Numerical Project: Implementation and Analysis of 3D Poisson Solvers for gravity

P. OCVIRK, Numerical Simulations Course

September 26, 2025

1 Introduction

1.1 Astrophysical Context

Gravity is the dominant force shaping the structure of the universe at large scales. From the cosmic web of dark matter to galactic dynamics, gravitational interactions govern the evolution of cosmic structures. The collective gravitational effect of matter determines how galaxies cluster, how dark matter halos form, and how the large-scale structure of the universe evolves over billions of years.

In astrophysical simulations, efficient gravity solvers are therefore paramount to our computational efforts. The evolution of cosmic structures requires solving the gravitational field equations at each timestep of a simulation, and the computational efficiency of these solvers directly impacts our ability to model and understand the universe. Whether we're studying galaxy formation, dark matter structure, or stellar dynamics, the Poisson equation for gravity lies at the heart of most computational astrophysics codes.

This project proposes to explore the numerical formulation of the Poisson equation describing gravity, along with several 3D Poisson solvers. We will investigate their inner workings, performance characteristics, scaling behavior, and convergence properties. Through this exploration, you will gain insight into both the fundamental physics of gravitational systems and the computational techniques that make modern astrophysical simulations possible.

2 Theoretical Background

2.1 The Poisson Equation

The gravitational potential ϕ is related to the mass density ρ through Poisson's equation:

$$\nabla^2\phi = 4\pi G\rho \tag{1}$$

where G is Newton's gravitational constant. The gravitational field $\vec{g}$ is then given by:

$$\vec{g} = -\nabla\phi \tag{2}$$

For this project, we will work with the normalized form:

$$\nabla^2\phi = \rho \tag{3}$$

where we have absorbed the $4\pi G$ factor into ρ for simplicity.

2.2 Discretization and Matrix Formulation

On a regular 3D grid with spacing h, we discretize the Laplacian operator using second-order central differences:

$$(\nabla^2\phi)_{i,j,k} = \frac{\phi_{i+1,j,k} + \phi_{i-1,j,k} + \phi_{i,j+1,k} + \phi_{i,j-1,k} + \phi_{i,j,k+1} + \phi_{i,j,k-1} - 6\phi_{i,j,k}}{h^2} \tag{4}$$

For a grid of size $N \times N \times N$, we can reshape the 3D arrays ϕ and ρ into vectors of length N^3, transforming the problem into a linear system:

$$A\phi = \rho \tag{5}$$

With periodic boundary conditions, the matrix A acquires several special properties that are crucial for understanding the behavior of our solvers. The diagonal elements are all -6, representing the central point in our finite difference stencil, while each row contains exactly six off-diagonal elements with value 1, connecting each grid point to its six nearest neighbors. The resulting matrix is symmetric and negative definite, but importantly, it is also singular since constant solutions lie in its null space. This singularity means that a solution exists only when the integrated density vanishes, that is, $\int \rho dV = 0$, corresponding to zero mean density across the computational domain.

2.3 Iterative Methods

2.3.1 Jacobi Method

The Jacobi method represents the simplest iterative approach to solving our linear system. At each iteration, every grid point is updated based on the values of its neighbors from the previous iteration:

$$\phi^{(n+1)}_{i,j,k} = \frac{1}{6}\left(\phi^{(n)}_{i+1,j,k} + \phi^{(n)}_{i-1,j,k} + \phi^{(n)}_{i,j+1,k} + \phi^{(n)}_{i,j-1,k} + \phi^{(n)}_{i,j,k+1} + \phi^{(n)}_{i,j,k-1} - h^2\rho_{i,j,k}\right) \tag{6}$$

In matrix form, this iteration can be expressed as:

$$\phi^{(n+1)} = D^{-1}(L + U)\phi^{(n)} + D^{-1}\rho \tag{7}$$

where D is the diagonal of A, and L and U are the strictly lower and upper triangular parts, respectively. While the Jacobi method is straightforward to implement and parallelize, its convergence rate deteriorates significantly as the grid size increases.

2.3.2 Successive Over-Relaxation (SOR)

The SOR method builds upon the Gauss-Seidel iteration by introducing an over-relaxation parameter ω that allows the algorithm to take larger steps in the direction of convergence:

$$\phi_{i,j,k}^{(n+1)} = (1-\omega)\phi_{i,j,k}^{(n)} + \frac{\omega}{6}\left(\phi_{i+1,j,k}^{(n)} + \phi_{i-1,j,k}^{(n+1)} + \phi_{i,j+1,k}^{(n)} + \phi_{i,j-1,k}^{(n+1)} + \phi_{i,j,k+1}^{(n)} + \phi_{i,j,k-1}^{(n+1)} - h^2\rho_{i,j,k}\right) \tag{8}$$

The key advantage of SOR lies in its use of updated values as soon as they become available within each iteration. For 3D periodic problems, the optimal relaxation parameter is given by:

$$\omega_{\text{opt}} = \frac{2}{1+\sin(\pi/N)} \tag{9}$$

When $\omega = 1$, the method reduces to Gauss-Seidel, while values between 1 and 2 provide over-relaxation that can dramatically accelerate convergence.

2.3.3 Conjugate Gradient Method

The Conjugate Gradient method approaches the problem from an optimization perspective, solving $A\phi = \rho$ by minimizing the quadratic form:

$$f(\phi) = \frac{1}{2}\phi^T A\phi - \phi^T\rho \tag{10}$$

This method generates a sequence of conjugate (A-orthogonal) search directions and provides theoretically exact convergence in at most N^3 steps for an N^3-dimensional system. In practice, CG often converges much faster than this theoretical upper bound, especially when the matrix has favorable spectral properties. Wikipedia has a very good page about this method and an algorithm you can implement directly.

3 Project Tasks

3.1 Physical Setup

Consider a cubic domain with periodic boundary conditions. The density field should represent a realistic astrophysical configuration with both large-scale structure and small-scale fluctuations:

$$\rho(x,y,z) = e^{-[(x-L/2)^2+(y-L/2)^2+(z-L/2)^2]/(2\sigma^2)} + \eta(x,y,z) \tag{11}$$

Here, $\sigma = 0.2L$ defines the characteristic size of the main density concentration, while η represents Gaussian random noise with standard deviation 0.1, simulating smaller-scale density fluctuations. It is crucial to ensure that the resulting density field has zero mean, as this is required for the existence of a solution with periodic boundary conditions.

3.2 Part 1: Jacobi Method

Begin your investigation with the Jacobi method, implementing it first in single precision arithmetic. Test your implementation across a range of grid sizes: $N = 8, 16, 32, 48, 64$, carefully monitoring the maximum residual as a function of iteration number. The residual should be computed as the L2 norm of $\rho - \nabla^2\phi$, not simply the change in solution between iterations.

After establishing the single-precision behavior, repeat the entire analysis using double precision arithmetic. This comparison will reveal the interplay between numerical accuracy and iterative convergence, particularly highlighting cases where round-off errors limit the achievable precision.

Pay special attention to any plateaus that appear in the residual evolution. These often indicate that the solver has reached the limits imposed by floating-point precision rather than the true solution accuracy. Understanding this distinction is crucial for interpreting the results of large-scale simulations.

3.3 Part 2: SOR Method

Implement the SOR method using the theoretically optimal value of ω, then systematically compare its convergence behavior with your Jacobi implementation. The improvement should be dramatic, particularly for larger grid sizes where the Jacobi method becomes prohibitively slow.

Investigate the sensitivity of the method to deviations from the optimal ω value by testing a range of relaxation parameters around ω_{opt}. This analysis will help you understand both the robustness of the method and the importance of proper parameter tuning. Consider what happens when ω exceeds 2, as this violates the stability condition and should lead to divergence.

Document how the convergence rate scales with grid size for the SOR method, and compare this scaling with your Jacobi results. The difference should highlight the superior performance of SOR for larger problems.

3.4 Part 3: Conjugate Gradient Method

Implement the Conjugate Gradient algorithm, referring to standard references such as Wikipedia for the explicit algorithmic steps. Once your implementation is working, compare its performance directly with the scipy implementation at `https://docs.scipy.org/doc/scipy-1.16.2/reference/generated/scipy.sparse.linalg.cg.html` to verify correctness and assess the quality of your implementation.

Conduct a comprehensive comparison with the previous methods, examining not only convergence speed but also computational cost per iteration, memory usage, and final accuracy. The CG method typically requires more memory due to the need to store multiple search directions, but this overhead may be justified by faster convergence, especially for well-conditioned problems.

Consider which method you would choose for different problem sizes and accuracy requirements. The answer may depend on factors such as available memory, desired precision,

and the specific properties of the density field.

4 Analysis Requirements

4.1 Performance Metrics

Your analysis should provide a comprehensive picture of each method's performance characteristics. Plot the convergence rate (residual versus iteration number) for each method and grid size, using log-scale plots to clearly show the exponential decrease of residuals. Measure and plot computational time as a function of grid size, looking for power-law relationships that reveal the algorithmic scaling.

Memory usage analysis is equally important, as this often becomes the limiting factor in large simulations. Document how memory requirements scale with problem size for each method, and use log-log plots to determine the scaling exponents. These scaling relationships will help predict the feasibility of applying these methods to even larger problems.

4.2 Accuracy Analysis

Compare the final solutions obtained by different methods to assess their consistency and accuracy. Even when all methods converge, small differences in the solutions can provide insight into their relative strengths and limitations. Examine the spatial distribution of residuals to identify any systematic patterns or regions where particular methods struggle.

The impact of floating-point precision deserves special attention. Document cases where single precision proves inadequate, and explain why double precision may be necessary for certain applications. This analysis connects directly to practical considerations in scientific computing.

5 Report Guidelines

Your report should present a clear methodology description that would allow another researcher to reproduce your results. Include well-labeled figures and plots that effectively communicate your findings, ensuring that axis labels, legends, and captions provide sufficient detail for interpretation.

The analysis of results should go beyond mere description to provide physical and numerical insights. Discuss the numerical challenges you encountered and how you addressed them. Your conclusions about method suitability should be supported by the data and should consider different scenarios a computational astrophysicist might face.

The report should demonstrate both technical competence and scientific communication skills, as both are essential for success in computational physics research.

6 Bonus Tasks

For students seeking additional challenges, consider studying different density distributions to understand how the structure of the source term affects solver performance. Explore the potential for parallel implementation, considering how the algorithms might be adapted for modern multi-core processors or distributed computing environments.

A comparison with FFT-based solutions could provide valuable perspective on the relative merits of direct versus iterative approaches. For the most ambitious students, implementing a multigrid solver offers insight into one of the most powerful techniques for solving elliptic partial differential equations, though this represents a substantial undertaking that should be approached with appropriate caution and preparation.