Stable and Accurate Outgoing Wave Filters for Anisotropic and Nonlocal Waves

Avy Soffer

Department of Mathematics, Rutgers University
Piscataway, NJ 08854, USA

Chris Stucchio

Courant Institute of Mathematical Sciences, New York University
NY, NY, 10012-1185, USA
http://cims.nyu.edu/∼stucchio/

The Perfectly Matched Layer (PML) is currently the mainstay of absorbing boundary conditions. For some anisotropic wave equations the PML is exponentially unstable in time. We present in this work a new method of open boundaries, the phase space filter, which is stable for all wave equations.

Outgoing waves can be are waves located near the boundary of the computational domain with group velocities pointing out. Phase space filtering involves periodically removing only outgoing waves from the solution, leaving non-outgoing waves unchanged. We apply this method to the Euler equations (linearized about a jet flow), Maxwell equations in a birefringent medium and the quasi-geostrophic equations.

1. Introduction

We consider numerical approximations of linear wave equations:

$$\vec{u}_t(\vec{x},t) = \mathcal{H}\vec{u}(\vec{x},t) \tag{1}$$

$\vec{u} : \mathbb{R}^N \to \mathbb{R}^n$ (or $\mathbb{R}^N \to \mathbb{C}^n$) is a vector-valued wave-field and $\mathcal{H} = \mathcal{H}(i\nabla)$ is a skew-adjoint linear differential operator. Eq. (1) has plane wave solutions $e^{i(\vec{k}\cdot\vec{x}-\omega_j(\vec{k})t)}\vec{d}_j(k)$ with $\vec{d}_j(\vec{k})$ the j'th eigenvector of $\mathcal{H}(\vec{k})$ and $\omega_j(\vec{k})$ the j-th eigenvalue of $\mathcal{H}(\vec{k})$ in the frequency domain. If we are only interested in $\vec{u}(x,t)$ in a finite region $B \subset \mathbb{R}^N$, it simplifies the computations to solve (1) only on this region. Steps must be taken to prevent spurious reflection from the artificial boundary, which is the topic of this article.

Exact transparent boundary conditions can be constructed, but can be difficult to work with.[1–4] The PML is a more versatile,[5] but for some

anisotropic wave equations it can be exponentially unstable in time[5–7] which makes it inaccurate*.

The Time Dependent Phase Space Filter (TDPSF) is a new approach to the problem of open boundaries,[10–13] and in this work we apply it the problem of anisotropic waves. The key idea in this approach is that outgoing waves live in certain regions of *phase space*; by filtering these regions, outgoing waves can be removed before they reach the computational boundary.

The phase space projections originally used were based on the Gaussian windowed Fourier transform.[10–12] Here, we extend the work of[10–13] to more general wave equations, and simultaneously simplify the method. We begin by briefly reviewing the dynamics of linear waves.

1.1. *Outgoing waves*

To pin down notation, we review wave propagation. Recall that $\mathcal{H}(i\nabla)$ is skew-adjoint, i.e. for each $\vec{k}$, $\mathcal{H}(\vec{k})$ is a skew-adjoint matrix (with complex entries). $\mathcal{H}(\vec{k})$ can be diagonalized by the matrix $D = D(\vec{k})$, which is the (unitary) matrix having j'th row equal to $\vec{d}_j(k)$:

$$\mathcal{H} = D^\dagger \begin{bmatrix} i\omega_1(\vec{k}) & \dots & 0 \\ \dots & i\omega_j(\vec{k}) & \dots \\ 0 & \dots & i\omega_n(\vec{k}) \end{bmatrix} D \tag{2}$$

The function $\vec{d}_j(\vec{k})e^{i(\vec{k}\cdot x - \omega_j(\vec{k})t)}$ is a plane wave solution to (1). The operator $e^{\mathcal{H}t}$ is the propagator for (1), i.e. $e^{\mathcal{H}t}\vec{u}(x,0) = \vec{u}(x,t)$. In practice, we compute this with Fast Fourier Transforms, i.e. $\mathrm{FFT}^{-1}\exp(\mathcal{H}(\vec{k})t)\mathrm{FFT}\vec{u}(x,0)$.

Consider an initial condition $\vec{u}(x,0) = e^{i\vec{k}_0 x}\vec{d}_j(\vec{k}_0)g(\vec{x}-\vec{a})$, with $g(\vec{x})$ smooth and well localized (e.g. a gaussian). Stationary phase shows that:

$$\begin{aligned} \vec{u}(x,t) &= e^{\mathcal{H}(\vec{k})t}\vec{d}_j(\vec{k}_0)\hat{g}(\vec{k}-\vec{k}_0) \\ &= \exp([\omega_j(\vec{k}_0) + \nabla_k\omega_j(\vec{k}_0) + O((\vec{k}-\vec{k}_0)^2)]t)\vec{d}_j(\vec{k}_0)e^{i\vec{k}\cdot\vec{a}}\hat{g}(\vec{k}-\vec{k}_0) \\ &\approx e^{i(\vec{k}_0 x - \omega_j(\vec{k}_0)t)}g_t(\vec{x}-\vec{a}-\nabla_k\omega_j(\vec{k}_0)t) \end{aligned} \tag{3}$$

The envelope $g_t(x)$ disperses due to the $O((\vec{k}-\vec{k}_0)^2)$ term. Thus, wavepackets of the form $\vec{u}(x,0) = e^{i\vec{k}_0 x}\vec{d}_j(\vec{k}_0)g(\vec{x}-\vec{a})$ propagate along the trajectory $\vec{a}+\nabla_k\omega_j(\vec{k}_0)t$ while spreading out. The TDPSF algorithm consists of identifying and removing wave components with outgoing trajectories (and only outgoing trajectories) before they reach the computational boundary.

*It can also be polynomially unstable in time[8] due to problems near $\vec{k} = 0$ which is unrelated to anisotropy. This issue has been resolved.[9]

2. The TDPSF Algorithm

The TDPSF algorithm involves solving (1) on the extended region $[-L-w, L+w]^N$ with any accurate interior solver, with $[-L, L]^N$ the region of interest, and the extra space a filtering buffer.

We assume that $\vec{u}(x, 0)$ has small high frequency components, as well as a more technical assumption that the frequency content of $\widehat{\vec{u}}_0(\vec{k})$ is localized away from the regions where the group velocity turns around (see Section 2.1). This requirement is necessary due to the Heisenberg uncertainty principle. In what follows, $v_{\max}$ is the largest relevant group velocity:

Algorithm 1. ***TDPSF Propagation***

Input:

- *The dispersion relations and diagonalizing matrices, $\omega_j(\vec{k})$ and D.*
- *$e^{\mathcal{H}_b t}$, a propagator that accurately solves the interior problem.*
- *$k_{\max}$, the maximal frequency of the problem.*

Algorithm:

`let` $\vec{u}_d(x, 0)$ `:=` $\vec{u}_0(x)$ *# Initial condition*
`let` $T_{\text{step}} \leq w/3v_{\max}$ *# Time between filtering operations*
`let` M `:=` $\lceil T_{\max}/T_{\text{step}} \rceil$ *#Number of iterations*

`for` $n \in \{0, 1, \ldots, M = T_{\max}/T_{\text{step}}\}$ `do:`

(1) Propagation step:
`let` $\vec{v}(x)$ `:=` $e^{\mathcal{H} T_{\text{step}}} \vec{u}_d(x, nT_{\text{step}})$
(2) Filtering step ($\mathcal{O}_j^{\pm}$ are defined in Section 2.1):
`let` $\vec{u}_d(\vec{x}, (n+1)T_{\text{step}})$ `:=` $\left[\prod_{j=1}^{N}(1-\mathcal{O}_j^{+})(1-\mathcal{O}_k^{-})\right]\vec{v}(x)$
(3) `output` $\vec{u}_d(x, (n+1)T_{\text{step}})$.

The algorithm consists of propagating over time intervals T_{step} too short for even the fastest waves to cross the buffer region and reach the boundary (step 1 inside the for-loop). Step 2 inside the loop consists of filtering outgoing waves, which allows the solution to be propagated further (the waves we filtered are the waves which *would have* reached the boundary. All that remains is to construct the filters, $\mathcal{O}_j^{\pm}$.

2.1. *Construction of the boundary filter*

For a fixed boundary region (say the right boundary), if $\partial_1 \omega_j(\vec{k}) > 0$, then waves with frequency $\vec{k}$ are outgoing at this boundary. The outgoing region

of *phase space* at the right boundary is:

$$\{(\vec{x},\vec{k}) \in \mathbb{R}^N \times \mathbb{R}^N : \vec{x}_1 > L \text{ and } \partial_1 \omega_j(\vec{k}) > 0\}. \quad (4)$$

We construct a projection onto this region. The Heisenberg uncertainty principle makes exact projections impossible, but we can come close. Define:

$$\chi_j^{\pm}(x) = (1/2)[\mathrm{erf}(\sigma^{-1}(x - L - w/3)) - \mathrm{erf}(\sigma^{-1}(x - L - 2w/3))] \quad (5)$$

The parameter $\sigma = O(w/\ln(\delta^{-1})^{1/2})$ and must satisfy (6) to ensure that $\chi_j^{\pm}(x) < \delta$ for $x_j \notin [\pm L, \pm(L+w)$ or $x_k \notin [-L-w, L+w]$. Multiplication by $\chi_j(x)$ smoothly projects onto the buffer on the j'th side of the box. δ is an error tolerance, and $\star$ denotes convolution.

Define $R_{j,l} = \{\vec{k} \in \mathbb{R}^N : \partial_{k_j}\omega_l(\vec{k}) > 0\}$ to be the set of frequencies with the l'th branch of the group velocity pointing right and $R_{j,l,\delta} = \{\vec{k} \in R_k : d(\vec{k}, R_{j,l}^C) > k_b\}$ to be the same set excluding a "buffer" region around the place where the group velocity turns around. The buffer ensures that frequency spreading caused $\chi_j(x)$ does not cause waves to turn around. Given k_b, we choose $\sigma \geq O({k_b}^{-1}\ln(\delta^{-1})^{1/2})$ to make $\chi_j(x)$ smooth enough to minimize frequency spreading.

We *assume* that the frequency content of $\widehat{\vec{u}}(\vec{k},t)$ is not contained in $R_{j,l} \setminus R_{j,l,\delta}$. The exact constraints on σ, w and δ are:

$$ {k_b}^{-1}\left(\ln(\delta^{-1}) + \ln\left(w^2 L^{N-1} 2^{3N}\sigma^{3N}\pi^{-3N/2}\right)\right)^{1/2} \leq \sigma \leq \frac{w}{\sqrt{\ln(\delta^{-1}) + N\ln(2\sigma\pi^{-1/2})}} \quad (6)$$

This constraint ensures that the phase space filters are accurate.[14]

Define $P_{j,l,\delta}(k) = (2\sigma\pi^{-1/2})^N e^{-k^2\sigma^2} \star_k \mathrm{diag}[1_{R_{j,l,\delta}}(\vec{k}), \ldots, 1_{R_{j,n,\delta}}(\vec{k})]$, which is a a smooth projection (in the basis of eigenvectors of $\mathcal{H}$) onto wave-vectors propagating rightward. Then the operator $D^\dagger P(\vec{k}) D$ is the projection in the basis of wave-vectors. Finally, we define the operator:

$$\mathcal{O}_1^+ = \chi_1^+(x) D^\dagger P(\vec{k}) D \chi_1^+(x) \quad (7)$$

This operator is an approximate projection onto waves with group velocity pointing outward, and localized in the boundary region.

2.2. *Implementation Details*

One useful property of the TDPSF algorithm is that it is compatible with any reasonable interior solver including Fourier spectral methods. Fourier-based spectral methods are highly accurate independent of the timestep; the

only errors are frequency aliasing, machine errors and boundary errors[10,13] (which tend to dominate the others). This is the method we use.

The outgoing wave filters are calculated by truncating to the boundary region after multiplying by $\chi_j^{\pm}(\vec{x})$ and computing an FFT to apply the frequency domain operators[†]. Therefore, as a practical matter, it is useful to take $w = 2^m \delta x$, with δx the lattice spacing in $\vec{x}$ and $m \in \mathbb{N}$.

The simulations were implemented in Python using numpy and matplotlib[‡]. Source code is available from the webpage of the second author.

3. Examples

3.1. *A warm-up: 1-dimensional Schrödinger equation*

Phase space filters were originally developed for the Schrödinger equation ($u(x,t)$ is a scalar field, and $\mathcal{H} = i\Delta$). In this case there is no diagonalizing operator and $\nabla_k \omega(\vec{k}) = k$. Outgoing waves at the right boundary are waves of positive frequency, so $P(k) = \left(2\sigma\pi^{-1/2}\right) e^{-k^2/\sigma^2} \star 1_{k>k_b}(k)$ and:

$$\mathcal{O}_1^+ = \chi_1(x)(2\sigma\pi^{-1/2})[e^{-k^2/\sigma^2} \star 1_{k>k_b}(k)]\chi_1(x)$$

We solved the Schrödinger equation on a lattice of 1024 pts ($\delta x = 0.1$) with initial condition $u(x, t = 0) = e^{ikx} e^{-x^2/2\cdot 7^2}$. We measured the errors for various values of δ, taking $\sigma = 1$. The results are plotted in Figure 1; the error floor at 10^{-8} is due to machine errors. The error tolerance is achieved, except for waves close to $k = 0$ (where the group velocity turns around).

3.2. *Hyperbolic Systems: the Euler and Maxwell equations*

We now consider two cases where the standard PML is unstable. The Euler equations (linearized about a jet flow with mach number M) can be written in the form (1) with $\vec{u} = (p, v_1, v_2)$ (p is pressure, $\vec{v}$ is fluid velocity):

$$\mathcal{H} = \begin{bmatrix} M\partial_{x_1} & -\partial_{x_1} & -\partial_{x_2} \\ -\partial_{x_1} & M\partial_{x_1} & 0 \\ -\partial_{x_2} & 0 & M\partial_{x_1} \end{bmatrix} \tag{8}$$

Maxwell's equations can be written similarly; with μ a scalar and as-

[†]This causes negligable error since $\chi_j^{\pm}(\vec{x}) \leq \delta$ for $\vec{x}$ outside the boundary region.

[‡]Numpy is available from http://numpy.scipy.org, and matplotlib is available from http://matplotlib.sourceforge.net/

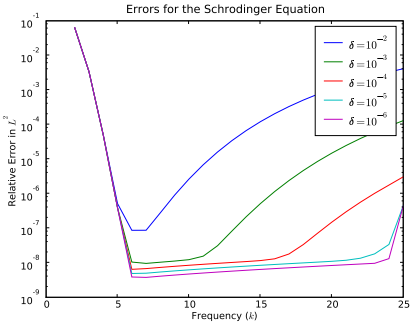

Fig. 1. The relative error for the 1-dimensional test of the Schrodinger equation, as a function of k and the parameter δ.

suming z-independence, they take the form (1) with $\vec{u} = (\sqrt{\mu}\vec{H}, \sqrt{\epsilon}\vec{E})^T$

$$\mathcal{H} = \begin{bmatrix} 0 & -\mu^{-1/2}\nabla \times \epsilon^{-1/2} \\ \epsilon^{-1/2}\nabla \times \mu^{-1/2} & 0 \end{bmatrix}, \ \epsilon = \begin{bmatrix} 1 & b & 0 \\ b & 1 & 0 \\ 0 & 0 & c \end{bmatrix}. \tag{9}$$

We solved these examples on the computational region $[-32, 32]^2$ with $\delta x = 0.125$ (512^2 lattice points), $T_{\max} = 50$ and $k_{\max} = \pi/\delta x = 25.1$. The filter parameters are $w = 16$ (128 lattice points), $T_{\text{step}} = 1.5$ and $\sigma = 1.0$. The initial condition was $u_1(x, t = 0) = r^2 e^{-r^2/9} \cos(Kr)$ with $r = \sqrt{(x-8)^2 + y^2}$ with K varying from 1 to 20, which is localized in frequency near near $|\vec{k}| = K$. For $K > 4$, L^2 error of 10^{-3} is achieved.

3.3. *Linearized Quasi-Geostrophic Equations*

We now consider the linearized Quasi-Geostrophic equations (also called the midlatitude planetary equations). By linearizing about a streamfunction $\psi(\vec{x}, t) = -Vy$ (meaning the velocity $\vec{v} = [-\partial_y\psi, \partial_x\psi]$), the quasi-geostrophic equations take the form:[15]

$$\partial_t\psi - V\partial_x\psi + \tilde{\beta}(-\Delta + F)^{-1}\partial_x\psi = 0$$

This has a dispersion relation $\omega(\vec{k}) = k_1(V - \tilde{\beta}(|\vec{k}|^2 + F)^{-1})$. Here, V is the mean wind, F is a constant proportional to f^2/g, with f the rotation frequency of the earth, g the gravitational attraction and $\tilde{\beta} = FV + \beta$ and $\beta = R\cos(\phi)$ where R is the radius of the earth and ϕ is the latitude[§]. The

[§]We work in the β-plane approximation, i.e., we study the equations in spherical coordinates, expand trigonometric terms (functions of lattitude and longitude) in a Taylor

numerical parameters are the same as above, and we took $V = 1$, $F = 10$, $\beta = 100$; the results are displayed in Fig. 3. Although the results are not as good as for the Euler or Maxwell equations, they are acceptable, and can be improved by taking a larger buffer region. This is due primarily to the non-locality of the equation, not the anisotropy.

In,[7] it is proven for hyperbolic systems that, the PML on the i-th side is stable only when $k_i v_{g,i}(\vec{k}) > 0$. While the anisotropic Euler and Maxwell equations do not satisfy this criteria, the PML can be made stable for those cases (essentially by completing the square). The quasi-geostrophic equations can not be fixed in this way.

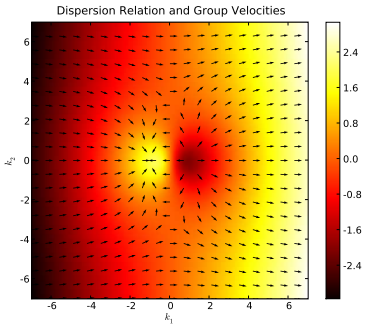

Fig. 2. The dispersion relation and group velocities for the quasigeostrophic equation.

4. Stability

We have the following rigorous estimate[14] regarding Algorithm 1's stability:

$$\|\vec{u}_d(x,t)\|_{L^2} \leq \|\vec{u}(x,t')\|_{L^2} \text{ if } t > t' \tag{10}$$

To test this, we solved the Euler, Maxwell and Schrödinger equations up to time $t = 2000$ and measured the energy,[14] confirming (10).

5. The Low Frequency Problem

Figures 1 and 3 indicate that the TDPSF performs poorly for waves with low frequency. Increasing the width of the filter imposes a computational cost of order $O(k_{\max}/k_b)$ just to resolve the buffer, which is undesirable.

series and work on a small cartesian region.

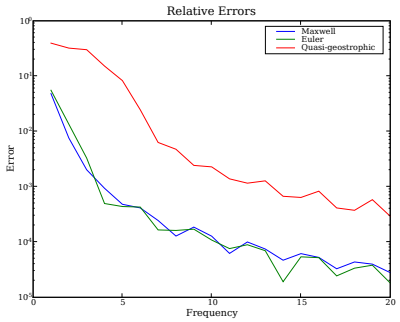

Fig. 3. The relative errors (measured in various norms) for Euler and Maxwell systems as a function of the frequency of the initial condition.

This can be remedied by a multiscale method which imposes cost only $O(\log_2(k_{\max}/k_b))$; we believe this is close to the best possible.[13]

Acknowledgements: We thank Peter Petropoulos for pointing out to us the problem of PML instability and Tom Hagstrom for comments which led to Section 3.3.

References

1. B. Alpert, L. Greengard and T. Hagstrom, *J. Comput. Phys.* **180**, 270 (2002).
2. A. Bayliss and E. Turkel, *Comm. Pure Appl. Math.* **33**, 707 (1980).
3. B. Engquist and A. Majda, *Proc. Nat. Acad. Sci. U.S.A.* **74**, 1765 (1977).
4. S. Jiang and L. Greengard, *Communications in Pure and Applied Mathematics* **61**, 261 (2008).
5. J.-P. Berenger, *J. Comput. Phys.* **114**, 185 (1994).
6. F. Hu, *J. Comput. Phys.* **129**, 201 (1996).
7. E. Becache, S. Fauqueux and P. Joly, *J. Comput. Phys.* **188**, 399 (2003).
8. D. G. S. Abarbanel and J. Hesthaven, *Journal of Scientific Computing* **17** (2002).
9. P. G. Becache, E. Petropoulos and S. D. Gedney, *IEEE Trans. Ant. Prop.* **52** (2004).
10. A. Soffer and C. Stucchio (2006), in preparation.
11. A. Soffer and C. Stucchio, *Journal of Computational Physics* **225**, 1218 (2007).
12. C. Stucchio, Selected Problems in Quantum Mechanics, PhD thesis, Rutgers University, (Piscataway, NJ, 2008).
13. A. Soffer and C. Stucchio, *Communications in Pure and Applied Mathematics* **(Accepted)**.
14. A. Soffer and C. Stucchio, *(Submitted)* (2008).
15. A. Majda, *Introduction to PDEs and Waves for the Atmosphere and Ocean* (American Mathematical Society, Providence, RI, 2003).