Electronic Seismologist

PyKonal: A Python Package for Solving the Eikonal Equation in Spherical and Cartesian Coordinates Using the Fast Marching Method

Malcolm C. A. White*[1], Hongjian Fang[2], Nori Nakata[2], and Yehuda Ben-Zion[1]

Abstract

This article introduces PyKonal: a new open-source Python package for computing travel times and tracing ray paths in 2D or 3D heterogeneous media using the fast marching method for solving the eikonal equation in spherical and Cartesian coordinates. Compiled with the Cython compiler framework, PyKonal offers a Python application program interface (API) with execution speeds comparable to C or Fortran codes. Designed to be accurate, stable, fast, general, extensible, and easy to use, PyKonal offers low- and high-level API functions for full control and convenience, respectively. A scale-independent implementation allows problems to be solved at micro, local, regional, and global scales, and precision can be improved over existing open-source codes by combining different coordinate systems. The resulting code makes state-of-the-art computational capabilities accessible to novice programmers and is efficient enough for modern research problems in seismology.

Cite this article as White, M. C. A., H. Fang, N. Nakata, and Y. Ben-Zion (2020). PyKonal: A Python Package for Solving the Eikonal Equation in Spherical and Cartesian Coordinates Using the Fast Marching Method, *Seismol. Res. Lett.* **91**, 2378–2389, doi: 10.1785/0220190318.

Supplemental Material

Introduction

How long do seismic waves generated at one point take to reach another? What path does the energy take to get there? These basic questions are key for seismologists' ability to locate earthquakes, image subsurface structure, and pursue many other fundamental studies. The vast literature documenting different approaches to answering these questions dwarfs the number of corresponding practical tools freely available to seismologists. Here, we present PyKonal: a new open-source Python package for computing travel times and tracing ray paths using the fast marching method (FMM) for solving the eikonal equation.

Most numerical approaches for computing travel times and ray paths can be classified into one of two main categories: ray-based methods, which solve the kinematic ray equations; and grid-based methods, which usually solve the eikonal equation using finite differences or use Dijkstra-like network algorithms (Dijkstra, 1959) to determine the shortest path between two points. Ray-based methods compute travel times along individual ray paths and provide no explicit information about travel times along alternative ray paths. They are favorable for their speed when the number of ray paths that the user is interested in is small, but they suffer from inaccuracy and instability in regions with significant velocity heterogeneity. Grid-based methods, on the other hand, compute the entire travel-time field—that is, from the source to every point on a grid—and then use the gradient of the travel-time field to determine individual ray paths. While less efficient for determining individual ray paths, they are preferable for their stability and accuracy in complex media and are more efficient when travel times are needed at a large number of points.

The kinematic ray equations can be formulated as an initial-value problem, which can be solved using shooting methods (e.g., White, 1989), or as a boundary-value problem, which can be solved using bending methods (e.g., Julian and Gubbins, 1977; Thurber and Ellsworth, 1980; Um and Thurber, 1987). See Rawlinson *et al.* (2008) for a thorough review of ray- and grid-based methods.

A plethora of grid-based methods for solving the eikonal equation have been proposed since the late 1980s, each with

1. Department of Earth Sciences, University of Southern California, Zumberge Hall of Science (ZHS), Los Angeles, California, U.S.A.; 2. Department of Earth, Atmosphere and Planetary Sciences, Massachusetts Institute of Technology, Cambridge, Massachusetts, U.S.A.

*Corresponding author: malcolm.white@usc.edu

different computational complexity and stability properties, originating primarily in the fields of computational mathematics and physics. Most grid-based eikonal solvers implement the concept of a narrowband computational front (a limited region of the computational domain for which information is strategically propagated from regions where the solution is known to regions where it is unknown). Sweeping methods are one class of methods, stemming from the work of Zhao (2004), that omit the concept of a narrowband, which, despite their computational efficiency, find little use in seismological applications because of their instability in heterogeneous media.

Grid-based eikonal solvers implementing a narrowband generally use either an expanding box or the expanding wavefront as the computational front. Reshef and Kosloff (1986) were probably the first to propose a grid-based method for solving the eikonal equation in seismological applications, and Vidale (1988) contributed significantly to the expanding-box formulation. The most severe limitation of Vidale's method is that it breaches the causality of the eikonal equation under certain conditions and is thus unstable. A number of efforts have been directed at resolving this instability and improving the efficiency and generality of that expanding-box formulation (e.g., Vidale, 1990; Podvin and Lecomte, 1991; van Trier and Symes, 1991; Schneider *et al.*, 1992; Hole and Zelt, 1995; Afnimar and Koketsu, 2000). High-order essentially nonoscillatory (ENO) finite-difference schemes (Harten and Osher, 1987) and their weighted ENO extensions (Liu *et al.*, 1994) have been implemented in the expanding-box framework for seismological applications (e.g., Kim and Cook, 1999; Qian and Symes, 2002a,b; Buske and Kastner, 2004); however the postprocessing proposed to overcome the instability inherent in expanding-box methods (e.g., Kim and Cook, 1999) increases algorithmic complexity.

Qin *et al.* (1992) were apparently the first to replace the expanding box with a computational front that conforms approximately to the shape of the advancing wavefront. This expanding-wavefront formulation always honors the causality of the eikonal equation, and is thus unconditionally stable, but does so at the cost of increased computational expense because a sorted list of narrowband values must always be maintained. In an independent effort, Sethian (1996) combined the stability of the expanding-wavefront formulation with the computational efficiency of heap-sort technology to develop the FMM, which simultaneously offers unconditional stability and computational efficiency. Sethian and Popovici (1999) introduced the FMM to the seismological community and Rawlinson and Sambridge (2004a) generalized the method for multiple reflection and transmission phases.

Although previous work established a solid practical foundation for estimating travel times and ray paths used to advance the understanding of the Earth's interior, improving precision over and providing greater flexibility than existing software can lead to better results. We implement the FMM here, because it is relatively easy to code (reducing the likelihood of bugs), unconditionally stable, and computationally efficient. The presented new Python package, PyKonal, can improve on various seismic imaging applications. PyKonal can solve the eikonal equation in two or three dimensions using Cartesian or spherical coordinates and combinations thereof. With possible applications to locating earthquakes, performing seismic tomography, computing ray-path parameters, and Kirchhoff depth migration, PyKonal is designed to be accurate, stable, fast, general, extensible, and easy to use.

In the following sections, we outline the theory and implemented methodology (see the Method section), present the performance of the implementation with respect to the design criteria mentioned earlier (see the Results section), and compare with existing tools and discuss potential extensions and use cases (see the Discussion section). Concluding remarks, including discussion of ongoing work, are finally offered (see the Conclusions section).

Method

In this section, we review the derivation of the eikonal equation following Rawlinson *et al.* (2008), state the problem solved by the FMM, and describe how we implement it.

Deriving the eikonal equation

The homogeneous scalar wave equation

$$\nabla^2\psi(\mathbf{r},t)=\frac{1}{v^2(\mathbf{r})}\frac{\partial^2\psi(\mathbf{r},t)}{\partial t^2}, \tag{1}$$

is a second-order linear partial differential equation in space and time with general solutions of the following form:

$$\psi(\mathbf{r},t)=A(\mathbf{r})e^{\pm i\omega(\tau(\mathbf{r})-t)}, \tag{2}$$

in which ψ, $\mathbf{r}$, t, v, A, ω, and τ represent the wavefunction, position vector, time, wave velocity, wave amplitude, angular frequency, and the spatially dependent travel time, respectively.

Substituting equation (2) into equation (1) gives

$$(\nabla^2A-\omega^2|\nabla\tau|^2\pm i\omega(2\nabla A\cdot\nabla\tau+A\nabla^2\tau))\psi=-\frac{1}{v^2}\omega^2\psi, \tag{3}$$

in which the centered dot represents the dot product operator. After factoring out ψ and separating the real and imaginary parts, equation (3) yields a pair of equations:

$$\nabla^2A-\omega^2|\nabla\tau|^2=-\frac{\omega^2}{v^2}, \tag{4}$$

and

$$2\nabla A\cdot\nabla\tau+A\nabla^2\tau=0. \tag{5}$$

Dividing equation (4) by ω^2 and taking the high-frequency limit $\omega \to \infty$ gives the eikonal equation:

$$|\nabla\tau|^2 = \frac{1}{v^2}. \tag{6}$$

The high-frequency approximation made to obtain equation (6) is conventionally held to be valid when the wavelength of the propagating wave is much shorter than the scale of velocity structure heterogeneity, although discontinuities are permitted (Rawlinson *et al.*, 2008).

Expression (5) is called the transport equation and can be used to compute the amplitude of the propagating wave as a function of space if the travel-time field is known (e.g., Qian and Symes, 2002a; Buske and Kastner, 2004).

Statement of the problem

Given the velocity field, $v(\mathbf{r})$, we aim to determine the travel-time field, $\tau(\mathbf{r})$, by solving equation (6) under various boundary conditions.

Consider a parametric surface in $\mathbb{R}^3$, $\Gamma(t)$, representing a zero-phase wavefront that propagates perpendicular to itself with velocity v assumed to be a strictly positive function of space, and let $\tau(\mathbf{r})$ be the time it takes to reach a point with position vector $\mathbf{r}$ from an arbitrary boundary value, $\Gamma(t = 0)$. The eikonal equation (6) approximates the relationship between $\tau(\mathbf{r})$ and $v(\mathbf{r})$ at high frequencies, and $\Gamma(t)$ is the surface defining the wavefront at time t—that is, the set of $\mathbf{r}$ such that $\tau(\mathbf{r}) = t$. The evolution of $\Gamma(t)$ can be tracked by solving equation (6) for $\tau(\mathbf{r})$, given $v(\mathbf{r})$ and appropriate boundary conditions. The problem is to solve equation (6) for $\tau(\mathbf{r})$, given $v(\mathbf{r})$ and $\Gamma(t = 0)$.

Solution

Sethian (1996) developed an efficient numerical method—the FMM—for approximating a solution to equation (6), subject to an entropy condition: the wavefront can only cross each point once. Imposing this condition implies that the solution obtained comprises only first arrivals; the wavefront propagates along the minimal path in terms of time between any two points.

The next section presents the FMM as implemented herein. See the original paper by Sethian (1996) for a detailed derivation and proof that the method produces a valid solution of the eikonal equation.

The FMM

Evaluating equation (6) at a discrete number of points gives

$$|(\nabla\tau)_{ijk}|^2 = \frac{1}{v_{ijk}^2}, \tag{7}$$

in which

$$f_{ijk} \equiv f(i\Delta\xi_1, j\Delta\xi_2, k\Delta\xi_3), \tag{8}$$

with ijk being indexes belonging to the set of integers, and $\Delta\xi_1$, $\Delta\xi_2$, and $\Delta\xi_3$ being discretization intervals along the ξ_1, ξ_2, and ξ_3 axes of a general orthogonal coordinate system spanning $\mathbb{R}^3$, respectively.

To satisfy the entropy condition imposed on the approximate solution to equation (6) obtained by the FMM, the gradient operator must be approximated in such a way that each component of the travel-time gradient is always nonnegative. Doing so ensures that information always flows in the same direction as the propagating first-arrival wavefront. Different entropy-satisfying approximations have been proposed (e.g., Osher and Sethian, 1988; Rouy and Tourin, 1992) and, as in Sethian and Popovici (1999), we choose the scheme from Rouy and Tourin (1992):

$$|(\nabla\tau)_{ijk}| \approx \sum_{a=1}^{3} \max\left(\frac{1}{h_{\xi_a}} D_{ijk}^{-\xi_a}, -\frac{1}{h_{\xi_a}} D_{ijk}^{+\xi_a}, 0\right), \tag{9}$$

in which h_{ξ_a} represents the scale factor along the ξ_a axis for the used coordinate system, and $D_{ijk}^{-\xi_a}$ and $D_{ijk}^{+\xi_a}$ represent, respectively, backward and forward finite-difference approximations to the derivative of τ at ijk along the ξ_a axis. Substituting equation (9) into equation (7) gives

$$\sum_{a=1}^{3} \max\left(\frac{1}{h_{\xi_a}} D_{ijk}^{-\xi_a}, -\frac{1}{h_{\xi_a}} D_{ijk}^{+\xi_a}, 0\right)^2 - \frac{1}{v_{ijk}^2} \approx 0. \tag{10}$$

Equation (10) has an upwind difference structure—a causality relationship implying that the value of τ_{ijk} is fully determined by neighboring values, τ_{lmn}, such that $\tau_{lmn} < \tau_{ijk}$. In other words, information propagates in one direction—from lesser values of τ to greater—and unknown values of τ can be derived from neighboring known values by solving equation (10). Thus, starting with known values $\tau_{ijk} = 0$, the domain of known values can be expanded until it includes all values by iteratively solving equation (10). This amounts to solving a quadratic equation in τ_{ijk} after expansion. The efficiency of the FMM rests on carefully choosing the order in which to update the unknown values, as presented in the following paragraph and in Figure 1.

Initialize three empty sets: "Known," "Trial," and "Unknown." Assign all nodes to Unknown. Set $\tau_{ijk} = 0$ for all nodes on the initial wavefront and transfer them from Unknown to Trial. Observe that the upwind structure of equation (10) implies that the node in Trial with the smallest value of τ cannot change (if multiple values are equally the smallest, choose one at random—this will not affect the solution); transfer it from Trial to Known and temporarily label it "Active." Update the value of each neighbor of Active that is not in Known using values in Known to solve equation (10) and transfer any of them that are in Unknown to Trial. Iteratively, transfer the node in Trial with the smallest value

```
foreach ijk in grid do
    τ_ijk ← ∞;
    Append ijk to Unknown;
end
foreach ijk ∈ Γ (t = 0) do
    τ_ijk ← 0;
    Transfer ijk from Unknown to Trial;
end
while length(Trial) > 0 do
    ĩjk ← index of node with smallest value of τ in Trial;
    Transfer ĩjk from Trial to Known;
    foreach Neighbour, lmn, of ĩjk do
        τ*_lmn ← solution of Eqn (10);
        if τ*_lmn < τ_lmn then
            τ_lmn ← τ*_lmn;
            if lmn ∈ Unknown then
                Transfer lmn from Unkown to Trial;
            end
        end
    end
end
```

Figure 1. Essential elements of the fast marching method algorithm for solving the eikonal equation (6) implemented by PyKonal.

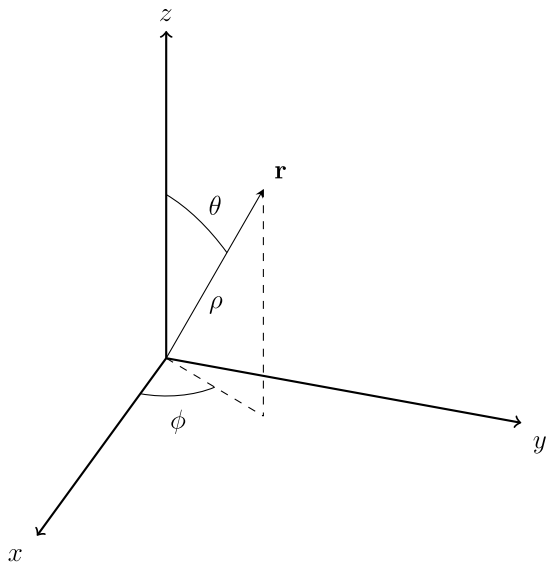

Figure 2. International Organization for Standardization coordinate-system convention adopted for spherical coordinates, in which x, y, and z are standard Cartesian coordinates, and ρ, θ, and φ are radial, polar, and azimuthal coordinates, respectively, with $\rho \in [0, \infty]$, $\theta \in [0, \pi]$, and $\phi \in [0, 2\pi)$.

of τ to Known, and update its neighbors until all nodes are in Known.

Any sorting algorithm can be used to determine the node in Trial with the smallest value of τ. Following Sethian (1996), we use a heap-sort algorithm, which has $\mathcal{O}(N \log N)$ worst-case performance in which N is the total number of grid nodes. In practice, the computational efficiency of the FMM is nearly linear in N.

Coordinate systems

The method has been presented in a general coordinate system thus far, but a specific choice must be made in practice, and each coordinate system has strengths and weaknesses, which are illustrated in the Results section. Choosing a particular set of coordinates is tantamount to defining the scale factors, h_{ξ_n}, in equation (10). In Cartesian coordinates, the scale factors are $h_x = h_y = h_z = 1$, and in spherical coordinates they are $h_\rho = 1$, $h_\theta = \rho$, and $h_\varphi = \rho \sin\theta$, in which the International Organization for Standardization's convention is adopted for spherical coordinates (Fig. 2).

Spherical coordinates are ideal for tracking spherical wavefronts, whereas Cartesian coordinates are ideal for tracking planar wavefronts (see the Results section). Seismologists often treat sources of seismic energy as point sources, making spherical coordinates suitable at small and intermediate distances. Wavefronts from sufficiently distant sources are often assumed to be planar, for which Cartesian coordinates are ideal. When wavefronts must be tracked over great distances and the sphericity of the Earth must be accounted for, spherical coordinates are again naturally suited to the problem. It is often desirable to combine different coordinate systems when solving a particular problem.

Implementing the FMM solution of the eikonal equation in a general coordinate system makes integrating different systems easy. To combine different coordinate systems, we first solve the eikonal equation in one, then map the solution into a second, and solve the eikonal equation in the remaining unknown region. This procedure can be iterated ad infinitum, but combining two or three coordinate systems is likely sufficient for most problems in seismology.

The unconditional stability of the FMM owes to a proper entropy-condition-satisfying approximation for the gradient and has only been proven in the case of first-order-accurate finite-difference operators. Strictly speaking, transitioning between coordinate systems may destabilize the FMM by violating the entropy condition. Transitioning from the first coordinate system to the second assumes that the wavefront does not propagate from the second back into the first anywhere. This is likely of little concern in the vast majority of practical use cases; however, users should be aware of this aspect and use appropriate caution.

One further caveat that users should be aware of is the singularity in the gradient operator at the origin of spherical coordinate systems. The eikonal equation is unsolvable there, because the gradient operator is undefined. Thus, PyKonal will raise an error if the user attempts to place a node at the origin of a spherical grid. Grid nodes can be placed sufficiently close

to the origin that this has little effect in practice, but any wavefront propagating through the origin will be distorted. Future updates to PyKonal may include implementing an unstructured-mesh update scheme (Sethian, 1999) at the origin of spherical coordinate systems to mitigate the singularity there.

Order of the finite-difference operators

The accuracy of the FMM depends partially on the order of the finite-difference operators used for which generic operators are specified in equation (10). The simplest but most inaccurate scheme uses only first-order operators:

$$D_{ijk}^{-\xi_1} = D_{ijk}^{-\xi_1^1} \equiv \frac{\tau_{ijk} - \tau_{i-1jk}}{\Delta\xi_1}, \tag{11}$$

$$D_{ijk}^{+\xi_1} = D_{ijk}^{+\xi_1^1} \equiv \frac{\tau_{i+1jk} - \tau_{ijk}}{\Delta\xi_1}, \tag{12}$$

and similar definitions in ξ_2 and ξ_3, in which ξ_m^n indicates an nth-order finite-difference operator along the ξ_m axis (e.g., $D_{ijk}^{+\xi_1^1}$ and $D_{ijk}^{-\xi_3^2}$ represent the finite-difference approximation of the derivative of τ computed using a first-order forward operator along the ξ_1 axis and a second-order backward operator along the ξ_3 axis, respectively). More accurate approximations can be obtained using higher-order operators, for example, second-order operators:

$$D_{ijk}^{-\xi_1} = D_{ijk}^{-\xi_1^2} \equiv \frac{\tau_{i+2jk} - 4\tau_{i+1jk} + 3\tau_{ijk}}{2\Delta\xi_1}, \tag{13}$$

$$D_{ijk}^{+\xi_1} = D_{ijk}^{+\xi_1^2} \equiv -\frac{\tau_{i-2jk} - 4\tau_{i-1jk} + 3\tau_{ijk}}{2\Delta\xi_1}, \tag{14}$$

and similar definitions in ξ_2 and ξ_3.

The causality of equation (10) implies that using higher-order operators is only permissible when the travel-time field at all nodes involved in an update is monotonically decreasing away from the node being updated. For example, a second-order update along the x axis using a backward finite-difference operator is only permissible when $\tau_{i-2jk} \leq \tau_{i-1jk} \leq \tau_{ijk}$. As in Sethian (1999), we implement second-order operators whenever known values satisfy the causality condition and first-order operators otherwise, which means that the update scheme is of mixed order.

Raytracing

Energy propagates perpendicular to the wavefronts in isotropic media, so the gradient of the travel-time field is always tangent to the ray path between two points. Thus, the ray path can be expressed as a parametric curve, $\mathbf{R}(s)$, satisfying

$$\frac{\mathrm{d}\mathbf{R}(s)}{\mathrm{d}s} = \nabla\tau, \tag{15}$$

in which s represents distance along the curve. Equation (15) is a first-order ordinary differential equation, which can be solved by integrating over ds using the Runge–Kutta method. PyKonal implements a simple first-order Runge–Kutta method (Euler's method).

Because the source location necessarily coincides with the global minimum of the travel-time field, the gradient of the travel time vanishes there, and we cannot integrate equation (15) from source to receiver. Instead, we integrate equation (15) in the reverse direction, from receiver to source, which means that we find the path of steepest descent (in terms of travel time) between the receiver and source.

The fact that the global minimum of the travel-time field coincides with the source location presents a convenient condition for stopping integration of equation (15). Because we are following the path of steepest travel-time descent, the travel time at each point along the ray path should be lower than at every point that precedes it. Thus, the ray will satisfy $\tau(\mathbf{R}(s_1)) < \tau(\mathbf{R}(s_2))$ for all $s_1 > s_2$. If the ray overshoots the source location, it will start traveling "uphill" and will violate this condition. Thus, we stop integrating as soon as the ray encounters a point associated with a travel-time value exceeding that of the immediately preceding point.

Results

We now demonstrate that five of our six goals in building this tool are achieved: The code is accurate, stable, fast, general, and extensible. Users may judge whether our tool satisfies our sixth goal: easy to use.

Accuracy

Consider the trivial case of waves emanating from a point source with $v = 1$ km/s everywhere (Fig. 3). Spherical wavefronts expand outward from the source, reaching a given point after an amount of time that is in direct proportion to the distance from the source. Comparing this analytical solution to a numerical solution computed using Cartesian coordinates reveals numerical anisotropy. As discussed by Alkhalifah and Fomel (2001), errors are the greatest in the near-source region—where wavefront curvature is large relative to the node spacing—and in regions where the wavefront is oblique to the coordinate axes. In spherical coordinates, however, the numerical solution is exact up to computational precision, because the radial axis is always orthogonal to the wavefront and the partial derivative of the travel-time field with respect to azimuth (and polar angle in 3D) is always zero. This simple test suggests that, even for more complicated velocity fields, spherical coordinates centered on a point source are more accurate than Cartesian coordinates in the near-field region where wavefront curvature is high.

If the point source in the previous example is replaced by a line (in the 2D case) or a plane (in the 3D case) source, the opposite holds and Cartesian coordinates are more accurate

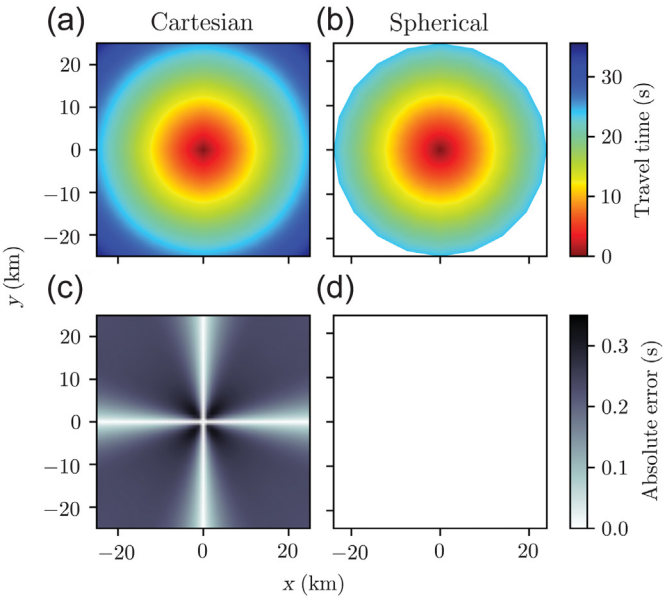

Figure 3. (a,b) Travel-time fields and (c,d) absolute errors for a point source at $(x, y) = (0, 0)$ computed in (a,c) Cartesian coordinates and (b,d) spherical coordinates. (b,d) Portions of the plots are blank because the radial axis of the spherical grid used extends to a maximum of 25 km.

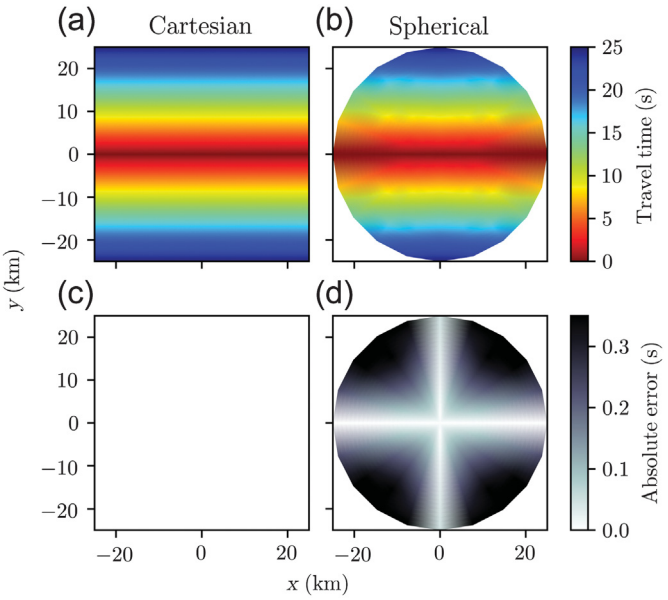

Figure 4. (a,b) Travel-time fields and (c,d) absolute errors for a line source at $y = 0$ computed in (a,c) Cartesian coordinates and (b,d) spherical coordinates. (b,d) Portions of the plots are blank because the radial axis of the spherical grid used extends to a maximum of 25 km.

(Fig. 4). In this case, the y axis is always orthogonal to the wavefronts leading to an accurate solution, whereas spherical coordinates suffer from numerical anisotropy.

Much work has been devoted to reducing numerical anisotropy introduced by finite-difference schemes, particularly in the field of hyperbolic partial differential equations (see Sescu, 2015, for a thorough review). Adapting such approaches to our context seems plausible; however, the preceding two examples elucidate an important feature of the FMM for solving the eikonal equation: judicious design of the computational grid can yield highly accurate solutions while effectively minimizing numerical anisotropy. Because numerical anisotropy can be effectively reduced in this manner, we avoid endeavoring here to implement complicated finite-difference schemes to further reduce it.

Stability

To be useful for solving modern problems in seismology, any raytracing tool must be numerically stable in highly complex velocity structures. The FMM is unconditionally stable (Sethian and Popovici, 1999)—meaning that solutions converge to a stable solution that is consistent with the exact solution in the limit that the node intervals go to zero—making it well suited to any problem in seismology for which the eikonal equation is valid. To demonstrate this convergent behavior, we consider a point source at the surface of the Marmousi2 velocity model (Versteeg, 1994)—a standard model in exploration seismology comprising complex structures with lateral and vertical gradients, discontinuities, inversions, folded layers, pinch outs, and lenses (Fig. 5). Solutions converge toward a stable solution as the node intervals are successively halved. Unconditional stability results from proper choice of an entropy-satisfying gradient approximation (expression 9), and although it has only been proven in the case of first-order accurate finite-difference operators, these results suggest that the algorithm remains stable with higher-order operators for complex practical use cases.

The unconditional stability of the FMM implies that ray paths obtained by integrating equation (15) can be made arbitrarily accurate by making the computational grid sufficiently dense (Fig. 6). Errors accumulate along the integration path, so they are least at the beginning of the integration path (near the receiver) and greatest at the end (near the source). This error accumulation is exacerbated by relatively large travel-time errors in the source region. In addition, the criteria used for terminating integration permits inaccurate ray paths to overshoot the source location; however, the amount of overshoot decreases as grid density increases.

Speed

Having established the accuracy of our FMM implementation for the simplest cases (see the Accuracy section) and its stability in the complex case of the Marmousi2 model (see the Stability section), we turn to its execution speed. We consider a suite of 3D problems of various sizes with a random velocity model and a source at a corner of the grid. The execution

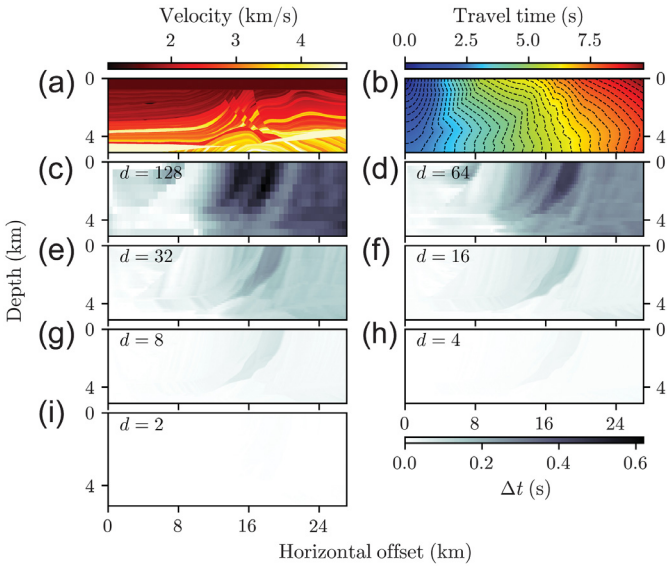

Figure 5. (a) The Marmousi2 velocity model; (b) travel-time field computed on a Cartesian grid with 6801 and 1401 nodes along the x and y axes, respectively, for a source located at the top left corner $(x, y) = (0, 0)$. Black dashed lines indicate wavefronts at 0.25 s intervals. (c–i) Difference, Δt, at coincident nodes between travel-time field in (b) and travel-time field computed on a grid decimated by a factor, d, specified in the top left corner of each plot.

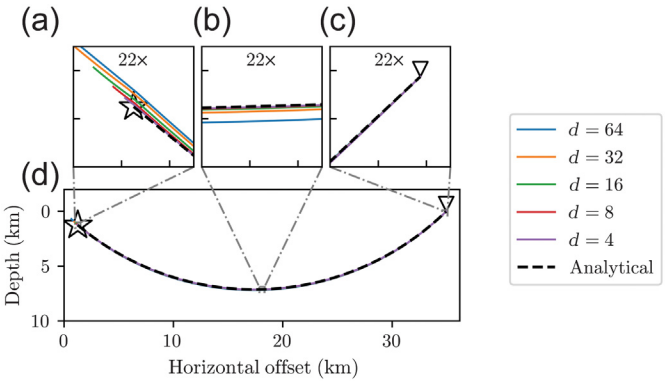

Figure 6. Rays traced through a medium with linear velocity gradient $v(z) = (4.5 + 0.25z)$(km/s) using different grid densities. The densest grid contains 4096 and 1024 nodes in the x and y directions, respectively, and the grid density is iteratively decimated by a factor of 2, in which d represents the decimation factor relative to the densest grid (e.g., the grid for $d = 8$ has $4096/8 = 512$ and $1024/8 = 128$ nodes in the x and y directions, respectively). Decimation factors corresponding to each ray are given in the legend at right. The analytical solution for the ray path is shown as a dashed black line. Black stars and inverted black triangles represent the source and receiver locations, respectively. (a–c) Zoomed in regions (22× magnification; axes ticks are at 0.2 km intervals) near the (a) source, (b) midpoint, and (c) receiver; (d) the entire ray path.

timescales nearly linearly with grid size (Table 1) making large problems ($>10^6$ nodes) tractable and small problems ($<10^6$ nodes) trivial.

Generality

In the Stability section, we demonstrated that our implementation of the FMM in Cartesian coordinates is applicable at local scales for which the curvature of the Earth is negligible. Here, we demonstrate that our implementation in spherical coordinates works well at global scales and the property of stable convergence holds. It is the exact same code base that solves the eikonal equation in both Cartesian and spherical coordinates, eliminating the need for multiple implementations—only the scaling factors in equation (10) change between operating modes, and this is done automatically at run time.

We repeat the exercise from Figure 5, but this time we use a 2D slice from the Li *et al.* (2008) global mantle P-wave model (MITP2008) in spherical coordinates. As in the case of Cartesian coordinates, the travel-time field converges to a stable solution as the node intervals decrease (Fig. 7).

Extensibility

Tracking secondary phases. The FMM is limited to tracking first arrivals, but secondary reflected arrivals contain much useful information for seismologists. These secondary arrivals can be tracked using the multistage approach developed by Rawlinson and Sambridge (2004b) and extended to 3D spherical coordinates by de Kool *et al.* (2006). We now show that our code can readily be extended to implement such a multistage approach for tracking secondary arrivals.

To show this, we track waves reflected from the lower boundary of a rectangular section (Fig. 8). First, we compute the

TABLE 1
Execution Time for Problems of Various Grid Sizes

Grid Size	Number of Nodes	Execution Time (s)
2 × 2 × 2	2^3	1.4×10^{-4}
4 × 4 × 4	2^6	1.1×10^{-4}
8 × 8 × 8	2^9	6.5×10^{-4}
16 × 16 × 16	2^{12}	2.8×10^{-3}
32 × 32 × 32	2^{15}	2.8×10^{-2}
64 × 64 × 64	2^{18}	2.5×10^{-1}
128 × 128 × 128	2^{21}	2.6
256 × 256 × 256	2^{24}	3.7×10^{1}
512 × 512 × 512	2^{27}	4.4×10^{2}

Simulations were run on a Dell Optiplex 9020 workstation with a quad-core 3.4 GHz Intel Core i7-4770 CPU.

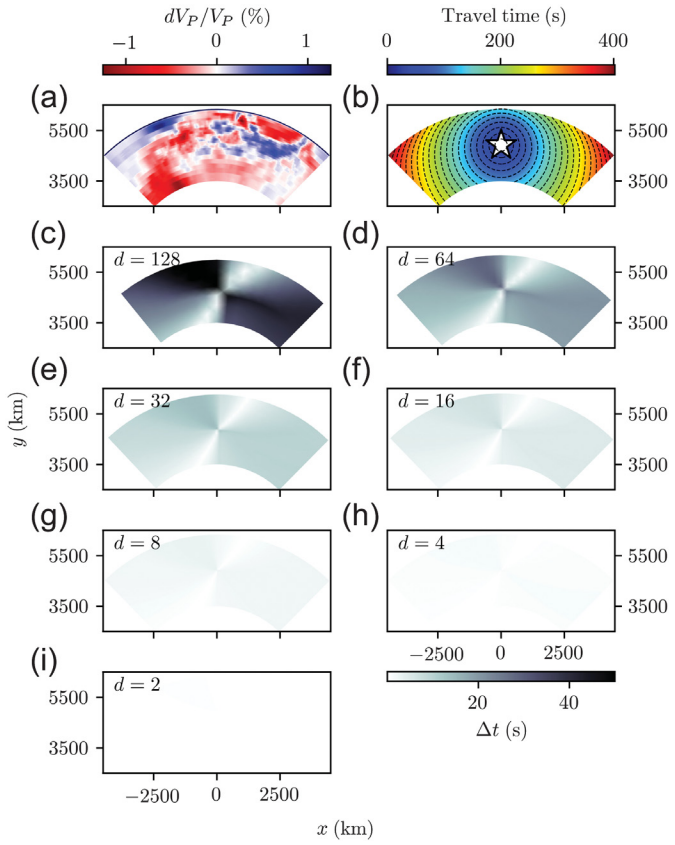

Figure 7. (a) A 2D slice of the MITP2008 velocity model shown as percent *P*-wave deviation from the ak135 reference model (Kennett *et al.*, 1995). (b) Travel-time field computed on a spherical grid with 1024 and 2048 nodes along the ρ and ϕ axes, respectively, for a source located at 1444 km depth. The white star with black outline and black dashed lines indicate the source location and wavefronts at 20 s intervals, respectively. (c–i) Difference, Δt, at coincident nodes between travel-time field in (b) and travel-time field computed on a grid decimated by a factor, d, specified in the top left corner of each plot.

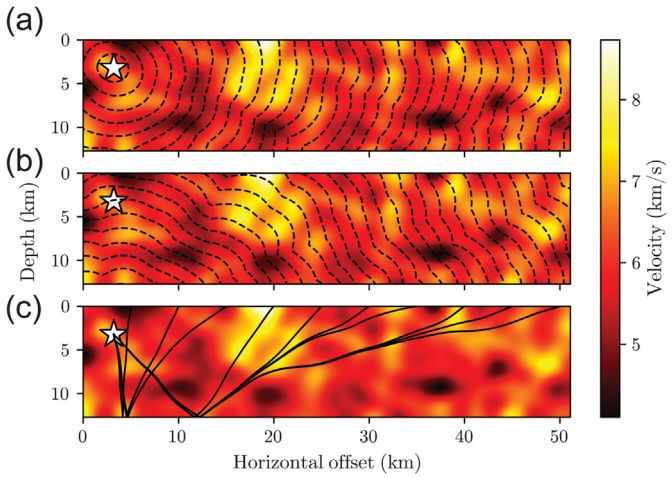

Figure 8. Reflected waves propagating through a velocity model with random velocity perturbations. The white star with black outline indicates the source location. (a,b) Black dashed lines represent wavefronts at 0.25 s intervals for (a) downgoing wavefronts and (b) wavefronts reflected by the lower boundary near 12 km depth. (c) Black lines represent ray paths reflected by the lower boundary and arriving at 5 km intervals at the surface.

downgoing travel-time field from the source to all points. Then, we reinitialize the travel-time field and set the travel time at each node along the bottom edge to the travel time of the incident downgoing wave. The upgoing reflected waves are then tracked to all points. This relatively simple example serves as a proof of concept: PyKonal can be extended to solve more complex problems than tracking first arrivals. Tracking multiple reflections and refractions from complex undulating surfaces, however, requires additional functionality that PyKonal does not currently offer. FM3D (Rawlinson and Sambridge, 2004b; de Kool *et al.*, 2006) is a more flexible and mature software package with respect to this class of problems.

Locating earthquakes. Seismic analysts locate earthquakes on a daily basis and typically use 1D velocity models to represent the 3D Earth structure, because simple algorithms that can be incorporated into relevant workflows are not available. Research seismologists are left to determine more accurate locations that account for 3D structure, which could be obtained in the first place with the necessary tools. As a final demonstration, we show that it is straightforward to extend the PyKonal algorithm to locate earthquakes.

To show this, we relocate a set of 148 well-constrained events in southern California, taken from the catalog of White *et al.* (2019), using a simple grid search followed by differential evolution optimization (Storn and Price, 1997) in a small neighborhood around the optimal grid point. Differential evolution is a genetic algorithm that performs a stochastic global search to optimize complex multivariate functions. Each of the chosen events is in the focus region of the study by White *et al.* (2019), is associated with at least 128 arrivals with root mean square (rms) residual <1 s, and was located using the NonLinLoc software package (Lomax *et al.*, 2009) for probabilistic nonlinear earthquake location in 3D heterogeneous Earth models and a 3D velocity model derived from the results of Fang *et al.* (2016).

To relocate events, we use the source–receiver reciprocity and compute travel-time fields for each receiver by treating them as sources. We use the same velocity model as in White *et al.* (2019) and a spherical grid with 64, 128, and 128 nodes in the radial, polar, and azimuthal directions, respectively. The origin time is estimated for each node-arrival pair by subtracting the node-to-station travel time from the observed arrival time, and the node that minimizes the standard deviation of origin time estimates for all arrivals provides an initial estimate of the hypocenter location. The average of

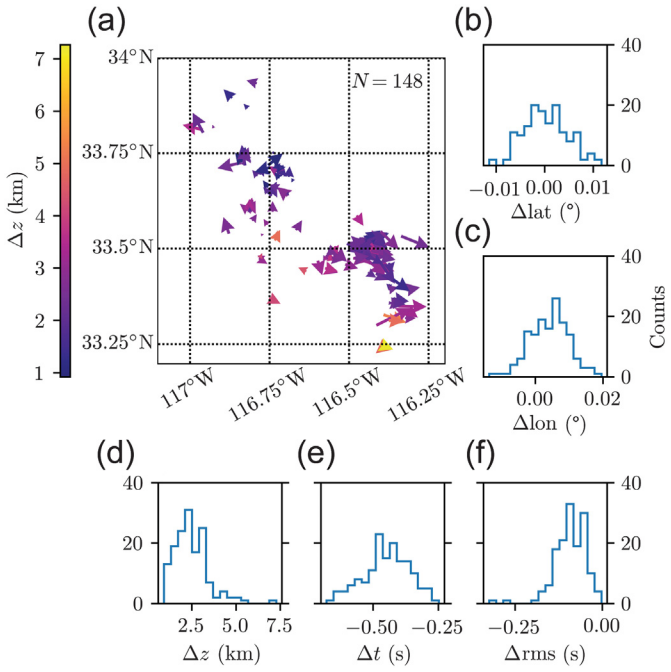

Figure 9. Comparing event locations obtained using NonLinLoc with locations obtained using the algorithm outlined in the Locating Earthquakes section. Changes in all quantities, ΔQ, are calculated as $\Delta Q = Q_{\text{NonLinLoc}} - Q_{\text{PyKonal}}$, in which $Q_{\text{NonLinLoc}}$ and Q_{PyKonal} are the quantities associated with NonLinLoc and PyKonal locations, respectively. (a) Map of displacement vectors pointing from locations computed using NonLinLoc to locations computed as earlier. (b–f) Histograms of location parameter changes after relocating events: (b) change in event latitude; (c) change in event longitude; (d) change in event depth; (e) change in event origin time; and (f) change in root mean square (rms) residual between observed and synthetic arrival times.

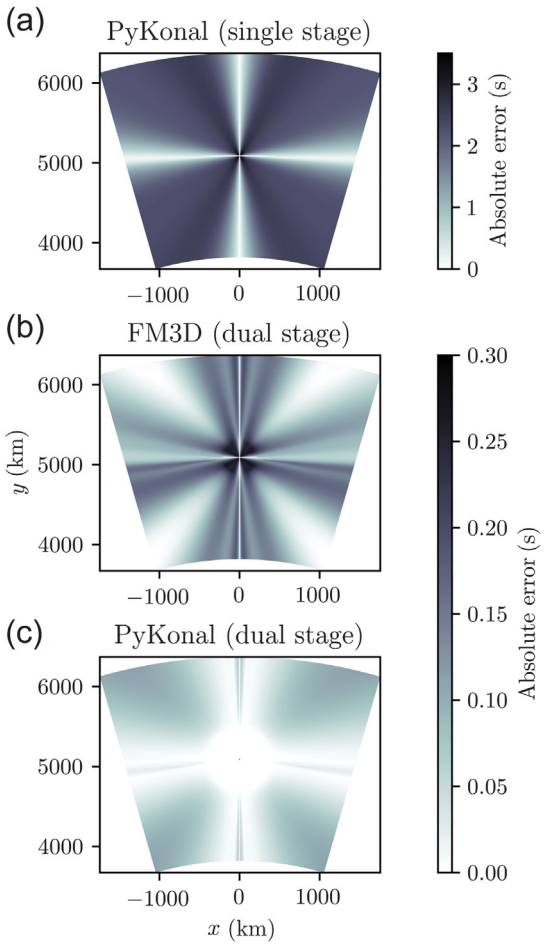

Figure 10. Absolute errors for travel-time fields computed through a homogeneous velocity model ($v = 1$ km/s) using three different methods: (a) PyKonal with a single coarse grid; (b) FM3D with a refined source grid and a coarse far-field grid; and (c) PyKonal with a refined source grid and coarse far-field grid.

the origin times estimated for the best-fitting node provides an initial origin time value. The differential evolution algorithm implemented by the scipy.optimize.differential_evolution function in the SciPy Python package (Jones *et al.*, 2001) then refines this initial location by searching a small region around the initial location for the value that minimizes the rms between observed and synthetic arrival times.

Relocated events are consistent with NonLinLoc locations (Fig. 9), but have lower rms in all 148 cases suggesting that this rudimentary algorithm consistently finds small adjustments that better fit the data. More sophisticated location algorithms can be designed using PyKonal to solve the forward problem.

Discussion

Comparing with FM3D

PyKonal builds on the foundation laid by FM3D (de Kool *et al.*, 2006) in four important ways: (a) it can improve accuracy using hybrid coordinate systems; (b) it accounts for the azimuthal periodicity inherent in spherical coordinates to allow wavefronts to propagate across the $\phi = 0$ plane, making many global-scale problems easier to solve; (c) it solves both 2D and 3D problems; and (d) it offers a simple Python application program interface (API) to accommodate diverse use cases.

Repeating the simple point-source error analysis from the Accuracy section for identical problems solved using FM3D and PyKonal demonstrates the improved accuracy achieved by PyKonal (Fig. 10). We consider a point source in homogeneous velocity structure and compare errors associated with FM3D and two different configurations of PyKonal. In all three cases, the computational grid comprises 256, 5, and 256 nodes along the radial, polar, and azimuthal axes, with 10 km, 0.125°, and 0.125° node intervals, respectively. Both FM3D and PyKonal can significantly reduce the error throughout the computational domain using a refined source grid. The refined

source grid for FM3D in this test spans 20 coarse-grid nodes along each coordinate axis, and the node intervals are decreased by a factor of 10; we chose these parameters because they provide a reasonable trade-off between execution time and accuracy. The refined source grid for PyKonal consists of a spherical grid centered on the source with a grid refinement factor of 5 and extending 40 coarse grid nodes in the radial direction. The accuracy of FM3D in this test approaches that of PyKonal in the region far from the source, but required over an order of magnitude more execution time (41.80 s compared to 1.12 s) to do so on our workstation (a Dell Optiplex 9020 workstation with a quad-core 3.4 GHz Intel Core i7-4770 CPU). Although PyKonal can achieve better accuracy with less execution time than FM3D, and is thus preferable in certain use cases, we note that FM3D offers more flexibility as it can track multiple reflection and transmission phases as well as teleseismic phases propagating through local 3D structure.

Wavefronts that cross over the $\phi = 0$ plane need to be carefully treated to solve problems at global scales. If the periodicity across this plane is neglected, the obtained solutions will, in the best-case scenario, only be valid in a hemisphere centered on the source. By properly accounting for this periodicity, our tool accommodates global-scale problems without any extra effort required of the user.

Python is a popular high-level interpreted programming language that suffers from slow execution as a result of the fact that it checks data types at run time. Languages that check data types during compilation, such as C, C++, and Fortran, execute quickly but are slow for developments. We leverage the high-level scripting capabilities of Python, while maintaining C-like speeds by producing compiled C extensions for Python using the Cython compiler framework. The code functions as any pure Python package would, but with the speed of C. This makes the presented tool user friendly while standing up to computationally demanding tasks.

Applications

A robust ray tracer has many applications; we highlight the four uses that we had in mind while developing this tool: locating earthquakes, body-wave tomography, computing takeoff angles, and Kirchhoff depth migration.

Locating earthquakes typically involves finding the set of spacetime coordinates that minimizes a misfit function between observed and synthetic arrival times. In seismically active fault zones where highly damaged rocks are juxtaposed against one or two different host rocks (e.g., Ben-Zion and Sammis, 2003; Fang *et al.*, 2016) and other areas where wavespeed varies significantly, 3D heterogeneities need to be considered to accurately locate events. In the Locating Earthquakes section, we showed that developing an inversion framework to locate earthquakes using PyKonal to solve the forward problem is straightforward. More sophisticated algorithms than the one presented here can be developed. One potential application of PyKonal is to incorporate it into a probabilistic location algorithm that uses a Markov chain Monte Carlo approach to include a priori information about the data and model. Such an algorithm may provide more reliable location estimates and associated uncertainties.

Similar to locating earthquakes, body-wave tomography typically involves minimizing a misfit function between observed and synthetic arrival times. In addition to the optimal spacetime coordinates of the sources, the optimal velocity structure is sought in body-wave tomography. In active-source surveys, for which the source locations are known, only the velocity model is sought. Again, being able to accurately synthesize travel times in the presence of 3D heterogeneity is crucial for deriving accurate velocity models.

Inverting first-motion polarity data for source mechanisms depends on the takeoff angles at which observed rays leave the focal sphere, and small differences in takeoff angles can produce significantly different responses at distant observation points. Because PyKonal can reduce travel-time errors in the near-source region (Fig. 10), more accurate ray paths, and thus takeoff angles, can be computed. More accurate takeoff angles should yield more accurate focal mechanisms and lead to better resolution of rupture kinematics.

Kirchhoff depth migration (Schneider, 1978) is a method for transforming seismic data from the time domain to the depth domain that finds wide use in seismic image processing as a means of resolving structural complexities by backpropagating scattered energy to the scatterer's position in the image. Migrating data in this way requires computing travel-time fields that account for 3D structural heterogeneity. PyKonal can be integrated easily into imaging workflows to facilitate clearer and more accurate images of subsurface structures.

Jupyter notebook examples and documentation

To promote reproducibility and provide examples of how to use the code, we include as supplemental material Jupyter notebooks that can be used to recreate Figures 3–8. We encourage interested readers to begin familiarizing themselves with the tool by reproducing the figures in this article. After running the example notebooks, users will benefit from API documentation, which is being actively developed (see Data and Resources).

Conclusions

PyKonal is a new open-source Python package for computing travel times and tracing ray paths in 2D and 3D based on the FMM for solving the eikonal equation (Sethian, 1996). It is designed to be easy enough for novice programmers to use and efficient enough to solve complex research problems in seismology without sacrificing generality or extensibility. The most recent release of the code can be downloaded from GitHub website (see Data and Resources). Users are encouraged to submit bug reports via GitHub or via email to the first author at malcolm.white@usc.edu.

We are now using PyKonal in a flexible tomographic method requiring minimal a priori information based on Fang *et al.* (2020), and are extending that method to include a nonlinear event relocation step similar to the one in the Locating Earthquakes section. We are applying the new methods to complementary data sets (Hutton *et al.*, 2010; White *et al.*, 2019) to derive integrated hierarchical images of the crust in southern California, with focus on the San Jacinto fault zone and region surrounding the 5 July 2019 M_w 7.1 Ridgecrest earthquake sequence.

Data and Resources

The MITP2008 velocity model used in this article is available via the cited reference (Li *et al.*, 2008). Figures 3–10 were made using Matplotlib (Hunter, 2007). The other relevant data are from the following sources: https://malcolmw.github.io/pykonal-docs and https://github.com/malcolmw/pykonal/releases. All websites were last accessed in May 2020. Supplemental material for this article includes six Jupyter Notebooks to recreate Figures 3–8.

Acknowledgments

The study was supported by the U.S. Department of Energy (Award DE-SC0016520). This article benefitted from useful comments by N. Rawlinson and an anonymous reviewer.

References

Afnimar, and K. Koketsu (2000). Finite difference traveltime calculation for head waves travelling along an irregular interface, *Geophys. J. Int.* **143,** no. 3, 729–734, doi: 10.1046/j.1365-246X.2000.00269.x.

Alkhalifah, T., and S. Fomel (2001). Implementing the fast marching eikonal solver: Spherical versus Cartesian coordinates, *Geophys. Prospect.* **49,** no. 2, 165–178, doi: 10.1046/j.1365-2478.2001.00245.x.

Ben-Zion, Y., and C. G. Sammis (2003). Characterization of fault zones, *Pure Appl. Geophys.* **160,** no. 3, 677–715, doi: 10.1007/PL00012554.

Buske, S., and U. Kastner (2004). Efficient and accurate computation of seismic travel-times and amplitudes, *Geophys. Prospect.* **52,** no. 4, 313–322, doi: 10.1111/j.1365-2478.2004.00417.x.

de Kool, M., N. Rawlinson, and M. Sambridge (2006). A practical grid-based method for tracking multiple refraction and reflection phases in three-dimensional heterogeneous media, *Geophys. J. Int.* **167,** no. 1, 253–270, doi: 10.1111/j.1365-246X.2006.03078.x.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs, *Numer. Math.* **1,** no. 1, 269–271, doi: 10.1007/BF01386390.

Fang, H., R. D. van der Hilst, M. V. de Hoop, K. Kothari, S. Gupta, and I. Dokmanić (2020). Parsimonious seismic tomography with Poisson Voronoi projections: Methodology and validation, *Seismol. Res. Lett.* **91,** no. 1, 343–355, doi: 10.1785/0220190141.

Fang, H., H. Zhang, H. Yao, A. Allam, D. Zigone, Y. Ben-Zion, C. Thurber, and R. van der Hilst (2016). A new algorithm for three-dimensional joint inversion of body wave and surface wave data and its application to the Southern California plate boundary region, *J. Geophys. Res.* **121,** no. 5, 3557–3569, doi: 10.1002/2015JB012702.

Harten, A., and S. Osher (1987). Uniformly high-order accurate non-oscillatory schemes. I, *SIAM J. Numer. Anal.* **24,** no. 2, 279–309, doi: 10.1137/0724022.

Hole, J. A., and B. C. Zelt (1995). 3-D finite-difference reflection travel times, *Geophys. J. Int.* **121,** no. 2, 427–434, doi: 10.1111/j.1365-246X.1995.tb05723.x.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment, *Comput. Sci. Eng.* **9,** no. 3, 90–95, doi: 10.1109/MCSE.2007.55.

Hutton, K., J. Woessner, and E. Hauksson (2010). Earthquake monitoring in Southern California for seventy-seven years (1932-2008), *Bull. Seismol. Soc. Am.* **100,** no. 2, 423–446, doi: 10.1785/0120090130.

Jones, E., T. Oliphant, and P. Peterson (2001). SciPy: Open source scientific tools for Python, Retrieved from http://www.scipy.org/ (last accessed May 2020).

Julian, B. R., and D. Gubbins (1977). Three-dimensional seismic ray tracing, *J. Geophys.* **43,** 95–114.

Kennett, B. L. N., E. R. Engdahl, and R. Buland (1995). Constraints on seismic velocities in the Earth from traveltimes, *Geophys. J. Int.* **122,** no. 1, 108–124, doi: 10.1111/j.1365-246X.1995.tb03540.x.

Kim, S., and R. Cook (1999). 3-D traveltime computation using second-order ENO scheme, *Geophysics* **64,** no. 6, 1867–1876, doi: 10.1190/1.1444693.

Li, C., R. D. van der Hilst, E. R. Engdahl, and S. Burdick (2008). A new global model for P wave speed variations in Earth's mantle, *Geochem. Geophys. Geosys.* **9,** no. 5, doi: 10.1029/2007GC001806.

Liu, X.-D., S. Osher, and T. Chan (1994). Multistep weighted essentially non-oscillatory schemes, *J. Comput. Phys.* **115,** no. 1, 200–212, doi: 10.1002/fld.3889.

Lomax, A., A. Michelini, and A. Curtis (2009). Earthquake location, direct, global-search methods, in *Encyclopedia of Complexity and Systems Science*, Springer New York, New York, New York, 2449–2473, doi: 10.1007/978-0-387-30440-3_150.

Osher, S., and J. A. Sethian (1988). Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations, *J. Comput. Phys.* **79,** no. 1, 12–49, doi: 10.1016/0021-9991(88)90002-2.

Podvin, P., and I. Lecomte (1991). Finite difference computation of traveltimes in very contrasted velocity models: A massively parallel approach and its associated tools, *Geophys. J. Int.* **105,** no. 1, 271–284, doi: 10.1111/j.1365-246X.1991.tb03461.x.

Qian, J., and W. W. Symes (2002a). An adaptive finite-difference method for traveltimes and amplitudes, *Geophysics* **67,** no. 1, 167–176, doi: 10.1190/1.1451472.

Qian, J., and W. W. Symes (2002b). Finite-difference quasi-P traveltimes for anisotropic media, *Geophysics* **67,** no. 1, 147–155, doi: 10.1190/1.1451438.

Qin, F., Y. Luo, K. B. Olsen, W. Cai, and G. T. Schuster (1992). Finite-difference solution of the eikonal equation along expanding wavefronts, *Geophysics* **57,** no. 3, 478–487, doi: 10.1190/1.1443263.

Rawlinson, N., and M. Sambridge (2004a). Multiple reflection and transmission phases in complex layered media using a multistage fast marching method, *Geophysics* **69,** no. 5, 1338–1350, doi: 10.1190/1.1801950.

Rawlinson, N., and M. Sambridge (2004b). Wave front evolution in strongly heterogeneous layered media using the fast marching method, *Geophys. J. Int.* **156,** no. 3, 631–647, doi: 10.1111/j.1365-246X.2004.02153.x.

Rawlinson, N., J. Hauser, and M. Sambridge (2008). Seismic ray tracing and wavefront tracking in laterally heterogeneous media, *Adv. Geophys.* **49,** no. 7, 203–273, doi: 10.1016/S0065-2687(07)49003-3.

Reshef, M., and D. Kosloff (1986). Migration of common-shot gathers, *Geophysics* **51,** no. 2, 324–331, doi: 10.1190/1.1442091.

Rouy, E., and A. Tourin (1992). A viscosity solutions approach to shape-from-shading, *SIAM J. Numer. Anal.* **29,** no. 3, 867–884, doi: 10.1137/0729053.

Schneider, W. A. (1978). Integral formulation for migration in two and three dimensions, *Geophysics* **43,** no. 1, 49–76, doi: 10.1190/1.1440828.

Schneider, W. A., K. A. Ranzinger, A. H. Balch, and C. Kruse (1992). A dynamic programming approach to first arrival traveltime computation in media with arbitrarily distributed velocities, *Geophysics* **57,** no. 1, 39–50, doi: 10.1190/1.1443187.

Sescu, A. (2015). Numerical anisotropy in finite differencing, *Adv. Differ. Equ.* **2015,** no. 1, 9, doi: 10.1186/s13662-014-0343-0.

Sethian, J. A. (1996). A fast marching level set method for monotonically advancing fronts, *Proc. Natl. Acad. Sci. Unit. States Am.* **93,** no. 4, 1591–1595, doi: 10.1073/pnas.93.4.1591.

Sethian, J. A. (1999). Fast marching methods, *SIAM Rev.* **41,** no. 2, 199–235, doi: 10.1137/S0036144598347059.

Sethian, J. A., and A. M. Popovici (1999). 3-D traveltime computation using the fast marching method, *Geophysics* **64,** no. 2, 516–523, doi: 10.1190/1.1444558.

Storn, R., and K. Price (1997). Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.* **11,** no. 4, 341–359, doi: 10.1023/A:1008202821328.

Thurber, C. H., and W. L. Ellsworth (1980). Rapid solution of ray tracing problems in heterogeneous media, *Bull. Seismol. Soc. Am.* **70,** no. 4, 1137–1148.

Um, J., and C. Thurber (1987). A fast algorithm for two-point seismic ray tracing, *Bull. Seismol. Soc. Am.* **77,** no. 3, 972–986.

van Trier, J., and W. W. Symes (1991). Upwind finite-difference calculation of traveltimes, *Geophysics* **56,** no. 6, 812–821, doi: 10.1190/1.1443099.

Versteeg, R. (1994). The Marmousi experience: Velocity model determination on a synthetic complex data set, *The Leading Edge* **13,** no. 9, 927–936, doi: 10.1190/1.1437051.

Vidale, J. E. (1988). Finite-difference calculation of travel times, *Bull. Seismol. Soc. Am.* **78,** no. 6, 2062–2076.

Vidale, J. E. (1990). Finite-difference calculation of traveltimes in three dimensions, *Geophysics* **55,** no. 5, 521–526, doi: 10.1190/1.1442863.

White, D. J. (1989). Two-dimensional seismic refraction tomography, *Geophys. J. Int.* **97,** no. 2, 223–245, doi: 10.1111/j.1365-246X.1989.tb00498.x.

White, M. C. A., Y. Ben-Zion, and F. L. Vernon (2019). A detailed earthquake catalog for the San Jacinto fault-zone region in Southern California, *J. Geophys. Res.* **124,** 6908–6930, doi: 10.1029/2019JB017641.

Zhao, H. (2004). A fast sweeping method for Eikonal equations, *Math. Comput.* **74,** no. 250, 603–628, doi: 10.1090/S0025-5718-04-01678-3.

Manuscript received 17 October 2019
Published online 3 June 2020