

On Several Green's Function Methods for Fast Poisson Solver in Free Space

Dawei Zheng and Ursula van Rienen

Abstract We summarize four closely related numerical solution methods for Poisson's equation in free space: Green's function method, integrated Green's function method, reduced integrated Green's function method, and cutting integrated Green's function method. A new and final routine called cutting reduced Green's function method is carried out as well. These methods can be used for different practical problems to accelerate the calculation. Numerical examples are also given to compare the introduced methods.

1 Introduction

Poisson's equation is broadly used in many areas, such as electrostatics, mechanical engineering and theoretical physics—for instance in gravitational potential calculation and in beam dynamics simulations in particle accelerators. Particle accelerators have a long history. In fact, the first basics go back to Crookes who discovered cathode rays (1870), Thompson who showed that the cathode rays are composed of electrons (1896) and Röntgen who discovered X-rays (1895). The first milestone on the path to particle accelerators for high energy physics was Rutherford's scattering of alpha particles on a gold foil (1909). Modern accelerators for high energy physics still basically use the principle of scattering experiments. Since the energy of the electrons in cathodic ray tubes is limited, in the 1930s new types of generators for higher electric fields have been developed. Examples are the Van de Graff generator (1929) and the Cockcroft-Walton generator (1932) or the first cyclotron by Lawrence (1931). To overcome the limitations of these machines and achieve much higher energies of the electrons, radio-frequency (RF) cavities started to be used (and now are key elements of all accelerators) in which the energy of the high-frequency field is transferred to the passing electrons (or other elementary particles). Accelerators for high energy physics are either ring-like machines such as the Large Hadron Collider (LHC) at CERN or linear accelerators such as the

D. Zheng (✉) • U. van Rienen

Institute of General Electrical Engineering, University of Rostock, Albert-Einstein-Str. 2, 18059 Rostock, Germany

e-mail: dawei.zheng@uni-rostock.de; ursula.van-rienen@uni-rostock.de

design study for the International Linear Collider (ILC). The elementary particles are highly relativistic, i.e. have practically speed of light, and have very high energies. Synchrotron light sources, which are used for material studies, exploit the electromagnetic radiation which arises when an electron is forced (by magnets) on a curved path. Elaborating this principle more and more, new generations of brilliant light sources have been designed such as the European X-ray free-electron laser (XFEL) which is currently being built at DESY in Hamburg. Further, accelerators are used in medicine for cancer therapy.

No matter which type of accelerator is regarded, all of them start with a particle source where the elementary particles are produced (e.g. cathode, photocathode, ion source), some magnetic focusing elements and sections in which the stream of particles is bunched and a first acceleration takes place. Thus, a bunch is a large number of charged elementary particles. It achieves its relativistic speed and its higher and higher energy passing through RF cavities.

As long as the particles are non-relativistic, their self-electric space charge field is influencing the particles in the bunch while space charge fields don't play a role anymore for the highly relativistic particles. The space charge effect is of crucial importance for the next generation accelerators with their ultrashort, very dense bunches of high power, such as in the XFEL, since this naturally implies higher space charge effects. If one wouldn't do a careful design study, one possible effect could be e.g. that the bunch, which indeed should stay in tight dimensions, extremely grows due to the space charge effect and hits the wall of the vacuum chamber. The specific bunch characteristics of future accelerators makes simulation studies of space charge effects more challenging than before.

The most prominent, classical methodology for numerical space charge studies is known as the Particle-in-Cell (PIC) model [1]. The considered bunch is embedded in the computational domain Ω , which usually is a cubic or a cylindrical domain. The computational domain Ω is discretized and the charge of the particles inside each cell is assigned to neighboring grid points by algorithms like the Nearest Grid Point (NGP) or the Cloud in Cell (CIC) schemes. Note, that so-called macro-particles are introduced in order to achieve a computational load which is still manageable. Then, the space charge has to be calculated, applied to the (macro-)particles and the equation of motion has to be solved. A usual procedure is to use the Lorentz transformation in each time step to transfer between the laboratory system and the rest frame (of special relativity) and then compute the space charge fields in the rest frame. The self-electric field can be derived by solving Poisson's equation (in the rest frame). It is transferred back to the laboratory system by the Lorentz transformation with the Lorentz factor γ .

In this contribution, we concentrate on the efficient solution of Poisson's equation:

$$\left(\frac{d^2\varphi(x, y, z)}{dx^2} + \frac{d^2\varphi(x, y, z)}{dy^2} + \frac{d^2\varphi(x, y, z)}{dz^2} \right) = -\frac{\rho(x, y, z)}{\varepsilon_0}, \quad (x, y, z) \in \Omega$$

where $\rho(x, y, z)$ is the charge density, ε_0 is the permittivity of vacuum and $\varphi(x, y, z)$ is the electrostatic potential, i.e. we study this problem in Cartesian coordinates in 3D. Free space boundary (or some say open boundary) conditions are regarded. Although this is not true in the real accelerator, this consideration is well-introduced and most common in the simulation of space charge effects as long as the bunch is far enough apart from the walls of the enclosing vacuum tube. The common way to solve this equation is to convolute the density of charged particles and the Green's function in free space, known as the Green's function method. However, in some cases such as a very long cigar-shape or short pancake-shape bunch the numerical calculation may suffer from errors. The so-called integrated Green's function (IGF) [2, 3] has especially been invented for such issues. It deals with an analytical integration rather than a numerical integration. However, the computation is rather involved and time-consuming and thus calls for an improvement to higher efficiency.

We present some appropriate methods, as accurate as the IGF method yet costing less CPU time for different practical problems. In general, the reduced integrated Green's function (RIGF) method is suitable for all problems applying the IGF method—for instance the near-bunch field calculation. In contrast, the cutting (integrated) Green's function (CIGF) method is only advantageous for far-bunch field calculation. A further new method, denoted as cutting reduced integrated Green's function (CRIGF) method can accelerate the calculations even more. This routine can also be used in other Poisson solver codes to improve efficiency.

2 GF, IGF and RIGF Integral for Poisson's Equation

The Green's function-type methods are often-used methods to solve Poisson's equation in free space, i.e. with open boundaries. The Green's function is given as:

$$G(x, x', y, y', z, z') = \frac{1}{\sqrt{(x-x')^2 + (y-y')^2 + (z-z')^2}}. \quad (1)$$

Using the Green's function, the solution of Poisson's equation in \mathbf{R}^3 , i.e. the continuous electrostatic potential φ , reads as [1, 2]:

$$\varphi(x, y, z) = \frac{1}{4\pi\varepsilon_0} \cdot \int \int \int \rho(x', y', z') G(x, x', y, y', z, z') dx' dy' dz'. \quad (2)$$

Now, regard a cubic computational domain Ω which is discretized by N_x, N_y and N_z steps, respectively, in each coordinate direction with equidistant step sizes h_x, h_y, h_z . Then, the discrete integral formula is given by

$$\varphi(x_i, y_j, z_k) \approx \frac{1}{4\pi\varepsilon_0} \cdot \sum_{i'=1}^{N_x} \sum_{j'=1}^{N_y} \sum_{k'=1}^{N_z} \rho(x_{i'}, y_{j'}, z_{k'}) \tilde{G}(x_i, x_{i'}, y_j, y_{j'}, z_k, z_{k'}), \quad (3)$$

where the grid points (x_i, y_j, z_k) are the center points of each integral. The integral cell is equal to the individual grid cells with side lengths h_x , h_y and h_z . Thus, the integral over one grid cell reads as:

$$\tilde{G}(x_i, x_i', y_j, y_j', z_k, z_k') = \int_{x_i-h_x/2}^{x_i+h_x/2} \int_{y_j-h_y/2}^{y_j+h_y/2} \int_{z_k-h_z/2}^{z_k+h_z/2} G(x_i, x', y_j, y', z_k, z') dx' dy' dz'. \quad (4)$$

In the following, for the different kinds of integrals, \tilde{G} will be specified by different subscripts. Further, regarding the calculation of the Green's function values we will apply a coordinate translation, substituting $w-w'$ by w for w in $\{x, y, z\}$ and thus use w instead of $w-w'$. If we apply the midpoint rule, the numerical integral is known as GF integral:

$$\tilde{G}_{GF}(x_i, y_j, z_k) = h_x h_y h_z G(x_i, y_j, z_k). \quad (5)$$

In many applications, the midpoint rule can readily be used. Yet, often higher accuracy is needed. This can be achieved by higher order numerical integration rules or by the IGF integral:

$$\begin{aligned} \tilde{G}_{IGF}(x_i, y_j, z_k) &= \int_{x_i-h_x/2}^{x_i+h_x/2} \int_{y_j-h_y/2}^{y_j+h_y/2} \int_{z_k-h_z/2}^{z_k+h_z/2} G(x', y', z') dx' dy' dz' \\ &= IGF(x_i + \frac{h_x}{2}, y_j + \frac{h_y}{2}, z_k + \frac{h_z}{2}) - IGF(x_i + \frac{h_x}{2}, y_j + \frac{h_y}{2}, z_k - \frac{h_z}{2}) \\ &\quad - IGF(x_i + \frac{h_x}{2}, y_j - \frac{h_y}{2}, z_k + \frac{h_z}{2}) - IGF(x_i - \frac{h_x}{2}, y_j + \frac{h_y}{2}, z_k + \frac{h_z}{2}) \\ &\quad + IGF(x_i - \frac{h_x}{2}, y_j - \frac{h_y}{2}, z_k + \frac{h_z}{2}) + IGF(x_i + \frac{h_x}{2}, y_j - \frac{h_y}{2}, z_k - \frac{h_z}{2}) \\ &\quad + IGF(x_i - \frac{h_x}{2}, y_j + \frac{h_y}{2}, z_k - \frac{h_z}{2}) - IGF(x_i - \frac{h_x}{2}, y_j - \frac{h_y}{2}, z_k - \frac{h_z}{2}), \end{aligned} \quad (6)$$

where the $IGF(x, y, z)$ function is the primitive function (antiderivative) of (1), which can be expressed as:

$$\begin{aligned} IGF(x, y, z) &\doteq \iiint \frac{1}{\sqrt{x^2 + y^2 + z^2}} dx dy dz = -\frac{z^2}{2} \arctan\left(\frac{xy}{z\sqrt{x^2 + y^2 + z^2}}\right) \\ &\quad - \frac{y^2}{2} \arctan\left(\frac{xz}{y\sqrt{x^2 + y^2 + z^2}}\right) - \frac{x^2}{2} \arctan\left(\frac{yz}{x\sqrt{x^2 + y^2 + z^2}}\right) + yz \ln(x \\ &\quad + \sqrt{x^2 + y^2 + z^2}) + xz \ln(y + \sqrt{x^2 + y^2 + z^2}) + xy \ln(z + \sqrt{x^2 + y^2 + z^2}). \end{aligned} \quad (7)$$

Here, we present the simple form from [3].

RIGF integral: In order to figure out the improvement of the IGF integral compared to the GF integral, we define the local Green’s function integral relative fraction as: $\eta_G(x_i, y_j, z_k) = |\delta\tilde{G}_{GF}(x_i, y_j, z_k)/\tilde{G}_{IGF}(x_i, y_j, z_k)|$, where $\delta\tilde{G}_k = \|\tilde{G}_{IGF_k} - \tilde{G}_{GF_k}\|$. To evaluate the variation of $\eta_G(x_i, y_j, z_k)$ visually and easily in the grid, we chose a computational domain with a large aspect ratio: $L_x : L_y : L_z = 1 : 1 : 30$, where L_x, L_y, L_z are the edge lengths of the cubical domain Ω . It is discretized by $32 \times 32 \times 32 = 32,768$ grid points (In calculation, Green’s function needs one more point on each axis, i.e. $33 \times 33 \times 33 = 35,937$ [2]). In Fig. 1 (left), we use a boxplot of $\eta_G(:, :, z_k)$. Each column corresponds to one slice of index k . We can observe that the local relative errors exponentially decrease with an increasing value of k (z_k). Only in the very first slices, the errors are large and strongly varying. For increasing k , the errors inside a slice and compared with the neighbor slices errors coincide more and more.

The motivation of the RIGF integral is relatively natural and simple. In the calculation of $\tilde{G}(x_i, y_j, z_k)$, the IGF integral $\tilde{G}_{IGF}(x_i, y_j, z_k)$ has higher complexity than the numerical GF integral $\tilde{G}_{GF}(x_i, y_j, z_k)$, i.e. for each $\tilde{G}_{IGF}(x_i, y_j, z_k)$ we have to calculate eight terms in (6) and every term should be calculated by (7). Yet, $\tilde{G}_{GF}(x_i, y_j, z_k)$ has just one simple term, which is also faster to be evaluated. Thus, we calculate the $\tilde{G}_{IGF}(x_i, y_j, z_k)$ by the exact integral only over those grid cells where it is necessary and everywhere else we replace it by the numerical integral $\tilde{G}_{GF}(x_i, y_j, z_k)$. Practically, this means that only the near-origin parts, where the bunch is located, are treated by the IGF model. The remaining parts of the integral are calculated by the simpler standard GF model. We determine integer parameters (R_x, R_y, R_z) indicating at which grid line to switch from the IGF model to the GF model (see Fig. 1 (right) blue line between Ω_{IGF} and Ω_{GF}). For the following, we suppose that the large

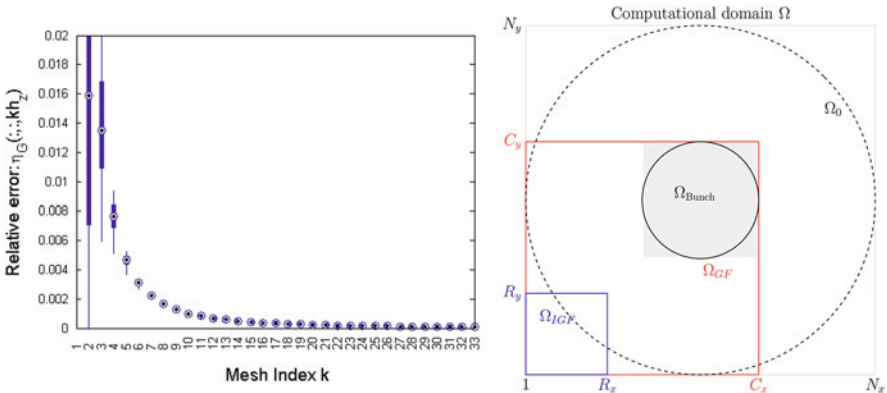


Fig. 1 (Left): The local relative error of the GF integral. (Right): A schematic plot of cutting Green’s function domain for Cartesian coordinates

aspect ratio direction is on the z -axis. Then the new integral reads as follows:

$$\tilde{G}_{RIGF}(x_i, y_j, z_k) = \begin{cases} \tilde{G}_{IGF}(x_i, y_j, z_k), & (1, 1, 1) \leq (i, j, k) \leq (R_x, R_y, R_z); \\ \tilde{G}_{GF}(x_i, y_j, z_k), & \text{otherwise;} \end{cases}$$

It has been investigated how these parameters (R_x, R_y, R_z) should be chosen. There are two general key aspects which should be balanced in the chosen strategy: the computational time and the achieved accuracy.

With respect to the computational time, it would be an option to choose $R_w = (N_w + 1)/s_w$ for w in $\{x, y, z\}$. The larger value of s , the less computational time is needed by the IGF calculation. Regarding the cigar-shape bunch as an example, it is reasonable to choose $R_w = N_w + 1$ for w in $\{x, y\}$. Then, the computational time depends linearly on s which ranges from 1 (IGF routine) to $N_z + 1$ (GF routine):

$$t_{RIGF} = \frac{N_z + 1 - R_z}{N_z + 1} t_{GF} + \frac{R_z}{N_z + 1} t_{IGF} = \frac{R_z}{N_z + 1} (t_{IGF} - t_{GF}) + t_{GF}. \quad (8)$$

On the other hand, with respect to the computational errors of the numerical integral which imply errors of the final result as well, a different strategy would be appropriate. Since the \tilde{G}_{IGF} is decreasing very fast with respect to the distance from the center of the bunch, the location where it starts to remain more or less stationary should be determined first. In practice, we use a reference function $f(N_z)$ to locate the stationary area. For example, we choose $1/\log_2 N_z$ as $f(N_z)$ to locate k by $\|\tilde{G}_{IGFk-1} - \tilde{G}_{IGFk}\|/\tilde{G}_{IGFk-1} < f(N_z)$. Secondly, we determine the accuracy tolerance: Choose the proper R_z given by the first k for which the magnitude of $\delta\tilde{G}_k/\delta\tilde{G}_k$ stable drops down to 10^{-s} , $s \geq 0$, where $\delta\tilde{G}_k = \|\tilde{G}_{IGFk} - \tilde{G}_{GFk}\|$. Note, s is the accuracy control integer for the RIGF. Of course, these parameters have to be determined individually for different problems under study.

3 CRIGF Method for Poisson's Equation

In many applications, the computational domain will be considerably larger than the domain occupied by the charged bunch. As shown in Fig. 1 (right), the bunch domain Ω_{Bunch} (the shadowed domain) lies in the center of the computational domain Ω . In this case, of course, all terms with zero charge density ρ (factor of the tilde Green's function) can be omitted in the summation of (3). Based on the convolution theory, the irrelevance of these terms should be still true if we take a Fourier transform and use it in the fast Poisson solver. Therefore, the CIGF [4] integral is recommended for high efficiency:

$$\tilde{G}_{CIGF}(x_i, y_j, z_k) = \begin{cases} \tilde{G}_{IGF}(x_i, y_j, z_k), & (1, 1, 1) \leq (i, j, k) \leq (C_x, C_y, C_z); \\ 0, & \text{otherwise;} \end{cases}$$

where (C_x, C_y, C_z) is determined by the domain-bunch ratio $\alpha_w = L_w \text{Bunch} / L_w \text{Domain}$, L_w is the length for w in $\{x, y, z\}$ and $C_w = [(2 + \alpha_w) / 2\alpha_w]$. The CIGF is as accurate as the IGF. For far-bunch domain space charge simulation, the CIGF integral is efficient and does not waste calculations. When the near-bunch domain simulation takes place, the CIGF is not valid anymore. However, the RIGF can always be applied.

In total, we have the following CRIGF integral: The combination of RIGF and CIGF as the CRIGF should be more efficient than the pure CIGF for the same problem,

$$\tilde{G}_{CRIGF}(x_i, y_j, z_k) = \begin{cases} \tilde{G}_{IGF}(x_i, y_j, z_k), & (1, 1, 1) \leq (i, j, k) \leq (R_x, R_y, R_z); \\ \tilde{G}_{GF}(x_i, y_j, z_k), & (R_x, R_y, R_z) \leq (i, j, k) \leq (C_x, C_y, C_z); \\ 0, & \text{otherwise;} \end{cases}$$

where (C_x, C_y, C_z) and (R_x, R_y, R_z) are chosen as above.

In order to make the calculation of (3) more efficient, we should implement it as a cyclic convolution. The charge density ρ_{ex} is obtained by padding ρ with zeros in all expansion grid points, the tilde Green's function \tilde{G} is expanded symmetrically as \tilde{G}_{ex} . Using 3D discrete Fourier transformation \mathfrak{F} and convolution theory, the expanded potential expression is given by:

$$[\varphi_{ex}]_{i,j,k} = \frac{1}{4\pi\epsilon_0} \mathfrak{F}^{-1} \{ \{ \mathfrak{F} \tilde{G}_{ex} \}_{i,j,k} \cdot \{ \mathfrak{F} \rho_{ex} \}_{2N_x, 2N_y, 2N_z} \}. \quad (9)$$

The routine of (9) can be further improved with respect to both- less storage requirement and less time consumption [1]. We use a similar procedure. Then, the storage requirement of the convolution method is $2N_x \times N_y \times N_z$ plus two temporary 2D arrays sized $2N_y \times N_z$ and $2N_x \times 2N_z$. In fact, our algorithm uses a pruned Fourier transform, whose purpose is to save time while avoiding the wasteful transforms of zeros in each direction.

4 Discussions and Examples

We regard a uniformly charged ellipsoid to achieve an analytical validation. For the longitudinal-to-transverse ratio, we choose 30. The domain-bunch ratio is 2. The relative errors are defined as follows:

$$\eta_\varphi(i, j, k) := \frac{|\varphi_{i,j,k} - \varphi_{true_{i,j,k}}|}{\max_{i,j,k} |\varphi_{true_{i,j,k}}|}, \text{ and } \hat{\eta}_\varphi := \max_{i,j,k} (\eta_\varphi(i, j, k)).$$

Here, the notations are, $\eta_\varphi(i, j, k)$, $\hat{\eta}_\varphi$, $\varphi_{i,j,k}$ and $\varphi_{true_{i,j,k}}$ as the relative error of the potential at index (i, j, k) , the global relative error of the potential, the computed

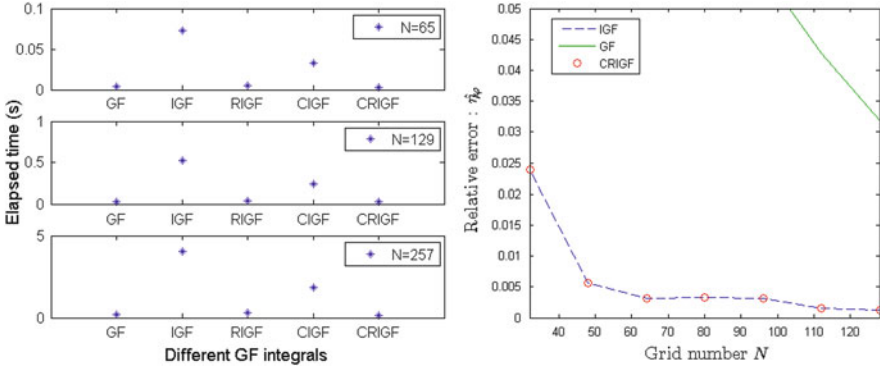


Fig. 2 (Left): Comparison of different Green’s function integrals’ elapsed time with increasing grid resolution. (Right): Convergence study of CRIGF, IGF and GF method

potential at index (i, j, k) and the true potential for the same index, respectively. The algorithm is implemented in C language on an Intel 2.6 GHz CPU.

Firstly, we compare the computation time of different GF integrals with increasing grid resolution as shown in Fig. 2 (left). Here $N = N_w, R_w = 8$ with w in $\{x, y, z\}$. RIGF is nearly 10 times faster than IGF, while CRIGF is more than 20 times faster for a domain-bunch ratio of 2. In Fig. 2 (right), we study the convergence of CRIGF with $s = 2$ for the accuracy control comparing to IGF and GF. As we can see, the CRIGF method agrees very well with the IGF method.

The whole implementation is carried out by using the FFTW [5] package. For the serial algorithm, the speed-up is around 15–25% including the calculation of the discrete convolution. This needs to be further improved, since the convolution’s calculation still takes most of the computational time. The efficiency results will be updated in our future studies, either by implementing a parallel algorithm or by applying a different discrete convolution routine.

5 Conclusion

In this paper, we introduced a 3D RIGF Poisson solver together with a routine called CRIGF method for beam dynamics simulations. We tested the new method with a model problem. On the practical side, RIGF is less time consuming, while it achieves almost the same accuracy as IGF for the electric potential. So we suggest to use the CRIGF routine rather than the IGF in order to speed up calculations.

References

1. Hockney, R.W., Eastwood, J.W.: *Computer Simulation Using Particles*. Institute of Physics Publishing, Bristol (1992)
2. Qiang, J., Lidia, S., Ryne, R.D., Limborg-Deprey, C.: Three-dimensional quasistatic model for high brightness beam dynamics simulation. *Phys. Rev. ST Accel. Beams* **9**:044204 (2006)
3. Qiang, J., Lidia, S., Ryne, R.D., Limborg-Deprey, C.: Erratum: three-dimensional quasistatic model for high brightness beam dynamics simulation. *Phys. Rev. ST Accel. Beams* **10**:129901 (2007)
4. Zheng, D., Markovič, A., Pöplau, G., van Rienen, U.: Study of a fast convolution method for solving the space charge fields of charged particle bunches. *Proc. IPAC* 418–420 (2014)
5. Frigo, M., Johnson, S.G.: The design and implementation of FFTW3. *Proc. IEEE* **93**:216–231 (2005)