Particle-in-Cell Simulation Using a Fast Poisson Solver with Truncated Green's Functions

Chong Shik Park, Ph.D.

Department of Accelerator Science

Korea University, Sejong

Space Charge Solvers

- Wave Equations or Helmholtz Equations:

$$\left[\vec{\nabla}^2 - \frac{1}{c^2}\frac{\partial^2}{\partial t^2}\right]\begin{Bmatrix}\phi \\ \vec{A}\end{Bmatrix} = -\begin{Bmatrix}\rho/\varepsilon_0 \\ \mu_0\vec{J}\end{Bmatrix}$$

- Poisson Equation:

$$\vec{\nabla}^2\phi = -\frac{\rho}{\varepsilon_0}$$

- General Solution of the Poisson Equation with Green's function

$$\phi(\vec{r}) = \frac{1}{\varepsilon_0}\int G(\vec{r},\vec{r}')\rho(\vec{r}')d^3\vec{r}' = \frac{1}{4\pi\varepsilon_0}\int \frac{1}{|\vec{r}-\vec{r}'|}\rho(\vec{r}')d^3\vec{r}'$$

- Inclusion of boundary conditions makes the problem more complicated.
  - Open boundary condition is preferred.
  - This is true if the pipe radius in an accelerator is much larger than the beam bunch transverse size

Hockney-Eastwood Algorithm

Poisson Solver

- One of most popular algorithms in beam physics codes is Hockney-Eastwood(HE) in which Fast Fourier Transform(FFT) with zero-padding is used thanks to the convolution theorem
- The potential at a mesh point (p, q) can always be written as the sum of contributions from all other source points (p', q')

$$\phi(p,q) = \frac{h_x h_y h_z}{4\pi\varepsilon_0} \sum G(p,q;p',q')\rho(p',q')$$

- The potential as the convolution of the source distribution ρ with the Green's function of the interaction potential G.

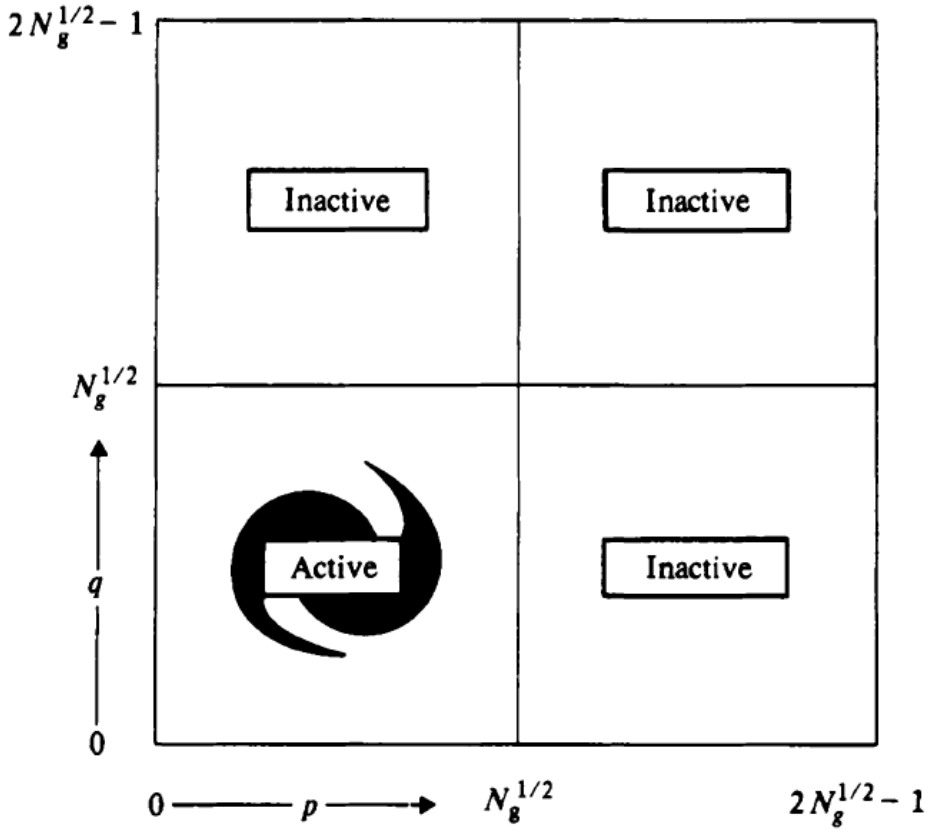
$$\mathcal{F}\{\hat{\phi}\}_{k,l} = \mathcal{F}\{\hat{G}\}_{k,l}\, \mathcal{F}\{\hat{\rho}\}_{k,l}$$

$$\phi(\vec{r}) = \frac{h_x h_y h_z}{4\pi\varepsilon_0} \mathcal{F}^{-1}\left\{\sum \mathcal{F}\{\hat{G}\}\mathcal{F}\{\hat{\rho}\}\right\}$$

- The convolution method will solve a periodic system of sources with an arbitrary form of interaction.
- **No conductors or boundaries are permitted in the system.**
  - This is true if the pipe radius in an accelerator is much larger than the beam bunch transverse size.

Integrated Green's Function

- For a beam with a large aspect ratio, the direct use of the FFT-based method will be inefficient since a large number of grid points are needed to resolve the variation of the Green function inside a grid cell.
- This convolution can be replaced by a cyclic convolution in a double-gridded computational domain.
- This makes use of the symmetry of the Green function
- The potential outside the original domain is incorrect but is irrelevant to the physical domain.



More Green's Function Techniques

2.5 D Hockney-Eastwood Solver

- Can be implemented to the beam with a **large transverse-to-longitudinal aspect ratio**
- The beam is squeezed to a single slice
- Space charge fields are calculated only in the transverse dimensions.
- Then, the fields are scaled with the longitudinal line density to be applied in the all 3-dimensional directions.

Shifted Green's Function

- In order include beam-beam effects or image charge effects, the computational domain need to contain both the particle domain and the field domain, i.e., both beams.
- This results in a poor spatial resolution of the beams because of a large empty space b/w two beams.
- This is also computationally inefficient because the electric fields in the empty space between two beams are not used.
- Using the shifted Green function, **the center of the field domain is shifted to the center of the particle domain.**
- Then, the FFT can be used to calculate the cyclic convolution using the new Green function.

Vico-Greengard-Ferrando Methods

- In the Hockney-Eastwood FFT-based method, the Green's function is defined in long range and has singular at $\vec{r} = \vec{r}'$
- Recently, Vico *et al* introduced **a truncated spectral kernel** for Green's function by replacing as follows:

$$G(\vec{r}) \Longrightarrow\ G^L(\vec{r}) = G(\vec{r})\mathrm{rect}\left(\frac{r}{2L}\right),$$

where $L > \sqrt{d}$, and the indicator function $\mathrm{rect}(x)$ is defined as

$$\mathrm{rect}(x) = \begin{cases} 1, & x < 1/2 \\ 0, & x > 1/2 \end{cases}.$$

- In the Vico-Greengard-Ferrando (VGF) Poisson Solver, the Fourier transform of **the Green's function can be solved analytically** and computed in the truncated dimensions:

$$\mathcal{F}\{G^L\} = \frac{2}{\varepsilon_0}\left[\frac{\sin\left(\frac{L|\vec{k}|}{2}\right)}{|\vec{k}|}\right]^2$$

- Then, the potential becomes:

$$\phi(\vec{r}) = \frac{2}{(2\pi)^3\varepsilon_0}\int e^{i\vec{k}\cdot\vec{r}'}\left[\frac{\sin\left(\frac{L|\vec{k}|}{2}\right)}{|\vec{k}|}\right]^2 \rho(\vec{r}')d^3\vec{k}$$

Implementation of Algorithms and Comparison

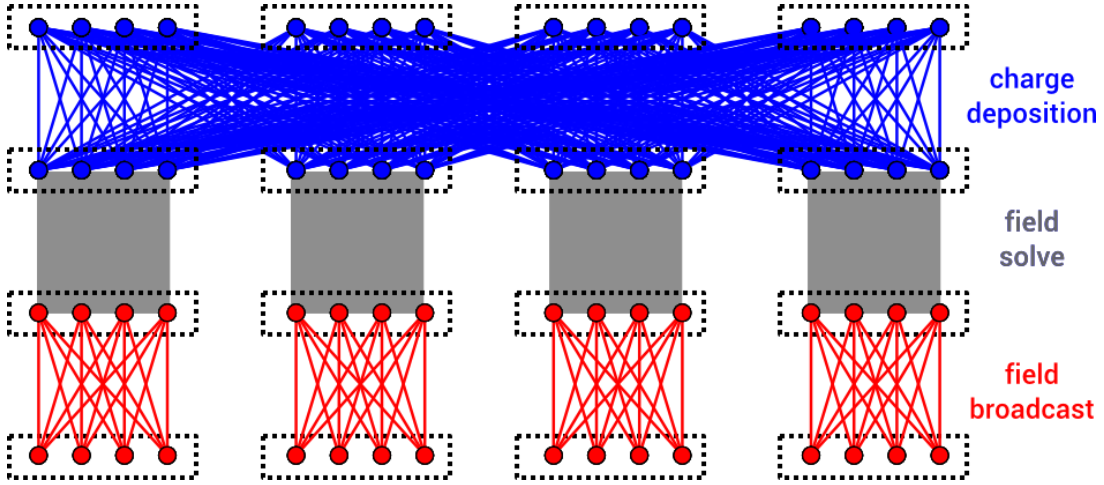
Charge Distribution and Exact Solution

- Using a 3-D Gaussian charge distribution

$$\rho(\vec{r}) = \frac{Q}{\sigma^3(2\pi)^{3/2}} e^{\left(-\frac{r^2}{2\sigma^2}\right)},$$

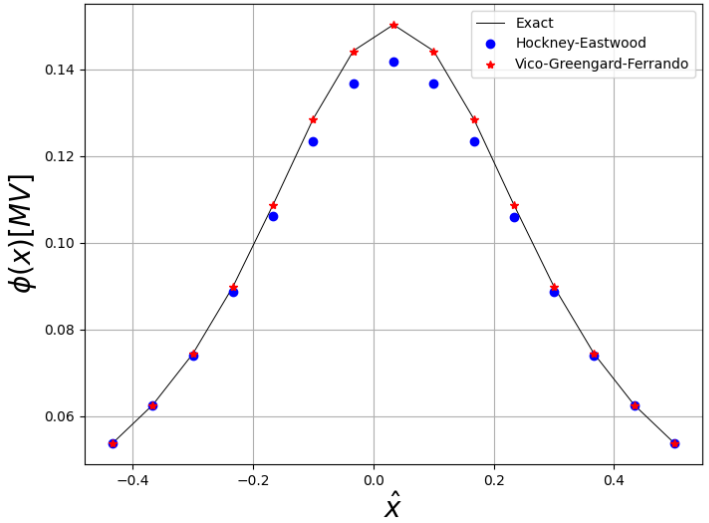
- Generate and deposit macro-particles on grids
  - $N_x \times N_y \times N_z$ grid domain
  - For simplifying the problem, assume that $N_x = N_y = N_z$
- The exact solution of the Poisson equation is

$$\phi(\vec{r}) = \frac{Q}{4\pi\varepsilon_0}\frac{1}{r}\mathrm{erf}\left(\frac{r}{\sqrt{2}\sigma}\right)$$

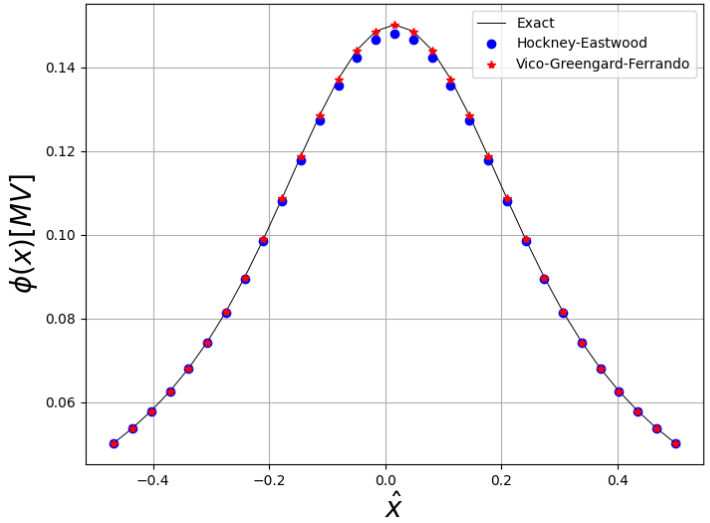


Space Charge Potentials along the Beam Axis

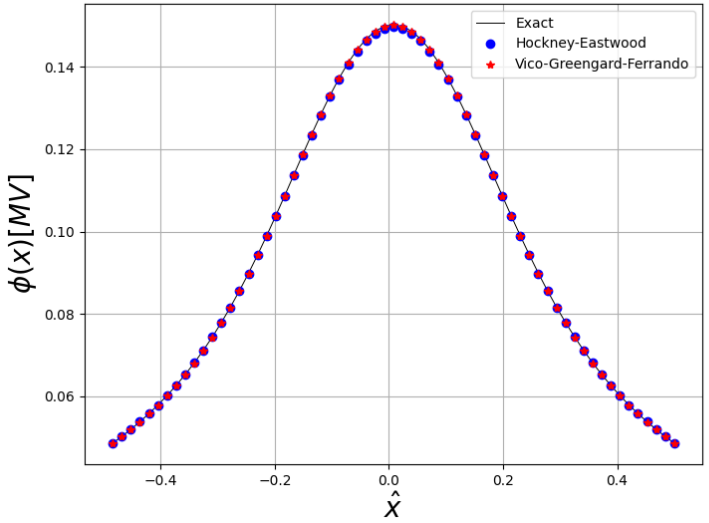
For a different number of grids



N = 16



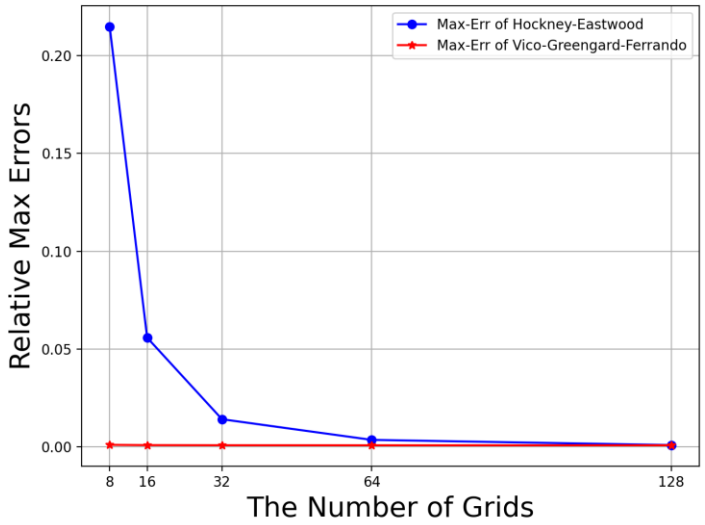
N = 32

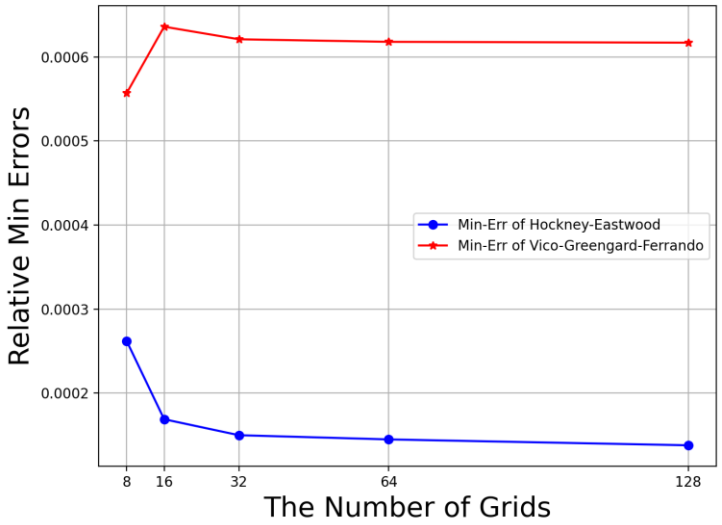


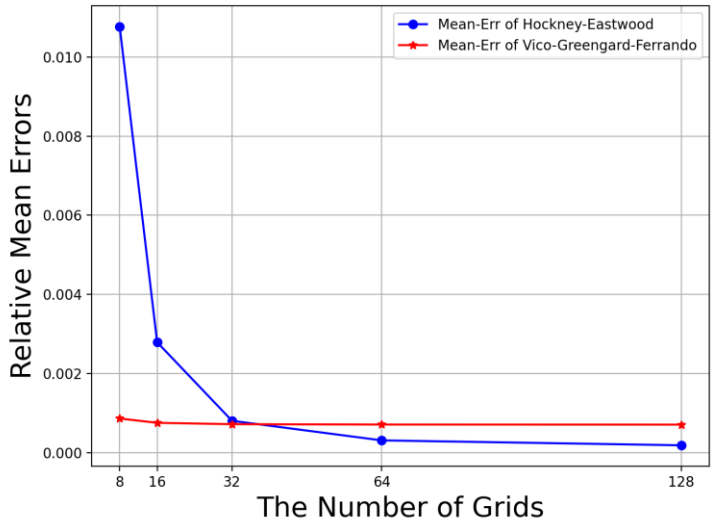
N = 64

- The electric potentials along the longitudinal beam axis for each algorithm with the different number of grids.
- With small N, the potential with the HE algorithm has large deviation at the center of the beam.
- As N is increased, this deviation is decreased.

Numerical Errors

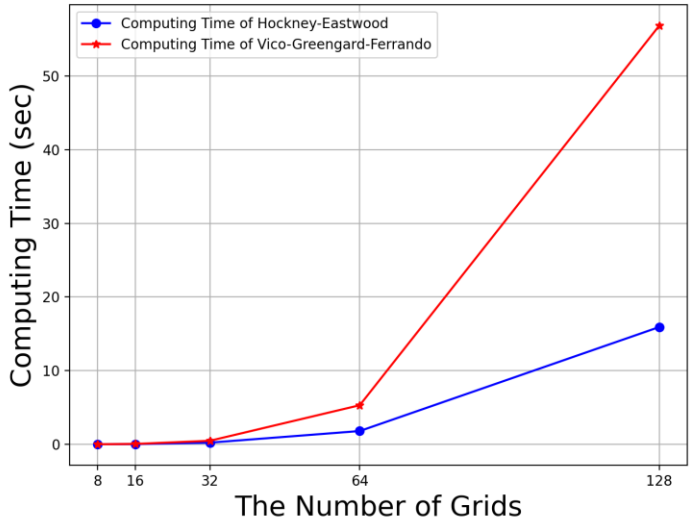


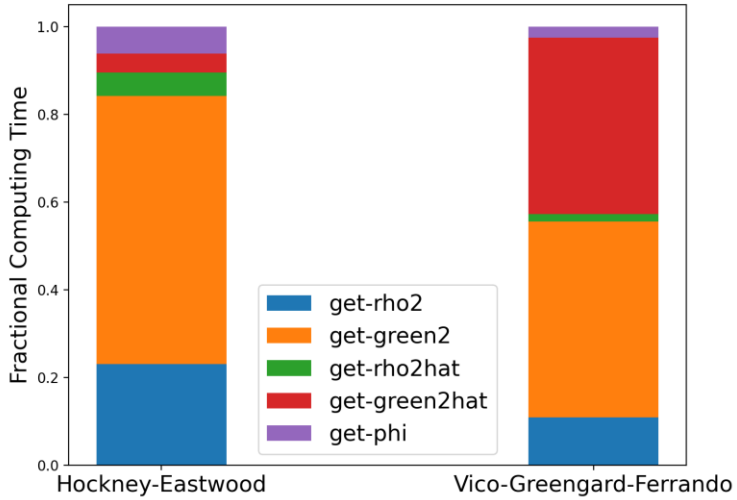




- The relative errors to the exact solutions in both algorithms are compared.
- The VGF algorithm has smaller maximum and mean errors for small grid sizes, but larger minimum errors for all grid sizes.
- The maximum relative error for VGF occurs at the edge of the grid, but at the center for HE.
- In the case of minimum relative error, the opposite is true.
- Moreover, unlike HE, the accuracy of the VGF algorithm does not depend significantly on the number of grid sizes.

Computing Time





- With the VGF algorithm, **the computation time increases significantly** as the number of grids increases.
- However, it can easily compensate for computation time by showing fast convergence with **a smaller number of grids**.
- This can also be overcome by recent advances in **vectorization techniques** for processing large matrices.