Three-dimensional cylindrical Poisson solver with vacuum boundary conditions

S Moon
Department of Physics & Astronomy, Seoul National University, Seoul 08826, Republic of Korea
E-mail: moon@astro.snu.ac.kr

Abstract. Self-gravity and rotation are two key ingredients in dynamics of astronomical disk systems such as galactic and protostellar disks. Despite its importance, there has not yet been so far an efficient algorithm to solve the Poisson equation in three-dimensional cylindrical coordinates under vacuum boundary conditions. By generalizing the James algorithm to cylindrical coordinates, we develop an accurate (second-order convergence) and efficient (faster than MHD) cylindrical Poisson solver that is scalable up to $\sim 10^4$ cores. We develop a method to calculate the cylindrical discrete Green’s function, which is an essential element of the James algorithm to establish its second-order accuracy. We implement our cylindrical version of the James algorithm in Athena++ code and demonstrate its accuracy and efficiency by performing the convergence test and the weak scaling test.

1. Introduction
Self-gravity often plays an important role in dynamical evolution of astronomical gas disks (e.g., [1–3]). To follow such evolution, it is crucial to solve the Poisson equation together with equations of (magneto-)hydrodynamics (MHD). In some cases, it is sufficient to assume the disk is axisymmetric or razor-thin and use analytic expressions given in, e.g., [4]. In reality, however, non-axisymmetric structures are always present and of great interest (e.g., [5, 6]) whose investigation requires a fully general solution of the Poisson equation,

$$\nabla^2 \Phi = 4\pi G \rho. \quad (1)$$

In numerical simulations, it is therefore critical to develop an efficient algorithm to solve Equation (1) in a discretized computational domain. Because polar coordinates is a natural coordinate system for rotating disks, there have been many developments to solve the Poisson equation in two-dimensional cylindrical grid [7–11]. In contrast, there has been little progress in developing fast and accurate Poisson solver working in three-dimensional cylindrical grid.

The main difficulty associated with three-dimensional cylindrical Poisson solver is a proper handling of the vacuum boundary conditions. In principle, all methods that involve integration with the Green’s function are able to produce correct (i.e., consistent with vacuum boundary condition) gravitational potential. One such method is the multipole expansion method, in which the Green’s function is expanded in terms of spherical harmonics and only first few terms are kept for computational efficiency. In spherical coordinates, one needs to calculate only a single interior and exterior multipole moments for each harmonics, because all mass is bounded...
between the inner and outer radial boundary. For example, the (interior) monopole moment is always equal to the total mass for any cell at the outer radial boundary. In cylindrical coordinates, however, the limits of integration for the multipole moments are different for most boundary cells and one has to calculate separate sets of multipole moments for each boundary cells [12]. For example, the (interior) monopole moment at the top boundary is different from the total mass in general, and varies from cell to cell. This significantly increases the computational cost, making multipole expansion quite expensive in cylindrical grid. In addition, flattened mass distributions that are frequently simulated in cylindrical grid requires higher multipole moments for accurate potential, further raising the computational cost.

Another common method based on the Green’s function is the convolution method. In Cartesian coordinates, the integration with the Green’s function takes the form of multi-dimensional convolution, which can be computed very efficiently with an aid of the fast fourier transform (FFT) [13]. In three-dimensional cylindrical coordinates, however, the integration with the Green’s function does not take the form of fully three-dimensional convolution so that the FFT convolution technique apply only to the two of the three dimensions. The resulting computational complexity is super-linear, requiring $O(N^{4/3} + N \log N)$ operations where $N$ is the total number of grid cells, even when only the boundary potential is considered. It is worth to note that, however, for the boundary potential, the computational cost can be lowered to $O(N \log N)$, by using the compact cylindrical Green’s function (CCGF) expression discovered by Cohl and Tohline [12].

Another very interesting method to deal with the vacuum boundary condition is the James algorithm [14]. It can calculate the boundary potential with only $O(N + N^{2/3} \log N)$ operations, provided that the preliminary solution of the discretized Poisson equation with zero boundary condition is given. This algorithm has been used only in Cartesian coordinates, but recently we extended it into cylindrical coordinates [15]. In this proceeding, we review the James algorithm in a language that is coordinate-independent as possible, to emphasize that the James algorithm is not necessarily restricted to Cartesian coordinates. We refer the reader to [15] for detailed description of the James algorithm in cylindrical coordinates.

2. Method

Figure 1. A schematic figure describing the James algorithm. The zero electrostatic potential at the surface is a sum of a contribution from the electrons and positive charges. Subtracting the contribution from electrons gives correct boundary condition for the electrostatic potential generated from the positive charges under vacuum boundary conditions.
The idea of the James algorithm can be best understood via electrostatic analogy. Consider a metallic object where positive electric charges are glued in its interior. While the electrostatic potential inside the metal under the vacuum boundary condition can be found by integrating the positive charges with the Green’s function, it can be more efficiently found by solving discretized Poisson equation on a grid, provided that the boundary potential at the surface is known a priori. The boundary potential can in principle be found by volume integration of the positive charges with the Green’s function, but it is computationally expensive in general. The James algorithm provides alternative route of finding the boundary potential. Suppose that the metal is grounded (Figure 1); electrons will then be distributed along the surface of the metal, until the potential becomes zero at the surface. Because this zero potential is a sum of a contribution from the positive charges and the electrons, it is possible to find the desired boundary potential due to the positive charges by instead calculating the contribution from the electrons and subtracting it from zero. Because electrons are located only at the surface, it only requires two-dimensional integration to find the boundary potential, thus saving the computational cost.

\[ \Psi = 0 \]

Figure 2. Calculation of the screening charge. The discrete Poisson equation is solved in \( \mathcal{R} \) with zero boundary condition given at \( \partial \mathcal{R} \). The screening charge \( \sigma \) is found by applying the discrete Laplace operator to \( \partial \mathcal{R} \), where the potential exterior to \( \partial \mathcal{R} \) is taken to be zero (see text).

To find the charge distribution due to the screening electrons, one may apply Gauss’s law at the surface, after solving the Poisson equation with zero boundary conditions. In a discrete domain, one need to solve the discretized Poisson equation with zero boundary conditions, and apply discrete Laplace operator at the boundary to find the screening charges (i.e., electrons) \[15\]. This is demonstrated in Figure 2, where a two-dimensional discrete Cartesian domain \( \mathcal{R} \) is shown for demonstration. The solution of the discrete Poisson equation \( \Psi \) with the zero boundary condition \( \Psi = 0 \) at \( \partial \mathcal{R} \) is defined only inside the domain \( \mathcal{R} \), represented by the thick solid lines. However, to apply the discrete Laplace operator at \( \partial \mathcal{R} \) to find the screening charge \( \sigma \), one need information at extra cells exterior to \( \partial \mathcal{R} \), where \( \Psi \) is undefined.

What value of \( \Psi \) should one take at this undefined region? this question can be answered by following arguments. Suppose that the domain of definition of \( \Psi \) is extended to the infinite discrete domain \( \mathcal{R}_\infty \), which comprises all grid cells in the infinite discrete Cartesian grid (including \( \partial \mathcal{R} \) and \( \mathcal{R} \)). By setting \( \Psi = 0 \) at newly defined cells outside \( \partial \mathcal{R} \), one can see that \( \Psi \) satisfies the discrete Poisson equation at every cell in the infinite discrete domain. One can therefore calculate \( \sigma \) by applying the discrete Laplace operator at \( \partial \mathcal{R} \) while taking \( \Psi = 0 \) at the cells exterior to \( \partial \mathcal{R} \). In this way, \( \Psi \) satisfies the discrete Poisson equation at every cells.
including $\partial R$. Note that $\Psi$ is consistent with the vacuum boundary condition, because $\Psi = 0$ at infinity.

To find the potential due to the screening charges, one need to integrate them with the Green’s function. In a continuous domain, a Green’s function $G = -G/|x - x'|$ of the Poisson equation is determined in relation to the Laplace operator, through a defining equation

$$\nabla^2 G = 4\pi G \delta(x - x'),$$

(2)

where $\delta(x)$ is the Dirac delta function. In the James algorithm, however, the Green’s function should be defined in relation to the discrete Laplace operator, because the screening charges themselves are determined by the application of the discrete Laplace operator. We define the discrete Green’s function as a solution of a finite-difference equation,

$$\Delta^2 G_{i,j,k} = 4\pi G \delta_{ii} \delta_{jj} \delta_{kk} / V_{i,j,k},$$

(3)

where $\delta_{ii}$ is the Kronecker delta function, $V_{i,j,k}$ is the cell volume, and $\Delta^2$ is the discrete Laplace operator, whose exact form depends on a coordinate system and a desired level of approximation.

Figure 3. Comparison of the discrete (red open circles) and continuous (blue solid line) Green’s function in a logarithmic cylindrical grid. A point source is located at the first ghost cell near the inner radial boundary $R_{\text{min}} = 0.01$. Vertical black solid lines mark the faces of first few cells, where the outer radial boundary is located at $R_{\text{max}} = 1$, outside the plot range.

As a sample calculation, we solve Equation (3) in a logarithmic cylindrical grid $R \in [0.01, 1]$, $\phi \in [0, 2\pi]$, and $z \in [-0.25, 0.25]$, where the point source (represented by the Kronecker delta functions) is placed at the inner radial boundary. We use second-order accurate finite-difference approximation in logarithmic cylindrical coordinates (Section 2.3 in [15]). Figure 3 plots the resulting discrete Green’s function along the radial coordinates, together with a continuous Green’s function, $-G/|x - x'|$, for comparison. We set the gravitational constant $G = 1$. It is evident that the discrete Green’s function significantly deviates from the continuous Green’s function, especially near the source position. Note that the discrete Green’s function has a finite value even at the source location, at which the continuous Green’s function is singular. Because the discrete Green’s function depends only on the underlying grid structure and not on the mass distribution, it only needs to be calculated once and for all.
The potential due to the screening charges can be obtained by integrating $\sigma$ over the discrete Green’s function, such that

$$\Theta_{ijk} = \sum_{i',j',k'} G_{i,i',j,j',k,k'} \sigma_{i',j',k'} V_{i'}.$$  \hspace{1cm} (4)

Note that we only need $\Theta$ at the boundary, and that the $\sigma$ is nonzero only at the boundary. The computational cost associated with Equation (4) is then $O(N^{4/3})$, which is still super-linear. In cylindrical coordinates, however, the azimuthal summation can be done via FFT such that the Equation (4) can be evaluated with $O(N + N^{2/3} \log N)$ operations [15].

Subtracting $\Theta$ from zero yields the boundary potential which would have been produced by the positive charges before the metal is grounded. Therefore, $-\Theta$ is the correct boundary condition for solving the discretized Poisson equation in a finite computational domain under the vacuum boundary condition. We use FFT and tridiagonal matrix algorithm to solve the discretized Poisson equation, but other methods also can be used. For example, [16] used FFT with the cyclic reduction method, and [17] used FFT with the alternating direction implicit (ADI) method to solve the discretized Poisson equation in three-dimensional cylindrical grid. Regardless of the detailed algorithm, we term them interior solver to denote any methods that solve the discretized Poisson equation under a given boundary condition. As a counterpart, we use the term boundary solver to denote the parts of the James algorithm that are responsible for the calculation of $\sigma$ and $\Theta$.

Figure 4. A flowchart of the James algorithm. The discrete Green’s function is computed once and for all, before a main simulation loop begins. In the main loop, the James algorithm consists of four steps: 1) initial solve with zero boundary potential, 2) screening charge calculation, 3) boundary potential calculation, and 4) second solve with physical boundary condition.

Figure 4 shows a flowchart of the James algorithm. At the beginning of the simulation, the discrete Green’s function is computed numerically and stored in memory. In the main simulation loop, the interior solver is invoked with zero boundary condition to simulate the potential of the grounded metal. From this potential, the screening charges at the surface
are evaluated. The potential due to these screening charges are found by integration with the
discrete Green’s function (Equation (4)). Finally, a second call to the interior solver yields the
desired gravitational potential with the vacuum boundary condition. Note that the interior
solver is called twice during a one cycle of the James algorithm.

3. Tests

![Diagram of gravitational potential](image)

**Figure 5.** Sample calculation of the gravitational potential. (a) A top-down view of three-
dimensional computational grid, showing a test mass distribution (uniform rectangular torus segment). (b) Radial profile of the gravitational potential. Blue and red circles show one-
dimensional cut profiles (at the midplane) along the blue and red solid lines in (a), respectively.
Black dashed lines are the analytic solutions along each cut.

We implement our cylindrical version of the James algorithm in the public version of the
Athena++ code (Stone et al. 2019, in prep.). As a test problem, we calculate the gravitational
potential of a rectangular torus segment using our Poisson solver. Left panel of Figure 5
depicts the top-down view of three-dimensional logarithmic cylindrical grid with \( R_{\text{min}} = 0.01, \)
\( R_{\text{max}} = 1, \phi_{\text{min}} = 0, \phi_{\text{max}} = 2\pi, z_{\text{min}} = -0.25, \) and \( z_{\text{max}} = 0.25. \) We set the density \( \rho = 1 \)
inside the torus segment defined by \( 0.31622777 \leq R \leq 0.74989421, 0 \leq \phi \leq 1.1780972, \) and
\(-0.0625 \leq z \leq 0.1875, \) and \( \rho = 0 \) otherwise. The number of cells in each dimension is
\( N_R = N_\phi = N_z = 64. \) We make one-dimensional cuts through the blue and red lines, and
plot the gravitational potential along each cuts in the right panel with corresponding colors.
The analytic solution given by [18] is overplotted with black dashed lines, showing the numerical
solution is exceedingly close to the analytic solution. The relative error of the numerical solution
to the analytic solution averaged over the entire grid is 0.035%. The mean and the maximum
relative errors in different resolution is listed in Table 1. The mean relative error decreases at
a second-order rate as expected, while the maximum relative error decreases slightly shallower
than second-order, which might be due to the discontinuity of the mass distribution.

We measure the average wall clock time taken by the interior solver and the boundary solver
and compare them to the time taken by the MHD solver. Figure 6 shows the wall clock time of
each component normalized by the time taken by the MHD solver, for different number of cores
(or MPI ranks). The number of cells per core is fixed to \( 64^3, \) such that the resolution reaches
\( N_R = N_\phi = N_z = 1024 \) at \( N_{\text{core}} = 4096. \) The time taken by the boundary solver (blue squares)
is 1% of the MHD solver at \( N_{\text{core}} = 1, \) rising to 2.5% at \( N_{\text{core}} = 4096. \) This rise is due to the
Table 1. The relative error in the gravitational potential of the rectangular torus segment

| $N_R = N_\phi = N_z$ | mean relative error | maximum relative error |
|-----------------------|----------------------|------------------------|
| 16                    | $6.14 \times 10^{-3}$| $9.42 \times 10^{-2}$  |
| 32                    | $1.04 \times 10^{-3}$| $4.27 \times 10^{-2}$  |
| 64                    | $3.49 \times 10^{-4}$| $1.61 \times 10^{-2}$  |
| 128                   | $1.02 \times 10^{-4}$| $4.99 \times 10^{-3}$  |
| 256                   | $2.71 \times 10^{-5}$| $1.39 \times 10^{-3}$  |
| 512                   | $6.94 \times 10^{-6}$| $3.85 \times 10^{-4}$  |

two-dimensional MPI communications required for FFT along the azimuthal direction, and the logarithmic complexity of the FFT algorithm itself. Considering the inevitable cost due to the MHD solver, the boundary solver does not contribute to overall computational cost at all. Our interior solver (FFT with tridiagonal matrix algorithm; green circles) is computationally more expensive than the boundary solver, because it requires FFT on three-dimensional data. Since the James algorithm requires two calls to the interior solver and a single call to the boundary solver, the total cost associated with the James algorithm (red stars) is higher than that of interior solver by a factor of two. However, it still takes less time than MHD solver even at $N_{\text{core}} = 4096$, leading us to conclude that our Poisson solver does not significantly contribute to overall computational cost of self-gravitating MHD simulations.

Figure 6. A weak scaling test of our Poisson solver. Each symbols represent the ratio of the average wall clock time taken by the single call to the boundary solver (blue squares), interior solver (green circles), and the total James algorithm (red stars), to that of the MHD solver. Note that the James algorithm consists of two calls to the interior solver and a single call to the boundary solver.
4. Summary and Discussion
We have presented that the James algorithm can be applied to the cylindrical coordinate system. It enables us to find an exact solution of the second-order finite-difference Poisson equation with vacuum boundary condition. We demonstrate an accuracy and efficiency of cylindrical James algorithm using test problems. The boundary solver, which deals with the vacuum boundary condition, takes only a few percent of computational time required by MHD solver. The computational cost associated with our interior solver is higher than the boundary solver, requiring 4 – 25% of the time required by the MHD solver.

Considering the large communication cost associated with FFT in our interior solver, the computational cost may be lowered if one adopts alternative mesh-based algorithms (e.g., multigrid, ADI, etc.) as the interior solver. We also note that the extension of the James algorithm to spherical coordinates is straightforward, if an efficient interior solver is available.

References
[1] Baruteau C, Meru F and Paardekooper S J 2011 MNRAS 416 1971–1982
[2] Tobin J J, Kratter K M, Persson M V, Looney L W, Dunham M M, Segura-Cox D, Li Z Y, Chandler C J, Sadavoy S I, Harris R J, Melis C and Pérez L M 2016 Natur 538 483
[3] Schartmann M, Mould J, Wada K, Burkert A, Durré M, Behrendt M, Davies R I and Burtscher L 2018 MNRAS 473 953–968
[4] Binney J and Tremaine S 2008 Galactic Dynamics (Princeton University Press)
[5] Shetty R and Ostriker E C 2006 ApJ 647 997–1017
[6] Forgan D H, Hall C, Meru F and Rice W K M 2018 MNRAS 474 5036–5048
[7] Miller R H 1976 JCoPh 21 400
[8] Baruteau C and Masset F 2008 ApJ 678 483–497
[9] Shetty R and Ostriker E C 2008 ApJ 684 978–995
[10] Li S, Buoni M J and Li H 2009 ApJS 181 244–254
[11] Wang H H, Yen D C C and Taam R E 2015 ApJS 221 4–19
[12] Cohl H S and Tohline J E 1999 ApJ 527 86–101
[13] Hockney R W and Eastwood J W 1988 Computer Simulation Using Particles (CRC Press)
[14] James R A 1977 JCoPh 25 71–93
[15] Moon S, Kim W T and Ostriker E C 2019 ApJS 241 24
[16] Tohline J E 1980 ApJ 235 866–881
[17] Cohl H S 1999 On the Numerical Solution of the Cylindrical Poisson Equation for Isolated Self-Gravitating Systems Ph.D. thesis Louisiana State University
[18] Huré J M, Trova A and Hersant F 2014 CeMDA 118 299–314