$\mathcal{O}(N)$ OPTIMAL ALGORITHMS FOR THE EIKONAL EQUATION

Seongjai Kim

Department of Mathematics
University of Kentucky
skim@ms.uky.edu

INTRODUCTION

- **The Eikonal Equation**

$$|\nabla \tau(\mathbf{x})| = \frac{1}{v(\mathbf{x})}, \ \mathbf{x} \in \Omega$$
$$\tau(\mathbf{x}) = \tau_0, \qquad \mathbf{x} \in \Omega_0 \subset \Omega$$

  where v is the normal velocity

- **Difficulties**
  - Turning rays
  - Development of corners and discontinuities
  - Multi-valued solution

Eikonal Solvers

Depending on Marching:

- **Expanding-wavefront** algorithm:
  (e.g. Level set methods)

  + Stable
  – Basically, first-order
  – Data access: almost-random

- **Expanding-box** algorithm
  (e.g. Standard FD schemes)

  – Unstable for turning rays
  + Possibly, higher-order
  + Data access: systematic

OUTLINE

- **ENO Schemes** (Osher & Sethian, 1988, JCP)
- **Fast Marching Method (FMM)**
  (Tsitsiklis, 1995; Sethian, 1996)
  - Cost: $\mathcal{O}(N \log_2 N)$
- **Group Marching Method (GMM)**
  (Kim, 2001, SISC)
  - Cost: $\mathcal{O}(N)$
- **Expanding-Box Scheme**
  (Vidale, 1988, BSSA; Kim & Cook, 1999, Geophys.)
  - Cost: $\mathcal{O}(N)$
  - Accuracy: $\mathcal{O}(h^2)$
- **Numerical Experiments**
- **Conclusions & Future Work**

ENO DIFFERENCES

(Osher & Sethian, 1988, JCP)

For τ_x at $\mathbf{x}_{i,j}^{k}$

Define the forward & backward difference operators:

$$D_x^{\pm}\tau_{i,j}^{k} = \pm\frac{\tau_{i\pm1,j}^{k} - \tau_{i,j}^{k}}{\Delta x} = D_x^{\pm,1}\tau_{i,j}^{k}$$

Then, from Taylor expansion,

$$\tau_x = D_x^{\pm}\tau \mp \frac{1}{2}\Delta x \cdot \tau_{xx} + \mathcal{O}(\Delta x^2)$$

Now, approximate

$$\tau_{xx} \approx \mathrm{m}(D_x^{-}D_x^{+}\tau,\ D_x^{\pm}D_x^{\pm}\tau)$$

where

$$\mathrm{m}(a,b) = \begin{cases} 0, & \text{if } ab \le 0, \\ a, & \text{if } |a| \le |b| \text{ and } ab > 0, \\ b, & \text{if } |a| > |b| \text{ and } ab > 0. \end{cases}$$

Then, the ENO2 reads

$$\tau_x \approx D_x^{\pm,2}\tau \equiv D_x^{\pm,1}\tau \mp \frac{1}{2}\Delta x \cdot \mathrm{m}(D_x^{-}D_x^{+}\tau,\ D_x^{\pm}D_x^{\pm}\tau)$$

ENO2

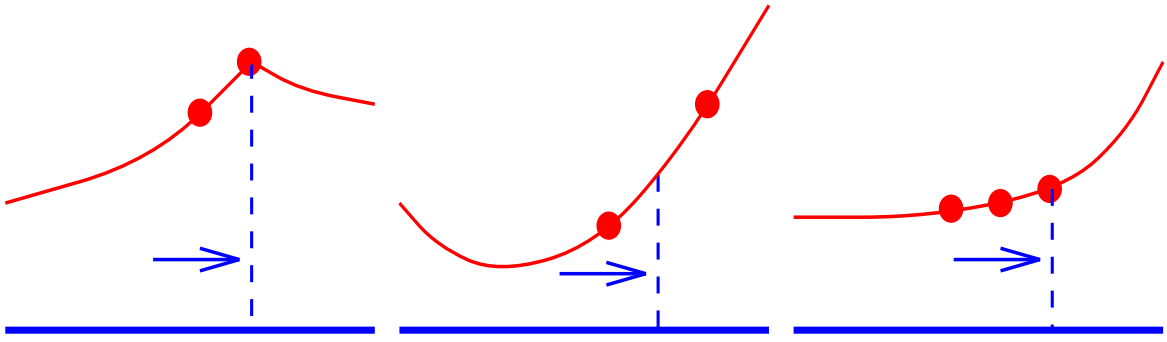
$$\tau_x \approx D_x^{\pm,2}\tau \equiv D_x^{\pm,1}\tau \mp \frac{1}{2}\Delta x \cdot \mathrm{m}(D_x^- D_x^+ \tau,\ D_x^\pm D_x^\pm \tau),$$

where

$$\mathrm{m}(a,b) = \begin{cases} 0, & \text{if } ab \le 0, \\ a, & \text{if } |a| \le |b| \text{ and } ab > 0, \\ b, & \text{if } |a| > |b| \text{ and } ab > 0. \end{cases}$$

The backward ENO2 can be explicitly written as

$$D_x^{-,2}\tau_{i,j}^k = \begin{cases} D_x^-\tau_{i,j}^k, \\ D_x^-\tau_{i,j}^k + \frac{1}{2}\Delta x \cdot D_x^- D_x^+ \tau_{i,j}^k = \dfrac{\tau_{i+1,j}^k - \tau_{i-1,j}^k}{2\Delta x}, \quad \text{or} \\ D_x^-\tau_{i,j}^k + \frac{1}{2}\Delta x \cdot D_x^- D_x^- \tau_{i,j}^k = \dfrac{1}{\Delta x}(\frac{3}{2}\tau_{i,j}^k - 2\tau_{i-1,j}^k + \frac{1}{2}\tau_{i-2,j}^k) \end{cases}$$



Choose the most smooth component!!

ENO3

Taylor expansion:

$$\tau_x = D_x^{\pm}\tau \mp \frac{\Delta x}{2}\tau_{xx} - \frac{\Delta x^2}{6}\tau_{xxx} + \mathcal{O}(\Delta x^3)$$

The third-order ENO difference for τ_x:

$$D_x^{\pm,3}\tau = D_x^{\pm,2}\tau - \frac{(\Delta x)^2}{6}\mathrm{m}_3(D_x^{\pm}D_x^{\pm}D_x^{\pm}\tau,\ D_x^{+}D_x^{+}D_x^{-}\tau,\ D_x^{+}D_x^{-}D_x^{-}\tau),$$

where

$$\mathrm{m}_3(a,b,c) = \mathrm{m}(\mathrm{m}(a,b),c)$$

Upwind ENO differences

$$\widehat{D}_x^{\ell}\tau_{i,j}^{k} = \text{mod\_max}(\ \max\left(D_x^{-,\ell}\tau_{i,j}^{k}, 0\right), \min\left(D_x^{+,\ell}\tau_{i,j}^{k}, 0\right)),$$

where $\ell = 1, 2, 3$, and
"mod_max" returns maximum in modulus.

Note that

$$|\widehat{D}_x^{\ell}\tau_{i,j}^{k}| = \max\left(D_x^{-,\ell}\tau_{i,j}^{k},\ -D_x^{+,\ell}\tau_{i,j}^{k},\ 0\right), \quad \ell = 1, 2, 3$$

The upwind ENO algorithm

$$(\widehat{D}_x^{\ell}\tau_{i,j}^k)^2 + (\widehat{D}_y^{\ell}\tau_{i,j}^k)^2 + (\widehat{D}_z^{\ell}\tau_{i,j}^k)^2 = (s_{i,j}^k)^2 \qquad (1)$$

For down-going wavefronts (expanding-box):

- First-order algorithm:

$$\tau_{i,j}^{k+1} = \tau_{i,j}^k + \Delta z \cdot H^1[\tau]_{i,j}^k,$$

- Second-order (Runge-Kutta) algorithm:

$$\begin{aligned}
\tau_{i,j}^{k+\frac{1}{2}} &= \tau_{i,j}^k + \Delta z_{\mathrm{CFL}} \cdot H^2[\tau]_{i,j}^k, \\
\tau_{i,j}^{k+1} &= \frac{1}{2}(\tau_{i,j}^k + \tau_{i,j}^{k+\frac{1}{2}} + \Delta z_{\mathrm{CFL}} \cdot H^2[\tau]_{i,j}^{k+1/2}),
\end{aligned}$$

where

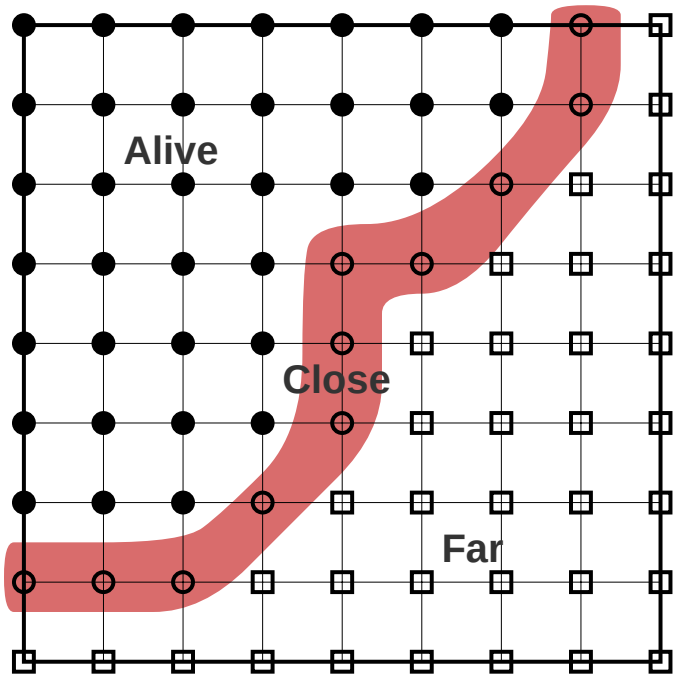
$$H^{\ell}[\tau]_{i,j}^k = \sqrt{(s_{i,j}^k)^2 - (\widehat{D}_x^{\ell}\tau_{i,j}^k)^2 - (\widehat{D}_y^{\ell}\tau_{i,j}^k)^2}, \quad \ell = 1, 2$$

FAST MARCHING METHOD (FMM)

(Tsitsiklis, 1995; Sethian, 1996)

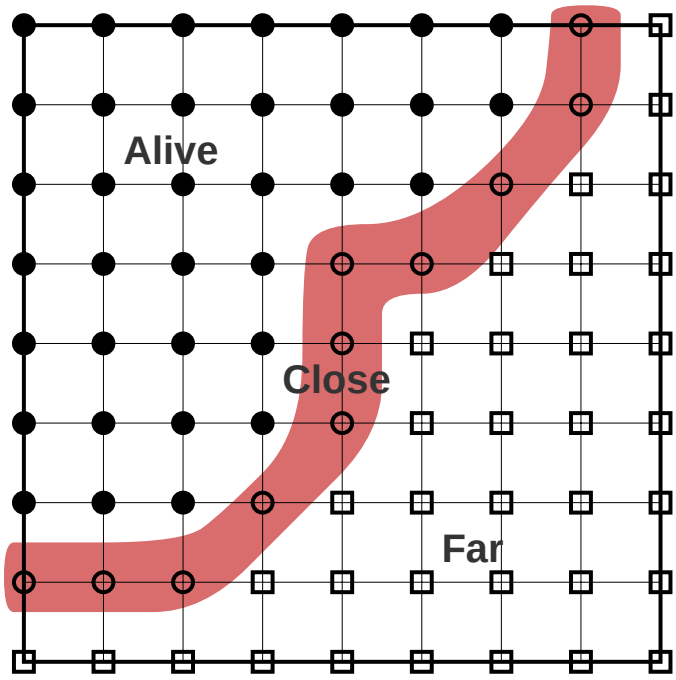
Narrow band approach:

- Narrow band forms a neighbor of wavefronts
- Update the solution from the smallest value (to satisfy **causality**; Huygens's principle)



FMM: Pseudo-code

1. Let *Trial* be the point in *Close* having the smallest traveltime.
2. Tag as *Close* all neighbors of *Trial* that are not *Alive*. (If the neighbor is in *Far*, remove it from the list and add it to the set *Close*.)
3. Recompute the traveltimes at all *Close* neighbors of *Trial* using (1).
4. Remove the point *Trial* from *Close* and add it to *Alive*.
5. If *Close* is not empty, go to top of loop.



Remarks on FMM

- Step 1: requires *sorting*
  - The binary tree sorting costs $\mathcal{O}(\log_2 N)$
  - The total cost for FMM: $\mathcal{O}(N \log_2 N)$

- Step 3: *updates* the solution by recomputation

  Let $\tau_{i,j}^k = \tau(\mathbf{x}_{i,j}^k)$ be to be updated. Then,

  1. Compute real solutions from 8 neighboring cells, e.g.,

$$\left(\frac{\tau - \tau_{i-1,j}^k}{\Delta x}\right)^2 + \left(\frac{\tau - \tau_{i,j-1}^k}{\Delta y}\right)^2 + \left(\frac{\tau - \tau_{i,j}^{k-1}}{\Delta z}\right)^2 = \left(s_{ij}^k\right)^2$$

     (For non-available values, drop the corresponding term.)

  2. Choose the smallest one for $\tau_{i,j}^k$

GROUP MARCHING METHOD (GMM)

(Kim, 2001)

Object: *to drop sorting & march a group*

- Use the "narrow band" approach as in FMM
- Observations:
  - All local minima can be updated at a time
  - Solutions of a small difference cannot affect each other in the update procedure

If $|\tau^{k_1}_{i_1,j_1} - \tau^{k_2}_{i_2,j_2}| \le \delta\tau,\ \ \delta\tau = \frac{1}{\sqrt{3}} \cdot h \cdot s_{\min}$, the wavefront normal is rather perpendicular to $\overline{\mathbf{x}^{k_1}_{i_1,j_1}\mathbf{x}^{k_2}_{i_2,j_2}}$ than parallel.

GMM: The choice of group

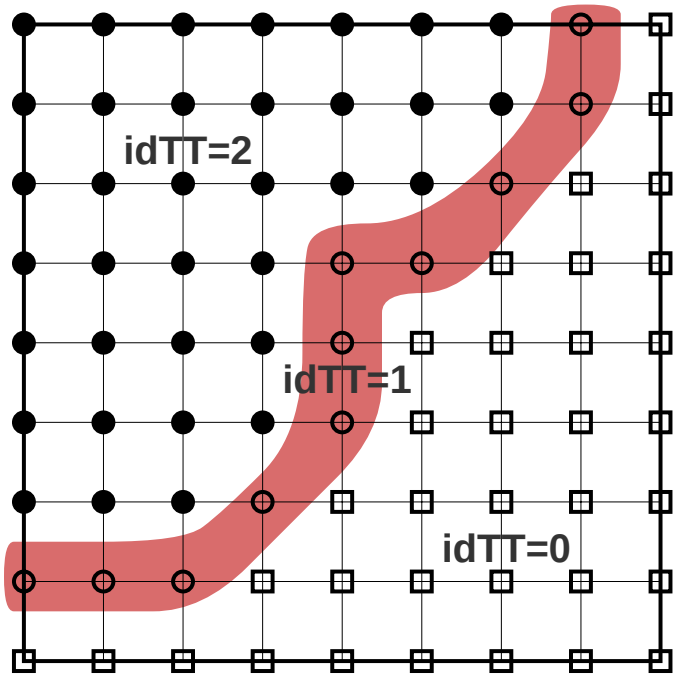
$$G = \{\mathbf{x}_{ij}^k \in \Gamma \,:\, \tau(\mathbf{x}_{ij}^k) \leq \tau_{\Gamma,\min} + \delta\tau\},$$

where

Γ is the narrow band

$$\tau_{\Gamma,\min} = \min\{\tau_{ij}^k : \mathbf{x}_{ij}^k \in \Gamma\}$$

$$\delta\tau = \frac{1}{\sqrt{3}} \cdot h \cdot s_{\Gamma,\min}$$



GMM: Pseudo-code

- Initialization:

  (I1) Assign a large number to **the solution** `TT`, e.g., $\texttt{TT}(\cdot,\cdot,\cdot) \equiv 1.0e5$;

  (I2) Set zero for the **solution index** `idTT`, i.e., $\texttt{idTT}(\cdot,\cdot,\cdot) \equiv 0$;

  (I3) Set $\texttt{delTAU} = \frac{1}{\sqrt{3}} \cdot \min(\Delta x, \Delta y, \Delta z) \cdot \min_{i,j,k} s_{i,j}^k$;

  (I4) On the box of $(2 \times 2 \times 2)$ cells having the source at its center,
  - assign the analytic value of `TT` on the box;
  - set $\texttt{idTT}(\cdot,\cdot,\cdot) = 2$, at the source;
  - set $\texttt{idTT}(\cdot,\cdot,\cdot) \equiv 1$, on the surface of the box;
    save those point indices to the **interface indicator** $\texttt{GAMMA}(\cdot,\cdot,\cdot)$;
  - set `TM` to be the minimum of `TT` on the surface of the box;

- Marching Forward:

(M1) Set $\texttt{TM} = \texttt{TM} + \texttt{delTAU}$;

(M2) For each (i,j,k) in `GAMMA`, in the reverse order, if ($\texttt{TT}(i,j,k) \le \texttt{TM}$), update neighboring points (ℓ,m,n) where $\texttt{idTT} \le 1$;

(M3) For each (i,j,k) in `GAMMA`, in the forward order, if ($\texttt{TT}(i,j,k) \le \texttt{TM}$),

  (a) update neighboring points (ℓ,m,n) where $\texttt{idTT} \le 1$;

  (b) if $\texttt{idTT} = 0$ at a neighboring point (ℓ,m,n), set $\texttt{idTT}(\ell,m,n) = 1$ and save (ℓ,m,n) into `GAMMA`;

  (c) remove the index (i,j,k) out of `GAMMA`; set $\texttt{idTT}(i,j,k) = 2$;

(M4) If $\texttt{GAMMA} \neq \emptyset$, go to (M1);

Remarks on GMM

- GMM converges in two iterations;
  it is between FMM and the pure iteration
  - FMM: $\delta\tau \to 0$ (Tsitsiklis, 1995; Sethian, 1996)
  - Iteration: $\delta\tau = \infty$
    (Rouy & Tourin, 1992; Schneider et al., 1992)

- Double-computation in (M2) & (M3): But the cost is slightly higher than single-computaion

EXPANDING-BOX SCHEME

The basic algorithm: ENO2+RK2

$$\begin{aligned} \tau_{i,j}^{k+\frac{1}{2}} &= \tau_{i,j}^{k} + \Delta z_{\mathrm{CFL}} \cdot H[\tau]_{i,j}^{k}, \\ \tau_{i,j}^{k+1} &= \frac{1}{2}(\tau_{i,j}^{k} + \tau_{i,j}^{k+\frac{1}{2}} + \Delta z_{\mathrm{CFL}} \cdot H[\tau]_{i,j}^{k+1/2}), \end{aligned} \tag{2}$$

where

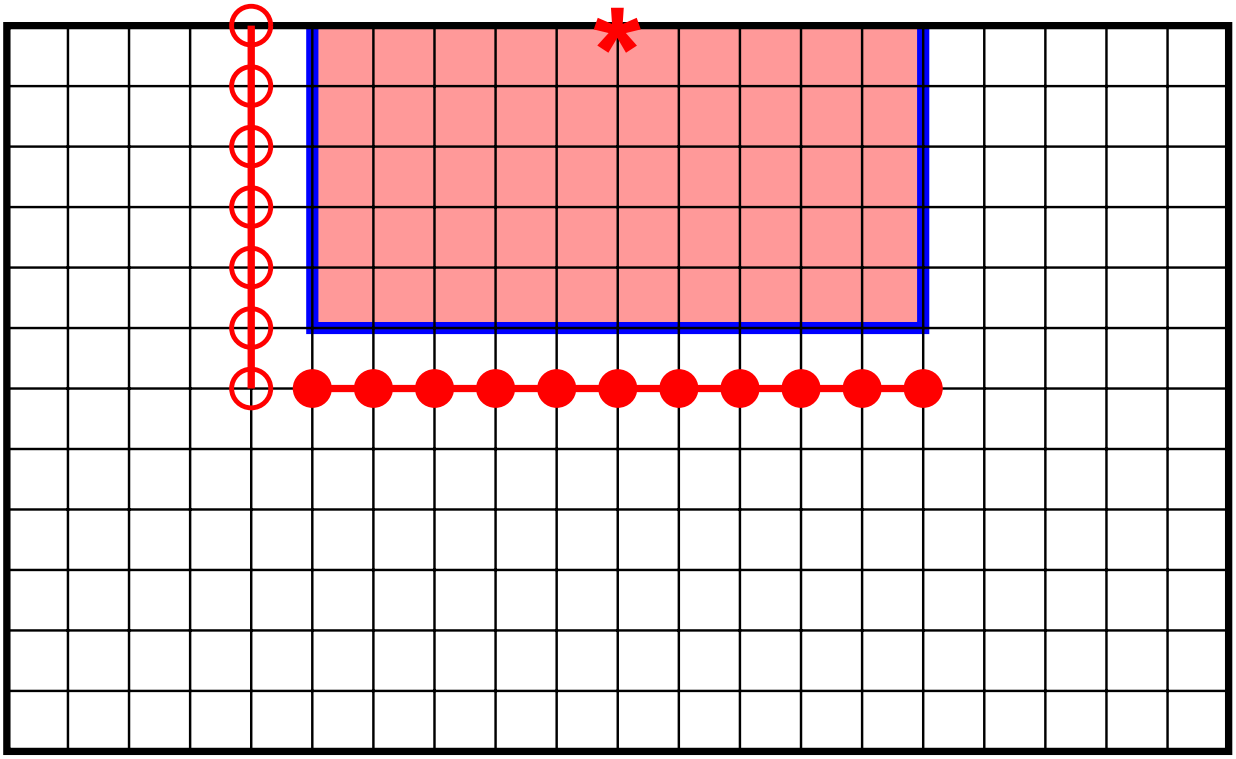
$$H[\tau]_{i,j}^{k} = \sqrt{(s_{i,j}^{k})^2 - (\widehat{D}_x^2 \tau_{i,j}^{k})^2 - (\widehat{D}_y^2 \tau_{i,j}^{k})^2}$$

– Unstable for turning rays

\+ Possibly, higher-order

\+ Data access: systematic

DNO (Down-'N'-Out)



PS (Post Sweeping)

Set $\tau^{(0,0)}$ = the solution of the DNO marching;

Set DIR(:)="$(x+), (x-), (y+), (y-), (z+), (z-)$";

For $\ell = 0, 1, \cdots$

For $i = 1, 2, \cdots, 6$

get $\tau^{(\ell,i)}$ by marching in the DIR(i)-direction

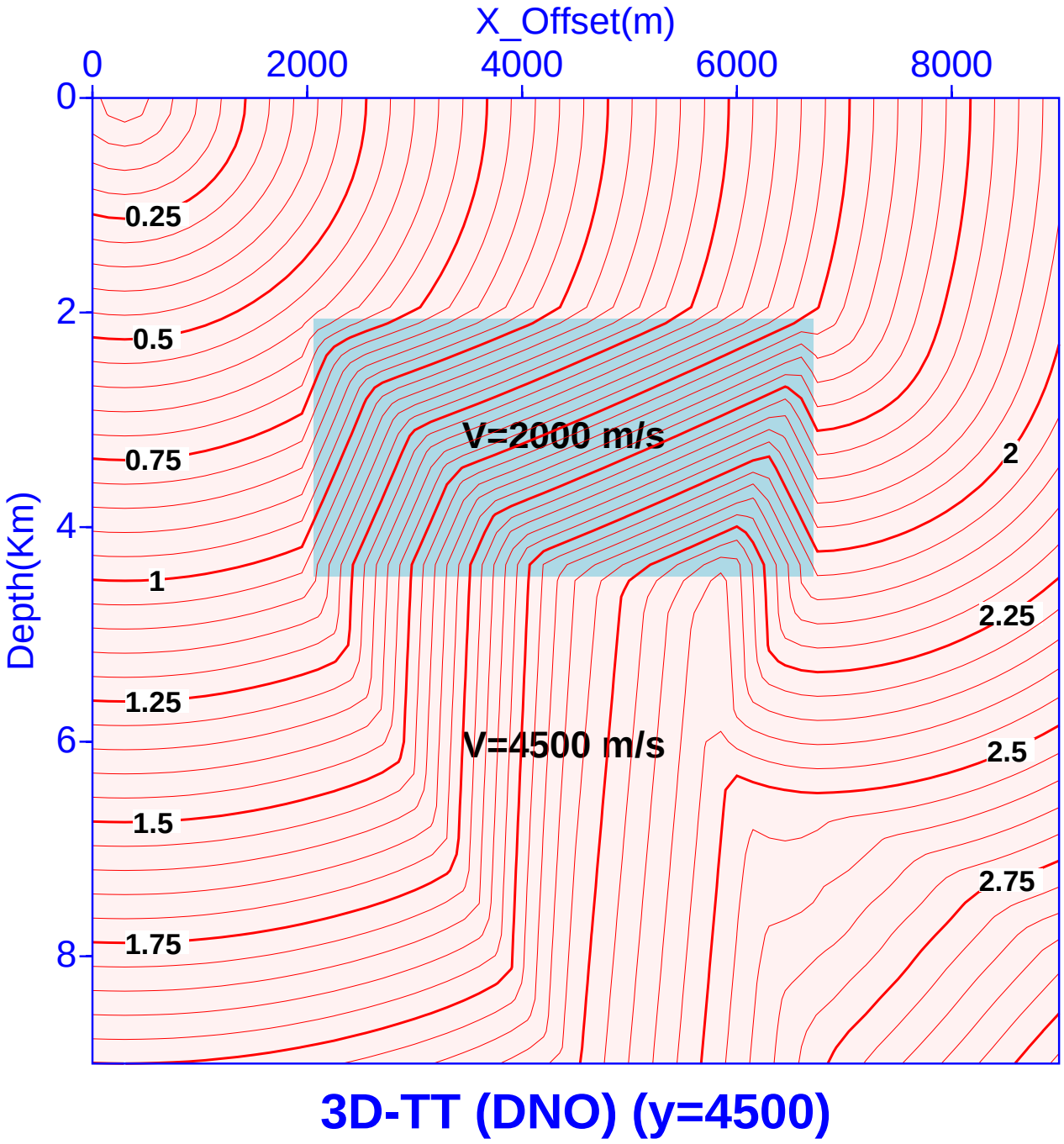
update $\tau^{(\ell,i)} = \min(\tau^{(\ell,i)}, \tau^{(\ell,i-1)})$;

End for

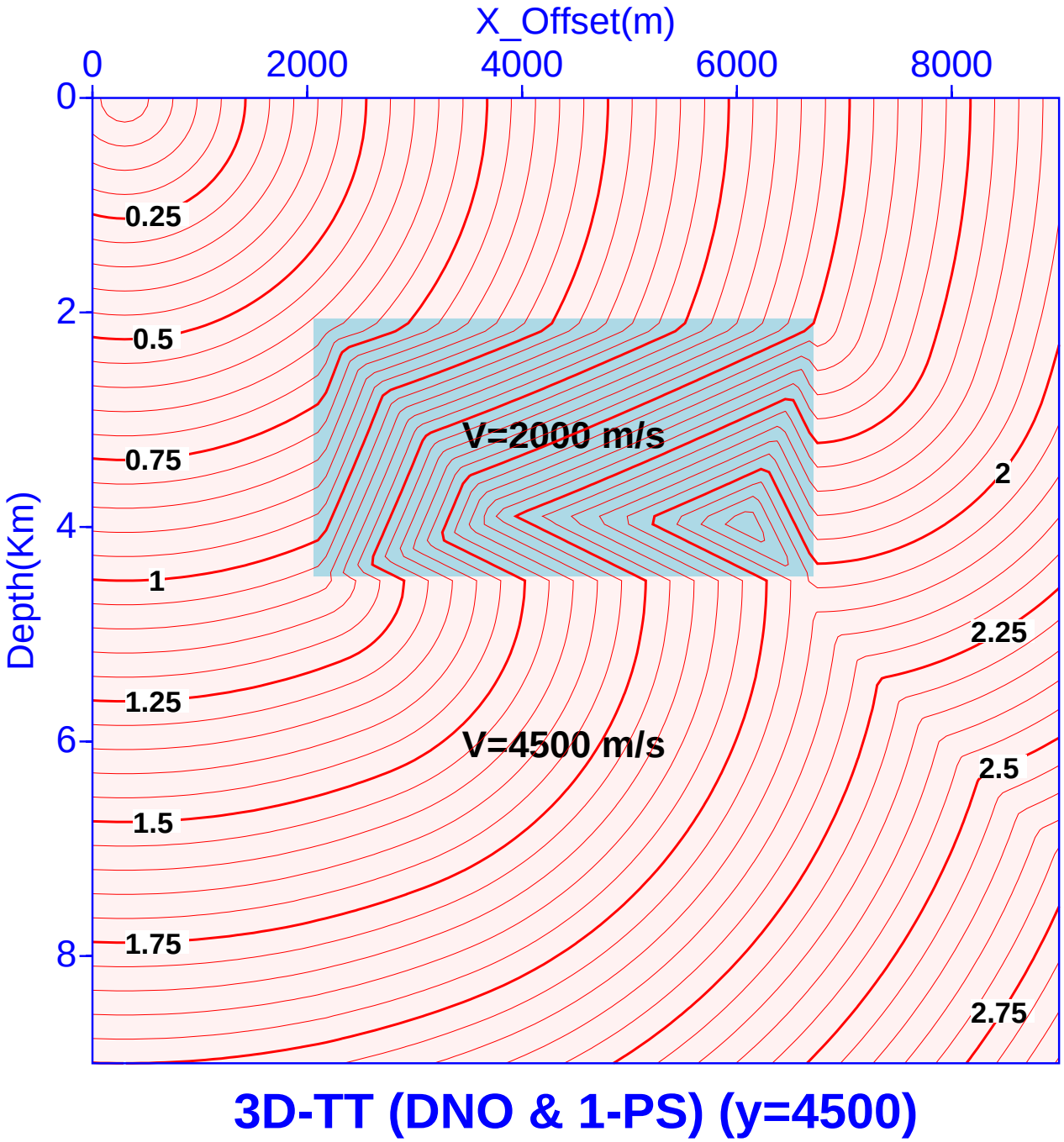
Set $\tau^{(\ell+1,0)} = \tau^{(\ell,6)}$;

If $|\tau^{(\ell+1,0)} - \tau^{(\ell,0)}| \leq \varepsilon$, stop;

End for



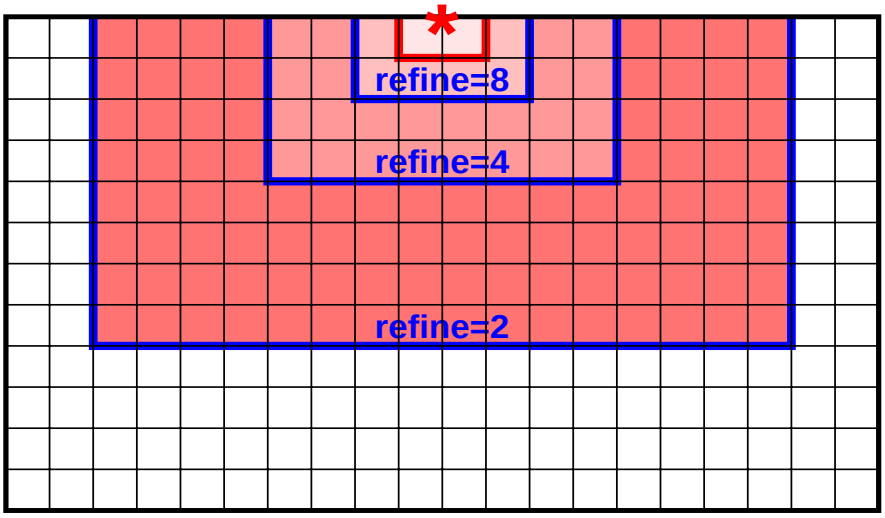
3D-TT (DNO) (y=4500)



3D-TT (DNO & 1-PS) (y=4500)

ACCURACY & EFFICIENCY ISSUES

- Locally Uniform Mesh Refinement (LUMR)



  - near the source
  - or, wherever desired

- Average Normal Velocity
  - High-order schemes require an average velocity

- Maximum Angle Condition (expanding-box)

$$\tau_z = \sqrt{\max(s^2 \cos^2 \theta_{\max},\ s^2 - \tau_x^2 - \tau_y^2)}$$

- Cache-Based Implementation (expanding-box)

NUMERICAL EXPERIMENTS

Implement

- FMM (ENO1, expanding-wavefront)
- GMM (ENO1, expanding-wavefront)
- ENO-DNO-PS (ENO2, expanding-box)

Problem setting

$\Omega = (0, 6000\,\mathrm{m})^3$

Grid size: $h = \Delta x = \Delta y = \Delta z = 6000/n$

$$v_1(\mathbf{x}) = \begin{cases} 4500\ \mathrm{m/s}, & \text{if } \mathbf{x} \in [1500, 4500]^3 \\ 2000\ \mathrm{m/s}, & \text{else} \end{cases}$$

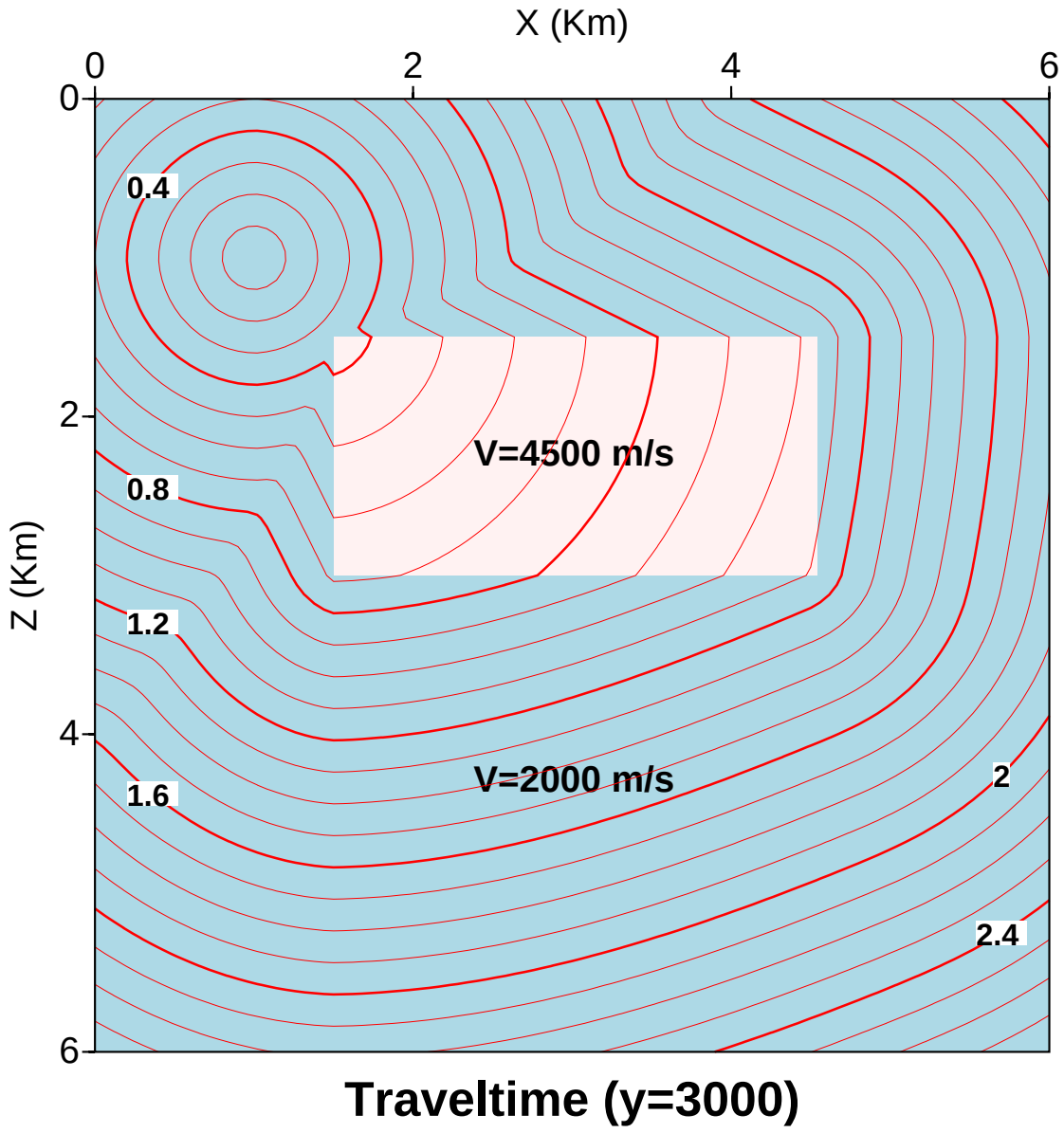
$$v_2(\mathbf{x}) = 1000 + 0.3x + 0.2y + 0.4z\ \mathrm{m/s}$$

$$v_3(\mathbf{x}) = 1000 + 0.2x + 0.2y + 0.5z\ \mathrm{m/s}$$

Computer: 266 MHz laptop

$v = v_1$

- $h = 60\text{m}$ (Problem size: 1M)



$v = v_2$

- $\mathbf{x}_s = (3000, 3000, 1000)$
- Error $= 1000 * \|\tau^h - \tau_{\text{analytic}}\|_\infty$ (ms)

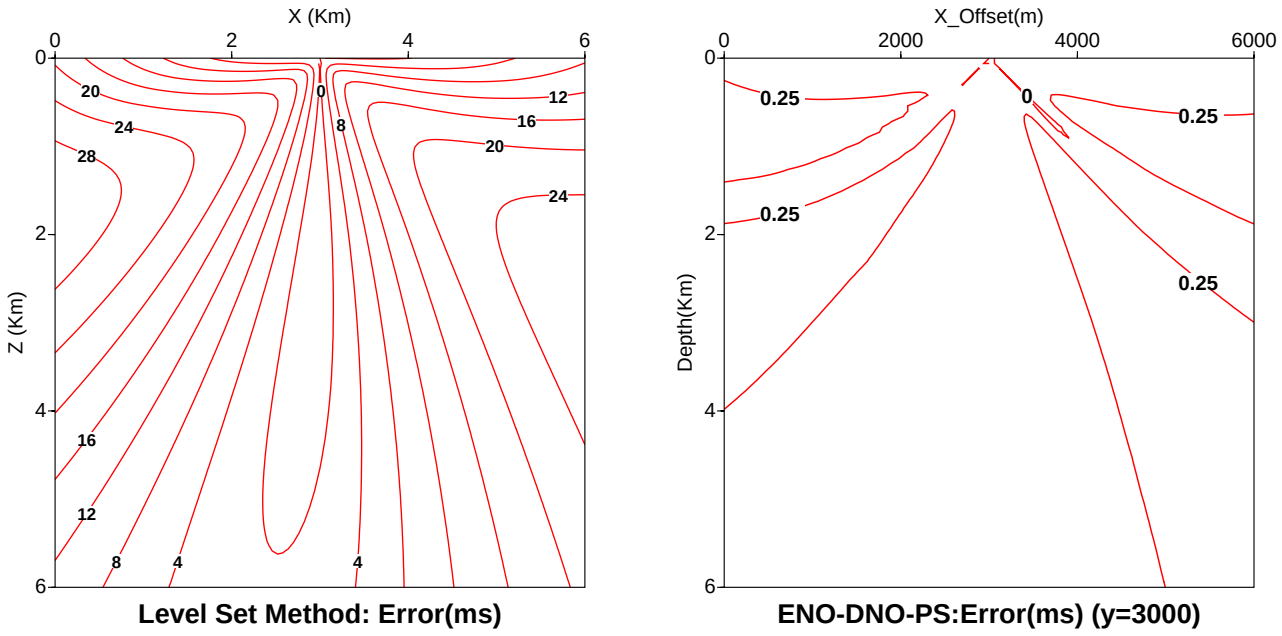
n^3	FMM		GMM	
	CPU	$E(\tau^h)$	CPU	$E(\tau^h)$
30^3	1.37	120	0.98	120
60^3	19.81	62	7.88	62
120^3	395.8	32	64.06	32

- GMM costs $\mathcal{O}(N)$
- GMM is one-order faster, for realistic accuracy!!

$v = v_3$

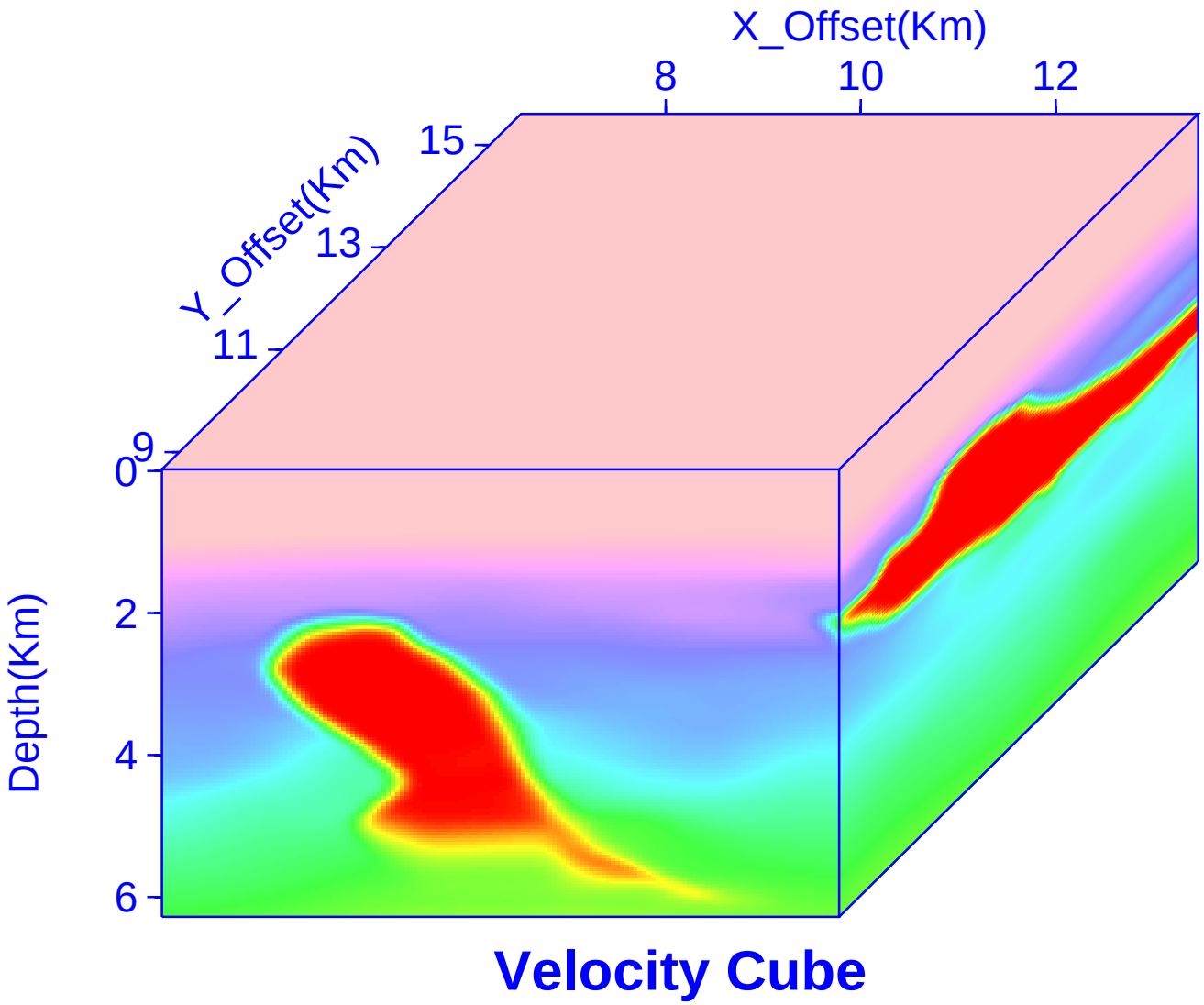
- $h = 60$m (Problem size: 1M)
- Error $= 1000 * \|\tau^h - \tau_{\text{analytic}}\|_\infty$ (ms)

GMM		ENO-DNO-PS	
CPU	Error	CPU	Error
38.5s	**59ms**	**(8.6+9.7)s**	**0.58ms**

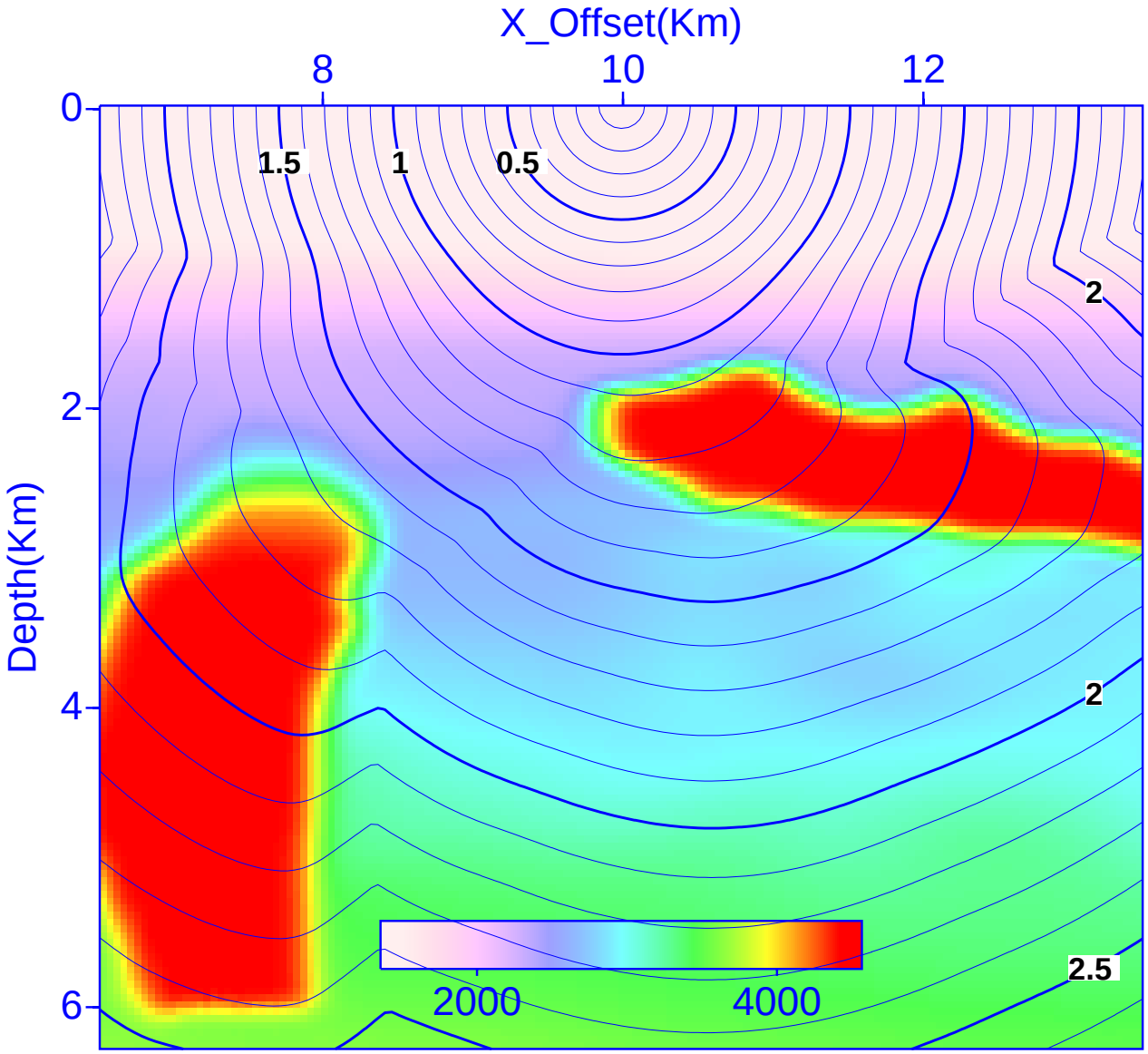


The errors (in millisecond)

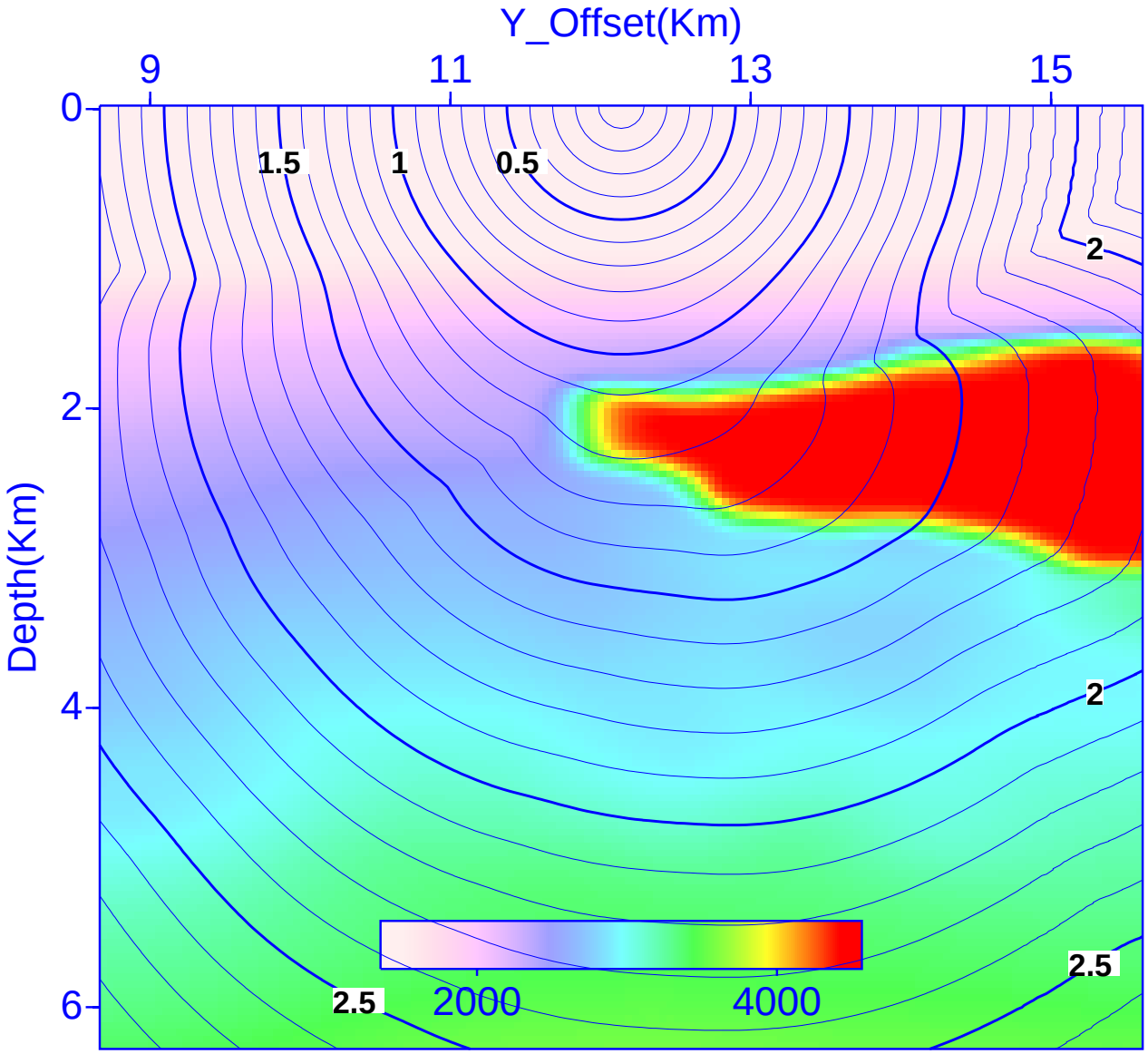
Real Model



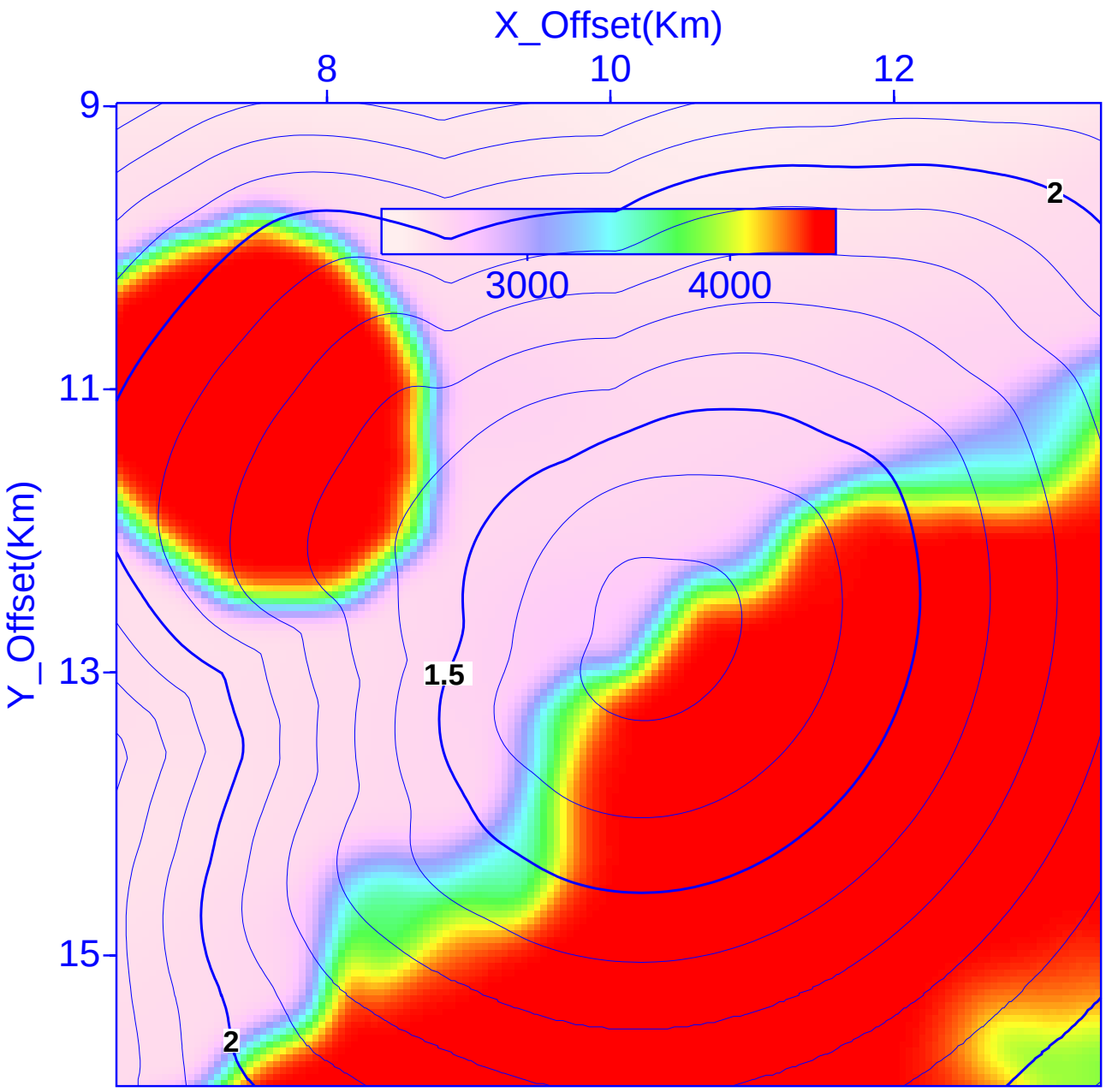
$150 \times 150 \times 136$ cells (> 3M grid points)



Velocity & 3D-TT (y=12116m)



Velocity & 3D-TT (x=9967m)



Velocity & 3D-TT (z=2606m)

CONCLUSIONS

- **FMM** (ENO1, expanding-wavefront):
  - Cost: $\mathcal{O}(N \log_2 N)$
- **GMM** (ENO1, expanding-wavefront):
  - Cost: $\mathcal{O}(N)$
- **ENO-DNO-PS** (ENO2, expanding-box):
  - Cost: $\mathcal{O}(N)$
  - DNO-PS: introduced for enforcing stability
  - Applications to anisotropic wavefronts
- **Comparisons**:
  - GMM is one-order faster than FMM (in 3D)
  - Yet, $GMM \ll ENO\text{-}DNO\text{-}PS$

FUTURE WORK

- **Accuracy for Level Set Methods**
  - Possible: $\mathcal{O}(h^2)$?
- **Multi-valued Solutions**
  - Preferred: $\mathcal{O}(N)$, $\mathcal{O}(h^{2+\alpha})$, and in **3.5D**
- **Most-Energetic Traveltime**
  - Preferred: $\mathcal{O}(N)$, $\mathcal{O}(h^{2+\alpha})$, and in **3D**
- **Applications**
  - Seismology (up to date)
  - Other physical problems (??)