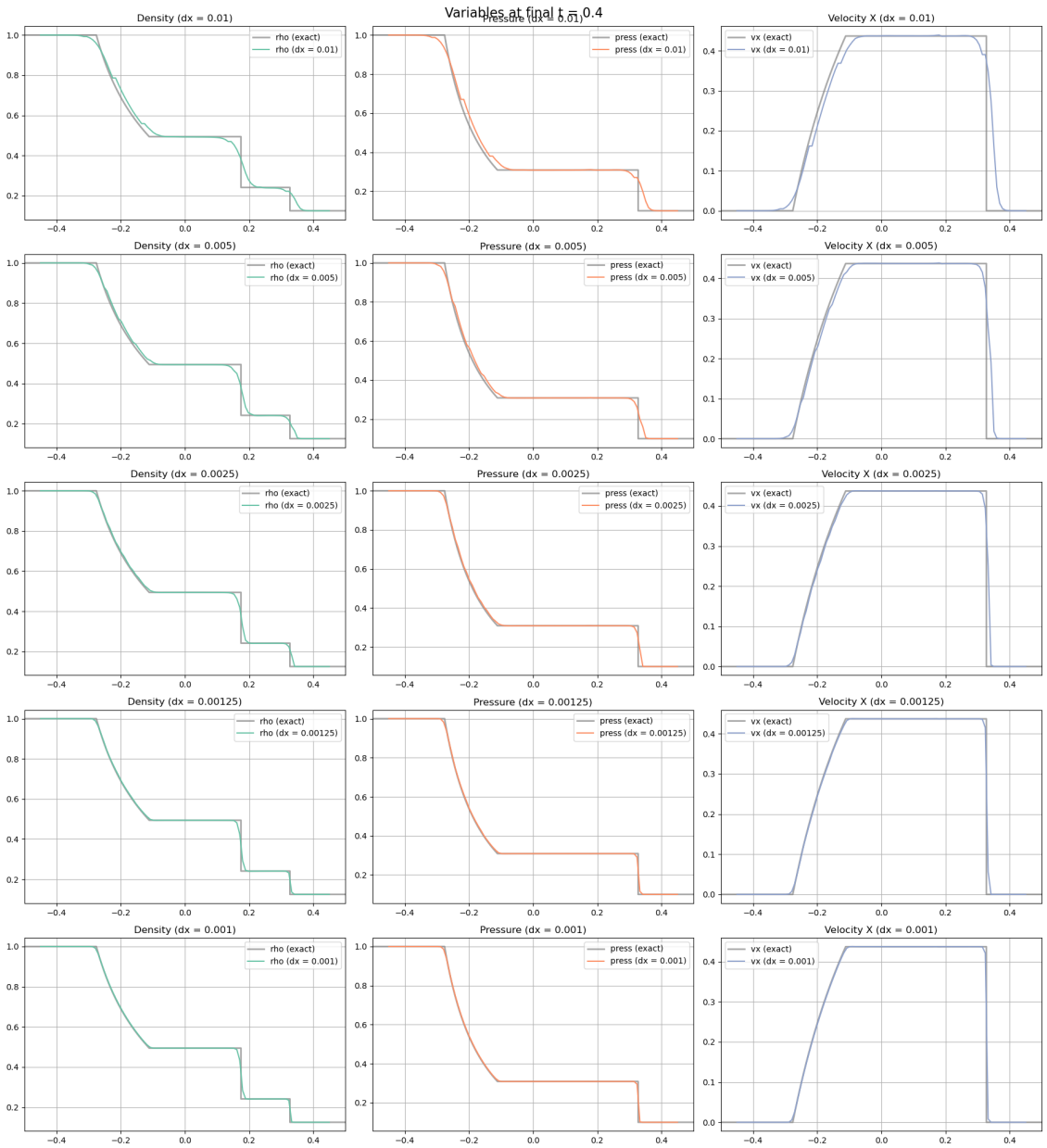


Figure 2.2: Conserved quantities profiles at different resolutions. Increased resolution improves the sharpness of discontinuities.

The equations used to model the system are the Euler equations, which express the conservation of mass, momentum, and energy for an ideal fluid. Evolving this system reveals three characteristic waves, each propagating with a different speed.

The three conserved quantities all show the presence of a rarefaction wave and a shock wave. In the rest mass density profile we can recognize a formation of a rarefaction wave on the left, a contact discontinuity in the center, and a shock wave on the right.

The rarefaction wave travels to the left and originates from the rapid expansion of the high-pressure fluid that initially was on that side.

On the opposite side, a shock wave propagates to the right. It forms when the high-pressure gas compresses the lower-pressure fluid ahead of it.

Between the two, a contact discontinuity can be seen. It shows the boundary between regions with different densities but equal pressure and velocity.

Changing the resolutions we notice how low resolution blurs shocks and contact discontinuities and high resolution closely matches the expected analytical profile.

2.2 TOV Evolution

In this section I show the results of the simulation of the evolution of the maximum rest mass density (ρ_c) in a stable Tolman–Oppenheimer–Volkoff (TOV) solution using the Einstein Toolkit.

I explored the evolution of ρ_c, indicator of the simulated neutron star stability, with different grid resolutions.

I compared results from two setups: unperturbed initial data ($poly_k = 100$) and data with a small pressure perturbation ($ploy_k = 95$).

In the table below there are some of the parameters I used for the TOV simulation:

Parameter	**Value**
Final time	`Cactus::cctk_final_time = 400`
Grid resolution	`CoordBase::dx = dy = dz = 2`
Grid domain	`CoordBase::xmax = ymax = zmax = 24`
Grid refinement	`CarpetRegrid2::num_levels_1 = 2`
	1 refinement level `RL1` with half resolution at 12
Reflection symmetry	`ReflectionSymmetry::reflection_x/y/z = yes`
	(only 1 octant evolved)
MoL (Method of Lines)	`MoL::ODE_Method = "rk4"`
Hydro evolution	`HydroBase::evolution_method = "GRHydro"`
Equation of State	Gamma-law EOS with `Gamma = 1.3`

Initial rest-mass density profile

The initial configuration is a static, spherically symmetric TOV solution in equilibrium. The plot below shows the distribution of the rest-mass density at $t = 0$.

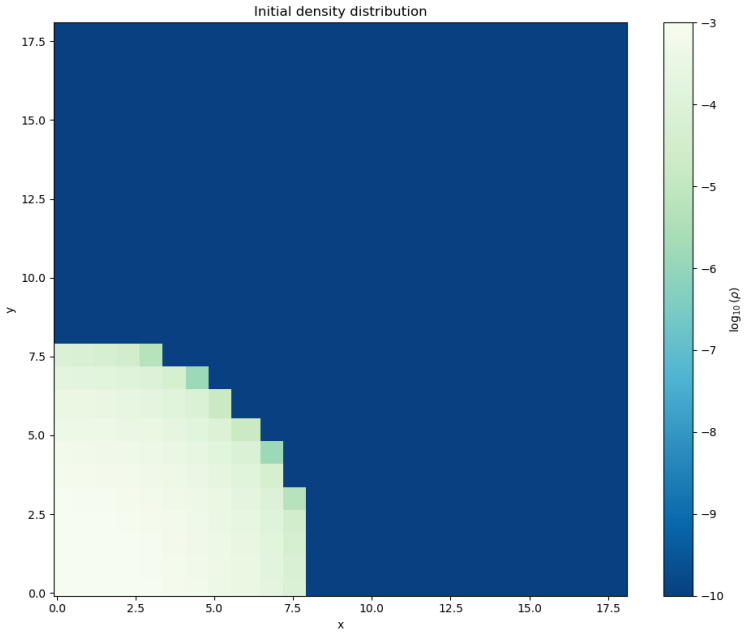
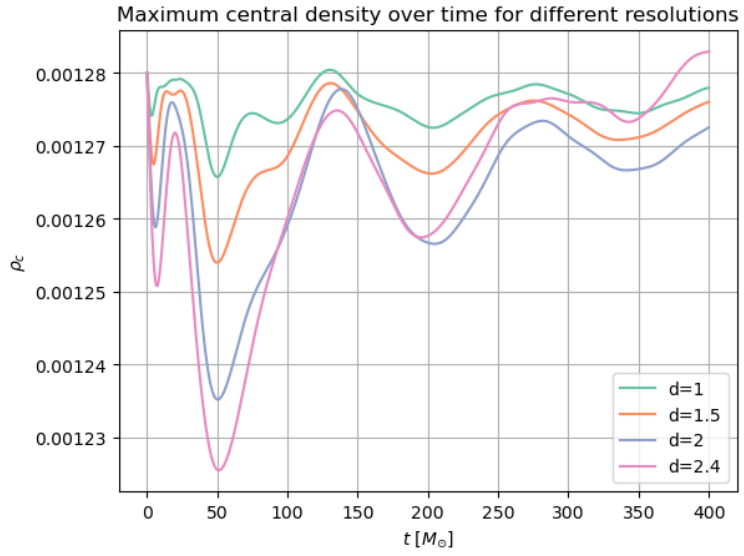


Figure 2.3: Initial rest-mass density profile of the TOV neutron star.

2.2.1 Unperturbed Evolution

In the absence of perturbations, the star remains close to equilibrium and the central density oscillates around a constant value. However, due to numerical errors, small oscillations of ρ_c appear. They decrease with increasing resolution. In the plots below the resolution decreases from left to right.



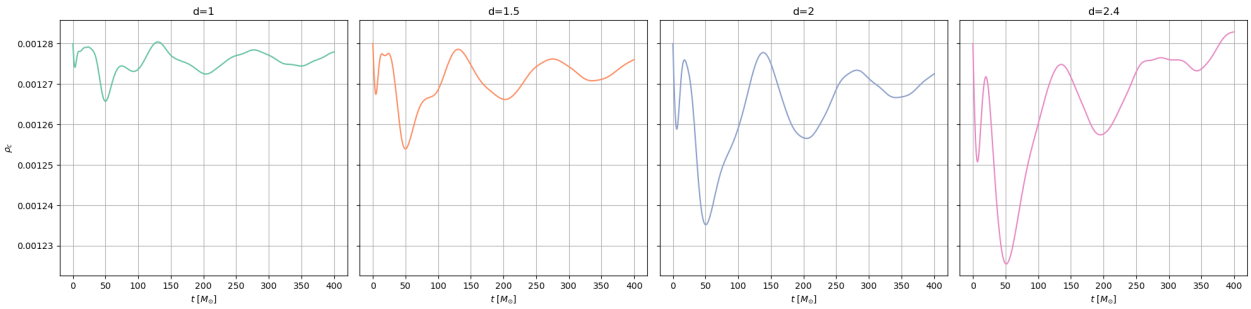


Figure 2.4: Time evolution of central density ρ_c for different resolutions. The amplitude of oscillations decreases as resolution increases.

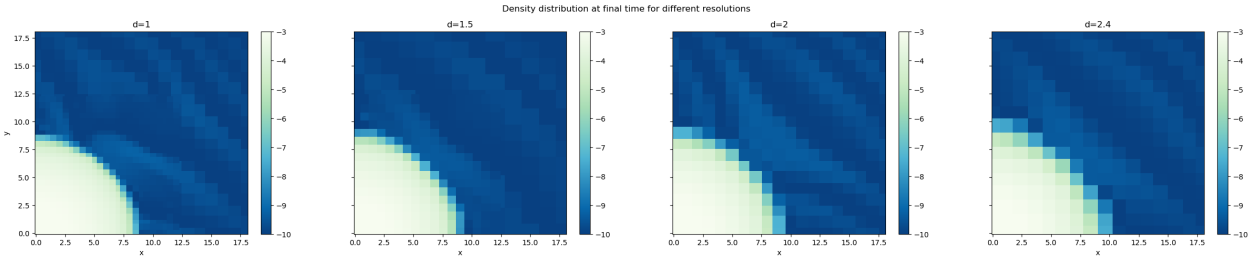


Figure 2.5: Final density ρ_c shown in 2D for different resolutions. The amplitude of oscillations decreases as resolution increases.

2.2.2 Perturbed Evolution

I then introduced a small pressure perturbation in the equilibrium configuration. This is done by reducing the polytropic constant, setting `poly_k = 95` instead of its original value `poly_k = 100`. This way I lowered the pressure throughout the star, perturbing the initial TOV solution.

The star is no longer in hydrostatic equilibrium and starts oscillating.

In all cases no gravitational collapse occurs, indicating that the configuration is still within the stability regime of TOV solutions.

The amplitude and frequency of the density oscillations show differences between resolutions, with higher resolutions providing more accurate representation of the, now physical, oscillations.

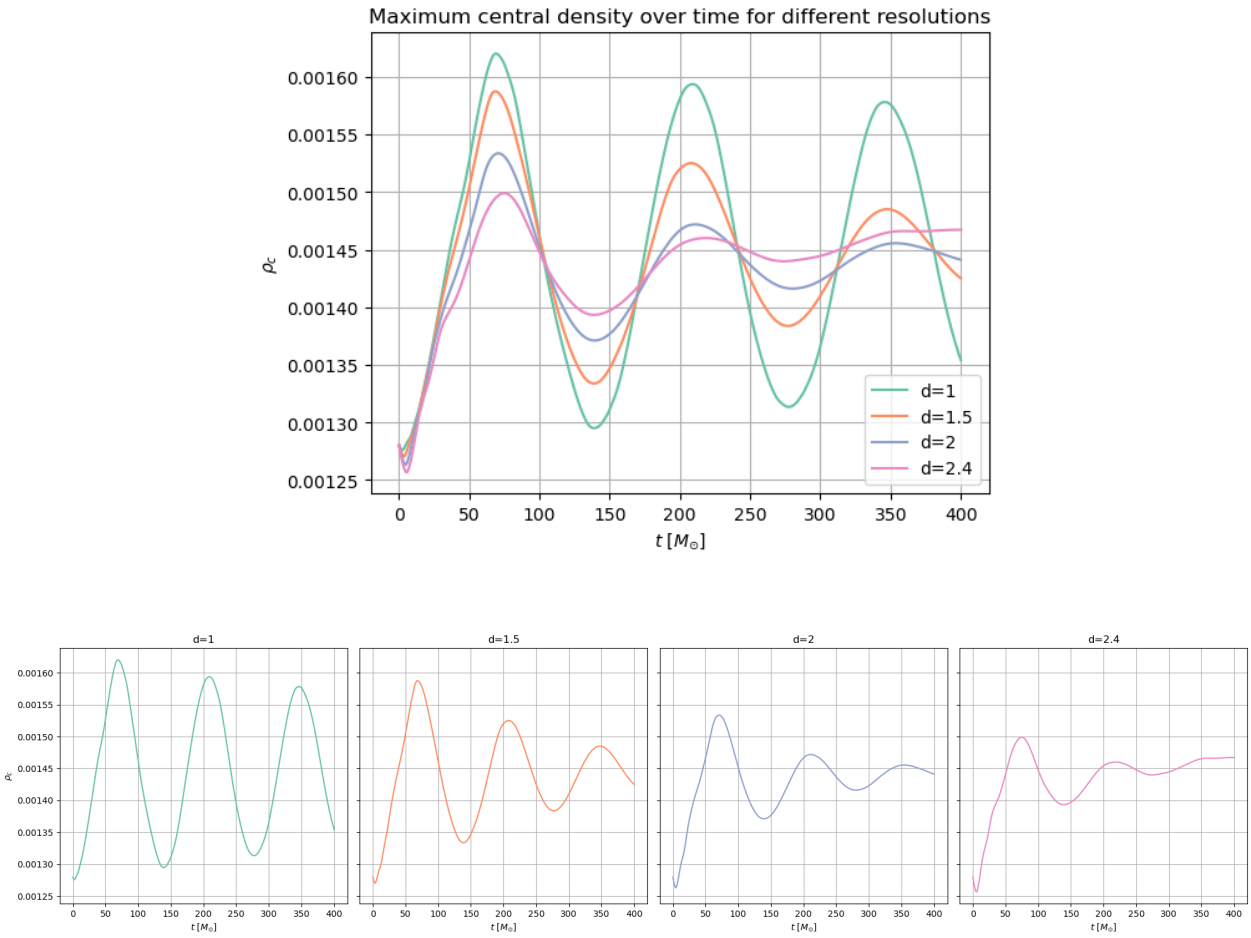


Figure 2.6: Time evolution of ρ_c with an initial pressure perturbation, for different resolutions.

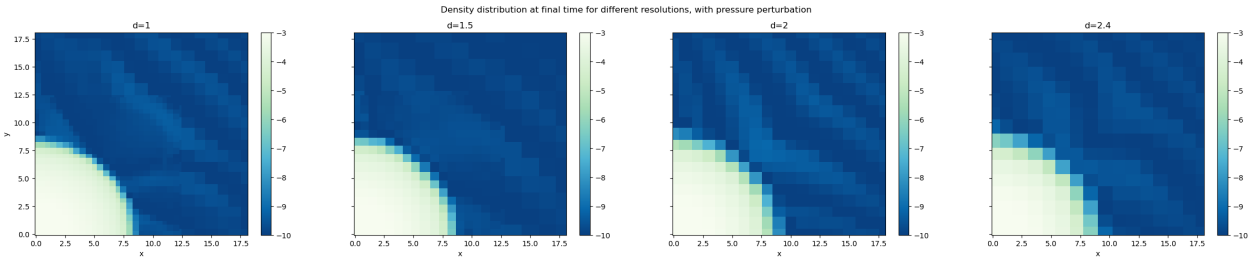


Figure 2.7: 2D final rest mass density with an initial pressure perturbation, for different resolutions.

In all cases, the evolution of ρ_c is consistent with expectations for a stable TOV solution. The results confirm that higher resolution improves accuracy and reduce numerical artifacts, both in perturbed and unperturbed setups.

2.2.3 Effect of grid setup

I also tested the sensitivity of ρ_c evolution to changes in the domain size and number of refinement levels.

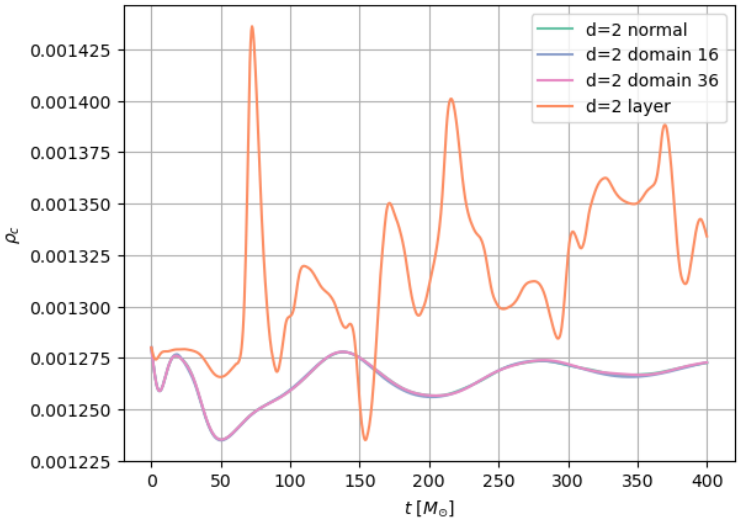


Figure 2.8: Central density evolution for different grid configurations. The plot shows ρ_c vs time for various domain sizes (d=2 domain 24, d=2 domain 16, d=2 domain 36) and with an added refinement level at 8 (d=2 layer).

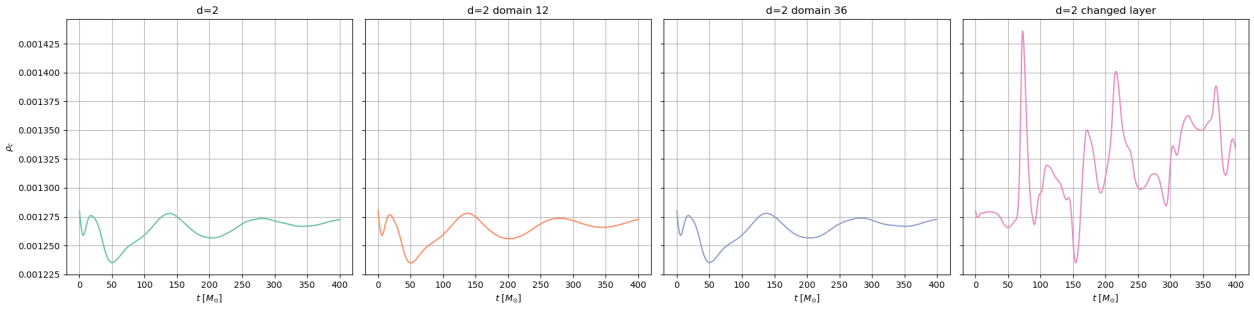


Figure 2.9: Central density evolution for different grid configurations (domain size/refinement levels).

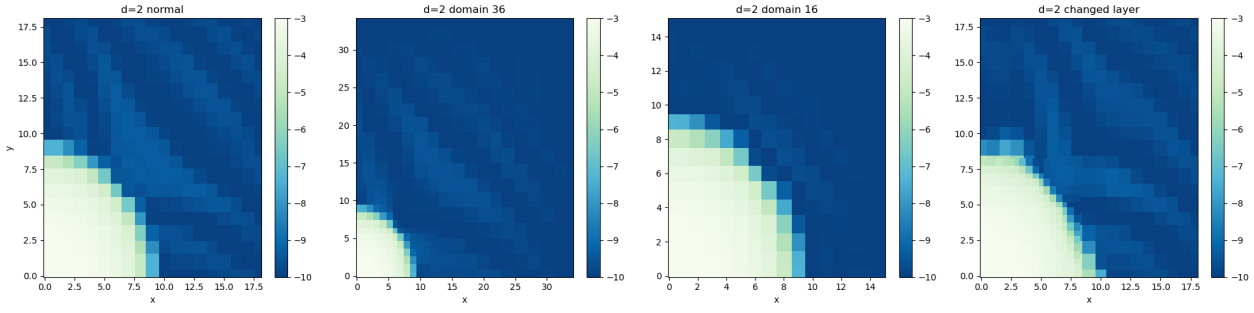


Figure 2.10: 2D final density for different grid configurations (domain size/refinement levels).

Changing the domain size from the standard configuration to larger and smaller domains (domain 16 and domain 36) shows minimal impact on the central density evolution, since the neutron star radius is around 8.

The addition of a refinement level instead introduces significant numerical instabilities, indicating that the refinement implementation may be introducing numerical artifacts.

2.3 Einstein Toolkit Parameters

```
MoL::ODE_Method = "rk4"
```

This parameter specifies the time integration method used by the Method of Lines (MoL) thorn. "rk4" refers to the fourth-order Runge-Kutta method.

The Method of Lines is an approach for solving partial differential equations (PDEs) where space and time are handled separately. In fact, if one is able to discretize the spatial derivatives can obtain a set of ordinary differential equations (ODEs) in time and evolve them using different time integration methods.

In particular, the fourth-order Runge-Kutta method updates the solution using a weighted combination of four slope evaluations.

```
GRHydro::recon_method = "ppm"
```

This parameter controls the reconstruction method in GRHydro, the general relativistic magnetohydrodynamics module. The reconstruction step is necessary because Godunov methods compute cell averages but require interface values between cells to compute fluxes for the conservation law $\partial_t \mathbf{u} + \partial_x \mathbf{f}(\mathbf{u}) = \mathbf{0}$.

We have seen slope limiters and TVD methods that reconstruct interface values using linear profiles. The Piecewise Parabolic Method (PPM) extends this concept and instead of using linear reconstruction with slopes, it constructs parabolic profiles in each cell and applies limiters to prevent oscillations near discontinuities. It also achieves higher accuracy in smooth regions.

```
GRHydro::riemann_solver = "Marquina"
```

This parameter selects the Riemann solver used in GRHydro. In the Godunov method, after reconstruction provides left and right states at each cell interface, we have to solve a Riemann problem: two different constant states separated by a discontinuity, that we need to evolve.

The Riemann solver determines the resulting flux $\mathbf{f}_{i+1/2}$ at the interface between cells, where we have $\mathbf{u}_L$ and $\mathbf{u}_R$: the left and right reconstructed states.

We have seen the HLLE solver, which approximates the solution using only the maximum eigenvalues (fastest wave speeds) of the Jacobian matrix $\partial \mathbf{f}/\partial \mathbf{u}$. The eigenvalues represent the characteristic speeds $\lambda = \frac{dx}{dt}$ at which information propagates.

The Marquina solver is more sophisticated: it utilizes the complete eigenvector and eigenvalue structure of the Jacobian matrix and captures the complete wave structure, including intermediate waves that HLLE misses, providing better resolution of contact discontinuities and rarefaction waves.

BSSN gauge parameters

```
ML_BSSN::harmonicN = 1 # 1+log
ML_BSSN::harmonicF = 2.0 # 1+log
ML_BSSN::ShiftGammaCoeff = 0.75 #this is 3/4
ML_BSSN::BetaDriver = 2.66 #common choices are 1/M or 1/2M
```

These parameters control the gauge conditions in the BSSN (Baumgarte-Shapiro-Shibata-Nakamura) formulation. BSSN is a reformulation of the ADM (Arnowitt-Deser-Misner) approach that achieves strong hyperbolicity, essential for stable numerical evolution.

Both ADM and BSSN use the 3+1 decomposition: they split spacetime into space and time evolution, separating the 4D Einstein equations into constraint equations and evolution equations

(that have to be integrated in time). While ADM is a weakly hyperbolic system [3] and can give rise to numerical instabilities, BSSN is strongly hyperbolic and well posed.

- `ML_BSSN::harmonicN = 1` and `ML_BSSN::harmonicF = 2.0` correspond to the $1 + \log$ slicing condition. This is a popular choice for singularity-avoiding gauge.

  A slicing condition determines how we choose spatial hypersurfaces. This choice is fundamental because numerical codes can crash when evolving near singularities and there the proper volume tends to increase dramatically, leading to the loss of precision.

  In fact, the slicing condition must satisfy three requirements:

  - If singularities are present, these should be avoided (singularity-avoiding slicing conditions).
  - If coordinate distortions take place, these should be counteracted
  - The gauge conditions should not be computationally expensive

  The Hyperbolic K-Driver Slicing is usually implemented in numerical simulations:

  $$(\partial_t - \beta^i \partial_i)\alpha = -f(\alpha)\,\alpha^2 (K - K_0)$$

  where $f(\alpha)$ determines the specific slicing: $f(\alpha) = 1$ gives harmonic slicing, $f(\alpha) = \frac{2}{\alpha}$ gives 1+log slicing (our choice).

- `ML_BSSN::ShiftGammaCoeff = 0.75` and `ML_BSSN::BetaDriver = 2.66` are part of the Gamma-driver shift condition, which controls the evolution of the shift vector. While the slicing condition sets how we slice spacetime in time, the shift determines how spatial coordinates move across these slices.

  The Gamma driver shift condition follows:

  $$\begin{cases} \partial_t \beta^i - \beta^j \partial_j \beta^i = \frac{3}{4} B^i \\ \partial_t B^i - \beta^j \partial_j B^i = \partial_t \tilde{\Gamma}^i - \beta^j \partial_j \tilde{\Gamma}^i - \eta B^i \end{cases} \tag{2.1}$$

  The coefficient before B^i in the first equation here is set to $0.75 = \frac{3}{4}$ and $\eta = 1/2M$.

Code and notebooks

All simulations, parameter files, and analysis scripts are available in the GitHub repository:

- **Main repository:** github.com/malvibellotti/numerical_rel
- **Homework 1 notebook:** Homework_1.ipynb
- **Homework 2 notebook:** Homework_2.ipynb

Bibliography

[1] Yosuke Mizuno and Luciano Rezzolla. *General-Relativistic Magnetohydrodynamic Equations: the bare essential*. 2024. arXiv: 2404.13824 [astro-ph.HE]. URL: https://arxiv.org/abs/2404.13824.

[2] Philipp Mösta et al. "GRHydro: a new open-source general-relativistic magnetohydrodynamics code for the Einstein toolkit". In: *Classical and Quantum Gravity* 31.1 (Nov. 2013), p. 015005. ISSN: 1361-6382. DOI: 10.1088/0264-9381/31/1/015005. URL: http://dx.doi.org/10.1088/0264-9381/31/1/015005.

[3] Oscar A. Reula. "Hyperbolic Methods for Einstein's Equations". In: *Living Reviews in Relativity* 1.1 (1998), p. 3. ISSN: 1433-8351. DOI: 10.12942/lrr-1998-3. URL: https://doi.org/10.12942/lrr-1998-3.