An Adaptive ENO Algorithm for the Eikonal Equation

Christopher D. Belfi, William W. Symes[*]

ABSTRACT

We have developed a cool adaptive grid algorithm that achieves major efficiency and accuracy gains over current methods for computing the point source travel time field.

Key words. travel times, eikonal equation, essentiallly nonoscillatory schemes

INTRODUCTION

Computation of traveltime fields is an important aspect of reflection seismology. Marching formulations of the eikonal equation have led to eikonal solvers which produce satisfactory traveltime computations. Furthermore, higher order eikonal solvers have been shown in (Belfi, 1997) to produce great efficiency and accuracy gains over first order methods. In the case of the point source problem, however, these higher order methods were shown to give no advantage. In fact, we observed first order convergence to the point source travel time fields, regardles of the formal order of accuracy. The difficulty arises in the fact that the solution to the point source problem is not differentiable at the source. Thus, the Taylor series expansions from which the higher order method derived their accuracy do not even exist. In order to address this problem, we propose an adaptive grid method using the second and third order ENO schemes presented in (Belfi, 1997), as building blocks. This algorithm results in major effeciency and accuracy gains compared to the currently used fixed grid methods. We present computational test results, and briefly describe the usage of the software we have created implementing this adaptive grid algorthm.

ALGORITHM DESIGN

The higher order ENO schemes use the locally smoothest (ie least oscillatory) interpolating polynomial when approximating the derivative. Our higher order schemes are also aperture limited, meaning that we restrict our attention to rays which travel at an angle of at most θ_{max} with the z-axis. Thus only these rays are computed accurately. This

[*]Department of Computational and Applied Mathematics, Rice University, Houston, TX 77251–1892

restriction is necessary in order to satisfy a CFL condition for stability. The aperture limitation directly influences the maximum stable step in the z direction.

The adaptive grid algorithm proceeds in two stages. In the first stage, we create a smooth initial condition at a small distance below the source point, by assuming a constant slowness over this small region. The question, then, is how accurate is this approximation. The accuracy can be estimated by analyzing the polar co-ordinate version of the eikonal equation and its power series expansion in the radial direction. (The details may be found in the appendix.) Thus we see that given the aperture, and an estimate of gradient of the slowness at the source, we can control the error (within the aperture) of the initialization step.

The second stage adjusts the grid at each depth to control the (global) error of the computation. Since we have both a second and third order method for solving the eikonal equation, which are both based on RK ODE methods, it is a straight-forward extension to create an adaptive method analogous to standard ODE solvers like RKF23 (Lambert, 1991), using a similar step size criterion. The local error of the second order scheme is estimated by using the output of the third order scheme as a substitute for the true solution, and this local error estimate is then the basis for controling the step size. Because the time field is changing rapidly near the source, the algorithm starts with a very fine grid. The step sizes then increase as the solution is advanced away from the source and becomes smoother. The end result is that we do no more work than necessary on the parts of the field which have smooth solutions, and still accurately compute less smooth section of the time field.

To avoid interpolation of the time data, and thus maintain computational efficiency, we elected in this first implementation to upsample the grid. The standard upsampling step is to throwout every other grid point. This and other design features of the algorithm result in a "sticky" tendency. One can often decrease the error tolerance somewhat, and so obtain a more accurate solution, without seeing any significant increase in computation time. This "sticky" behavior" will be evident in the experiments reported in the next section.

COMPUTATIONAL RESULTS

This algorithm has been tested on several different slowness fields. In the tables below, h refers to the user specified step size in the x direction, and ϵ refers to the user specified error control constant. The first test is a constant slowness field ($s = 1$), with a source at (0,-0.1) and a domain of $(-1.0, 1.0) \times (-1.0., 2.5)$ with a first arrival times computed at depth $z = 2.5$, using an aperture angle of 60 degrees. The errors listed are relative errors, with the exact solution being computed analystically. The cpu times listed are the results of running our code on a Sparc Ultra.

Fixed grid on constant slowness field:

h	error	cpu time (s)
0.1	.01683	0.17
0.05	.00824	0.56
0.025	.004077	2.33
0.0125	.00203	8.34
0.00625	.001017	33.06
0.003125	.0005298	130.07
0.0015625	.000347	518.59

Adaptive grid on constant slowness field:

ϵ	error	cpu time (s)
0.1	.00379	0.19
0.05	.00100	0.40
0.025	.000173	3.23
0.0125	.000144	11.55
0.00625	.000115	11.63
0.003125	.0000491	11.80
0.0015625	.0000174	46.06

The second test is a linear slowness field ($s = 1 + z$) on $(-1.0, 1.0) \times (-0.5, 2.5)$, with the source at (0,-0.1). Again the travel times were computed to the depth $z = 2.5$, using an aperture angle of 60 degrees. These errors are relative errors, with the "true" solution estimated by the difference between two fine grid computations. Again, the cpu times are from tests run on a Sparc Ultra.

Fixed grid on linear slowness field:

h	error	cpu time (s)
0.1	.0149	0.15
0.05	.00735	0.57
0.025	.00365	2.08
0.0125	.00182	8.33
0.00625	.000911	33.34
0.003125	.000456	132.36

Adaptive grid on linear slowness field:

ϵ	error	cpu time (s)
0.1	.00182	0.24
0.05	.000382	1.03
0.025	.000219	19.40
0.0125	.000193	89.17

PACKAGE USAGE

In order to use the travel time solver, you must create an input file in a simple parameter table format. example file: "example.par"

```
slowness="slow.sep"
traveltime="time.sep"
zs     =  1.0
xs     =  0.5
theta  =  60
tol    =  0.01
debug  =  0
```

The slowness and travel time output files refer to header files for SEP format data. Refer to the SGF section of the Hilbert Class Library documentation for details. We also require that z will be the first dimension in these SEP grids. The coordinates of the source point are zs and xs, and theta is the aperture angle in degrees. Tol is an error control constant which is roughly proportional to the error. Decreasing tol will increase the accuracy of the computation. Debug is a flag telling the program whether or not to print extra information. (The value 0 suppresses the output.) Finally, in order to run the program, simply type:

```
ttadapt example.par
```

CONCLUSION

Our adaptive grid algorithm for computing the travel time field of a point source is significantly more efficient than the similar fixed grid schemes. Although we are dealing with a PDE, the relatively small size of these computations allow us to take advantage of the methodology developed for solving ODEs, which in other problems is simply not an option. Furthermore, there is still room for even more efficiency gains. Since the finest grids are necessary near the source point, we can eliminate even more compuation by only computing the solution within the aperture. This would lead to far less computation on the finest grids, because there at small distances from the source there are few grid points within the aperture. Thus at the cost of extra coding, we may further reduce the computational expense of this method.

Acknowledgement and Disclaimer: This work was partially supported by the National Science Foundation under grant number DMS 9627355; the Office of Naval Research under grant number N00014-96-1-0156; the Air Force Office of Scientific Research, Air Force Materials Command, USAF, under grant number F49620-95-1-0339; and The Rice Inversion Project. TRIP Sponsors for 1998 are Amerada Hess, ARCO E & P Technology, Amoco Production Co., Conoco Inc., Exxon Production Research Co., Interactive Network Technologies, Landmark Graphics, Mobil Research and Development Corp., and Shell International Research. The US Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the Office of Naval Research, the National Science Foundation, or the US Government.

REFERENCES

Belfi, C. D. (1997). Second and third order eno methods for the eikonal equation. Technical report, The Rice Inversion Project, Rice University, Houston, Texas, USA.

Lambert, J. D. (1991). *Numerical Methods for Ordinary Differential Systems*. John Wiley and Sons.

APPENDIX

In our initialization stage, we approximate the solution near the source point by assuming a constant slowness field. Thus for the travel time we use the estimate:

$$\hat{\tau}(r,\theta) = r\tau_1(\theta) \tag{1}$$

and so we wish to analyze the difference between the estimated and true solutions. The power series for τ in r is:

$$\tau(r,\theta) = r\tau_1(\theta) + \frac{r^2}{2}\tau_2(\theta) + ... \tag{2}$$

. and so the dominant term in the error is $\tau_2(\theta)$. The eikonal equation in polar coordinates is written:

$$\left(\frac{\partial\tau}{\partial r}\right)^2 + \frac{1}{r^2}\left(\frac{\partial\tau}{\partial\theta}\right)^2 = s^2 \tag{3}$$

and then take partial derivatives with respect to r and θ.

$$\frac{\partial\tau}{\partial r} = \tau_1(\theta) + r\tau_2(\theta) + ... \tag{4}$$

$$\frac{\partial\tau}{\partial\theta} = r\frac{\partial\tau_1}{\partial\theta} + \frac{r^2}{2}\frac{\partial\tau_2}{\partial\theta} + ... \tag{5}$$

Also, we can write out the power series for s in r as:

$$s(r,\theta) = s_0 + rs_1(\theta) + ... \tag{6}$$

Thus the power series expansion of the eikonal equation in r becomes:

$$s_0^2 + 2rs_0s_1(\theta) + ... = \tau_1(\theta)^2 + \left(\frac{\partial\tau_1}{\partial\theta}\right)^2 + r\left(2\tau_1(\theta)\tau_2(\theta) + \frac{\partial\tau_1}{\partial\theta}\frac{\partial\tau_2}{\partial\theta}\right) + ... \tag{7}$$

Equating first term coefficients gives us:

$$\tau_1(\theta)^2 + \left(\frac{\partial\tau_1}{\partial\theta}\right)^2 = s_0^2 \tag{8}$$

which can be rewritten as:

$$\frac{\partial \tau_1}{\partial \theta} = \pm\sqrt{s_0^2 - \tau_1(\theta)^2} \tag{9}$$

Since $\tau_1(\theta)$ must be 2π periodic, the only solution is $\tau_1(\theta) = s_0$ identically, ie $\frac{\partial \tau_1}{\partial \theta} = 0$ Equating second term coefficients give us:

$$2s_1(\theta)s_0 = 2\tau_1(\theta)\tau_2(\theta) + 0 \tag{10}$$

Substitution then gives $s_1(\theta) = \tau_2(\theta)$. So what is s_1 in rectangular coordinates?

$$s(z,x) = s_0 + \frac{\partial s}{\partial x}x + \frac{\partial s}{\partial z}z \tag{11}$$

which can be rewritten as:

$$s(z,x) = s_0 + \frac{\partial s}{\partial x}r\sin\theta + \frac{\partial s}{\partial z}r\cos\theta \tag{12}$$

which implies that $s_1(\theta) = \frac{\partial s}{\partial x}\sin\theta + \frac{\partial s}{\partial z}\cos\theta$ and thus $s_1(\theta) \leq |\nabla s|$. Because we are bounding the error inside the aperture only, we substitute $x \leq z\tan\theta_{max}$ to show that the dominant error term is bounded by:

$$\begin{aligned}
\frac{r^2}{2}|\nabla s| &= \frac{1}{2}(x^2 + z^2)|\nabla s| \\
&\leq \frac{1}{2}(1 + \tan^2\theta_{max})|z|^2|\nabla s| \\
&= \frac{|z|^2|\nabla s|}{\cos^2\theta_{max}}
\end{aligned}$$