ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV048

# FAST MULTIPOLE METHOD (FMM)-BASED PARTICLE ACCELERATOR SIMULATIONS IN THE CONTEXT OF TUNE DEPRESSION STUDIES *

M. Harper Langston†[1], Richard Lethin[1], Pierre-David Letourneau[1], Julia Wei[1]
[1]Reservoir Labs Inc., New York, NY 10012, USA

*Abstract*

As part of the MACH-B (Multipole Accelerator Codes for Hadron Beams) project, Reservoir Labs has developed a Fast Multipole Method (FMM [1–7])-based tool for higher fidelity modeling of particle accelerators for high-energy physics within Fermilab's Synergia [8, 9] simulation package. We present results from our implementations with a focus on studying the difference between tune depression estimates obtained using PIC codes for computing the particle interactions versus those obtained using FMM-based algorithms integrated within Synergia. In simulating the self-interactions and macroparticle actions necessary for accurate simulations, we present a newly-developed kernel inside of a kernel-independent FMM, where near-field kernels are modified to incorporate smoothing while still maintaining consistency at the boundary of the far-field regime. Each simulation relies on Synergia with one major difference: the way in which particles interactions are computed. Specifically, following our integration of the FMM into Synergia, changes between PIC-based computations and FMM-based computations are made by simply selecting the desired method for near-field (and self) particle interactions.

## INTRODUCTION

The majority of numerical approaches for accelerator multiparticle-tracking solve the macroscale problem by employing Particle-In-Cell (PIC) methods [8–14]. These methods incorporate an Eulerian method for solving the necessary equations and Lagrangian techniques to advect particles through the domain (e.g., see Fig. 1).

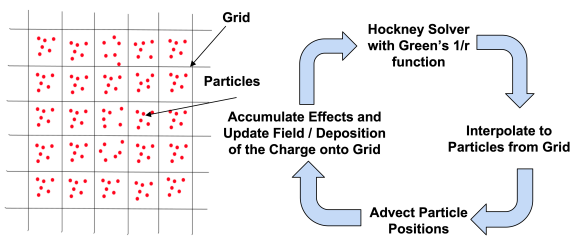


Figure 1: PIC-based Hockney solver. Given a cloud of charged particles, iterate (1) Grid charge deposition; (2) Compute potential; (3) Compute forces at grid points; (4) Compute forces at particle locations.

Since space-charge modeling in high-intensity hadron beams for the accelerator physics community requires scalable and high-fidelity algorithmic approaches, all new computational approaches must **(1)** be inherently multiscale, **(2)** exploit locality, **(3)** reduce expense of non-locality while handling accuracy, **(4)** guarantee high accuracy when needed, and **(5)** handle a variety of complex geometries.

Reservoir Labs' MACH-B (Multipole Accelerator Codes for Hadron Beams) project addresses the above five key elements, maintaining the strengths of PIC codes and approaches while further improving upon some of their weaknesses, allowing domain experts to evaluate and optimize various scenarios for complex high-energy physics experiments. The MACH-B technology is based on both existing and novel mathematical frameworks, providing scalable, high-performance algorithms that will assist in accurately and rapidly computing a variety of complex particle accelerator simulations; specifically, (1) **Fast Multipole Methods (FMM)** and (2) **Boundary Integral Solvers (BIS)**.

### *Introduction to Fast Multipole Methods*

FMM approaches achieve linear scaling by separating near- and far-field interactions (e.g., see Fig. 2) on a spatial hierarchy using tree data structures. As they achieve arbitrary precision at modest cost with straightforward error estimates [1,2,4–6,15–20], FMM techniques are well-suited for problems requiring high accuracy at large scales, such as in particle accelerator simulations.

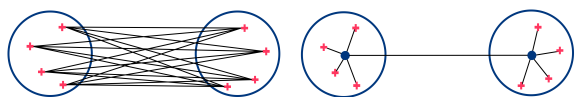

Figure 2: (*Left*): A naive $O(N^2)$ approach for computing the interactions between well-separated sources and targets. (*Right*): Using multipole and local expansions to reduce far-field costs, based on refinement.

FMMs are **inherently multiscale**, separating a regular domain into disjoint sets, using a tree structure to **exploit locality** as well as **reduce the expense of non-locality** through low-rank approximation multipole expansions [1, 4]. FMMs compute the total field at a domain $B$ as the sum of (a) the field due to the sources contained in its near field $\mathcal{N}^B$ and (b) its far field $\mathcal{F}^B$. Contributions from $\mathcal{N}^B$ are computed using direct, dense summations, while contributions from $\mathcal{F}^B$ are obtained by evaluating approximating expansion coefficients, constructed to achieve far-field low-rank approximations at computationally-efficient and provably-accurate levels of accuracy. Through two parameters ((1) for points per smallest grid in the hierarchy and (2) for number of coefficients in the expansions), **high accuracy is guaranteed** [2, 3, 21].

* This work was supported by the U.S. Department of Energy as part of the SBIR Phase I Project DE-SC0020934.
† langston@reservoir.com

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV048



The FMM uses *upward* and *downward passes* on a hierarchical tree structure, employing multiple operators for converting expansion (multipole and local) coefficients to **achieve optimal $O(n)$ complexity** (See Fig. 3 from [4]).

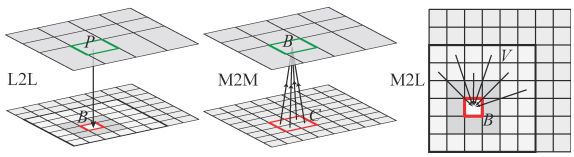


Figure 3: Local to Local (L2L), Multipole to Multipole (M2M) and Multipole to Local (M2L) operators translate coefficients efficiently throughout the FMM algorithm.

## INTEGRATING FMM INTO SYNERGIA

In this work, we focus on MACH-B's effort in integrating the FMM algorithm [1, 20] into Synergia [8, 9] for computing particle interactions as compared to existing PIC-based methods. We rely on the STKFMM from [15], itself a generalization of the highly-efficient PVFMM [6] implementation of the kernel-independent FMM [3–5, 19]. We chose this particular implementation of the FMM for two main reasons:

1. *Kernel-Independence*: rather than applying only to the Laplace kernel $(1/r)$, this version is kernel independent (in particular, it can handle any elliptic kernel).
2. *Computational Performance*: both the theory and implementation of this version are among the most performant currently existing.

Prior to the introduction of the FMM, Synergia offered two ways of computing interactions, both based on the Particle-In-Cell (PIC) method: `space_charge_2d_open_hockney()` and `space_charge_3d_open_hockney()`. All rely on Hockney's method (as seen in Fig. 1) and can be summarized as follows: given a cloud of charged particles,

1. *Grid Charge Deposition*: for each cloud particle, "deposit"/interpolate its charge on a regular uniform grid.
2. *Compute Potential*: compute the potential generated by charges from a grid point at all other grid points using the FFT (convolution with electrostatic kernel $1/r$).
3. *Compute Forces at Grid Points*: compute 3D forces at each grid point using finite differences.
4. *Compute Forces at Particles*: interpolate force values from the uniform grid to the particle locations.

The FMM was integrated into Synergia such that changes from native PIC-based routines to FMM-routines are performed with minimal effort; currently, switching between methods requires merely changing one word in the name of the aforementioned functions to: `space_charge_2d_open_fmm()` and `space_charge_3d_open_fmm()`, respectively.

### *Verifying Correctness of FMM in Synergia*

To verify correctness and performance of each implementation, we perform to three different sets of experiments:

- *Test 1*: Compare Synergia's 3D Hockney uniform grid routine potential computations with a naive approach.
- *Test 2*: Compare 3D FMM routine potential computations with a naive approach (same grid as Test 1).
- *Test 3*: Compare 3D FMM, 3D Hockney (Synergia) and naive force computations on arbitrary point clouds.

The first two tests verify the correctness of each implementation, positioning random charges at uniform grid points. The potential at each grid point is computed using a naive $(O(N^2))$ method and constitutes the ground truth against which we compare solutions. We limit ourselves to 32 points per dimension since computations through the naive method become to onerous beyond this point. Because particles are located at grid points, no interpolation is necessary for the PIC codes; hence, expected numerical accuracy for PIC-based methods matches that of the FFT itself(i.e., $\epsilon \approx 10^{-15}$ in double precision) as seen in Table 1.

Table 1: Test 1: Compare Synergia's 3D Hockney routine potential computations with naive approach on uniform grid. The PIC-based routine behaves as expected.

| $G_{pts}$ per dim | Total $G_{pts}$ | Rel. Err. |
|---|---|---|
| 8 | 512 | $4.56 \cdot 10^{-16}$ |
| 16 | 4096 | $1.36 \cdot 10^{-15}$ |
| 32 | 32768 | $3.78 \cdot 10^{-15}$ |

Test 2 is analogous to Test 1, except we employ the FMM for computing particle interactions rather than Synergia's PIC-based methods. The FMM is oblivious to charges abeing located at gridpoints. The FMM parameter, $p$, dictates its accuracy and computational costs; single precision can be expected with $p \approx 8$ and double precision with $p \approx 12$ while the latter takes about twice as long as the former. Table 2, shows results for different values of $p$, where it is observed that the FMM behaves as expected.

Table 2: Test 2: Compare 3D STKFMM routine potential computations with naive approach on uniform grid (same grid as Test 1). The FMM-based routine behaves as expected with higher accuracy as $p$ is increased.

| $G_{pts}$ per dim | Total $G_{pts}$ | Rel. Err. (p=8) | Rel. Err. (p=12) |
|---|---|---|---|
| 8 | 512 | $4.95 \cdot 10^{-9}$ | $1.91 \cdot 10^{-11}$ |
| 16 | 4096 | $2.95 \cdot 10^{-9}$ | $9.72 \cdot 10^{-12}$ |
| 32 | 32768 | $2.97 \cdot 10^{-9}$ | $5.16 \cdot 10^{-12}$ |

Test 3 compares the accuracy of Synergia's PIC-based methods and the FMM in a general case: a cloud of charged particles. For this purpose, we generate a 3D ensemble of

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV048

charges (positive/protons) containing up to 32768 particles and then compare to the naive solutions in Table 3. The FMM ($p = 8$) achieves full (single) precision while the PIC-based method suffers from significantly lower accuracy due to interpolations and finite difference steps (Step 1, Step 3 and Step 4 from Fig. 1) associated with PIC codes. When particles, are not at grid points (as in Test 1) and forces must be computed, the error resulting from these steps becomes dominant.

Table 3: Test 3: Compare 3D STKFMM, 3D Hockney (Synergia) and Naive force computations on arbitrary point clouds. The FMM preserves its accuracy regardless of the particle distribution, whereas the PIC-based methods suffers from significant numerical errors associated with interpolation and finite differences.

| $N_{particles}$ | Param. ($M/p$) | Rel. Err. (PIC/FMM) |
|---|---|---|
| 4096 | 32/8 | $0.228 / 1.28 \cdot 10^{-7}$ |
| 16384 | 32/8 | $0.166 / 2.07 \cdot 10^{-7}$ |
| 32768 | 32/8 | $0.141 / 2.66 \cdot 10^{-7}$ |
| 4096 | 64/8 | $0.180 / 1.28 \cdot 10^{-7}$ |
| 16384 | 64/8 | $0.141 / 2.07 \cdot 10^{-7}$ |
| 32768 | 64/8 | $0.123 / 2.66 \cdot 10^{-7}$ |
| 4096 | 128/8 | $0.116 / 1.28 \cdot 10^{-7}$ |
| 16384 | 128/8 | $0.106 / 2.07 \cdot 10^{-7}$ |
| 32768 | 128/8 | $0.0992 / 2.66 \cdot 10^{-7}$ |
| 4096 | 256/8 | $0.0587 / 1.28 \cdot 10^{-7}$ |
| 16384 | 256/8 | $0.0640 / 2.07 \cdot 10^{-7}$ |
| 32768 | 256/8 | $0.0656 / 2.66 \cdot 10^{-7}$ |

To these results, we note the following observations:

1. In the most general case (e.g., arbitrary particle clouds), the FMM offers greater accuracy and speed compared with PIC-based methods.

2. In special cases, including the case where the underlying solution is smooth (e.g., as the number of particles goes to infinity., i.e., Vlasov), PIC methods may exhibit higher accuracy (see tune depression case below), although such conditions may be difficult to verify.

## TUNE DEPRESSION SIMULATIONS

The $(x/y)$ *tune* of a particle is defined as the number of $(x/y)$ transverse oscillations a particle experiences per revolution around a closed-loop accelerator. Tune is heavily-dependent upon the nature of the accelerator lattice, but also depends strongly on the interactions between the particles (self-energy). For instance, researchers at Fermi Lab have shown an interesting relationship between the transverse $(x/y)$ offset of a particle's original position within a Gaussian bunch and its tune. Figure 4 shows the computation of the tune for particles with initial offset between $0\,\sigma$ and $5\,\sigma$ (measured in number of bunch transverse standard deviations: $\sigma$) In particular, this figure shows the importance of self-interactions. Indeed, no changes in the tune are observed

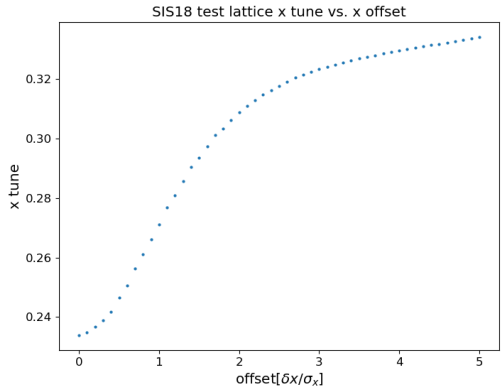


Figure 4: Tune depression (Hz; x-transverse oscillations per turn) vs initial x offset. Results obtained using Synergia PIC routine for computing particle interactions.

when the interactions are neglected, while an interesting relationship is present when they are. This demonstrates the importance of particle-particle interactions in this context and, more generally, for accelerator simulations.

In the context of MACH-B, we study the difference between tune depression estimates obtained in Synergia from PIC versus FMM, adopting the following methodology: first, create a Gaussian bunch of size approximately $10^{-5} \times 10^{-5} \times 10^{4}$ *mm* centered at the origin and containing approximately $10^{10}$ physical particles (protons) and between $10^{4}$ and $10^{7}$ macroparticles depending on the situation (see below for more information about macroparticles). Each macroparticle is randomly assigned a position and momentum following a Gaussian distribution with mean 0 and covariance matrix specific to the lattice, chosen so that the overall shape of the bunch remains constant at each iteration. We introduce "spectator particles" with specific initial offsets to monitor the tune as a function of offset.

After initializing the particle bunch, we set the particles in motion and record the transverse position of each spectator particles after each turn (50 to 500 with $10^{1}$ to $10^{2}$ time steps per simulation). We extract the tune as a function of offset by taking the Fourier transform of the $x/y$ transverse position as function of turn. The estimate for the tune is taken to be the dominant frequency identified using the resulting spectrum. Finally, we plot the relationship between the estimated tune and the original offset.

We reproduce the results obtained using PIC codes by using an FMM in its stead. Important differences are observed, however, and this leads to multiple key observations:

1. FMMs can reproduce observed results from PIC codes in the context of tune depression computations.

   (a) FMMs faithfully capture the physical interactions between particles up to numerical accuracy, while PIC methods introduce a large degree of regularization(often desirable).

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV048



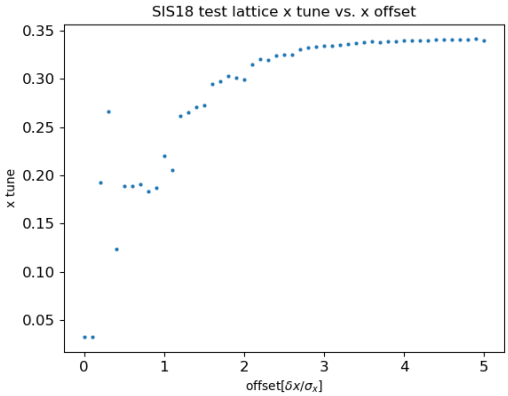


Figure 5: Tune depression (Hz; x-transverse oscillations per turn) vs initial x offset. Results obtained using FMM integrated within Synergia for computing particle interactions. PIC routines introduce more regularization than the FMM resulting in smoother (less noisy) estimates.

2. It is computationally necessary to use macroparticles to represent a particle bunch as the number of particles needed to capture the full dynamics ($> 10^{12}$).

   (a) The use of macroparticles requires the user to use a modified Laplace kernel for representing interactions with the FMM.

   (b) The number of macroparticles and type of modified kernel affects results; i.e., regularization of the singularity at the origin has the largest effects.

These observations are important for understanding the advantages and disadvantages of using FMMs for particle accelerator simulations as detailed below.

### *FMM vs. Tune Depression Simulations*

As previously mentioned and shown in Fig. 5, we successfully reproduced the relationship between tune depression and initial particle offset within a Gaussian bunch using Synergia with an FMMs replacing PIC. However, results obtained using the FMM tend to be more noisy than those obtained using Synergia's native PIC methods.

This phenomenon is mostly due to the inherent regularization associated with PIC methods; the key for computational efficiency here is the use of the FFT, which leads to an $O(N\log(N))$ scaling ($N$ being the number of grid points) rather than a prohibitive $O(N^2)$ scaling in the case of a naive computation. The use of the FFT, although efficient, comes at a cost: the use of a uniform (DFT) grid. The need to go between charges in an arbitrary configuration to (proxy) charges on a regular grid entails the use of interpolation (deposition) further introducing artificial regularization. Indeed, after deposition, two particles that were originally close to each other, and that would thus have exerted large forces upon each other, now find themselves much farther apart and thus exerting much weaker forces.

By contrast, the FMM can compute all interactions between particles with high numerical accuracy without the need for deposition, interpolation, or a regular grid or finite differences, thus avoiding regularization. In addition to preserving high accuracy, FMM costs remains low (i.e., $O(M)$ for $M$ particles). It turns out, however, that this absence of regularization may not always be beneficial.

Indeed, in this specific case (Gaussian bunch), regularization proved beneficial as seen from the qualitative differences between the smoother curve in Fig. 5 and noisier one in Fig. 5. The main reason behind this is that the underlying true (asymptotic as the number of particles goes to infinity) solution for this particular problem is itself smooth. In this context, regularization can capture the underlying physics (one does not lose information by smoothing/regularizing since the solution is already smooth). By contrast, the FMM suffers from strong fluctuations due to small local charge density inhomogeneities that are negligibly small in the asymptotic solution (e.g., Debye screening).

While PIC-based regularization may be advantageous in certain contexts, it is, however, not always appropriate since its success ultimately relies on strong un-verifiable hypotheses regarding the underlying physic. Instead, the FMM offers a fast alternate approach to computing particle interactions without the need for such strong assumptions. While a potential introduction of noise in the estimated solution is possible, this noise can be controlled through the use of macroparticles and modified interaction kernels.

When the charge density is not homogeneous, or when the interactions between multiple bunches are considered, PIC methods are further disadvantageous: the rectangular domain covered by the regular grid needed must contain all particles present in a PIC simulation. However, for an inhomogeneous density, this results in potentially more grid points than particles (many having zero charge), and therefore an FFT cost (proportional to number of gridpoints) much higher than that of the FMM (proportional to the number of particles).

### *Macroparticles and Incorporation into FMMs*

Despite significant advantages of FMMs over PIC methods, it necessary to use macroparticles for large simulations purposes. *Macroparticle* is an all-encompassing term for a family of heuristics, by which the number of particles in a simulation is significantly reduced compared with the number of physical particles.[1] In Synergia, macroparticles are created through a smaller number of particles with larger charges. For instance, to perform a PIC-based simulations in Synergia using $10^{12}$ physical particles of charge $e$, but only $10^7$ macroparticles, one would simply create a simulation with $10^7$ particles having charge $\frac{10^{12}}{10^7}\, e = 10^5\, e$.

Although often appropriate, such simple macroparticles may not always capture physical phenomena appropriately. In this sense, a more suitable way of creating macroparticles

[1] Given their fewer numbers, the interactions between marcoparticles *must* be modified if the solution is to represent a larger number of particles.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV048

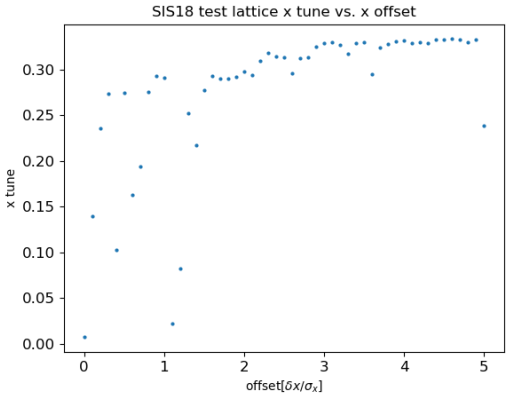


Figure 6: Tune depression ($x$) vs original $x$ offset for modified interaction kernel ($1/(r+\delta)$) using simulations with $6.4 \cdot 10^6$ physical particles and $6.4 \cdot 10^4$ macroparticles. In Fig. 5, $\delta = 10^{-2}$ was set while here, we set $\delta = 10^{-5}$. Note how less regularization leads to a much noisier outcome.

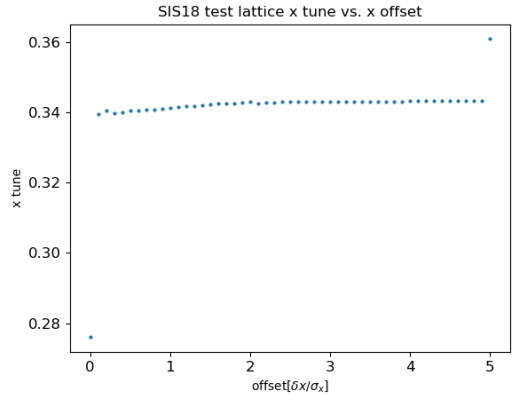


Figure 7: Tune depression ($x$) vs original $x$ offset for a varying number of physical and macroparticles. In Fig. 6, $N_{physical} = 6.4 \cdot 10^6$, $N_{macro} = 6.4 \cdot 10^6$, whereas here $N_{physical} = N_{macro} = 6.4 \cdot 10^4$. Note how using too few physical particles (or inappropriate macroparticles) fails to capture the appropriate behavior.

is through the use of a *modified interaction kernel*. Modifying the interaction kernel (electrostatic potential $1/r$) by reducing the strength of the singularity at the origin, can be used to introduce regularization as well as to emulate the behavior of the particles in the asymptotic limit (e.g., local averaging, Debye screening, etc.). In MACH-B, we modify this kernel by adding a small positive quantity to the radius, so that the interaction kernel now takes the form: $1/(r+\delta)$, for some $\delta > 0$, mollifying the strong forces at the origin (responsible for noise) while behaving like a regular electrostatic potential far from the origin.

The parameter $\delta$ can have a significant impact on the qualitative behavior of the solution for computing the tune depression in the previous section as shown in Fig. 6 as compared to Fig. 5. These figures show computations of the tune depression using the same bunch but with two different interaction kernel $\delta$ values. Figure 6 effectively approaches the unregularized case, whereas Fig. 5 represents macroparticles that interact like point particles when far away, but produce much smaller forces when close by. In particular, note the how much smoother the results are when using even a minimal ($\delta = 10^{-2}$) amount of regularization.

Macroparticles are not merely for computational convenience; their appropriate modeling is *necessary* for capturing the physics. For instance, for the tune depression simulations described in the previous section, one could simply try to reduce the number of physical particles rather introducing macroparticles. This, however, fails to generate the desired outcome as shown in Fig. 7 because a bunch with significantly fewer particles will behave significantly differently than a dense bunch. A dense bunch, however, cannot be simulated directly because too many physical particles are required.

## CONCLUSIONS AND FUTURE WORK

In MACH-B, we have sucesfully integrated FMMs into the Synergia particle accelerator simulation software suite and have shown the correctness of our implementation and have provided information favoring the use of FMM in many cases, especially when higher accuracy is needed or when the underlying solution may exhibit strong variations.

In our tune depression studies, the advantages of the FMM in this case are salient. Indeed, our FMM algorithm is constructed in such a way that it can handle a much larger family of macro-particle interaction kernels than can PIC codes, allowing the user to introduce various kinds of regularization when appropriate, or to simply do away with any sort of regularization without loss of accuracy. By constrast, PIC codes *always* introduce a significant amount of regularization which nature is quite rigid and difficult to modify. The FMM also offers better scalings than PIC codes in many cases (e.g., multiple bunches).

As we continue to modify our kernels and incorporate them into additional PIC-related codes, we anticipate the FMM libraries will become powerful tools for domain scientists and researchers for study and cross-verification, largely due to their ease of use and modification.

## ACKNOWLEDGMENT

Thank you to Drs. Eric Stern and Qiming Lu at Fermilab for helpful conversations and guiding us through Synergia as well as Drs. Robert Ryne and Jean-Luc Vay at LBNL for many very fruitful discussions.

## REFERENCES

[1] L. Greengard and V. Rokhlin, "A Fast Algorithm for Particle Simulations", *Journal of Computational Physics*, vol. 73,

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV048



no. 2, pp. 325–348, 1987. doi:10.1016/0021-9991(87)90140-9

[2] L. Greengard, "Fast Algorithms for Classical Physics", *Science*, vol. 265, no. 5174, pp. 909–914, 1994. doi:10.1126/science.265.5174.909

[3] L. Ying, G. Biros and D. Zorin, "A Kernel-Independent Adaptive Fast Multipole Algorithm in Two and Three Dimensions", *Journal of Computational Physics*, vol. 196, no. 2, pp. 591–626, 2004. doi:10.1016/j.jcp.2003.11.021

[4] M.H. Langston, L. Greengard, D. Zorin, "A Free-Space Adaptive FMM-Based PDE Solver in Three Dimensions", *Communications in Applied Mathematics and Computational Science*, vol. 6, no. 1, pp. 79–122, 2011. doi:10.2140/camcos.2011.6.79

[5] I. Lashuk *et al.*, "A Massively Parallel Adaptive Fast-Multipole Method on Heterogeneous Architectures", in *Proceedings of ACM/IEEE SC'09*, 2009. doi:10.1145/1654059.1654118

[6] D. Malhotra and G. Biros, "PVFMM: A Parallel Kernel Independent FMM for Particle and Volume Potentials", *Communications in Computational Physics*, vol. 18, no. 3, pp. 808–830, 2015. doi:10.4208/cicp.020215.150515sw

[7] W. Yan and R. Blackwell, "Kernel Aggregated Fast Multipole Method: Efficient Summation of Laplace and Stokes Kernel Functions", 2020. arXiv:2010.15155

[8] J. Amundson, P. Spentzouris, J. Qiang and R. Ryne, "Synergia: An Accelerator Modeling Tool with 3-D Space Charge", *Journal of Computational Physics*, vol. 211, no. 1, pp. 229–248, 2006.

[9] P. Spentzouris and J. Amundson, "Simulation of the Fermilab Booster using Synergia", *Journal of Physics: Conference Series*, vol. 16, no. 1, p. 215, 2015. doi:10.1088/1742-6596/16/1/029

[10] A. Friedman, D.P. Grote, and H. Irving, "Three-Dimensional Particle Simulation of Heavy-Ion Fusion Beams", *Phys. Fluids B*, vol. 4, pp. 2203–2210, 1992. doi:10.1063/1.860024

[11] R. Hockney and J. Eastwood, *Computer Simulation Using Particles*, CRC Press, 1988.

[12] J. Qiang, R.D. Ryne, Y.S. Habib and V. Decykz, "An Object-Oriented Parallel Particle-In-Cell Code for Beam Dynamics Simulation in Linear Accelerators", *Journal of Computational Physics*, vol. 163, no. 2, pp. 434–451, 2000. doi:10.1006/jcph.2000.6570

[13] P. Gibbon and G. Sutmann, "Long-Range Interactions in Many-Particle Simulation, Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms", in *Lecture Notes, J. Grotendorst, D. Marx, A. Muramatsu (Eds.), John von Neumann Institute for Computing, Jülich, NIC Series*, pp. 467–506, 2002.

[14] J. Qiang, "Fast 3D Poisson Solvers in Elliptical Conducting Pipe for Space-Charge Simulation", *Phys. Rev. Accel. Beams*, vol. 22, no. 10, p. 104601, 2019. doi:10.1103/PhysRevAccelBeams.22.104601

[15] W. Yan and M. Shelley, "Flexibly Imposing Periodicity in Kernel Independent FMM: A Multipole-to-Local Operator Approach", *Journal of Computational Physics*, vol. 355, no. 15, pp. 214–232, 2018. doi:10.1016/j.jcp.2017.11.012

[16] L. Ying, "An Efficient and High-Order Accurate Boundary Integral Solver for the Stokes Equations in Three Dimensional Complex Geometries", Ph.D. thesis, Courant Institute of Mathematical Sciences, New York University, 2004.

[17] A-K Tornberg and L. Greengard, "A Fast Multipole Method for the Three-Dimensional Stokes Equations", *Journal of Computational Physics*, vol. 227, no. 3, pp. 1613–1619, 2008. doi:10.1016/j.jcp.2007.06.029

[18] H. Zhang and M. Berz, "The Fast Multipole Method in the Differential Algebra Framework", *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment (The Eighth International Conference on Charged Particle Optics)*, vol. 645, no. 1, pp. 338–344, 2011.

[19] L. Ying, G. Biros, D. Zorin and M.H. Langston, "A New Parallel Kernel-Independent Fast Multipole Method", in *Proceedings of ACM/IEEE SC'03*, 2003. doi:10.1145/1048935.1050165

[20] M.H. Langston, M.M. Baskaran, B. Meister, N. Vasilache and R. Lethin, "Re-Introduction of Communication-Avoiding FMM-Accelerated FFTs with GPU Acceleration", in *IEEE Conference on High Performance Extreme Computing (HPEC'13)*, 2013. doi:10.1109/HPEC.2013.6670352

[21] R. Kress: *Linear Integral Equations*, Springer, 1989, p. 82.