Parallel reformulation of the sequential adjoint-state method

Bas Peters*, Tristan van Leeuwen# & Felix J. Herrmann*

*Seismic Laboratory for Imaging and Modeling (SLIM), University of British Columbia
Mathematical Institute, Utrecht University.

SLIM

University of British Columbia

Computational cost of Full-Waveform Inversion

Focus is on computational cost in turnaround **time**.

Not about:

- faster PDE solves
- optimization algorithms w/ a better rate of convergence
- better implementations & hardware/software interaction

Computational cost of Full-Waveform Inversion

This talk reformulates the basic algorithms

- mathematically equivalent method
- achieves factor 2X speedup in time for fixed resources
- requires source/receiver randomization & subsampling
- requires stochastic optimization
- only useful in parallel computing environment
- exploit special saddle-point structures

Function value and gradient

$$f(\mathbf{m}) = \frac{1}{2}\sum_{i=1}^{n_{\mathrm{src}}} \|PA(\mathbf{m})^{-1}\mathbf{q}_i - \mathbf{d}_i\|_2^2$$

$A(\mathbf{m}) \in \mathbb{C}^{N\times N}$ discrete PDE

$\mathbf{m} \in \mathbb{R}^N$ medium parameters

$P \in \mathbb{R}^{m\times N}$ selects field at receivers

$\mathbf{u} \in \mathbb{C}^N$ field

$\mathbf{d} \in \mathbb{C}^m$ observed data

$\mathbf{q} \in \mathbb{C}^N$ source

Function value and gradient – adjoint-state

$$f(\mathbf{m}) = \frac{1}{2}\sum_{i=1}^{n_{\mathrm{src}}} \|PA(\mathbf{m})^{-1}\mathbf{q}_i - \mathbf{d}_i\|_2^2$$

Algorithm 1 The conventional sequential adjoint-state algorithm to compute **g**.

1. $\mathbf{u}_i = A(\mathbf{m})^{-1}\mathbf{q}_i$ `//forward solve`
2. $\mathbf{v}_i = A(\mathbf{m})^{-*}(P^*(P\mathbf{u}_i - \mathbf{d}_i))$ `//adjoint solve`
3. $\mathbf{g}_i = \left(\frac{\partial A(\mathbf{m})\mathbf{u}_i}{\partial \mathbf{m}}\right)^* \mathbf{v}_i$ `//evaluate gradient`
4. $\mathbf{g} = \sum_{i=1}^{n_s} \mathbf{g}_i$ `//sum gradient components`

Function value and gradient – adjoint-state

Adjoint-state:

- parallel over sources & frequencies
- **2 sequential** PDE solves for each source/frequency

Limiting factor if:

- a large number of available compute nodes
- a small number of sources (stochastic optimization, simultaneous sources)

Proposed algorithm – parallel reformulation

Same objective:

$$f(\mathbf{m}) = \frac{1}{2}\sum_{i=1}^{n_{\mathrm{src}}} \|PA(\mathbf{m})^{-1}\mathbf{q}_i - \mathbf{d}_i\|_2^2$$

w/o derivation:

Algorithm 2 The parallel reformulation to compute $\mathbf{g}$.

1. $\mathbf{u}_i = A(\mathbf{m})^{-1}\mathbf{q}$ & $\boxed{W = A(\mathbf{m})^{-*}P^*}$ `//solve in parallel`
2. $\mathbf{v}_i = -W(P\mathbf{u}_i - \mathbf{d}_i))$ `//evaluate adjoint`
3. $\mathbf{g}_i = \left(\frac{\partial A(\mathbf{m})\mathbf{u}_i}{\partial \mathbf{m}}\right)^* \mathbf{v}_i$ `//evaluate gradient`
4. $\mathbf{g} = \sum_{i=1}^{n_s} \mathbf{g}_i$ `//sum gradient components`

Proposed algorithm – parallel reformulation

Same objective:

$$f(\mathbf{m}) = \frac{1}{2}\sum_{i=1}^{n_{\mathrm{src}}} \|PA(\mathbf{m})^{-1}\mathbf{q}_i - \mathbf{d}_i\|_2^2$$

w/o derivation:

Algorithm 2 The parallel reformulation to compute $\mathbf{g}$.

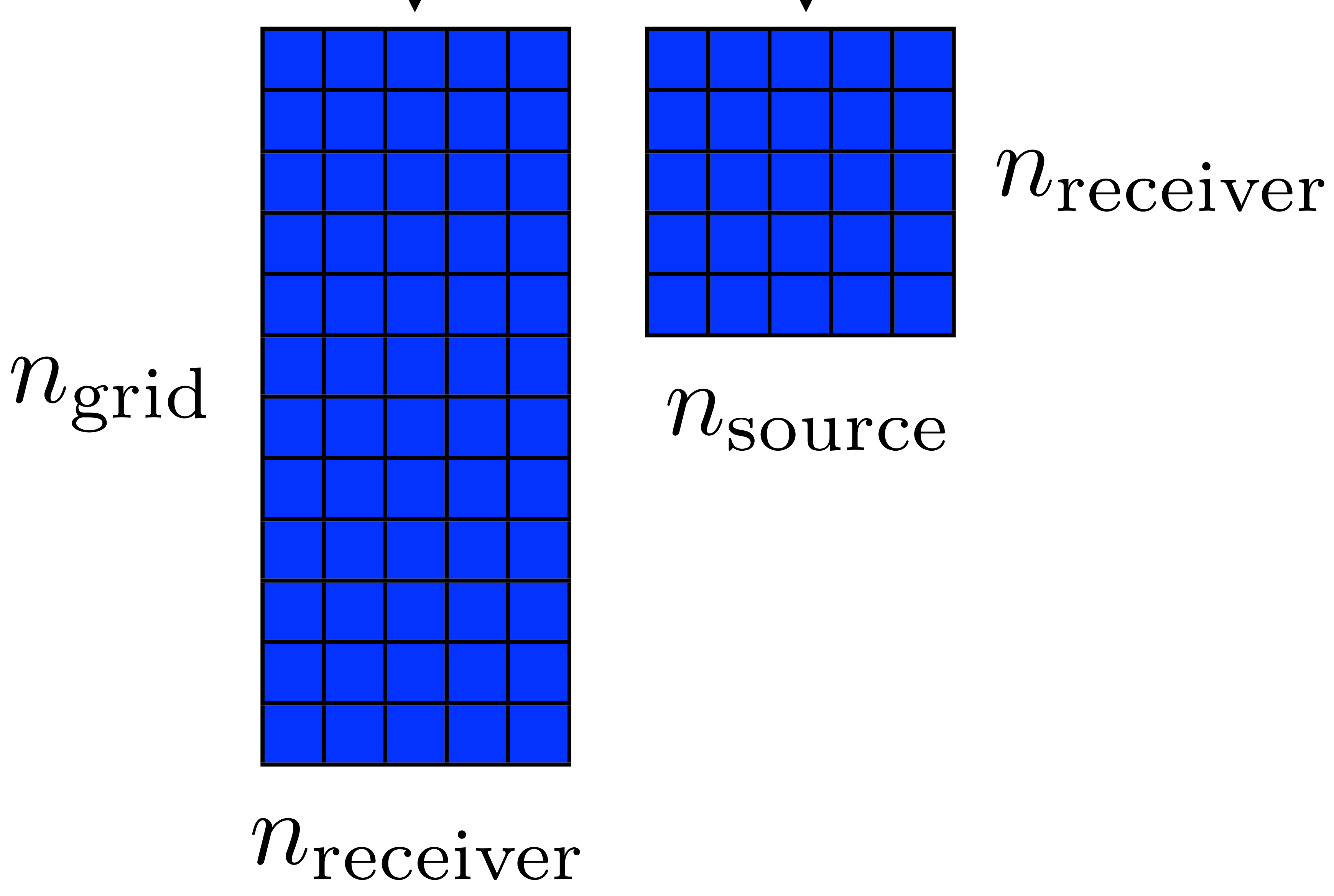
1. $\mathbf{u}_i = A(\mathbf{m})^{-1}\mathbf{q}$ & $W = A(\mathbf{m})^{-*}P^*$ `//solve in parallel`
2. $\mathbf{v}_i = -W(P\mathbf{u}_i - \mathbf{d}_i))$ `//evaluate adjoint`
3. $\mathbf{g}_i = \left(\frac{\partial A(\mathbf{m})\mathbf{u}_i}{\partial \mathbf{m}}\right)^* \mathbf{v}_i$ `//evaluate gradient`
4. $\mathbf{g} = \sum_{i=1}^{n_s} \mathbf{g}_i$ `//sum gradient components`

Proposed algorithm – parallel reformulation

- **2 PDE solves in parallel**
- solve 1 PDE per sources & 1 per receiver
- 1 distributed matrix (W) - local vector product to evaluate adjoint

Distributed computations

$$\mathbf{v} = \underbrace{-W}\underbrace{(P\mathbf{u} - \mathbf{d})}$$



- each column is computed & stored on a different node
- multiple columns per node if block-solvers are used
- result is also distributed

Proposed algorithm – derivation

Original problem: $\min_{\mathbf{m},\mathbf{u}} \frac{1}{2}\|P\mathbf{u}-\mathbf{d}\|_2^2 \quad \text{s.t.} \quad A(\mathbf{m})\mathbf{u}=\mathbf{q}$

Lagrangian: $L(\mathbf{m},\mathbf{u},\mathbf{v}) = \frac{1}{2}\|P\mathbf{u}-\mathbf{d}\|_2^2 + \mathbf{v}^*\big(A(\mathbf{m})\mathbf{u}-\mathbf{q}\big)$

Characterize the sub-problem for fixed $\mathbf{m}$ at every iteration:

(for one frequency)

$$\begin{pmatrix} P^*P & A^* \\ A & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \end{pmatrix} = \begin{pmatrix} P^*\mathbf{d} \\ \mathbf{q} \end{pmatrix}$$

Proposed algorithm – derivation

Any solution method for solving

$$\begin{pmatrix} P^*P & A^* \\ A & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \end{pmatrix} = \begin{pmatrix} P^*\mathbf{d} \\ \mathbf{q} \end{pmatrix}$$

results in the $\mathbf{u}$ & $\mathbf{v}$ required to compute objective & gradient of

$$f(\mathbf{m}) = \frac{1}{2} \sum_{i=1}^{n_{\mathrm{src}}} \| PA(\mathbf{m})^{-1}\mathbf{q}_i - \mathbf{d}_i \|_2^2$$

Proposed algorithm – derivation

$$\begin{pmatrix} P^*P & A^* \\ A & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \end{pmatrix} = \begin{pmatrix} P^*\mathbf{d} \\ \mathbf{q} \end{pmatrix}$$

- square saddle point system
- full-rank if A is full rank (satisfied by assumption)
- P^*P is square & rank-k, k=number of receivers

Remark:

- block elimination = discrete adjoint-state method

Proposed algorithm – derivation

$$\begin{pmatrix} P^*P & A^* \\ A & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & A^{-1} \\ A^{-*} & -A^{-*}P^*PA^{-1} \end{pmatrix}$$

$$\mathbf{u} = A^{-1}\mathbf{q}$$

$$\mathbf{v} = -A^{-*}P^*PA^{-1}\mathbf{q} + A^{-*}P^*\mathbf{d} = -\underbrace{A^{-*}P^*}_{\equiv W}\left(P\mathbf{u} - \mathbf{d}\right)$$

Computations become independent if we compute $W = A^{-*}P^*$

Proposed algorithm

Algorithm 2 The parallel reformulation to compute $\mathbf{g}$.

1. $\mathbf{u}_i = A(\mathbf{m})^{-1}\mathbf{q} \;\&\; W = A(\mathbf{m})^{-*}P^*$ `//solve in parallel`
2. $\mathbf{v}_i = -W(P\mathbf{u}_i - \mathbf{d}_i))$ `//evaluate adjoint`
3. $\mathbf{g}_i = \left(\frac{\partial A(\mathbf{m})\mathbf{u}_i}{\partial_{\mathbf{m}}}\right)^* \mathbf{v}_i$ `//evaluate gradient`
4. $\mathbf{g} = \sum_{i=1}^{n_s} \mathbf{g}_i$ `//sum gradient components`

Proposed algorithm

Solve in parallel:

1 PDE per source & 1 PDE per receiver

achieves 2X speedup in time if:

$n_{\mathrm{src}} + n_{\mathrm{rec}} \leq$ maximum number of PDE solves in parallel

We need

- stochastic optimization
- source & receiver randomization + subsampling

Algorithm 3 Stochastic optimization algorithm to minimize $f(\mathbf{m}) = \sum_{i=1}^{n_s} f_i(\mathbf{m})$.

iteration counter $k = 1$, set sufficient descent parameter c

while not converged

1a. $\tilde{\mathbf{q}}$ `//draw 1 or a few source samples`

1b. $\tilde{\mathbf{p}}$ `//draw 1 or a few receiver samples`

`//approximate function value and gradient`

2. $\tilde{f}(\mathbf{m}) = \frac{1}{2}\sum_{i=1}^{\tilde{n}_s} \|\tilde{P}A(\mathbf{m}_k)^{-1}\tilde{\mathbf{q}}_i - \tilde{\mathbf{d}}_i\|_2^2$

3. $\tilde{\mathbf{g}} = \sum_{i=1}^{\tilde{n}_s} \mathbf{g}_i$

4a. $\tilde{f}_{\text{ref}} = \{\tilde{f}_k, \tilde{f}_{k-1}, \dots, \tilde{f}_{k-M}\}$

4b. $\gamma = 1$

4c. **if** $\tilde{f}(\mathbf{m}_k - \gamma\tilde{\mathbf{g}}) < \max(f_{\text{ref}}) + c$

$\mathbf{m}_{k+1} = \mathbf{m}_k - \gamma\tilde{\mathbf{g}}$ `// update model estimate`

$k = k + 1$

else

$\gamma = \eta\gamma$ `//step size reduction,` $\eta < 1$

go back to 4c

end

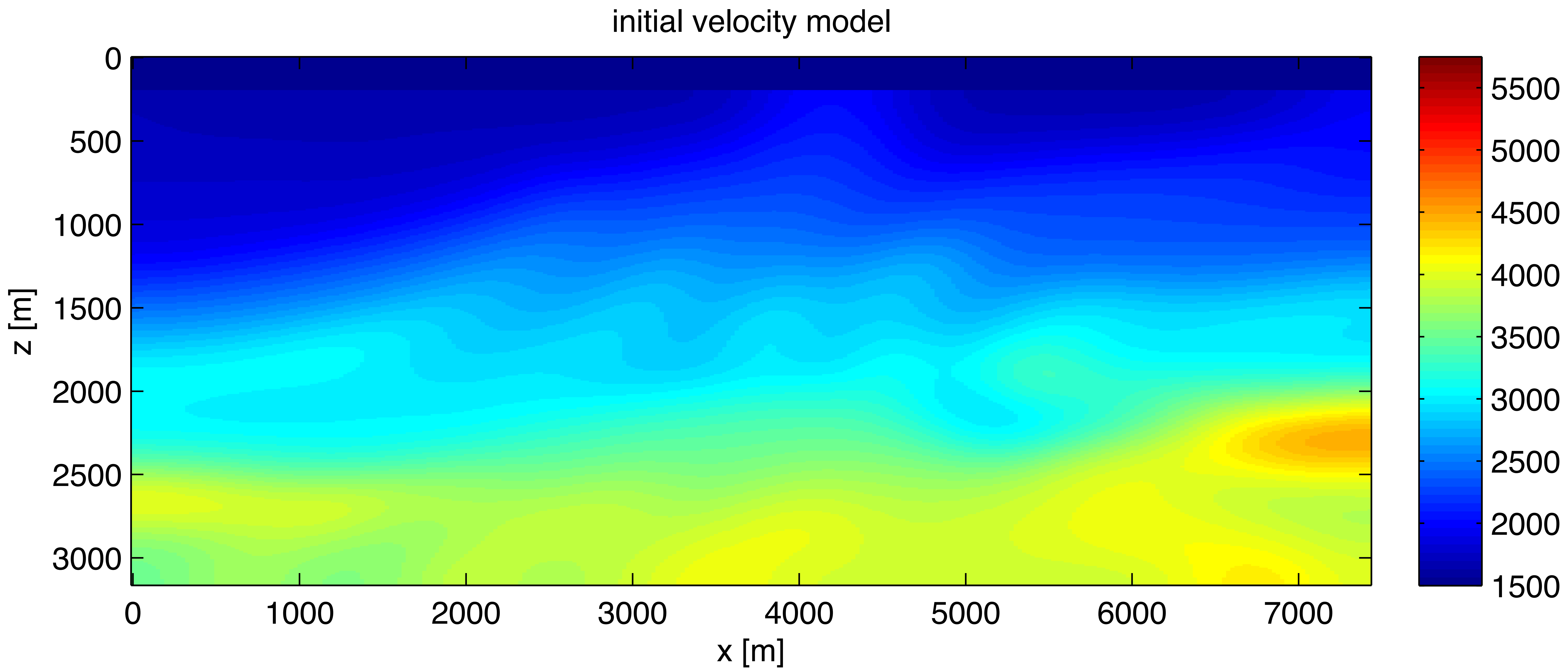
Numerical example 1

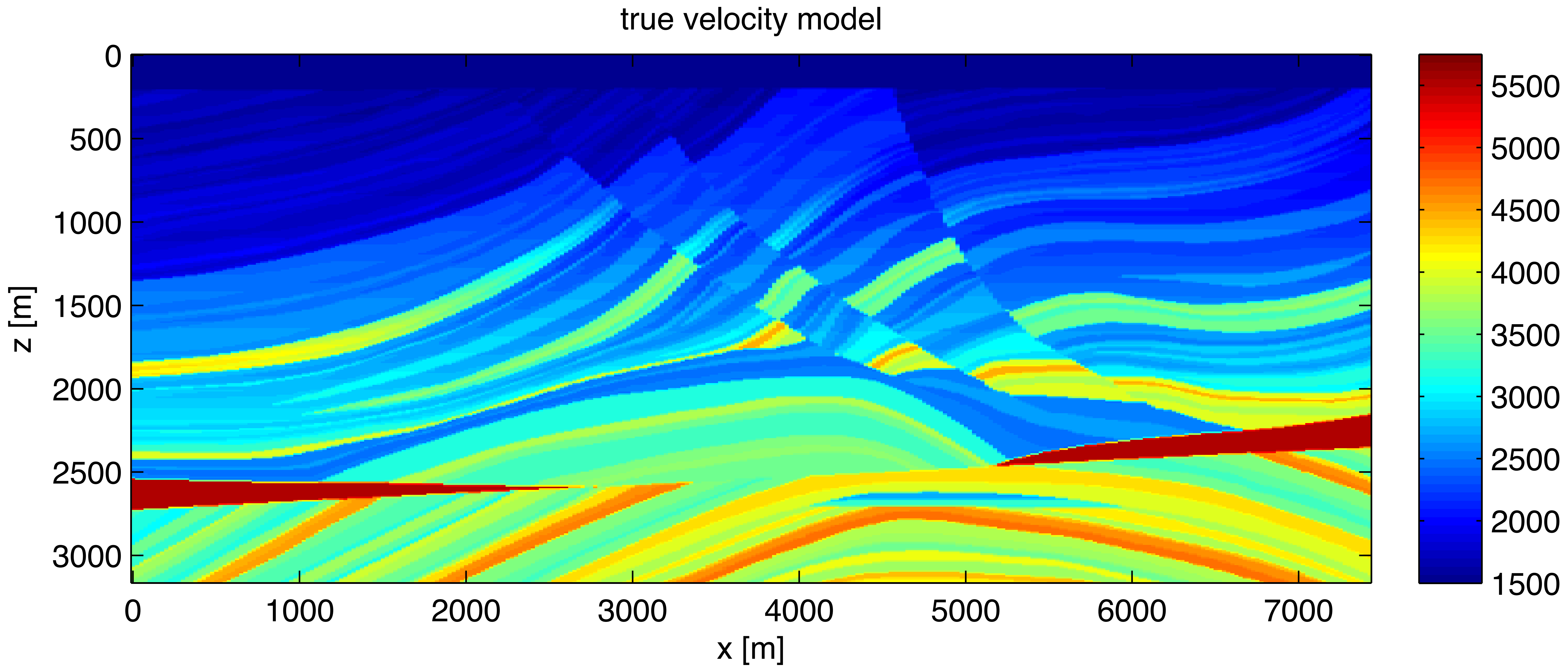
Optimization:

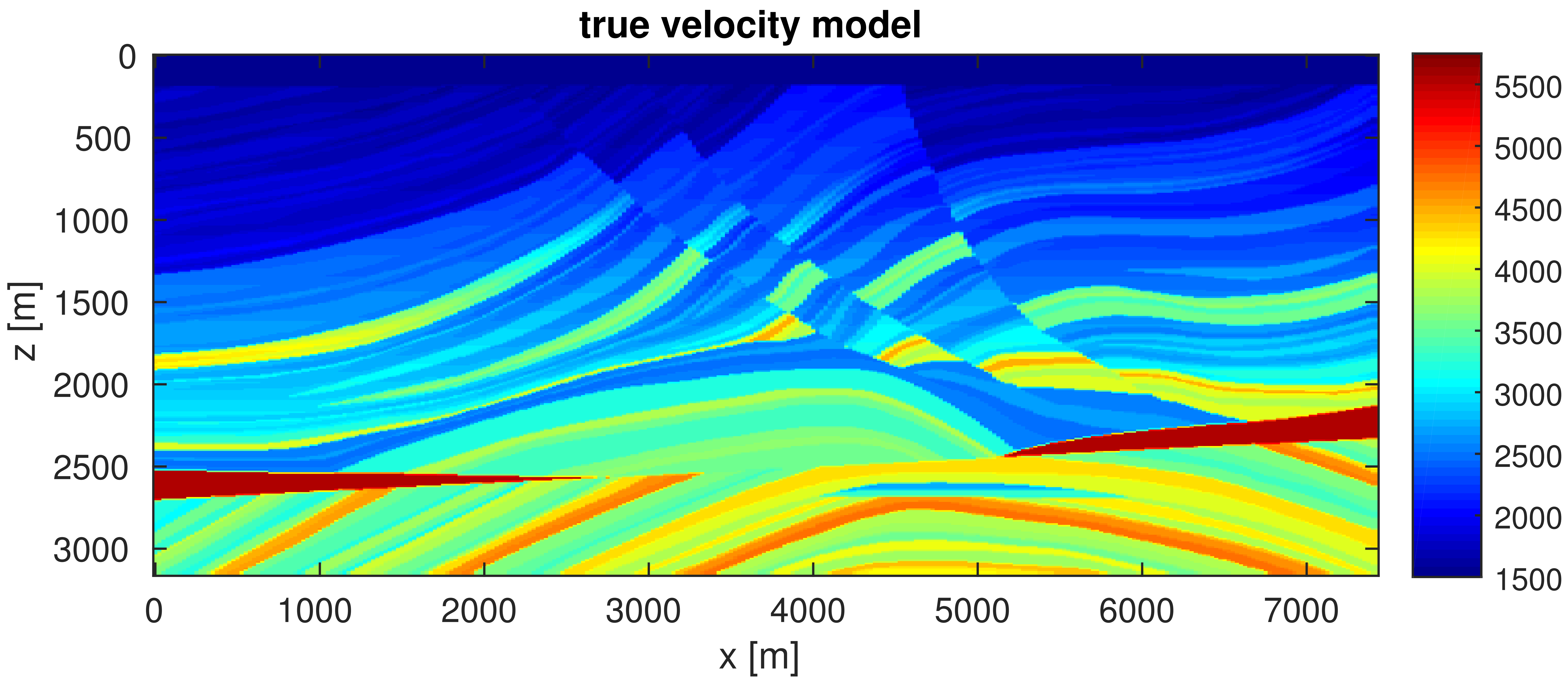
- stochastic gradient descent-type with bound-constraints
- redraw subset of sources and receivers every iteration
- non-monotone line-search

Experiment:

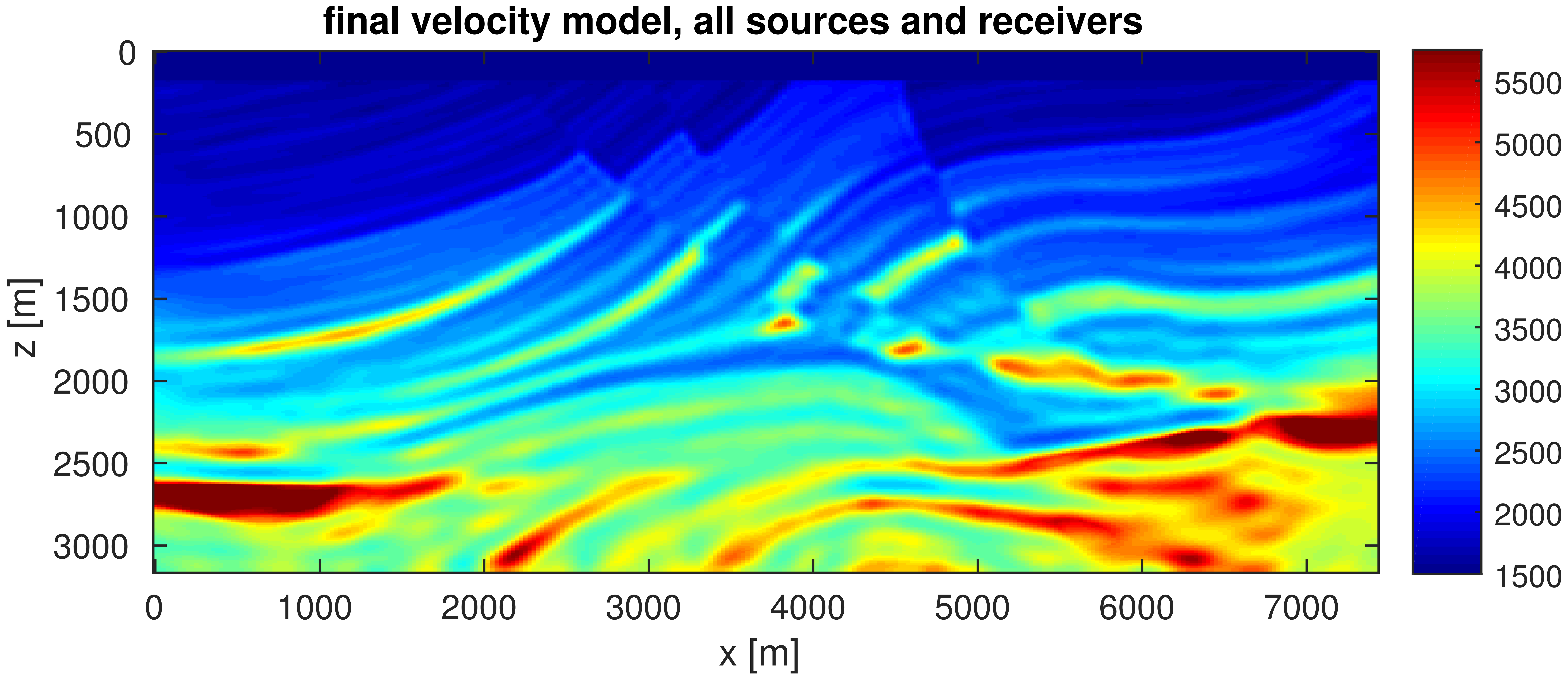
- 3 - 10 Hz data (2 cycles through the frequencies)
- Ricker wavelet, 10Hz peak
- source spacing: 50m, receiver spacing: 50m


initial velocity model
0
500
1000
1500
2000
2500
3000
z [m]
0
1000
2000
3000
4000
5000
6000
7000
x [m]
5500
5000
4500
4000
3500
3000
2500
2000
1500

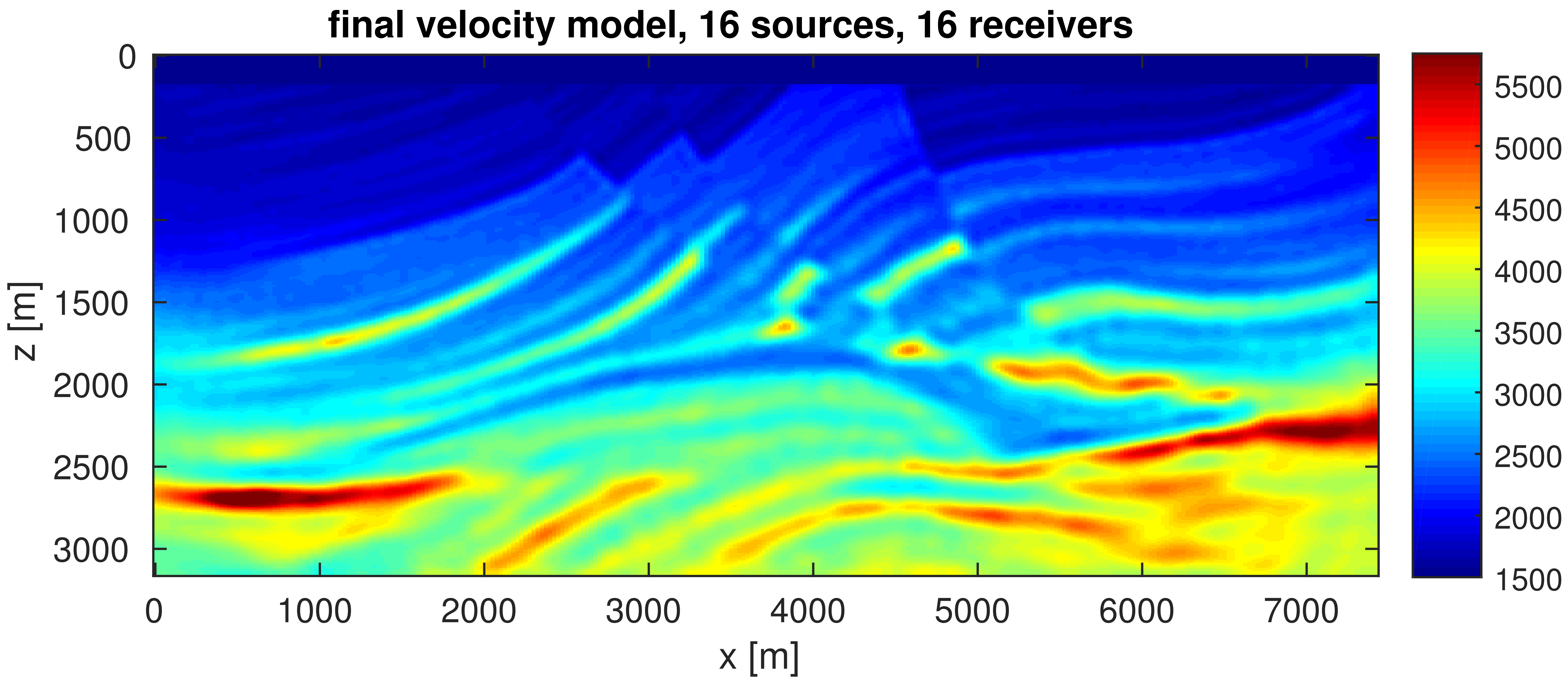

true velocity model
0
500
1000
1500
2000
2500
3000
z [m]
0
1000
2000
3000
4000
5000
6000
7000
x [m]
5500
5000
4500
4000
3500
3000
2500
2000
1500


true velocity model
z [m]
x [m]
0
500
1000
1500
2000
2500
3000
0
1000
2000
3000
4000
5000
6000
7000
5500
5000
4500
4000
3500
3000
2500
2000
1500

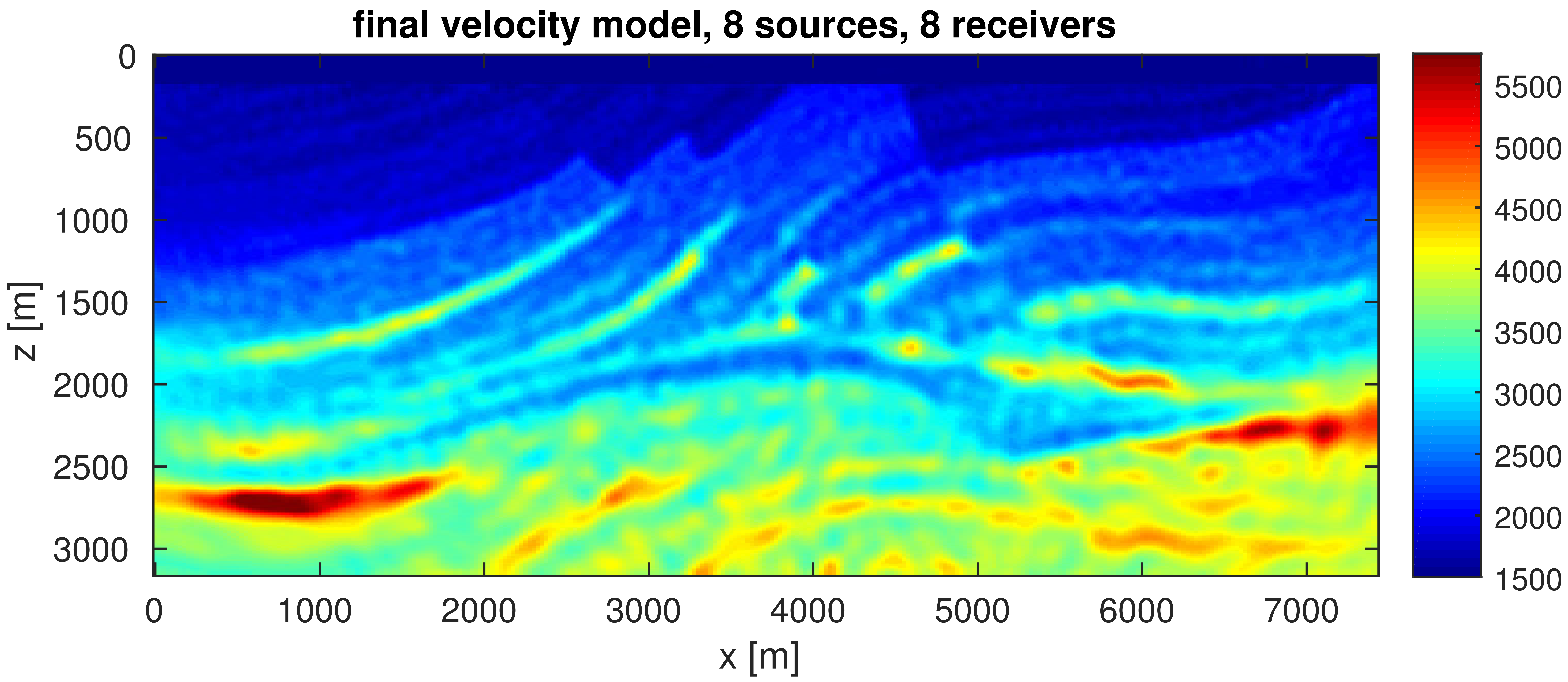
all sources & all receivers
bound-constraints



16 randomly selected sources
16 randomly selected receivers
bound-constraints



8 randomly selected sources
8 randomly selected receivers
bound-constraints



Connections to Gauss-Newton methods

Gauss-Newton Hessian (one frequency):

$$H_{\mathrm{GN}} = J^* J = \sum_{i=1}^{n_{\mathrm{s}}} G(\mathbf{u}_i)^* A^{-*} P^* P A^{-1} G(\mathbf{u}_i)$$

In a serial computing setting: solve and save the two types of fields :

$$H_{\mathrm{GN}} = \sum_{i=1}^{n_{\mathrm{s}}} G(\mathbf{u}_i)^* W W^* G(\mathbf{u}_i)$$

[Habashy et al. , 2011]

Combine with source & receiver compression to reduce storage.

Outlook

Number of sources & receivers can be reduced if prior information is available.

In this case we can solve:

$$f(\mathbf{m}) = \frac{1}{2}\sum_{i=1}^{n_{\mathrm{src}}} \|PA(\mathbf{m})^{-1}\mathbf{q}_i - \mathbf{d}_i\|_2^2 \quad \text{s.t.} \quad \mathbf{m} \in \mathcal{C}$$

$\mathcal{C}$: convex set, describing smoothness, total-variation, sparsity properties, etc

Conclusions

Proposed algorithm: different way to compute the gradient

- Solve all wave-equations in parallel.
- Twice as fast as adjoint state, provided sufficient parallel resources.
- Requires source and receiver subsampling to be practical.
- Applies to any inverse problem with the same structure as FWI.

Acknowledgements

This research was carried out as part of the SINBAD project with the support of the member organizations of the SINBAD Consortium.