A numerical method for solving Maxwell's equations in free-space using an approximate IVP Green's function

Victor Minden[1] Boris Lo[2] Phil Colella[3,4]

[1]Institute for Computational and Mathematical Engineering
Stanford University

[2]Applied Science & Technology
University of California, Berkeley

[3]Electrical Engineering & Computer Science Department
University of California, Berkeley

[4]Applied Numerical Algorithms Group
Lawrence Berkeley National Laboratory

September 5, 2014

Introduction

We are interested in numerically solving Maxwell's equations in a vacuum, which we write as

$$\begin{aligned} \frac{\partial \mathbf{E}}{\partial t} &= c\nabla \times \mathbf{B} - \mathbf{J} \\ \frac{\partial \mathbf{B}}{\partial t} &= -c\nabla \times \mathbf{E} \\ \nabla \cdot \mathbf{E} &= \rho \\ \nabla \cdot \mathbf{B} &= 0. \end{aligned}$$

Our motivating application is the field solve in particle-in-cell (PIC) methods for simulating, *e.g.*, plasma accelerators.

Assuming that divergence conditions are satisfied initially and that the continuity equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \mathbf{J} = 0$$

holds, the last two equations will be implicitly satisfied for all t, and we therefore only need to worry about solving

$$\frac{\partial}{\partial t}\begin{pmatrix} \mathbf{E} \\ \mathbf{B} \end{pmatrix} = \begin{pmatrix} c\nabla \times \mathbf{B} \\ -c\nabla \times \mathbf{E} \end{pmatrix} - \begin{pmatrix} \mathbf{J} \\ 0 \end{pmatrix} \equiv L\begin{pmatrix} \mathbf{E} \\ \mathbf{B} \end{pmatrix} - \begin{pmatrix} \mathbf{J} \\ 0 \end{pmatrix},$$

which is a first-order system of differential equations.

Main options:

- Finite-difference time-domain methods (*e.g.*, the Yee scheme)
  - Pro: nice to parallelize
  - Con: stability condition requires small time steps
- Pseudo-spectral methods
  - Pro: unconditionally stable
  - Con: requires Fourier transforms of source, which typically means costly global communications (notable exception: the pseudo-spectral analytical time-domain (PSATD) method of Vay *et al.*[1])

We take a different approach utilizing Green's functions, with the aim of obtaining a method that is easy to parallelize and does not face stability issues.

[1] Vay *et al. A domain decomposition method for pseudo-spectral electromagnetic simulations of plasmas. 2013. J. Comput. Physics, 260-268*

By Duhamel's principle the solution to the first order system is given in terms of the operator exponential as

$$\begin{pmatrix} \mathbf{E}(t) \\ \mathbf{B}(t) \end{pmatrix} = \exp(Lt) \begin{pmatrix} \mathbf{E}(0) \\ \mathbf{B}(0) \end{pmatrix} - \int_0^t \exp(Ls) \begin{pmatrix} \mathbf{J}(t-s) \\ 0 \end{pmatrix} ds,$$

and thus we simply need a numerical scheme for applying the propagator $P(t) \equiv \exp(Lt)$.

Questions

1. What is the propagator $P(t)$ analytically?
2. How do we treat it numerically?
3. Why do this?

After some analysis, the Fourier transform of our propagator is given by

$$\tilde{P}(t) = \begin{pmatrix} \tilde{P}_1(t) & \tilde{P}_2(t) \\ -\tilde{P}_2(t) & \tilde{P}_1(t) \end{pmatrix},$$

with blocks given by

$$\tilde{P}_1(t) \equiv \hat{\mathbf{k}}\hat{\mathbf{k}}^T + \frac{1}{c}\frac{\partial}{\partial t}\left(\frac{\sin c|\mathbf{k}|t}{|\mathbf{k}|}\right)(I - \hat{\mathbf{k}}\hat{\mathbf{k}}^T)$$
$$\tilde{P}_2(t) \equiv \frac{\sin c|\mathbf{k}|t}{|\mathbf{k}|}K,$$

where here K is the transform of the curl operator. Note the decomposition of $\tilde{P}_1$ into its effect on longitudinal (curl-free) and transverse (divergence-free) fields.

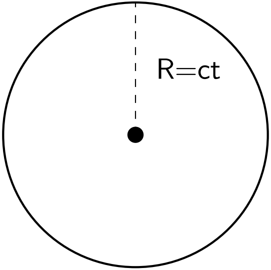
We assume that longitudinal fields are to be treated explicitly by some separate Poisson solve and thus that we need only worry about the effect of the propagator on transverse fields. Then, $P(t)$ applied to the transverse fields can be written in the spatial domain as

$$P\begin{pmatrix} \mathbf{E} \\ \mathbf{B} \end{pmatrix} = \begin{pmatrix} \frac{1}{c}\frac{\partial}{\partial t}\left(\frac{\delta(|\mathbf{x}|-ct)}{4\pi ct}\right) * \mathbf{E} + \frac{\delta(|\mathbf{x}|-ct)}{4\pi ct} * \nabla \times \mathbf{B} \\ -\frac{\delta(|\mathbf{x}|-ct)}{4\pi ct} * \nabla \times \mathbf{E} + \frac{1}{c}\frac{\partial}{\partial t}\left(\frac{\delta(|\mathbf{x}|-ct)}{4\pi ct}\right) * \mathbf{B} \end{pmatrix},$$

where calculus gives that

$$\frac{1}{c}\frac{\partial}{\partial t}\left(\frac{\delta(|\mathbf{x}|-ct)}{4\pi ct}\right) * f(\mathbf{x})$$
$$= \frac{1}{4\pi c^2 t^2}\left[\delta(|\mathbf{x}|-ct) * f(\mathbf{x}) - \sum_{i=1}^{3}[x_i \delta(|\mathbf{x}|-ct)] * \frac{\partial f(\mathbf{x})}{\partial x_i}\right].$$

Note that this is essentially Kirchhoff's integral theorem[2] for the solution of the wave equation in 3D in a different form, and corresponds (roughly) to convolving with the delta distribution supported on a sphere of radius ct.



[2] See, *e.g.*, `http://www.math.unm.edu/~lorenz/463/h.pdf`, section 4.3

Treating the delta distribution

We have reduced the problem of applying the propagator to the problem of accurately evaluating 3D spatial convolutions of the form

$$\delta(|\mathbf{x}| - ct) * f(\mathbf{x})$$

and

$$[x_i \delta(|\mathbf{x}| - ct)] * \partial_{x_i} f(\mathbf{x}).$$

To do this, we use finite differences for ∂_{x_i} and borrow a method typically used to treat embedded boundaries or singular source terms.

Consider the 1D case. We follow Beyer & LeVeque[3] and say that $W_h(x)$ is an approximation to the discrete delta distribution on the uniform grid with spacing h that is order-p accurate if

$$f(\alpha) - h \sum_j f(x_j) W_h(x_j - \alpha) = \mathcal{O}(h^p)$$

for any $\alpha \in \mathbb{R}$, *i.e.*, if W_h acts like the delta distribution when used as a convolutional operator. They show that a sufficient condition is that W_h satisfy the discrete moment conditions,

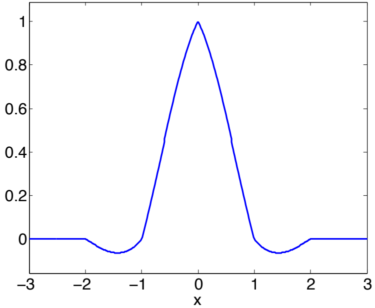
$$h \sum_j (x_j - \alpha)^q W_h(x_j - \alpha) = \delta_{q,0}$$

for $q = 0, \ldots, p-1$.

[3] Beyer & LeVeque. Analysis of a One-Dimensional Model for the Immersed Boundary Method. 1992. SIAM Journal on Numerical Analysis.

Spline interpolation

To generate such functions, one method is to use piecewise polynomial interpolation (B-splines and variants) of the true delta distribution. These are key in that they can attain the minimum neccesary support for a given order[4].



[4] Tornberg & Engquist. Numerical approximations of singular source terms in differential equations. 2004. Journal of Computational Physics.

To treat delta distributions in 3D on arbitrary surfaces with arbitrary weighting functions, we follow the tensor-product formulation of Peskin[5], where we approximate the 3D delta distribution on surface Γ with weighting g as

$$\delta_h(\Gamma, g, \mathbf{x}) \equiv \int_\Gamma \prod_{k=1}^{d} \delta_{h_k}(x_k - X_k(S)) g(S)\, dS,$$

which Tornberg and Engquist show is accurate to the same order p as the individual discrete deltas, in the sense that

$$\left(\prod_{k=1}^{d} h_k\right) \sum_{j\in\mathbb{Z}^d} \delta_h(\Gamma, g, \mathbf{x}_j) f(\mathbf{x}_j) - \int_\Gamma g(S) f(\mathbf{X}(S))\, dS = \mathcal{O}(h^p).$$

[5] Peskin. The immersed boundary method. 2002. Acta Numerica.

- If we use a finite-difference scheme that matches the order p of our delta approximation and a sufficiently high-order integration scheme to construct the propagator, we obtain a method that is consistent in the sense that the error in one application of the operator converges to 0 as h^p.
- However, in practice, we will not use this method as a one-step method because we want to treat sources, so we must consider stability, as mentioned.

Stability, abridged

The full stability statement is long, so we give an abridged statement here.

Theorem

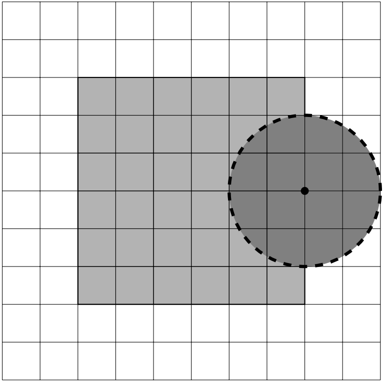
Under suitable assumptions on the finite-difference stencil used to compute derivatives, the numerical scheme involving convolution against these discrete operators satisfies the Von Neumann condition for

$$\sigma \equiv \frac{c\Delta t}{h} \geq 1.$$

Note that we don't observe any problems with smaller σ, we just can't prove anything.

Domain decomposition

We make an observation: our discretization respects the finite propagation speed of EM waves (on purpose). This admits domain decomposition.

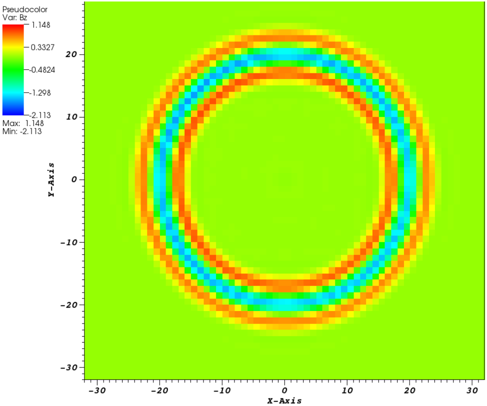


Algorithm

1. Decompose the domain and include an appropriate number of ghost cells for your choice of $c\Delta t$.
2. Pre-compute the convolutional propagator on each subdomain.
3. At each time step, tack on the source term and convolve with the propagator (simple Trapezoid rule for the time integral in Duhamel) using local FFTs (Hockney's method).

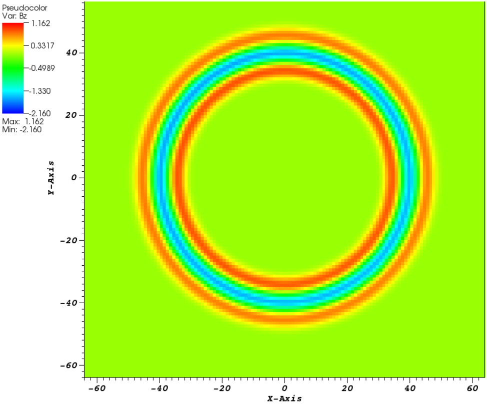
Results

A simple example: $\sigma = 10$, no source, some number of time-steps, look at cross-section of B_z when initial $\mathbf{E}$ is something small and smooth supported in the center. 64^3 points.



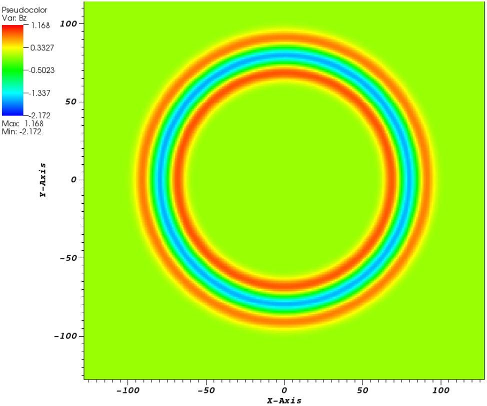
Results

A simple example: $\sigma = 10$, no source, some number of time-steps, look at cross-section of B_z when initial **E** is something small and smooth supported in the center. 128^3 points.



Results

A simple example: $\sigma = 10$, no source, some number of time-steps, look at cross-section of B_z when initial **E** is something small and smooth supported in the center. 256^3 points.



Conclusions and future work

- So far, seems like a pretty versatile way of handling singular propagators, but still running tests
- Need to actually use in PIC (else why are we doing this?)
- Would like to incorporate adaptive mesh refinement
- Tangential question for follow-up: found a way of generating some interesting B-splines for interpolation purposes, does the wavelet community know about these?

Acknowledgments

V. M. is supported by a U.S. Department of Energy Computational Science Graduate Fellowship under grant number DE-FG02-97ER25308.

Thanks!