Finite-difference wavefield propagation using superstepping

T. Nemeth[1], K. Nihei[1], A. Loddoch[1], A. Sekar[1], K. Bube[2], J. Washbourne[1], L. Decker[1], S Kaplan[1], Ch. Wu[1], A. Shabelansky[1], O. Cristea[1] and Z. Yin[3], [1]Chevron Technical Center, a division of Chevron U.S.A. Inc., [2]University of Washington, [3]Georgia Institute of Technology

Summary

This paper describes how to propagate wavefields for arbitrary numbers of traditional time steps in one step, called a superstep. It is achieved by implementing a computational tradeoff differing from traditional single step wavefield propagators by precomputing Green's functions for each model location for *k* timesteps (a superstep) and using these Green's functions to propagate the wavefield *k* time steps at once. This tradeoff separates physics (Green's function computation) from computer science (wavefield propagation) and allows each discipline to provide their optimal modular solutions.

Introduction

Traditional wavefield propagation uses finite-difference time domain (FD) schemes to advance a wavefield one timestep at a time to perform a matrix-vector multiplication of the current and previous wavefields with the combination of material properties (such as velocity or density) in a pattern described by the FD kernel. The accuracy, stability and dispersion of such FD schemes have been studied extensively and are well understood [1], [2], [3], [4], [5]. Today, most modeling and imaging applications are built on linear or linearized FD schemes from acoustic forward modeling to VTI, TTI and elastic imaging. Such schemes can be viewed as a series of matrices (one for each time step) applied to the current wavefield vector. It is an accepted view that computationally it is better to carry out this computation sequentially from "right" to "left", that is, start by multiplying the zeroth-time wavefield vector on the right with the matrix for the first time step resulting in a new vector; applying the next time step matrix to the new vector, and so on, until all matrix-vector multiplications are executed.

We have researched numerical formulations for wavefield propagation where the tradeoff for compute and storage is different from the traditional formulation and the algorithmic components are generalized. This effort led us to recast the process of wavefield propagation as two distinct steps of "Precompute" and "Compute". During Precompute, the common reusable components (Green's functions) are calculated and stored. This Precompute stage is separate from the subsequent Compute step and can therefore be executed on separate systems possibly with a different hardware architecture. During the Compute stage, those precomputed components (Green's functions) are used and re-used to achieve wavefield propagation.

The key observation to superstepping is to re-arrange the FD schemes in such a way that current and previous wavefields are related to future and subsequent wavefields via a square system matrix that can be raised to the kth power corresponding to superstepping of k traditional time steps. When the initial wavefield is a delta function for each spatial location, we can precompute the Green's functions at each location to the kth traditional time steps. These precomputed Green's functions become impulse response filters at these locations. Then we apply these filters with the actual wavefields to advance the wavefield by k steps.

Similar ideas have appeared in different contexts. Seismic interferometry [6], [7], [8] uses Green's functions to reconstruct wavefields from other wavefields. Similarly, Marchenko methods [9], [10] reconstruct subsurface Green's functions and use them for imaging. Superstepping can be considered as a special case of seismic interferometry. Recently many novel machine learning techniques were developed that train to predict the system states continuously from current time to up to a given future time. These methods, such as the Fourier Neural Operators [11], [12] use the inherent nonlinearity in the neural network to provide a model for inference. Although both machine learning methods and superstepping can produce future states of the system, superstepping is based on the linear properties of the system and is not an approximation to the traditional time stepping schemes.

Theory

A typical first-order system of equations with 2 state variables for wave propagation can be written in a matrix equation form as

$$\begin{pmatrix} \dot{\mathbf{u}} \\ \dot{\mathbf{v}} \end{pmatrix} = \begin{pmatrix} 0 & A \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \end{pmatrix} \quad , \qquad (1)$$

where $\dot{\mathbf{u}}$ and $\dot{\mathbf{v}}$ are temporal derivatives of state variables $\mathbf{u}$ and $\mathbf{v}$ (typically particle velocity and stress). Matrices A and B incorporate material parameter coefficients and partial derivative operators. This system can be discretized as a symplectic Euler scheme,

$$\begin{aligned} \mathbf{u}_{n+1} &= \mathbf{u}_n + A\,\mathbf{v}_n \\ \mathbf{v}_{n+1} &= \mathbf{v}_n + B\,\mathbf{u}_{n+1} \end{aligned} \qquad (2)$$

where $\dot{\mathbf{u}} = (\mathbf{u}_{n+1} - \mathbf{u}_n)/\Delta t$, $\dot{\mathbf{v}} = (\mathbf{v}_{n+1} - \mathbf{v}_n)/\Delta t$ and Δt is a suitably chosen time step that is absorbed into matrices A and B here for simplicity. Equation 2 can be expressed in matrix form as

$$\begin{pmatrix} \mathbf{u}_{n+1} \\ \mathbf{v}_{n+1} \end{pmatrix} = \begin{pmatrix} I & A \\ B & I + BA \end{pmatrix} \begin{pmatrix} \mathbf{u}_n \\ \mathbf{v}_n \end{pmatrix} \quad . \qquad (3)$$

Note that the system matrix in Equation 3 is a square matrix composed of 2x2 block matrices. The block matrices do not need to be square matrices themselves. Applying the system matrix once advances the wavefield one time step. Repeated applications of the system matrix in Equation 3 corresponds to repeated time steps. Therefore, applying *k* time steps corresponds to raising the system matrix to the *k*th power

$$\begin{pmatrix}\mathbf{u}_{n+k}\\ \mathbf{v}_{n+k}\end{pmatrix} = \begin{pmatrix} I & A\\ B & I+BA\end{pmatrix}^{k} \begin{pmatrix}\mathbf{u}_{n}\\ \mathbf{v}_{n}\end{pmatrix} \quad . \tag{4}$$

Equation 4 implies that we can calculate the $\mathbf{u}_{n+k}$, $\mathbf{v}_{n+k}$ wavefields from $\mathbf{u}_n$, $\mathbf{v}_n$ by either sequentially applying the propagator matrix *k* times as in traditional time stepping, or calculating the effects of the propagator matrix to the *k*th power and applying it to $\mathbf{u}_n$, $\mathbf{v}_n$.

Next, we extend to second-order systems based on the learnings from first-order equations. The corresponding second-order system (in time) derived from Equation 1 is

$$\begin{pmatrix}\ddot{\mathbf{u}}\\ \ddot{\mathbf{v}}\end{pmatrix} = \begin{pmatrix} 0 & A\\ B & 0\end{pmatrix} \begin{pmatrix}\dot{\mathbf{u}}\\ \dot{\mathbf{v}}\end{pmatrix} = \begin{pmatrix} AB & 0\\ 0 & BA\end{pmatrix} \begin{pmatrix}\mathbf{u}\\ \mathbf{v}\end{pmatrix} \quad . \tag{5}$$

After some manipulations and using the **v** state variable we show that the *n+k-1* and *n+k* wavefields can be expressed from the *n-1* and *n* wavefields by raising a square matrix composed of 2x2 block matrices to the *k*th power

$$\begin{pmatrix}\mathbf{v}_{n+k-1}\\ \mathbf{v}_{n+k}\end{pmatrix} = \begin{pmatrix} 0 & I\\ -I & 2I+BA\end{pmatrix}^{k} \begin{pmatrix}\mathbf{v}_{n-1}\\ \mathbf{v}_{n}\end{pmatrix} \quad . \tag{6}$$

Note that the structure of the system matrix corresponds to the structure of the system matrix for the first-order system, except that the block matrices in the first column are leading negative.

Algorithm

Both the first- and second-order *k*th-power propagator matrices in Equations 4 and 6 can be formally written as

$$S(k) = \begin{pmatrix} S_{11}(k) & S_{12}(k)\\ S_{21}(k) & S_{22}(k)\end{pmatrix} \tag{7}$$

where S(k) denotes the *k*th superstep propagator matrix with the pre-calculated block matrices. Each block matrix $S_{ij}(k)$ corresponds to the *i*th output state variable vector and the *j*th input state variable vector. The block matrices are discretized forms of linear integral kernel operators. In wave equations these kernel operators correspond to Green's functions and the functions these kernels operate on are the time source functions. Therefore, $S_{ij}(k)$ are a set of Green's functions, one for each row of each block matrix and $\mathbf{u}_n$, $\mathbf{v}_n$ are the time source functions. The Green's functions correspond to the unit input for each element in the input vector calculated by solving the wave equation for *k* time steps. This requires knowing the earth model parameters only and therefore can be pre-computed and stored for subsequent use. Advancing the wavefield *k* time steps at once then corresponds to computing the matrix-vector multiplies for each block matrix generating scalars and adding these scalars to generate a single scalar that corresponds to the *k*-advanced wavefield at that position. The multiplication of a row of a block matrix with the input vector is a dot product. This dot product evaluation is illustrated in Figure 1 on a subset of the Marmousi data.

We can generalize the number of state variables to more than two. While **u** and **v** shown in Equation 1 are representative, these vectors can be multi-component themselves. For example, **u** can be single-component if it represents a scalar variable such as pressure, or 3-component vector $\mathbf{u} = (u_x, u_y, u_z)$ if it represents a gradient vector field, or a 6-component vector $\mathbf{u} = (u_{xx}, u_{yy}, u_{zz}, u_{xy}, u_{xz}, u_{yz})$ if it represents the components of more complex differential operators. The number of components can be different for **u** and **v**. The full wavefield **w** is a combination of wavefields **u** and **v** and has N_c components where N_c is the sum of the number of components in **u** and **v**. At any location *i* at any time step *n* there are N_c state variable values, and we can represent them by denoting them as $\mathbf{w}_{i\alpha j}$. For scalar-component **u** and **v** this corresponds to $N_c = 2$. Since the state variables scale by N_c for both the input and the output, the corresponding system matrix is the size of $N_c \times N_c$ block matrices. These Green's functions at a given location corresponds to the various combinations of input and output components. We will denote these Green's functions as $G_{i\alpha\beta}(k)$ where *i* is the index of the physical location, α is the index of the output component, β is the index of the input component and *k* is the superstep size. Finally, index *j* corresponds to a particular shot record. Note that the Green's functions are independent of the shot records but the wavefield $\mathbf{w}_{i\alpha j}$ carries this index. The Algorithm Superstep Wavefield Propagation in the pseudo-code below implements the generalized propagator

Algorithm Superstep Wavefield Propagation

```
Function Superstep( k, $n_0$, $N_t$, $N_{mod}$, $N_c$, $N_s$, w($n_0$), G(k) )
  for n = $n_0$ : k : $N_t$ do                              ▷ iterate in superstep index
  | for i = 1 : $N_{mod}$ do                                ▷ iterate in model location
  | | for α = 1 : $N_c$ do                                  ▷ loop for output wavefield
  | | | for j = 1 : $N_s$ do                                ▷ iterate over shots
  | | | | for β = 1 : $N_c$ do                              ▷ loop for input wavefield
  | | | | | $G_{i\alpha\beta}(k) = GetGreensFunctions(i, \alpha, \beta)$
  | | | | | $\widetilde{w}_{i\alpha\beta j}(n) = CutWavefieldSegment(\mathbf{w}(n))$
  | | | | $w_{i\alpha j}(n+k) = \sum_{\beta=1}^{N_c} G_{i\alpha\beta}(k) \cdot \widetilde{w}_{i\alpha\beta j}(n)$
  return w
```

function. Function *GetGreensFunction* returns all the Green's functions $G_{i\alpha\beta}(k)$ for a given model location. Function *CutWavefieldSegment* isolates the wavefield segments from the overall wavefield that matches the spatial extent with their corresponding Green's functions returning $\widetilde{w}_{i\alpha\beta j}(n)$. The updated variables $\mathbf{w}_{i\alpha j}$(n + k) are the sum of the dot products of the Green's functions $G_{i\alpha\beta}(k)$ and the corresponding state variables $\widetilde{w}_{i\alpha\beta j}(n)$at the *n*th timestep.

Superstepping computational model

Superstepping creates two fundamentally different tasks to propagate wavefields: (a) Precompute - creating Green's functions for a given model and (b) Compute - using the Green's functions to advance the wavefields.

Precompute: Green's functions represent the propagator matrix and the wavefields represent the input and output vectors. Since the earth model parameters are all contained in the propagator matrix, the Greens' functions represent all the physics used in wavefield propagation, including the actual physics encoded in the partial differential equations, all the numerical issues of physics such as dispersion, boundary conditions, in addition to the distribution of the Green's functions in space and the number of Green's function at each spatial location. Therefore, they can be precomputed and apply to all measurement time ranges, and to all shot records. This presents an opportunity to generate high quality Green's functions where all the desired physics is incorporated. One way to accelerate the computation of Green's functions is to reduce their size to their physical domain of influence by removing the areas external to the wavefront. This preserves numerical precision. This can be thought of as bringing two different parts of physics into the Green's functions before discarding physics: (a) computing the Green's function coefficients based on physics and (b) determining the causal regions based on both math and physics. Another way to improve the computed Green's functions is to incorporate the physics of handling boundary reflections into them. We can also reduce dispersion in the Green's functions by employing high fidelity numerical schemes during their computation. Since Green's functions are shared in different time scales and among survey records, the optimization efforts provide significant value for large-scale runs. Green's functions represent the filter that moves the incoming wavefields *k* timesteps forward. These same Green's functions can also be used to move the incoming wavefield *k* timesteps backward essential for computing gradients (full waveform inversion) or migration images (reverse time migration). Green's functions can be further improved by generating representations to reduce grid effects or improving the wavefield continuity. These representations can be accomplished by either traditional signal processing or machine learning methods. Green's functions can also be used for approximations of the full wave equations, such as rays, dynamic rays, or other simpler geometrical optics approximations as smooth gradients with second order derivatives.

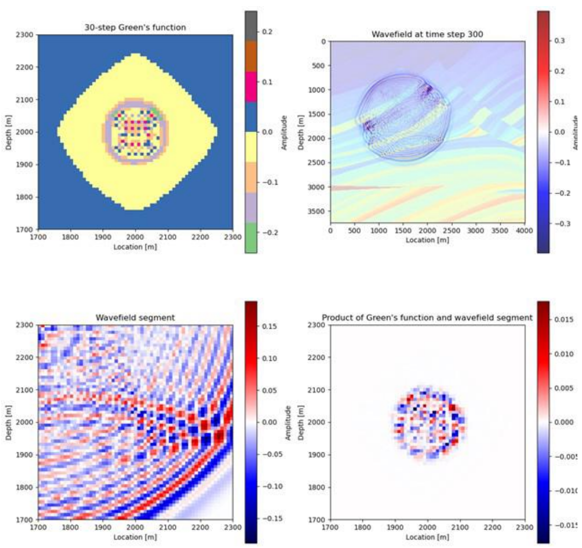

Figure 1: Dot product calculation between a Green's function and the corresponding wavefield. Top left: Green's function after 30 time steps. Top right: Wavefield after 300 timesteps in the physical 2D domain. Bottom left: Wavefield segment corresponding to the physical area of the Green's function. Bottom right: Element-wise multiplication of the Green's function and the wavefield segment. These values are summed up to a single scalar.

Compute: Wavefield propagation in superstepping is agnostic to physics as physics has already been parameterized and incorporated into Green's functions. Therefore, one can consider wavefield propagation to be a computer science task that uses parameterized Green's functions. From this perspective, the implementations need to be general to efficiently map to a broad range of compute resources and at the same time allow for efficient representations of Green's functions and their interactions with wavefields. The generality of mapping is facilitated by the simplification of tasks during superstepping: (a) keep Green's functions in memory awaiting wavefields; (b) select and crop the wavefields to the Green's function size and bring them to the corresponding memory and (c) compute the dot product between the Green's functions and the cropped wavefields. These operations capture the available resources on a compute node, namely, memory, interconnect and compute resources. Maximizing all these resources is thus a computational optimization goal. The Algorithm describes a distributed computing model that is unusual from traditional finite-difference modeling perspective. It favors parallelization by earth model locations and not by individual shot record locations. Note that the earth model locations (loops indexed by *i*) are outer to the shot records (loops indexed by *j*) in the Algorithm.

Numerical Example

The superstepping scheme was applied to a 300x300 grid section of the Marmousi model shown in the bottom left figure in Figure 2. The source location is in the middle of the model at (1500m, 1500m). The top left figure shows the wavefield after 300 traditional time steps and the top right shows the corresponding superstep-propagated wavefield after 9 30-traditional-step supersteps were applied to the initial wavefield. The bottom right figure shows the difference between the wavefields in the top left and right figures. As expected, the superstep results closely matched the results achieved using traditional time stepping.

Conclusions

We have developed a superstepping scheme for linear or linearized partial differential equations where the effect of the repeated application of the propagator matrix can be precomputed in an efficient manner. This result has several important consequences.

The superstep formulation is a general scheme that applies to all linear discretized PDEs. The presented Algorithm is applicable to acoustic or elastic wave equations, and even more complex discretized equations. The only difference between the various equations is in the content and the number of those Green's functions.

The superstep formulation clearly separates tasks that are associated with the physics and tasks associated with wavefield propagation. This was our original goal when we initiated this effort. The first tasks can be pre-computed and the resulting Green's functions used in a later wavefield propagation. We have shown that all physics can be accounted for during the generation of the Green's functions and these Green's functions can be further enhanced.

The wavefield propagation task is well-suited for computer science optimization and compute resource mapping. Specifically, the three steps in wavefield propagation: (a) keeping Green's functions in memory awaiting wavefields; (b) selecting and cropping the wavefields to the Green's function size and bring them to the corresponding memory and (c) compute the dot product between the Green's function and the cropped wavefield, are fully utilizing the resources on compute nodes and allowing computational optimization without the need to know the underlying physics.

Green's functions are computed for each grid location in the earth models and the locations indices are in the outer loops in the Algorithm. This results in a distributed computing scheme where the main distribution factor is the placement of sufficient number of Green's functions in a node's memory. Efficient memory utilization is the driving force for computational optimization, and it underscores that most large-scale wavefield propagation jobs are bandwidth limited.

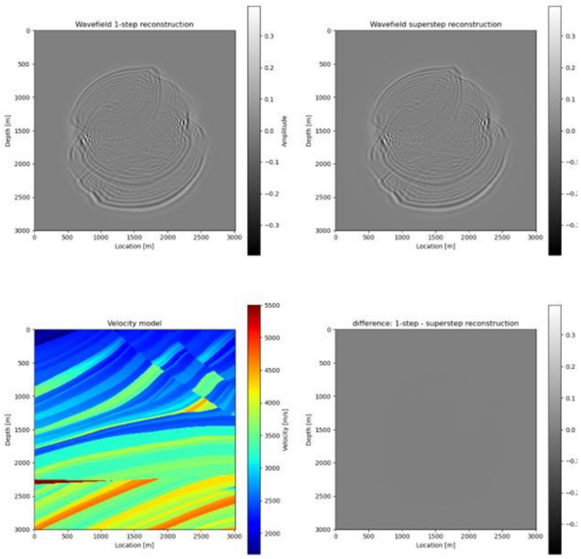

Figure 2: Wavefield after 300 traditional time steps. Top left: Wavefield after 300 steps calculated by traditional method; Top right: Wavefield after 300 steps calculated by superstep method. Initial wavefield at 30 steps was advanced by 30 steps at once using Greens functions; Bottom left: Velocity model (301x301 subset of Marmousi); Bottom right: Difference between the 1-step and superstep methods at the time scale of the wavefield.

It would be essentially futile to advance wavefields via superstepping if we would need to interpolate back to all regular time steps for imaging or adjoint-state-based gradient computation. We have developed imaging schemes using superstepping that avoid these interpolations and the wavefield propagation for gradients (FWI) and imaging (RTM) can proceed using superstepping.

Since the wavefield propagation is heavily influenced by computational optimization, the future success of this method fully depends on strong domain-specific language implementations.

Acknowledgements

We thank Chevron Technical Center for the permission to publish this paper.

REFERENCES

Bube, K. P., T. Nemeth, J. P. Stefani, R. Ergas, W. Liu, K. T. Nihei, and L. Zhang, 2012, On the instability in second-order systems for acoustic VTI and TTI media: Geophysics, **77**, no. 5, T171–T186, doi: https://doi.org/10.1190/geo2011-0250.1.

Bube, K. P., T. Nemeth, J. P. Stefani, W. Liu, K. T. Nihei, R. Ergas, and L. Zhang, 2012, First- order systems for elastic and acoustic variable-tilt TI media: Geophysics, **77**, no. 5, T157–T170, doi: https://doi.org/10.1190/geo2011-0249.1.

Chapman, C. H., 2004, Fundamentals of seismic wave propagation: Cambridge University Press.

Durran, D. R., 1999, Numerical methods for wave equations in geophysical fluid dynamics: Springer.

Grady, T. J., R. Khan, M. Louboutin, Z. Yin, P. A. Witte, R. Chandra, R. J. Hewett, and F. J. Herrmann, 2023, Model-parallel Fourier neural operators as learned surrogates for large-scale parametric PDEs: Computers & Geosciences, **178**, doi: https://doi.org/10.1016/j.cageo.2023.105402.

Hairer, E., G. Wanner, and C. Lubich, 2006, Geometric numerical integration. structure-preserving algorithms for ordinary differential equations, Springer, ISBN: 978-3-540-30666-5

Li, Z., N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and Anandkumar, A., 2020, Fourier neural operator for parametric partial differential equations. arXiv preprint arXiv:2010.08895.

Lomas, A., and A. Curtis, 2019, An introduction to Marchenko methods for imaging: Geophysics, **84**, no. 2, F35–F45, doi: https://doi.org/10.1190/geo2018-0068.1.

Schuster, G.T., 2009. Seismic interferometry: Cambridge Univ. Press.

Slob, E., K. Wapenaar, F. Broggini, and R. Snieder, 2014b, Seismic reflector imaging using internal multiples with Marchenko-type equations: Geophysics, **79**, no. 2, S63–S76, doi: https://doi.org/10.1190/geo2013-0095.1.

Wapenaar, K., and J. Fokkema, 2006, Greens function representations for seismic interferometry: Geophysics, **71**, no. 4, SI33–SI46, doi: https://doi.org/10.1190/1.2213955.

Wapenaar, K., J. Thorbecke, and J. van der Neut, 2016, A single-sided homogeneous Green's function representation for holographic imaging, inverse scattering, time-reversal acoustics and interferometric Green's function retrieval: Geophysical Supplements to the Monthly Notices of the Royal Astronomical Society, **205**, 531–535, doi: https://doi.org/10.1093/gji/ggw023.