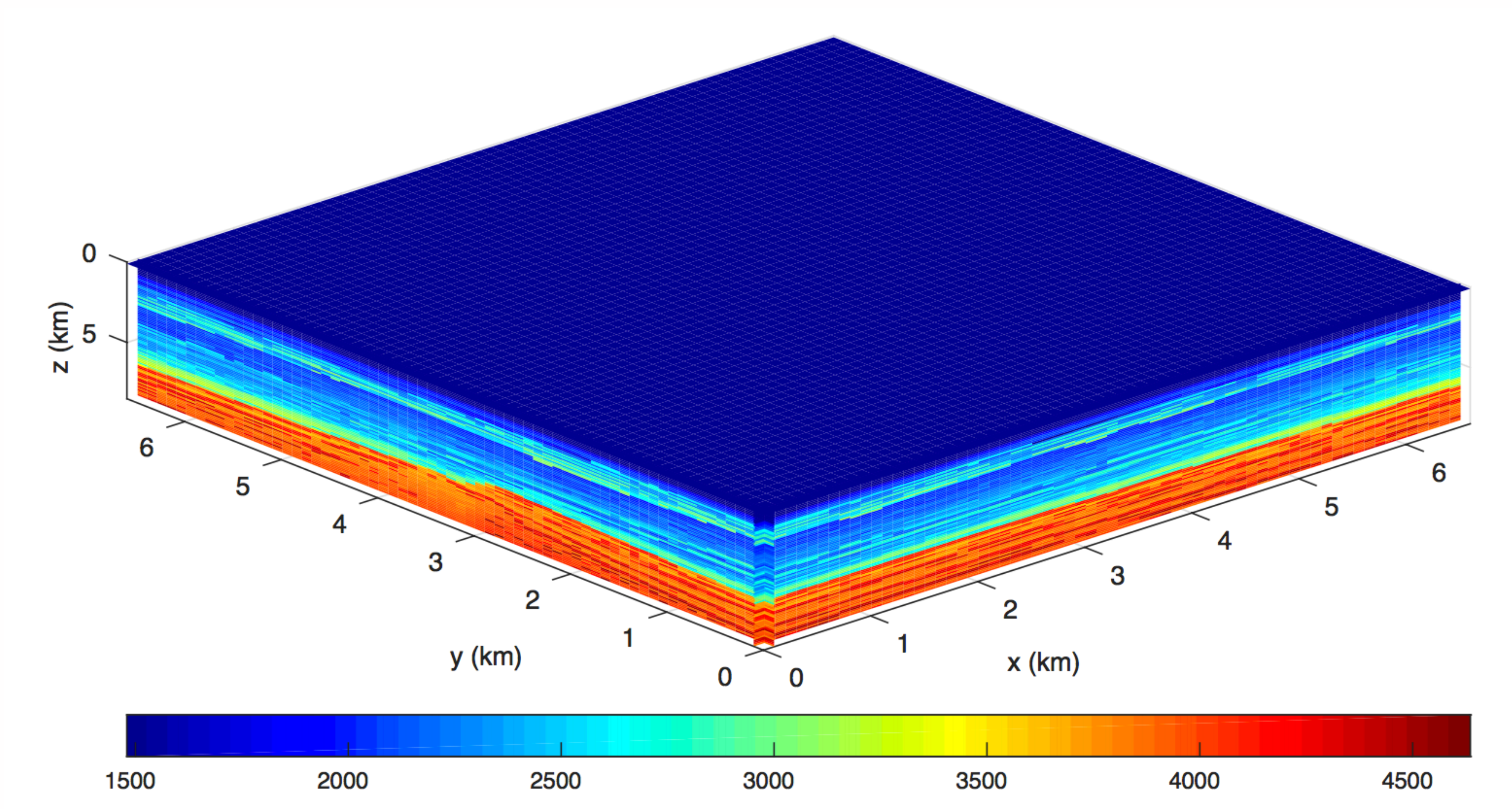
Massive 3D seismic data compression & interpolation w/ on-the-fly data extraction

Yiming Zhang, Curt Da Silva, Rajiv Kumar & Felix J. Herrmann




SLIM
University of British Columbia

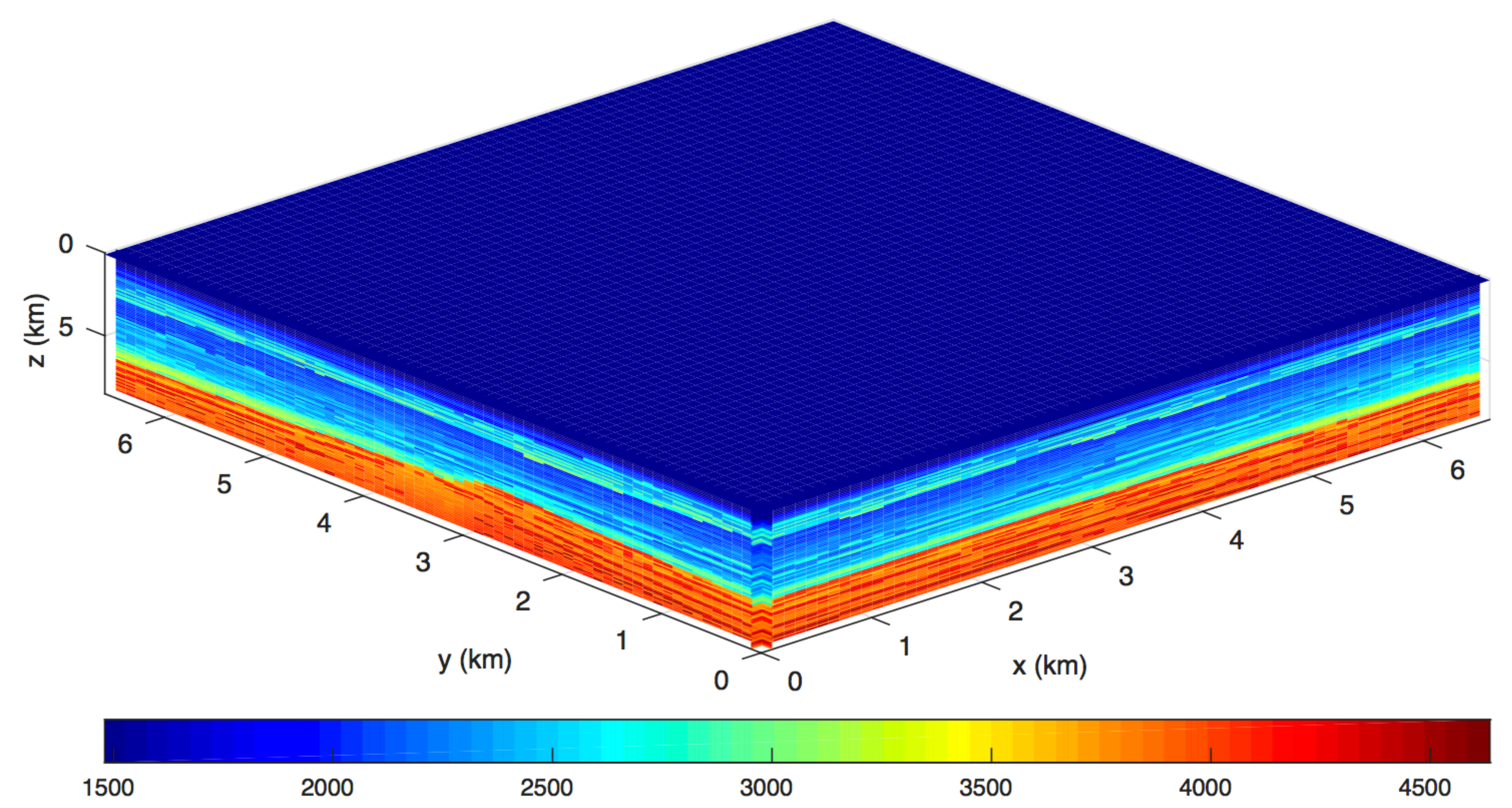
Motivation

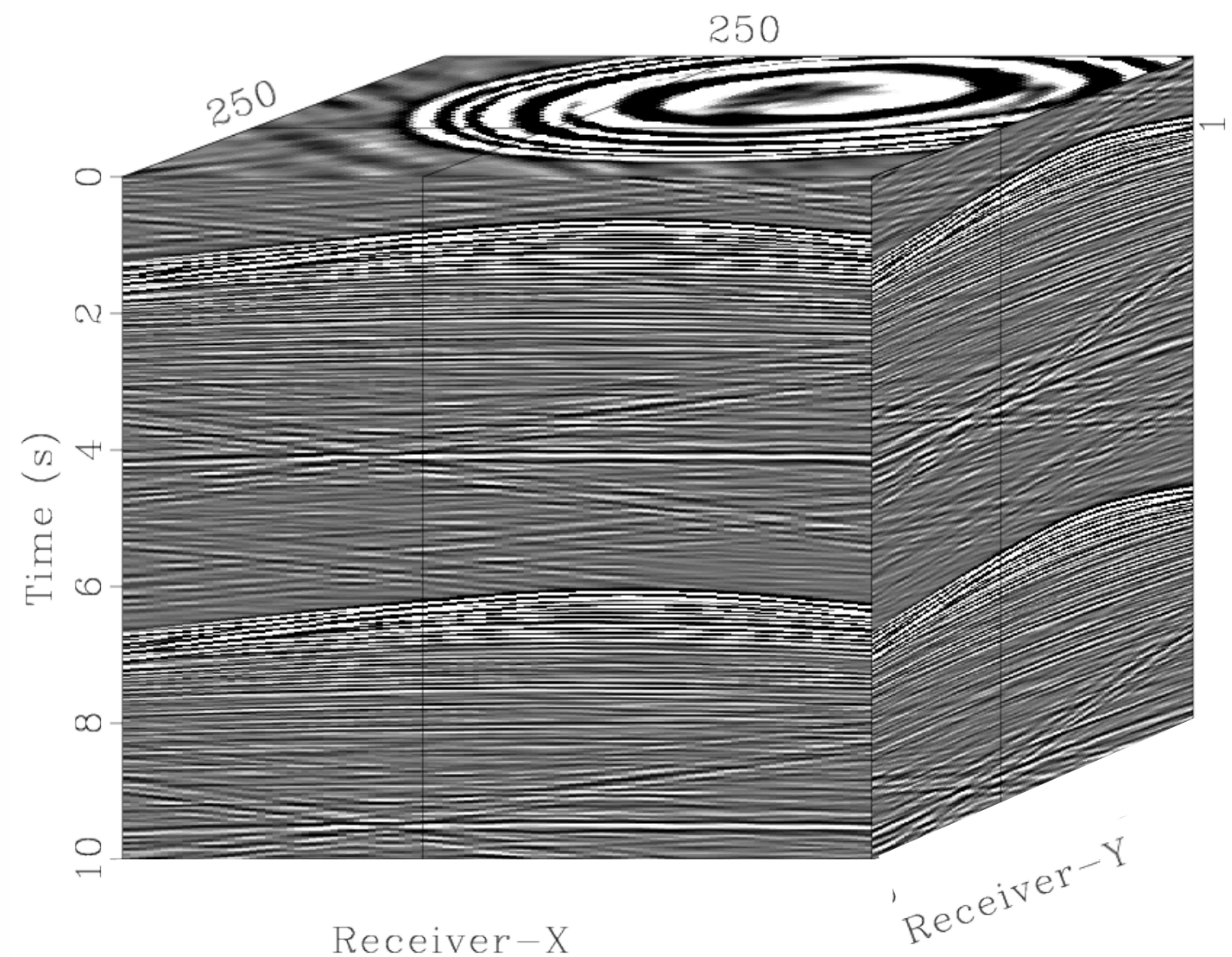

0
5
z (km)
6
5
4
3
2
1
0
y (km)
0
1
2
3
4
5
6
x (km)
1500
2000
2500
3000
3500
4000
4500

Motivation

- **Enormous** volumes of seismic data

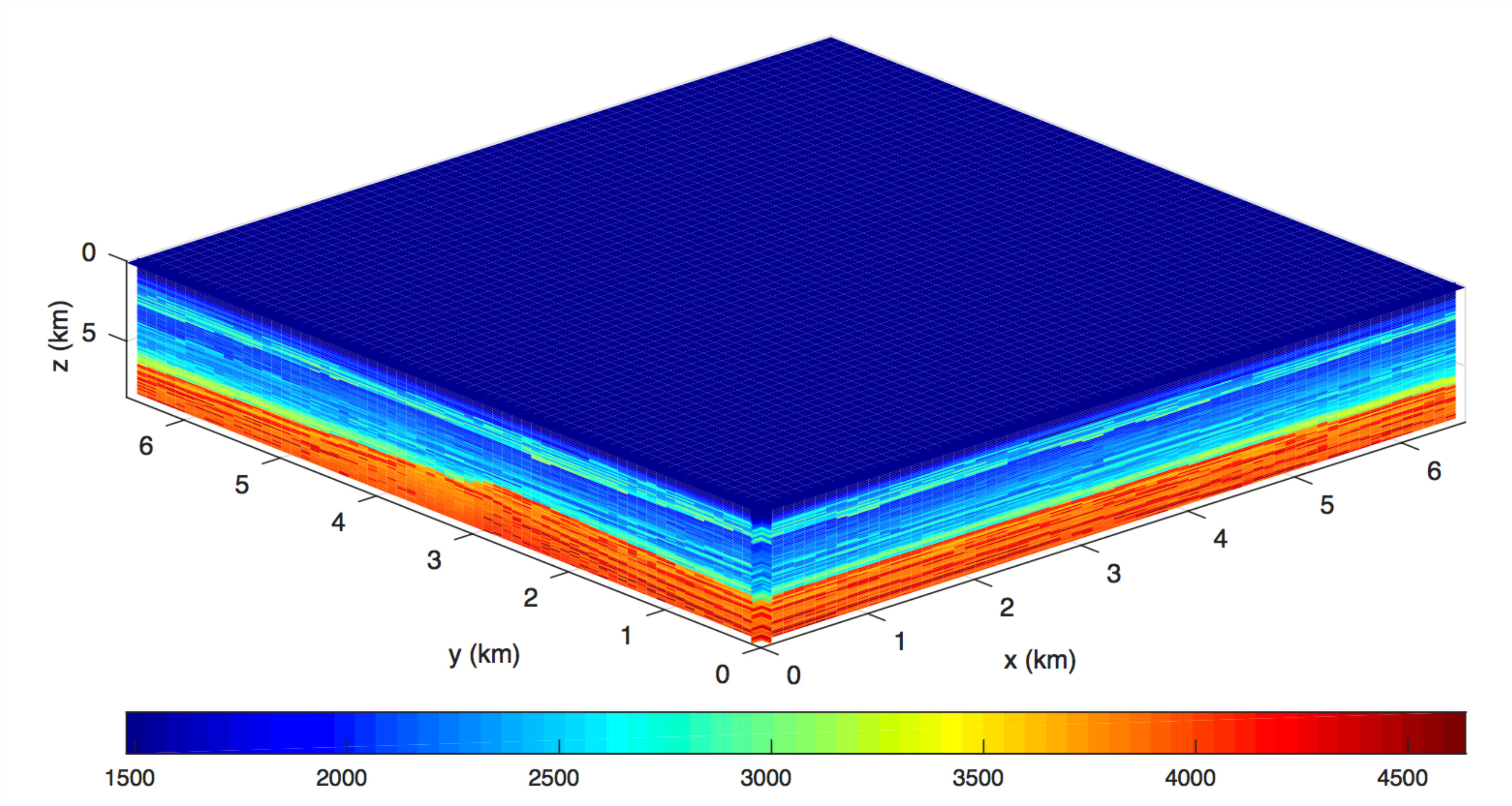
~1TB



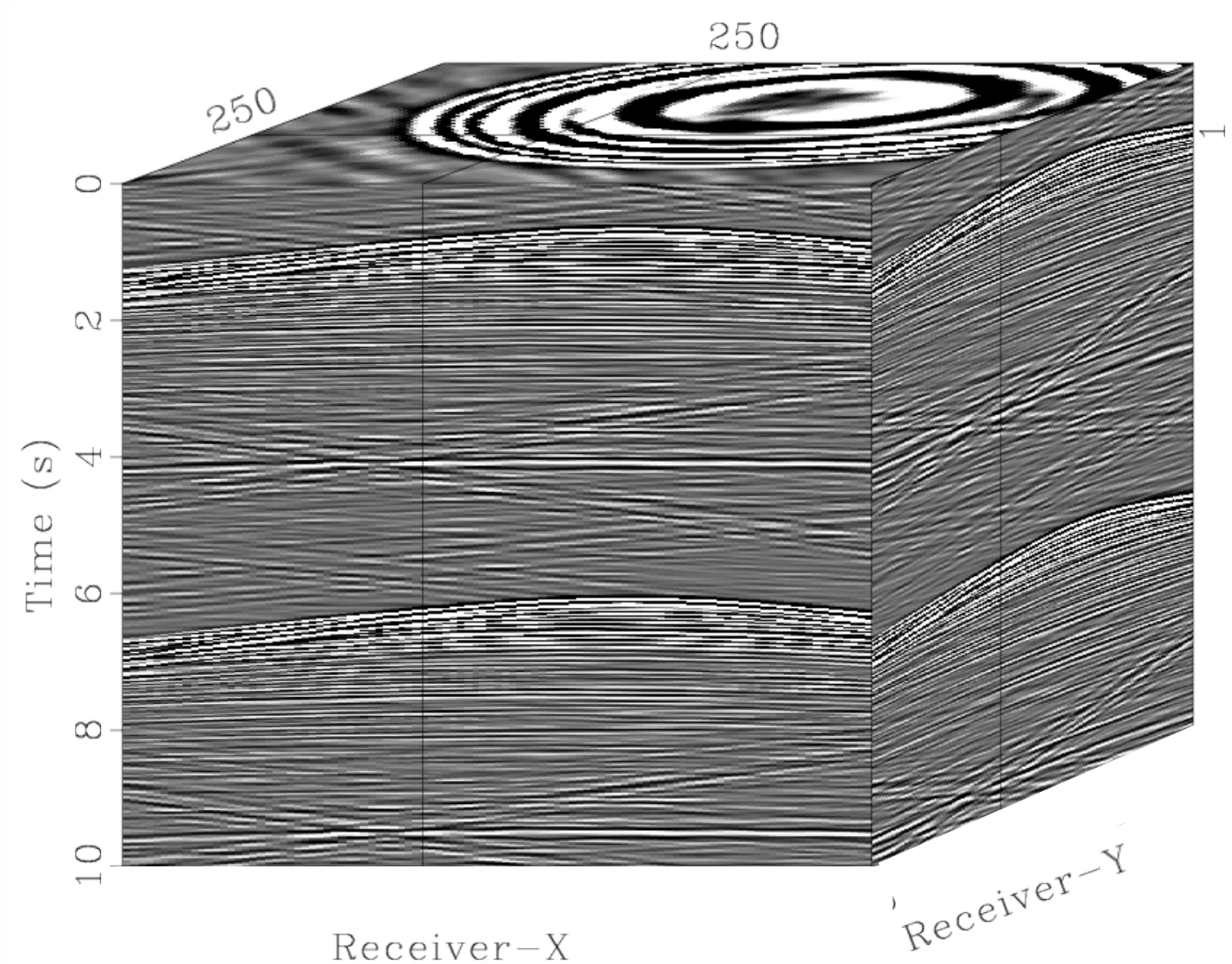


Motivation

- **Challenging** in inversion



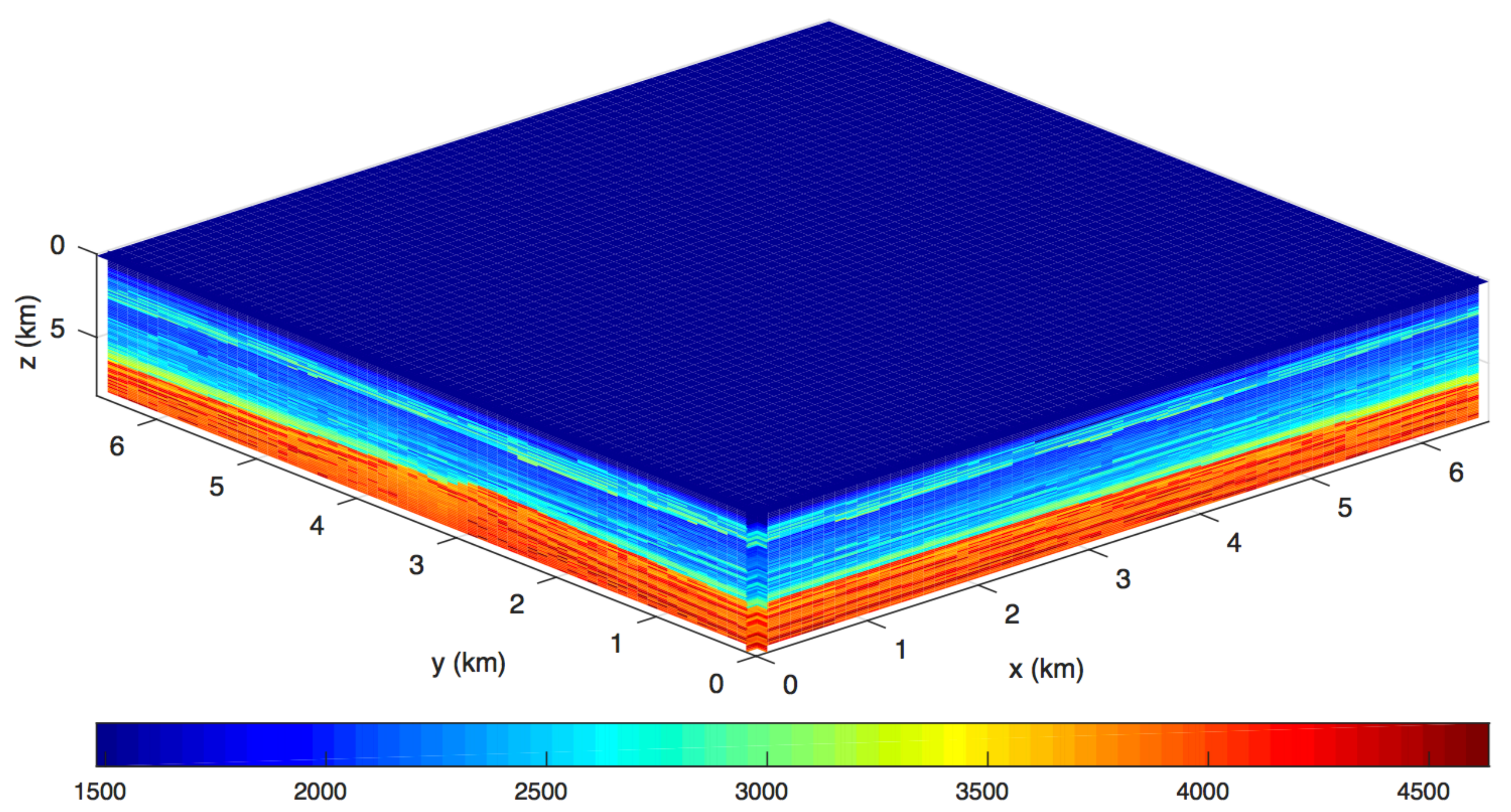
~1TB



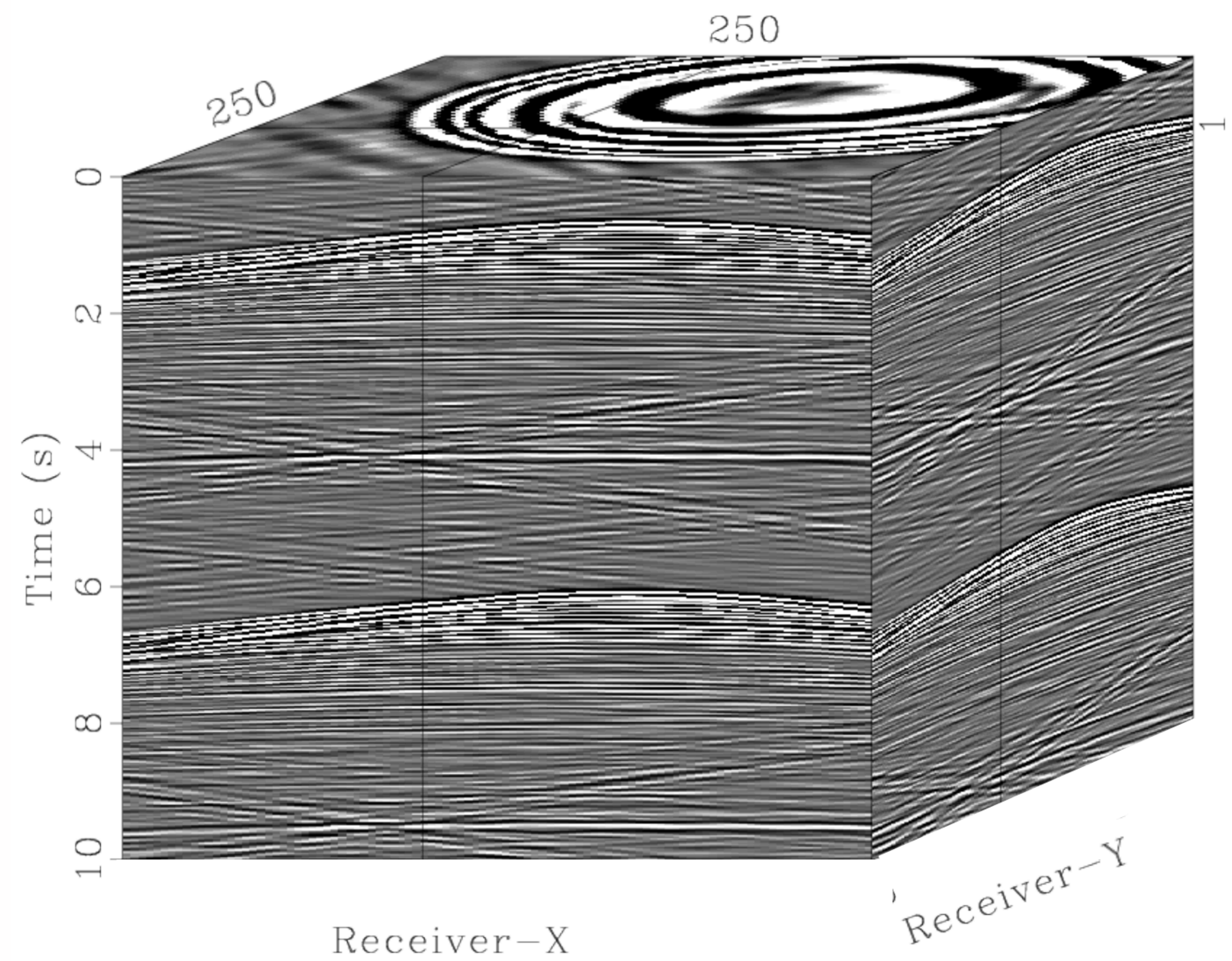
Motivation

- **Challenging** in inversion

~1TB



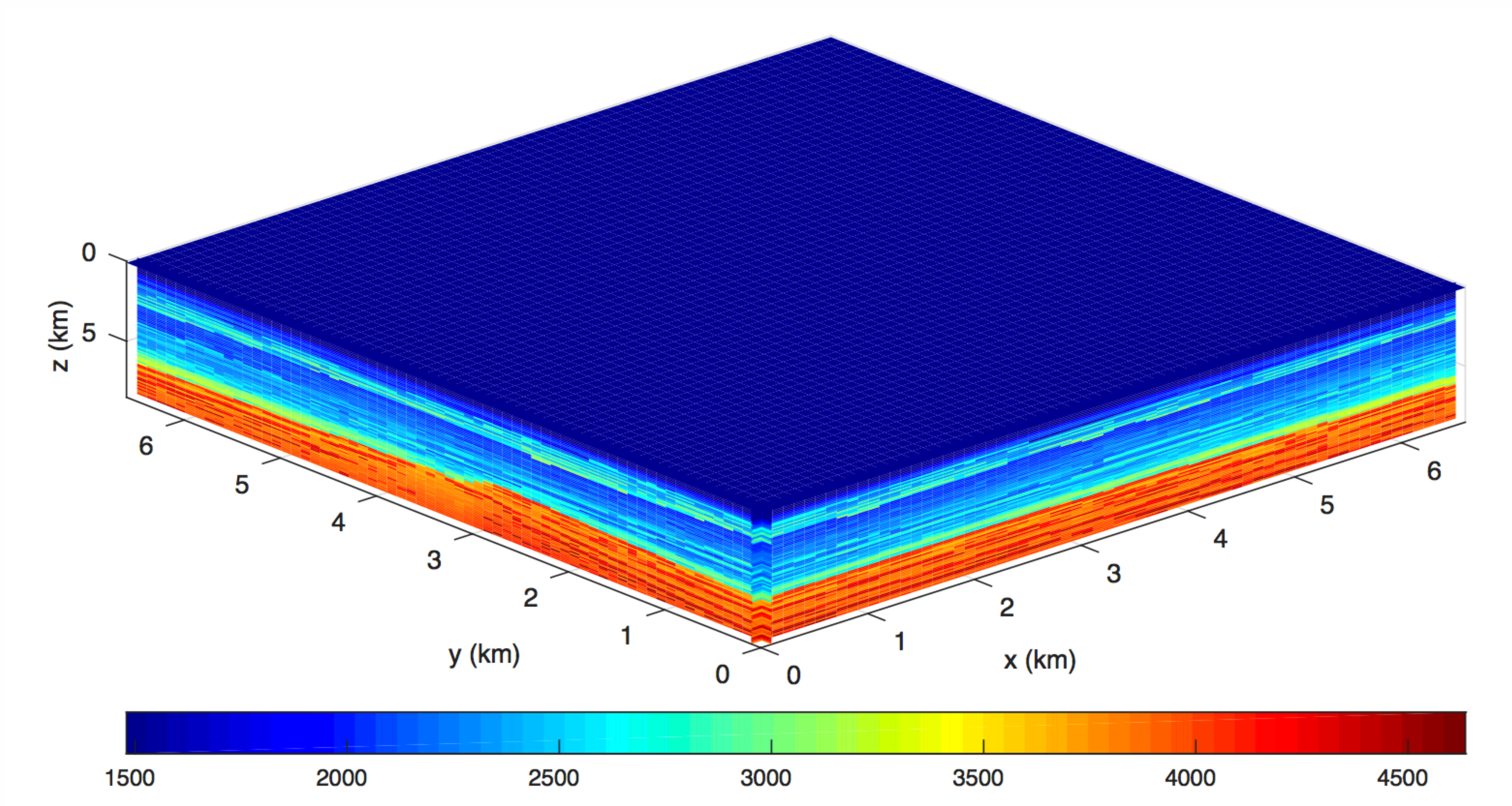
?

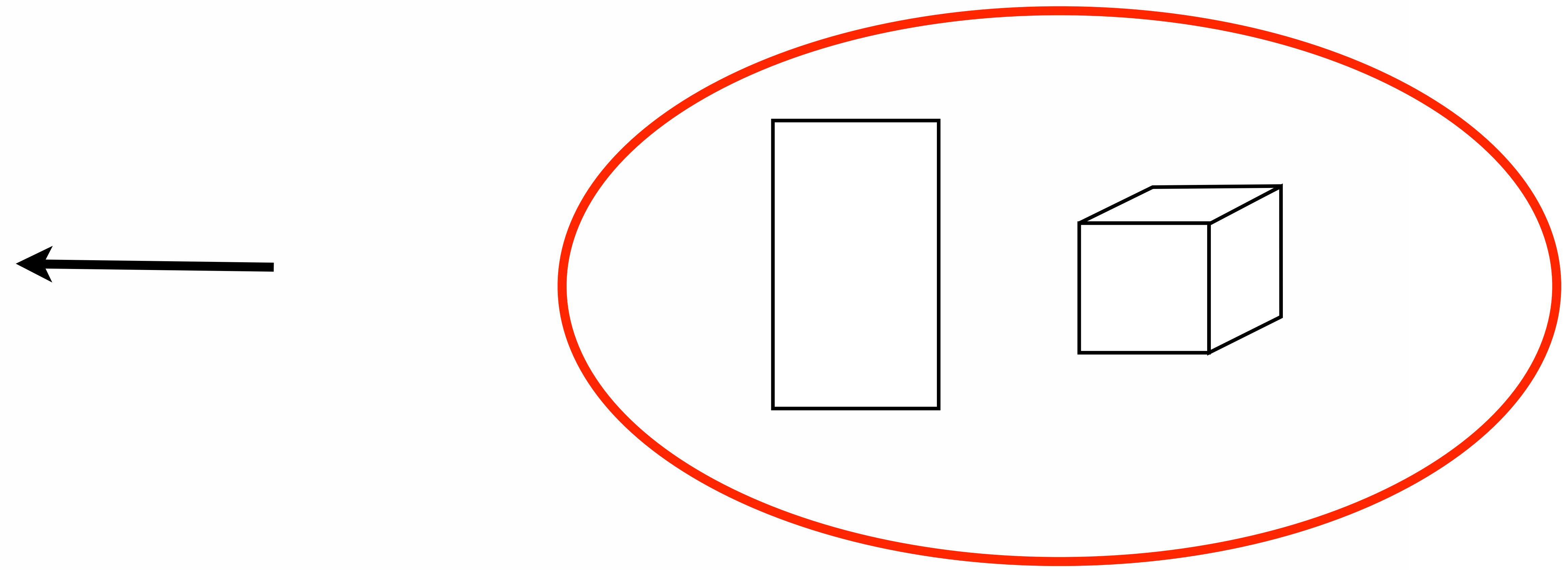


Motivation

- **How about in this way**

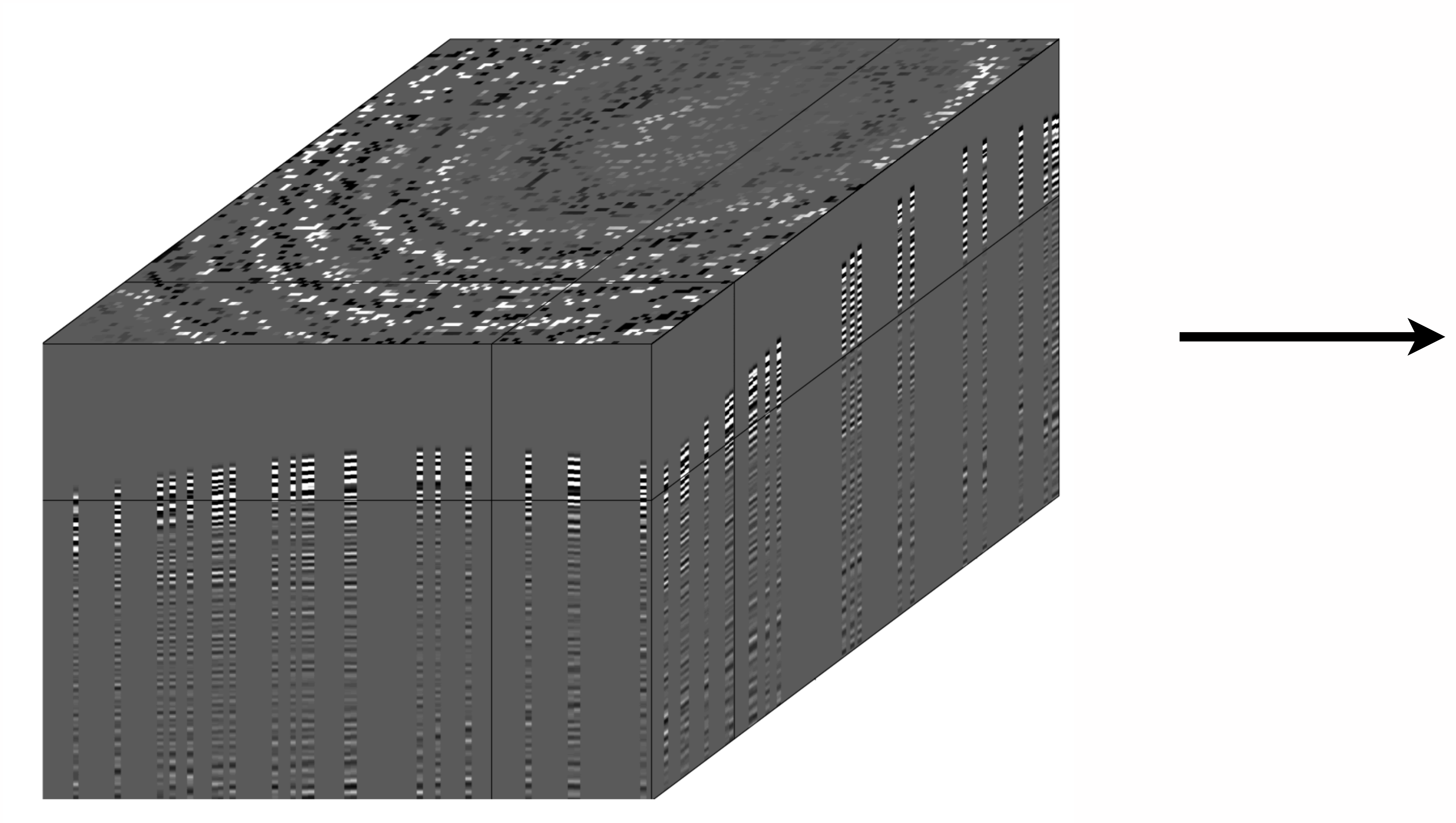
~1GB



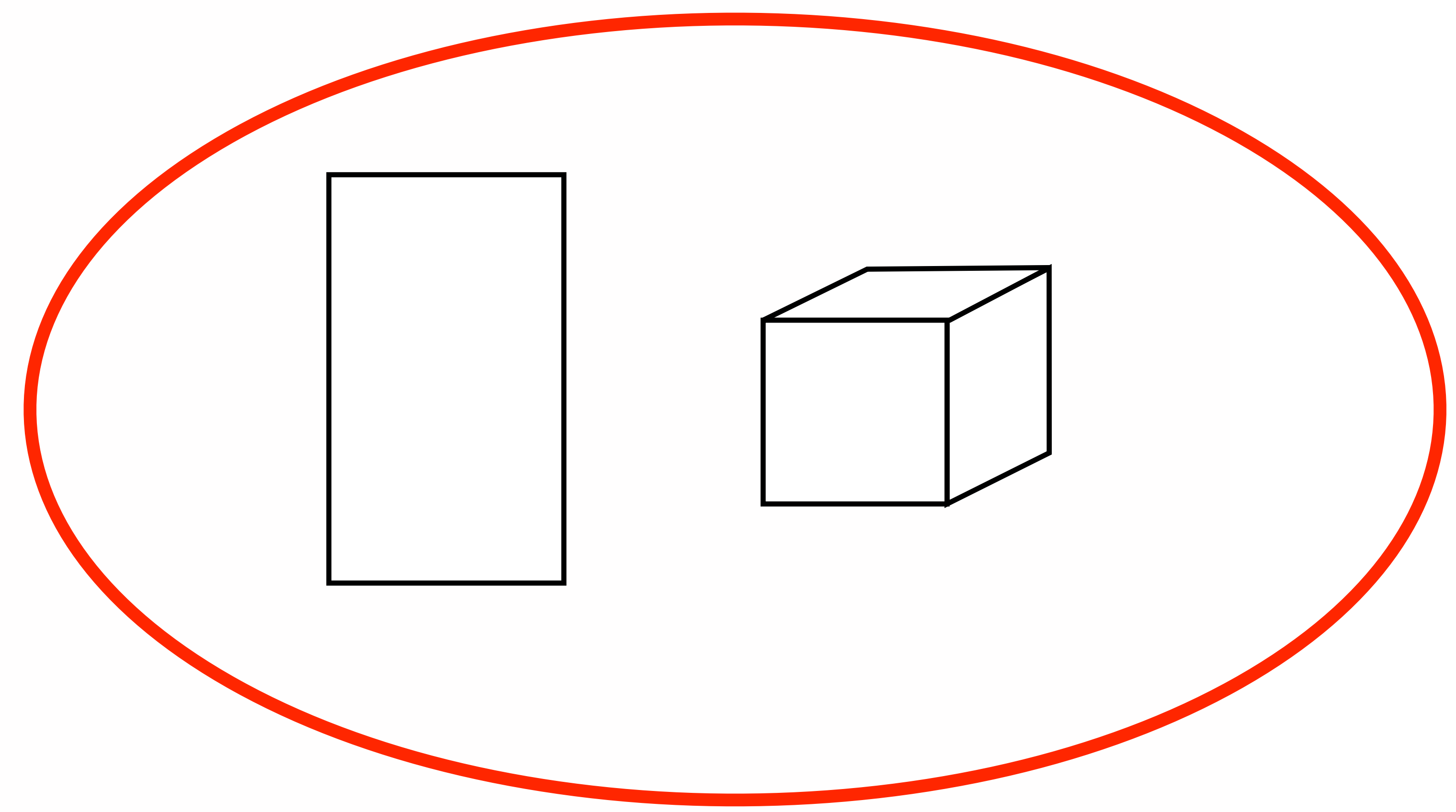


Motivation

- **Missing data scenarios**



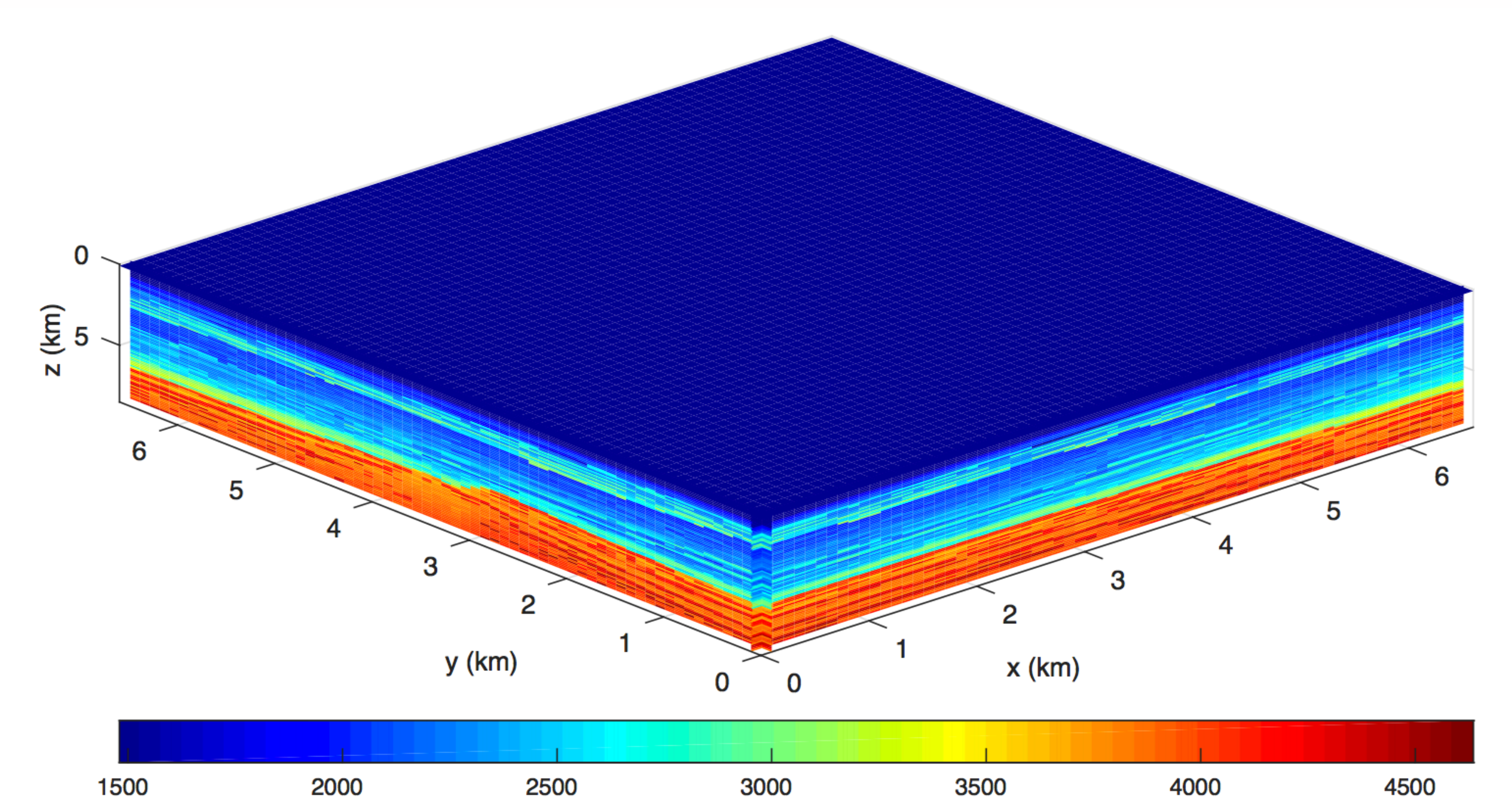
~1GB

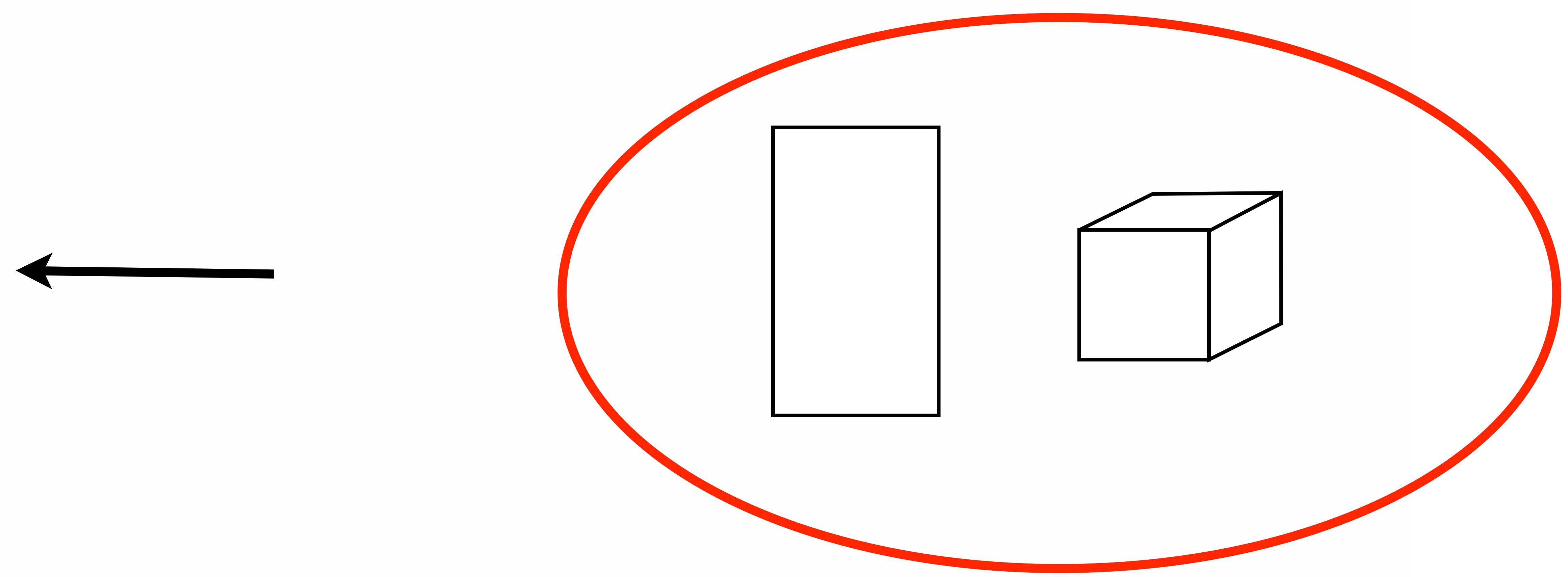


Motivation

- **can still in this way?**

~1GB





Motivation

How to fight "curse of dimensionality" ?

Motivation

How to fight "curse of dimensionality" ?

Situation:

- seismic data is redundant
- low-rank format can be exploited at low frequencies

Motivation

How to fight “curse of dimensionality” ?

Situation:

- seismic data is redundant
- low-rank format can be exploited at low frequencies

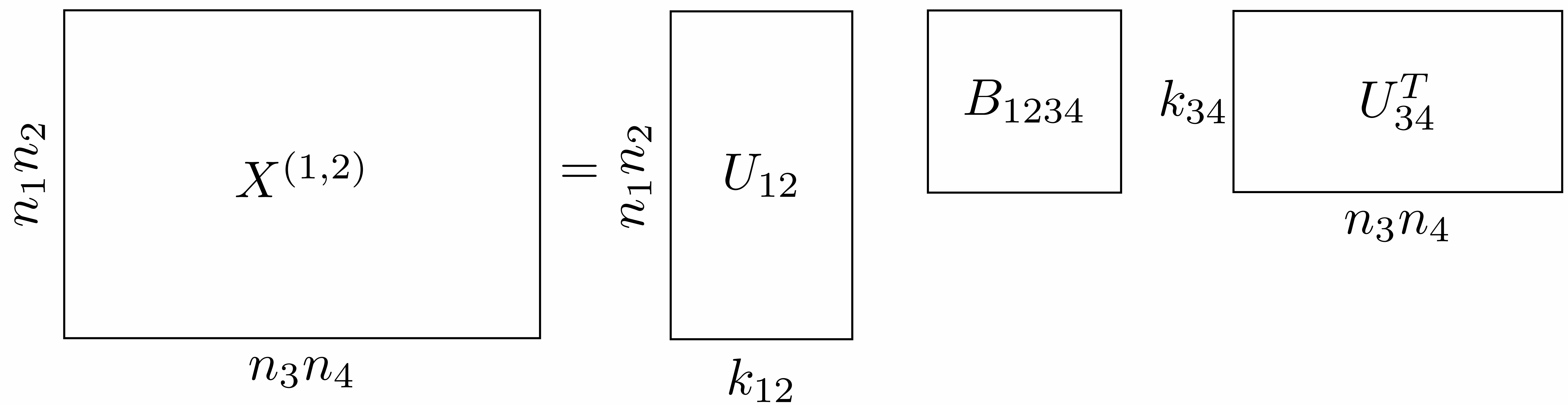
Solution:

- represent in hierarchical Tucker (HT) format
- interpolate HT format when missing data
- work w/ full data volume w/o forming them for later downstream processes, e.g. FWI

Da Silva, C., and F. J. Herrmann, 2015, Optimization on the hierarchical tucker manifold–applications to tensor completion: Linear Algebra and its Applications, 481, 131–173.

Hierarchical Tucker representation

$$X = n_1 \times n_2 \times n_3 \times n_4 \text{ tensor}$$

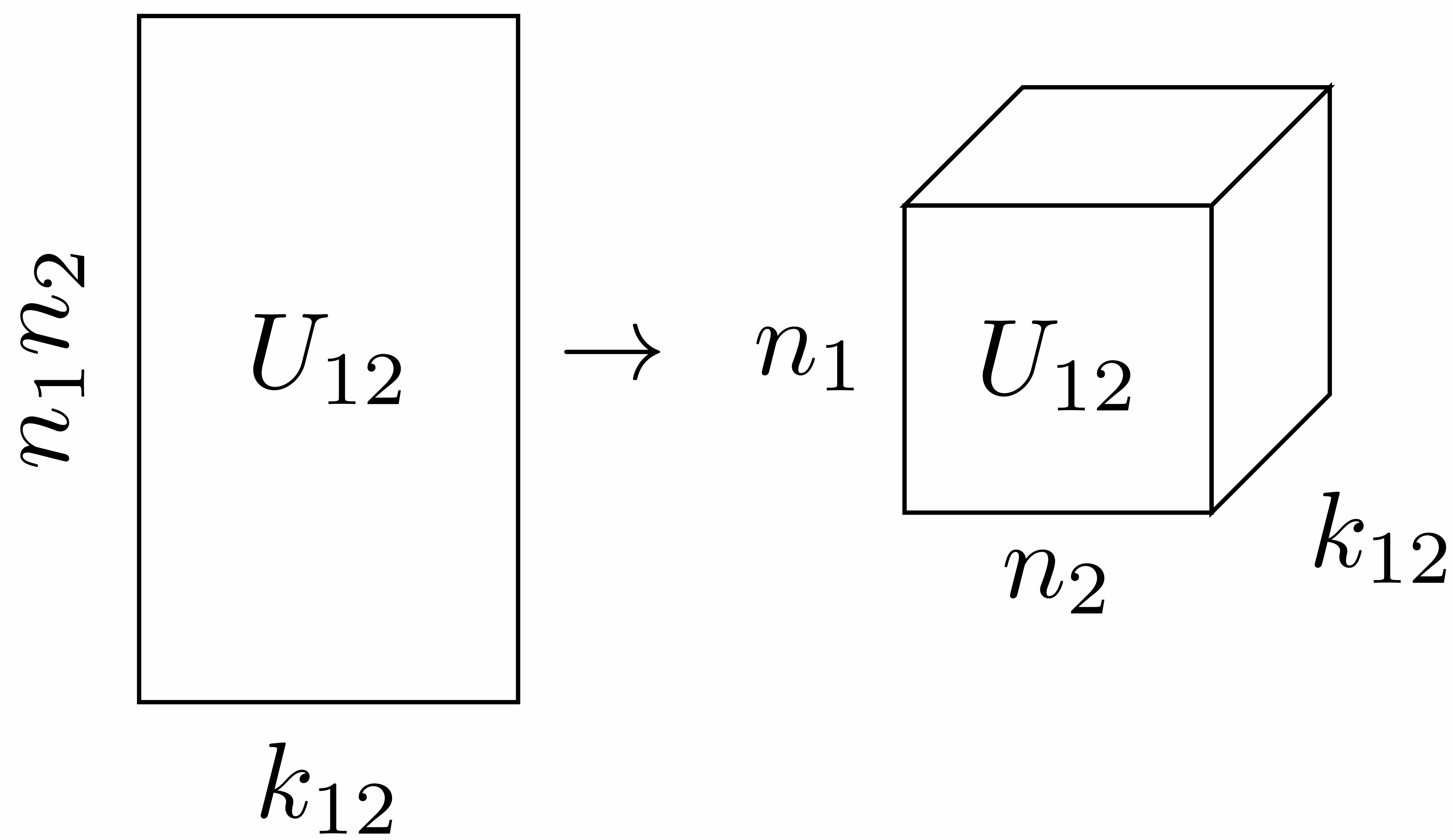


"SVD-like" factorization

Da Silva, C., and F. J. Herrmann, 2015, Optimization on the hierarchical tucker manifold–applications to tensor completion: Linear Algebra and its Applications, 481, 131–173.

Hierarchical Tucker representation

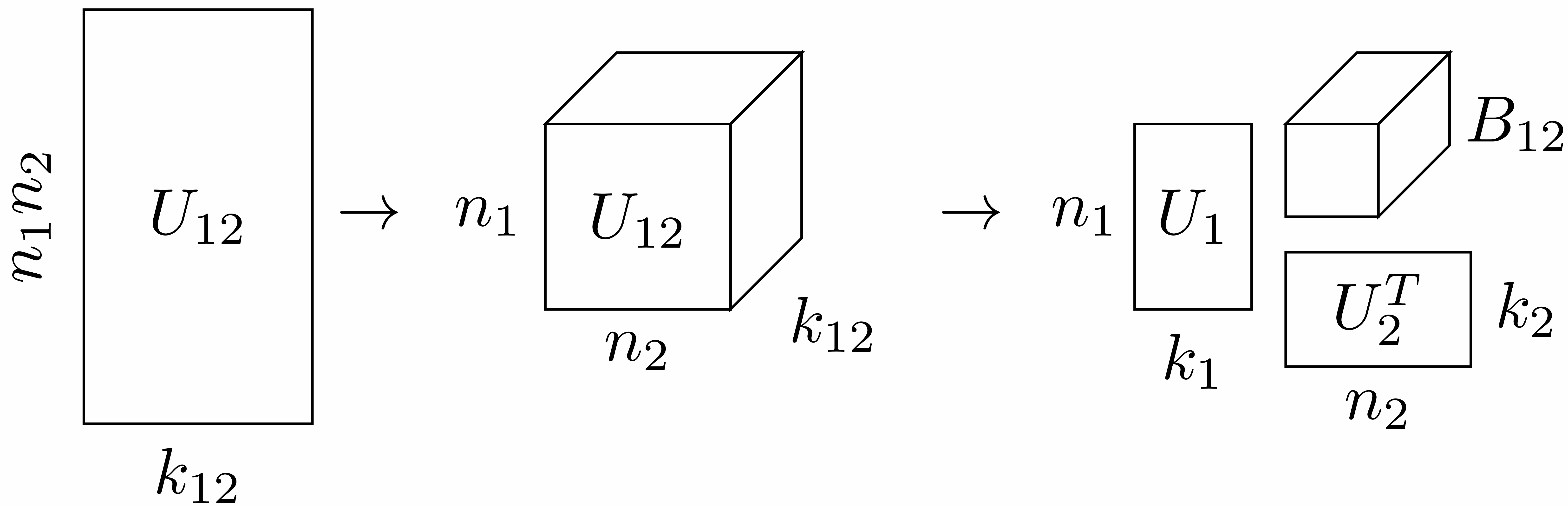
$X = n_1 \times n_2 \times n_3 \times n_4$ tensor



Da Silva, C., and F. J. Herrmann, 2015, Optimization on the hierarchical tucker manifold–applications to tensor completion: Linear Algebra and its Applications, 481, 131–173.

Hierarchical Tucker representation

$$X = n_1 \times n_2 \times n_3 \times n_4 \text{ tensor}$$



Hierarchical Tucker representation

This format is extremely storage-efficient

- not necessarily store **intermediate** matrices U_{12} and U_{34}
- storage $\leq dNk + (d-2)k^3 + k^2$
- compare to N^d parameters needed to store for the full data
- **computationally tractable** for high-dimensional problem

  $(k << N)$

Hierarchical Tucker representation

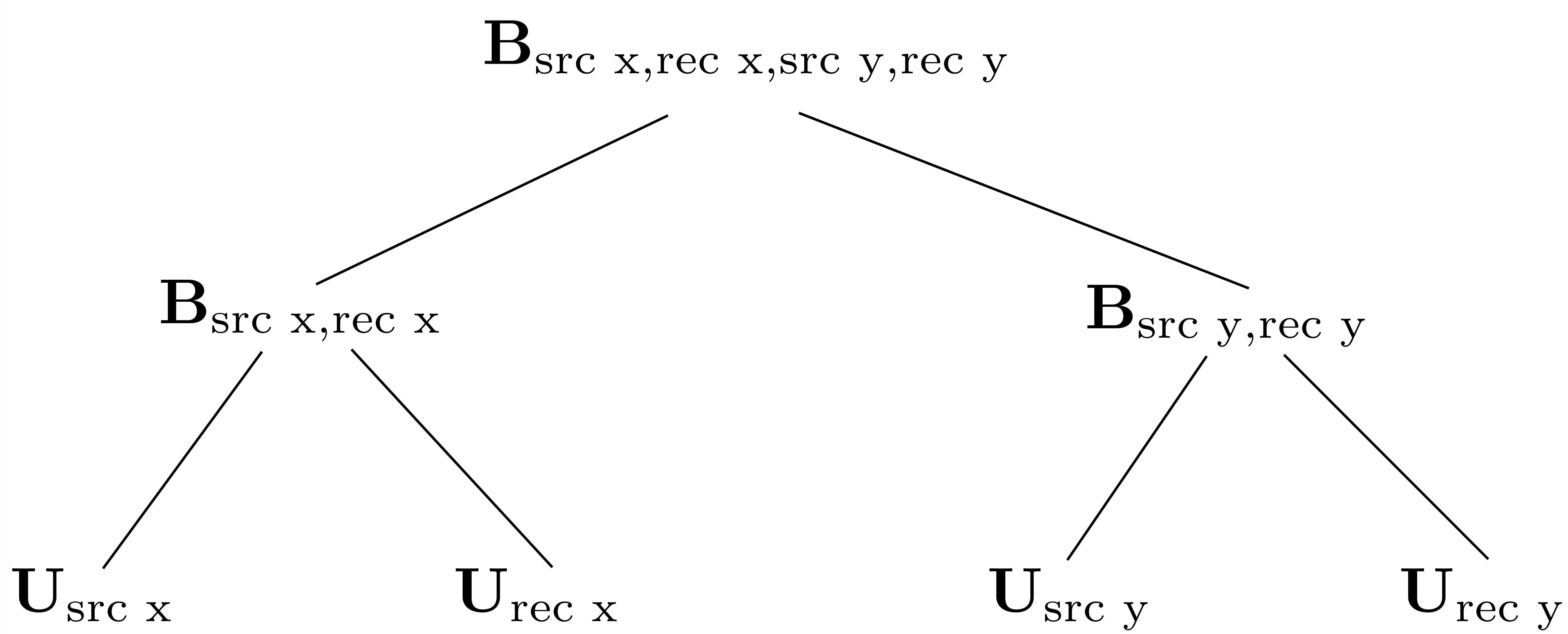
- a $100 \times 100 \times 100 \times 100$ tensor, max HT rank 20
- full storage: $100^4 = 10^8$ parameters
- HT storage: 24400 values
- compression ratio: 99.97%

Kumar, R., Da Silva, C., Akalin, O., Aravkin, A. Y., Mansour, H., Recht, B., & Herrmann, F. J. (2015). Efficient matrix completion for seismic data reconstruction. Geophysics, 80(5), V97-V114.

Da Silva, C., and F. J. Herrmann, 2015, Optimization on the hierarchical tucker manifold–applications to tensor completion: Linear Algebra and its Applications, 481, 131–173.

Seismic hierarchical Tucker

Given a frequency slice with coordinate (src x, src y, rec x, rec y), we introduce the **non-canonical dimension tree** for seismic data.

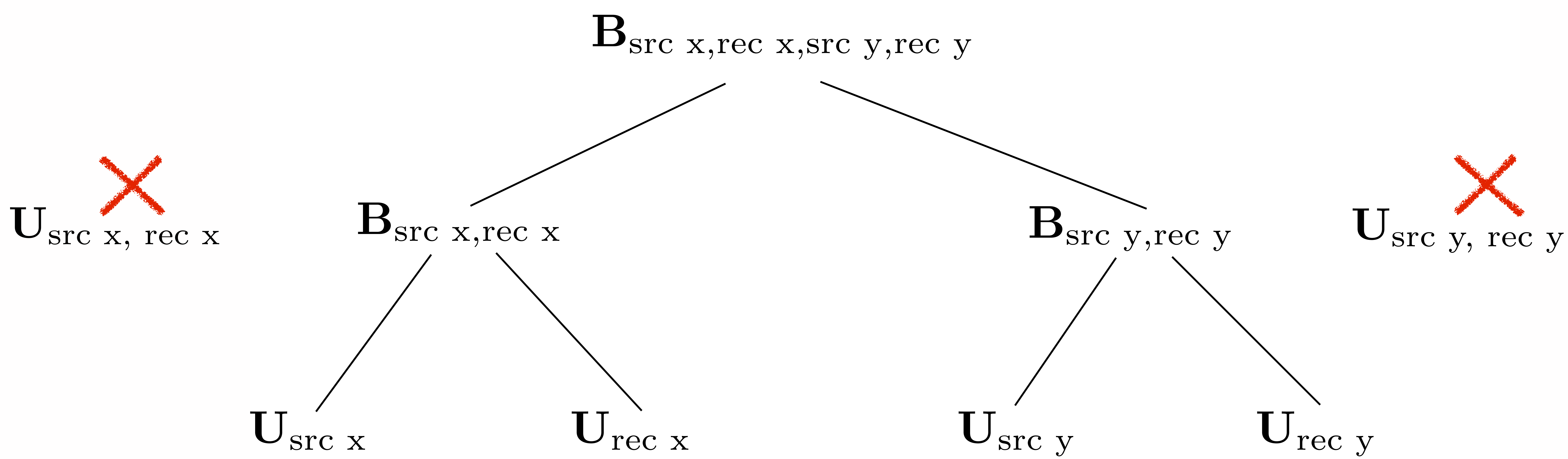


Kumar, R., Da Silva, C., Akalin, O., Aravkin, A. Y., Mansour, H., Recht, B., & Herrmann, F. J. (2015). Efficient matrix completion for seismic data reconstruction. Geophysics, 80(5), V97-V114.

Da Silva, C., and F. J. Herrmann, 2015, Optimization on the hierarchical tucker manifold–applications to tensor completion: Linear Algebra and its Applications, 481, 131–173.

Seismic hierarchical Tucker

Given a frequency slice with coordinate (src x, src y, rec x, rec y), we introduce the **non-canonical dimension tree** for seismic data.



Non-canonical vs. canonical – 396 x 396 x 50 x 50 volume (~5.8 GB)

	Frequency (Hz)	Parameter Size	SNR	Compression Ratio
Non-canonical	3	71MB	42.8	98.8%
canonical	3	501MB	42.9	91.6%
Non-canonical	6	421MB	43.0	92.9%
canonical	6	1194MB	43.1	79.9%

Seismic hierarchical Tucker

We can compress low-frequency seismic data in HT in either case listed below

- **full data**

Seismic hierarchical Tucker

We can compress low-frequency seismic data in HT in either case listed below

- **full data** $\longrightarrow$ HT **truncation** algorithm detail in *Tobler, 2012*

SLIM

Seismic hierarchical Tucker

We can compress low-frequency seismic data in HT in either case listed below

- **full data** $\longrightarrow$ HT **truncation** algorithm detail in *Tobler, 2012*
- **missing data**

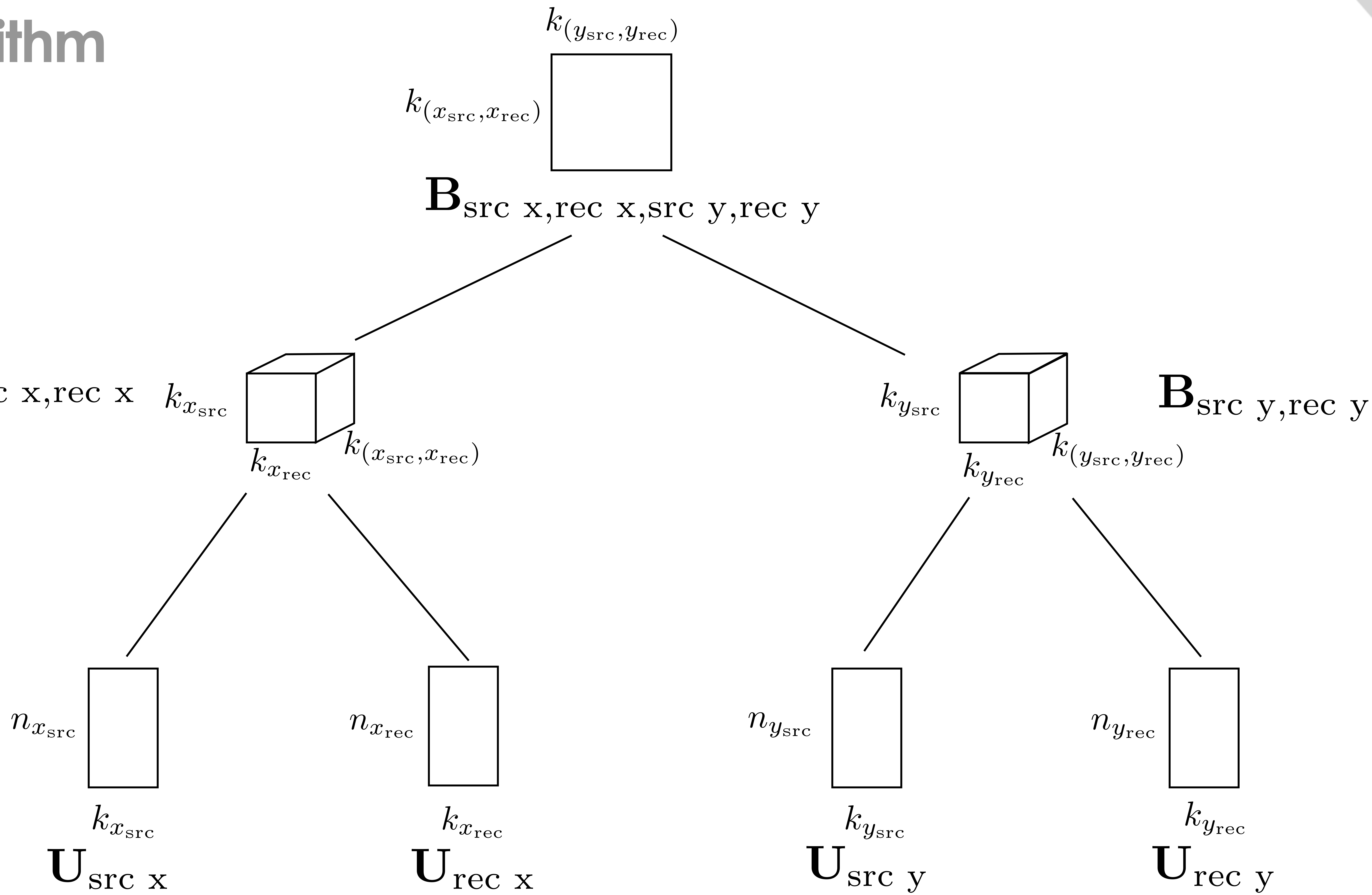
Seismic hierarchical Tucker

We can compress low-frequency seismic data in HT in either case listed below

- **full data** ⟶ HT **truncation** algorithm detail in *Tobler, 2012*
- **missing data** ⟶ **interpolate** HT format described in *Da Silva & Herrmann, 2015*

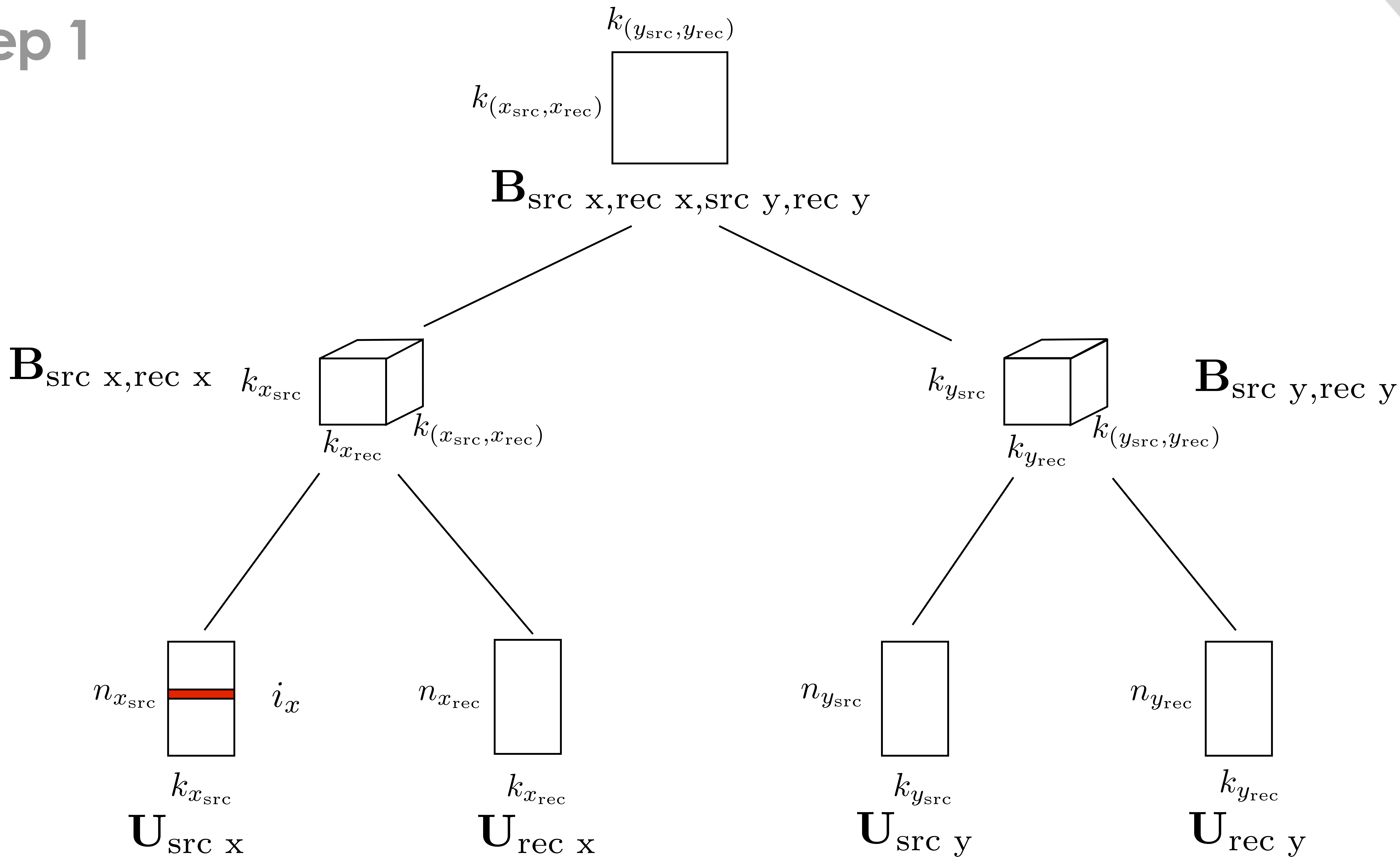
On-the-fly extraction of shots/receivers

Algorithm



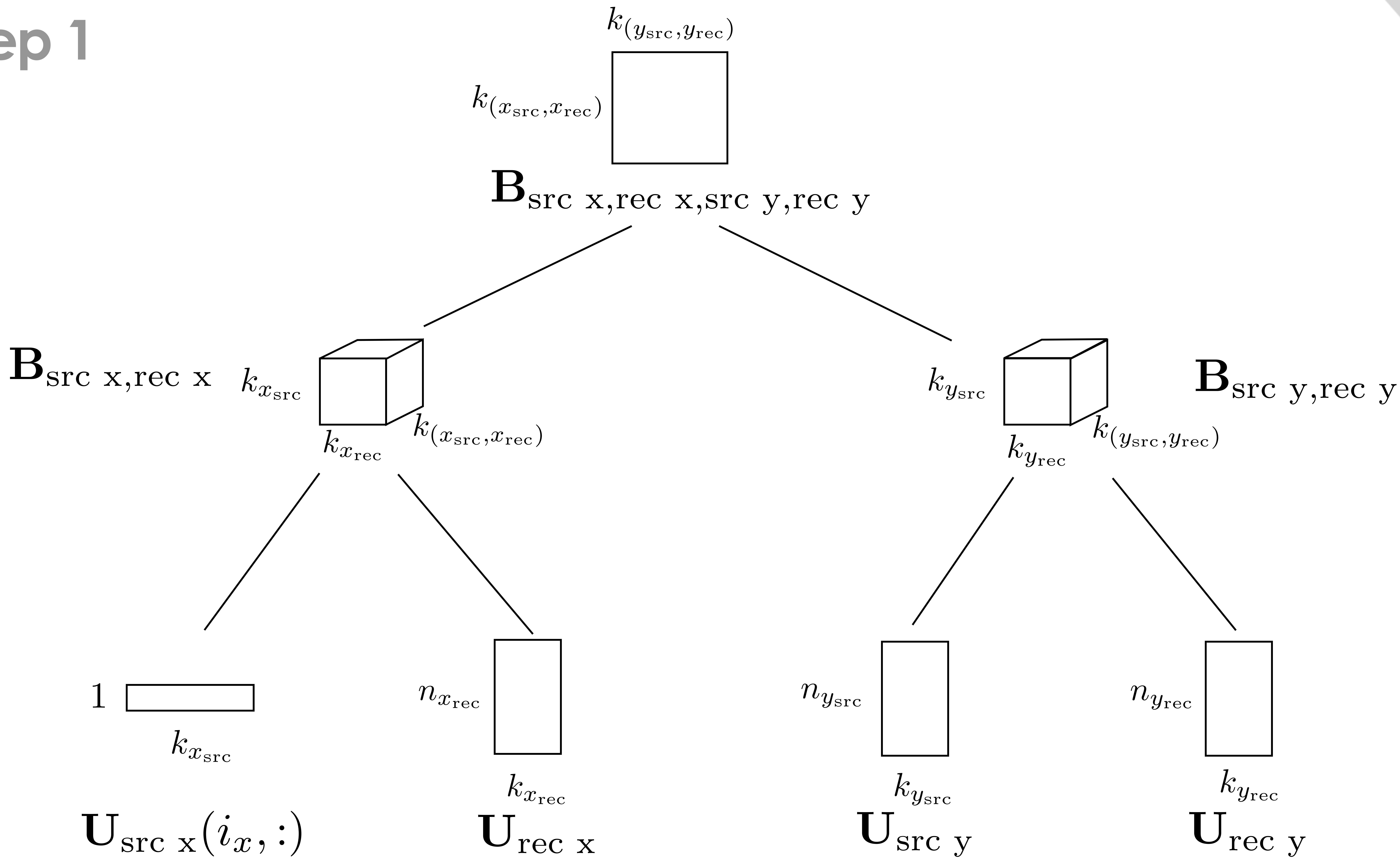
$k_{(y_{\mathrm{src}},y_{\mathrm{rec}})}$
$k_{(x_{\mathrm{src}},x_{\mathrm{rec}})}$
$\mathbf{B}_{\text{src x,rec x,src y,rec y}}$
$\mathbf{B}_{\text{src x,rec x}}$
$k_{x_{\mathrm{src}}}$
$k_{(x_{\mathrm{src}},x_{\mathrm{rec}})}$
$k_{x_{\mathrm{rec}}}$
$k_{y_{\mathrm{src}}}$
$\mathbf{B}_{\text{src y,rec y}}$
$k_{(y_{\mathrm{src}},y_{\mathrm{rec}})}$
$k_{y_{\mathrm{rec}}}$
$n_{x_{\mathrm{src}}}$
$k_{x_{\mathrm{src}}}$
$\mathbf{U}_{\text{src x}}$
$n_{x_{\mathrm{rec}}}$
$k_{x_{\mathrm{rec}}}$
$\mathbf{U}_{\text{rec x}}$
$n_{y_{\mathrm{src}}}$
$k_{y_{\mathrm{src}}}$
$\mathbf{U}_{\text{src y}}$
$n_{y_{\mathrm{rec}}}$
$k_{y_{\mathrm{rec}}}$
$\mathbf{U}_{\text{rec y}}$

Step 1



$k_{(y_{\text{src}}, y_{\text{rec}})}$
$k_{(x_{\text{src}}, x_{\text{rec}})}$
$\mathbf{B}_{\text{src x,rec x,src y,rec y}}$
$\mathbf{B}_{\text{src x,rec x}}$
$k_{x_{\text{src}}}$
$k_{x_{\text{rec}}}$
$k_{(x_{\text{src}}, x_{\text{rec}})}$
$k_{y_{\text{src}}}$
$\mathbf{B}_{\text{src y,rec y}}$
$k_{y_{\text{rec}}}$
$k_{(y_{\text{src}}, y_{\text{rec}})}$
$n_{x_{\text{src}}}$
i_x
$n_{x_{\text{rec}}}$
$n_{y_{\text{src}}}$
$n_{y_{\text{rec}}}$
$k_{x_{\text{src}}}$
$\mathbf{U}_{\text{src x}}$
$k_{x_{\text{rec}}}$
$\mathbf{U}_{\text{rec x}}$
$k_{y_{\text{src}}}$
$\mathbf{U}_{\text{src y}}$
$k_{y_{\text{rec}}}$
$\mathbf{U}_{\text{rec y}}$

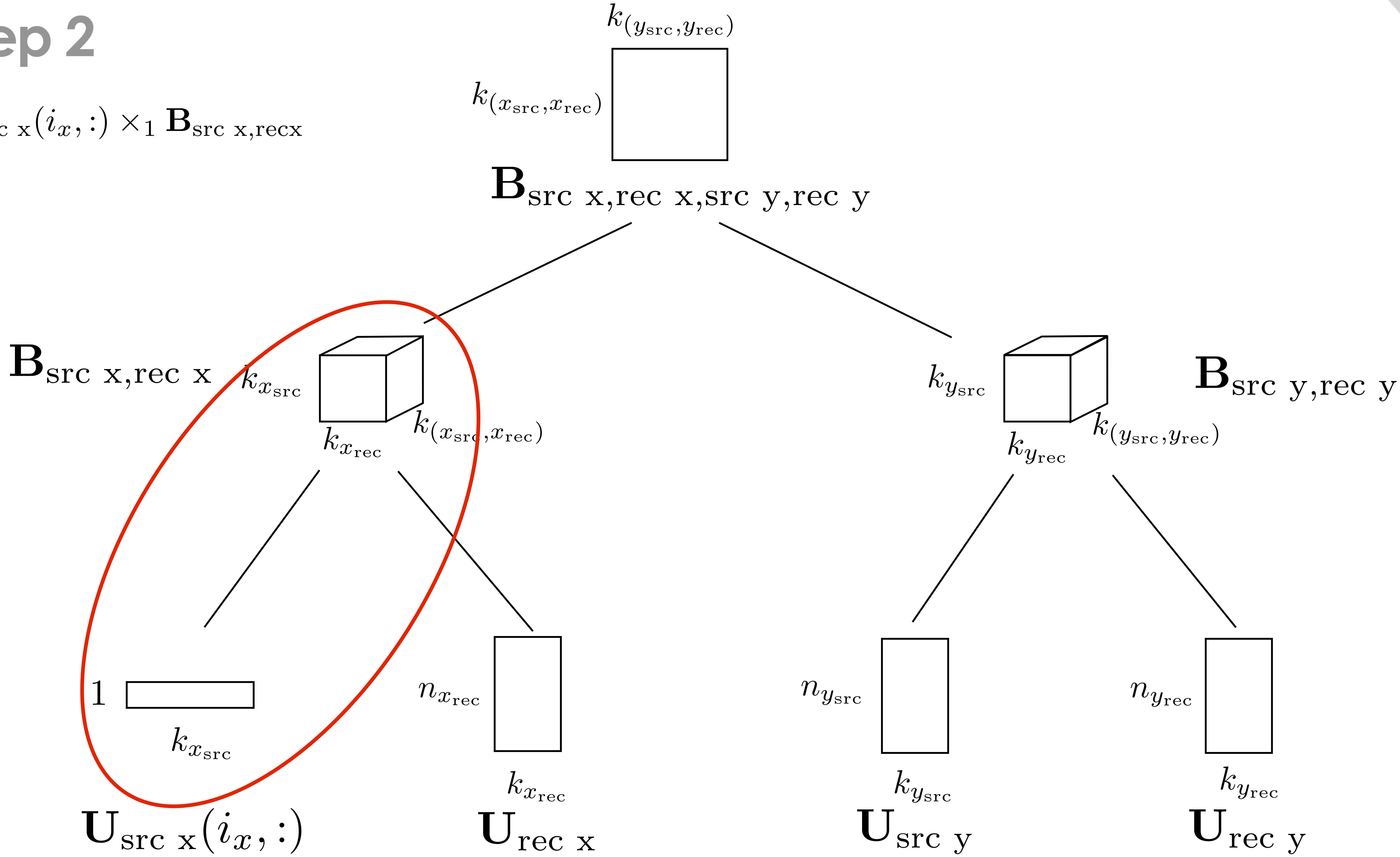
Step 1



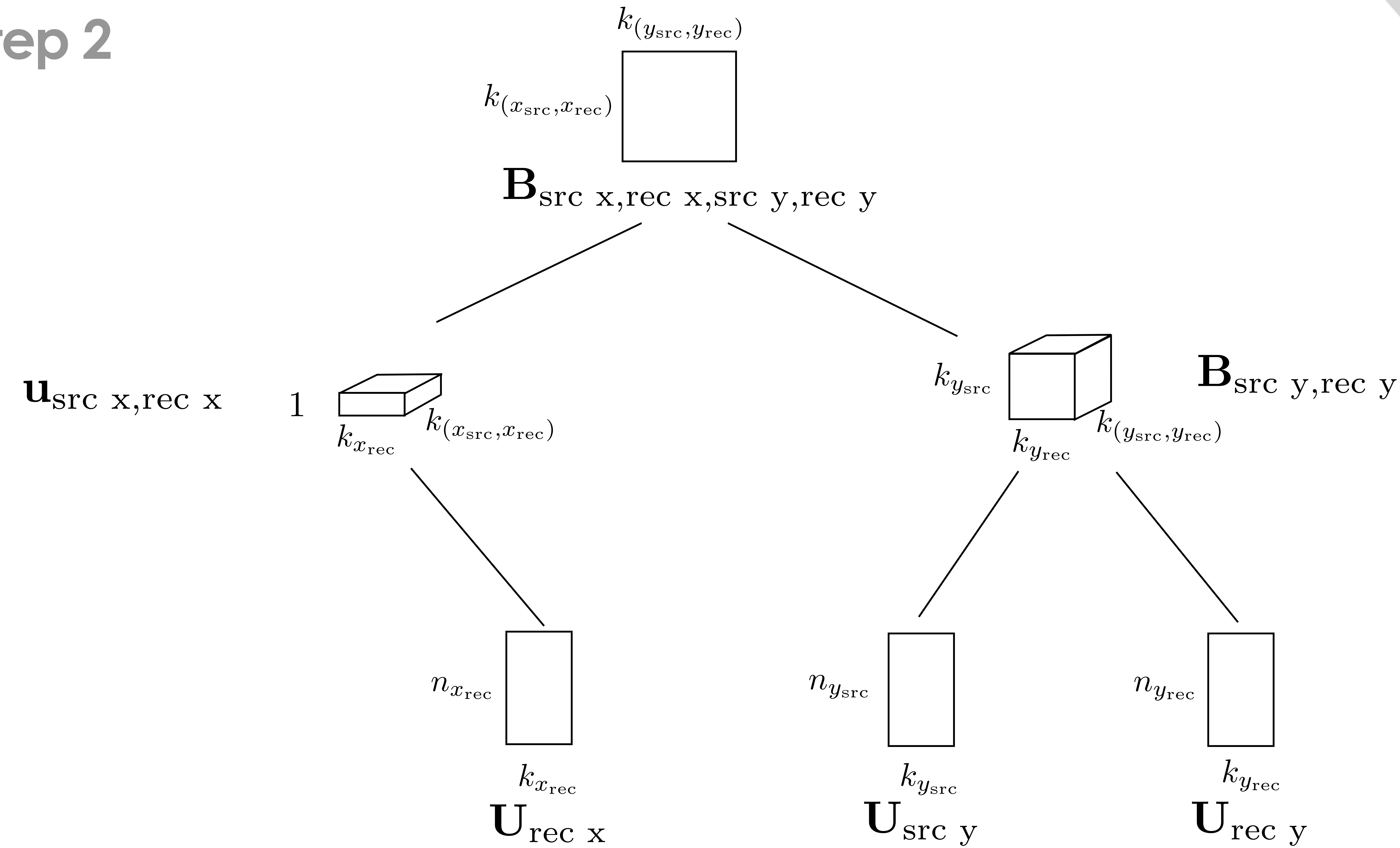
$k_{(y_{\mathrm{src}}, y_{\mathrm{rec}})}$
$k_{(x_{\mathrm{src}}, x_{\mathrm{rec}})}$
$\mathbf{B}_{\text{src x,rec x,src y,rec y}}$
$\mathbf{B}_{\text{src x,rec x}}$
$k_{x_{\mathrm{src}}}$
$k_{x_{\mathrm{rec}}}$
$k_{(x_{\mathrm{src}}, x_{\mathrm{rec}})}$
$k_{y_{\mathrm{src}}}$
$\mathbf{B}_{\text{src y,rec y}}$
$k_{y_{\mathrm{rec}}}$
$k_{(y_{\mathrm{src}}, y_{\mathrm{rec}})}$
1
$k_{x_{\mathrm{src}}}$
$n_{x_{\mathrm{rec}}}$
$k_{x_{\mathrm{rec}}}$
$n_{y_{\mathrm{src}}}$
$k_{y_{\mathrm{src}}}$
$n_{y_{\mathrm{rec}}}$
$k_{y_{\mathrm{rec}}}$
$\mathbf{U}_{\text{src x}}(i_x, :)$
$\mathbf{U}_{\text{rec x}}$
$\mathbf{U}_{\text{src y}}$
$\mathbf{U}_{\text{rec y}}$

Step 2

$\mathbf{U}_{\text{src x}}(i_x,:) \times_1 \mathbf{B}_{\text{src x,recx}}$



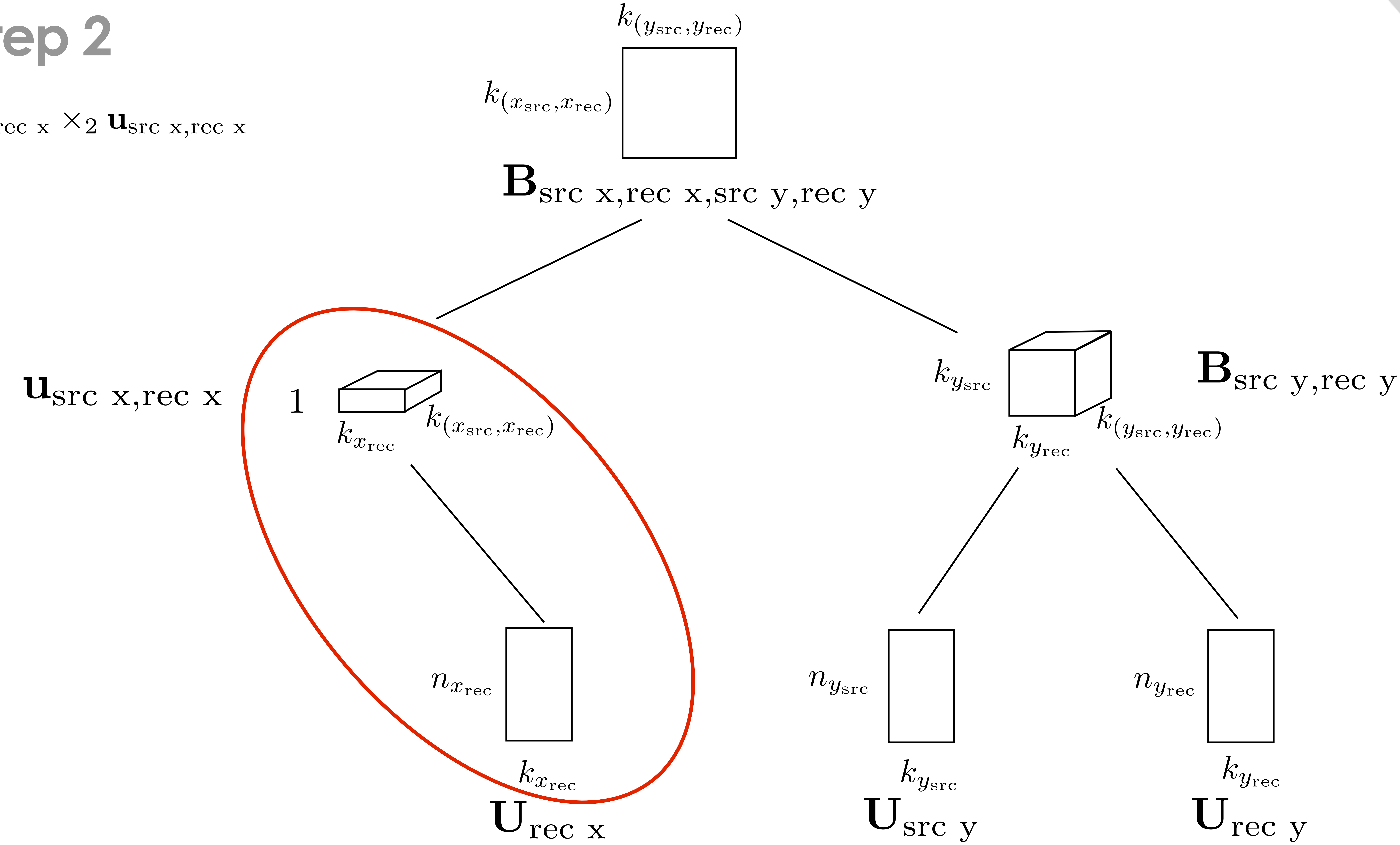
Step 2



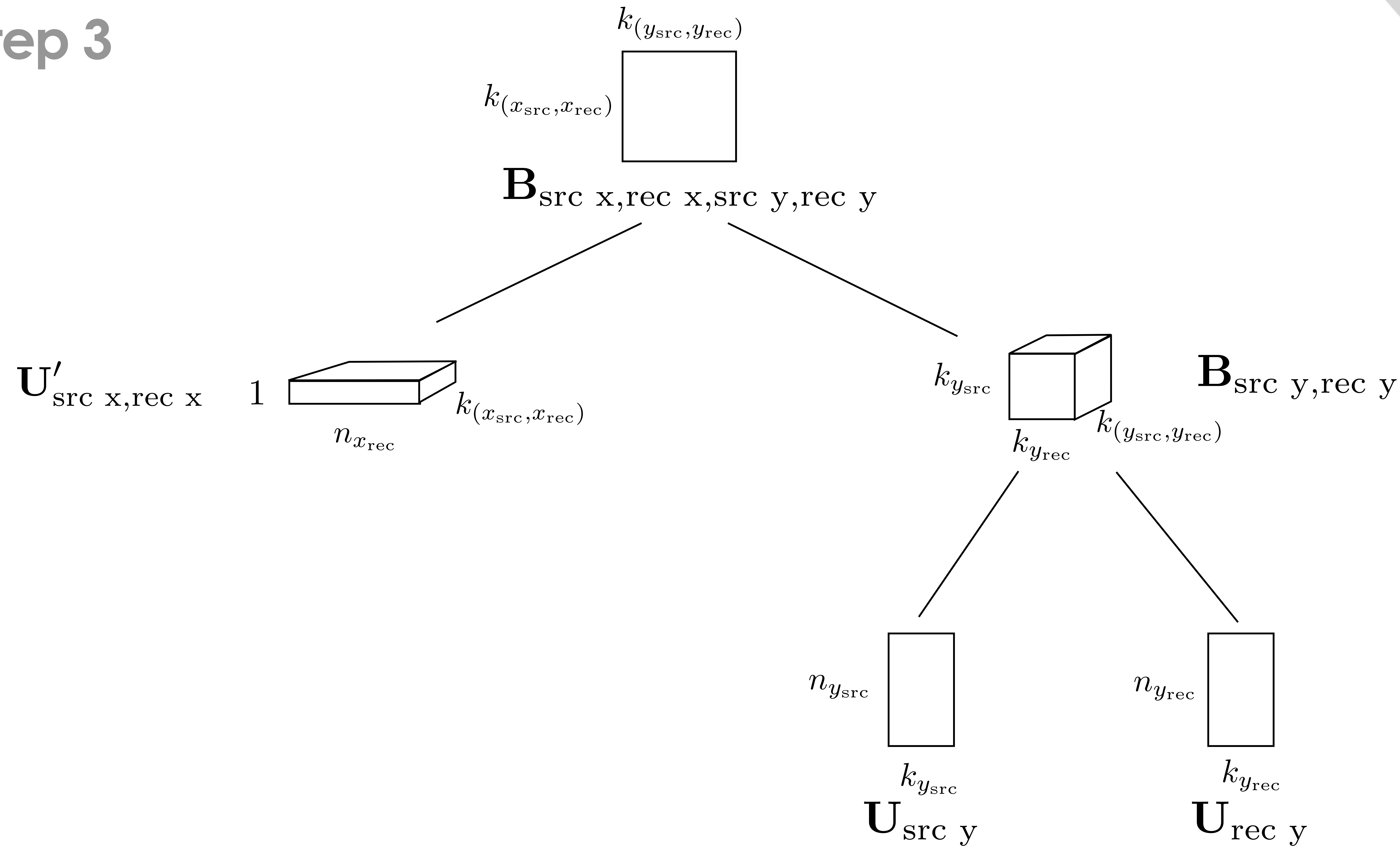
$k_{(y_{\mathrm{src}},y_{\mathrm{rec}})}$
$k_{(x_{\mathrm{src}},x_{\mathrm{rec}})}$
$\mathbf{B}_{\text{src x,rec x,src y,rec y}}$
$\mathbf{u}_{\text{src x,rec x}}$
1
$k_{x_{\mathrm{rec}}}$
$k_{(x_{\mathrm{src}},x_{\mathrm{rec}})}$
$k_{y_{\mathrm{src}}}$
$k_{y_{\mathrm{rec}}}$
$k_{(y_{\mathrm{src}},y_{\mathrm{rec}})}$
$\mathbf{B}_{\text{src y,rec y}}$
$n_{x_{\mathrm{rec}}}$
$k_{x_{\mathrm{rec}}}$
$\mathbf{U}_{\text{rec x}}$
$n_{y_{\mathrm{src}}}$
$k_{y_{\mathrm{src}}}$
$\mathbf{U}_{\text{src y}}$
$n_{y_{\mathrm{rec}}}$
$k_{y_{\mathrm{rec}}}$
$\mathbf{U}_{\text{rec y}}$

Step 2

$\mathbf{U}_{\text{rec x}} \times_2 \mathbf{u}_{\text{src x,rec x}}$



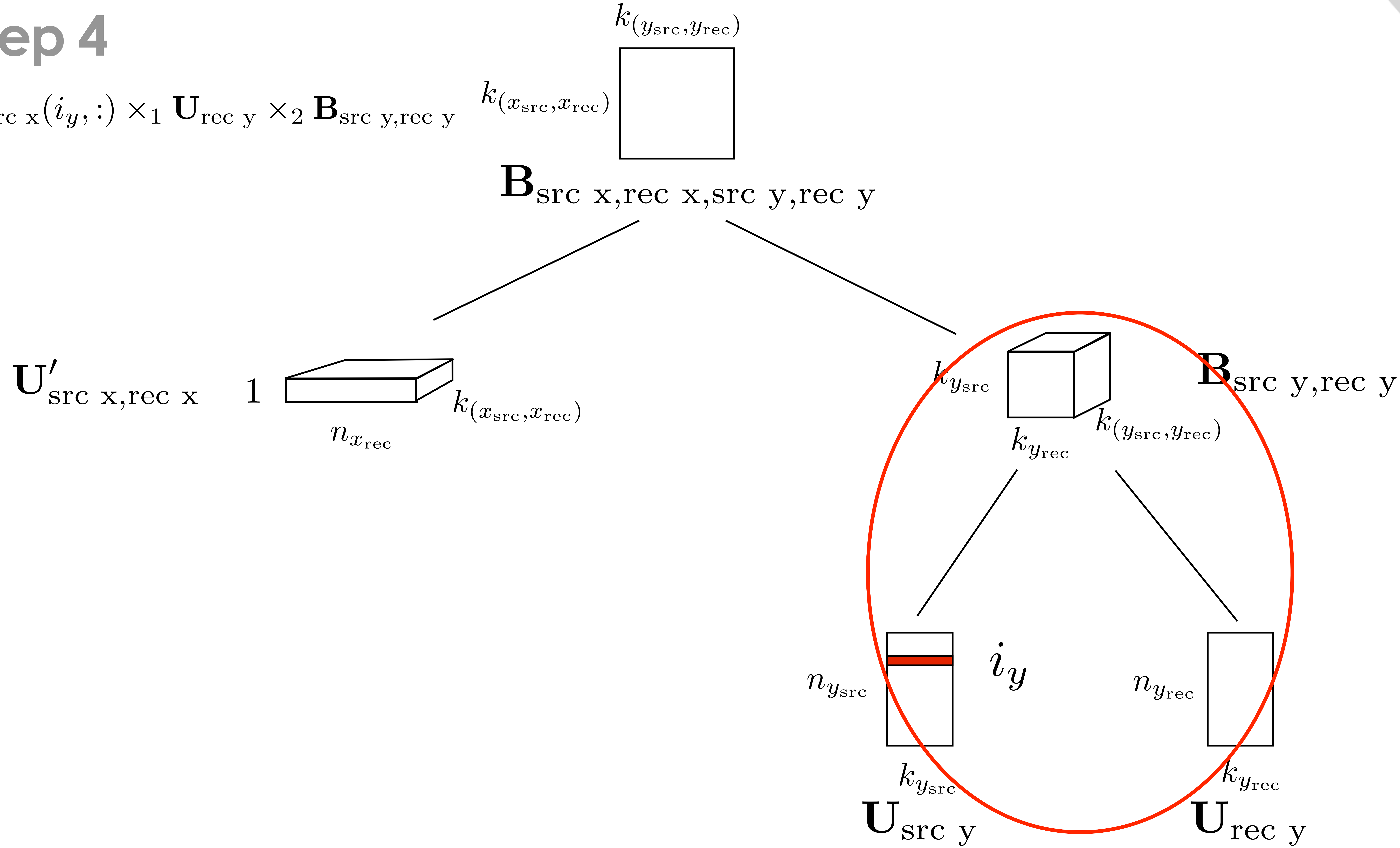
Step 3



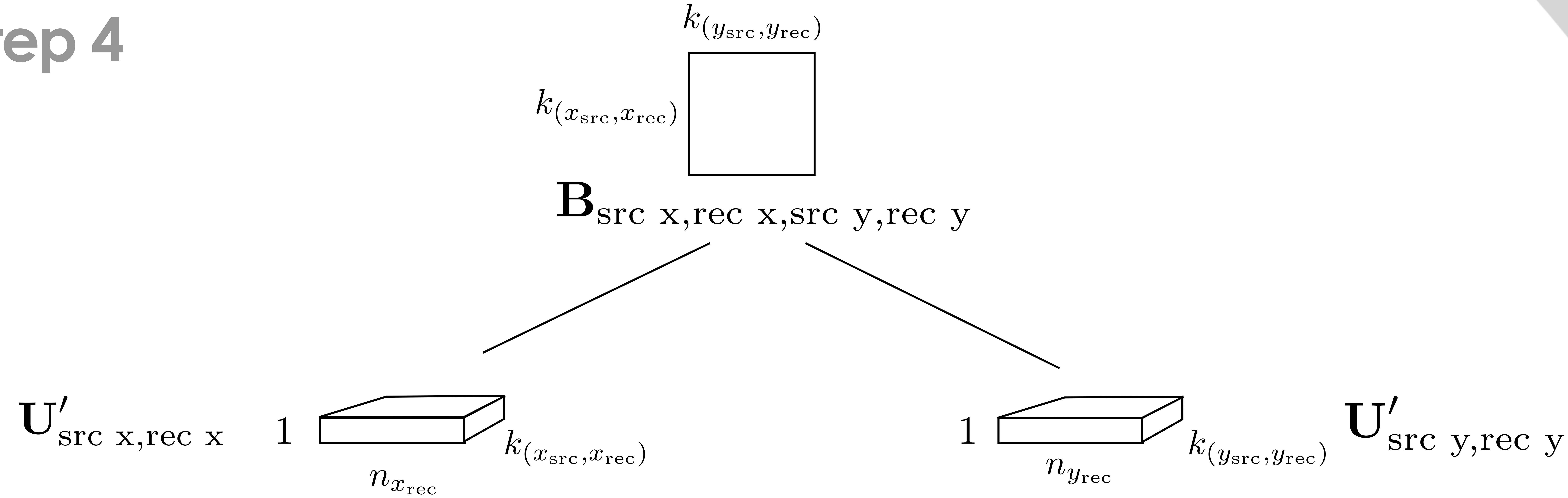
$k_{(y_{\text{src}},y_{\text{rec}})}$
$k_{(x_{\text{src}},x_{\text{rec}})}$
$\mathbf{B}_{\text{src x,rec x,src y,rec y}}$
$\mathbf{U}'_{\text{src x,rec x}}$
1
$k_{(x_{\text{src}},x_{\text{rec}})}$
$n_{x_{\text{rec}}}$
$k_{y_{\text{src}}}$
$\mathbf{B}_{\text{src y,rec y}}$
$k_{(y_{\text{src}},y_{\text{rec}})}$
$k_{y_{\text{rec}}}$
$n_{y_{\text{src}}}$
$n_{y_{\text{rec}}}$
$k_{y_{\text{src}}}$
$k_{y_{\text{rec}}}$
$\mathbf{U}_{\text{src y}}$
$\mathbf{U}_{\text{rec y}}$

Step 4

$\mathbf{U}_{\text{src x}}(i_y,:) \times_1 \mathbf{U}_{\text{rec y}} \times_2 \mathbf{B}_{\text{src y,rec y}}$



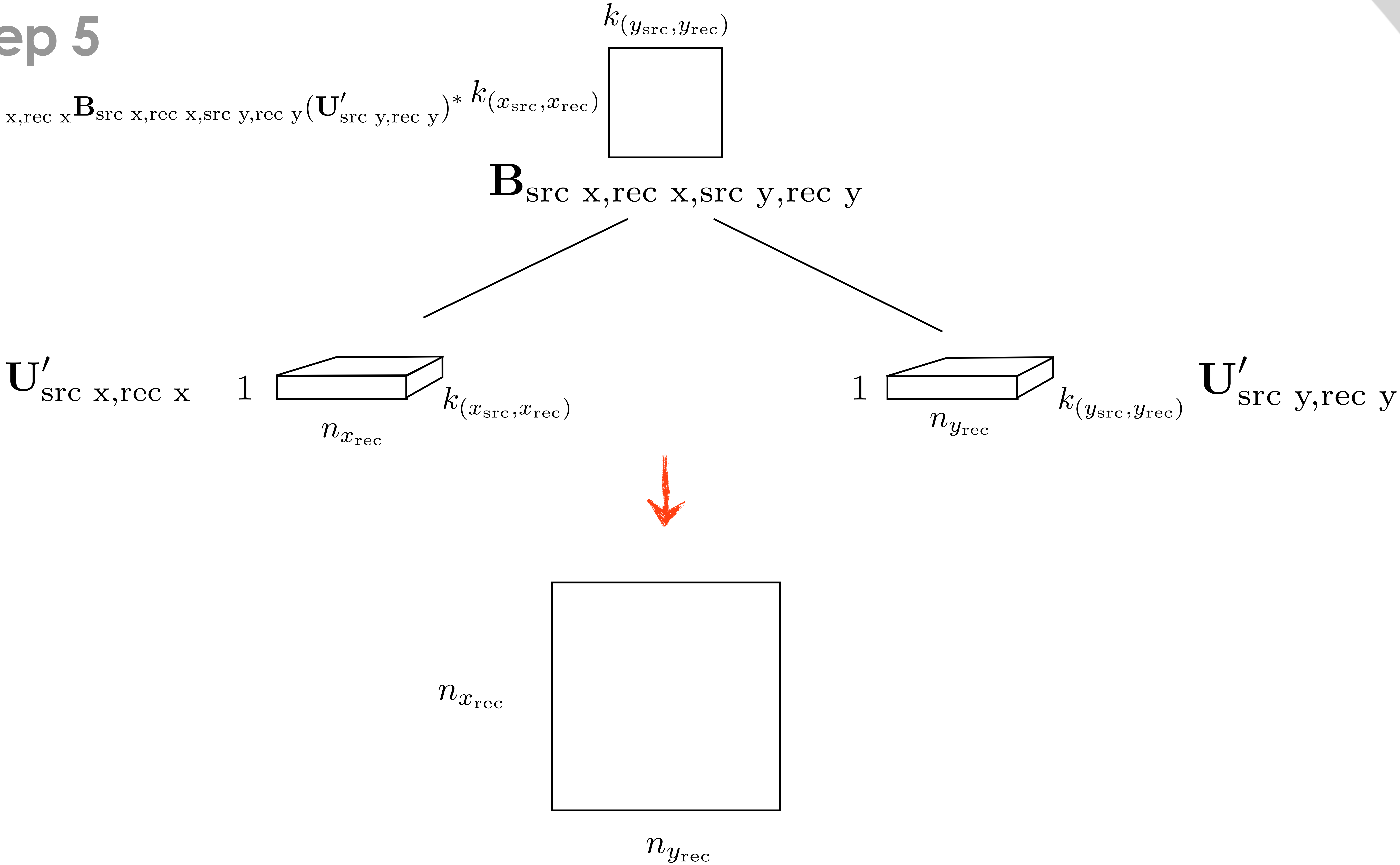
Step 4



$k_{(y_{\text{src}},y_{\text{rec}})}$
$k_{(x_{\text{src}},x_{\text{rec}})}$
$\mathbf{B}_{\text{src x,rec x,src y,rec y}}$
$\mathbf{U}'_{\text{src x,rec x}}$
1
$k_{(x_{\text{src}},x_{\text{rec}})}$
$n_{x_{\text{rec}}}$
1
$n_{y_{\text{rec}}}$
$k_{(y_{\text{src}},y_{\text{rec}})}$
$\mathbf{U}'_{\text{src y,rec y}}$

Step 5

$\mathbf{U}'_{\text{src x,rec x}}\mathbf{B}_{\text{src x,rec x,src y,rec y}}(\mathbf{U}'_{\text{src y,rec y}})^*$



On-the-fly extraction of shots/receivers

Once our data in HT representation

- Kronecker product or matrix-matrix multiplication
- intermediate qualities **much smaller** than ambient dimensionality
- extract common **receiver gather** in an analogous way
- compute **simultaneous** shots/receivers gathers

Case study 1: 3D FWI

FWI examples

3D FWI with stochastic optimization algorithm

- a **subset** of full shots per iteration
- partially minimize least-square objective function
- **LBFGS** w/ bound constrains, i.e. **minimum & maximum** velocities allowed
- **single freq.** inverted at a time

FWI examples

For HT compressed data

- over **90%** reduced data volume in size
- cheaply store **compressed** form of the full data on every node
- automatically determine **indices** for shots per iteration
- **query-based** access to the data volume on-the-fly w/ our proposed algorithm

FWI examples

Computational environment:

- SENAI Yemoja cluster
- 50 nodes, 256GB RAM each, 20 CPU cores
- 8 Parallel Matlab workers per node
- modeling code, WAVEFORM (Da Silva and Herrmann, 2016)

FWI examples on Overthrust model

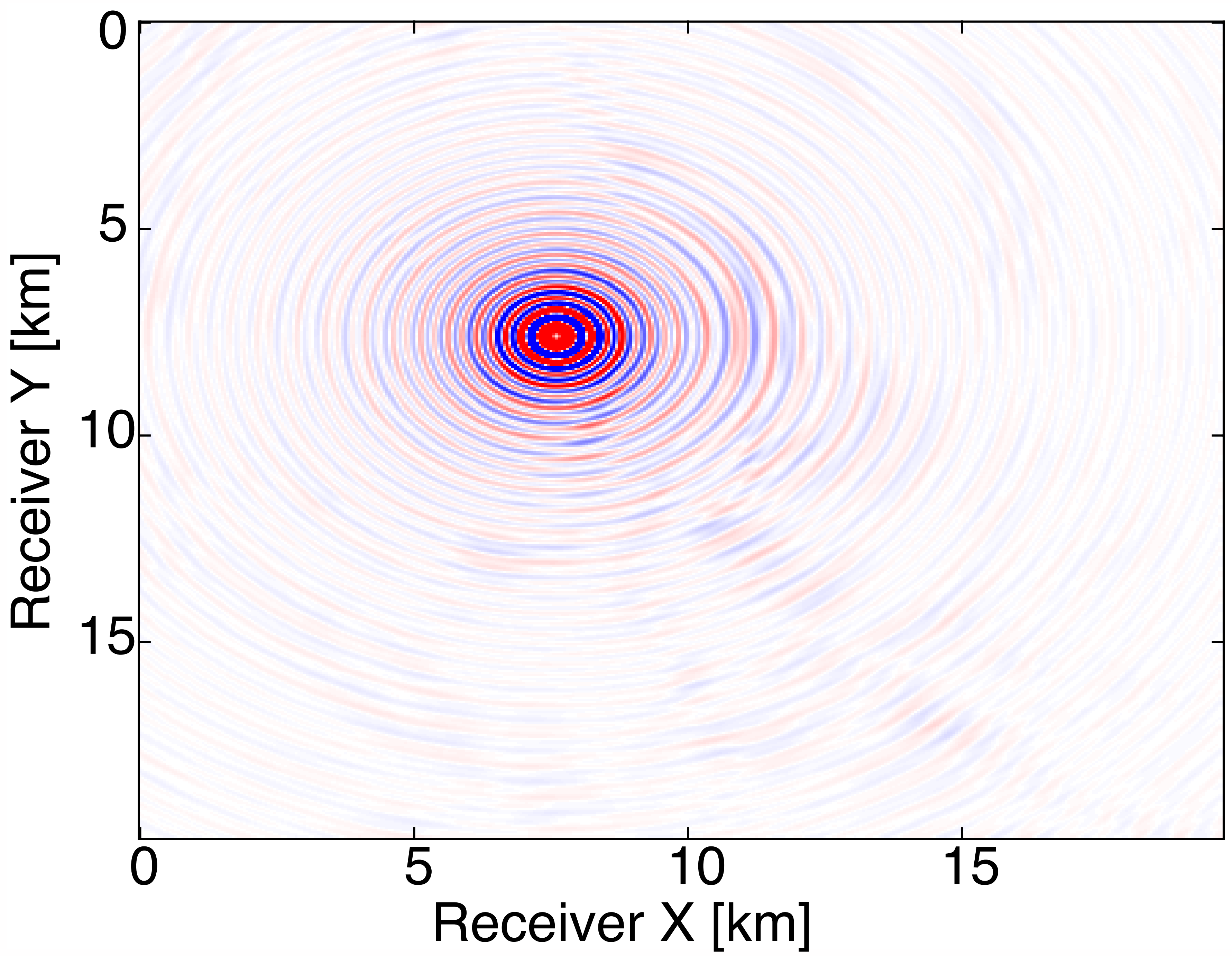
Model 1:

- 3D **Overthrust** model
- 20km x 20km x 4.6 km, adding 500m water
- 50m x 50m x 50 m spacing

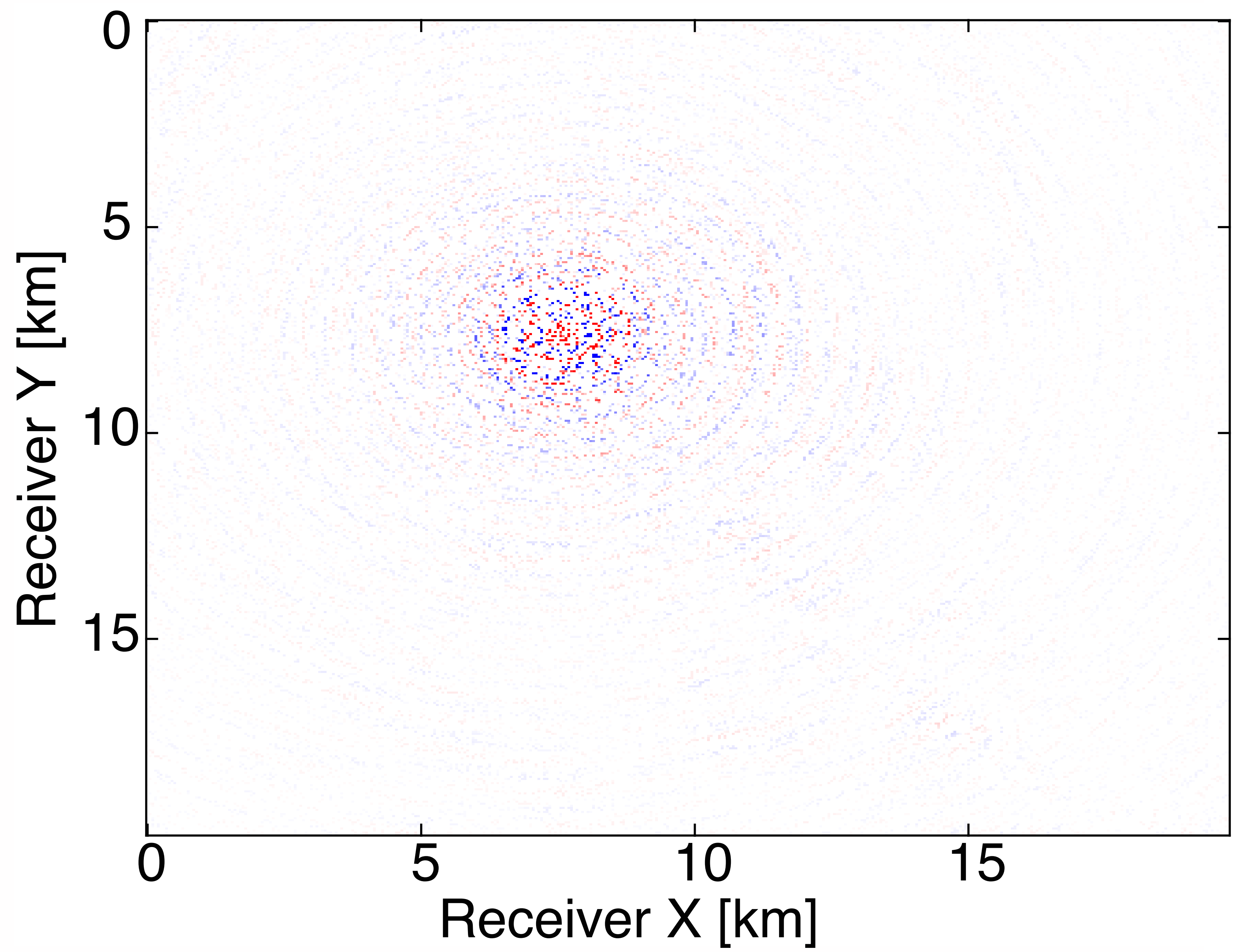
Data:

- 50 x 50 sources, 200m interval
- 396 x 396 receivers, 50m interval
- Ricker wavelet, 10Hz peak frequency
- 3Hz - 6Hz ranging, 1Hz interval
- **remove 80%** of random receivers

FWI examples on Overthrust model

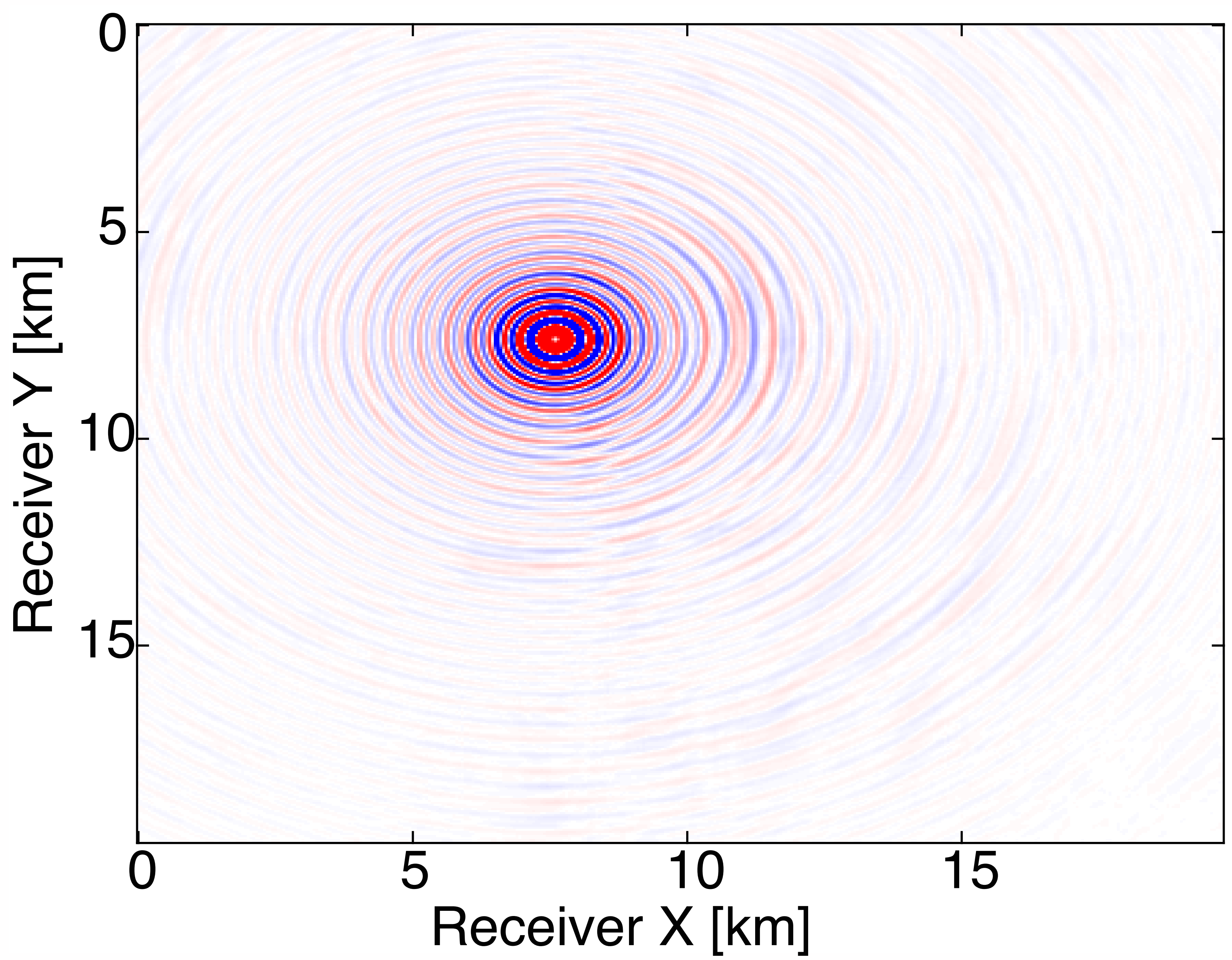


True data

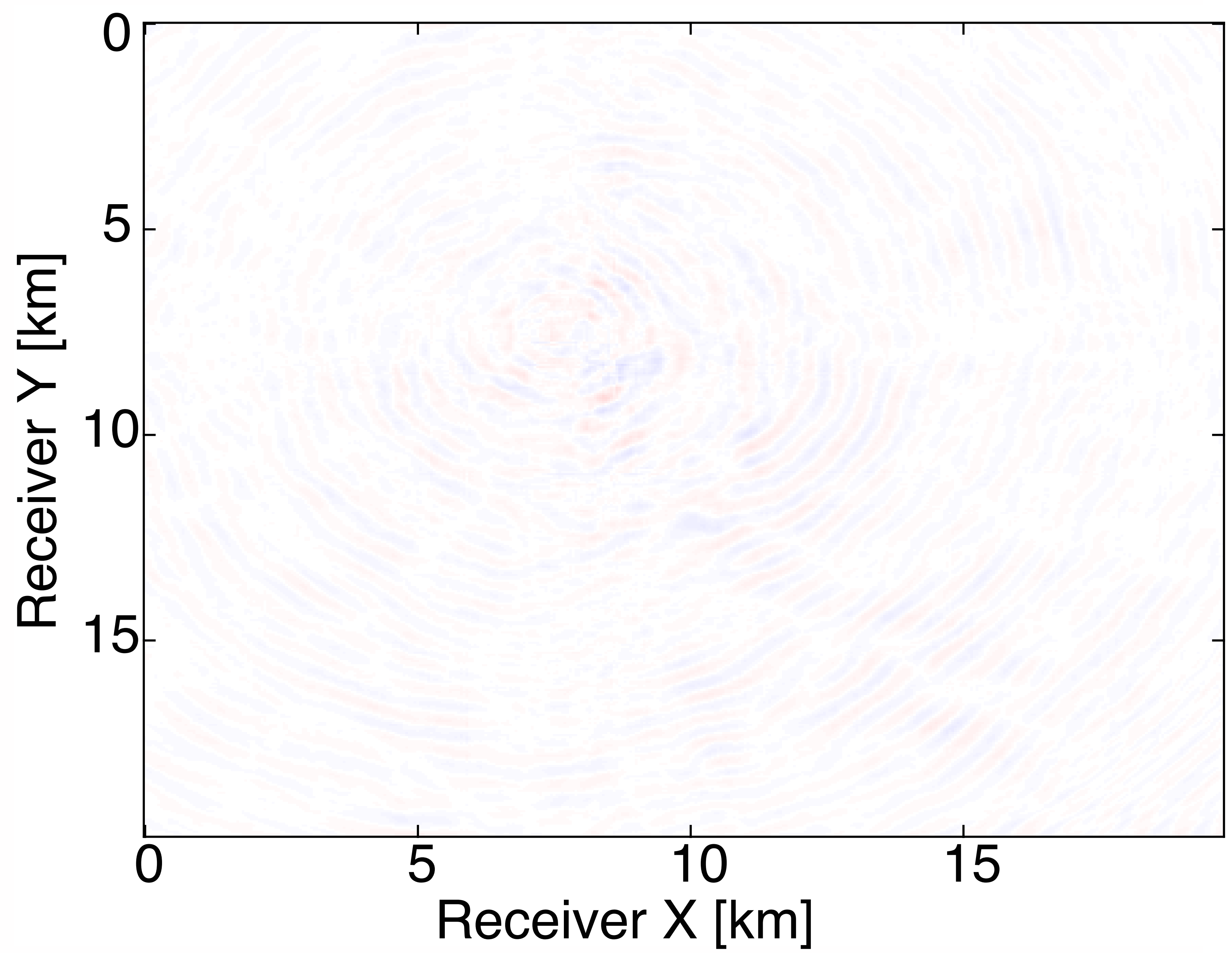


Missing 80% data

FWI examples on Overthrust model



Extract from compressed data



Residual

FWI examples on Overthrust model

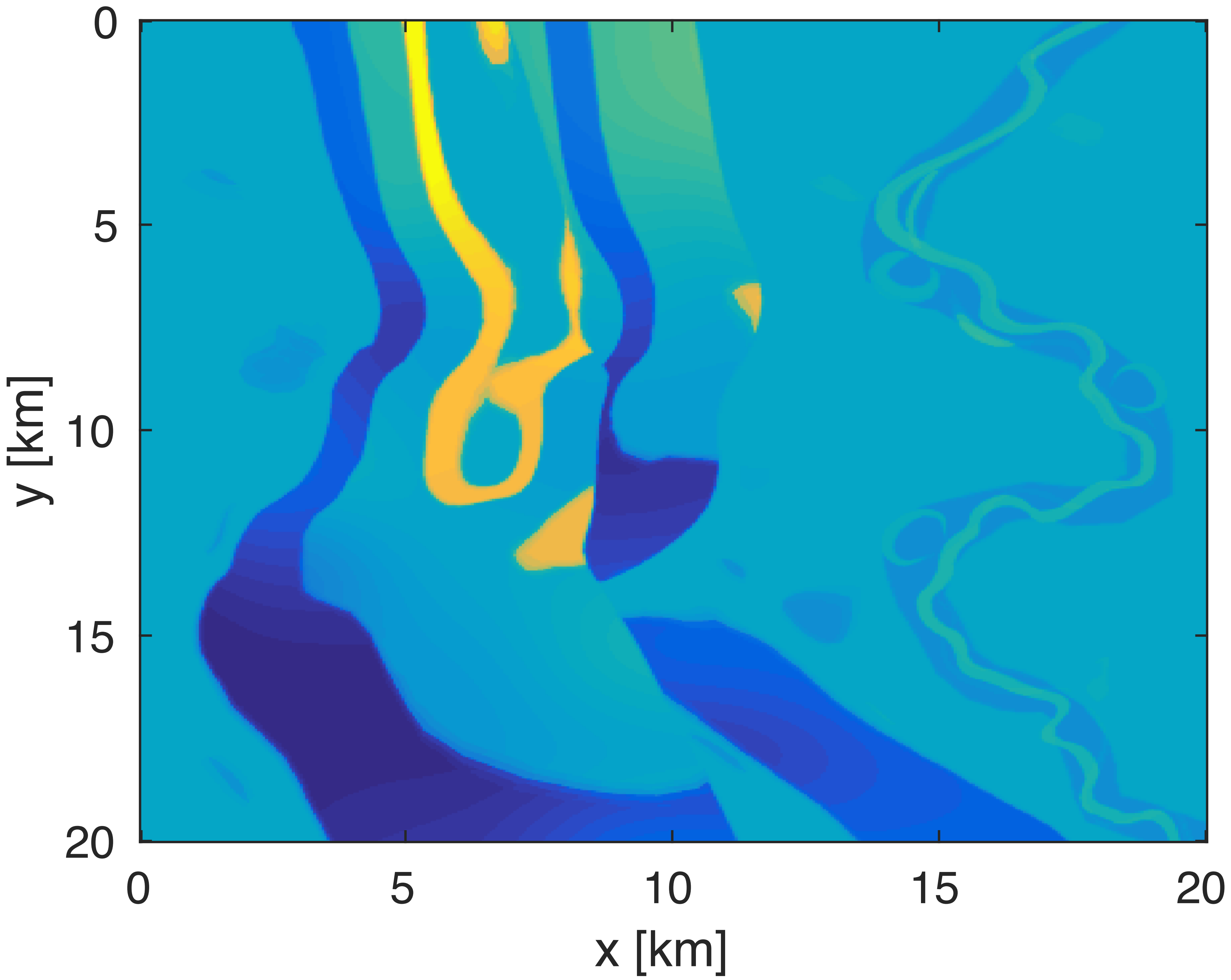
Stochastic FWI results inverted w/

- **full** data
- compressed **HT parameters** recovered from interpolation

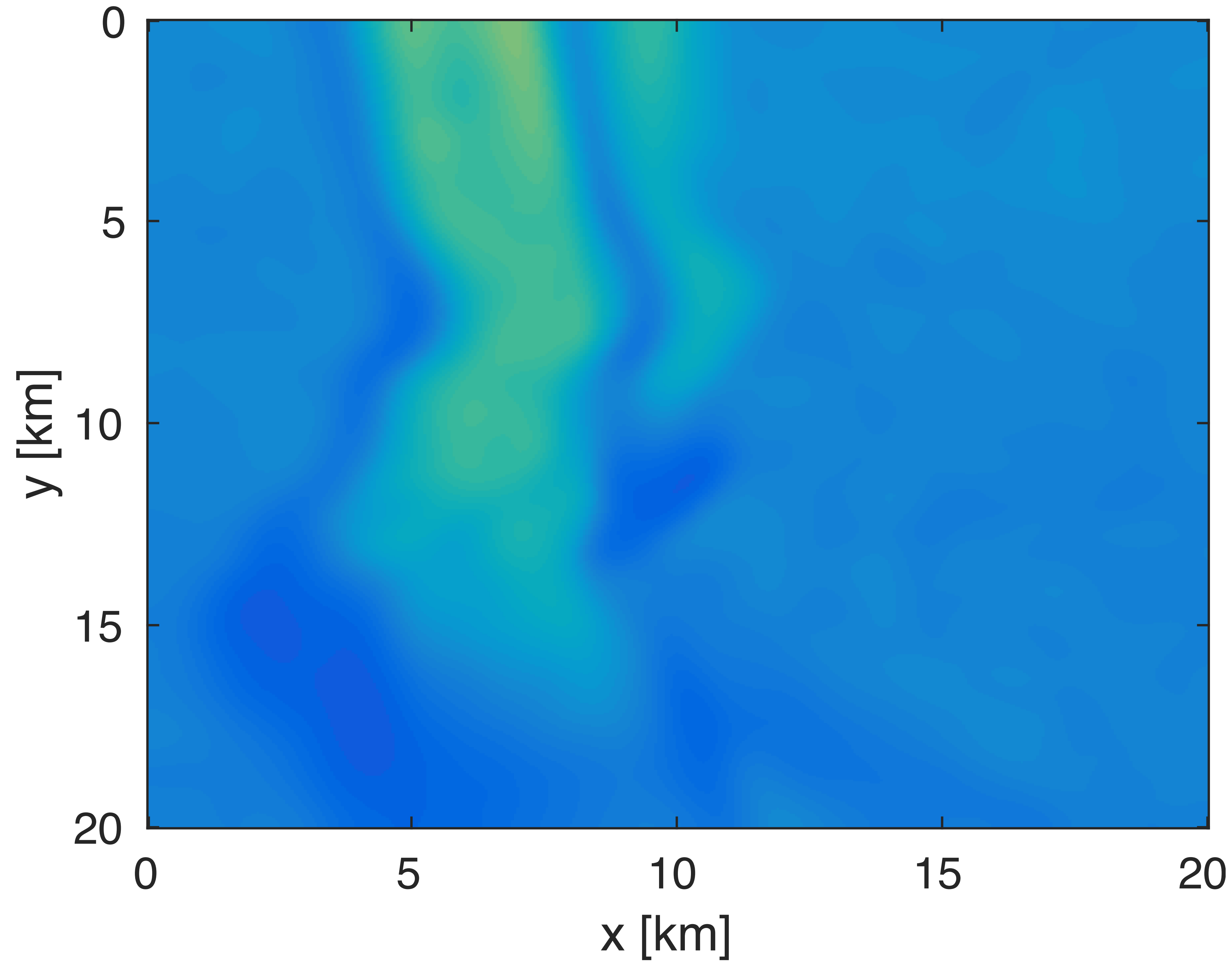
Same source indices for two examples

- **same** number of **PDE** solves
- **three** passes through the data

Z = 1000m depth slice

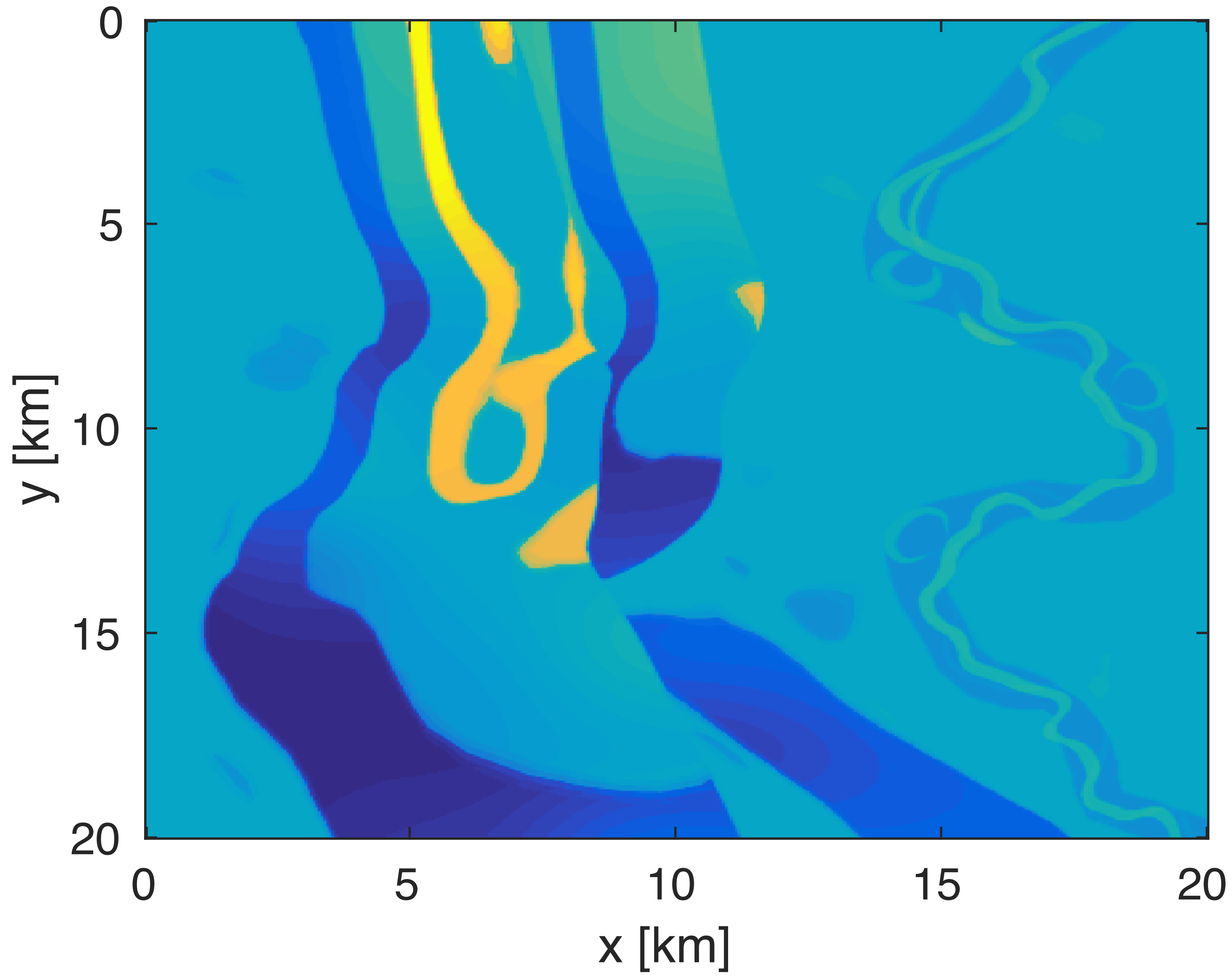


True model

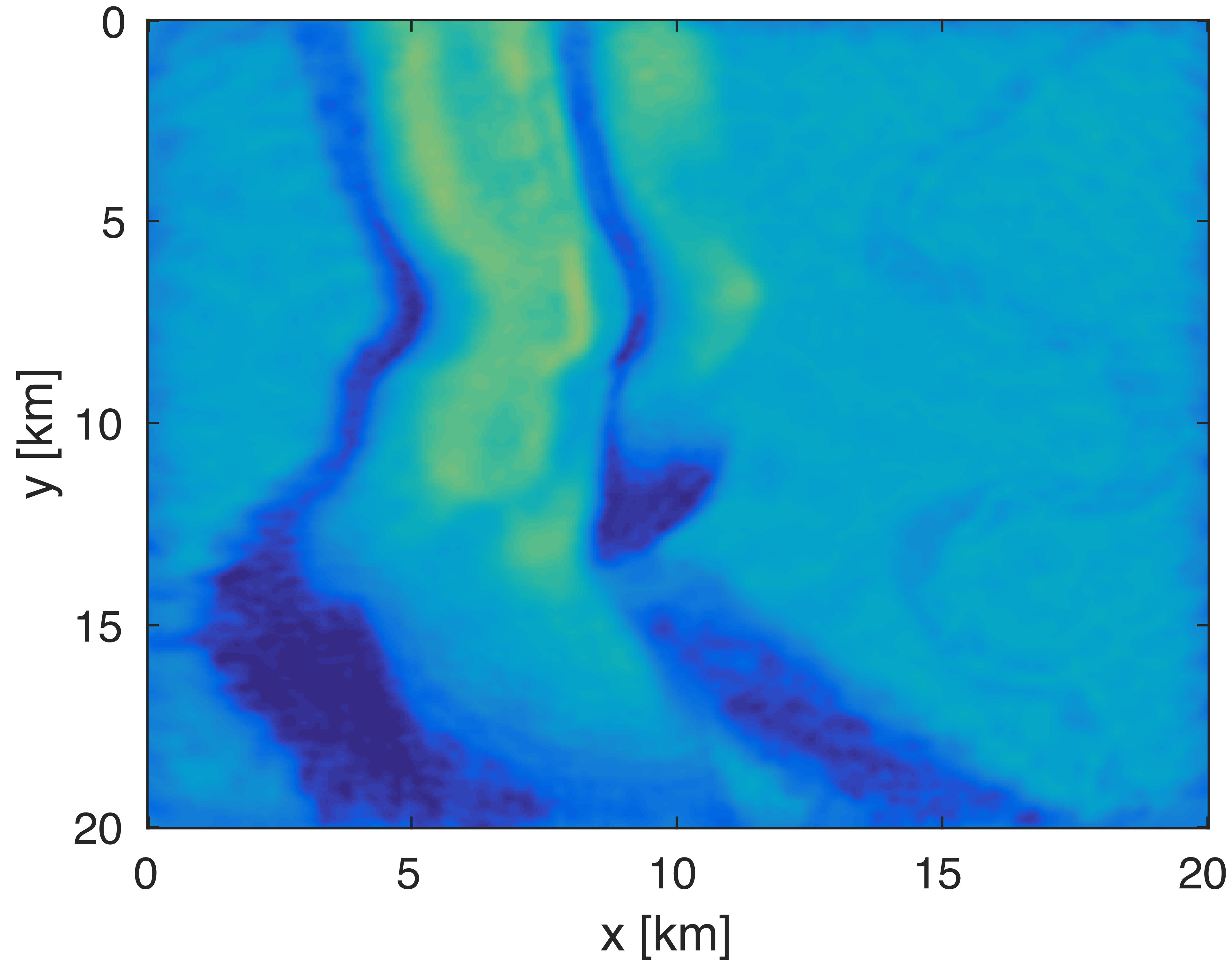


Initial model

Z = 1000m depth slice

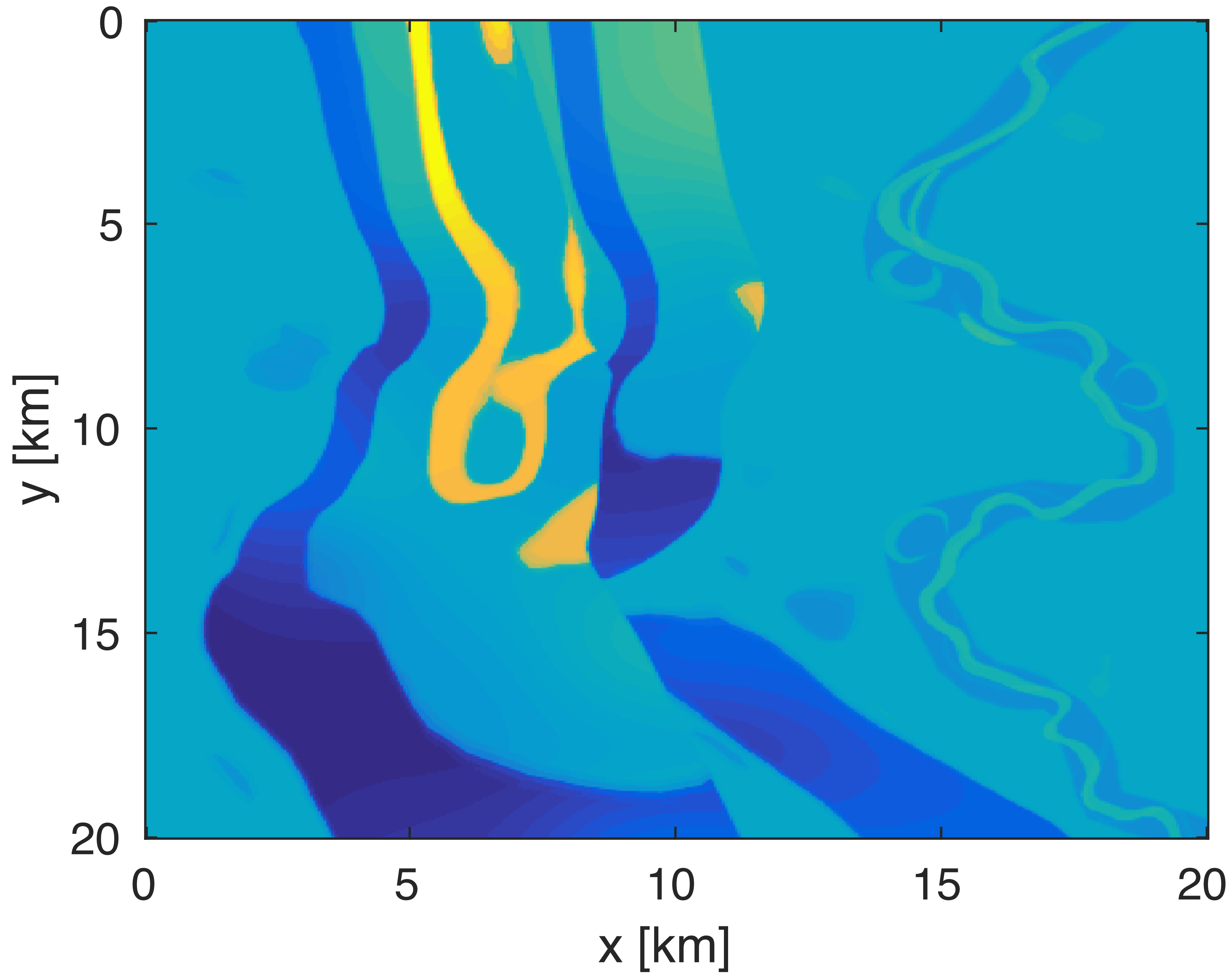


True model

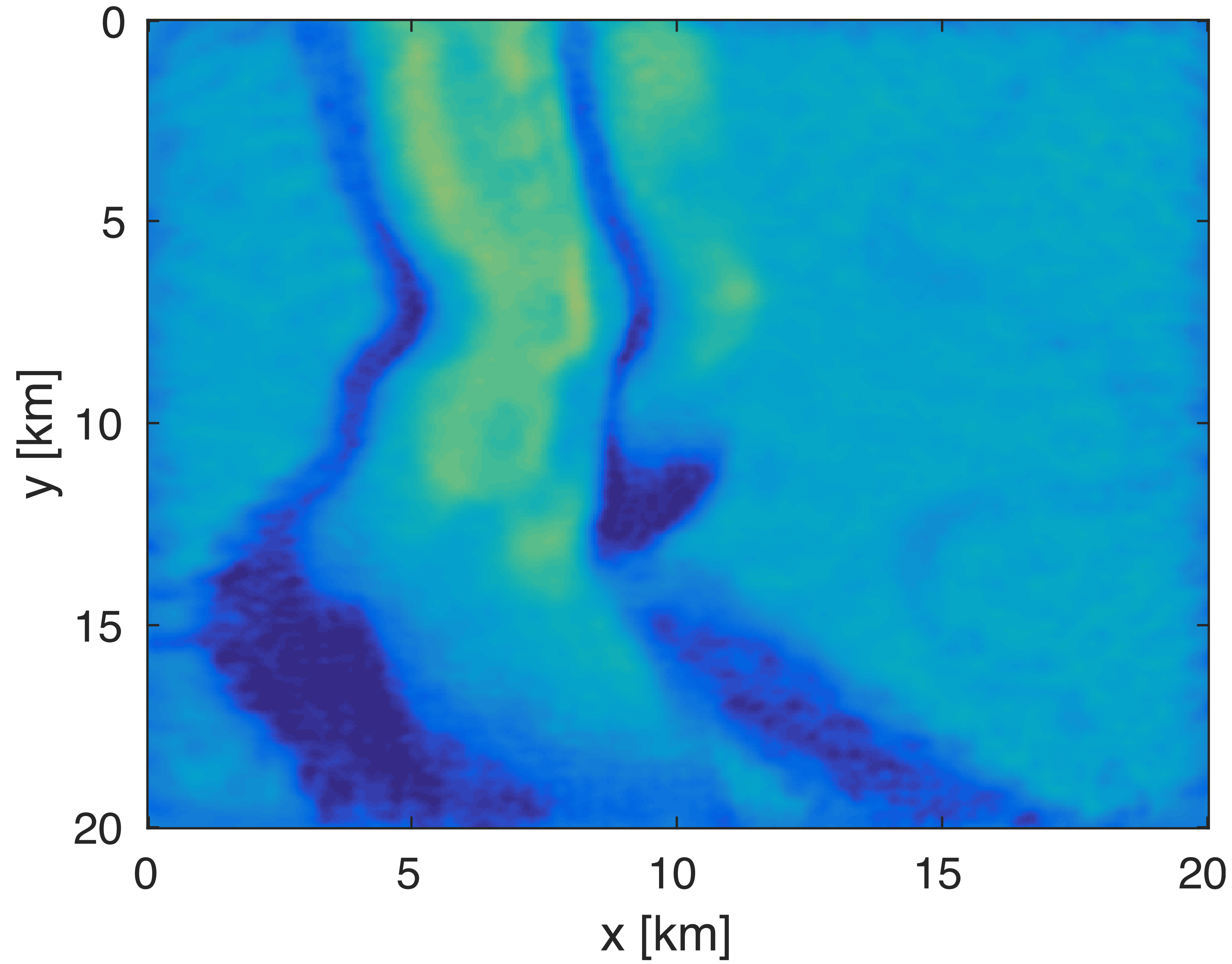


Full data

Z = 1000m depth slice

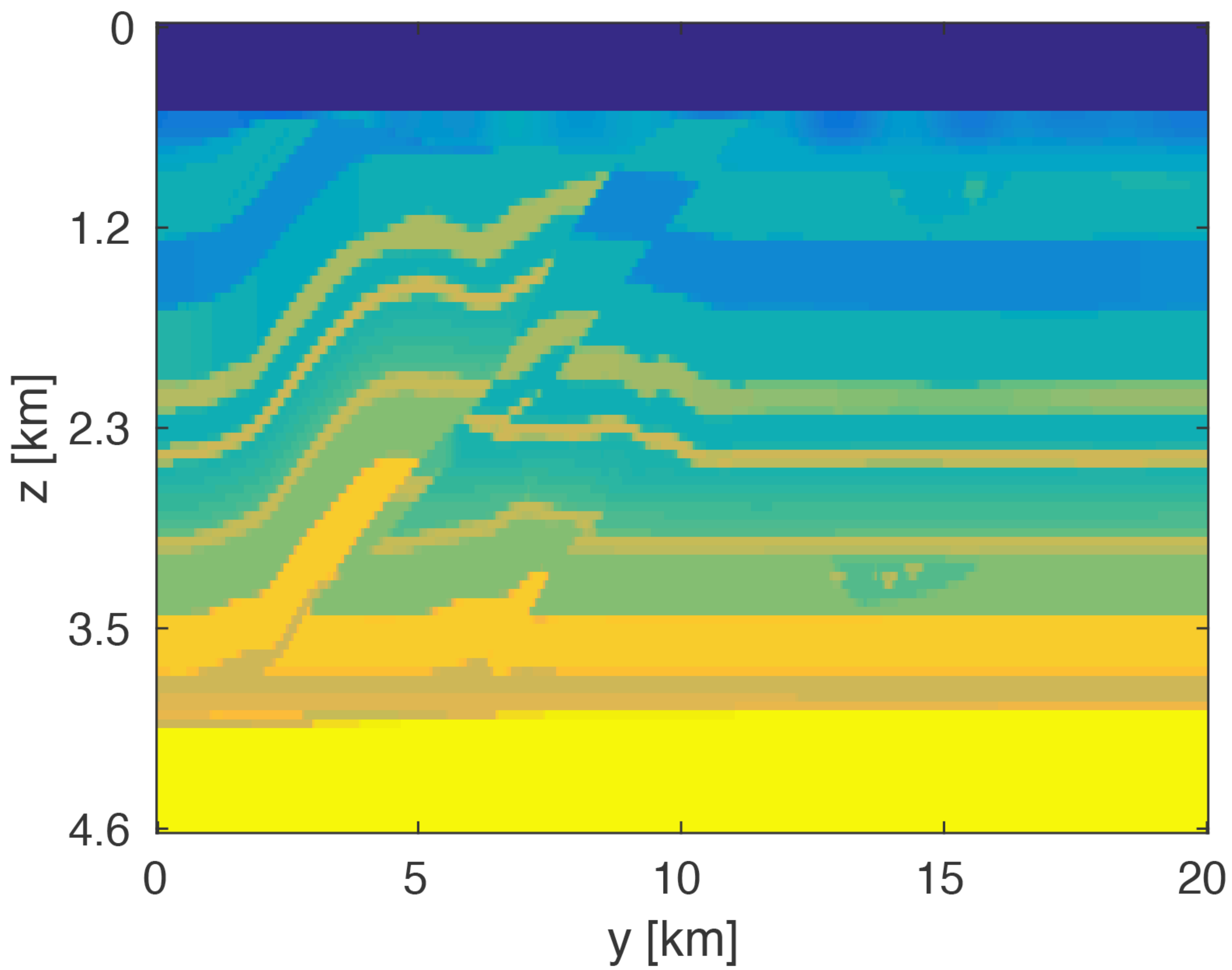


True model

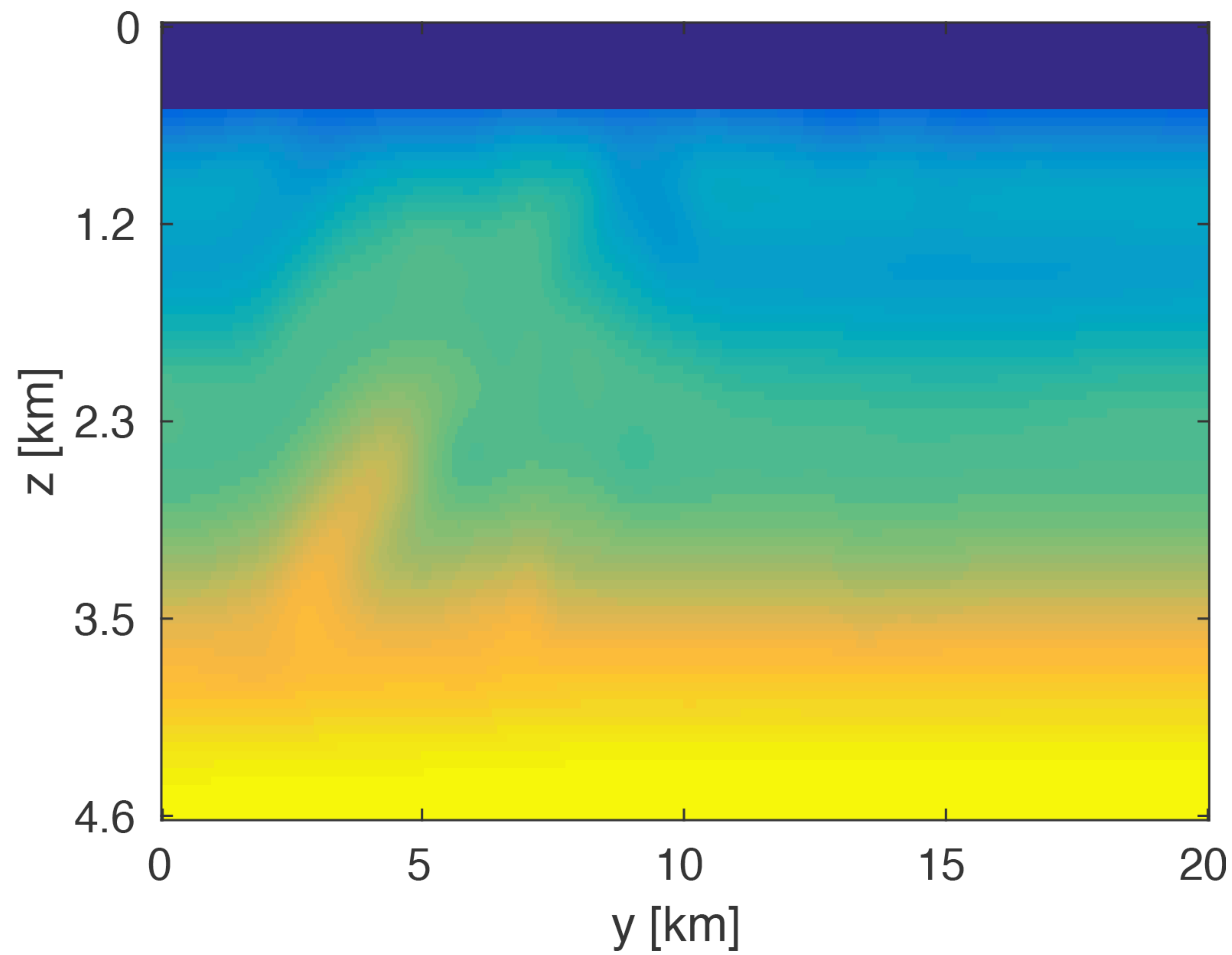


Compressed data

x = 12.5km lateral slice

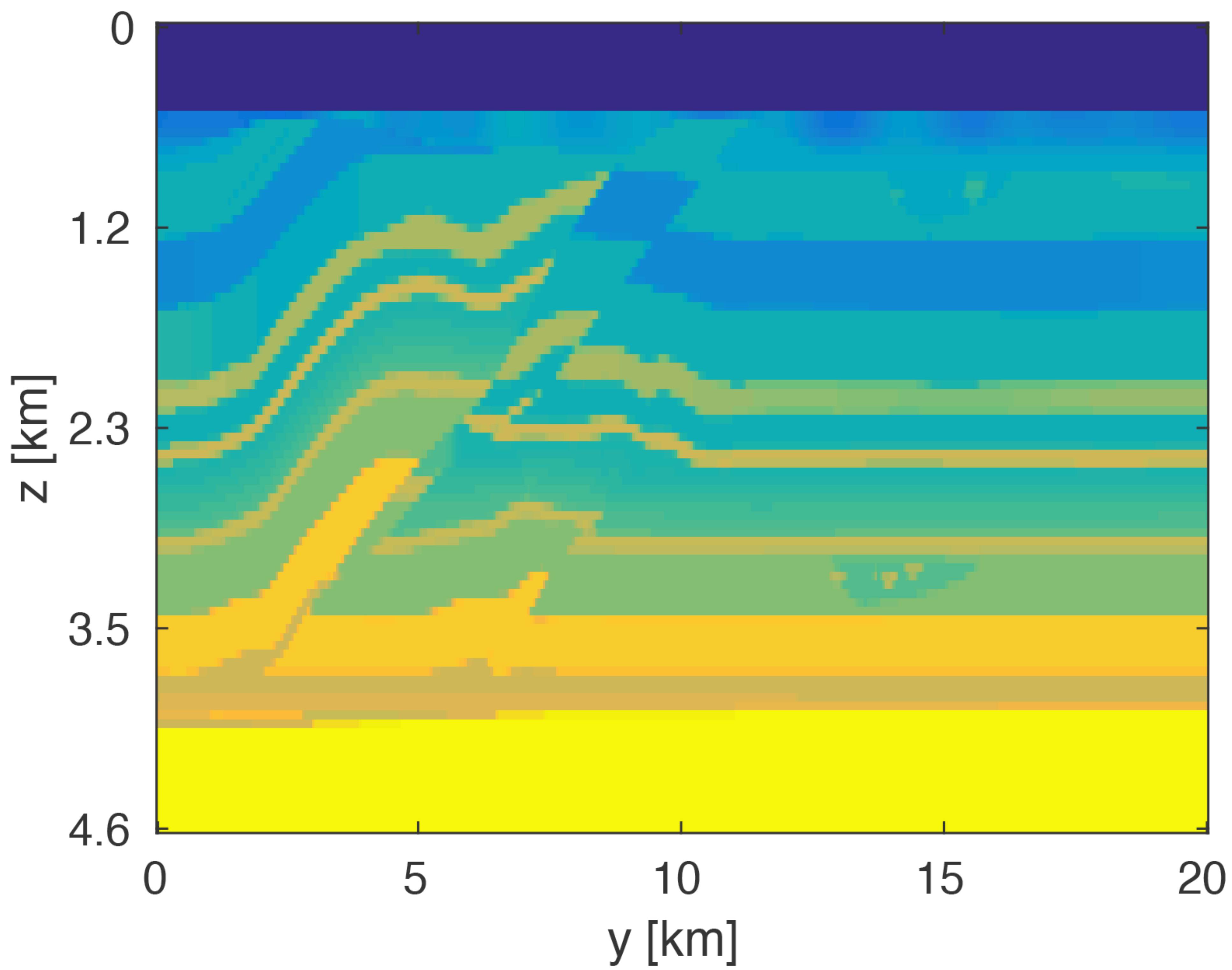


True model

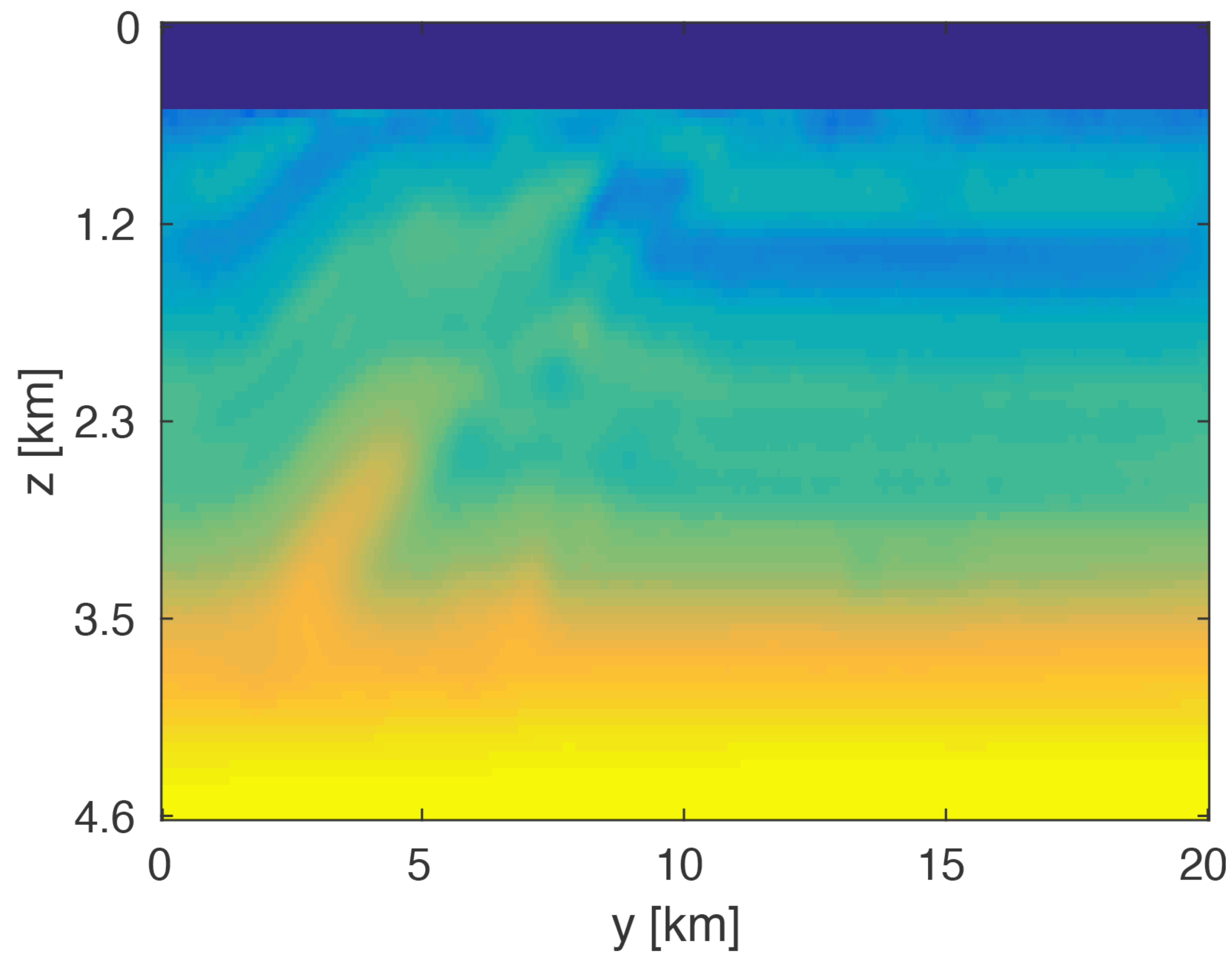


Initial model

x = 12.5km lateral slice

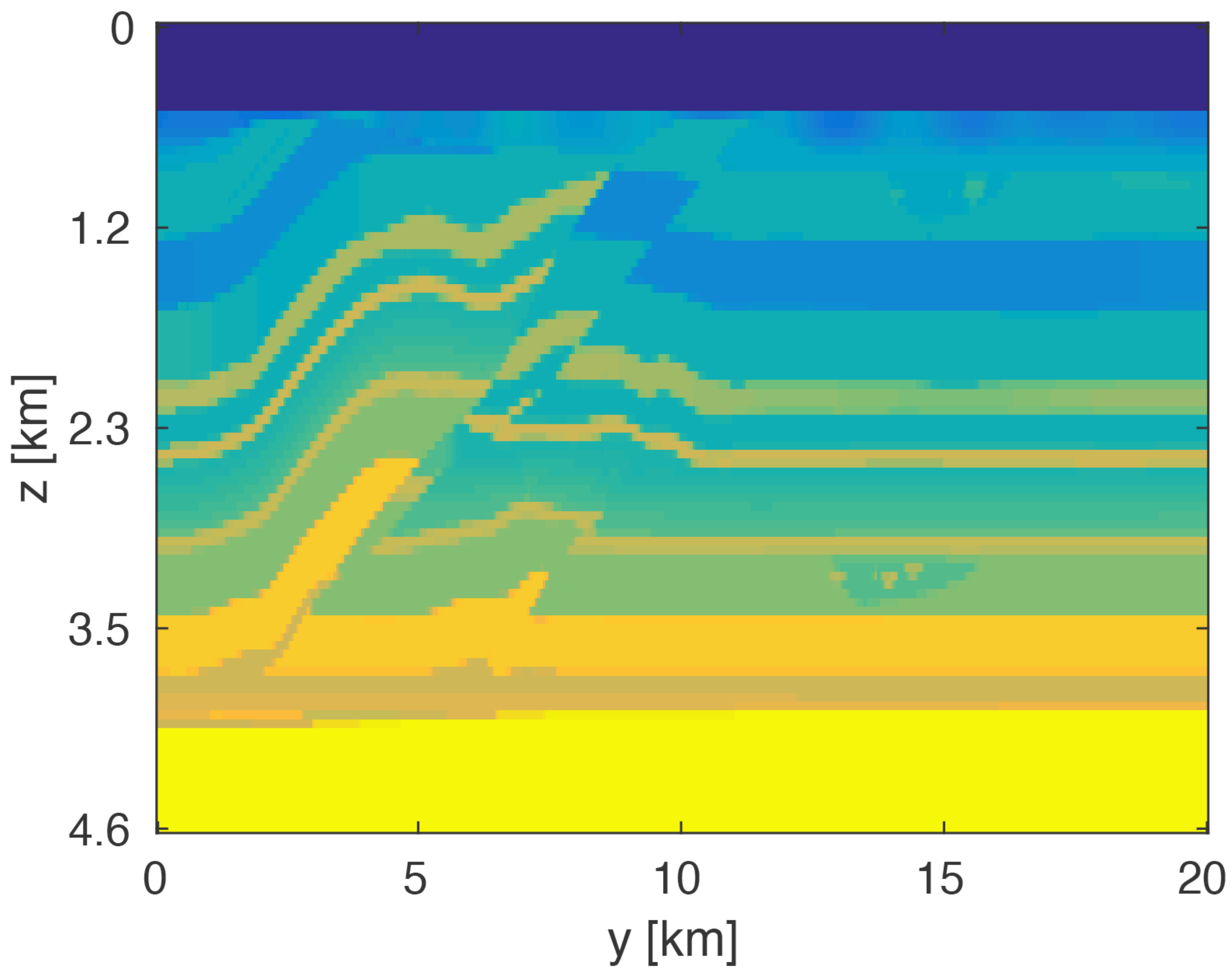


True model

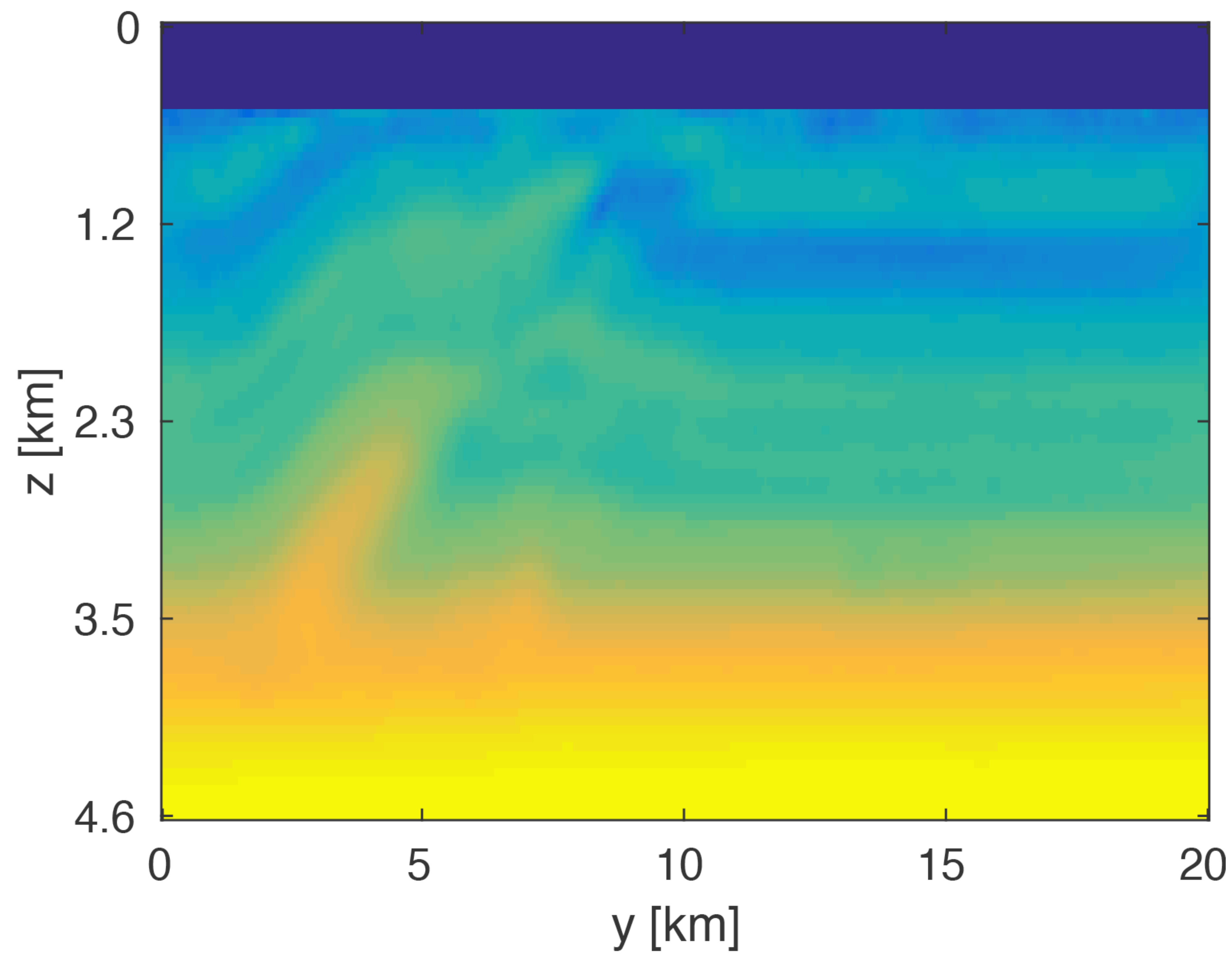


Full data

x = 12.5km lateral slice



True model



Compressed data

FWI examples on BG model

Model 2:

- 3D **BG** model
- 10km x 10km x 1.8 km
- 50m x 50m x 12 m spacing

Data:

- 49 x 49 sources, 200m interval
- 196 x 196 receivers, 50m interval
- Ricker wavelet, 10Hz peak frequency
- 3Hz - 6Hz ranging, 0.25Hz interval
- **remove 75%** of random receivers

FWI examples on BG model

Stochastic FWI results inverted w/

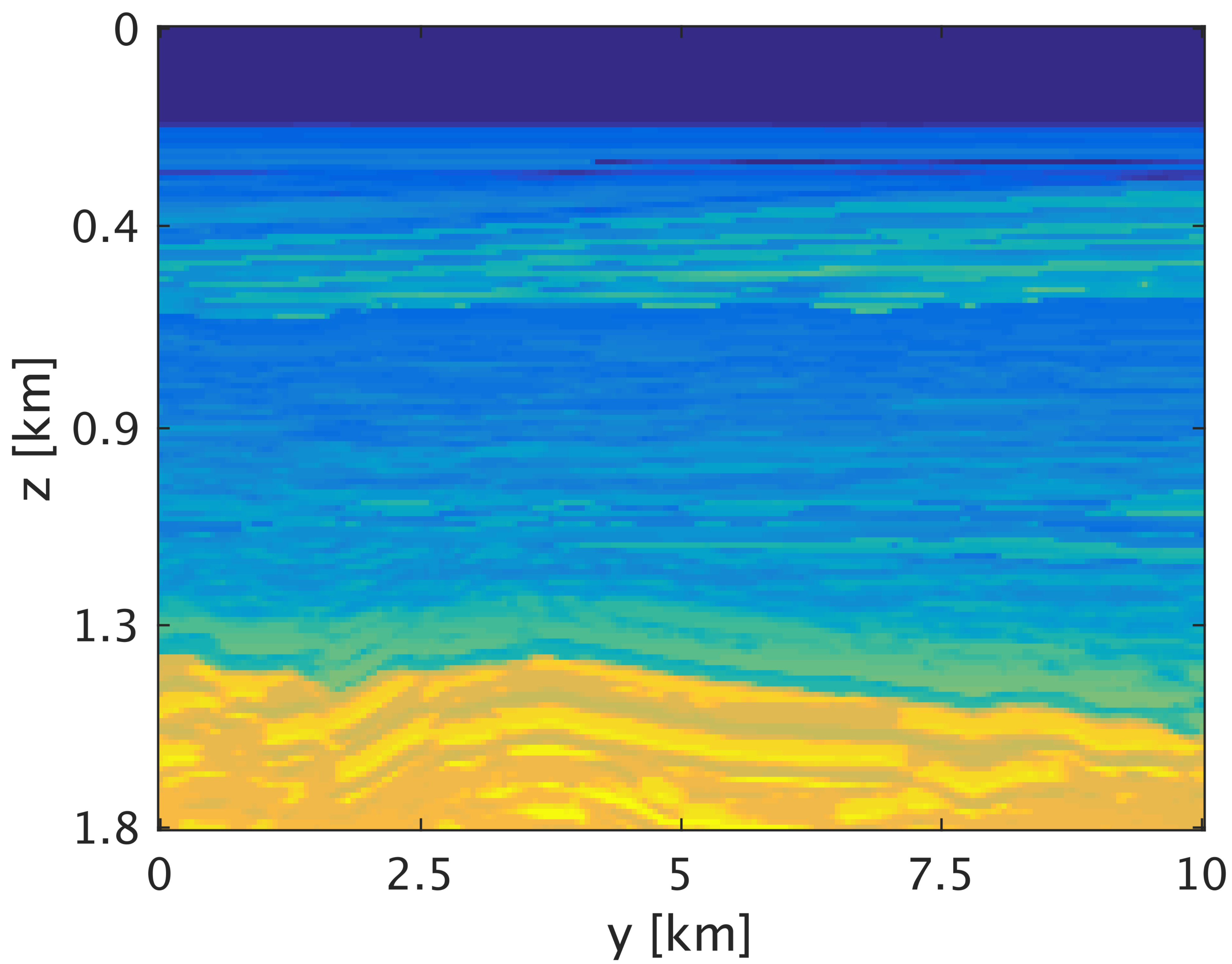
- **full** data
- compressed **HT** via **truncation**
- **subsampled** data
- compressed **HT** recovered from **interpolation**

Same source indices for four examples

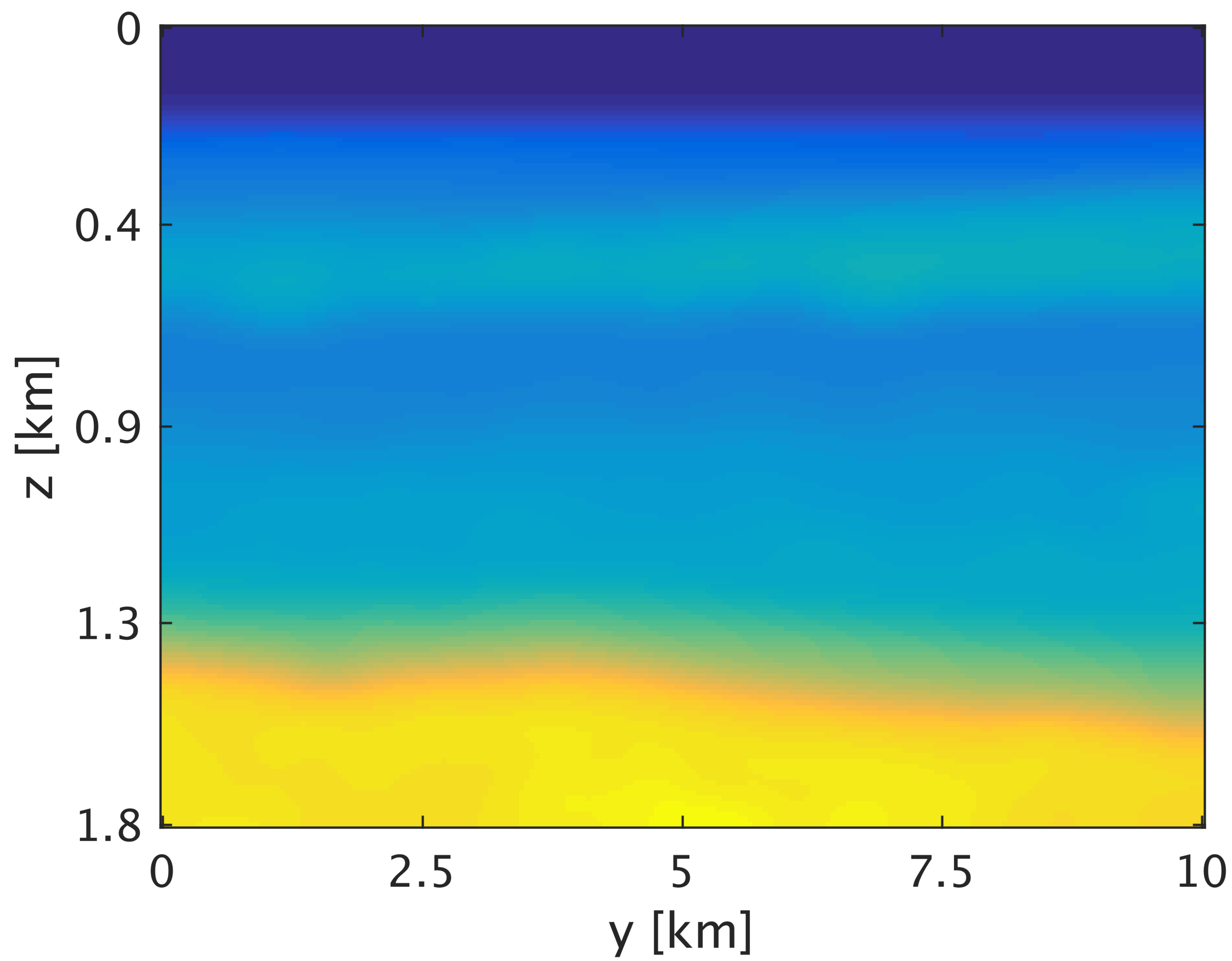
- **same** number of **PDE** solves
- **three** passes through the data

Full data & Compressed data

x = 4900m lateral slice

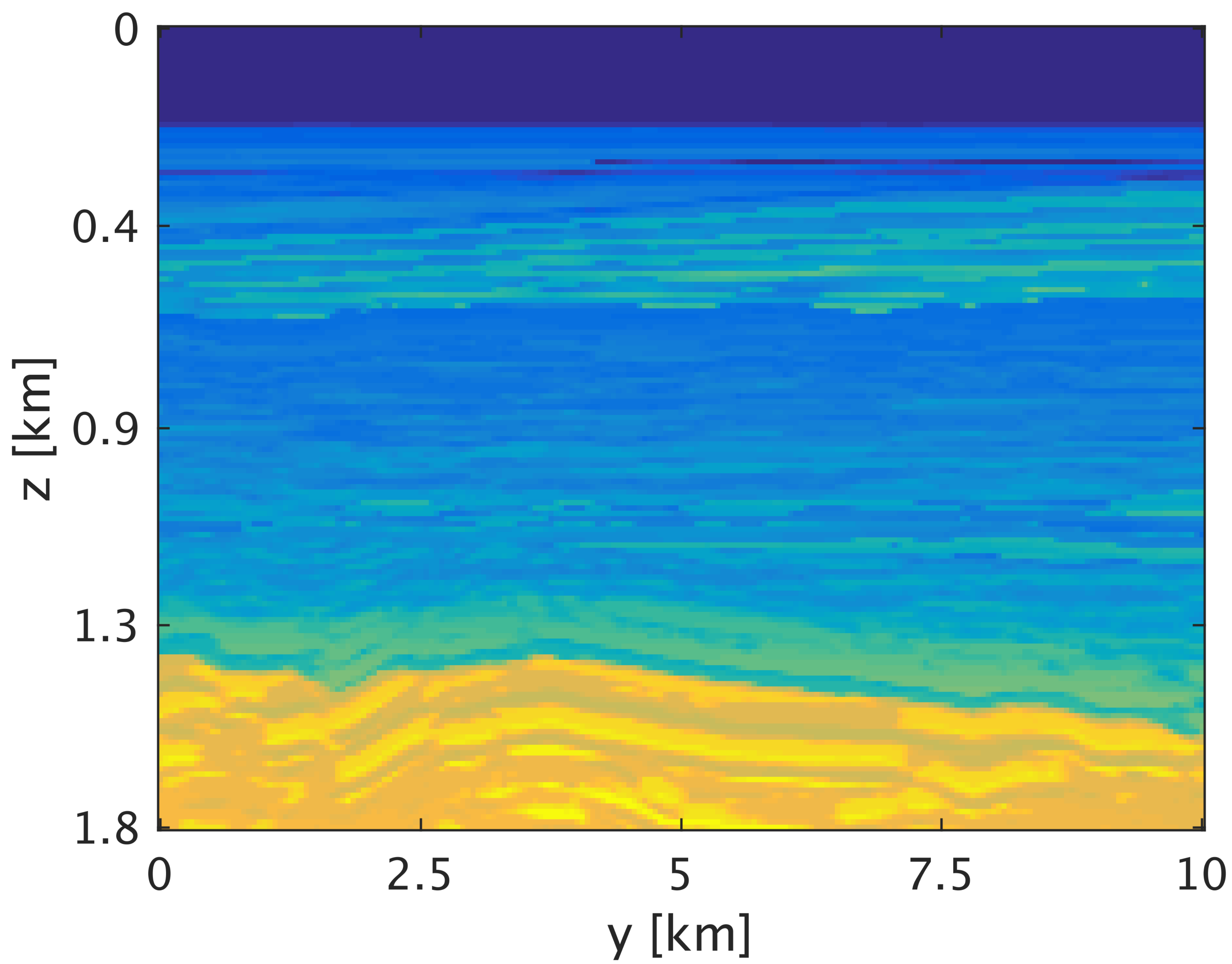


True model

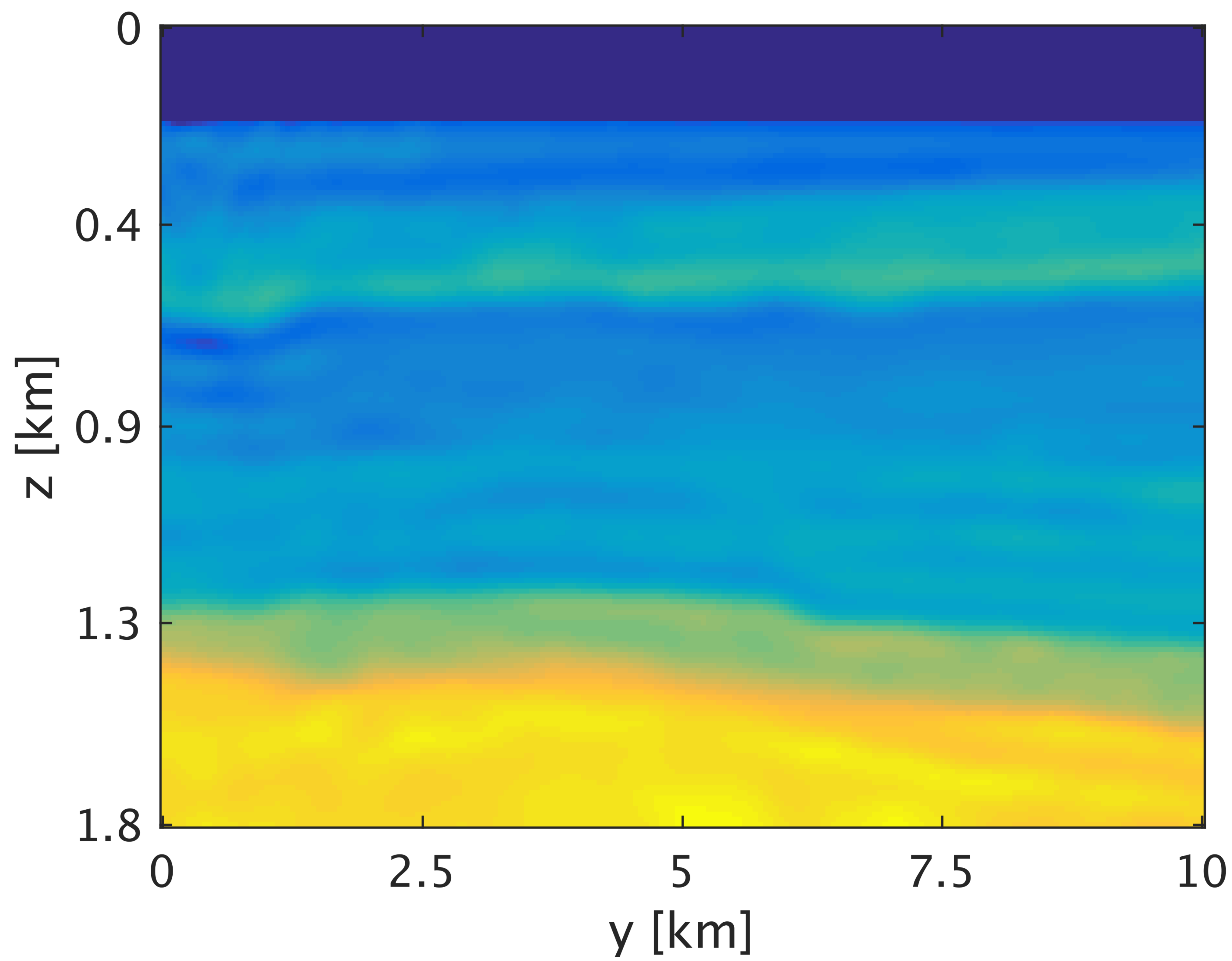


Initial model

x = 4900m lateral slice

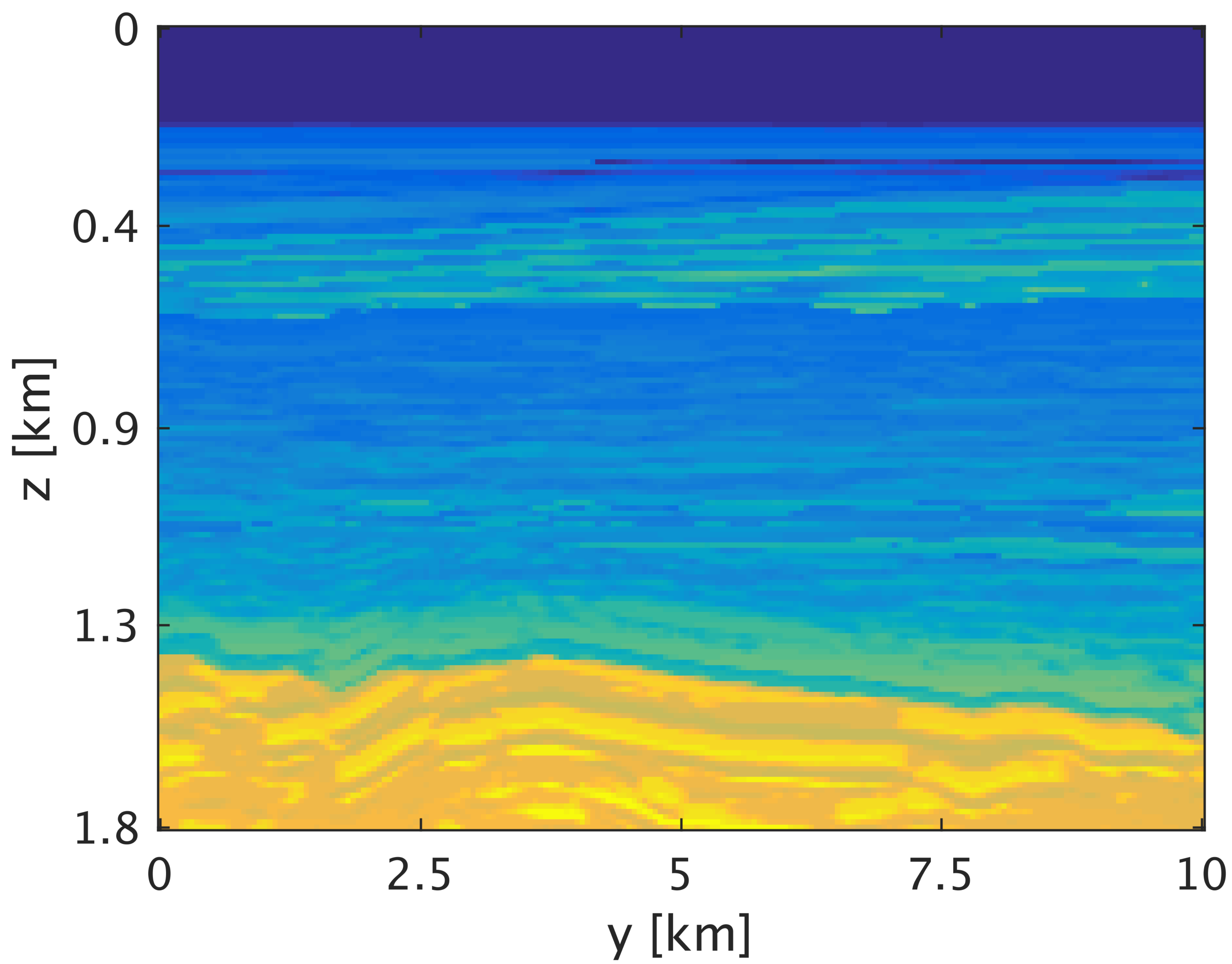


True model

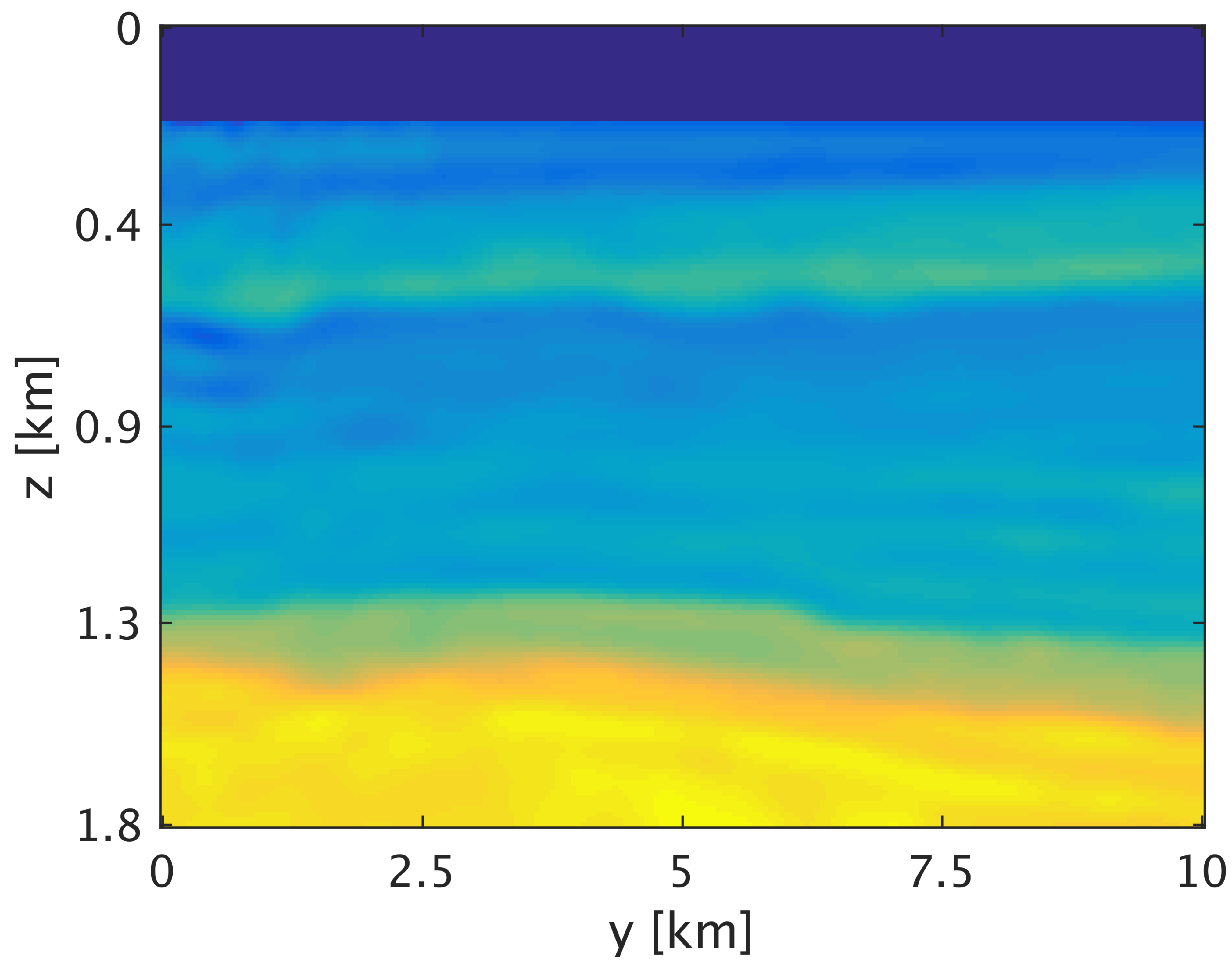


Full data

x = 4900m lateral slice

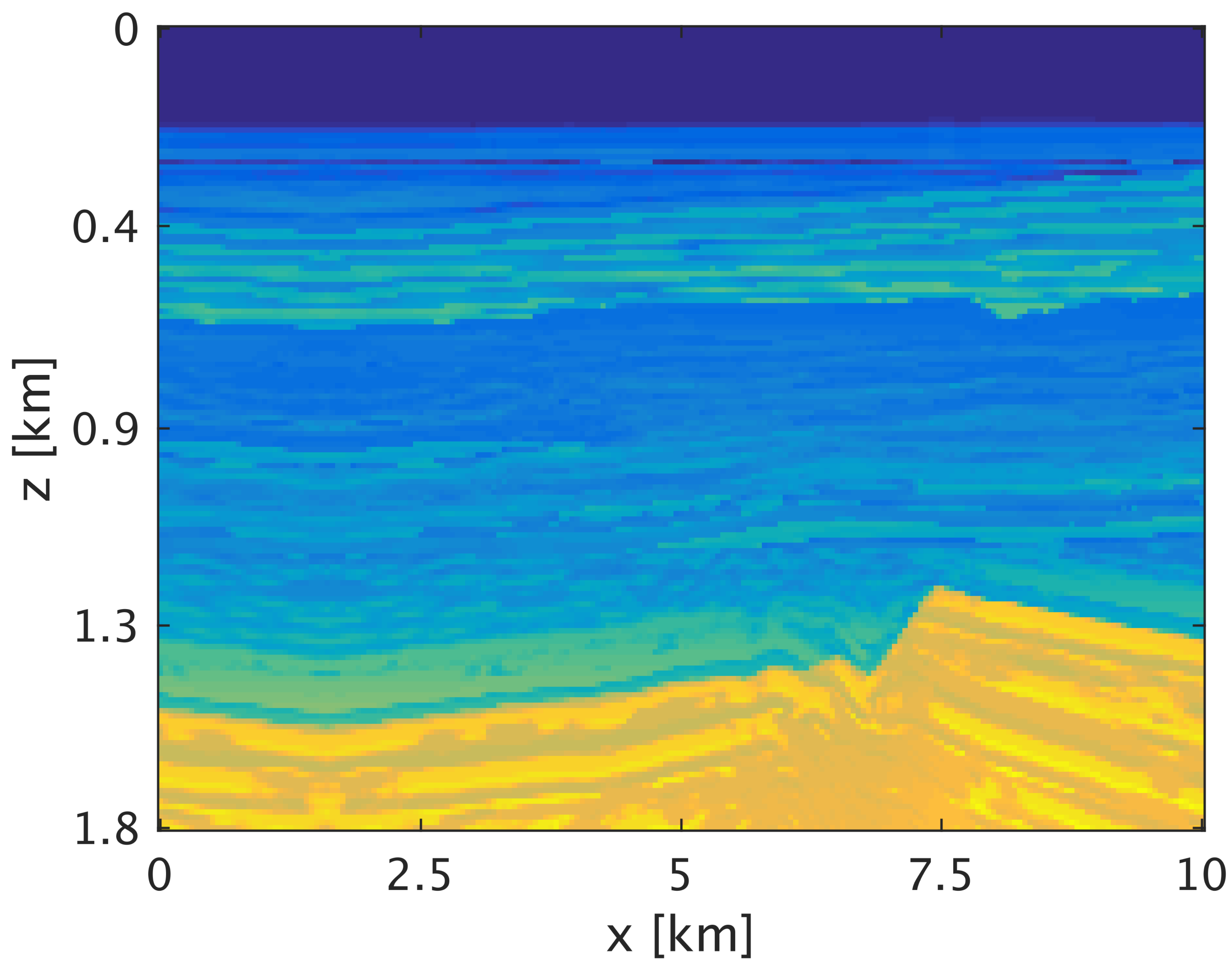


Full data

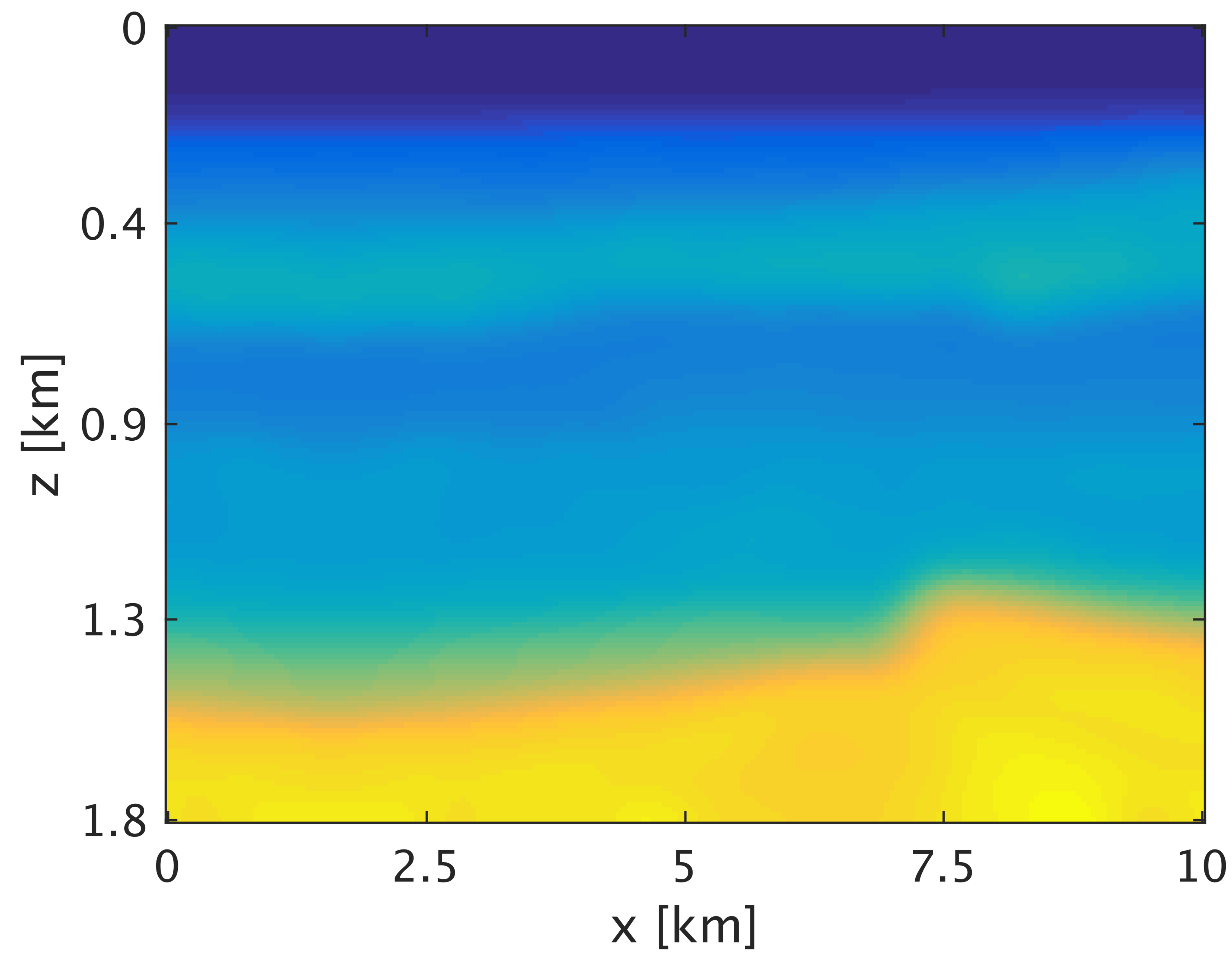


Compressed data

y = 5650m lateral slice

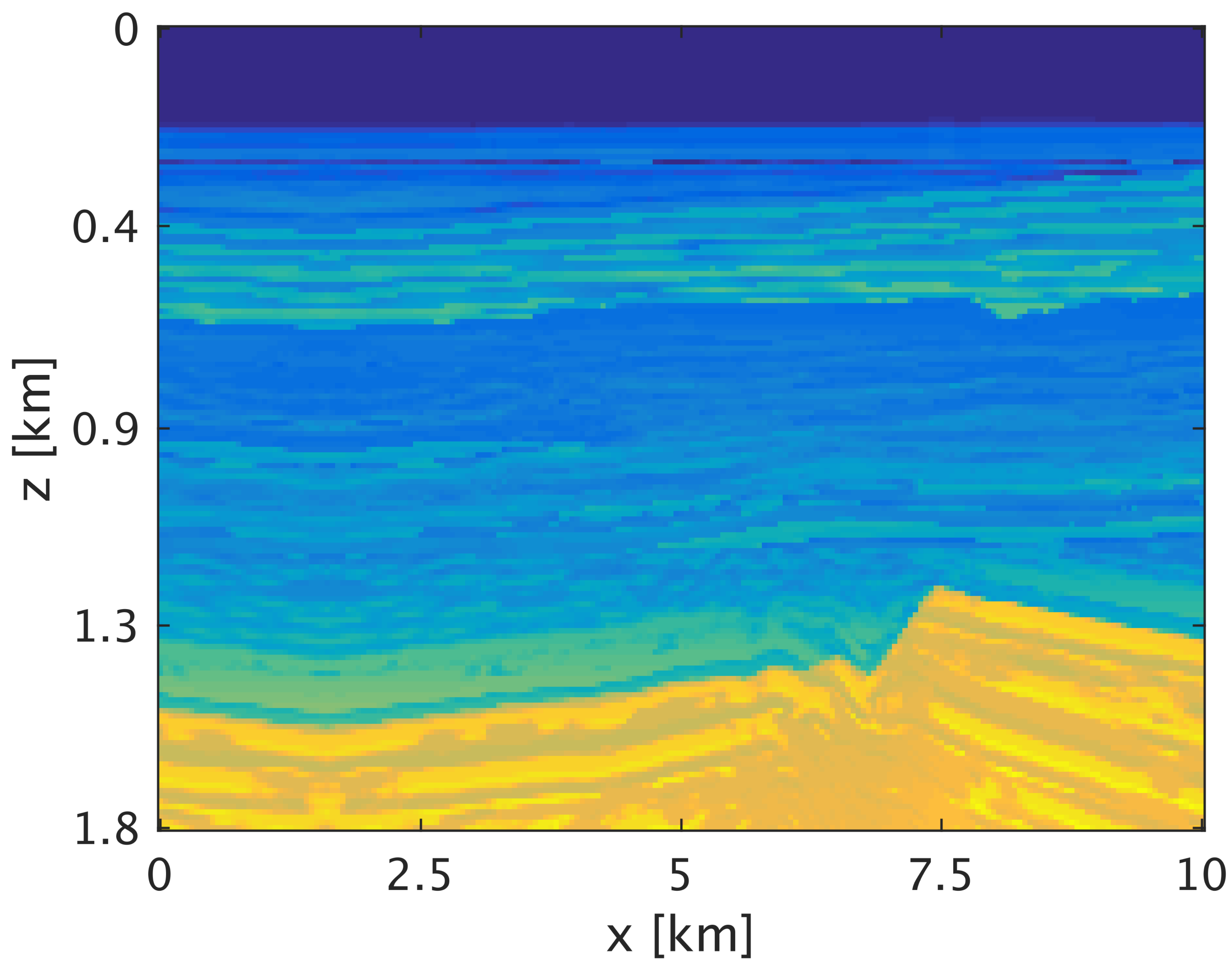


True model

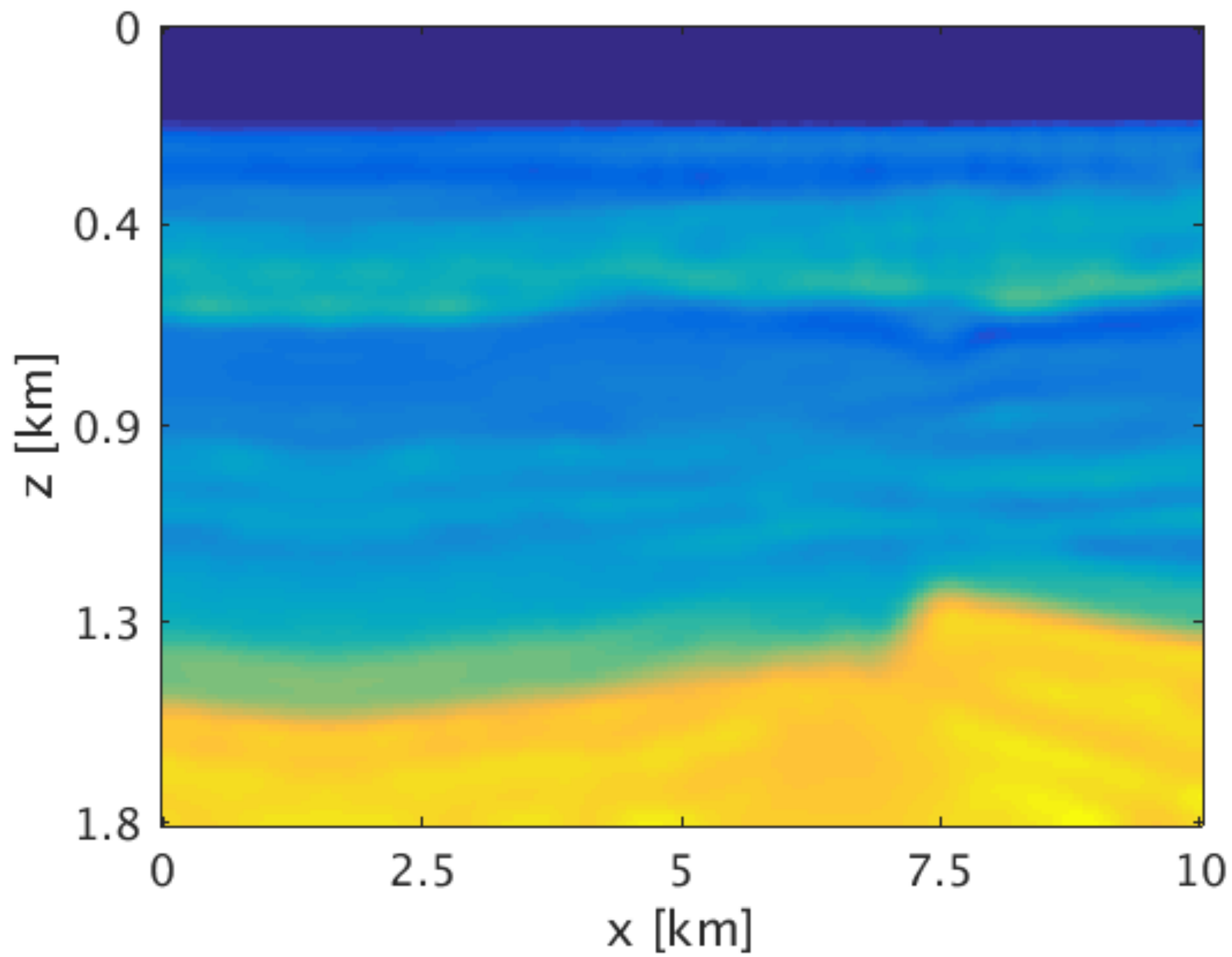


Initial model

y = 5650m lateral slice

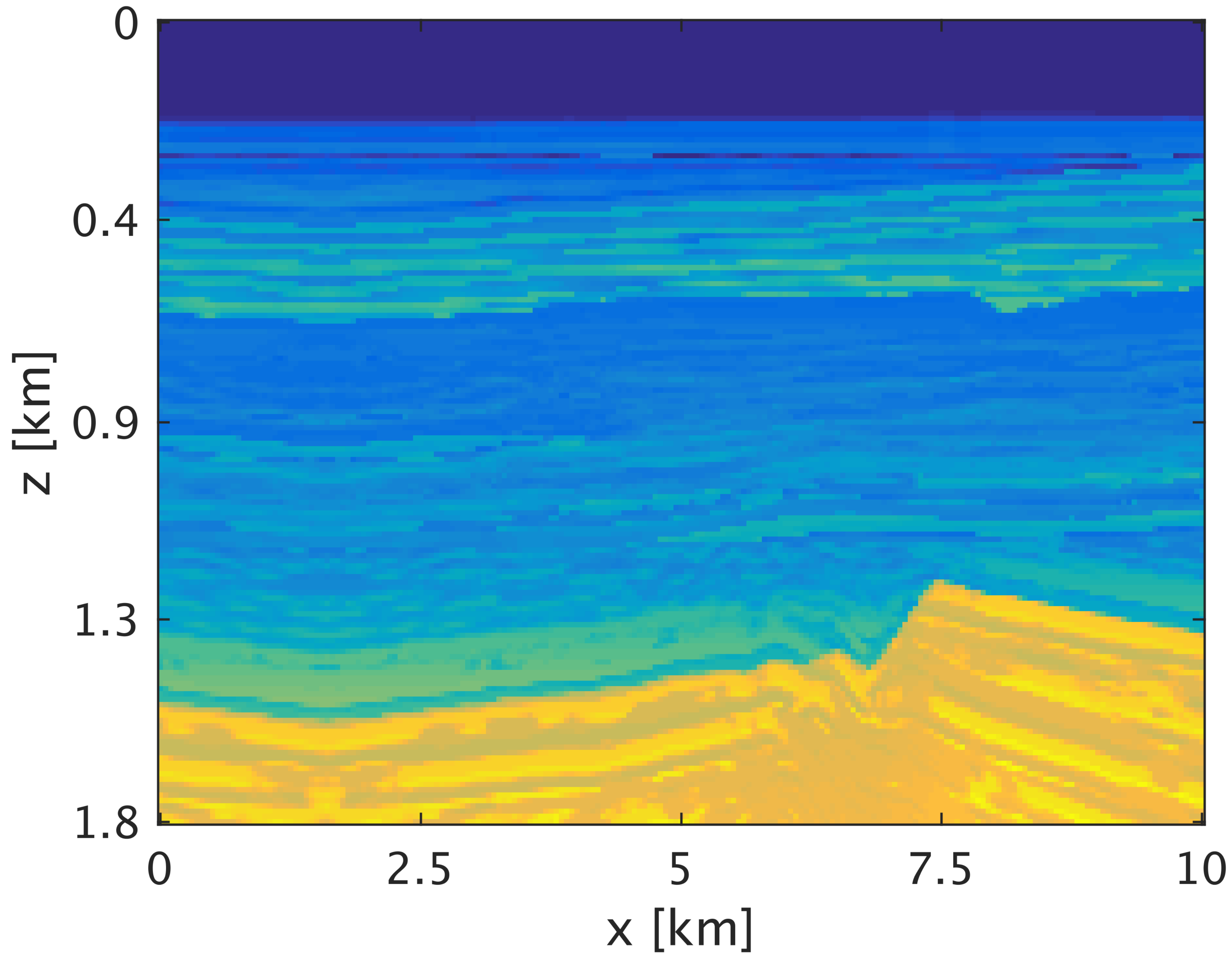


True model

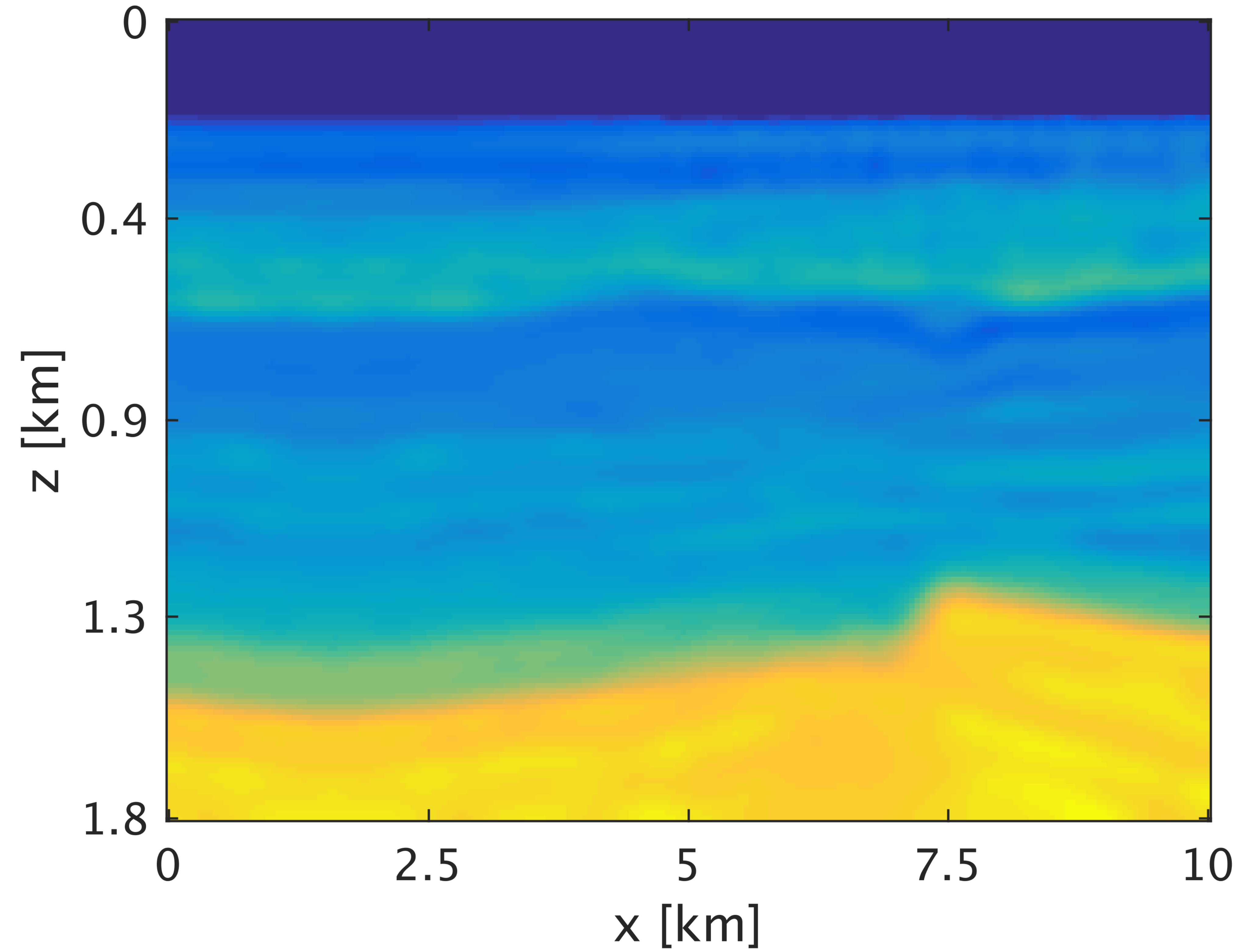


Full data

y = 5650m lateral slice

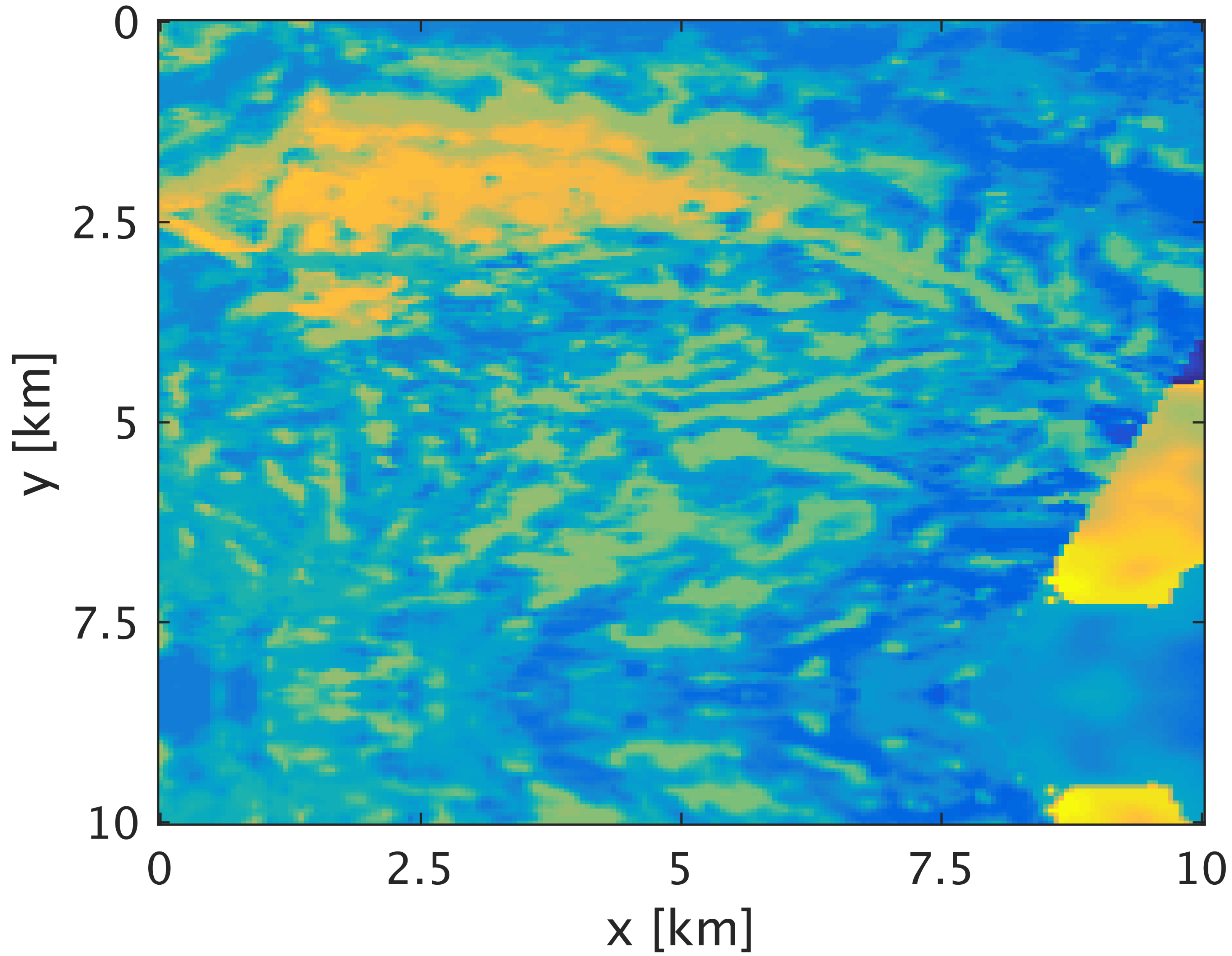


True model

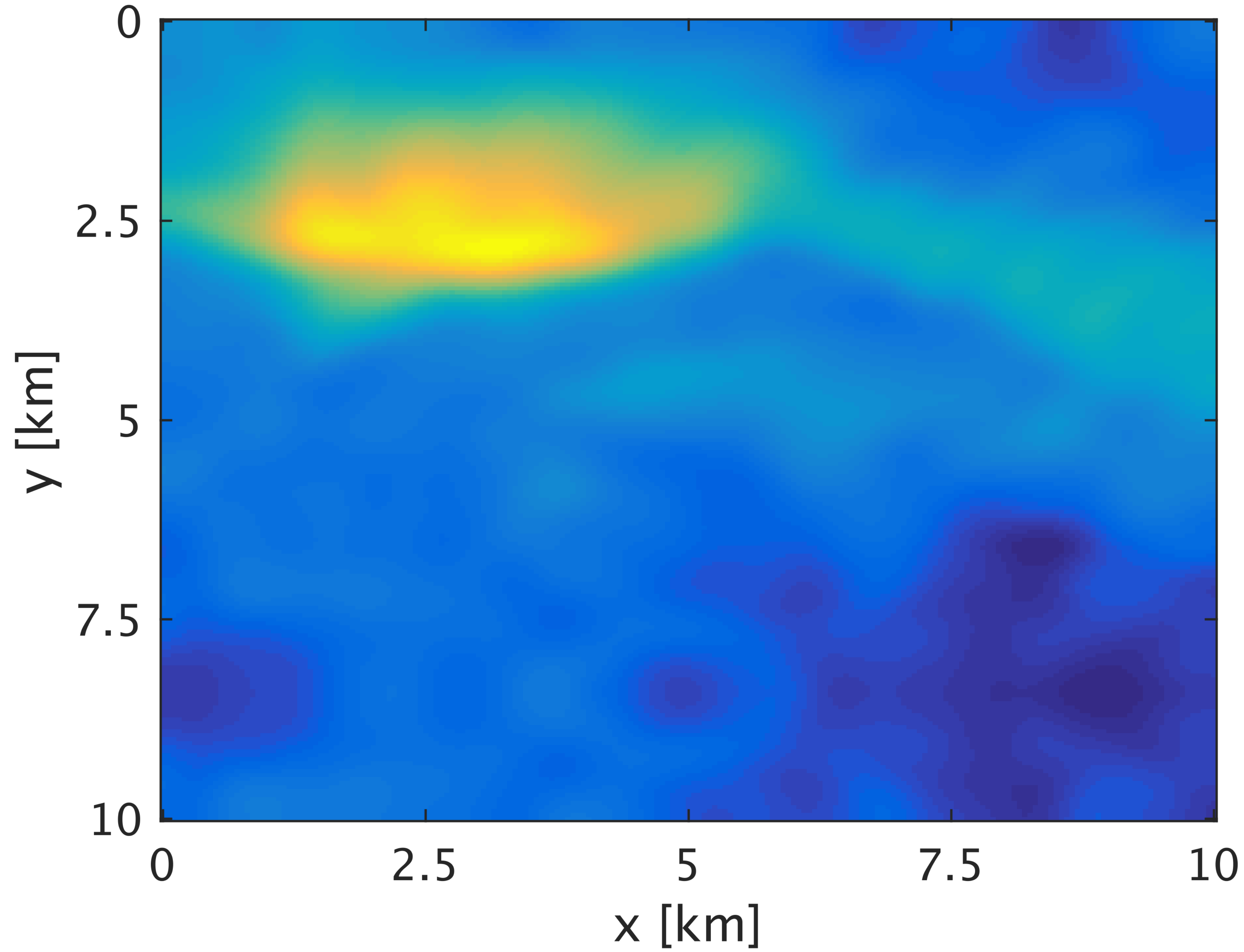


Compressed data

z = 1200m depth slice

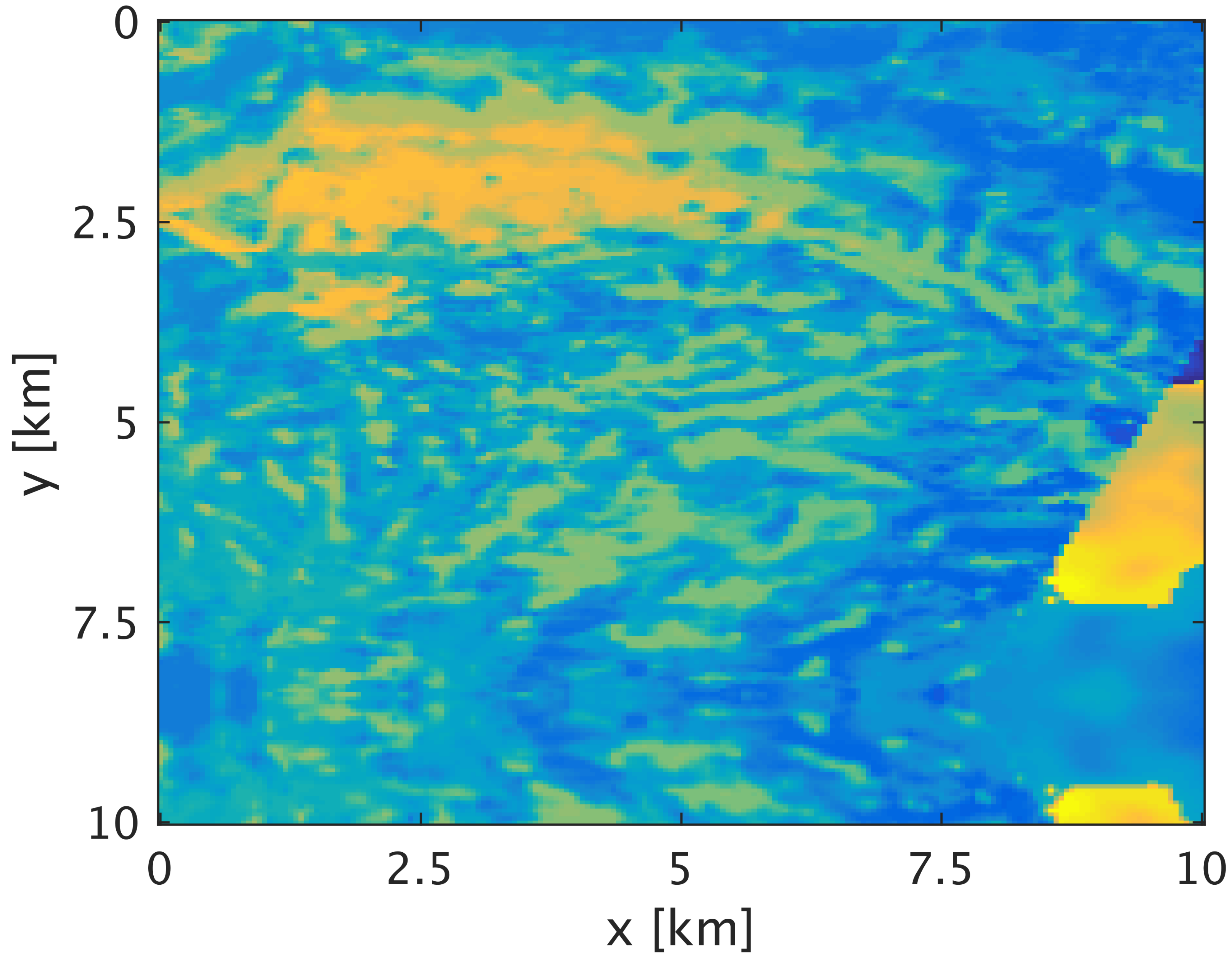


True model

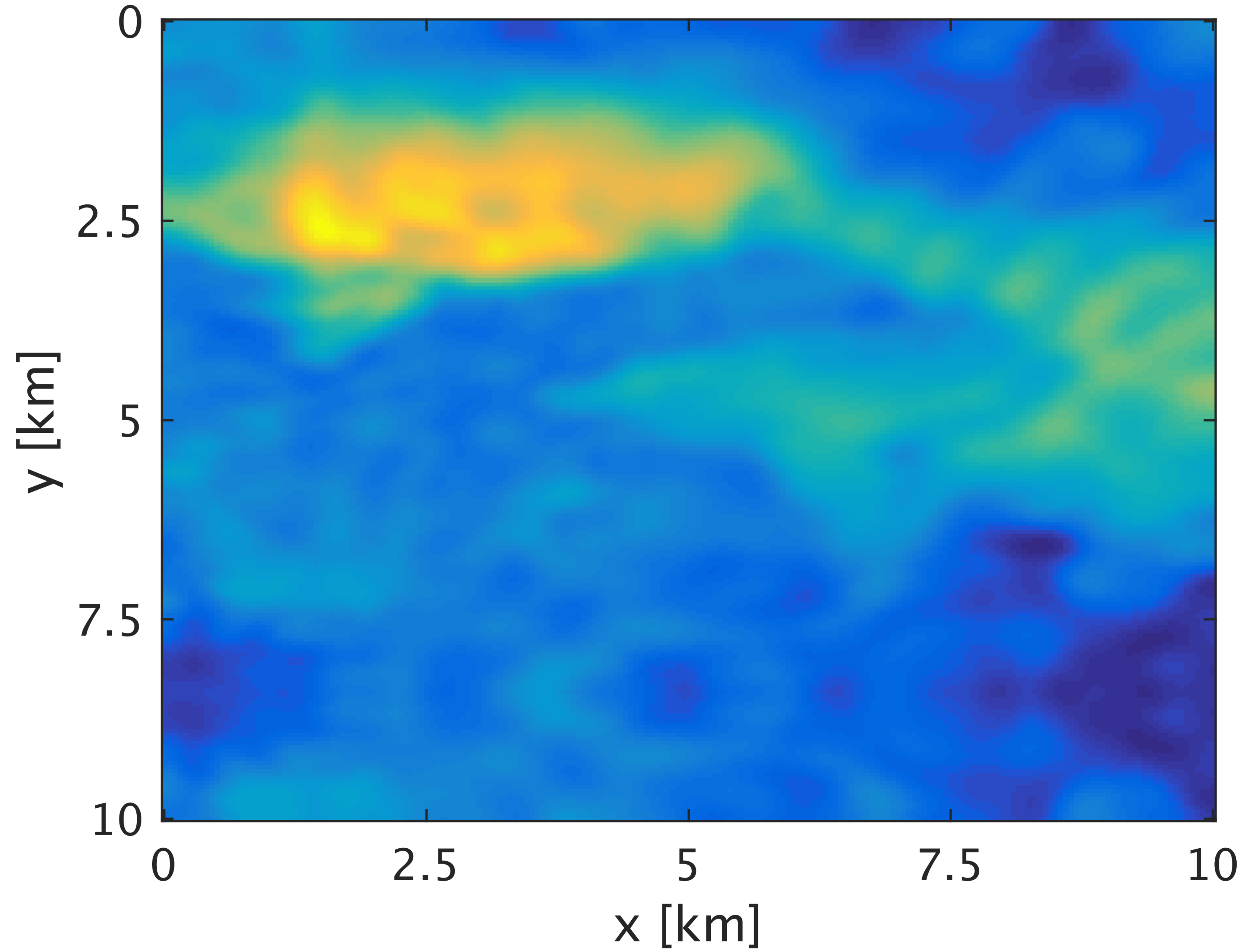


Initial model

z = 1200m depth slice

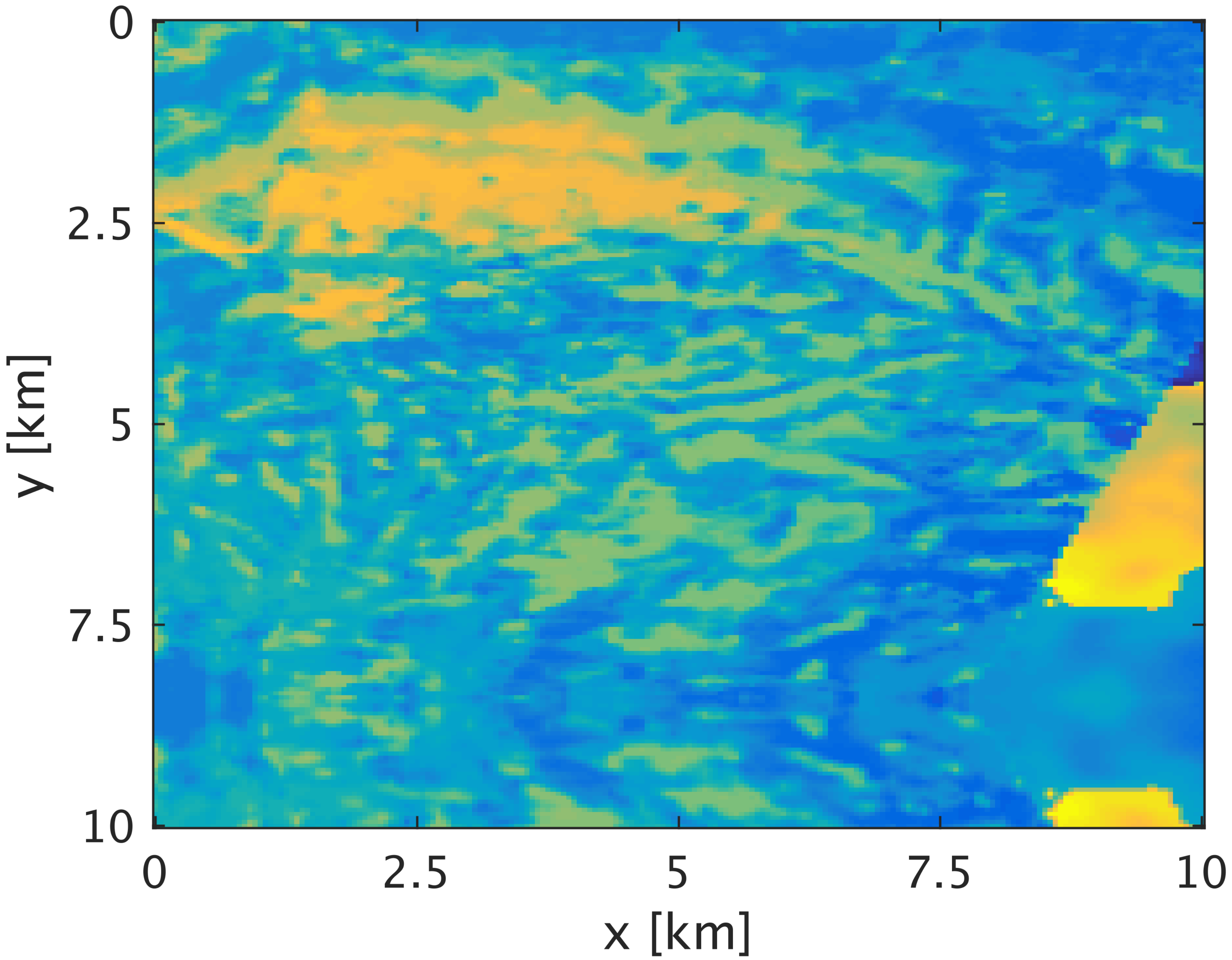


True model

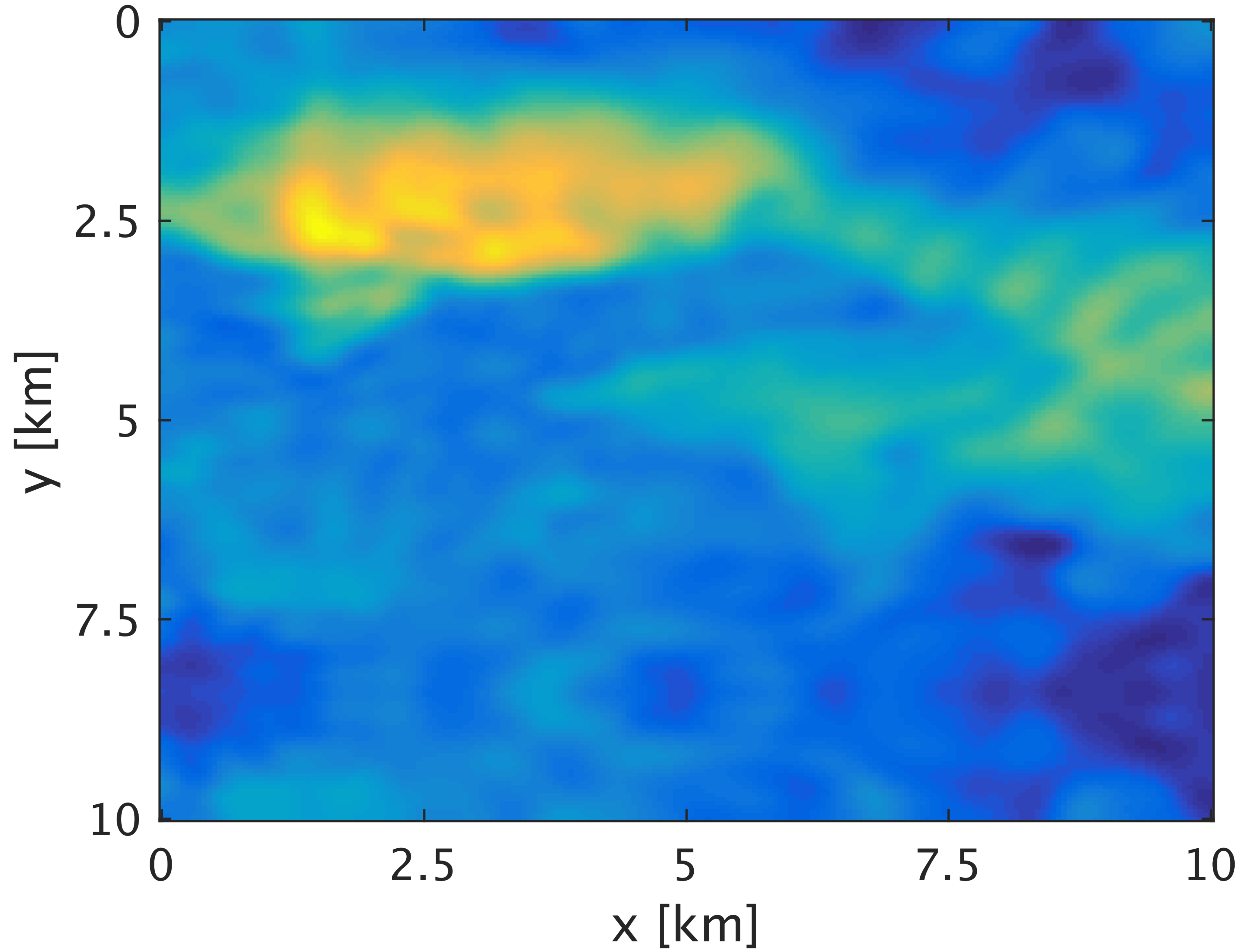


Full data

z = 1200m depth slice



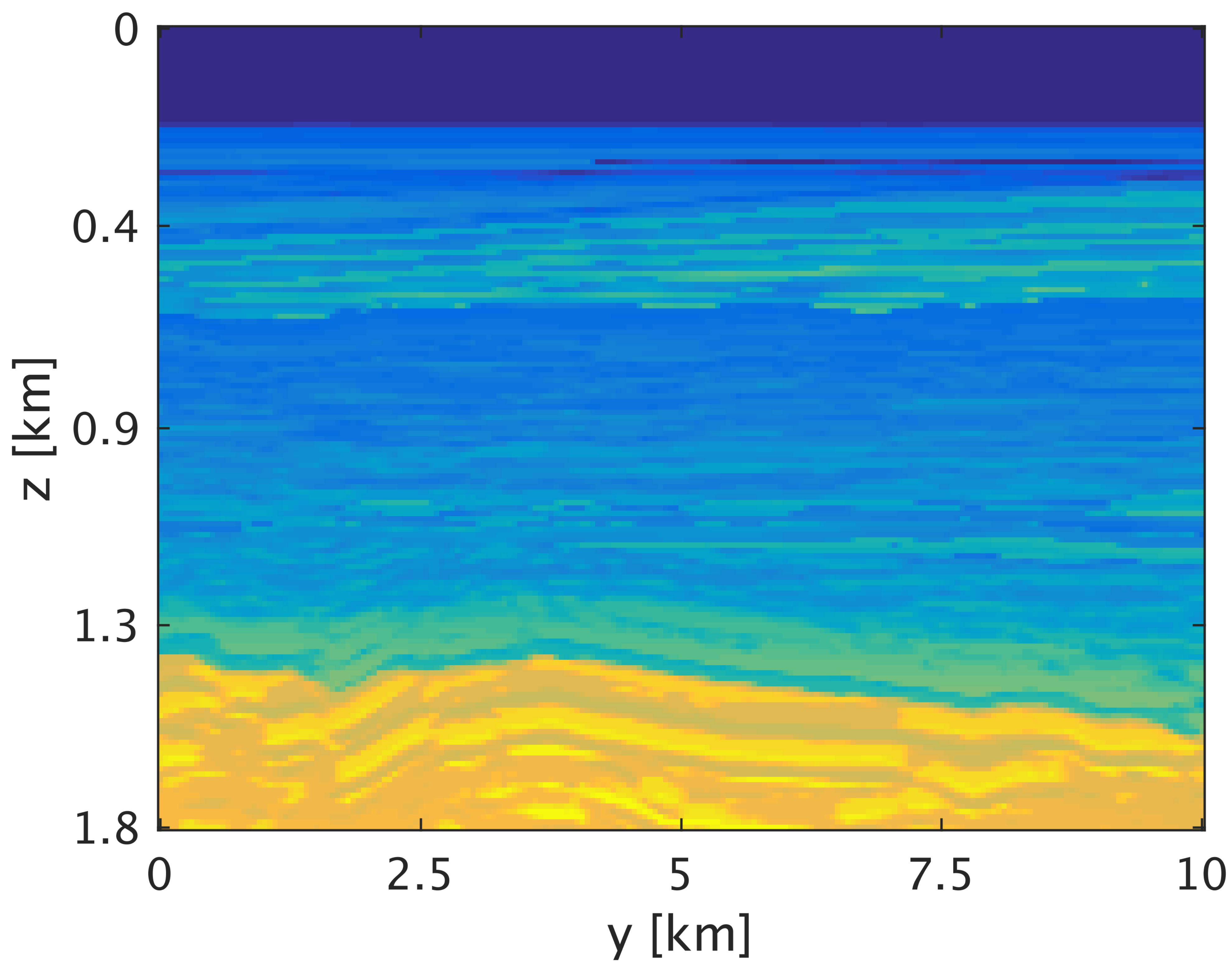
True model



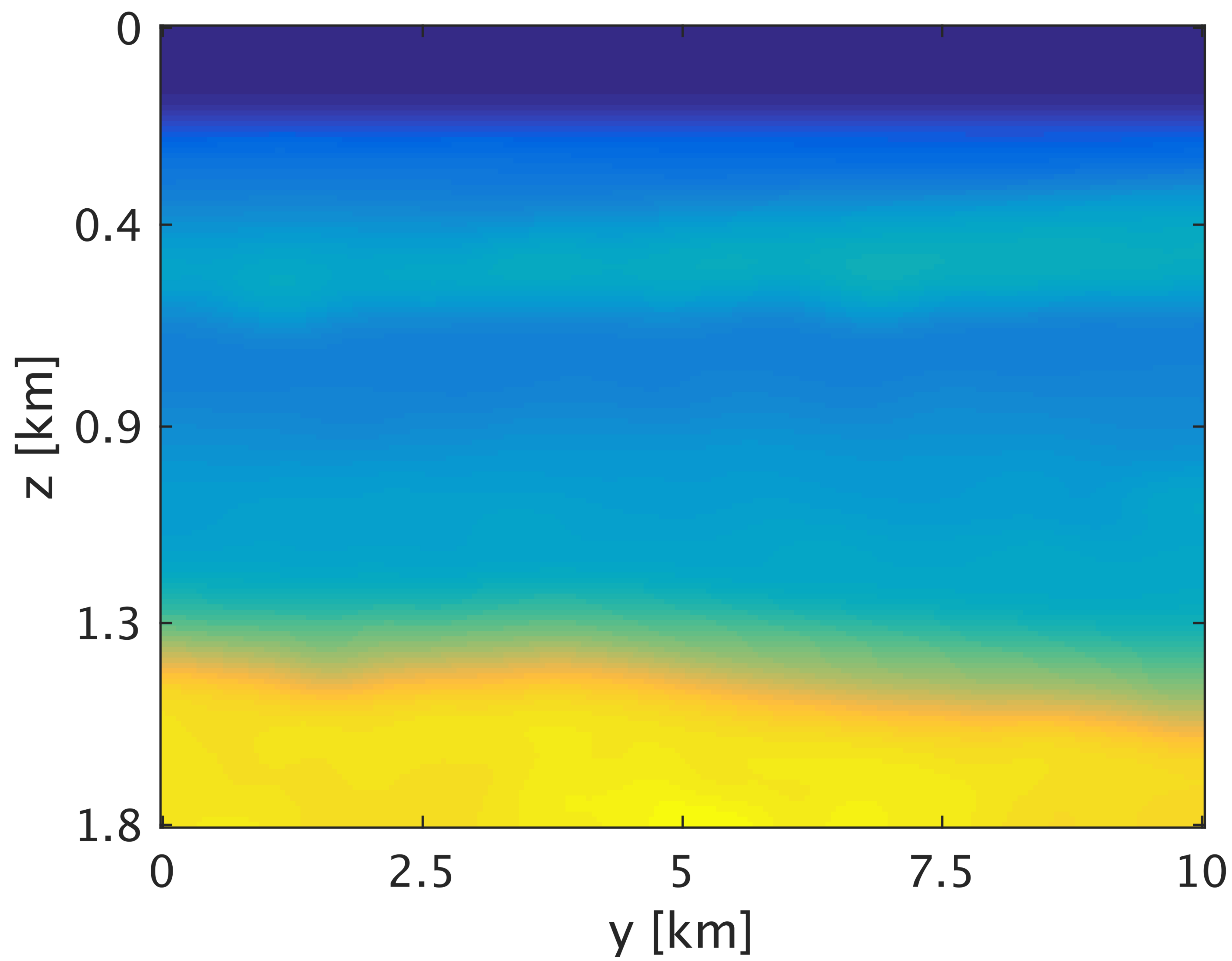
Compressed data

Subsampled data & Interpolated data

x = 4900m lateral slice

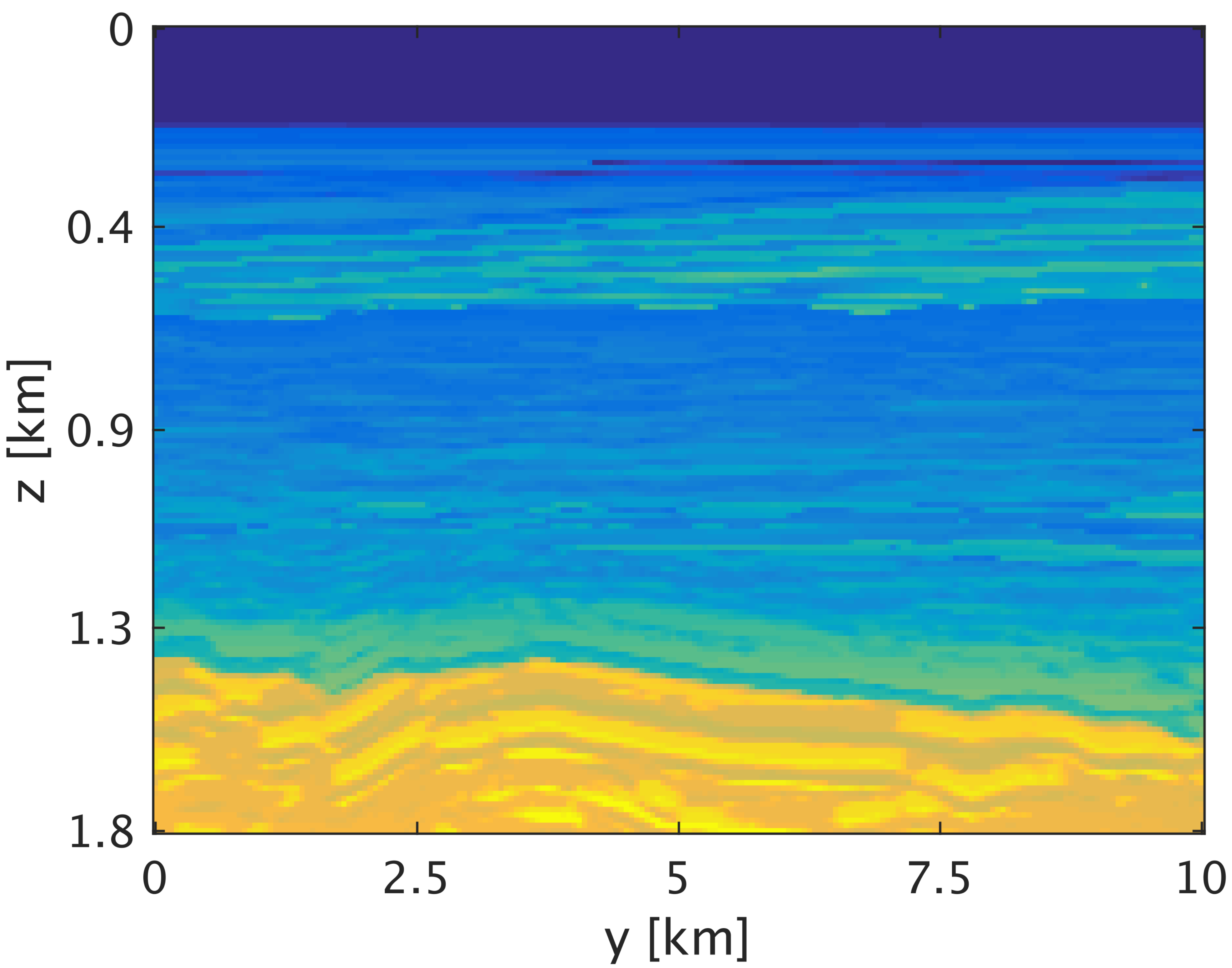


True model

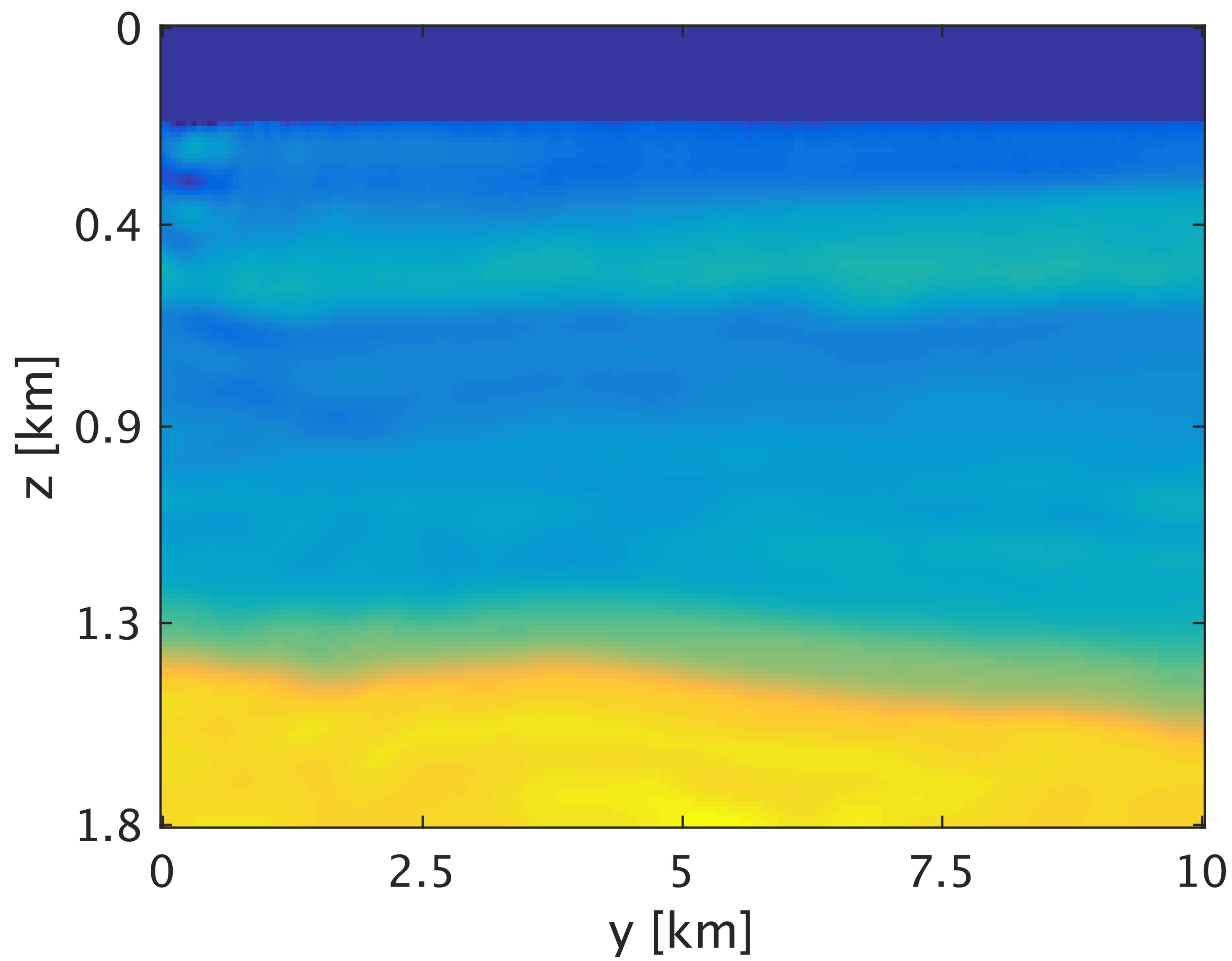


Initial model

x = 4900m lateral slice

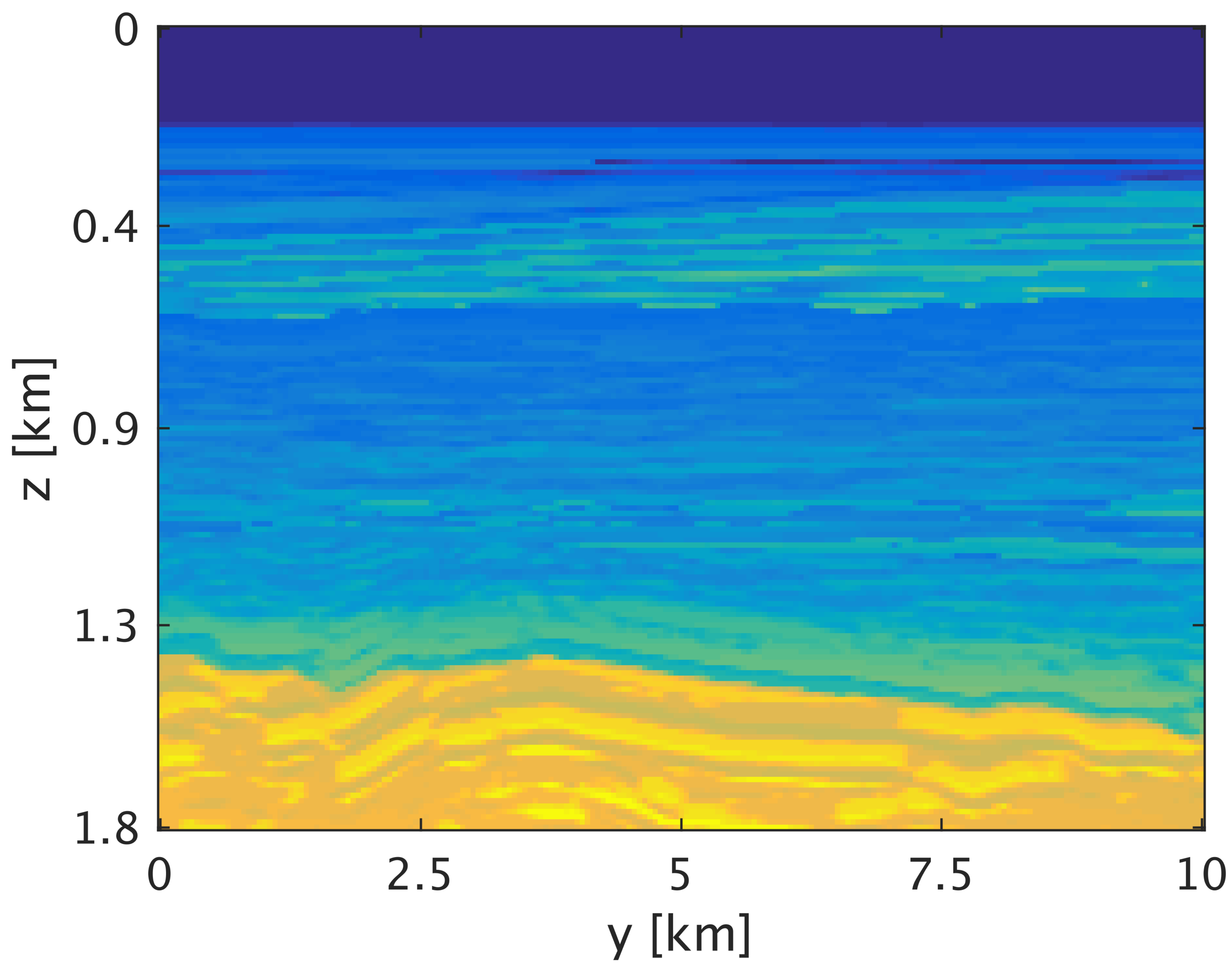


True model

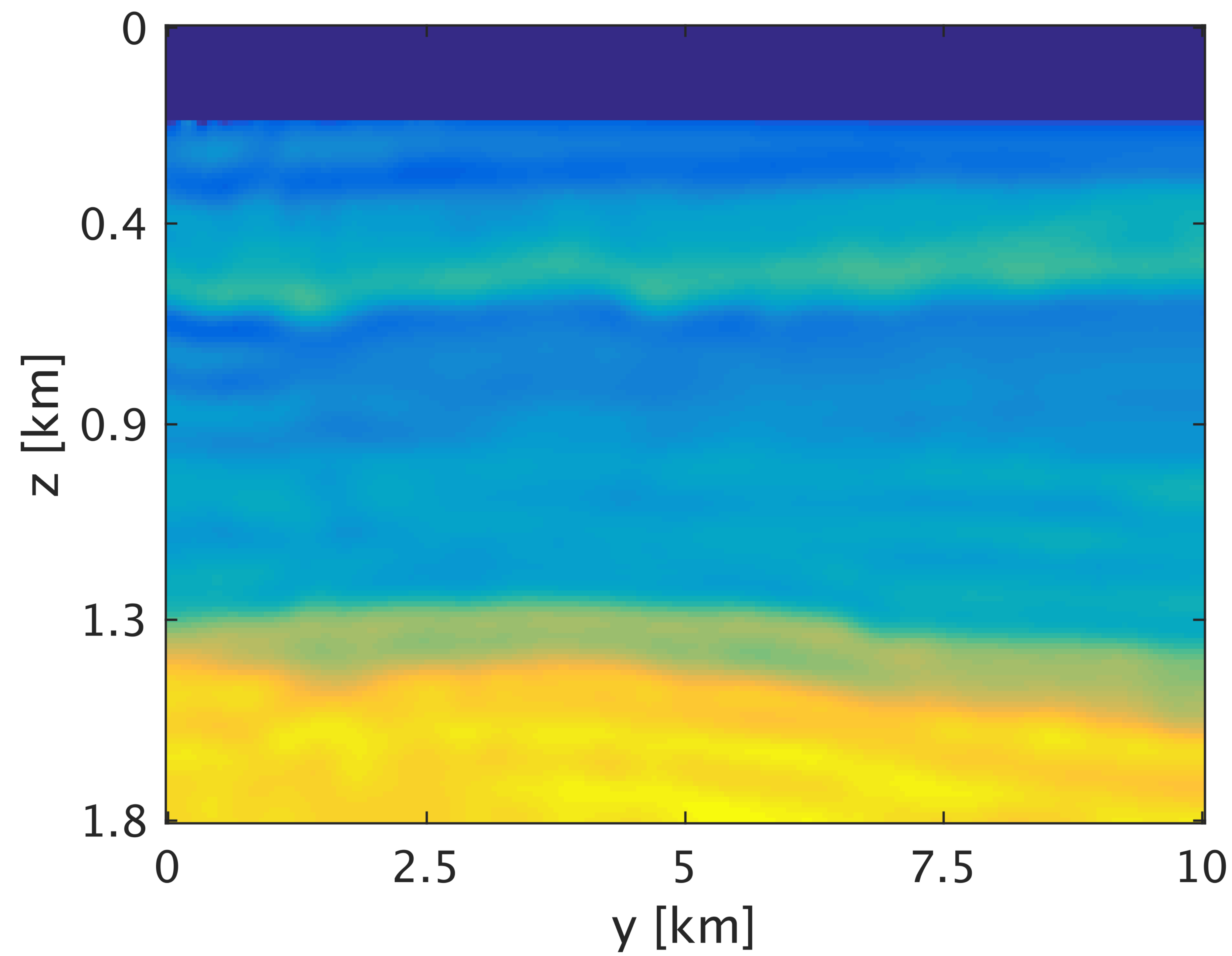


Subsampled data

x = 4900m lateral slice

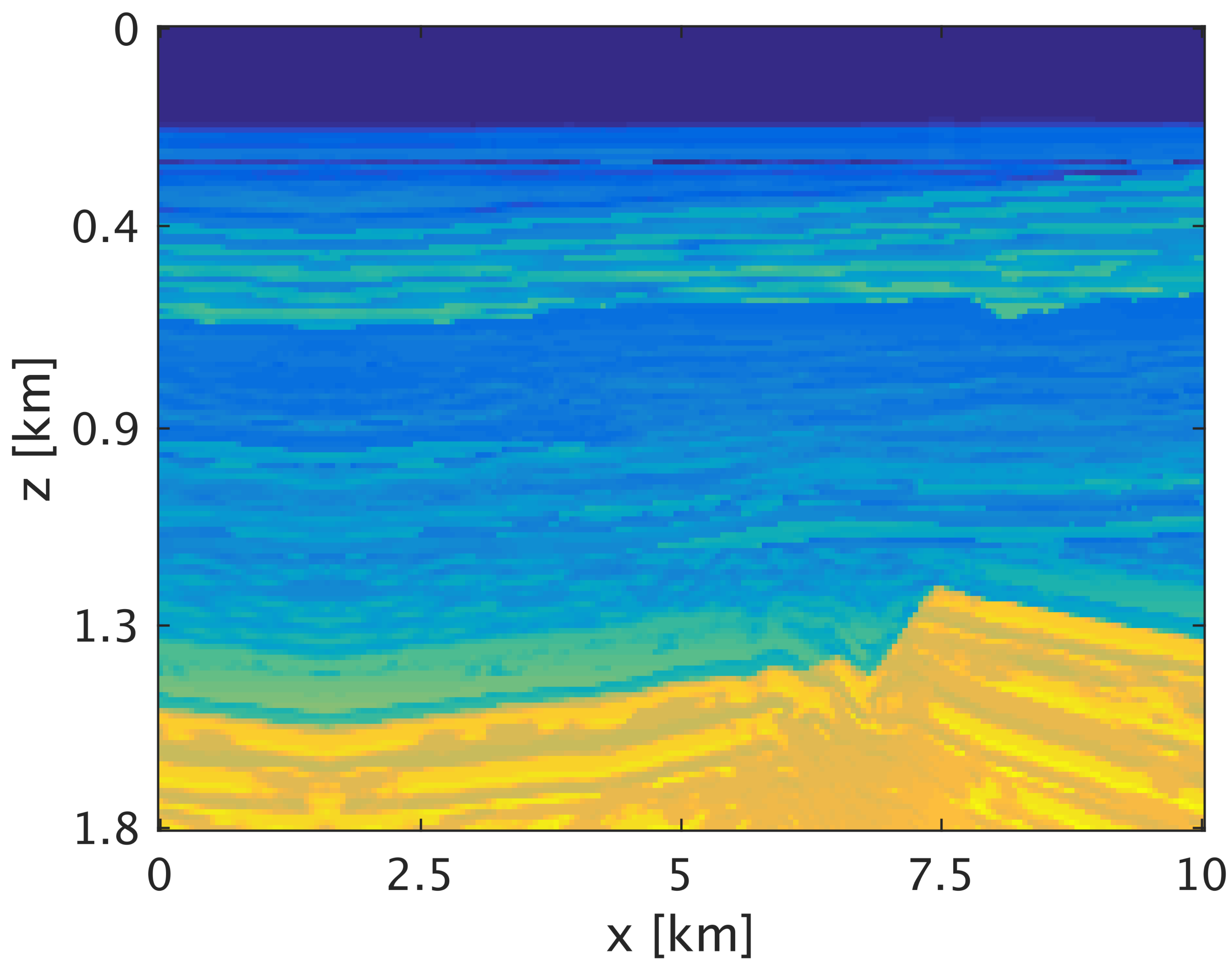


Full data

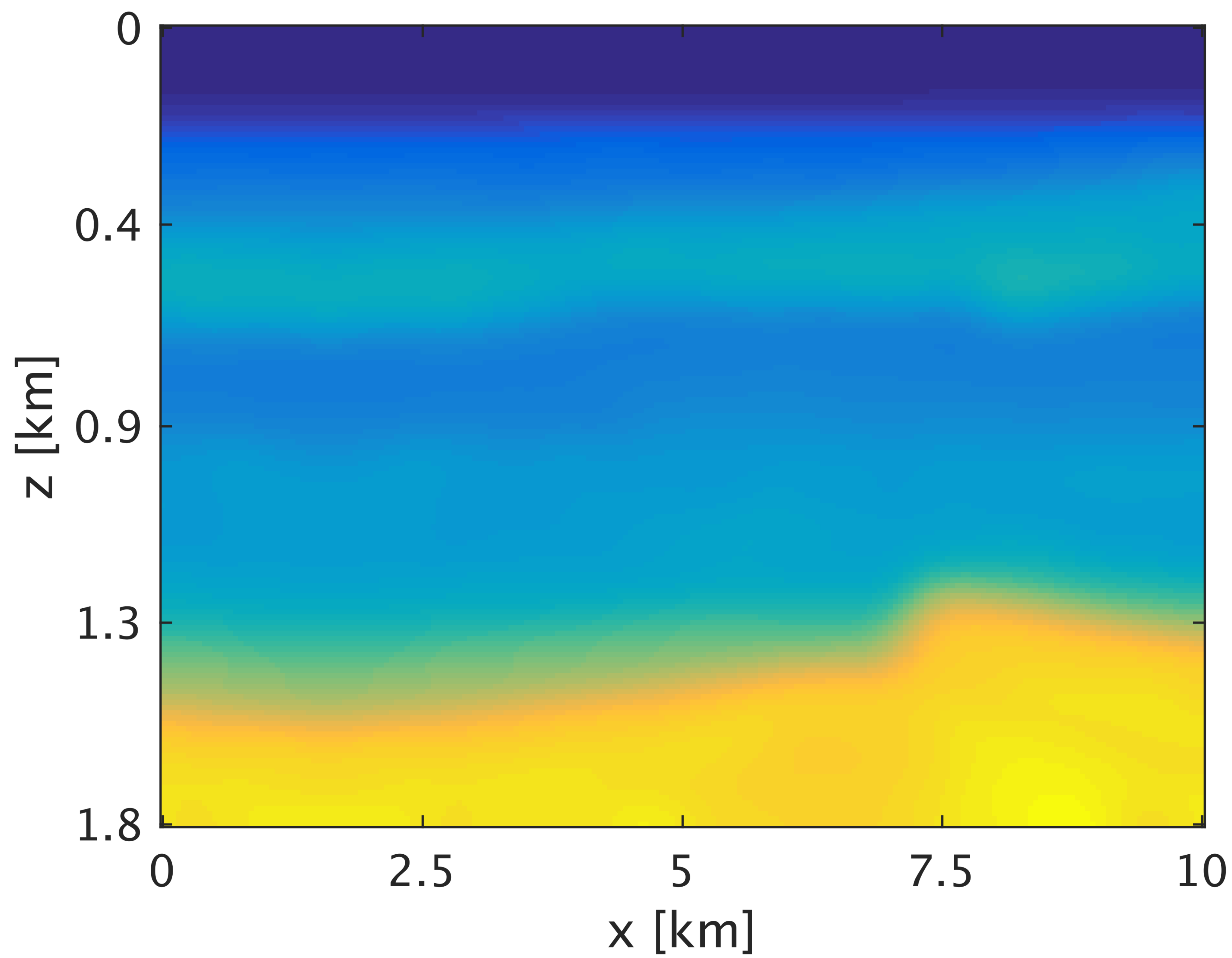


Interpolated data

y = 5650m lateral slice

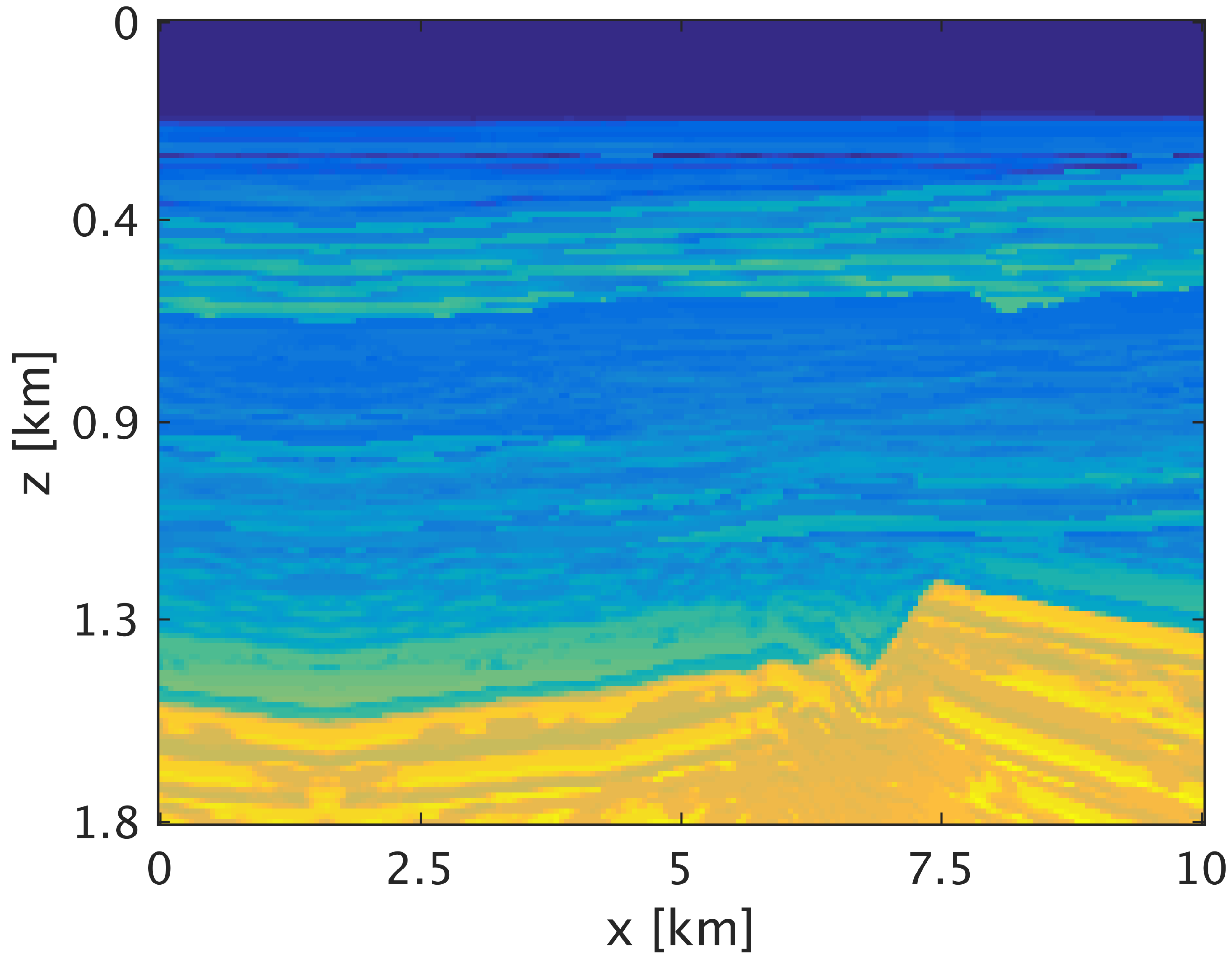


True model

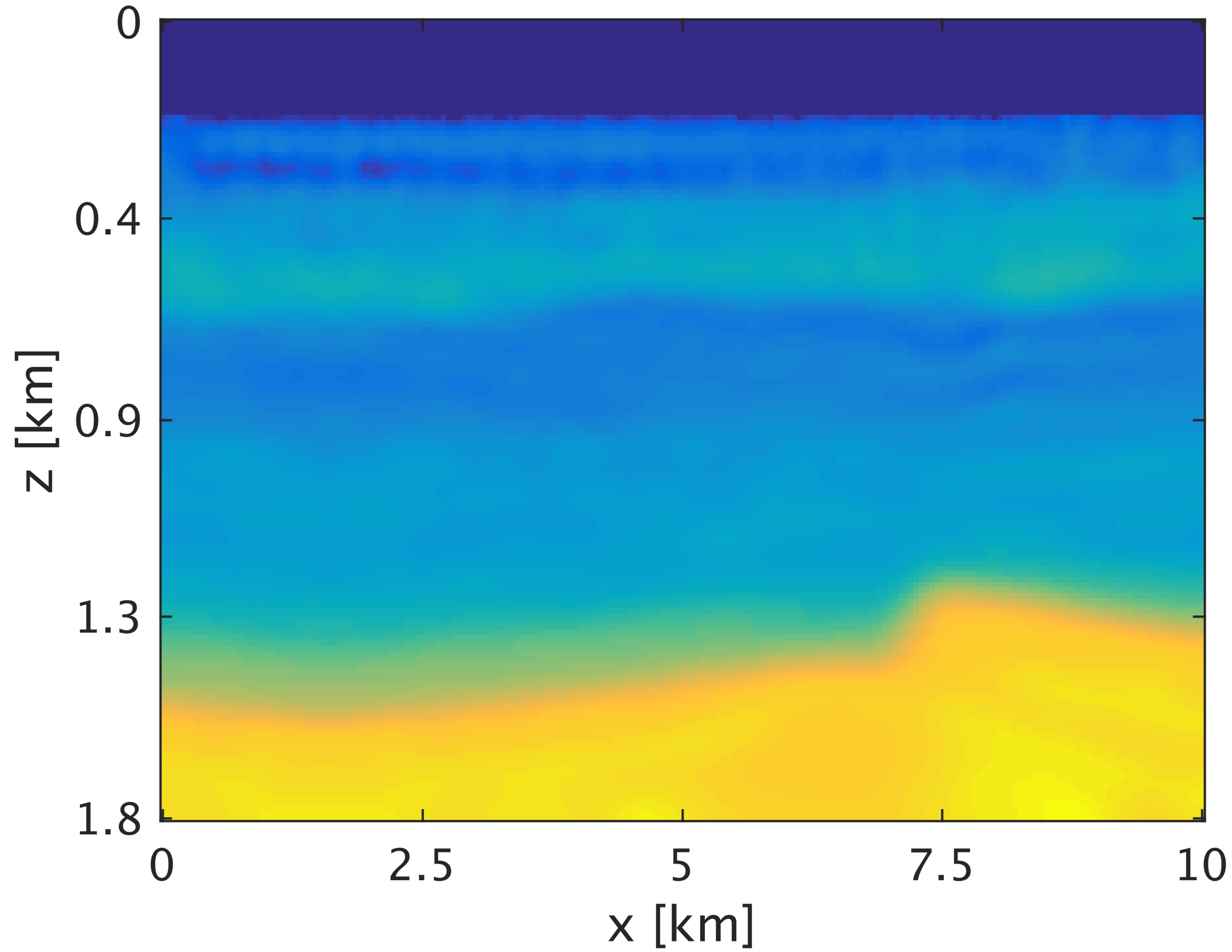


Initial model

y = 5650m lateral slice

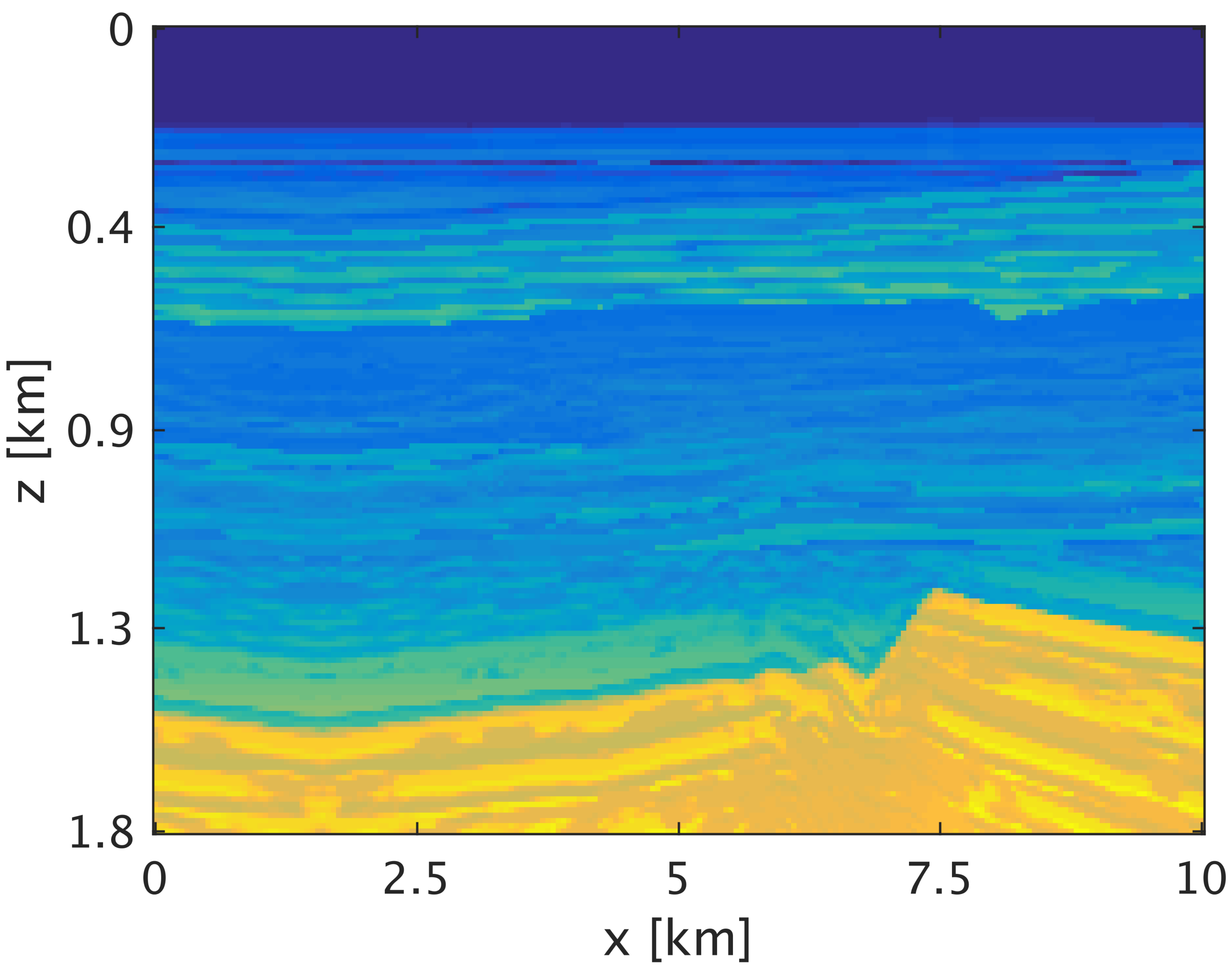


True model

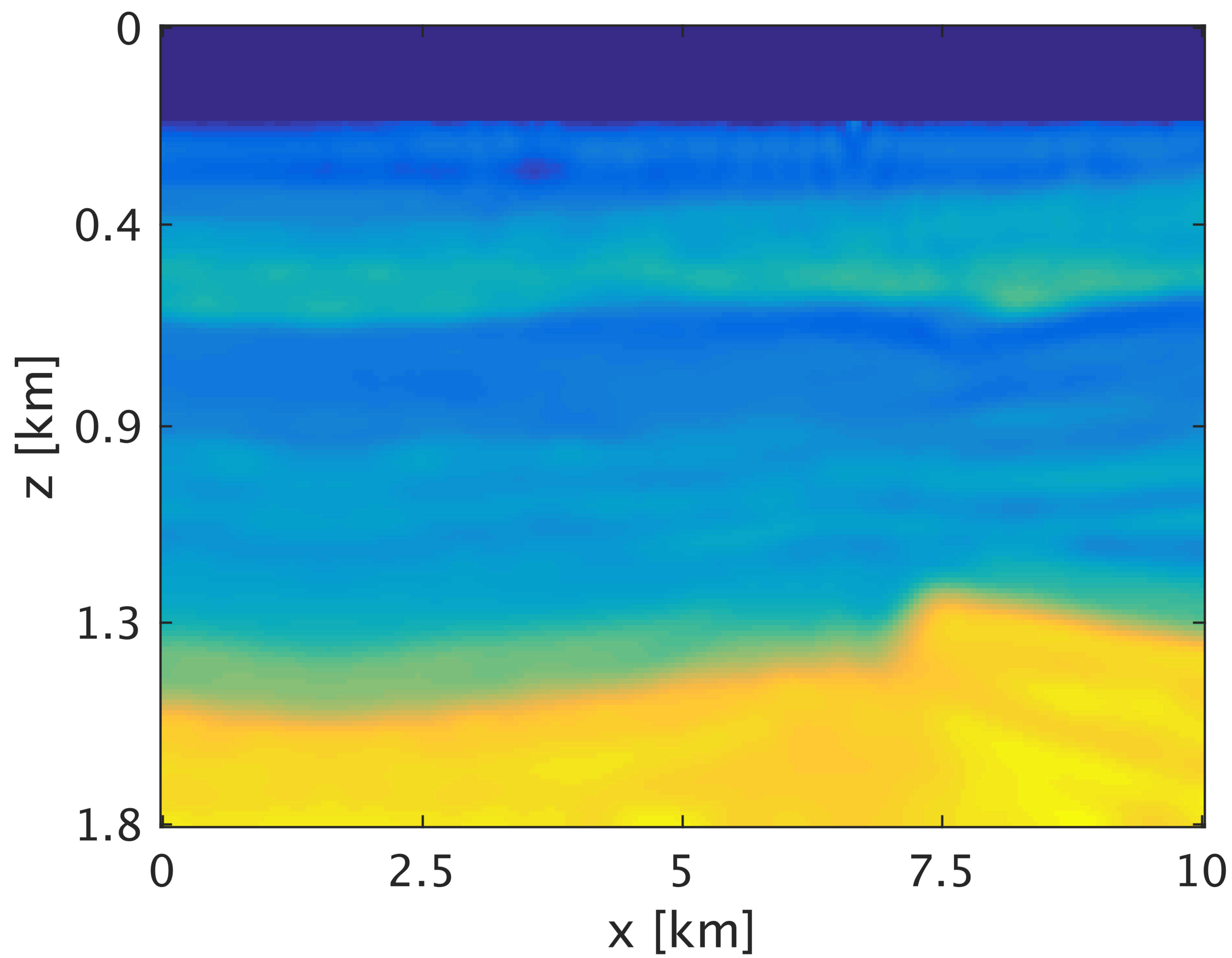


Subsampled data

y = 5650m lateral slice

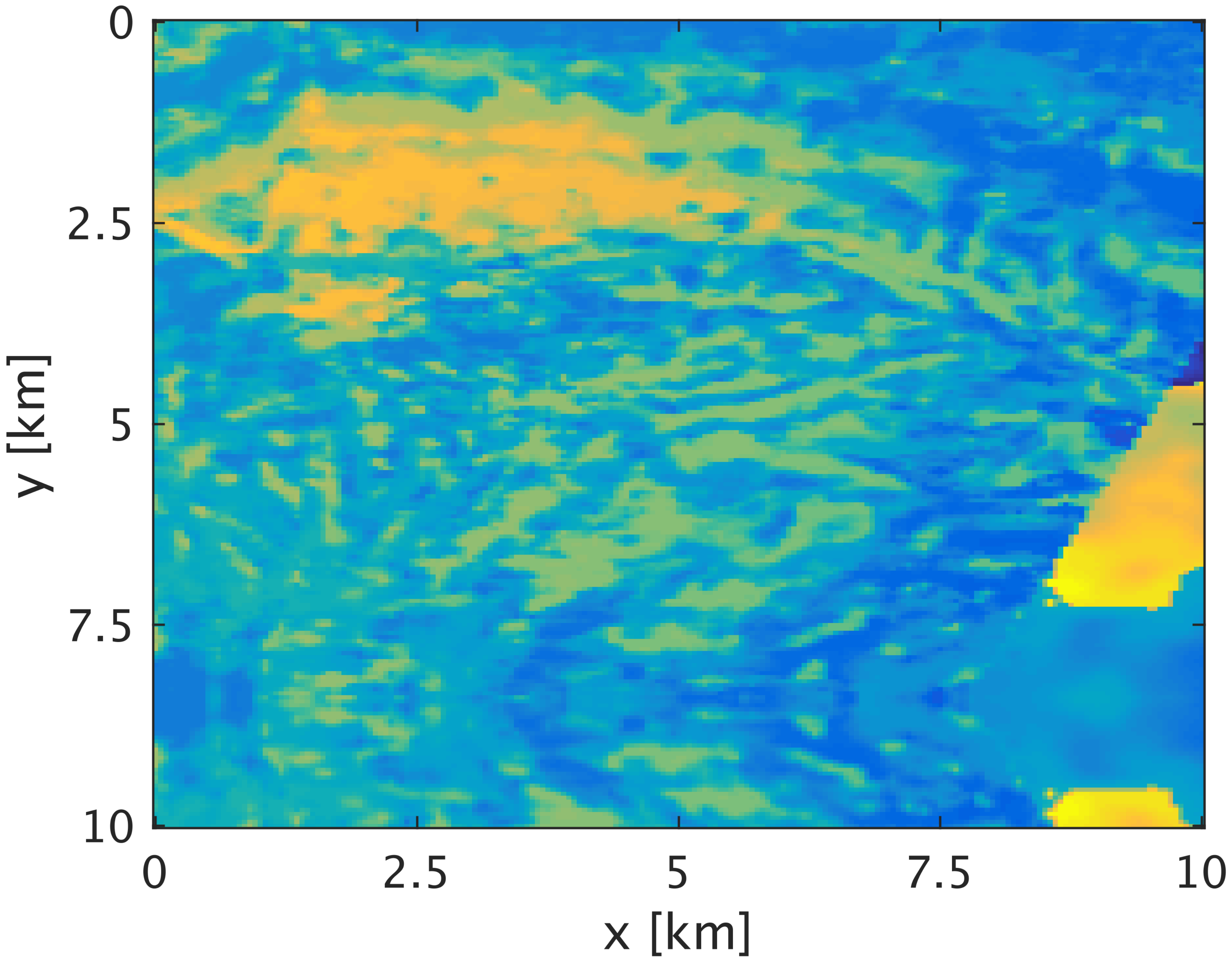


True model

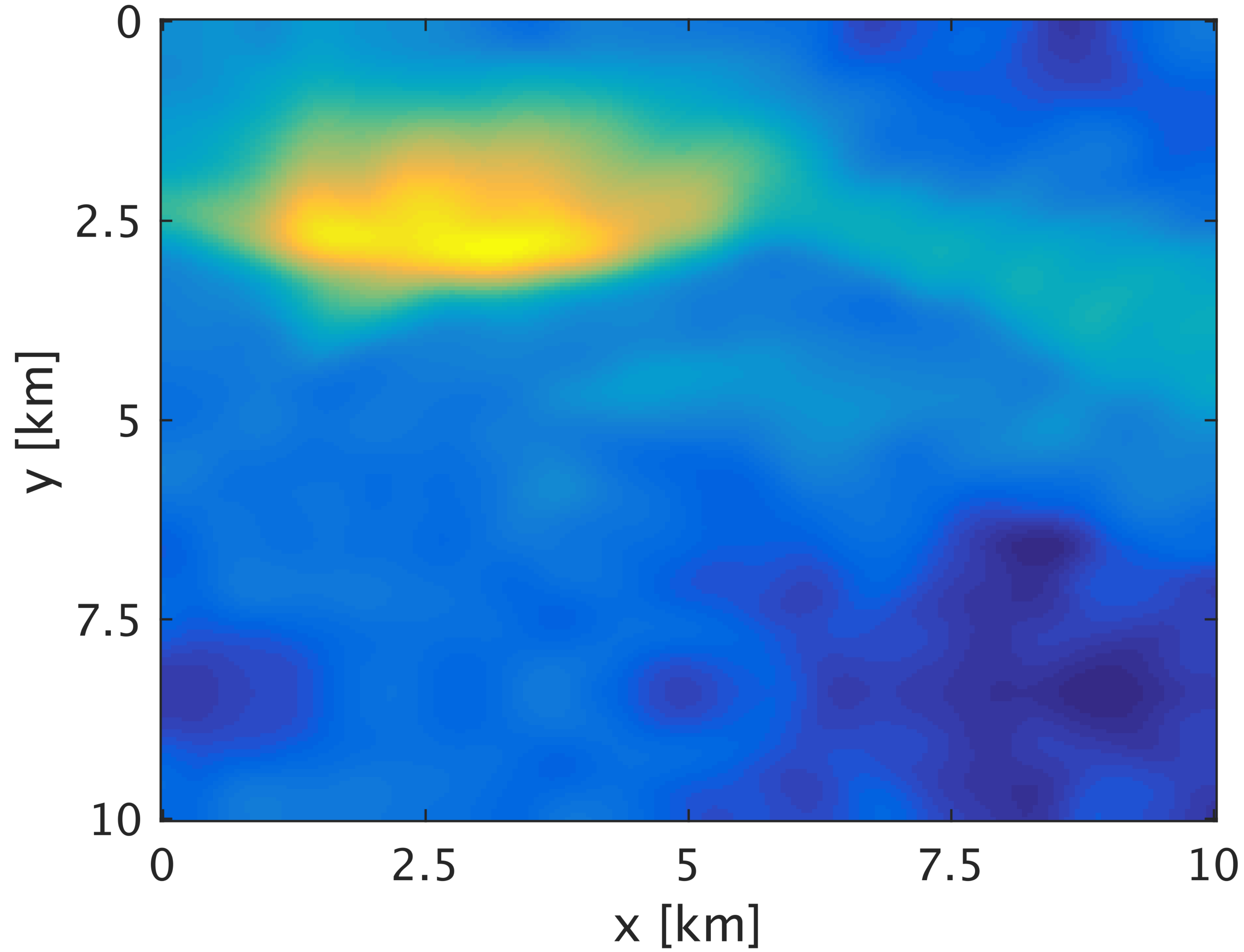


Interpolated data

z = 1200m depth slice

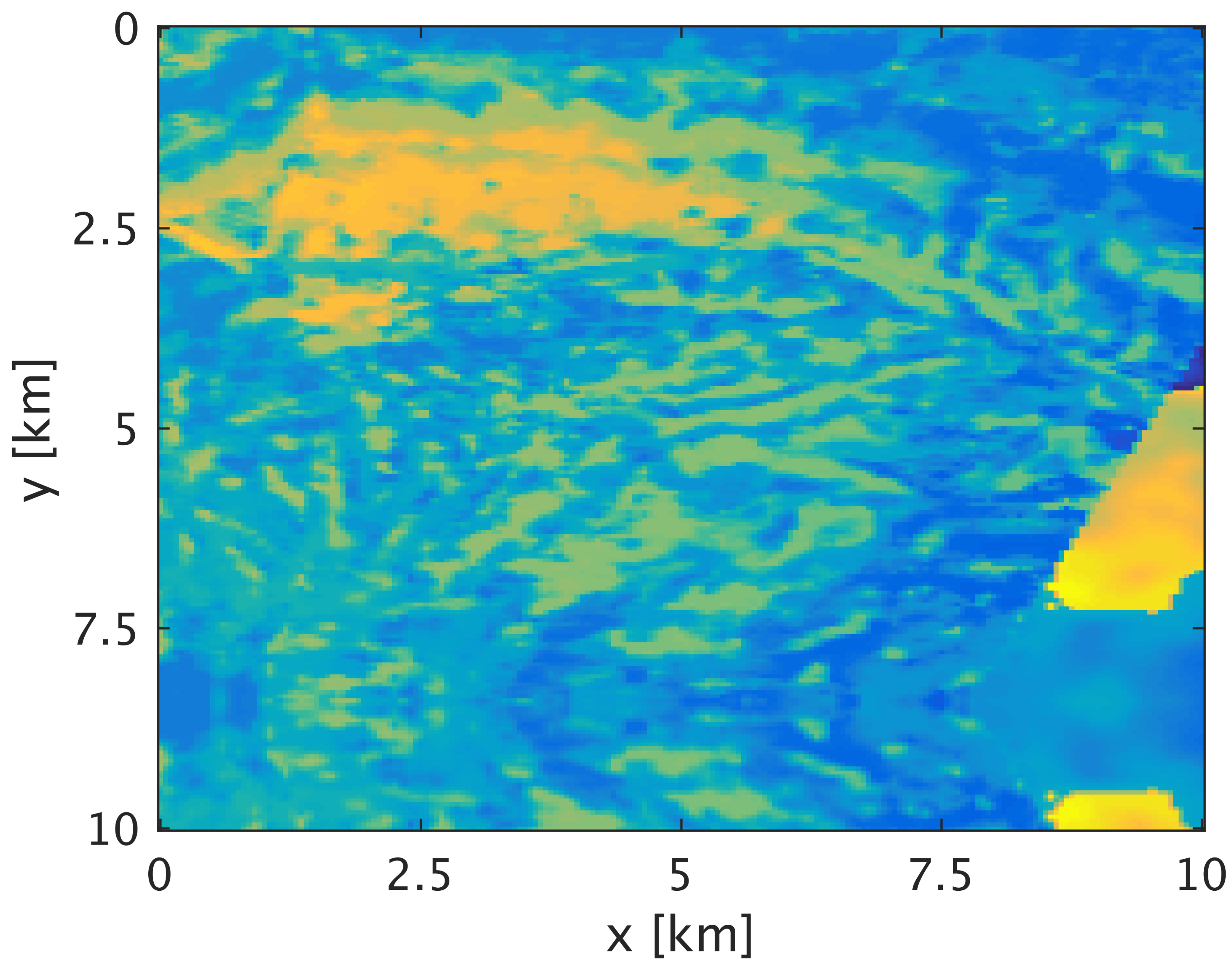


True model

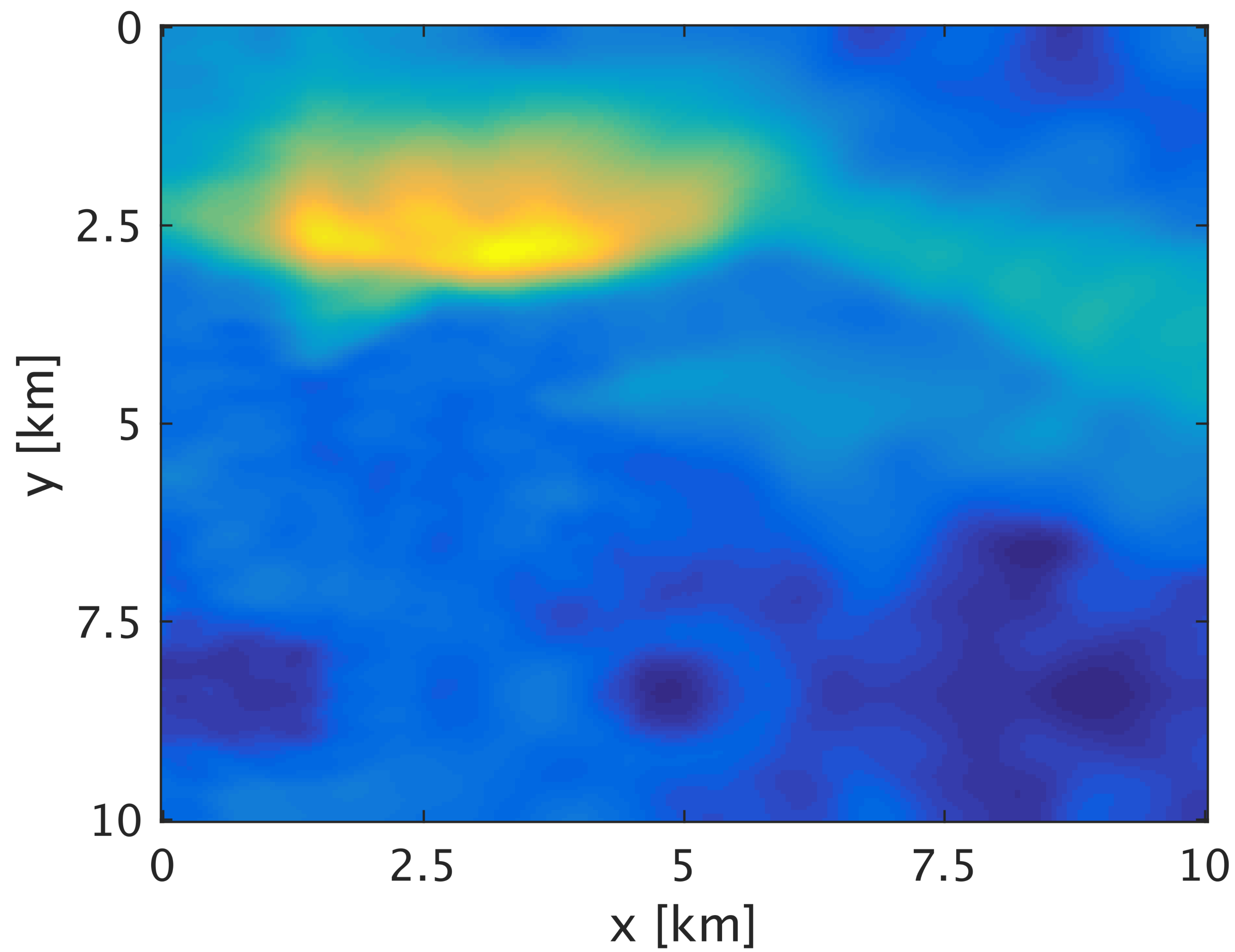


Initial model

z = 1200m depth slice

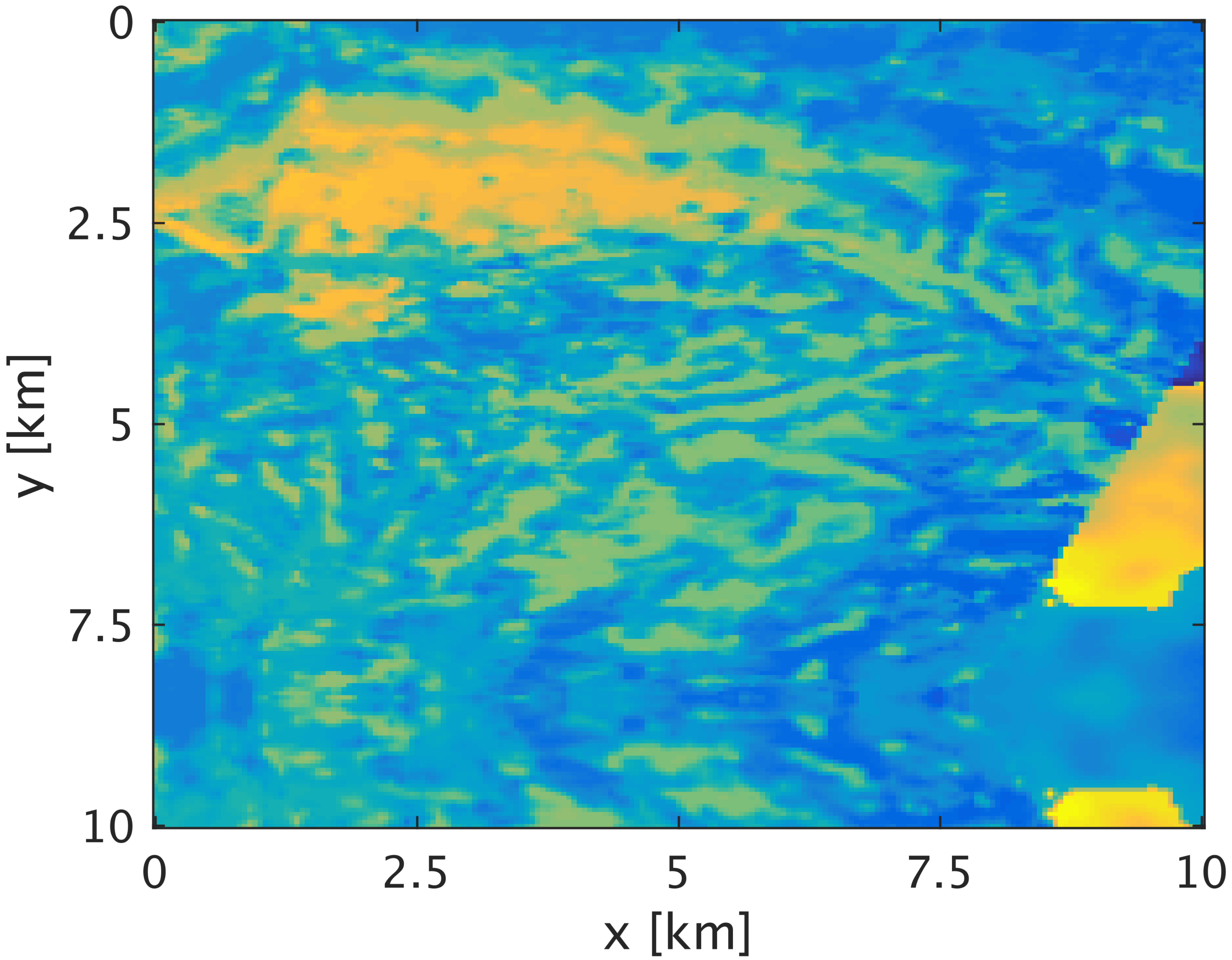


True model

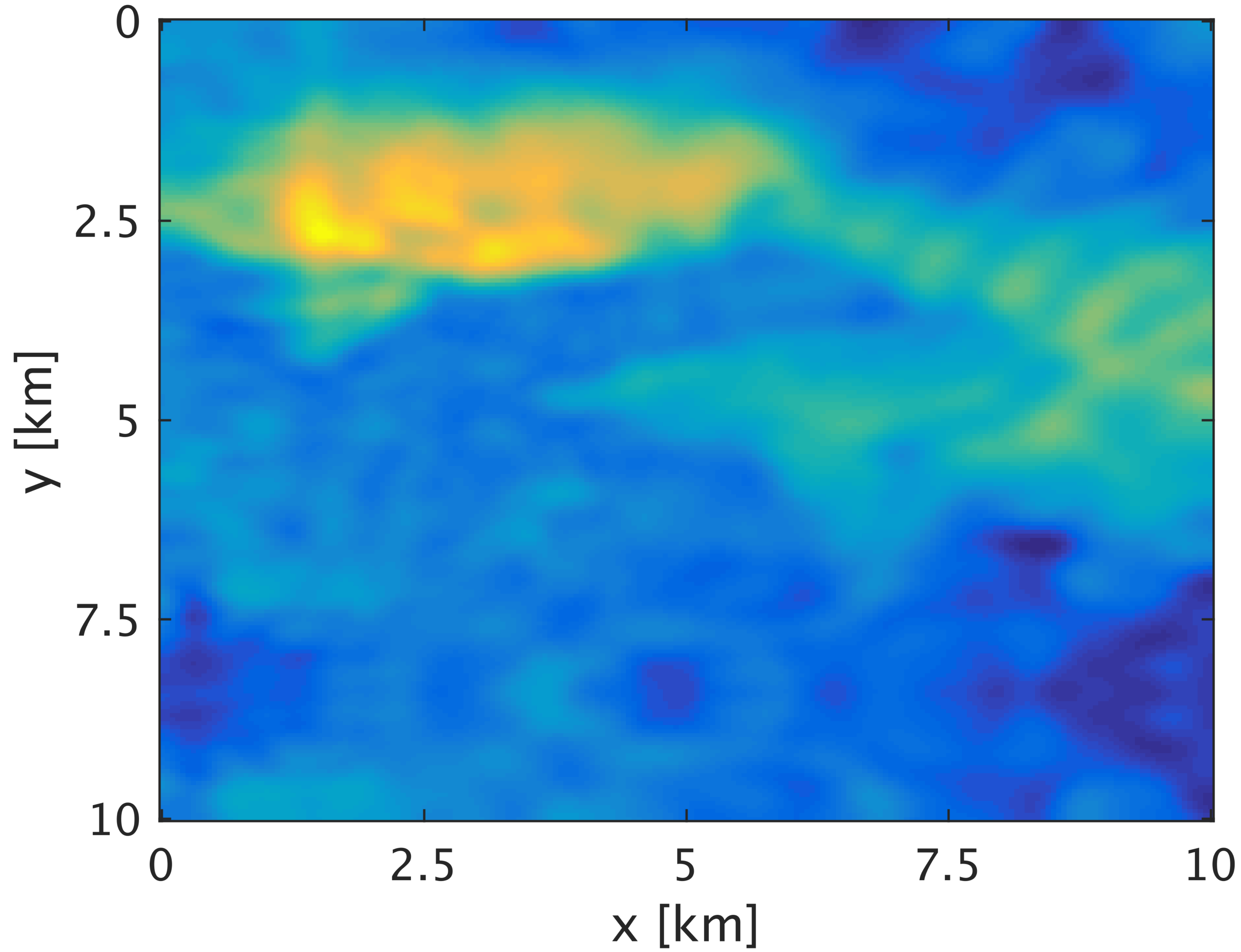


Subsampled data

z = 1200m depth slice



True model



Interpolated data

Case study 2: Extended Images

Extended images

Given two way wave equations, we define the **source** wavefield U and **receiver** wavefield V as

$$H(\mathbf{m})U = P_s^T Q$$
$$H(\mathbf{m})^* V = P_r^T D$$

where

$\mathbf{m}$: slowness

$H(\mathbf{m})$: discretization of the Helmholtz operator

$Q\ D$: source function and data matrix

$P_s^T\ P_r^T$: samples the wavefield at the source and receiver positions

Extended images

Organize wavefields in monochromatic data **matrices** where each **column** represents a **common** shot gather

Express image volume **tensor** for **single** frequency as a **matrix**

$$E = VU^*$$

Extended images

Image volume too l**arge** to form and too **expensive**

Instead, **probe** volume with **tall** matrix $W = [\mathbf{w}_1, \ldots, \mathbf{w}_\ell]$

$$\widetilde{E} = EW = H^{-*} P_r^\top D Q^* P_s H^{-*} W$$

where $\mathbf{w}_i = [0, \ldots, 0, 1, 0, \ldots, 0]$ represents **single** scattering points

Proposed method

$$\tilde{E} = H^{-*} P_r^T D Q^* P_s H^{-*} w$$

SLIM

Proposed method

$$\tilde{E} = H^{-*} P_r^T D Q^* P_s H^{-*} w$$

Proposed method

$$\tilde{E} = H^{-*} P_r^T D Q^* P_s H^{-*} w$$

$$\downarrow$$

$$v \in \mathbb{C}^{n_s \times 1}$$

Proposed method

$$\tilde{E} = H^{-*} P_r^T D Q^* P_s H^{-*} w$$

$$\downarrow$$

$$\tilde{E} = H^{-*} P_r^T D v$$

Algorithm 1

$$\tilde{E} = H^{-*} P_r^T D v$$

Single common shot gather extraction technique

Algorithm 1

$$\tilde{E} = H^{-*} P_r^T Dv$$

Single common shot gather extraction technique

Input: $\mathbf{v} = Q^* P_s \mathbf{H}^{-*} \mathbf{w}$
Output: $\mathbf{z} = \mathbf{D}\mathbf{v}$
For each source index $\mathbf{i} = (i_{\mathrm{src}_x}, i_{\mathrm{src}_y})$

1. Extract the common shot gather from the data using our proposed, resulting in $\mathbf{D_i}$;
2. Scale $\mathbf{D_i}$ by a scalar $\mathbf{v}_{\mathrm{i}}$ to produce $\mathbf{z}$;
3. Update $\mathbf{z}$ with addition of previous $\mathbf{z}$.

Algorithm 2

$$\tilde{E} = H^{-*} P_r^T Dv$$

Simultaneous common shot gathers extraction technique

Algorithm 2

$$\tilde{E} = H^{-*} P_r^T Dv$$

Simultaneous common shot gathers extraction technique

Input: $\mathbf{v} = Q^* P_s \mathbf{H}^{-*} \mathbf{w}$
Output: $\mathbf{z} = \mathbf{D}\mathbf{v}$
For source indices $\mathbf{i} = (i_{\mathrm{src}_x}, \mathrm{src}_y)$

1. Extract simultaneous common shot gathers from the data using our proposed, resulting in $\mathbf{D_i}$
2. Multiply $\mathbf{D_i}$ with $\mathbf{v}(i,:)$ to produce $\mathbf{z}$
3. Update $\mathbf{z}$ with addition of previous $\mathbf{z}$

Common Image-point gather

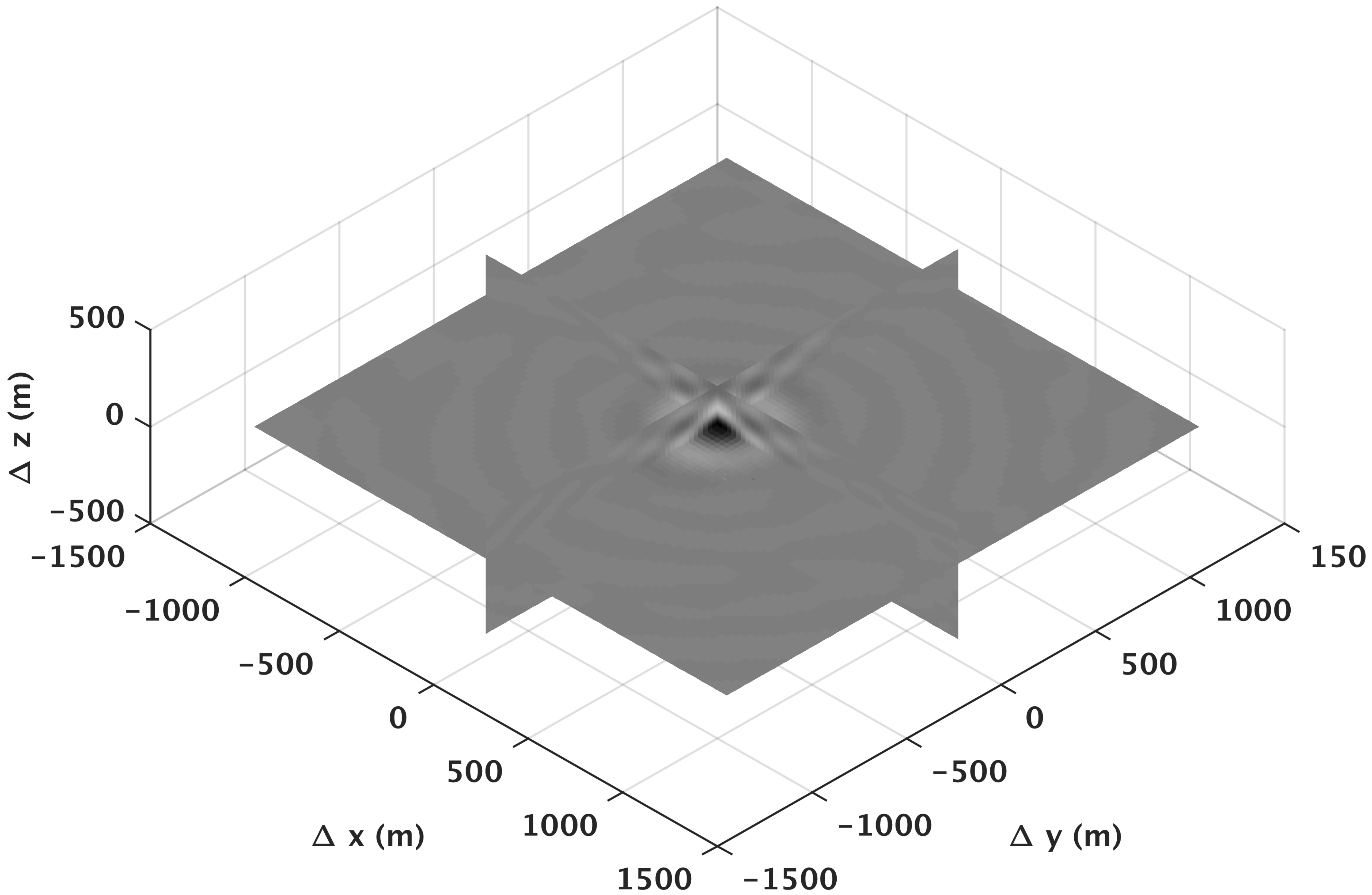
Model:

- 3D **BG** model
- 1.25km x 1.25km x 0.39 km
- 25m x 25m x 6 m spacing

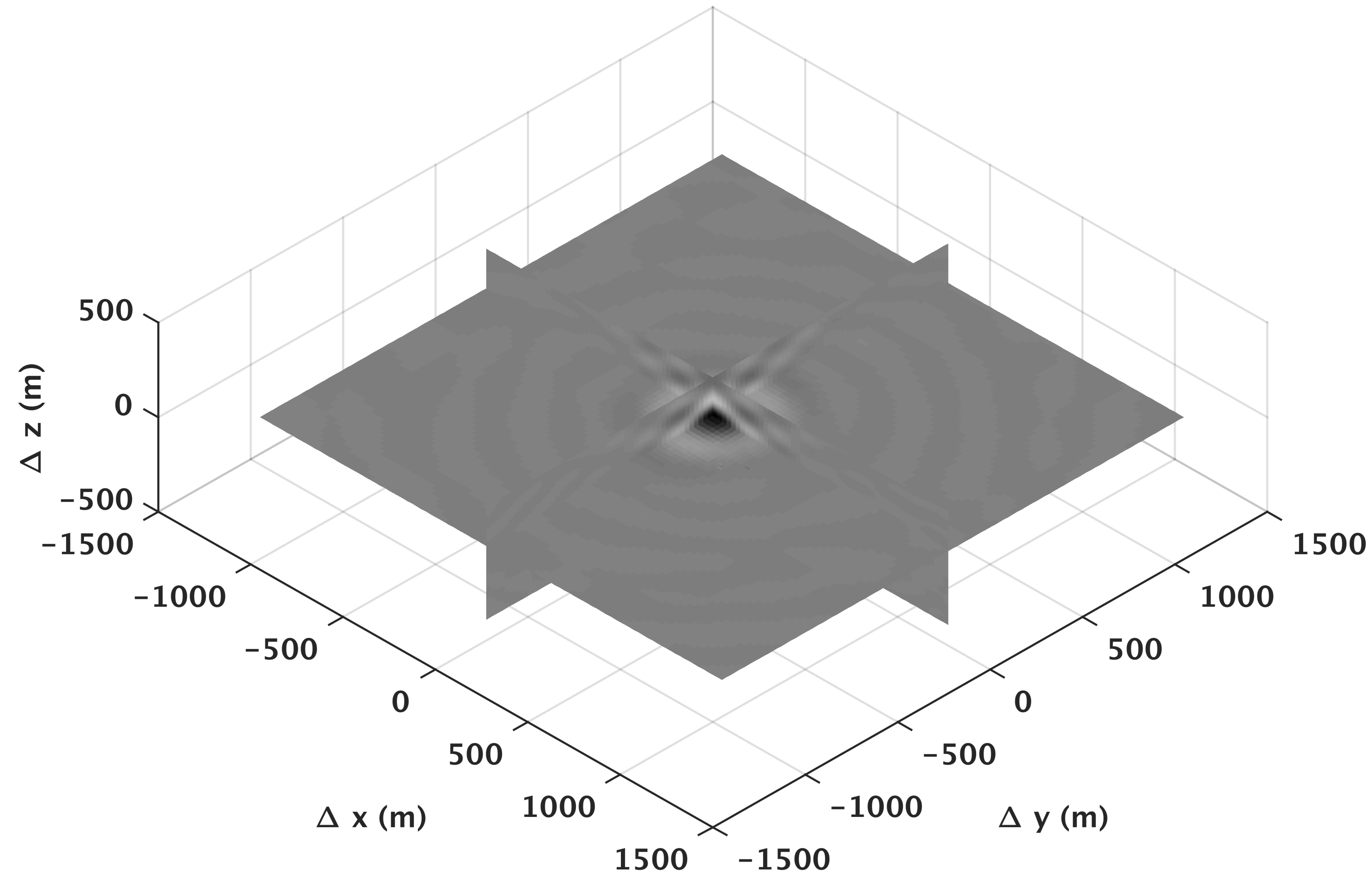
Experiment details:

- OBN acquisition
- 1156 sources (75m spacing), 2601 receivers (50m spacing)
- Ricker wavelet, 15 peak frequency
- 5-12 Hz, 0.5Hz interval

Common-point gather at (1250m, 1250m, 390m)

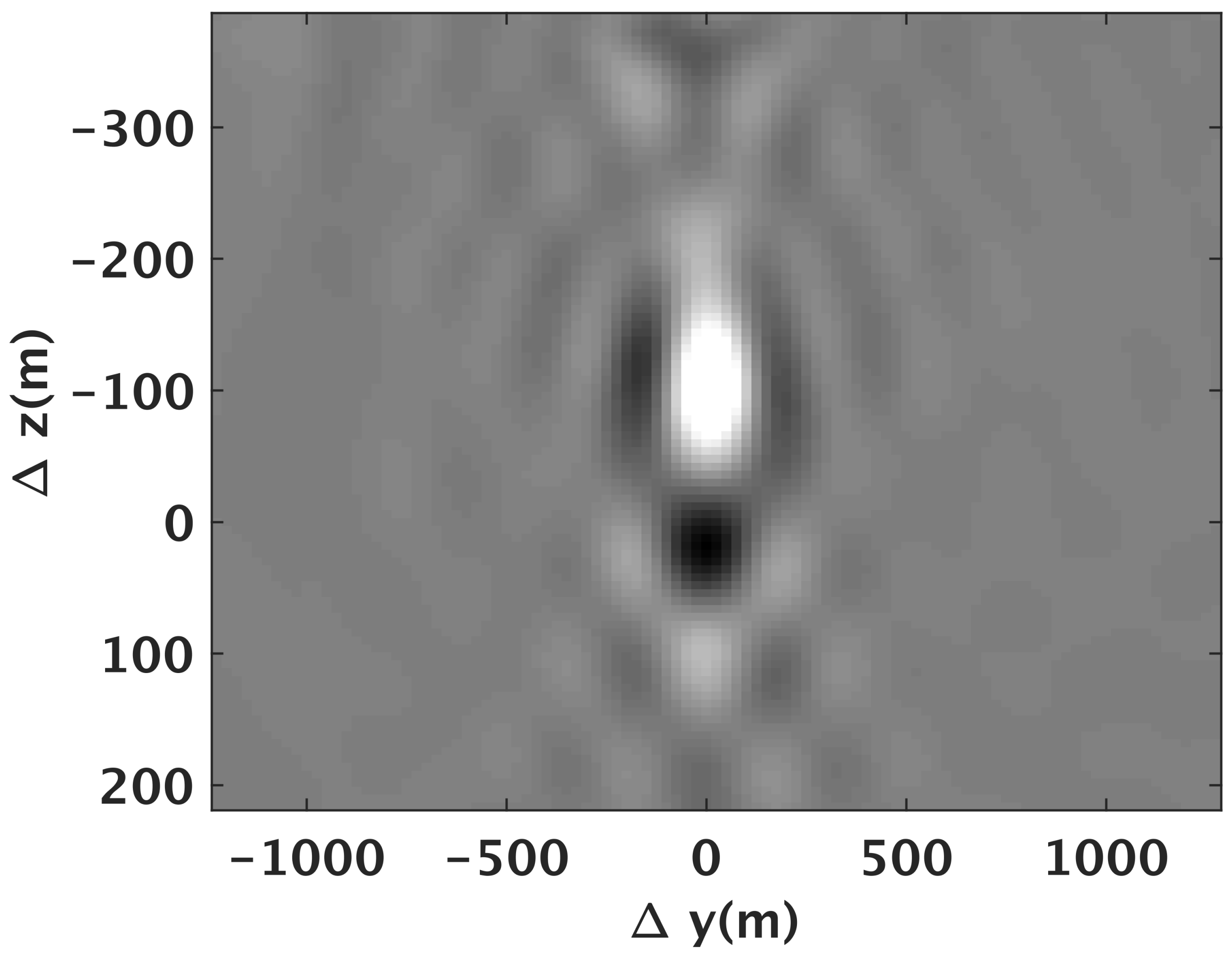


Full data

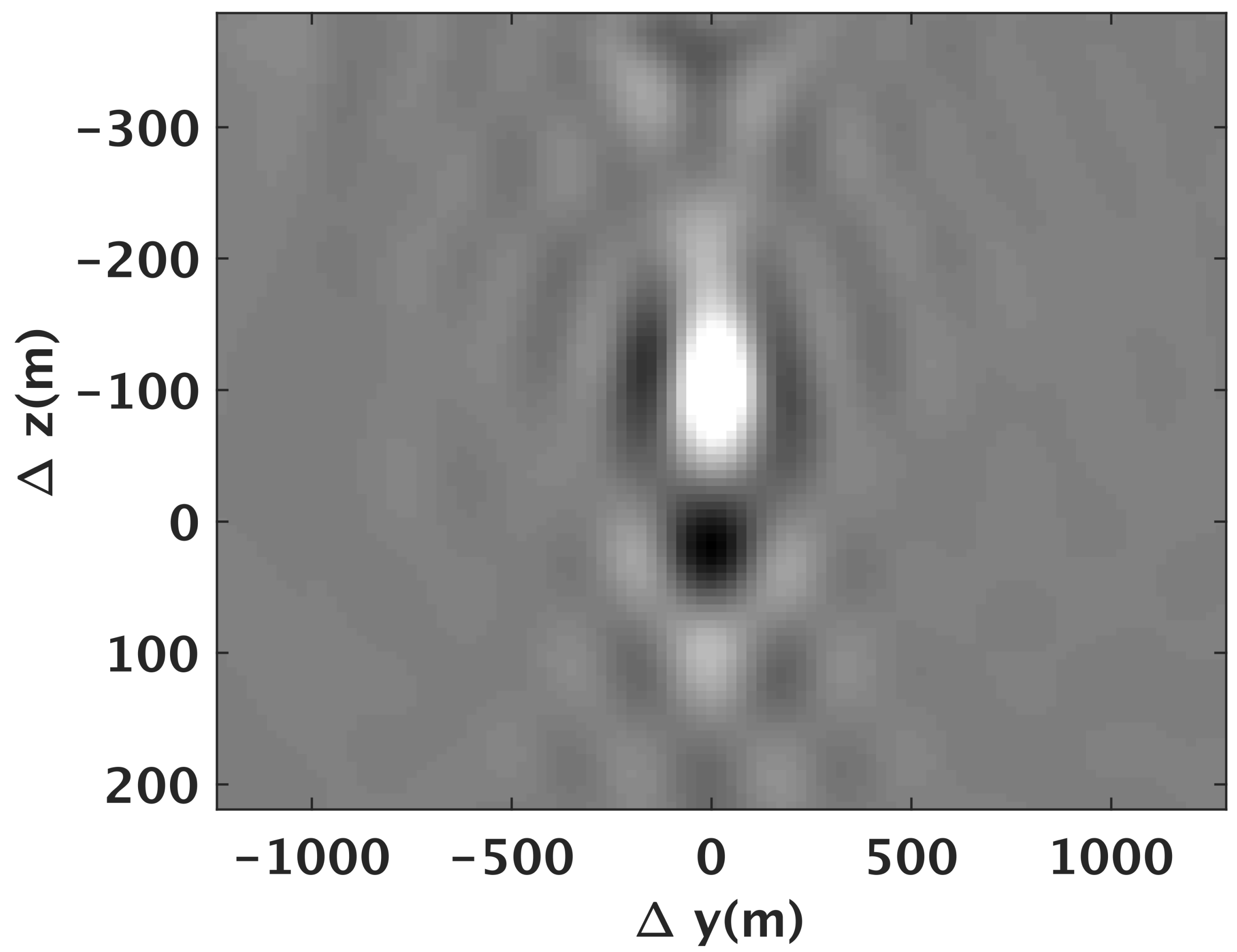


Compressed data

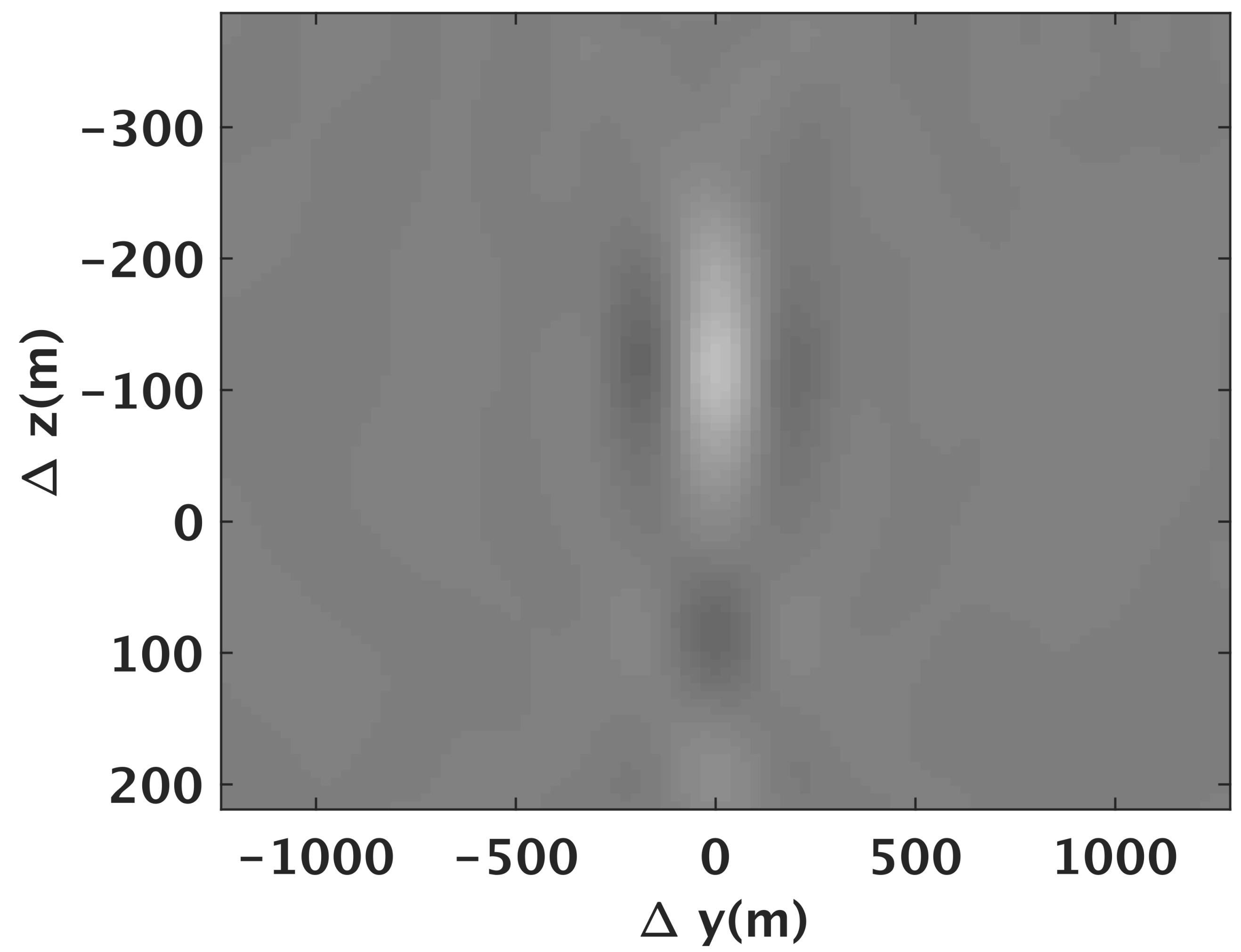
Along lateral offset direction



Full data

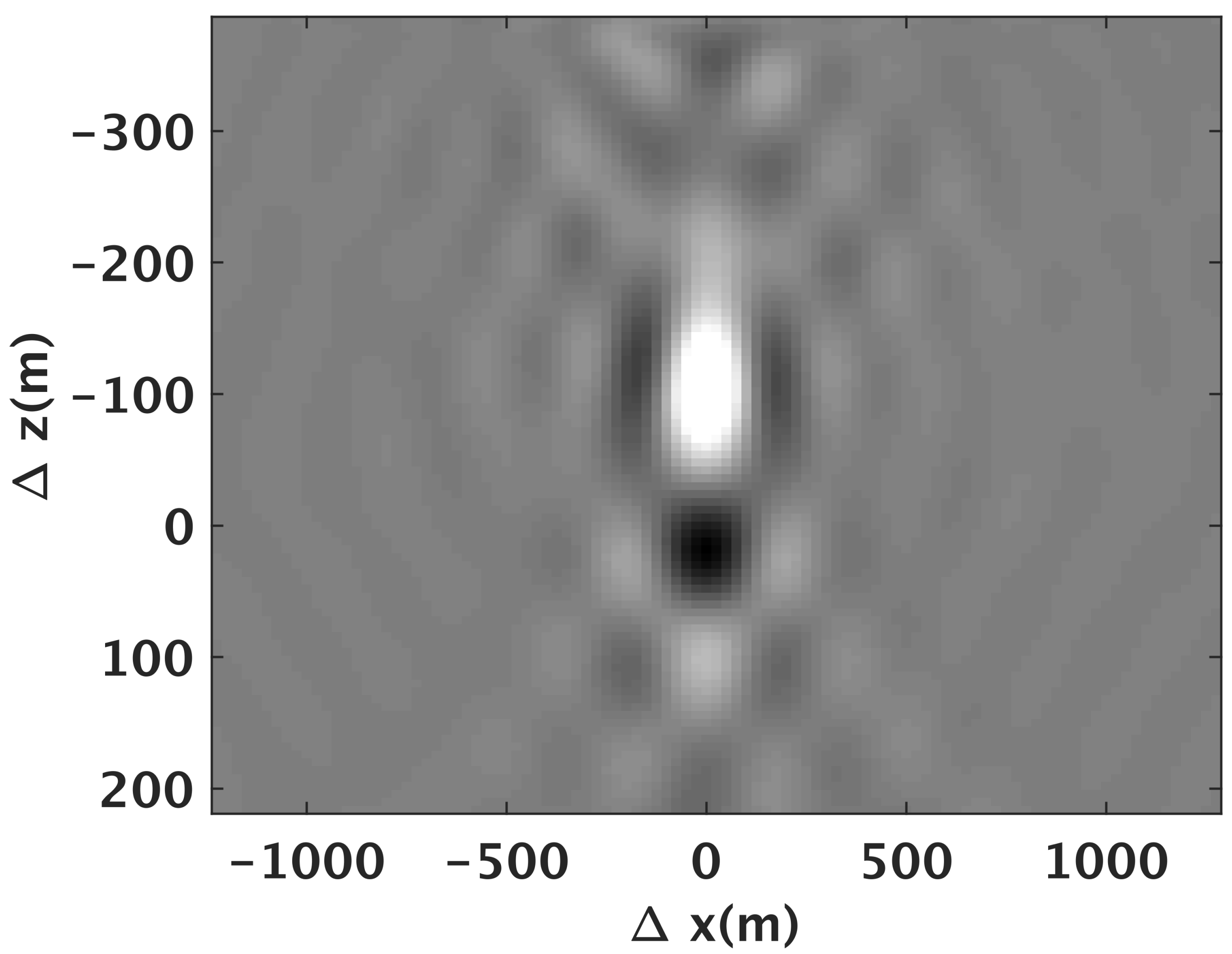


Compressed data

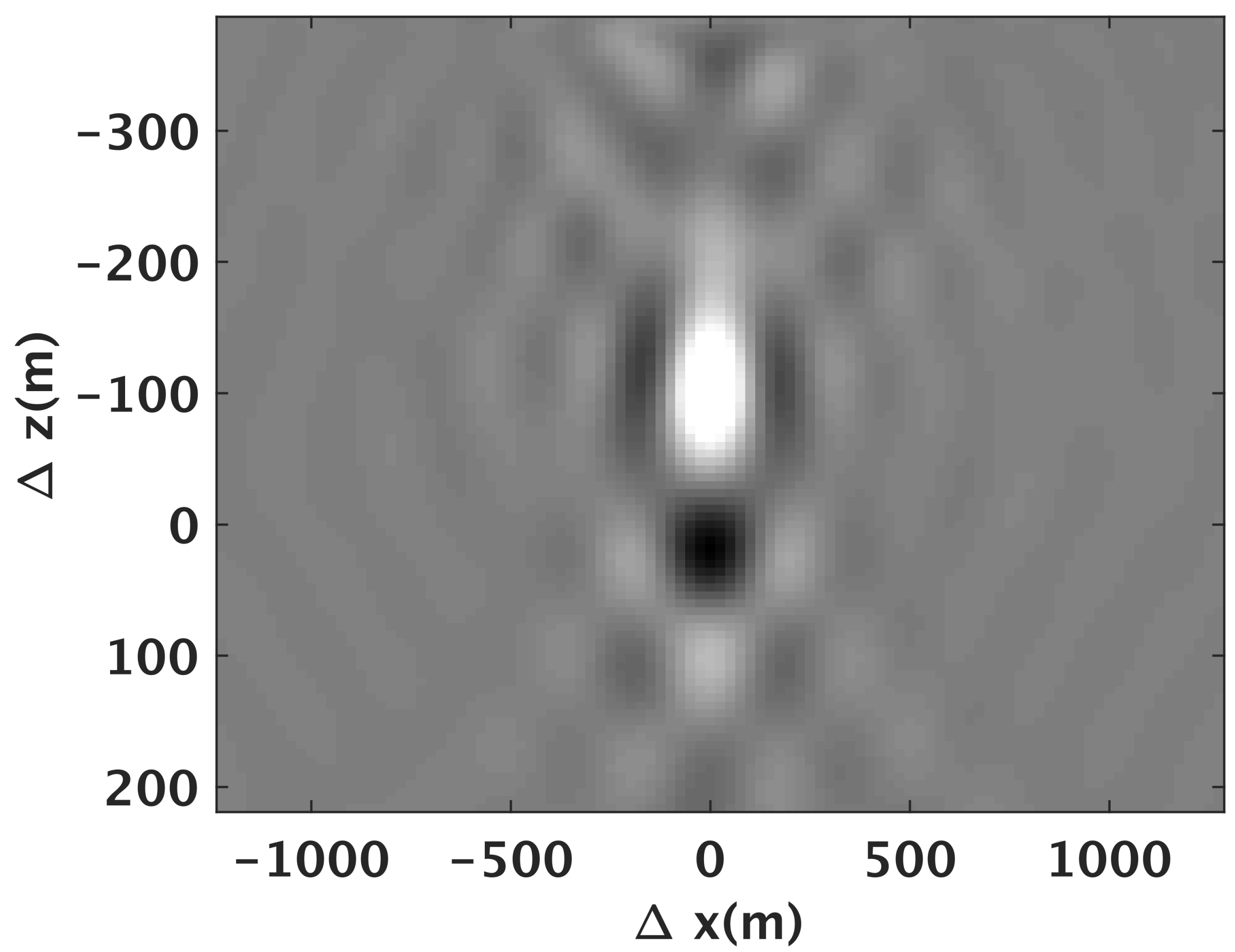


Difference x 100

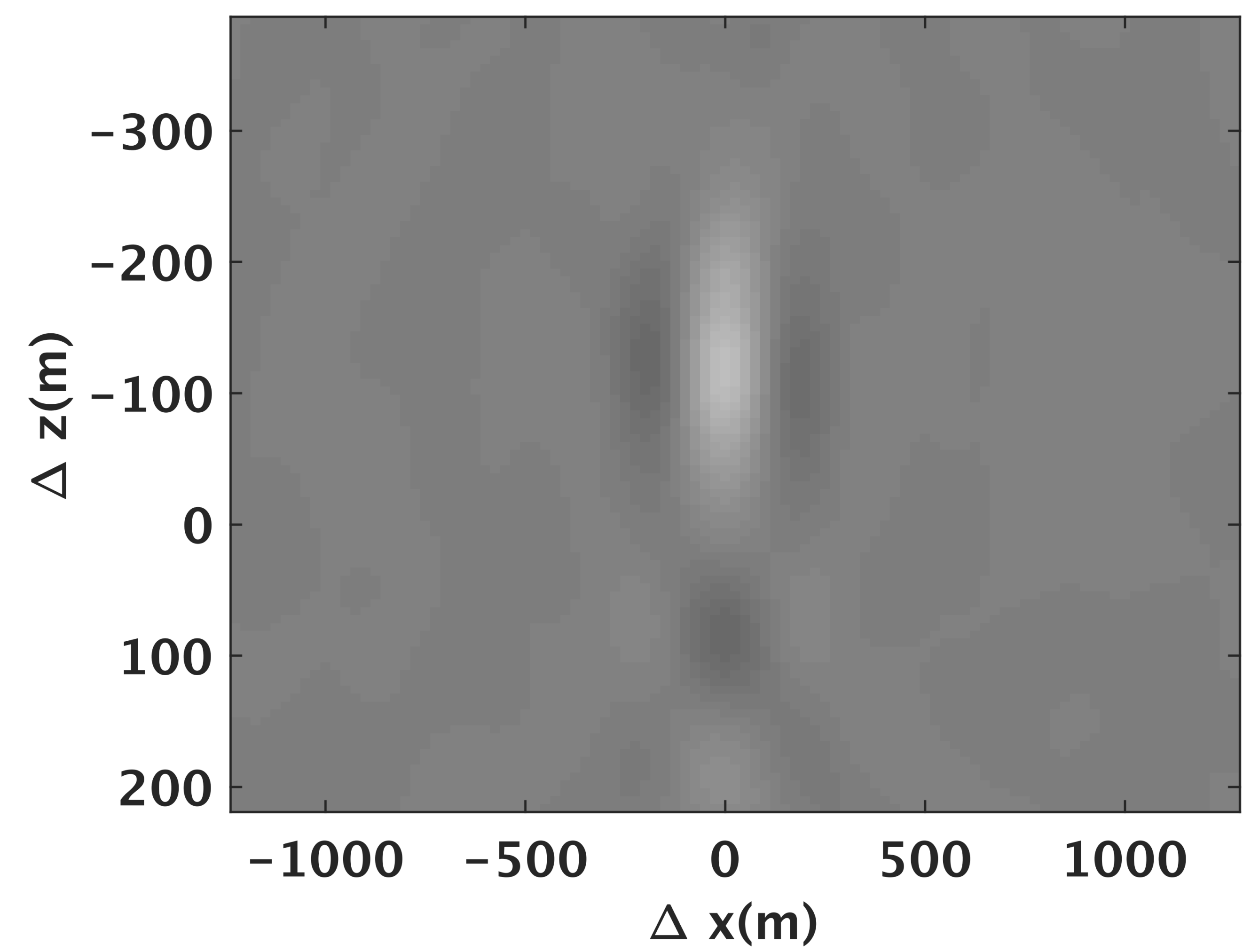
Along lateral offset direction



Full data

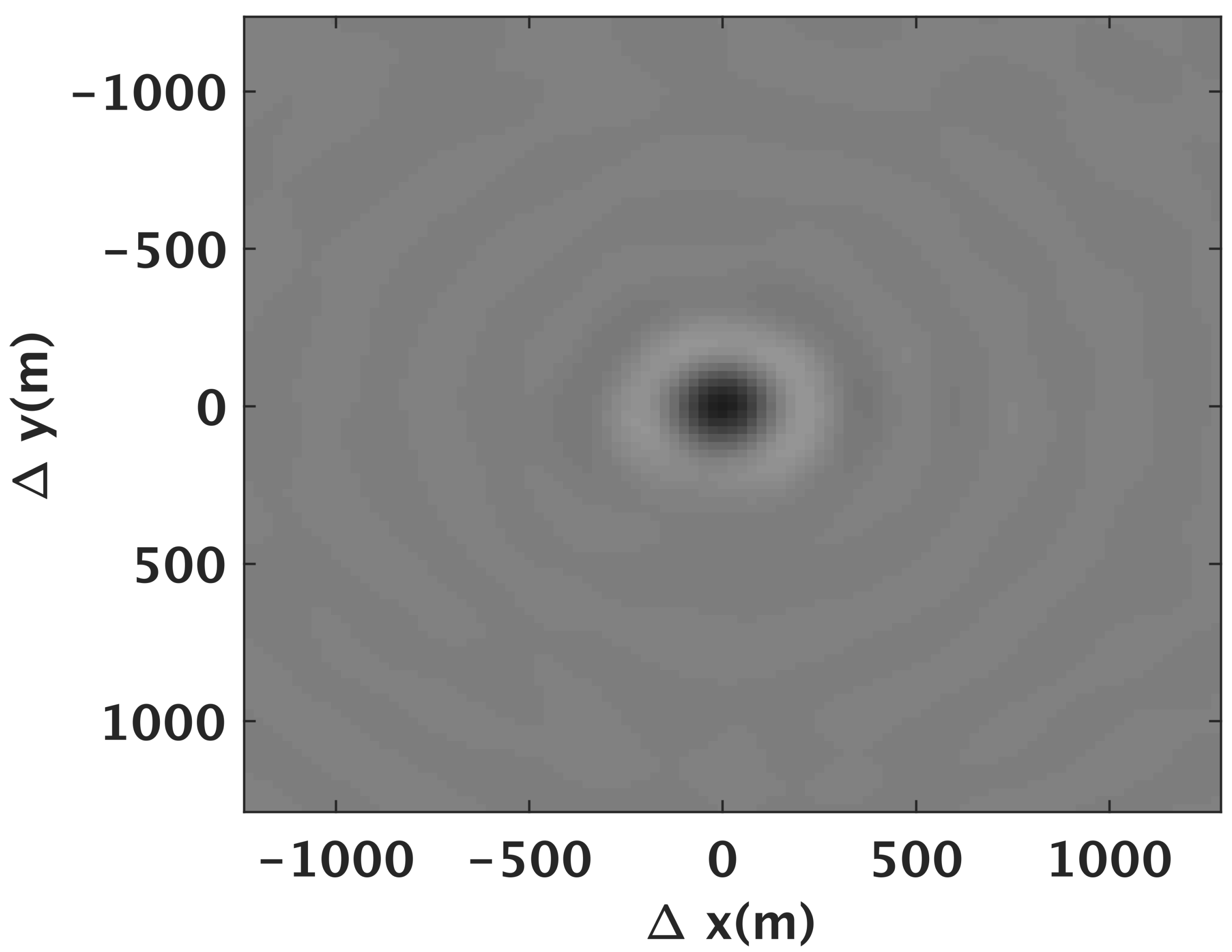


Compressed data

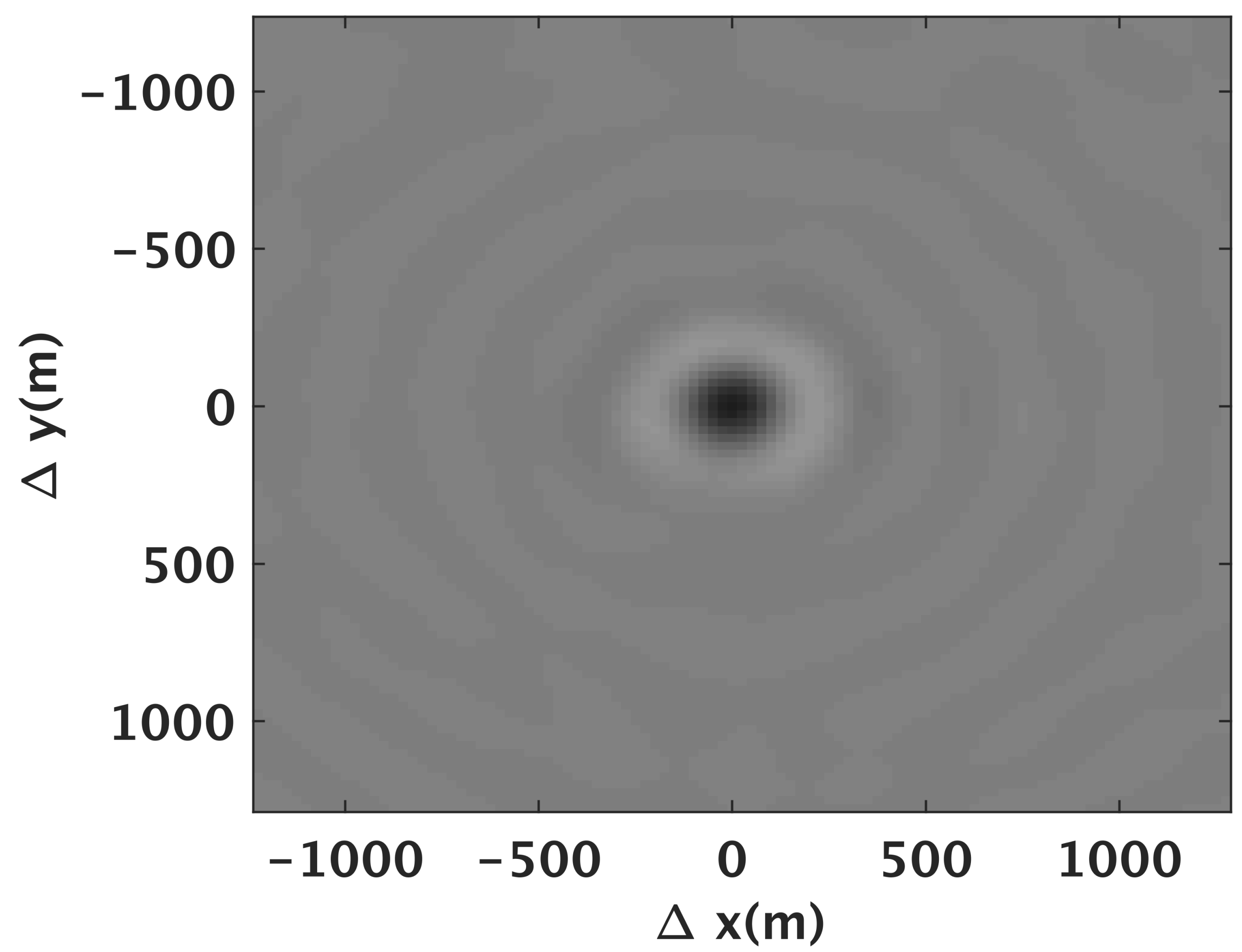


Difference x 100

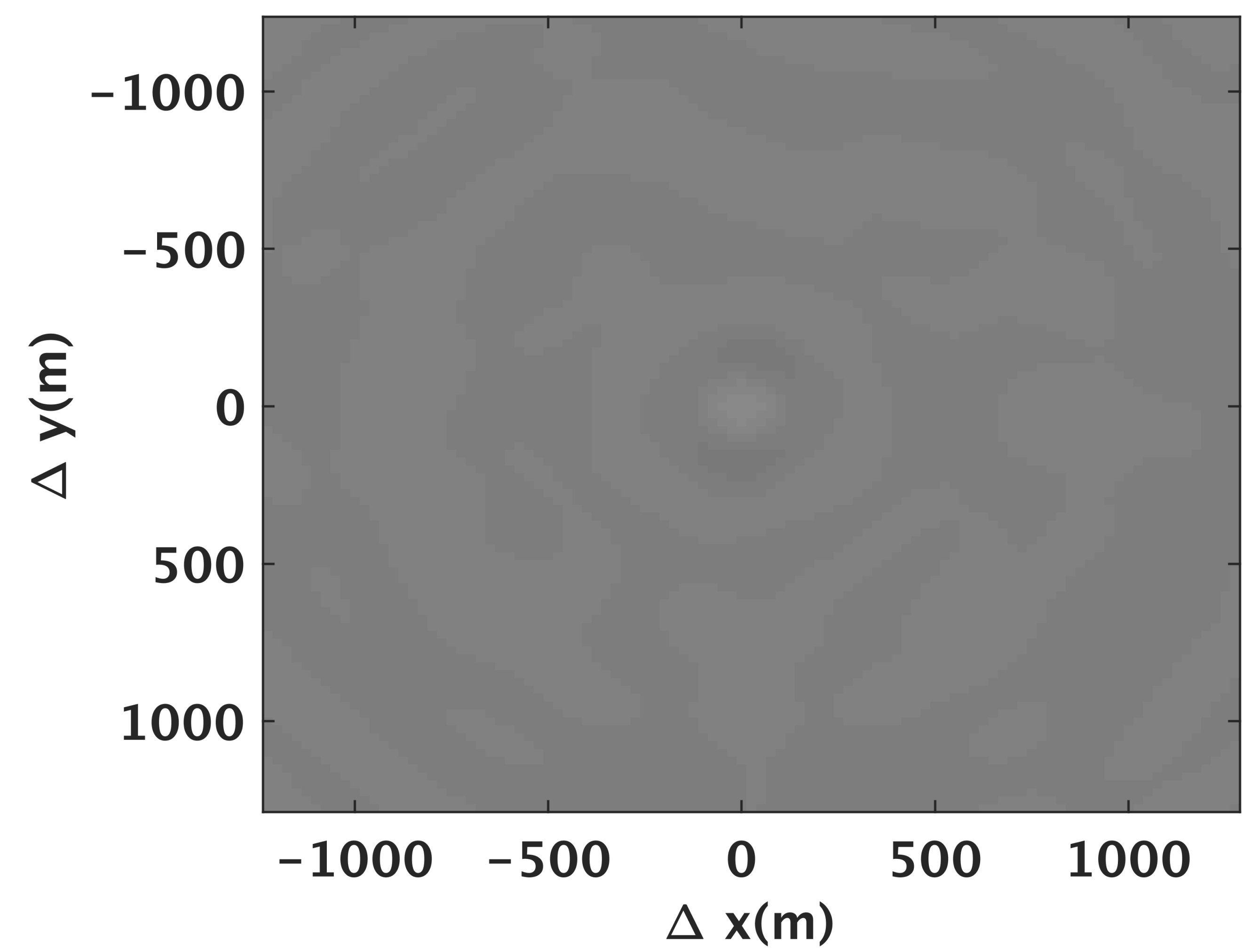
Along vertical offset direction



Full data



Compressed data



Difference x 100

Conclusions

- **high** compression ratio is achievable
- reduces **memory & computational** costs when combined w/ stochastic optimization/ probing technique
- codes **easily embedded** into other processing frameworks
- leads to major reduction in IO for **low-frequency** full waveform inversion & extended images
- suitable for both **fully sampled** data and **missing random** receivers/shots

Acknowledgements

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

SLIM

Acknowledgements

The authors wish to acknowledge the SENAI CIMATEC Supercomputing Center for Industrial Innovation, with support from BG Brasil, Shell, and the Brazilian Authority for Oil, Gas and Biofuels (ANP), for the provision and operation of computational facilities and the commitment to invest in Research & Development.

Thank you for your attention