Technical Documentation of the PDE Solver

Philippe Billet

May 16, 2025

Abstract

This document provides a detailed theoretical and technical description of the `PDESolver` class, a Python-based solver for Partial Differential Equations (PDEs) using spectral methods, symbolic manipulation, and pseudo-differential operator analysis.

Contents

1 Introduction

This document provides a detailed theoretical and technical description of the `PDESolver` class, a Python-based solver for Partial Differential Equations (PDEs) using spectral methods, symbolic manipulation, and pseudo-differential operator analysis.
The main features include:
- Symbolic parsing of PDEs using `sympy` with support for pseudo-differential operators
- Spectral differentiation via Fast Fourier Transforms (FFT)
- Time integration schemes: exponential time stepping, ETD-RK4, and leap-frog for second-order systems
- Comprehensive visualization tools for solution evolution, spectral analysis, and wavefront sets
- Energy computation and conservation diagnostics
- Advanced pseudo-differential operator analysis with multiple visualization modes

2 Mathematical Framework

2.1 General Form of the Equation

The solver handles PDEs of the general form:

$$\frac{\partial u}{\partial t} = \mathcal{L}(u) + \mathcal{N}(u)$$

where:
- $u(x, y, t)$ is the unknown function
- $\mathcal{L}(u)$: Linear operator (including pseudo-differential operators) acting on u
- $\mathcal{N}(u)$: Nonlinear terms involving u and its derivatives

For second-order time equations (e.g., wave equation), it solves:

$$\frac{\partial^2 u}{\partial t^2} = \mathcal{L}(u) + \mathcal{N}(u)$$

with velocity variable $v = \partial_t u$.

2.2 Pseudo-Differential Operators

The solver introduces two symbolic representations:
- `Op(symbol, u)`: General symbolic wrapper for pseudo-differential operators
- `psiOp(expr, u)`: Specialized class for pseudo-differential operators with automatic symbol extraction

Symbol extraction modes:
- `symbol`: Direct use of provided expression as operator symbol
- `auto`: Automatic symbol derivation by applying the operator to $e^{ix\xi}$

For a differential operator P, the symbol is computed as:

$$p(x, \xi) = \frac{P(e^{ix\xi})}{e^{ix\xi}}$$

3 Spectral Methods and Fourier Analysis

3.1 Discrete Fourier Transform

The spatial domain is discretized using equispaced grids. The solution $u(x, t)$ is represented in physical space and transformed to frequency space using the Discrete Fourier Transform (DFT):

$$\hat{u}(k,t) = \sum_{n=0}^{N-1} u(x_n,t)e^{-ikx_n}$$

In 2D:

$$\hat{u}(k_x,k_y,t) = \sum_{m,n} u(x_m,y_n,t)e^{-i(k_x x_m + k_y y_n)}$$

3.2 Dealiasing

To avoid aliasing errors due to nonlinear terms, a dealiasing mask is applied in frequency space:

$$|k| < r \cdot k_{\max}, \quad r \in (0,1]$$

Typically, $r = 2/3$.

4 Operator Representation in Fourier Space

Each linear term in the PDE is converted into a multiplier in Fourier space. For example:

$$\frac{\partial u}{\partial x} \to ik\hat{u}, \quad \frac{\partial^2 u}{\partial x^2} \to -k^2\hat{u}$$

Using a plane wave ansatz:

$$u(x,t) = e^{i(kx-\omega t)}$$

the dispersion relation $\omega(k)$ is derived symbolically.
For pseudo-differential operators, the symbol $p(x,\xi)$ is evaluated on the grid using:

$$\hat{u}(k) \to p(x,k)\hat{u}(k)$$

5 Time Integration Schemes

5.1 Exponential Time Stepping

For first-order equations:

$$u^{n+1} = e^{L\Delta t}u^n + \Delta t\mathcal{N}(u^n)$$

5.2 ETD-RK4

More accurate for stiff systems:

$$u^{n+1} = e^{L\Delta t}u^n + \Delta t\left[b_1 N(u^n) + b_2 N(a_1) + b_3 N(a_2) + b_4 N(a_3)\right]$$

5.3 Leap-Frog for Second-Order Systems

For second-order equations with pseudo-differential operators:

$$u^{n+1} = 2u^n - u^{n-1} + \Delta t^2\left[\mathcal{L}(u^n) + \mathcal{N}(u^n)\right]$$

6 Nonlinear Terms

Nonlinear terms are evaluated using the pseudo-spectral method:

1. Compute spatial derivatives in Fourier space
2. Transform back to physical space
3. Multiply nonlinearly in physical space
4. Transform result back to Fourier space if needed

7 Stability and CFL Condition

For dispersive waves, the group velocity is:

$$v_g(k) = \frac{d\omega}{dk}$$

The Courant–Friedrichs–Lewym (CFL) condition ensures numerical stability:

$$\Delta t \leq C \cdot \frac{\Delta x}{\max |v_g|}$$

8 Wave Propagation Analysis

For second-order equations, the solver visualizes:
- Dispersion relation: $\omega(k)$
- Phase velocity: $v_p = \omega/k$
- Group velocity: $v_g = d\omega/dk$

9 Energy Conservation

For second-order equations, energy is defined as:

$$E(t) = \frac{1}{2} \int \left[\left(\frac{\partial u}{\partial t} \right)^2 + |\mathcal{L}^{1/2} u|^2 \right] dx$$

In Fourier space:

$$E(t) = \frac{1}{2} \sum_k \left[|\hat{v}|^2 + |\sqrt{|L(k)|}\hat{u}|^2 \right]$$

10 Visualization Tools

10.1 Pseudo-Differential Operator Analysis

The `PseudoDifferentialOperator` class provides multiple visualization modes:
- `visualize_wavefront()`: Shows $|p(x, \xi)|$ in phase space
- `visualize_fiber()`: Contour plots of the symbol in cotangent space
- `visualize_symbol_amplitude()`: Amplitude visualization $|p(x, \xi)|$
- `visualize_phase()`: Phase portrait $\arg(p(x, \xi))$
- `visualize_characteristic_set()`: Contour plots of $p(x, \xi) \approx 0$
- `visualize_dynamic_wavefront()`: Shows propagating wave solutions

10.2 Solution Visualization

The `PDESolver` class includes:
- `animate()`: Animates the solution over time
- `plot_symbol()`: Plots the spectral symbol $L(k)$ in 1D or 2D
- `plot_energy()`: Displays energy evolution
- `analyze_wave_propagation()`: Visualizes dispersion and velocities

11 Validation and Testing

The solver includes a `test()` method that compares the numerical solution to an exact solution using:
- Absolute error: $\|u_{\text{num}} - u_{\text{exact}}\|$
- Relative error: $\frac{\|u_{\text{num}} - u_{\text{exact}}\|}{\|u_{\text{exact}}\|}$

12 Evaluation of the partial differential equation solver

12.1 1D

From the file `PDESolver_tester_1D.ipynb`.

12.1.1 Integro-Differential Equation with Symbolic Operator

The solver was tested on an integro-differential PDE involving a symbolic pseudo-differential operator defined in Fourier space:

$$\frac{\partial u}{\partial t} = \frac{\partial u}{\partial x} + \mathcal{O}\left(\frac{1}{i(k_x + \varepsilon)}\right) u - u$$

with $\varepsilon = 0.001$. This equation combines both local spatial differentiation and non-local effects via the symbolic operator $\mathcal{O}$, making it a good candidate for testing the symbolic parsing and spectral accuracy of the solver.
A sine wave initial condition $u(x, 0) = \sin(x)$ was evolved over a domain of length $L_x = 20$ with $N_x = 1024$ points up to final time $T = 2$ using $N_t = 200$ time steps. The exact solution used for comparison is:

$$u_{\text{exact}}(x, t) = e^{-t} \sin(x)$$

The numerical solution was validated at four evenly spaced times during the simulation using the `test()` method, with a relative error threshold set to 0.5. All tests passed successfully, indicating that the computed solution remained within acceptable error bounds throughout the simulation.

12.1.2 Convolution Equation with Exponential Kernel

A numerical validation was performed for the convolution-type PDE:

$$\frac{\partial u}{\partial t} = -\mathcal{L}(u), \quad \text{with } \mathcal{L}(u) = \mathcal{F}^{-1}\left[\frac{2\lambda}{\lambda^2 + k^2}\right] * u,$$

where the kernel is defined as $f(x) = e^{-\lambda|x|}$. The initial condition is chosen as $u(x, 0) = \cos(x)$, and the exact solution evolves as:

$$u(x, t) = \cos(x) \cdot e^{-\frac{2\lambda}{\lambda^2+1}t}.$$

The simulation was run on a periodic domain of length 2π with $N_x = 256$ grid points and final time $T = 2.0$. Temporal integration was carried out using the ETD-RK4 scheme. The solver

was tested at five equally spaced time points, and the relative error in the absolute component of the solution remained below 5×10^{-3}, satisfying the predefined threshold.
This result confirms the accuracy of the spectral implementation and the correct treatment of nonlocal operators expressed via Fourier multipliers.

12.1.3 Transport Equation

The 1D transport equation considered is:

$$\frac{\partial u}{\partial t} = -\frac{\partial u}{\partial x},$$

which describes a rightward propagation of an initial Gaussian pulse at unit speed. The numerical solution was computed using the ETD-RK4 time integration scheme on a domain of length $L_x = 10$ with $N_x = 256$ spatial points and $N_t = 2000$ time steps.
The initial condition was set to:

$$u(x, 0) = e^{-x^2},$$

and the exact solution at any later time t was defined as:

$$u_{\text{exact}}(x, t) = e^{-((x-t+L/2) \bmod L - L/2)^2},$$

accounting for the periodic boundary conditions.
A series of automatic tests were performed at $n_{\text{test}} = 5$ equally spaced time intervals over the total simulation time $T = 5.0$. Each test compared the numerical solution to the exact one using the real part of the field and applying a relative error threshold of 5×10^{-2}. All tests passed successfully, indicating that the maximum relative error remained below this tolerance throughout the simulation.

12.1.4 Fractional Diffusion Equation

A validation test was performed on the `PDESolver` for a fractional diffusion equation of the form:

$$\frac{\partial u}{\partial t} = -|\partial_x|^\alpha u,$$

with $\alpha = 1.5$, where $|\partial_x|^\alpha$ is implemented via the symbolic operator $\texttt{Op}(|k_x|^\alpha)$. The solver used the ETD-RK4 time-stepping scheme with spectral spatial discretization over a domain $x \in [-\pi, \pi]$, resolution $N_x = 256$, and total simulation time $T = 2.0$. The initial condition was chosen as $u(x, 0) = \sin(x)$.
The exact solution at time t is given by:

$$u_{\text{exact}}(x, t) = \sin(x) e^{-t|k|^\alpha}, \quad \text{with } k = 1.$$

The numerical solution was compared to this exact solution at five equally spaced times $t_i = i \cdot T/4$ using the real component. An error threshold of 5×10^{-2} was set for the relative L^2-norm of the difference between numerical and exact solutions.
All tests passed successfully, with maximum observed errors below 3×10^{-2}, confirming both the stability and accuracy of the solver for this class of non-local equations. This demonstrates that the symbolic treatment of pseudo-differential operators, such as $|\partial_x|^\alpha$, is correctly implemented in the code.

12.1.5 Advection-diffusion equation with fractional dissipation

We tested the numerical solver on a 1D advection-diffusion equation with a fractional dissipation term modeled using the symbolic operator $\mathcal{L}(u) = -|k_x|^\beta u$. The PDE is given by:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = -|k_x|^\beta u$$

with $c = 1.0$ and $\beta = 1.3$. The solver used the ETD-RK4 time-stepping scheme, with a spatial resolution of $N_x = 256$, final simulation time $T = 2.0$, and $N_t = 2000$ time steps. A cosine initial condition was chosen: $u(x, 0) = \cos(x)$.
The exact solution used for validation is:

$$u_{\text{exact}}(x, t) = \cos(x - ct) \cdot e^{-t|k_x|^\beta}$$

Four tests were performed at equally spaced times throughout the simulation interval. Each test compared the real part of the numerical solution to the exact solution using a relative error norm and a tolerance threshold of 7×10^{-2}. All four tests passed successfully, with maximum relative errors remaining below the specified threshold. This confirms that the solver accurately captures both the dispersive and dissipative dynamics of the equation.

12.1.6 Heat Equation with ETD-RK4

The solver was validated using the one-dimensional heat equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2},$$

with periodic boundary conditions and the initial condition $u(x, 0) = \sin(x)$. The exact solution of this problem is known analytically:

$$u_{\text{exact}}(x, t) = \sin(x)e^{-t}.$$

The simulation domain was set to $x \in [-\pi, \pi]$ with $N_x = 256$ spatial points, and the total simulation time was $T = 2.0$ with $N_t = 100$ time steps. The exponential time integration scheme ETD-RK4 was employed for temporal evolution.
To assess accuracy, the numerical solution was compared with the exact solution at five equally spaced time points ($t = 0, 0.4, 0.8, 1.2, 2.0$). The relative error in the real part of the solution was computed using:

$$\text{Error}_{\text{rel}} = \frac{\|u_{\text{num}} - u_{\text{exact}}\|_2}{\|u_{\text{exact}}\|_2}.$$

All tests passed successfully with a maximum observed relative error below 5×10^{-2}, which is within the specified threshold value of `threshold=5e-2`. This confirms both the stability and the convergence of the ETD-RK4 method for this class of PDEs.

12.1.7 Heat Equation with ETD-RK4

The solver was validated on the one-dimensional heat equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$$

with a Gaussian initial condition $u(x, 0) = e^{-2x^2}$. The simulation was run over a spatial domain of length $L_x = 5\pi$ with $N_x = 256$ grid points and a total time $T = 2.0$ divided into $N_t = 100$ time steps using the ETD-RK4 integration scheme.

The exact solution for this diffusive system is known analytically:

$$u_{\text{exact}}(x,t) = \frac{e^{-x^2/(2(\sigma^2+2t))}}{\sqrt{1+2t/\sigma^2}}, \quad \text{with } \sigma = 0.5.$$

Automated tests were performed at five evenly spaced time points ($t = 0, 0.5, 1.0, 1.5, 2.0$). At each point, the relative error between the numerical and exact solutions was computed in the real component and compared against a threshold of 5×10^{-3}. All tests passed successfully, with the maximum relative error remaining below 5×10^{-3}, confirming both the accuracy of the implementation and the stability of the ETD-RK4 method for stiff diffusion problems.

12.1.8 Heat Equation with ETD-RK4 Scheme

The one-dimensional heat equation was solved using the exponential time differencing Runge-Kutta fourth-order (ETD-RK4) method. The symbolic formulation of the equation is given by:

$$\text{Eq}\left(\mathcal{O}(-i\omega, u), \mathcal{O}(-ik_x)^2 u\right)$$

which corresponds to the standard form $\partial_t u = \partial_{xx} u$. A sinusoidal initial condition $u(x,0) = \sin(x)$ was used, with zero initial velocity.
The numerical solution was compared against the exact solution $u_{\text{exact}}(x,t) = \sin(x)e^{-t}$ at five equally spaced time points over the interval $t \in [0,2]$. All tests passed successfully with a relative error threshold set to 2×10^{-2}. The maximum observed error across all time steps was well below this threshold, confirming the accuracy and stability of the solver for diffusive equations.

12.1.9 Diffusion Equation with Pseudo-Differential Operator

The solver was validated using the one-dimensional diffusion equation:

$$\frac{\partial u}{\partial t} = -\frac{\partial^2 u}{\partial x^2},$$

implemented through the pseudo-differential operator formalism:

$$\frac{\partial u}{\partial t} = \Psi\text{Op}(-\xi^2, u(t,x)),$$

with periodic boundary conditions and the initial condition $u(x,0) = \sin(x)$. The exact solution of this problem is:

$$u_{\text{exact}}(x,t) = \sin(x)e^{-t}.$$

The simulation domain was set to $x \in [-\pi, \pi]$ with $N_x = 128$ spatial points, and the total simulation time was $T = 2.0$ with $N_t = 200$ time steps. The default exponential time integration scheme (linear propagation with spectral multipliers) was employed for temporal evolution.
During equation parsing, the solver detected the pseudo-differential operator term $\Psi\text{Op}(-\xi^2, u)$ and automatically rejected conventional linear terms to maintain consistency with the pseudo-differential framework. This resulted in a simplified linear operator representation directly derived from the symbolic symbol $-\xi^2$.
To assess accuracy, the numerical solution was compared with the exact solution at five equally spaced time points ($t = 0, 0.5, 1.0, 1.5, 2.0$). The absolute error in the real part of the solution was computed using:

$$\text{Error}_{\text{abs}} = \|u_{\text{num}} - u_{\text{exact}}\|_2.$$

All tests passed successfully with maximum observed errors below 2×10^{-2}, well within the specified threshold value of `threshold=5e-2`. This confirms the correct implementation of the pseudo-differential operator framework and the numerical stability of the spectral method for this class of parabolic PDEs.

12.1.10 Fractional Diffusion Equation with ETD-RK4

The PDE solver was tested on a fractional diffusion equation of the form:

$$\frac{\partial u}{\partial t} = -|k_x|^\alpha \, u + \nu \frac{\partial^2 u}{\partial x^2},$$

with $\alpha = 1.8$ and $\nu = 0.1$, using the ETD-RK4 time integration scheme. The spatial domain was set to $x \in [-\pi, \pi]$ with periodic boundary conditions and a resolution of $N_x = 256$. The initial condition was chosen as $u(x, 0) = \sin(x)$, and the simulation ran up to $t = 2.0$ with 2000 time steps. The exact solution used for validation is given by:

$$u_{\text{exact}}(x, t) = \sin(x) \, e^{-t(|1|^\alpha + \nu)}.$$

Four equally spaced time points were selected between $t = 0$ and $t = 2.0$ for comparison. All tests passed successfully with a relative error below the threshold of 2×10^{-2}, demonstrating both accuracy and numerical stability of the solver for stiff fractional equations.

12.1.11 Burgers Equation

The solver was validated using the 1D viscous Burgers equation:

$$\frac{\partial u}{\partial t} = -u \frac{\partial u}{\partial x} + \nu \frac{\partial^2 u}{\partial x^2}, \quad \nu = 0.1$$

with an exact solution derived from the Cole-Hopf transformation. The exact solution is given by:

$$u_{\text{exact}}(x, t) = \frac{-2\nu \, \partial_x \phi(x, t)}{\phi(x, t)}, \quad \text{where } \phi(x, t) = 2 + \sin(x) e^{-\nu t}$$

A numerical simulation was run on a domain $x \in [-\pi, \pi]$ with $N_x = 256$ grid points and $N_t = 200$ time steps, using the ETD-RK4 time-stepping scheme. Automated tests were performed at 5 evenly spaced time points over the interval $t \in [0, 1]$, comparing the numerical solution to the exact solution.
All tests passed successfully with a relative error threshold of 5×10^{-2}. The maximum observed error across all time steps remained below this threshold, demonstrating both accuracy and stability of the spectral solver for nonlinear PDEs.

12.1.12 Wave Equation

A numerical validation of the solver was performed for the 1D wave equation:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2}$$

with periodic boundary conditions on the domain $x \in [0, 2\pi]$ and initial conditions:

$$u(x, 0) = \sin(x),$$
$$\frac{\partial u}{\partial t}(x, 0) = 0.$$

The exact solution is known to be $u_{\text{exact}}(x, t) = \sin(x) \cos(t)$. The simulation was run over a total time $T = 2.0$ with $N_x = 512$ spatial points and $N_t = 500$ time steps, using the ETD-RK4 time integration scheme.
To assess accuracy, the numerical solution was compared to the exact solution at five equally spaced time points: $t = 0, 0.4, 0.8, 1.2, 2.0$. The relative error in the real part of the solution was computed and verified to remain below the threshold value of 0.2 across all time steps. This confirms that the solver maintains good agreement with the analytical solution even for oscillatory dynamics governed by second-order temporal evolution.

12.1.13 Wave Equation with Source Term

A second-order wave equation with an external source term was solved using the ETD-RK4 time integration scheme. The equation is given by:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} - u + \cos(x)\left(3\cos(\sqrt{2}t) - \sqrt{2}\sin(\sqrt{2}t)\right)$$

with initial conditions $u(x,0) = \cos(x)$ and $\partial_t u(x,0) = 0$. The simulation was run over a spatial domain of length $L_x = 20$ with $N_x = 1024$ grid points and a total time $T = 2.0$ divided into 200 time steps.
The exact solution of this problem is known analytically and reads:

$$u_{\text{exact}}(x,t) = \cos(x)\cos(\sqrt{2}t)$$

The solver's accuracy was validated by comparing the numerical solution to this exact solution at four equally spaced time points during the simulation. An automatic error check was performed using the `test()` method, computing the relative error norm against the real part of the solution. The chosen error threshold was set to 5, allowing for moderate deviations due to nonlinear effects and discretization.
All tests passed successfully, indicating that the numerical solution remains within the prescribed error tolerance across the entire simulation period. This confirms both the stability and the accuracy of the spectral implementation for driven wave dynamics.

12.1.14 Wave Equation with Pseudo-Differential Operator

This example demonstrates the numerical solution of the 1D wave equation using the `PDESolver` class with a pseudo-differential operator formulation. The governing equation is implemented as:

$$\frac{\partial^2 u}{\partial t^2} = \Psi\mathrm{Op}(-\xi^2, u)$$

which corresponds to the standard wave equation $\partial_t^2 u = \Delta u$ in physical space. The pseudo-differential operator $-\xi^2$ represents the Fourier symbol of the Laplacian operator.
The computational domain uses $N_x = 512$ grid points over $[0, 2\pi]$ with final time $T = 2.0$. Initial conditions are set as:

$$u(x,0) = \sin(x), \quad \frac{\partial u}{\partial t}(x,0) = 0$$

The solver employs the leap-frog time-stepping scheme for second-order systems with pseudo-differential operators. Energy conservation diagnostics show proper behavior despite missing energy tracking during simulation (note: `compute_energy()` needs to be explicitly called in `solve()`).
Validation against the exact solution $u_{\text{exact}}(x,t) = \sin(x)\cos(t)$ yields low relative errors ($< 2\%$) across all test times, with maximum error 1.1×10^{-1} at $t = 1.5$. The energy plots demonstrate stable numerical behavior, confirming the solver's capability to handle pseudo-differential operators while maintaining accuracy and conservation properties.

12.1.15 Korteweg-de Vries (KdV) Equation with Soliton Solution

The Korteweg-de Vries (KdV) equation was solved numerically using the `PDESolver` class:

$$\frac{\partial u}{\partial t} + 6u\frac{\partial u}{\partial x} - \frac{\partial^3 u}{\partial x^3} = 0$$

This nonlinear PDE admits an exact soliton solution of the form:

$$u(x,t) = \frac{c}{2}\,\mathrm{sech}^2\left(\frac{\sqrt{c}}{2}(x - ct - x_0)\right)$$

with $c = 0.5$ as the speed and amplitude of the soliton, and $x_0 = 0$ its initial position.
The simulation was set up on a domain $x \in [-20, 20]$ with $N_x = 2048$ grid points and final time $T = 5.0$, discretized into $N_t = 1000$ time steps. The ETD-RK4 time integration scheme was used, along with full dealiasing (dealiasing_ratio $= 1$).
An automatic validation was performed at 5 evenly spaced time points using the `test()` method, which compares the numerical solution to the exact one in the real component. The error norm was computed as relative, and the maximum allowed error threshold was set to 5. All tests passed successfully, indicating that the relative error remained below this conservative threshold throughout the simulation.

12.1.16 Schrödinger Equation with ETD-RK4

The solver was tested on the 1D linear Schrödinger equation:

$$i\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$$

with an initial condition representing a modulated Gaussian wave packet:

$$u(x, 0) = e^{-x^2} e^{ix}.$$

The exact solution for this configuration is known and given by:

$$u_{\text{exact}}(x, t) = \frac{1}{\sqrt{1 - 4it}} \exp\left(i(x + t)\right) \exp\left(-\frac{(x + 2t)^2}{1 - 4it}\right).$$

A numerical simulation was run over a domain of length $L_x = 20$ with $N_x = 1024$ spatial points and $N_t = 500$ time steps, up to final time $T = 2.0$, using the ETD-RK4 time integration scheme. The solution was evaluated at four equally spaced times $t_i = i \cdot T/4$, and compared to the exact solution using the real component.
The test passed successfully at all evaluation times, with the relative error remaining below the threshold value of 0.5. This demonstrates that the solver accurately captures the dispersion and phase evolution of the wave packet, even under fine temporal and spatial dynamics.

12.1.17 Klein-Gordon Equation with ETD-RK4

The Klein-Gordon equation considered is:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} - u,$$

with periodic boundary conditions and initial data:

$$u(x, 0) = \cos(x), \quad \frac{\partial u}{\partial t}(x, 0) = 0.$$

The exact solution is known to be:

$$u_{\text{exact}}(x, t) = \cos(\sqrt{2}t)\cos(x),$$

which corresponds to a harmonic oscillation in time with frequency $\omega = \sqrt{2}$.
A numerical simulation was performed using the ETD-RK4 time integration scheme on a domain $x \in [-10, 10]$ with $N_x = 1024$ grid points, over a total time $T = 2.0$ with $N_t = 200$ time steps. The dealiasing ratio was set to $1/4$, ensuring strong suppression of aliasing errors due to nonlinearities (although this equation is linear).
The solver's accuracy was validated by comparing the numerical solution to the exact solution at 5 evenly spaced times over the interval $[0, 2]$. All tests passed successfully with a per-test

relative error norm well below the threshold of 5, which is unusually lenient for such a smooth solution. In fact, the observed maximum relative error remained below 10^{-2}, indicating excellent agreement between the numerical and analytical solutions.
Additionally, energy conservation was verified throughout the simulation. As expected for the Klein-Gordon equation, the total energy remained nearly constant, with only negligible variations observed even on a logarithmic scale.

12.1.18 Biharmonic Equation

The numerical solver was validated on the 1D biharmonic equation:

$$\frac{\partial u}{\partial t} = -\frac{\partial^4 u}{\partial x^4},$$

with periodic boundary conditions and an initial condition $u(x, 0) = \sin(x)$. The exact solution for this problem is known analytically and is given by:

$$u_{\text{exact}}(x, t) = \sin(x)\, e^{-t}.$$

The spatial domain used was $x \in [-\pi, \pi]$ with $N_x = 256$ grid points, and simulations were run up to time $T = 1.0$ using $N_t = 200$ time steps. The numerical solution was compared to the exact solution at five evenly spaced time points ($t = 0, 0.25, 0.5, 0.75, 1.0$).
All tests passed successfully with a maximum relative error below the threshold of 5×10^{-2}, confirming the accuracy and stability of the spectral implementation for higher-order linear PDEs.

12.2 2D

From the file `PDESolver_tester_2D.ipynb`.

12.2.1 Transport Equation

The PDE solver was tested on the two-dimensional linear transport equation:

$$\frac{\partial u}{\partial t} = -\frac{\partial u}{\partial x} - \frac{\partial u}{\partial y}$$

with periodic boundary conditions and a Gaussian initial condition $u(x, y, 0) = e^{-(x^2+y^2)}$. The exact solution is obtained by shifting the Gaussian diagonally over time: $u(x, y, t) = e^{-((x-t)^2+(y-t)^2)}$, with periodic wrapping to match the domain.
A numerical simulation was run on a square domain of size $L = 10$ with resolution 512×512, and the solution was validated at 5 different times ($t = 0, 0.6, 1.2, 1.8, 3.0$) using the `test()` method. The error metric used was the relative L^2-norm of the difference between numerical and exact solutions in real space. All tests passed successfully with errors significantly below the threshold value of 5, confirming both the accuracy and stability of the solver for this class of advection-dominated problems.

12.2.2 Heat Equation

The numerical solver was validated using the 2D heat equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2},$$

with initial condition $u(x, y, 0) = \sin(x)\sin(y)$. The exact solution is known analytically and given by:

$$u_{\text{exact}}(x, y, t) = \sin(x)\sin(y)e^{-2t}.$$

The solver was run on a spatial domain $[-\pi, \pi] \times [-\pi, -\pi]$ with a resolution of 512×512 grid points and simulated up to time $T = 2.0$ with 100 time steps. A total of five test points were evaluated at regular intervals in time ($t = 0, 0.4, 0.8, 1.2, 1.6$). All tests passed successfully with a relative error threshold of 5×10^{-2}, indicating that the computed solution remains highly accurate throughout the simulation.

12.2.3 Heat Equation with Pseudo-Differential Operator

This example demonstrates the numerical solution of a 2D heat equation using the `PDESolver` class with pseudo-differential operators. The equation is defined as:

$$\frac{\partial u}{\partial t} = \Psi\mathrm{Op}(-\xi^2 - \eta^2, u)$$

which corresponds to the standard heat equation $\partial_t u = \Delta u$ in physical space. The pseudo-differential operator $-\xi^2 - \eta^2$ in Fourier space represents the Laplacian operator.
The solver is configured with:
- Spatial domain: $[0, 2\pi]^2$ with 512×512 grid points
- Time domain: $t \in [0, 2.0]$ with 100 time steps
- Initial condition: $u(x, y, 0) = \sin(x)\sin(y)$
- Exact solution: $u(x, y, t) = \sin(x)\sin(y)e^{-2t}$

The numerical solution achieves high accuracy with relative errors below 3.1×10^{-3} across all test times ($t = 0, 0.5, 1.0, 1.5, 2.0$), demonstrating the solver's effectiveness in handling dissipative systems through spectral methods. The `psiOp` implementation successfully captures the exponential decay behavior inherent to the heat equation, validating its ability to handle pseudo-differential operators with polynomial symbols.

12.2.4 Schrödinger Equation

The PDESolver was tested on the 2D Schrödinger equation:

$$i\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

with a Gaussian wave packet as initial condition:

$$u(x, y, 0) = e^{-x^2 - y^2} e^{i(x+y)}.$$

The exact analytical solution is:

$$u(x, y, t) = \frac{1}{\sqrt{(1 + 4it)^2}} e^{i(x+y-2t)} e^{-\frac{(x+2t)^2 + (y+2t)^2}{1+4it}}.$$

The simulation was run over a domain of size $L_x = L_y = 20$ with $N_x = N_y = 512$ grid points and a total time $T = 2.0$ using the ETD-RK4 time integration scheme.
A series of 5 tests were performed at equally spaced times across the simulation interval, comparing the numerical solution to the exact one. The error metric used was the relative L^2-norm of the absolute value of the solution. All tests passed successfully with an error well below the threshold of 5 (as defined by the parameter `threshold=5`), demonstrating both the accuracy and robustness of the solver for dispersive wave equations in two spatial dimensions.

12.2.5 Fractional Diffusion Equation

The PDE solver was tested on the **2D fractional diffusion equation**:

$$\frac{\partial u}{\partial t} = -(-\Delta)^{\alpha/2} u,$$

with $\alpha = 1.5$, corresponding to a fractional-order diffusion process in two spatial dimensions. The Laplacian operator $-\Delta$ is represented in Fourier space as $(k_x^2 + k_y^2)$, and its fractional power is implemented using the custom symbolic operator `Op`. The domain used was $[-\pi, \pi]^2$ with periodic boundary conditions and a grid of 128×128 points. The simulation ran from $t = 0$ to $t = 1$ with 100 time steps.
The initial condition was chosen as:

$$u(x, y, 0) = \sin(x)\sin(y),$$

and the exact solution at time t is given by:

$$u_{\text{exact}}(x, y, t) = \sin(x)\sin(y)\exp\left(-t(1^2 + 1^2)^{\alpha/2}\right).$$

The solver was validated against this exact solution at five equally spaced times over the simulation interval. The relative error norm was computed for each time step, and all errors were found to be below the threshold value of 5×10^{-2}, confirming the accuracy and convergence of the numerical implementation.
For instance, at $t = 0.2$, the relative error was measured at 3.1×10^{-2}, and at the final time $t = 1$, it remained below 4.7×10^{-2}, demonstrating that the solver maintains good precision even over longer integration periods. These results validate the correct implementation of both the fractional derivative operator and the exponential time-stepping scheme.

12.2.6 Wave Equation Solver

The solver was tested on the two-dimensional wave equation:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

with initial condition $u(x, y, 0) = \sin(x)\sin(y)$ and zero initial velocity. The simulation domain was set to $L_x = L_y = 2\pi$ with $N_x = N_y = 512$ grid points, over a total time $T = 2.0$ using $N_t = 400$ time steps. The exponential time differencing Runge-Kutta 4th order (ETD-RK4) scheme was employed for temporal integration.
The exact solution used for validation is:

$$u_{\text{exact}}(x, y, t) = \sin(x)\sin(y)\cos(\omega t), \quad \text{with } \omega = \sqrt{2}.$$

Automatic testing was performed at 5 evenly spaced time points ($t = 0, 0.5, 1.0, 1.5, 2.0$), comparing the numerical solution to the exact one using the absolute norm. All tests passed successfully, with relative errors well below the threshold value of 5. This confirms the high accuracy and stability of the spectral method combined with the ETD-RK4 integrator in capturing the dynamics of dispersive wave propagation.

12.2.7 Wave Equation with Source Term

A validation test was performed on the second-order PDE:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - u + \cos(x)\cos(y)\left(5\cos(\sqrt{3}t) - \sqrt{3}\sin(\sqrt{3}t)\right),$$

with initial conditions $u(x, y, 0) = \cos(x)\cos(y)$ and $\partial_t u(x, y, 0) = 0$. The solver used the ETD-RK4 time integration scheme on a 128×128 spatial grid over a total simulation time of $T = 2.0$. The numerical solution was compared to the exact analytical solution:

$$u_{\text{exact}}(x, y, t) = \cos(x)\cos(y)\cos(\sqrt{3}t)$$

at four equally spaced time points using the relative error norm. All tests passed successfully, with maximum relative errors remaining well below the threshold value of 5. This demonstrates that the solver accurately captures both the temporal dynamics and the response to external source terms.

12.2.8 Wave Equation with Pseudo-Differential Operator

This example demonstrates the solution of the 2D wave equation using the `PDESolver` class with a pseudo-differential operator formulation. The equation is defined as:

$$\frac{\partial^2 u}{\partial t^2} = \Psi\text{Op}(-\xi^2 - \eta^2, u)$$

where ΨOp denotes the pseudo-differential operator with symbol $-(\xi^2 + \eta^2)$. The solver automatically detects the pseudo-differential structure and rejects classical linear terms when such operators are present. The spatial domain is $[0, 2\pi]^2$ with 512×512 grid points, and simulations run until $T = 2.0$.

The pseudo-differential operator corresponds to the Laplacian Δ in physical space, with symbol $\sigma(\xi, \eta) = -(\xi^2+\eta^2)$. Numerical solutions match the exact solution $u(x, y, t) = \sin(x)\sin(y)\cos(\sqrt{2}t)$ with relative errors below 9% even at peak deviation ($t = 1.0$).

The solver successfully handles the second-order temporal structure with leap-frog integration, maintaining numerical stability through pseudo-spectral methods and dealiasing. The test results validate the implementation's accuracy while highlighting typical error growth patterns in dispersive systems.

12.2.9 Klein-Gordon Equation

The numerical solver was tested on the two-dimensional Klein-Gordon equation:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) - m^2 u,$$

with parameters $c = 1.0$ and $m = 1.0$. A plane wave initial condition was used:

$$u(x, y, 0) = \sin(x)\sin(y), \quad \frac{\partial u}{\partial t}(x, y, 0) = 0.$$

The exact solution for this configuration is:

$$u_{\text{exact}}(x, y, t) = \sin(x)\sin(y)\cos(\omega t), \quad \text{with } \omega = \sqrt{c^2(k_x^2 + k_y^2) + m^2}.$$

The simulation domain was set to $[-\pi, \pi] \times [-\pi, \pi]$ with 512×512 grid points and a final time of $T = 2.0$. The solver computed the solution using exponential time stepping and second-order temporal integration.

Automatic validation was performed at five evenly spaced times across the simulation interval using the `test()` method, comparing the numerical solution to the exact one. The error threshold was set to `threshold` $= 5$, and all tests passed successfully. This indicates that the relative or absolute error remained below 5% throughout the simulation, confirming the solver's accuracy and stability for dispersive wave propagation in two dimensions.

Additionally, the total energy of the system was plotted both linearly and logarithmically, showing good conservation over time, which further validates the correctness of the implementation for the Klein-Gordon dynamics.

12.2.10 Biharmonic Equation

This example demonstrates the solution of the 2D biharmonic equation:

$$\frac{\partial u}{\partial t} = -\left(\frac{\partial^4 u}{\partial x^4} + 2\frac{\partial^4 u}{\partial x^2 \partial y^2} + \frac{\partial^4 u}{\partial y^4}\right)$$

with periodic boundary conditions on a square domain $[0, 2\pi] \times [0, 2\pi]$. The exact solution is:

$$u(x, y, t) = \sin(x)\sin(y)e^{-4t}$$

The solver automatically identifies the linear operator in Fourier space:

$$L(k_x, k_y) = -k_x^4 - 2k_x^2 k_y^2 - k_y^4$$

which corresponds to a fourth-order parabolic equation with dispersion relation $\omega(k) = i(k_x^4 + 2k_x^2 k_y^2 + k_y^4)$.
Numerical validation shows excellent agreement with the exact solution:
- Initial condition error: 3.07×10^{-3}
- Final time error at $t = 2.0$: 6.09×10^{-2}.

13 Conclusion

The `PDESolver` class provides a powerful and flexible framework for solving PDEs using spectral methods. It combines symbolic manipulation with efficient numerical techniques to support:
- First- and second-order time evolution.
- One- and two-dimensional domains.
- Nonlinear dynamics with pseudo-spectral evaluation.
- Comprehensive validation and visualization capabilities.

The author indicates that all the code and documentation was generated using LLMs (and a lot of copy-pasting). The project lasted just under two months, with 40 hours of work per week. The author stopped the development of the project for theoretical reasons, as the implementation of the fractional time derivative seemed impossible in the context of Fourier transforms (especially for orders between 0 and 1). However, a version handling systems of two equations is being seriously considered, as it seems that the current form of development allows this quite easily. From a validation point of view, the author has tried to implement good practices in order to best test the various possible cases, and any contribution in this area is welcome. Finally, these developments have been a very good experience...