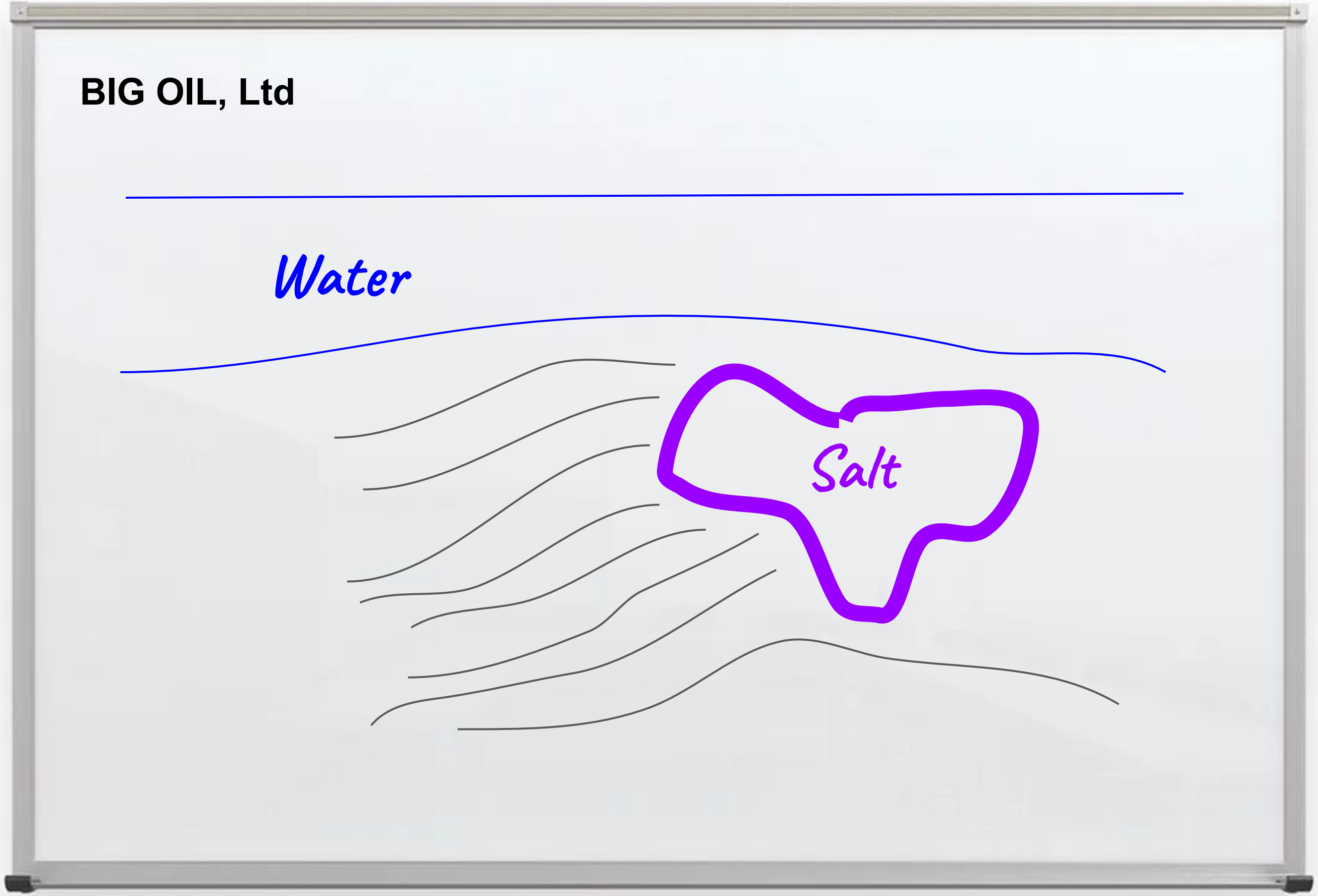
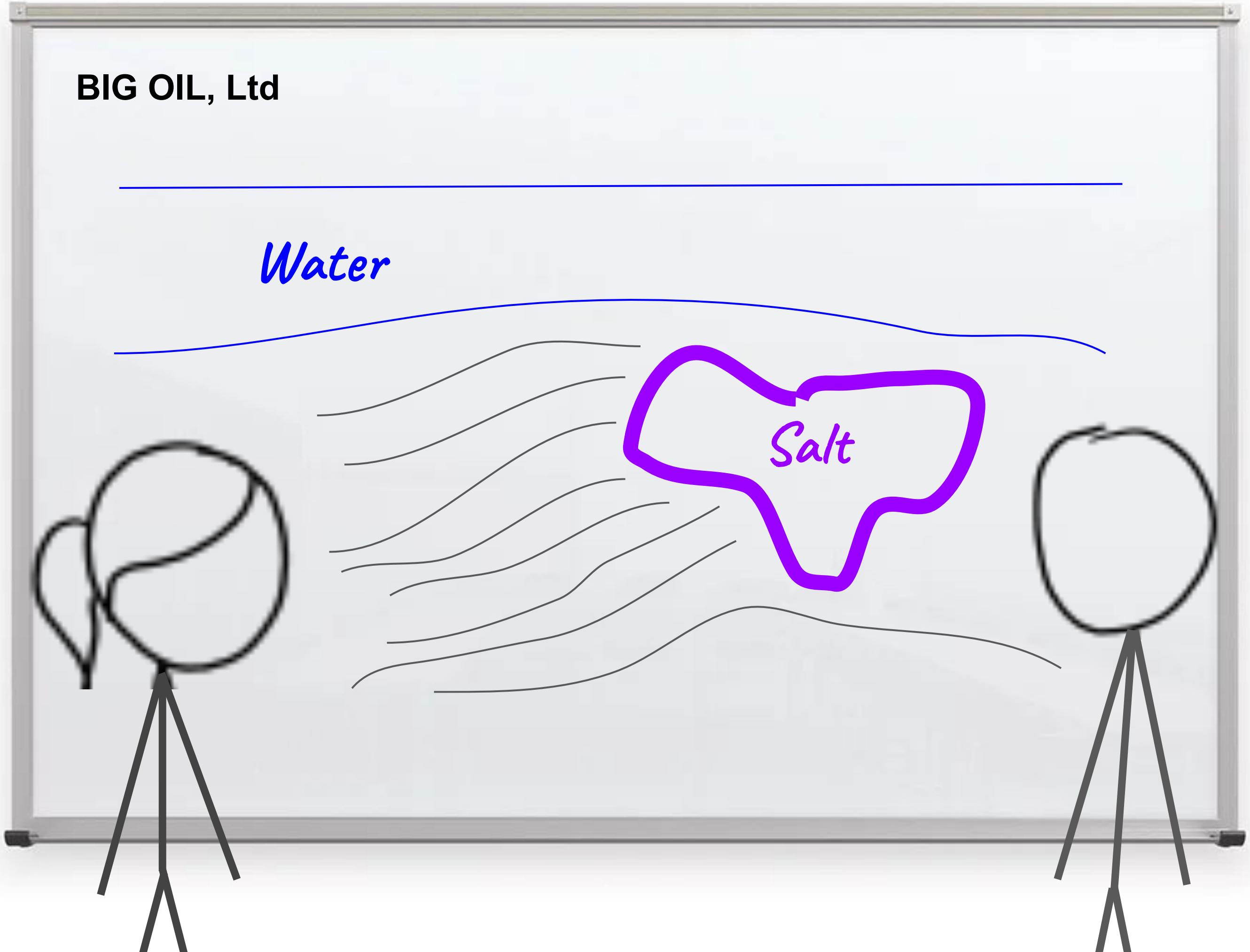
Velocity model building using shape optimization applied to level sets

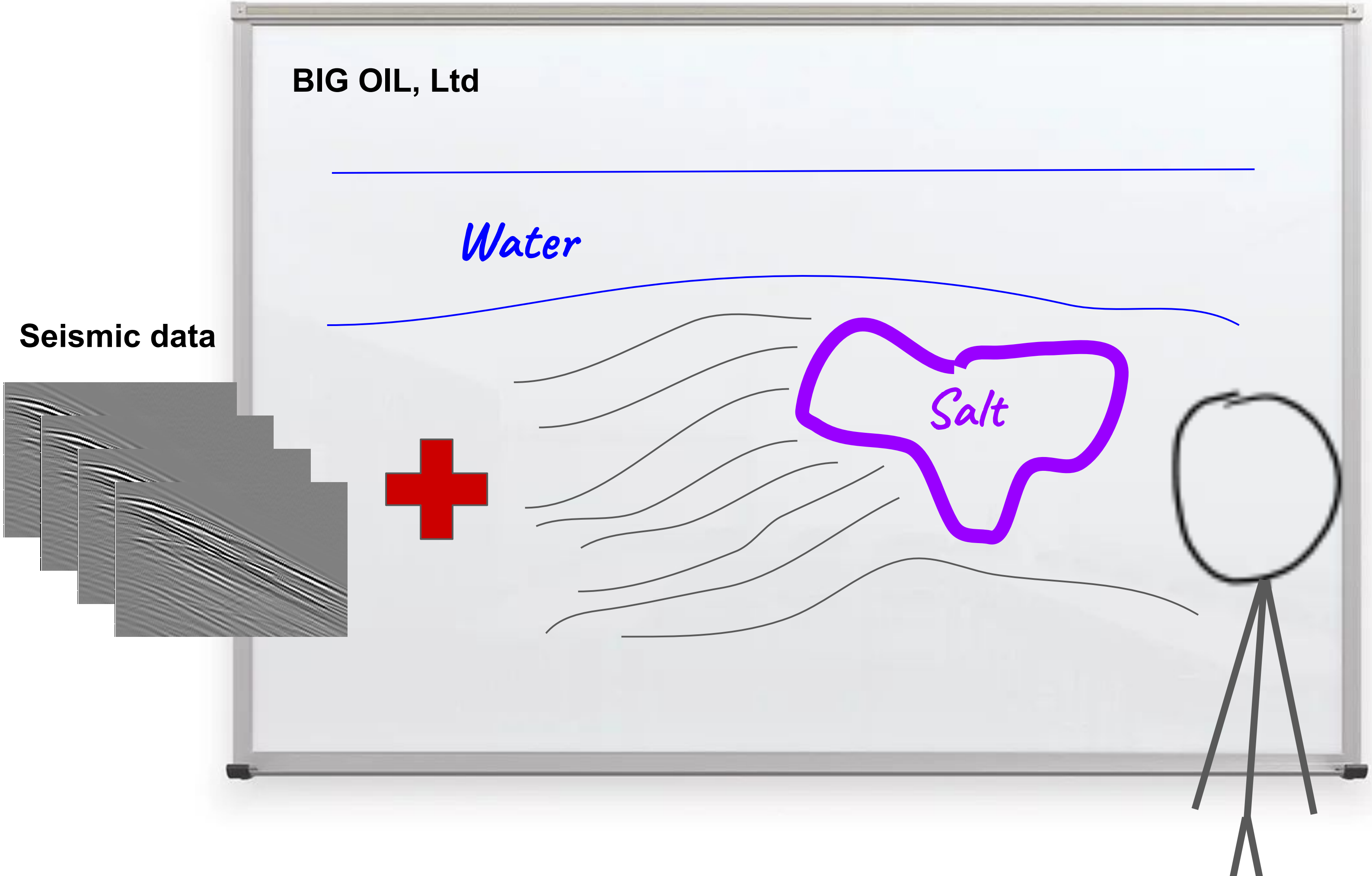
Thesis defense

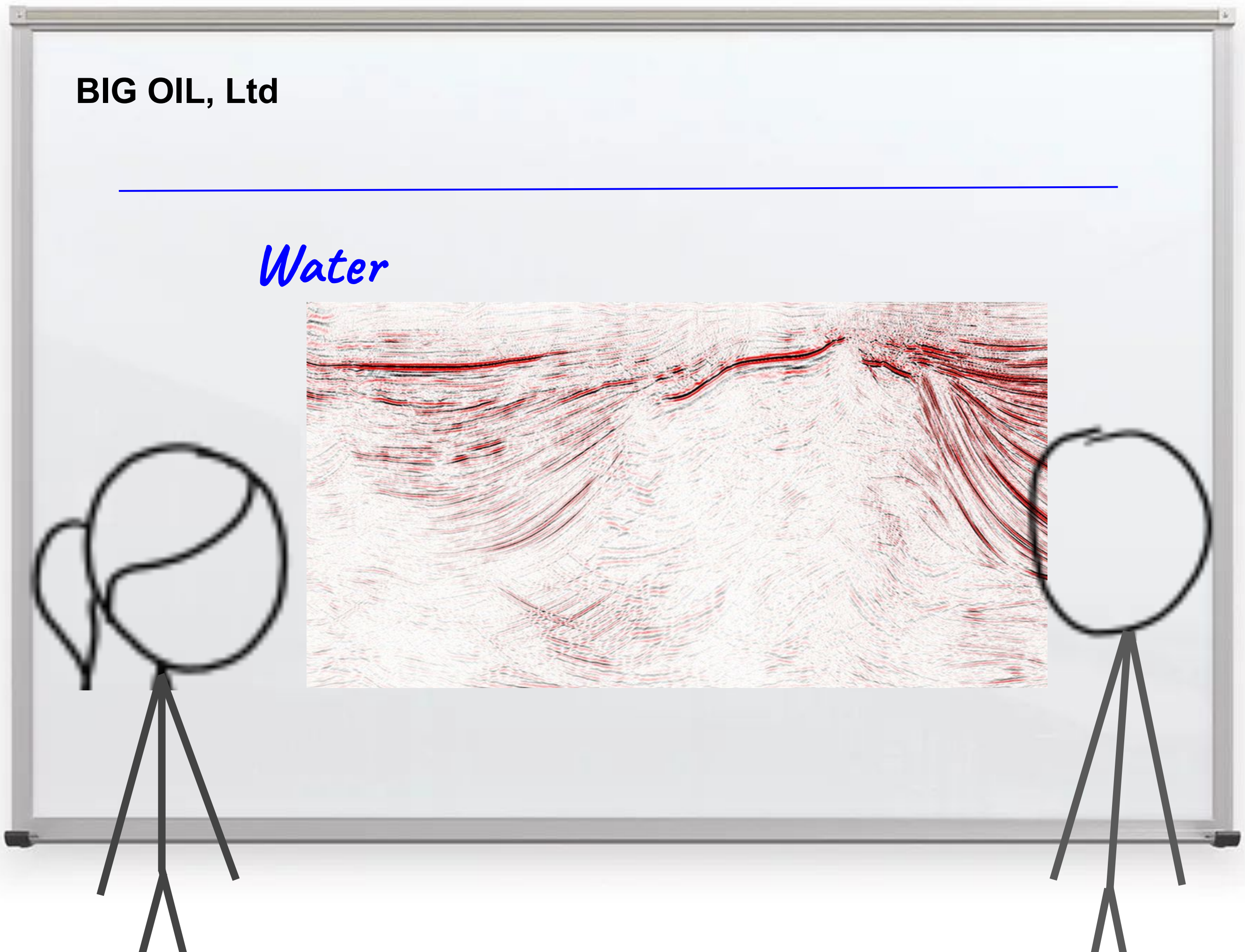
Taylor Dahlke, 4/1/2019

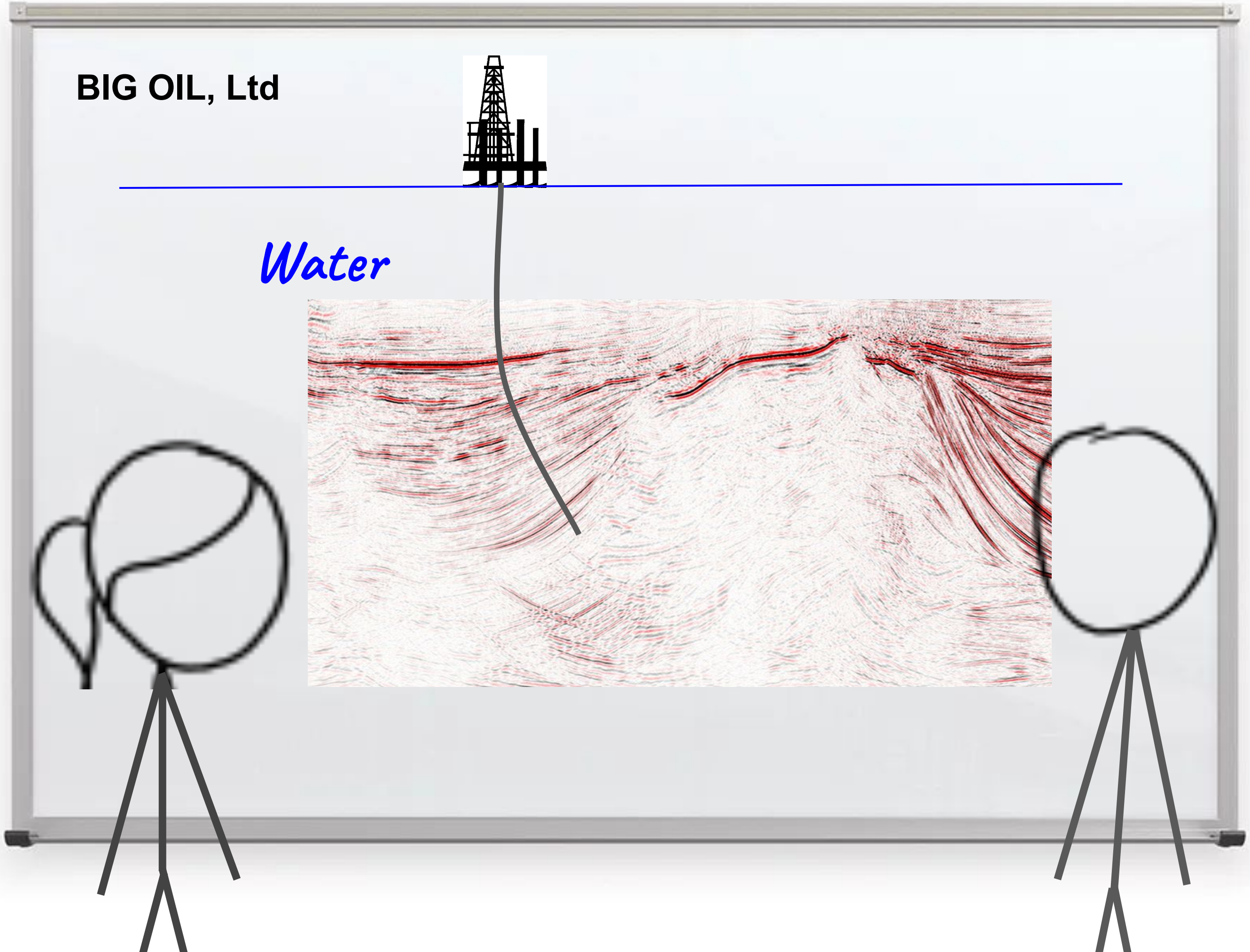

BIG OIL, Ltd

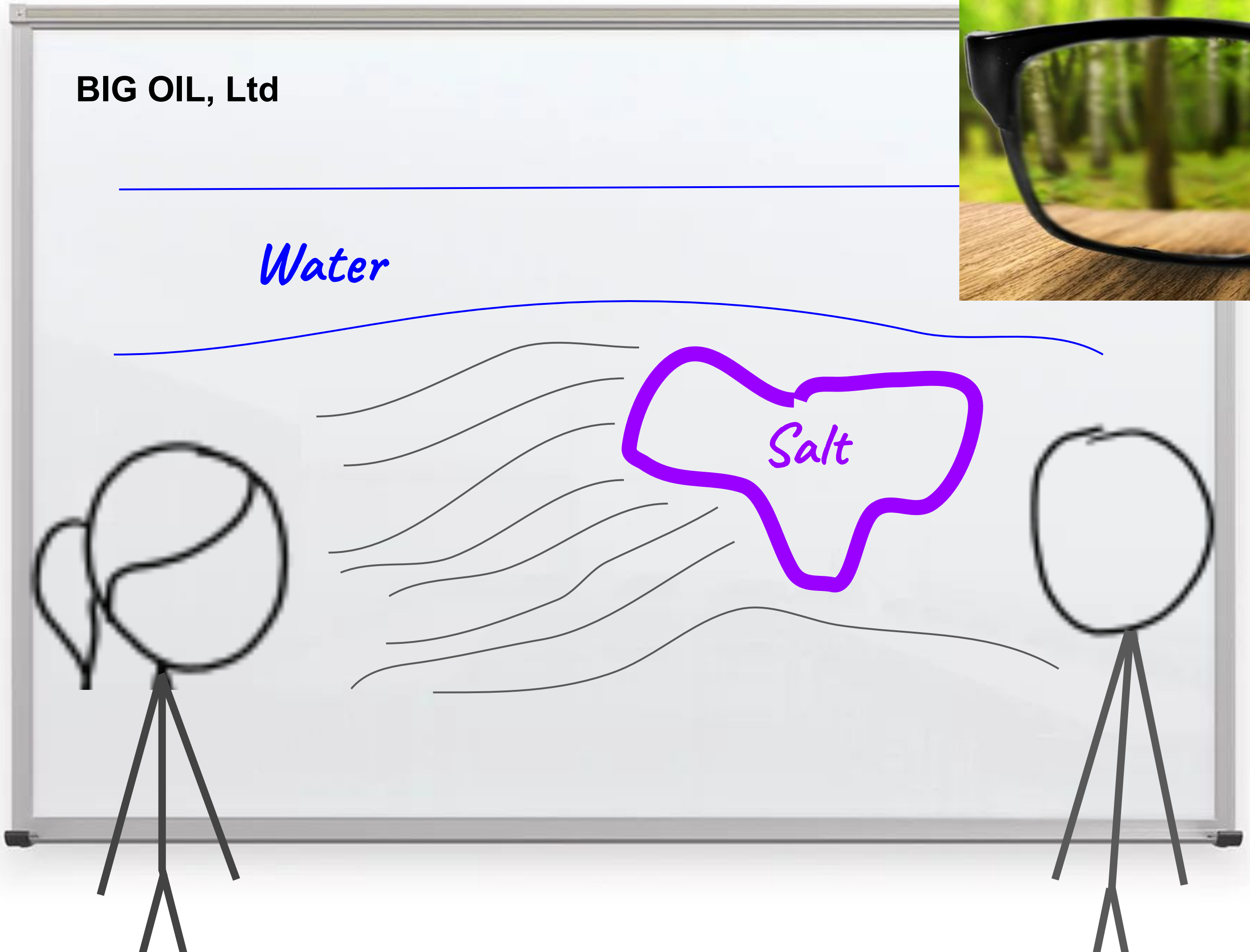

BIG OIL, Ltd
Water
Salt

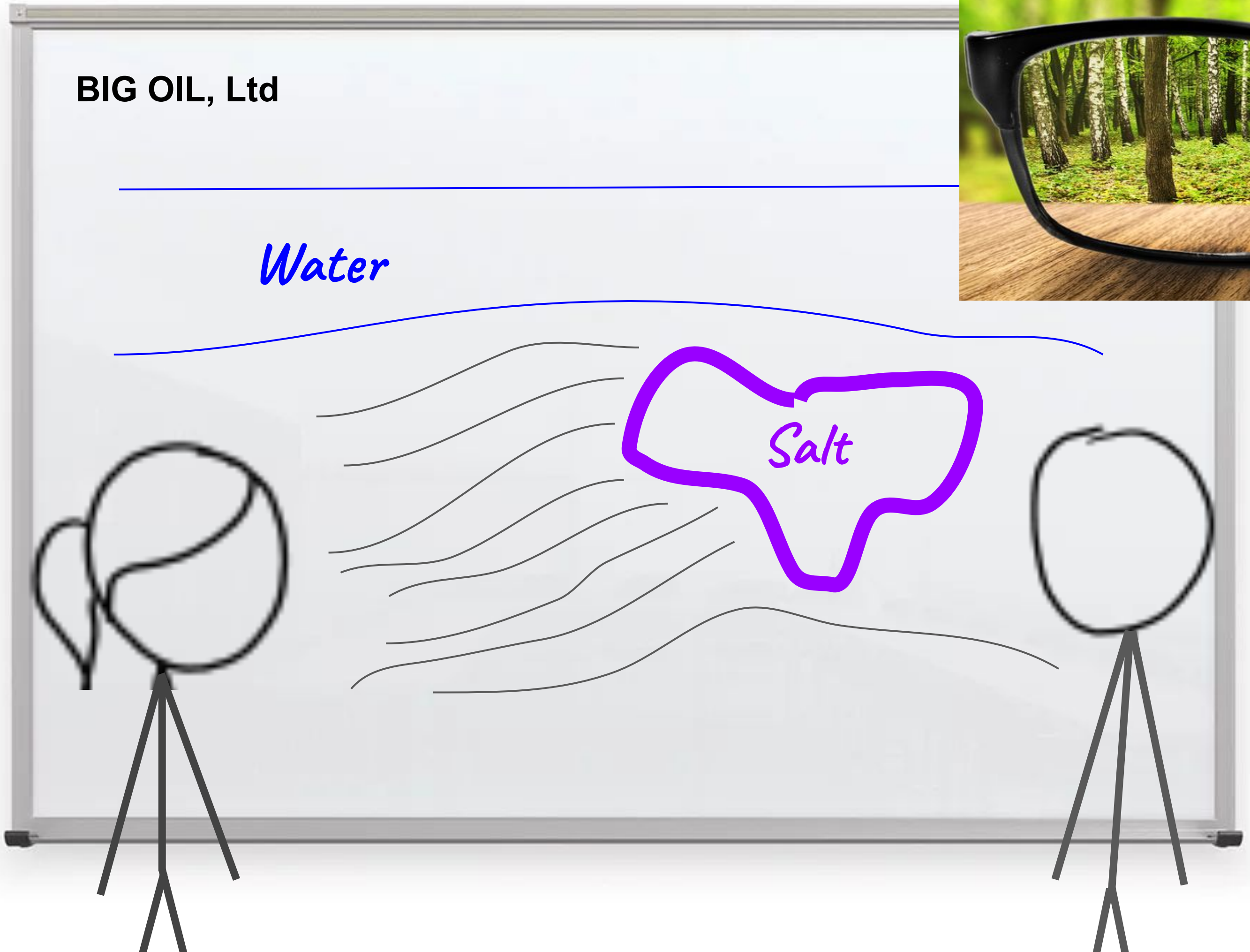

BIG OIL, Ltd
Water
Salt

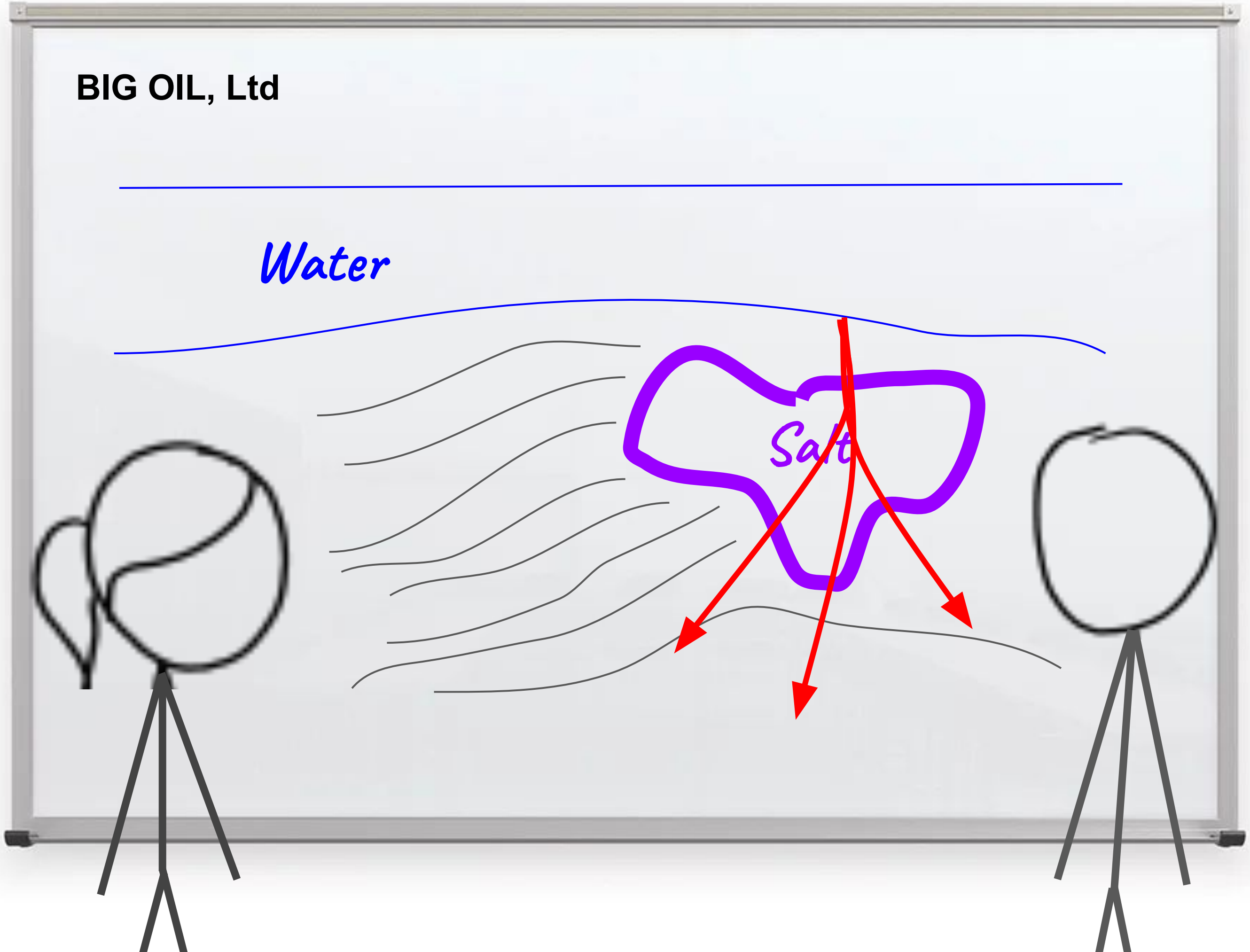

BIG OIL, Ltd
Water
Salt
Seismic data

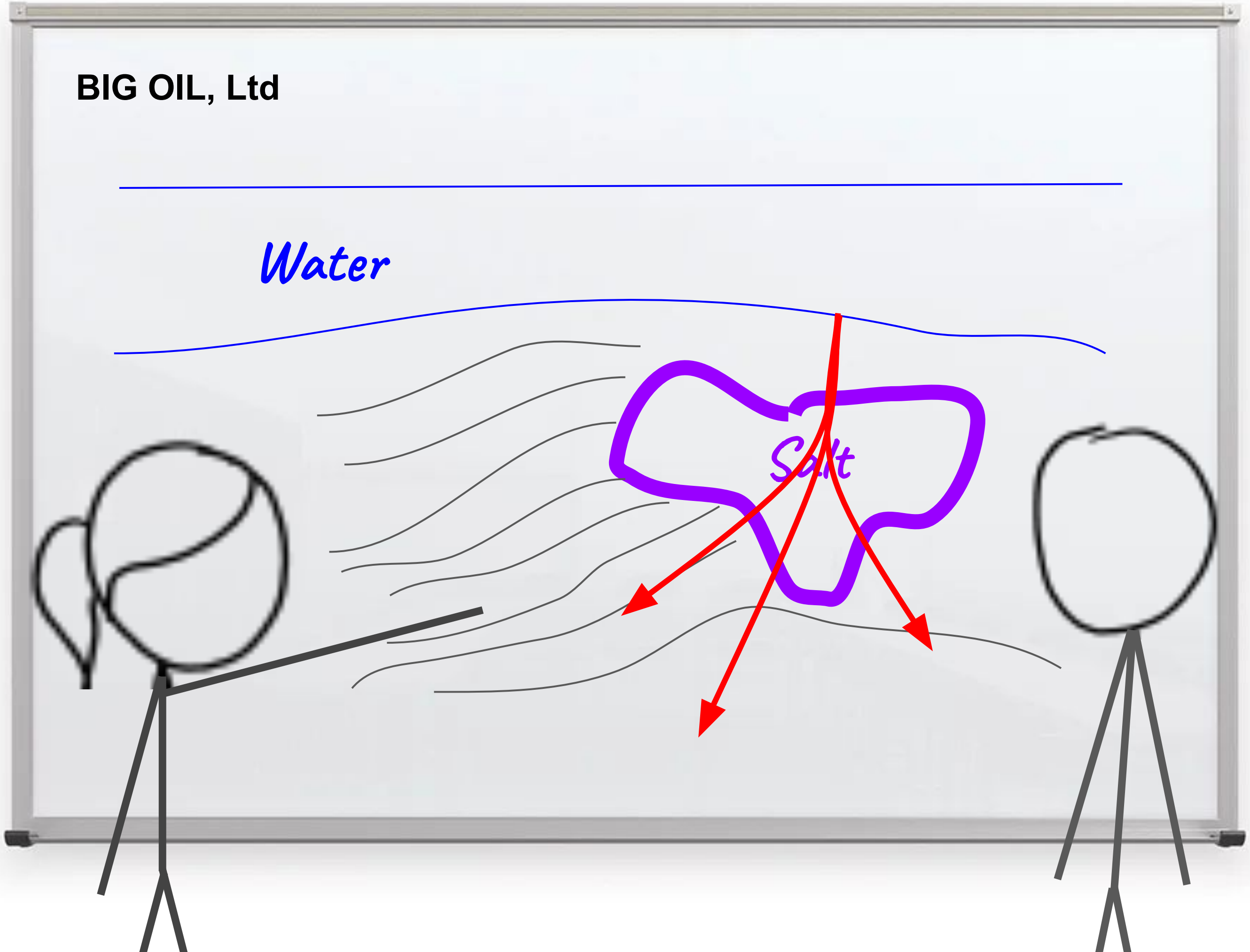

BIG OIL, Ltd
Water

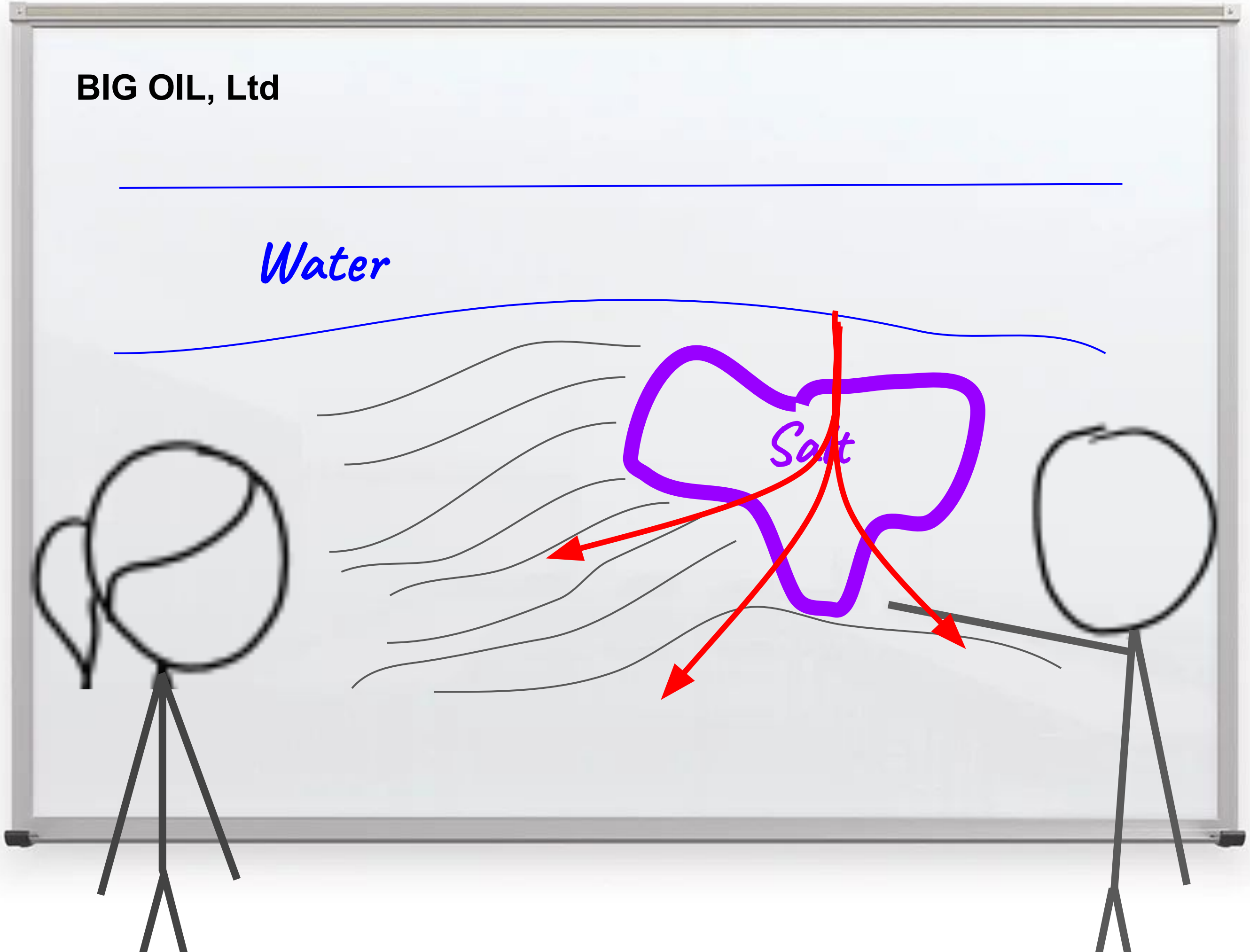

BIG OIL, Ltd
Water

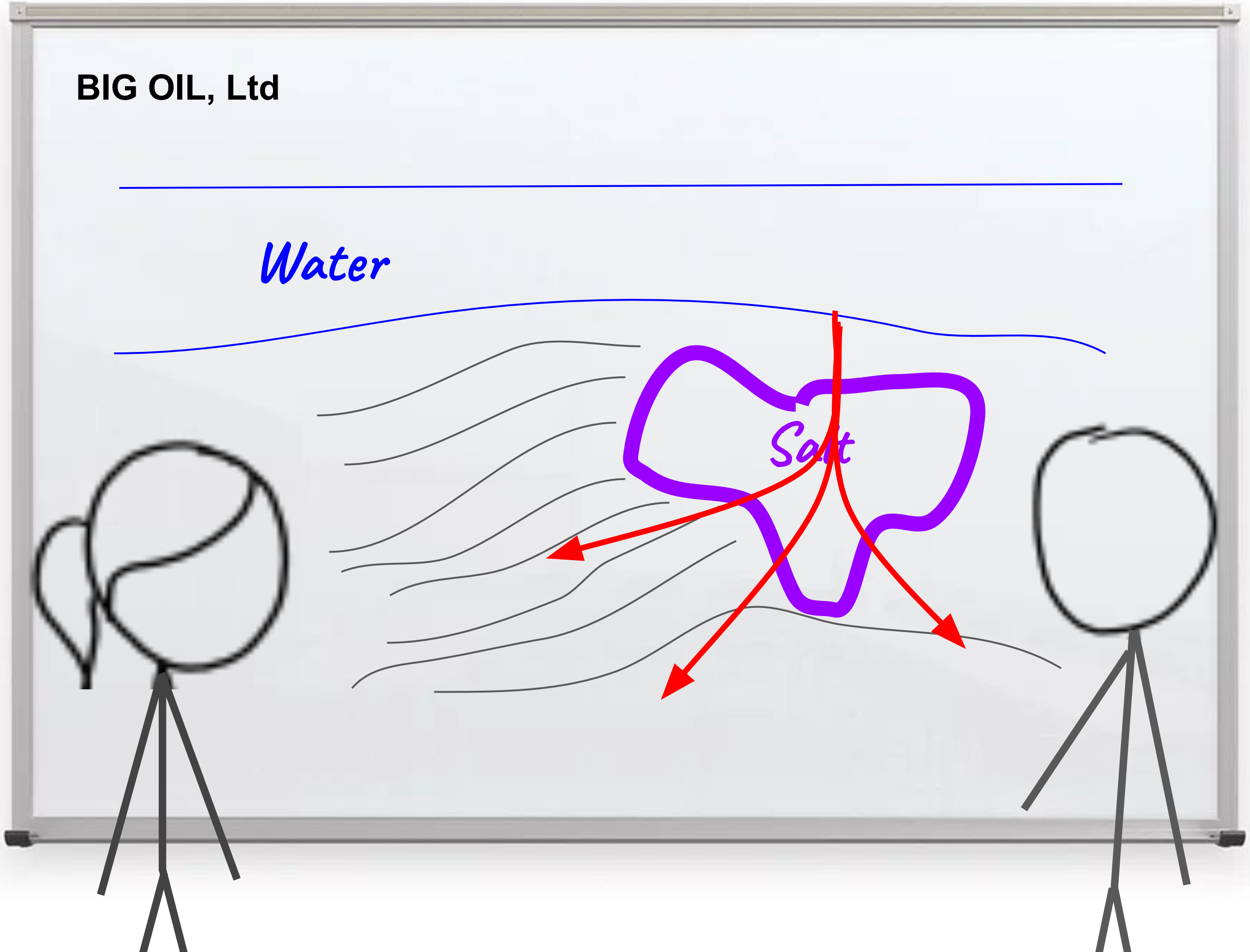

BIG OIL, Ltd
Water
Salt

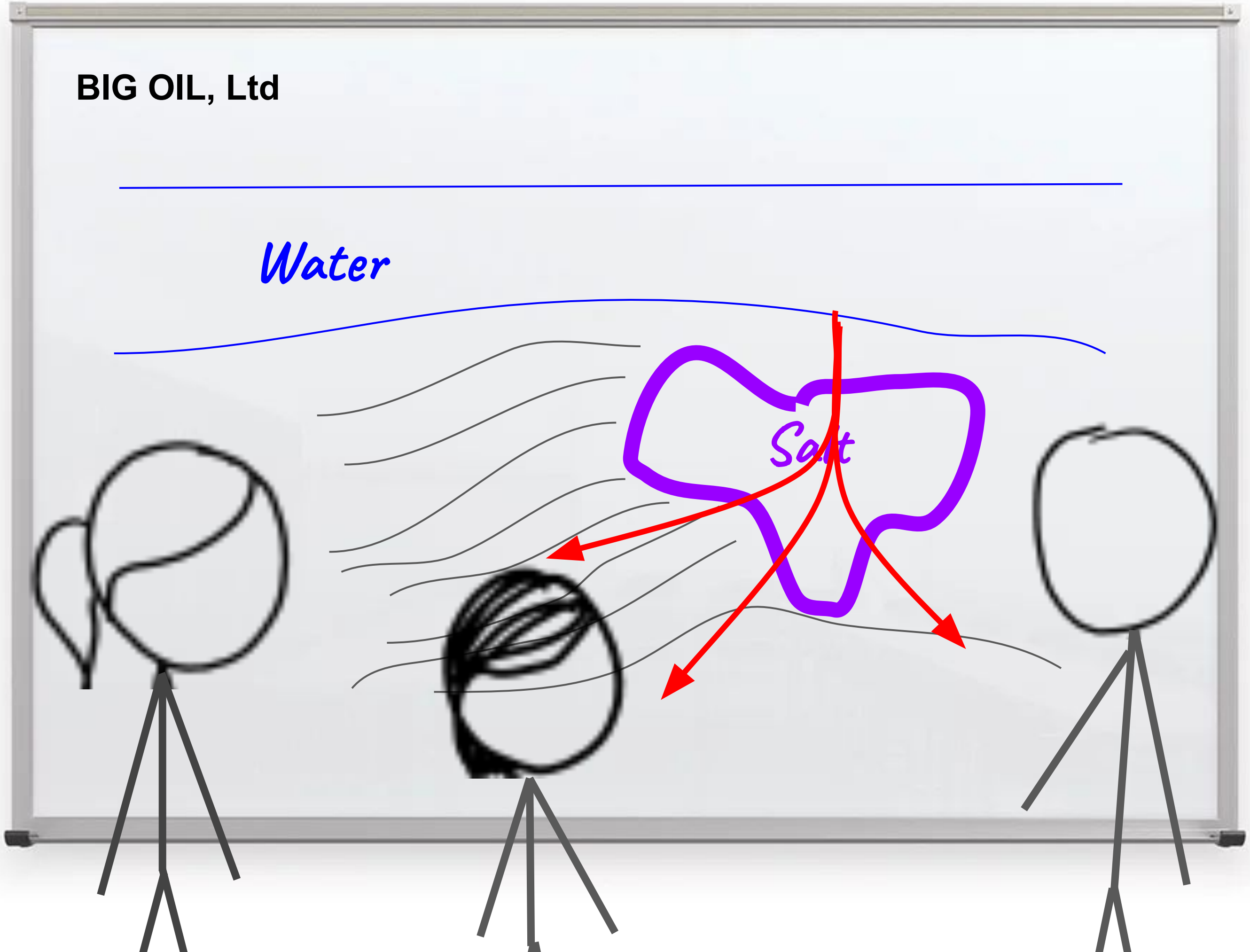

BIG OIL, Ltd
Water
Salt

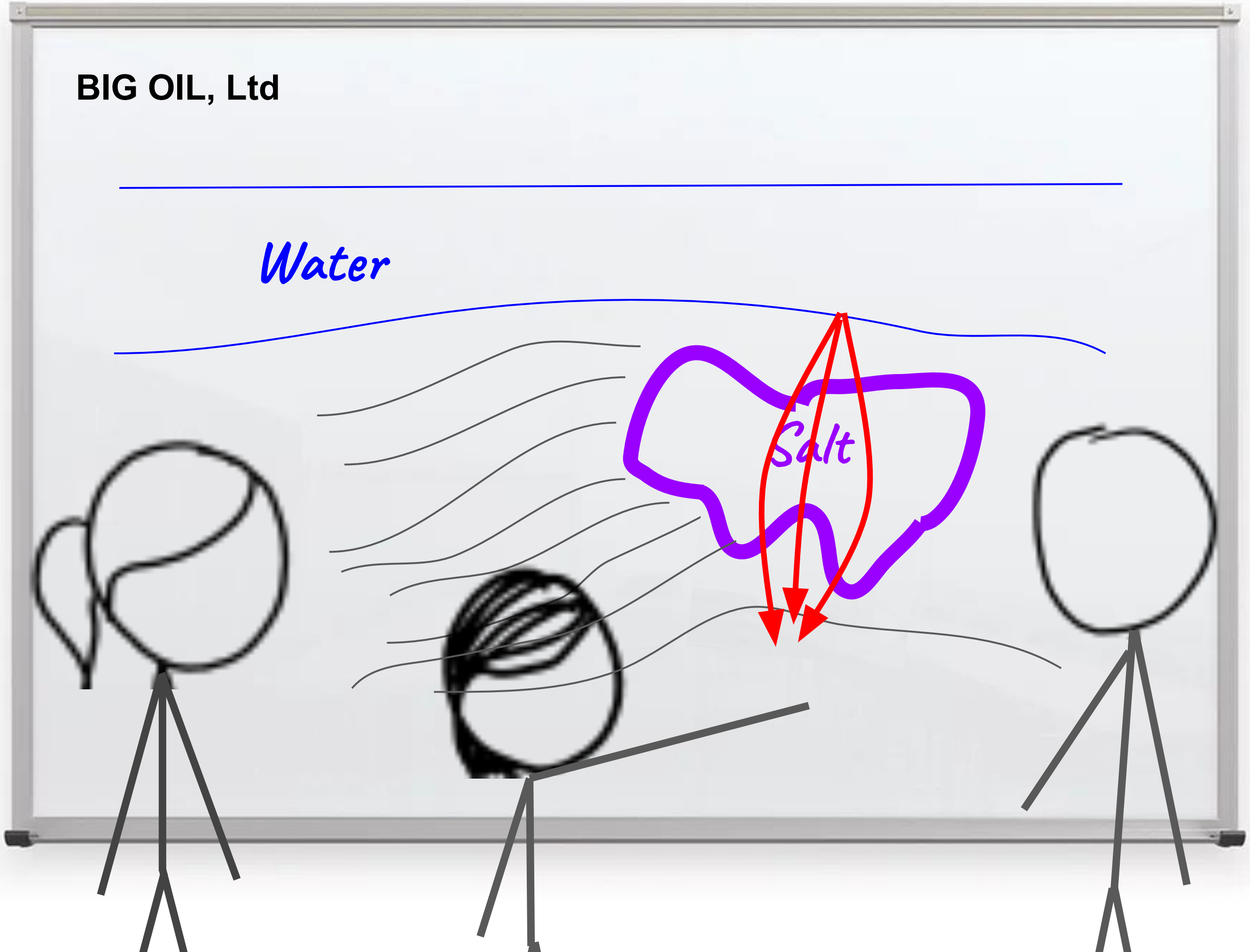

BIG OIL, Ltd
Water
Salt


BIG OIL, Ltd
Water
Salt

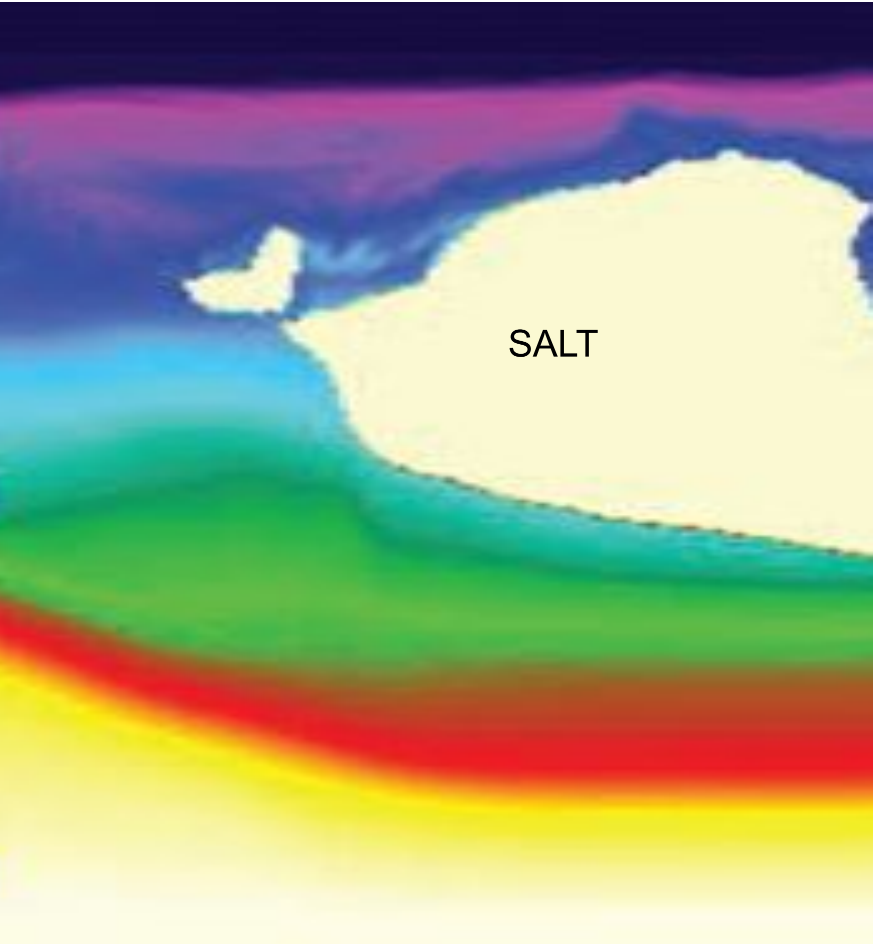

BIG OIL, Ltd
Water
Salt


BIG OIL, Ltd
Water
Salt

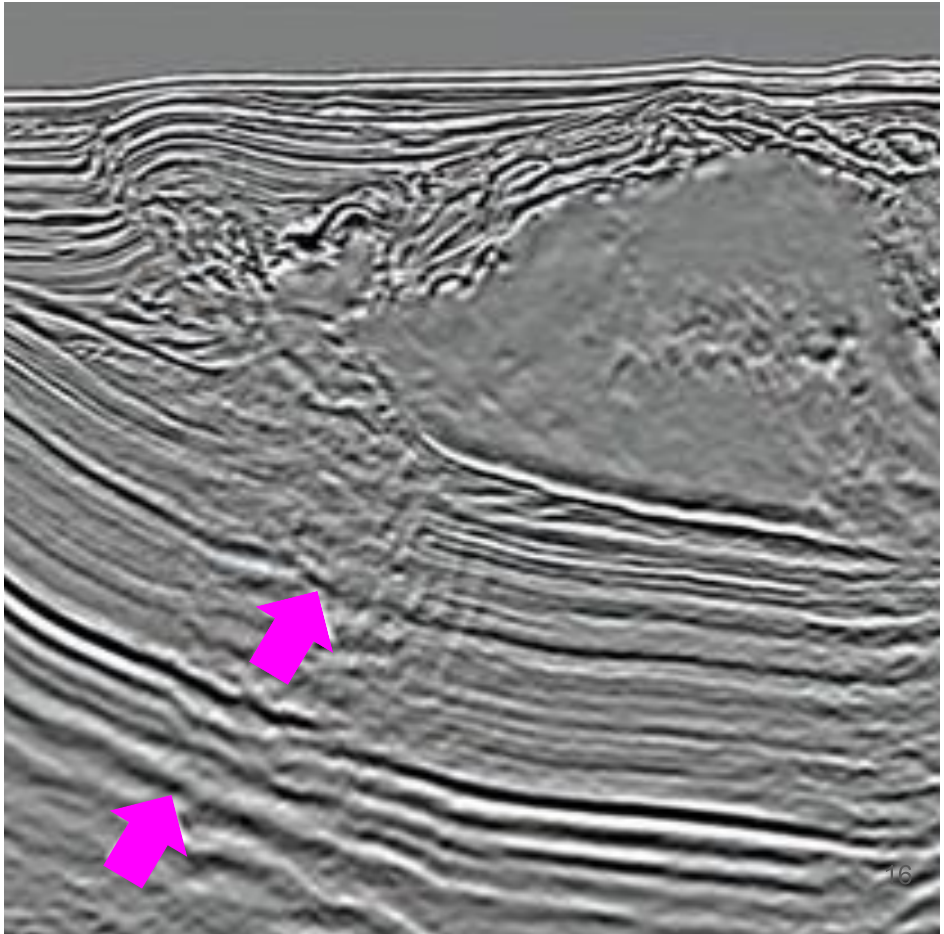

BIG OIL, Ltd
Water
Salt


BIG OIL, Ltd
Water
Salt

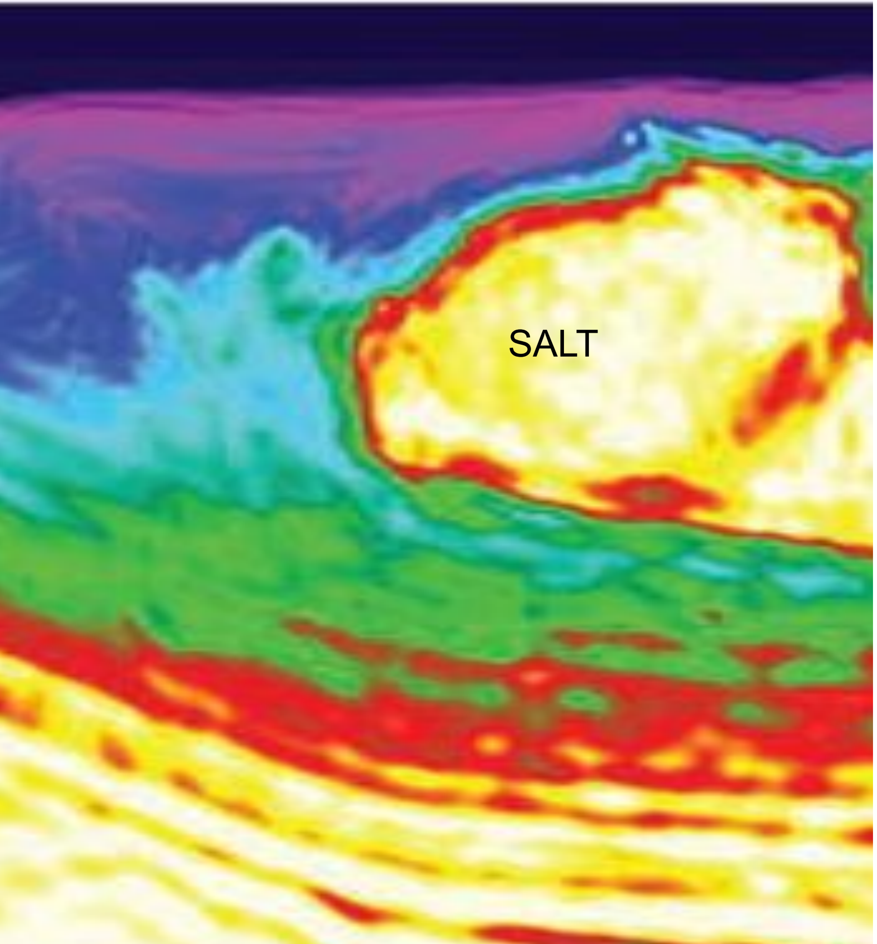
Initial model



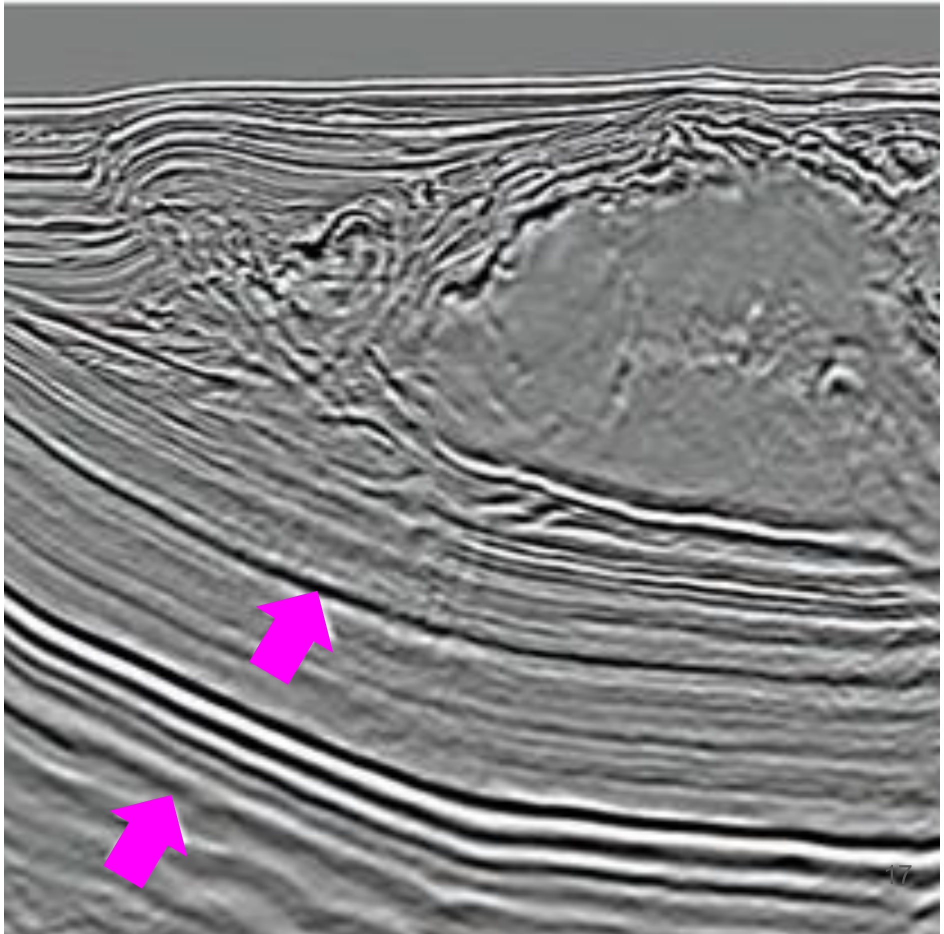
Initial seismic image



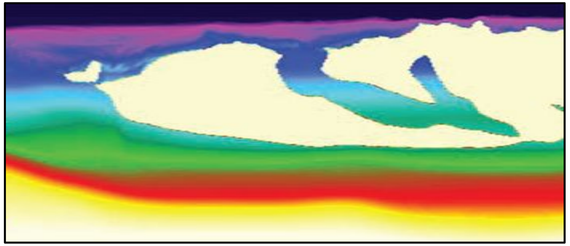
Refined model



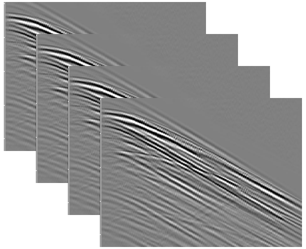
Refined seismic image



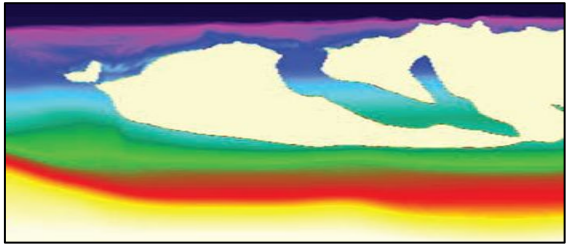
Velocity model



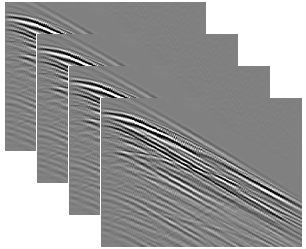
Seismic data



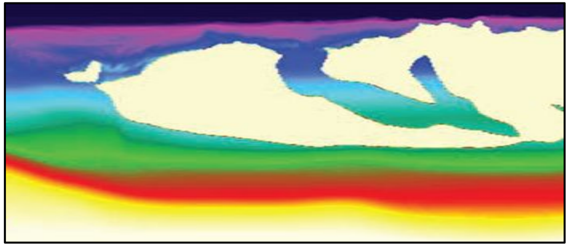
Velocity model

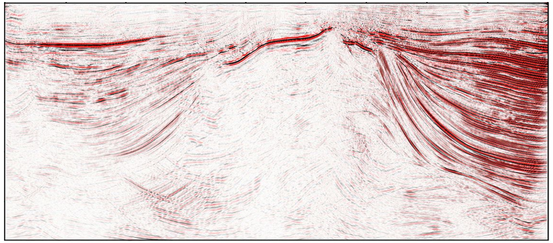


Seismic data

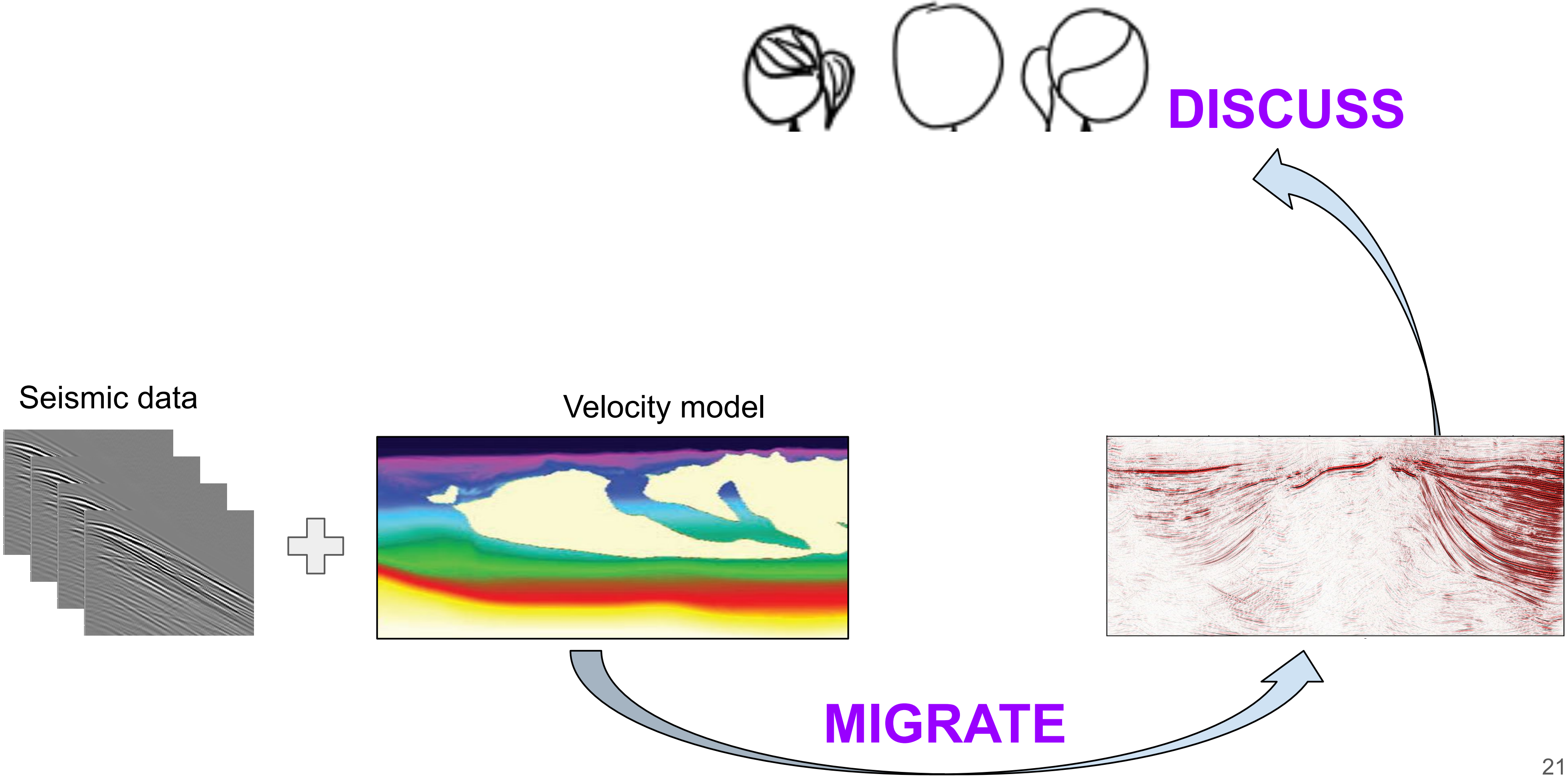


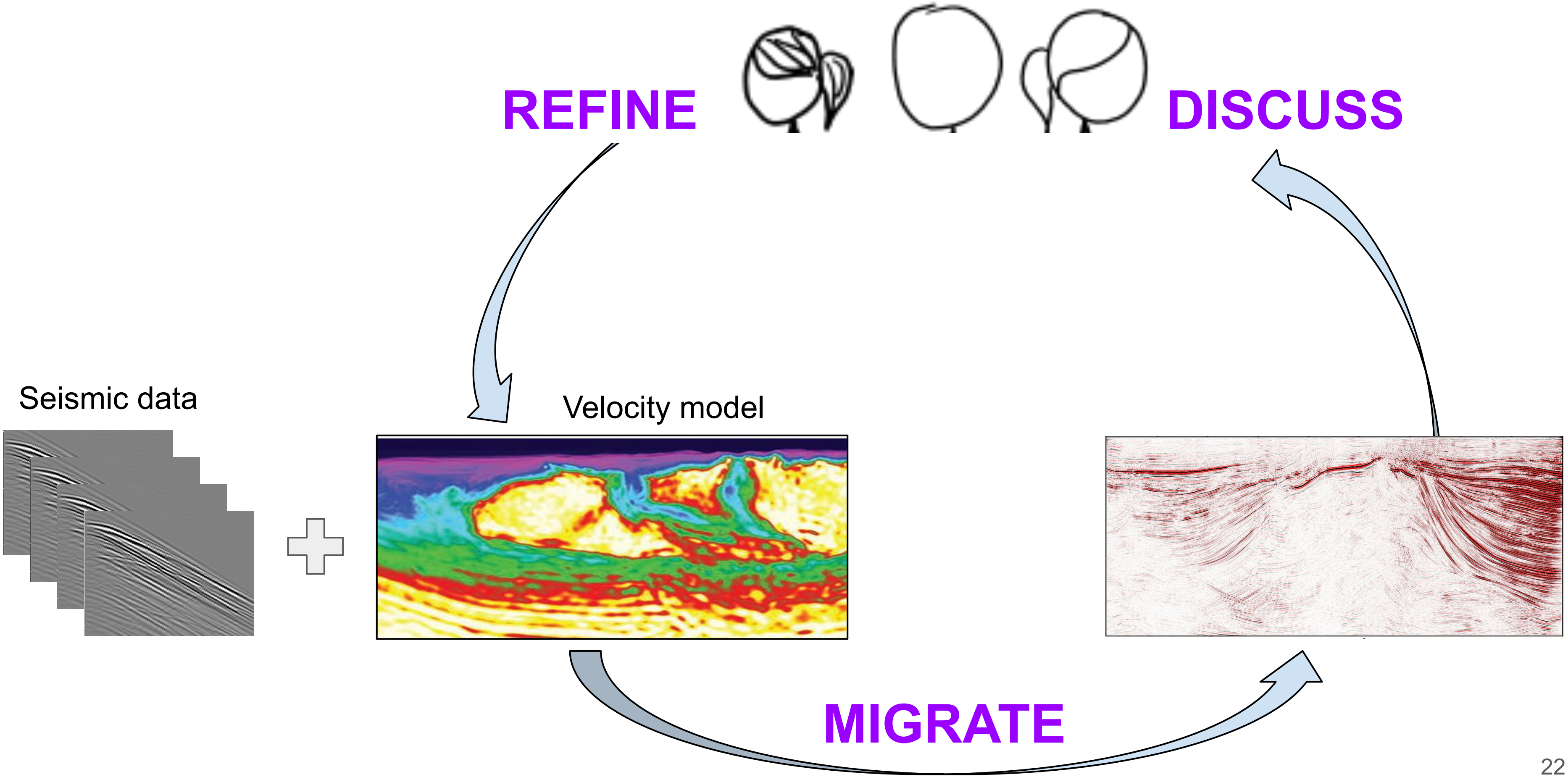
Velocity model

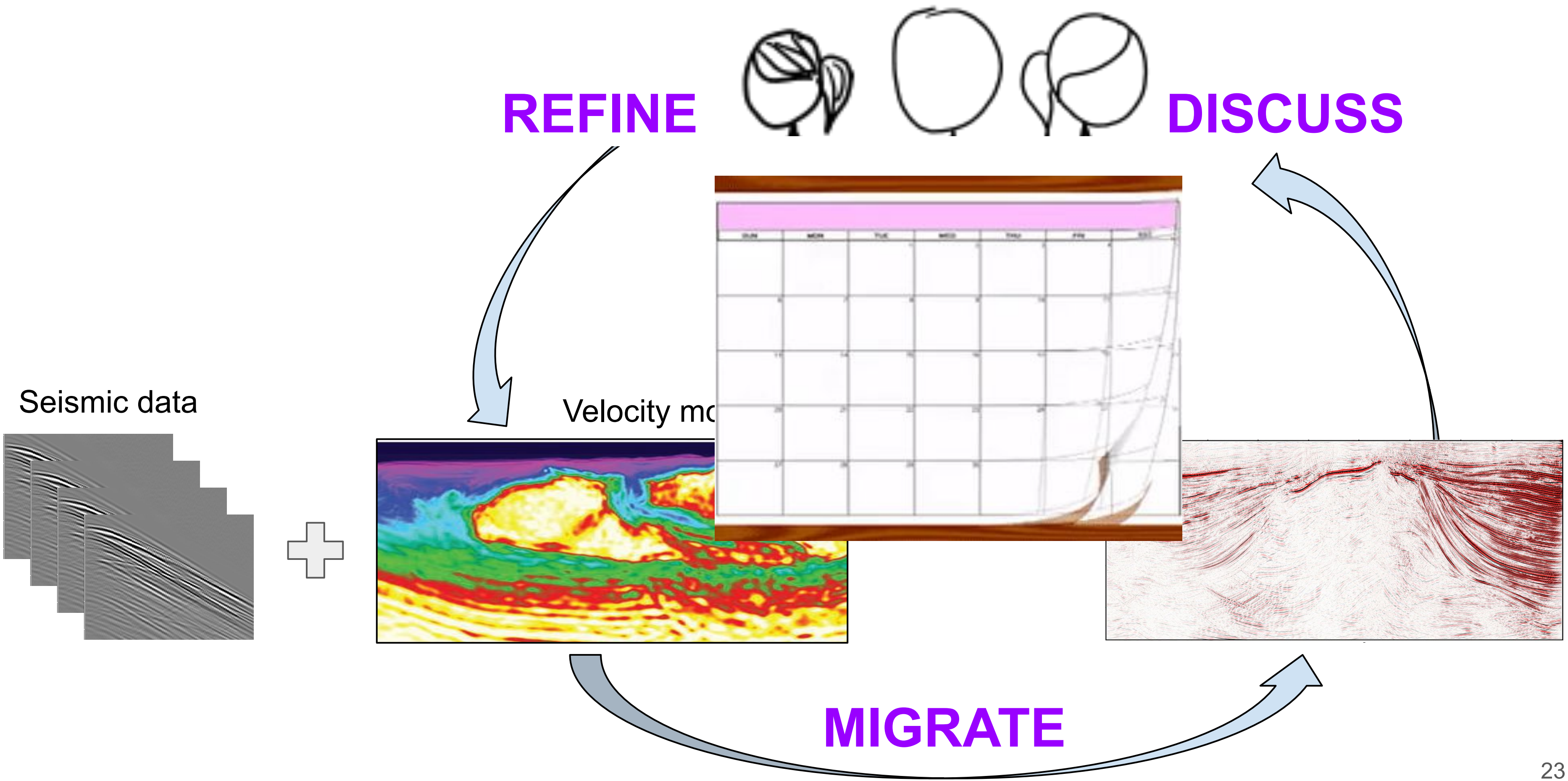




MIGRATE


DISCUSS
Seismic data
Velocity model
MIGRATE


REFINE
DISCUSS
Seismic data
Velocity model
MIGRATE



REFINE
DISCUSS
Seismic data
Velocity mo
MIGRATE

Before Full-Waveform Inversion (FWI)

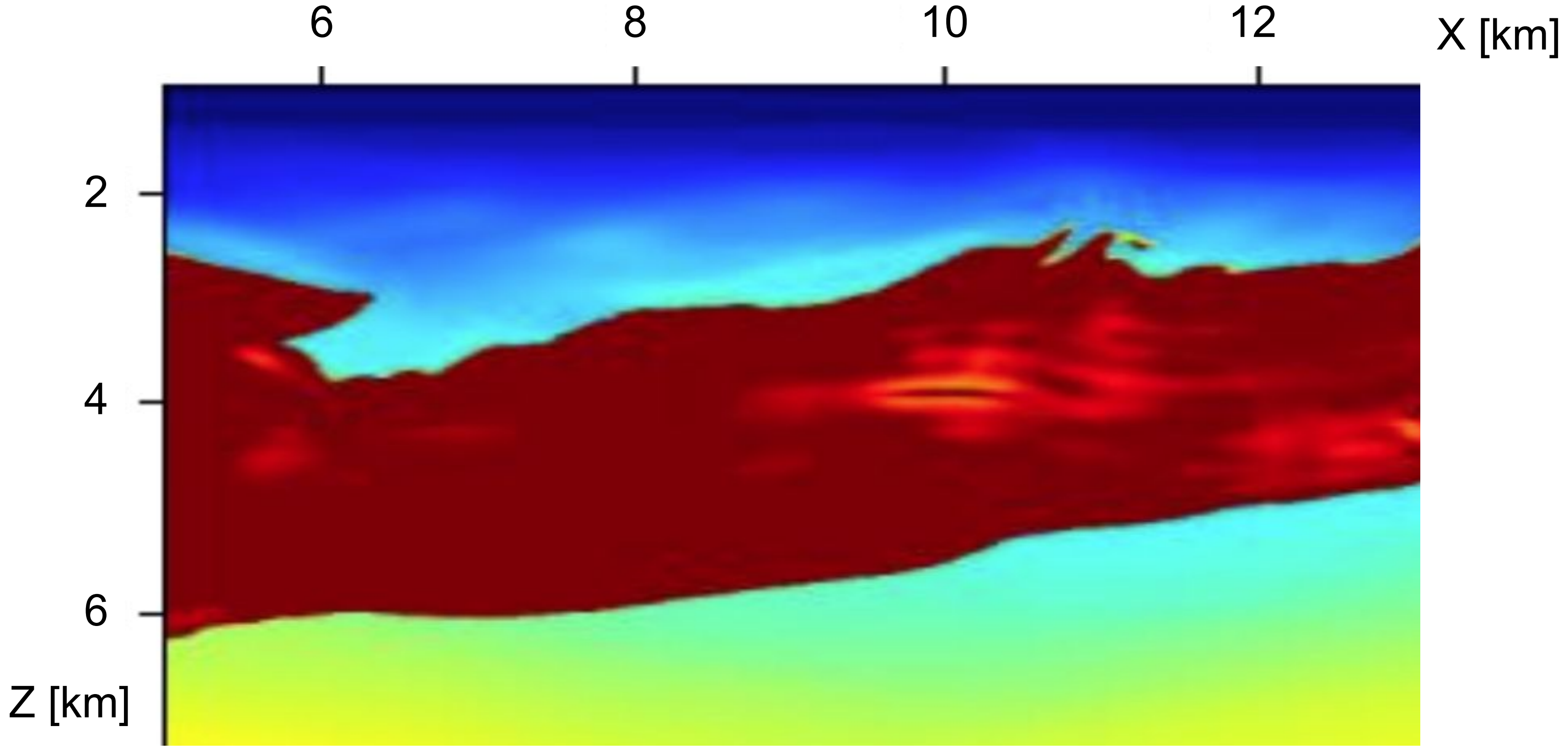


Figure: Xukai Shen, “Salt model building at Atlantis with Full Waveform Inversion“, SEG 2017

After Full-Waveform Inversion (FWI)

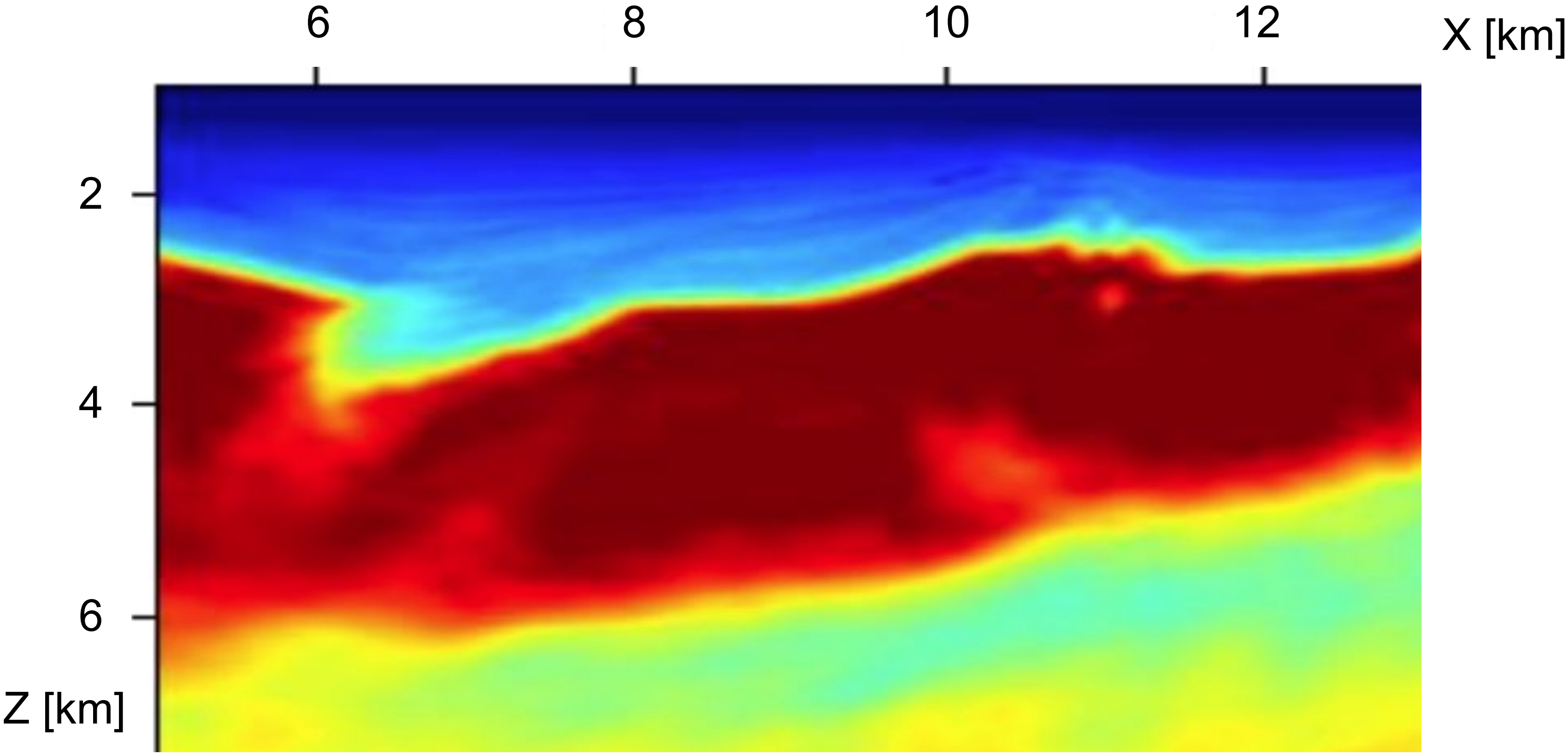
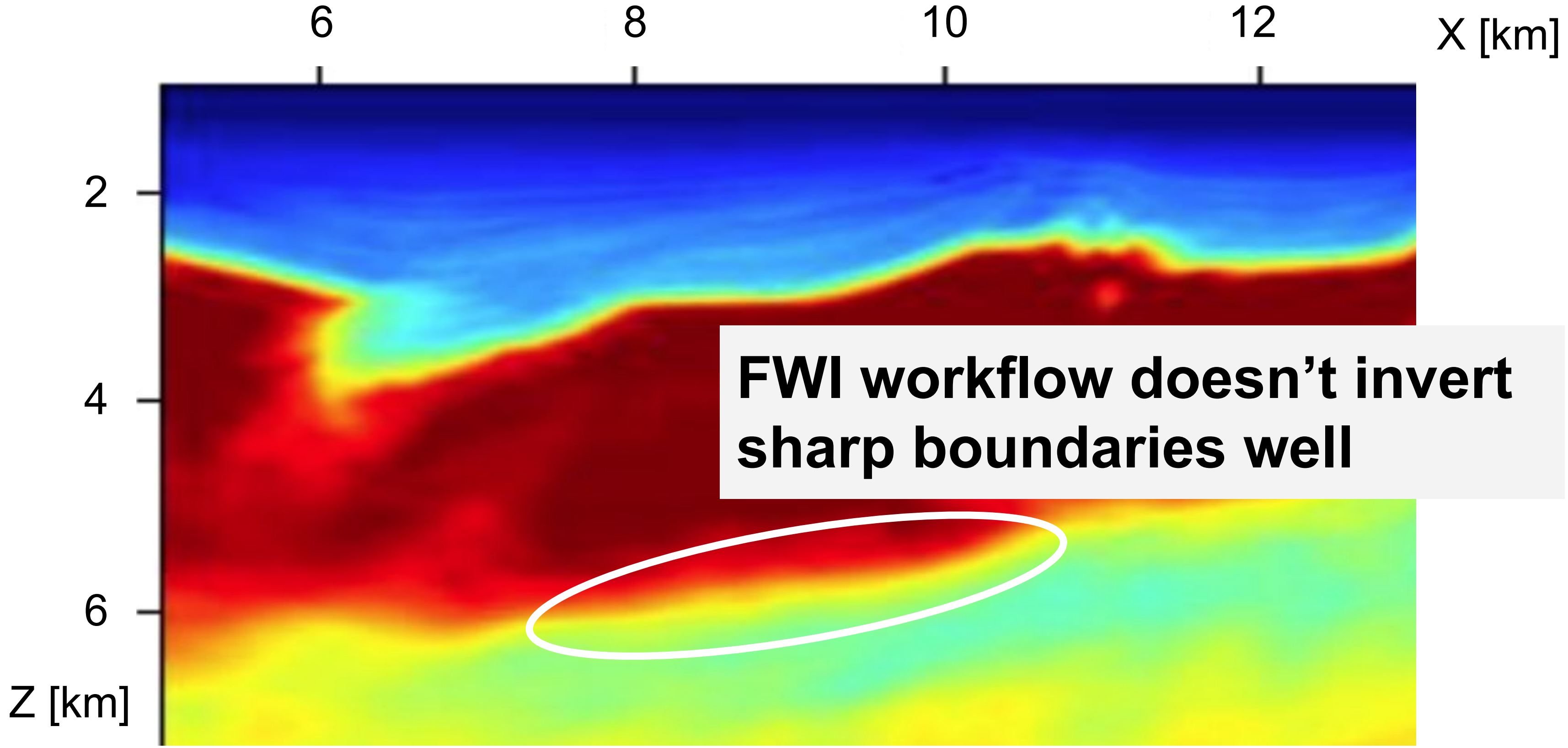


Figure: Xukai Shen, “Salt model building at Atlantis with Full Waveform Inversion“, SEG 2017

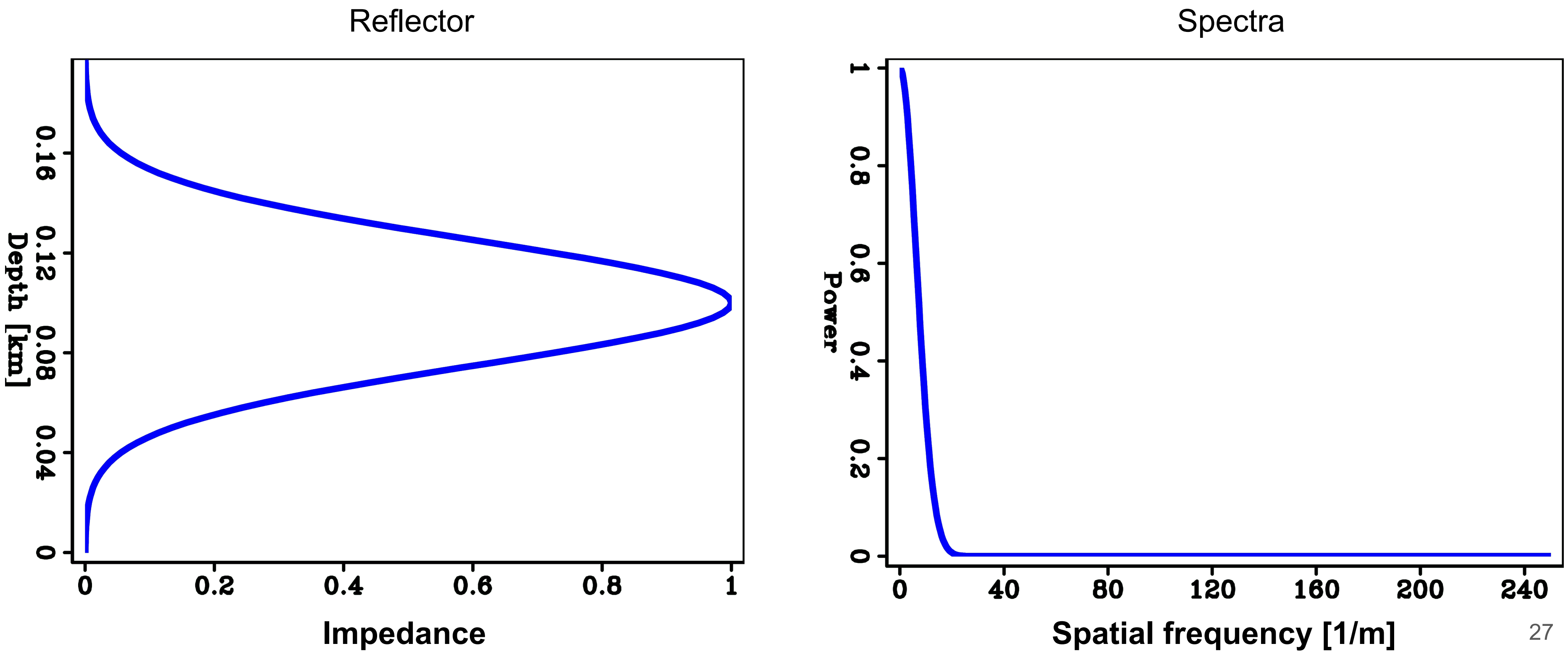
After Full-Waveform Inversion (FWI)



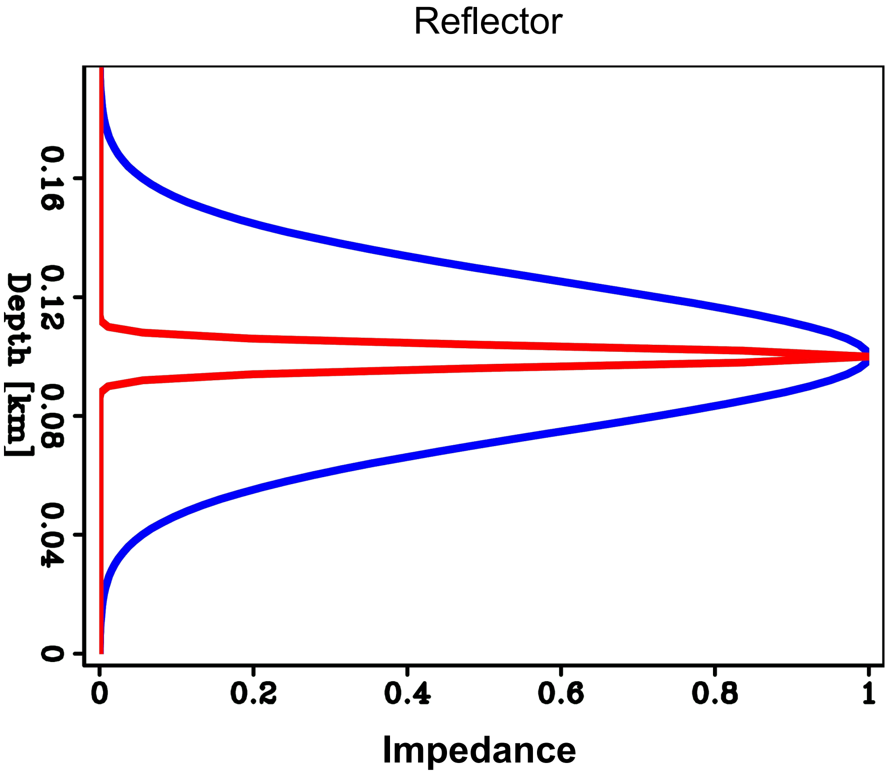
FWI workflow doesn’t invert sharp boundaries well

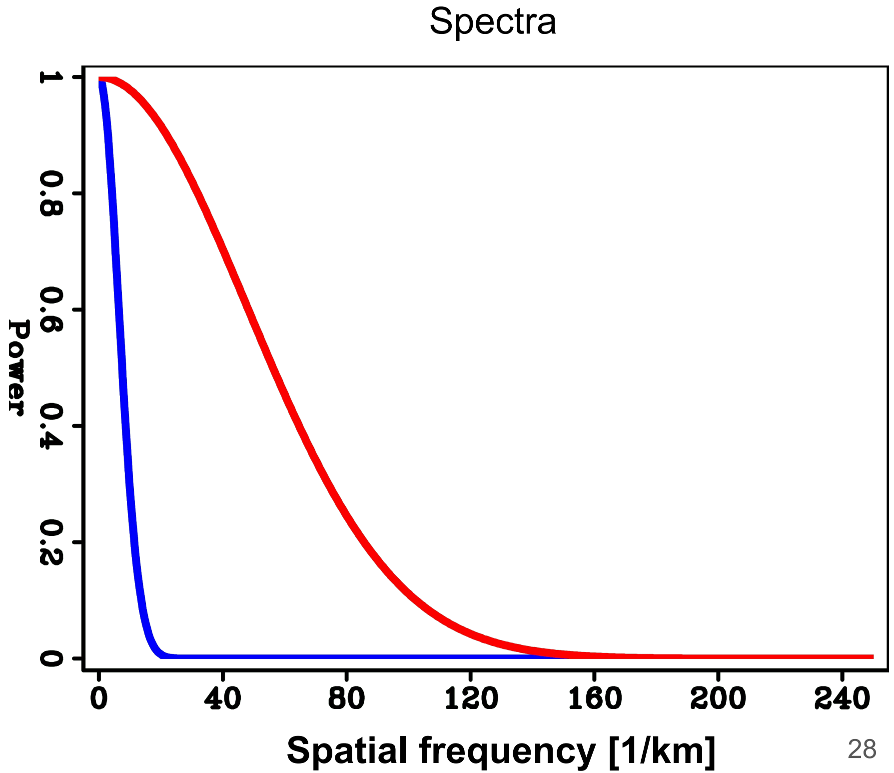
Figure: Xukai Shen, “Salt model building at Atlantis with Full Waveform Inversion“, SEG 2017

Sharp interfaces require more high frequencies

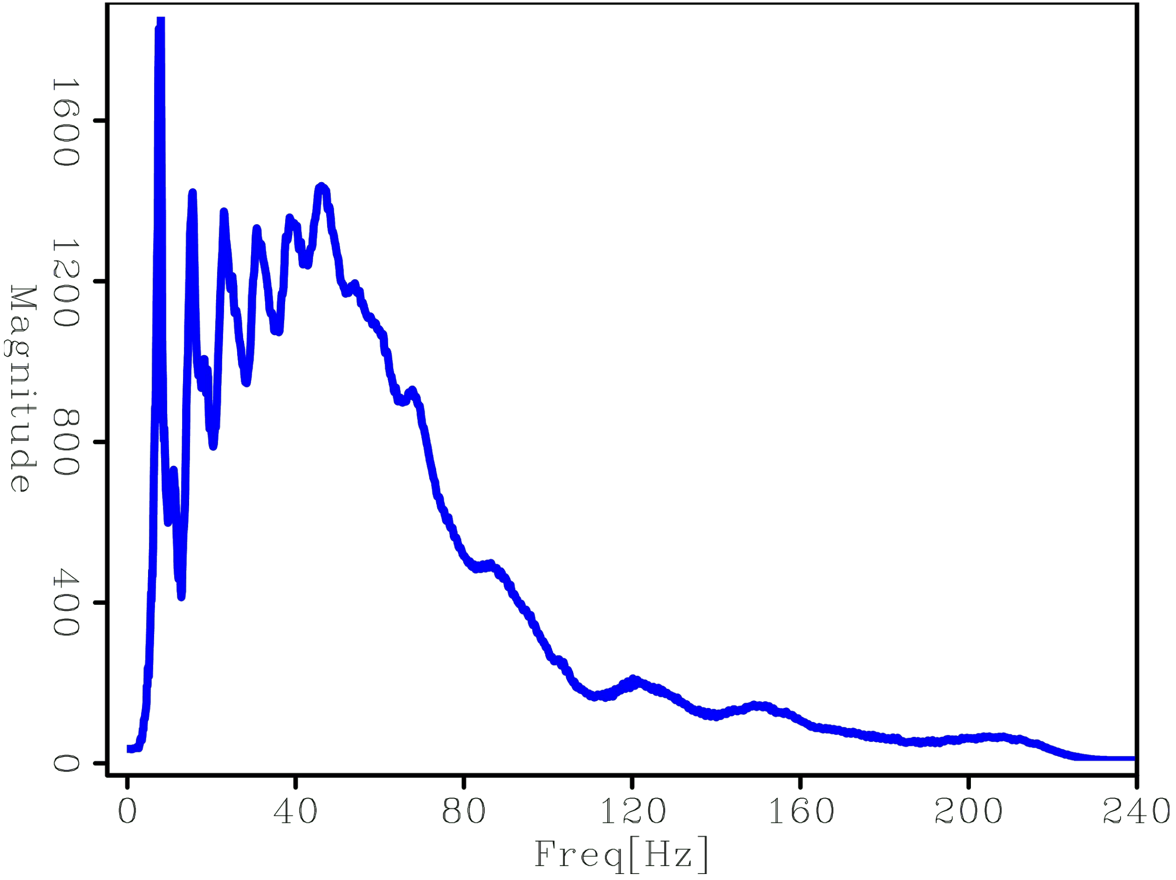


Sharp interfaces require **more** high frequencies

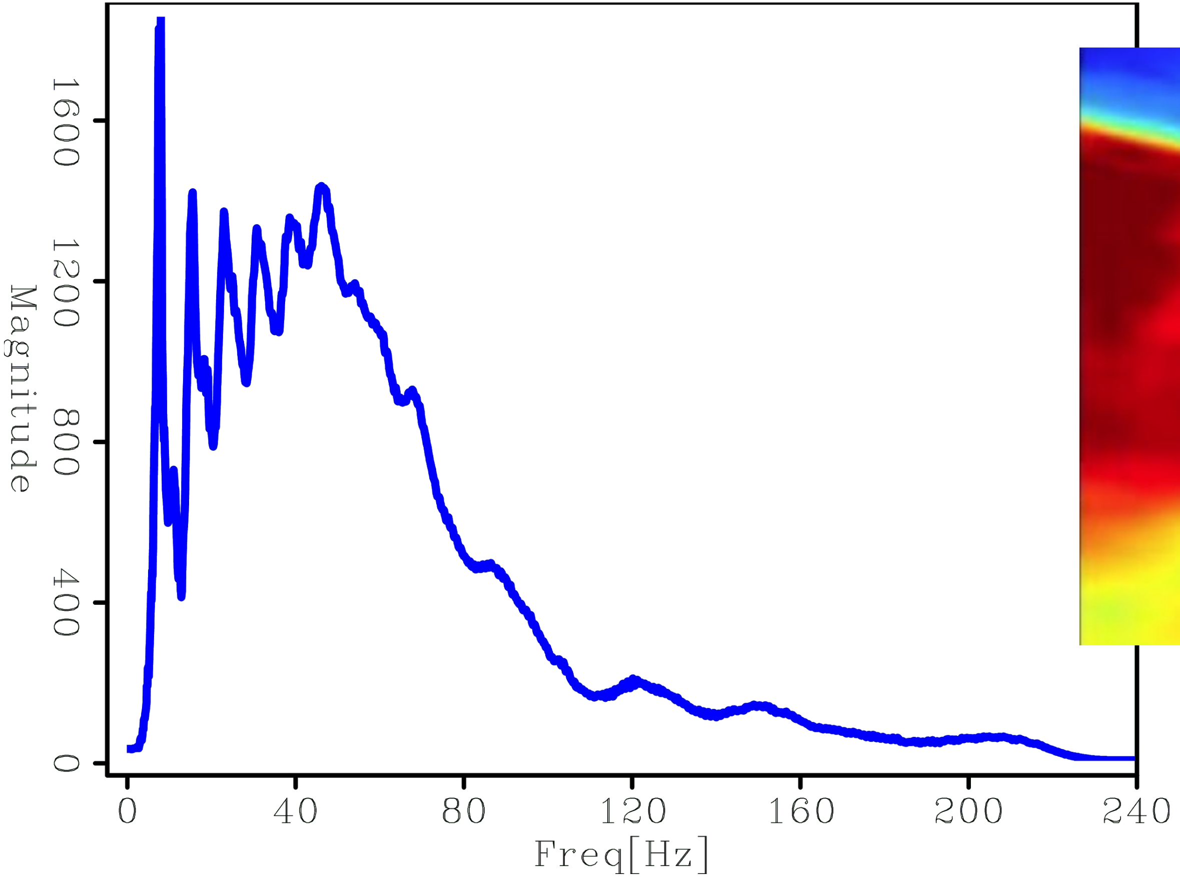


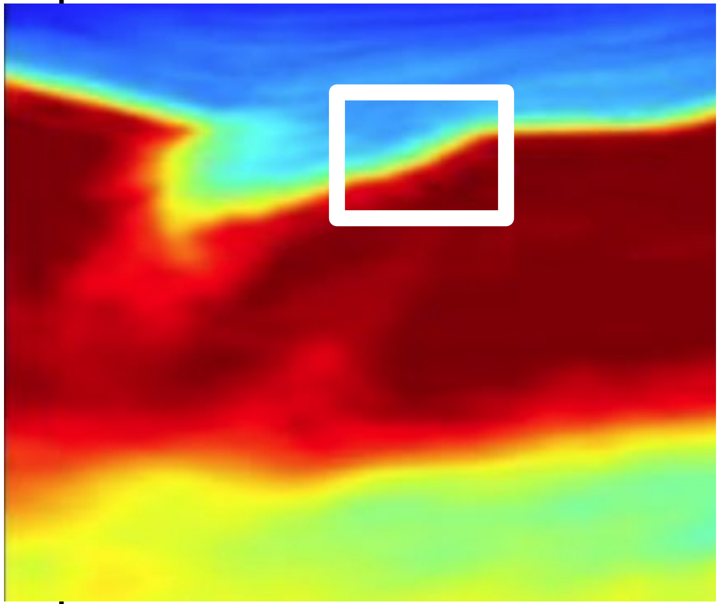


Field data has limited bandwidth

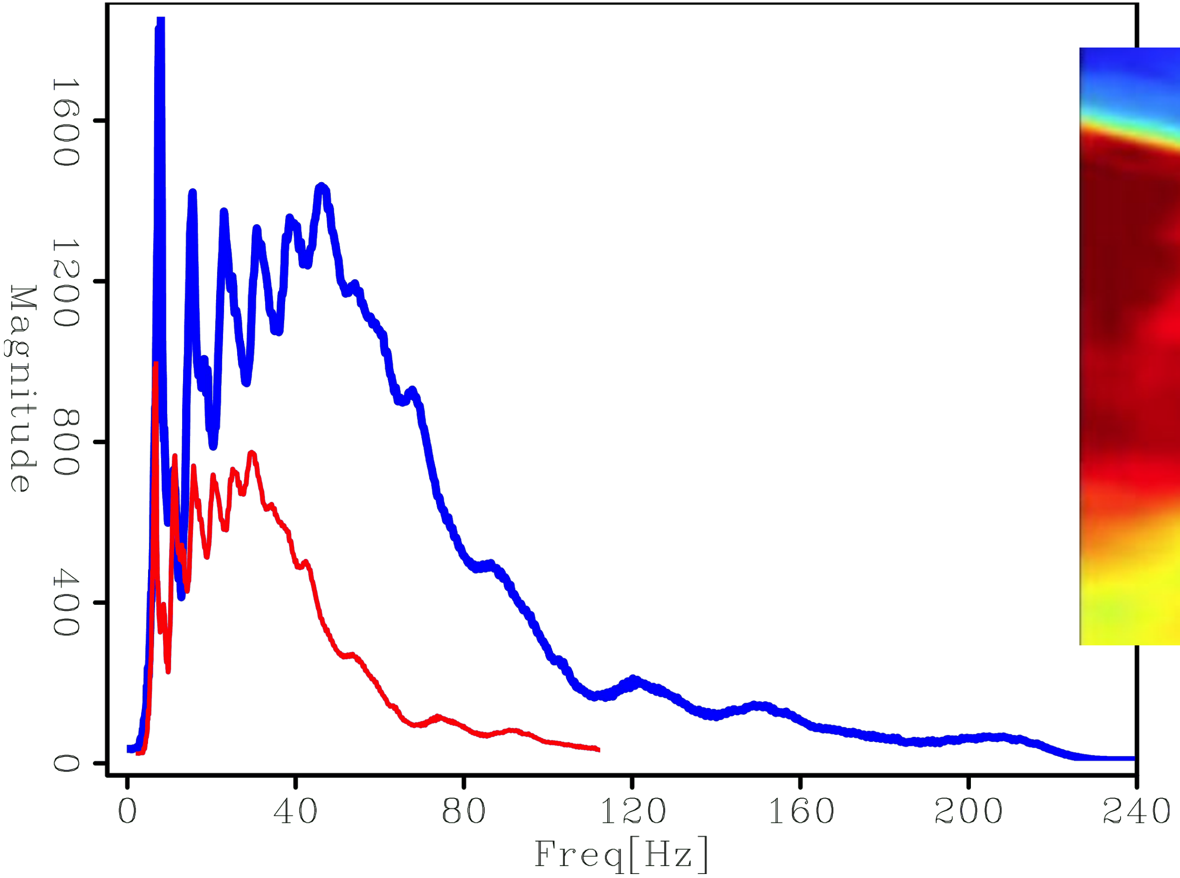


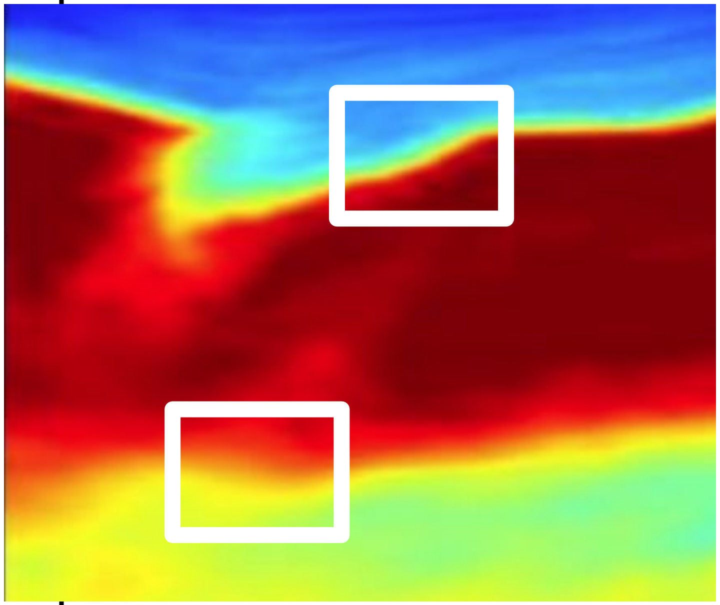
Field data has limited bandwidth



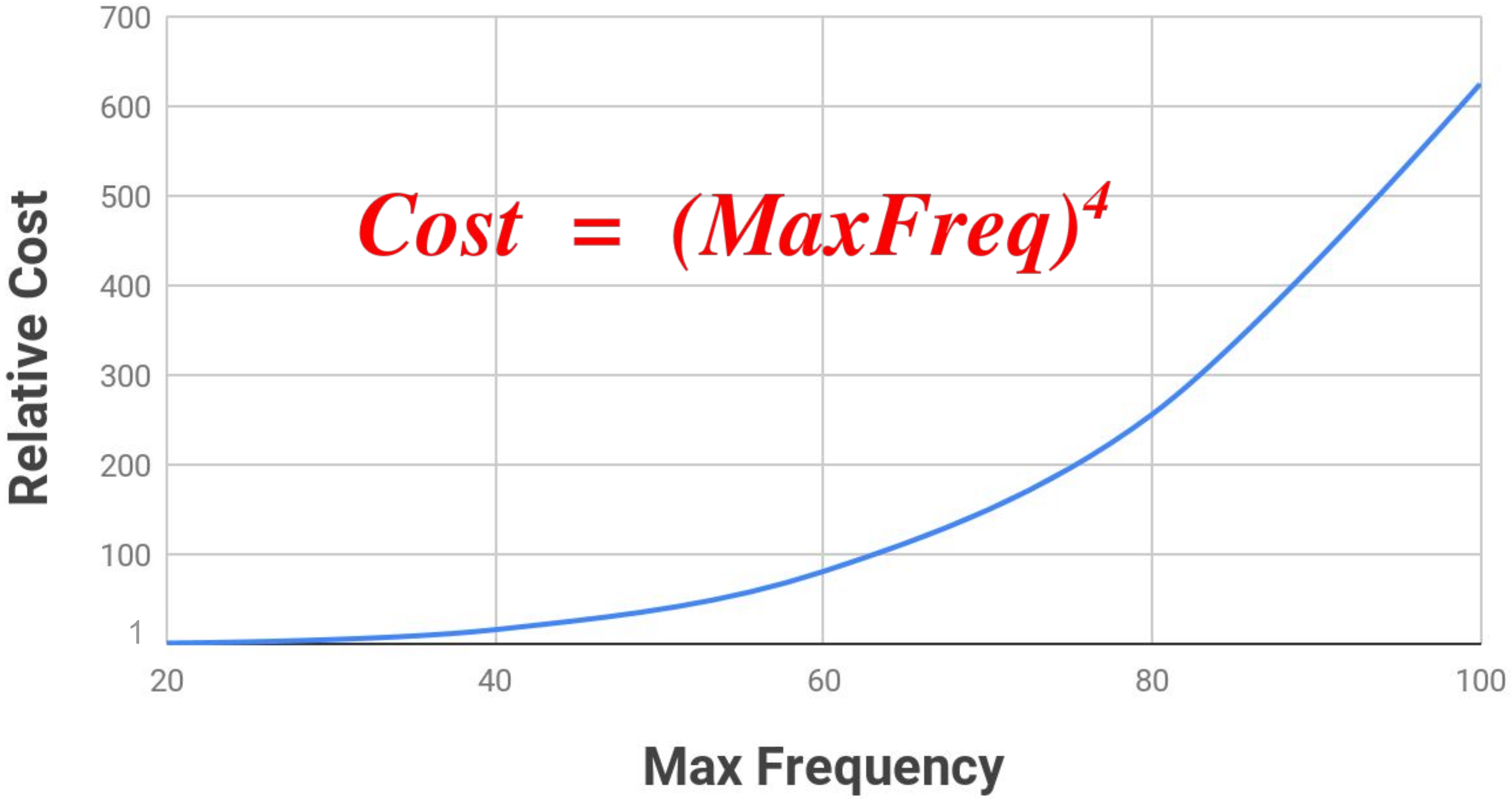


Field data has limited bandwidth

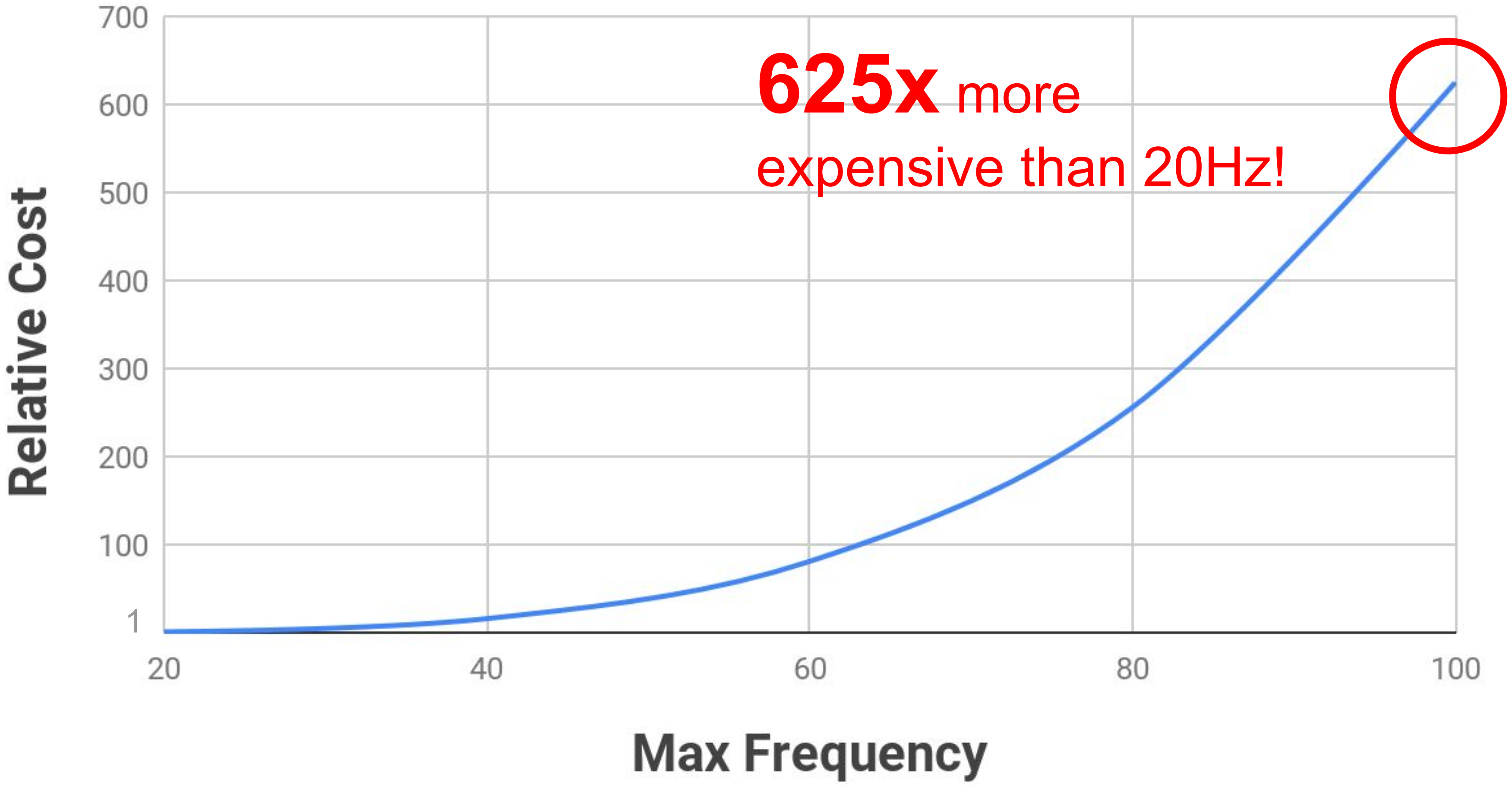




Relative FWI cost



Relative FWI cost



Instead of treating all areas the same ...

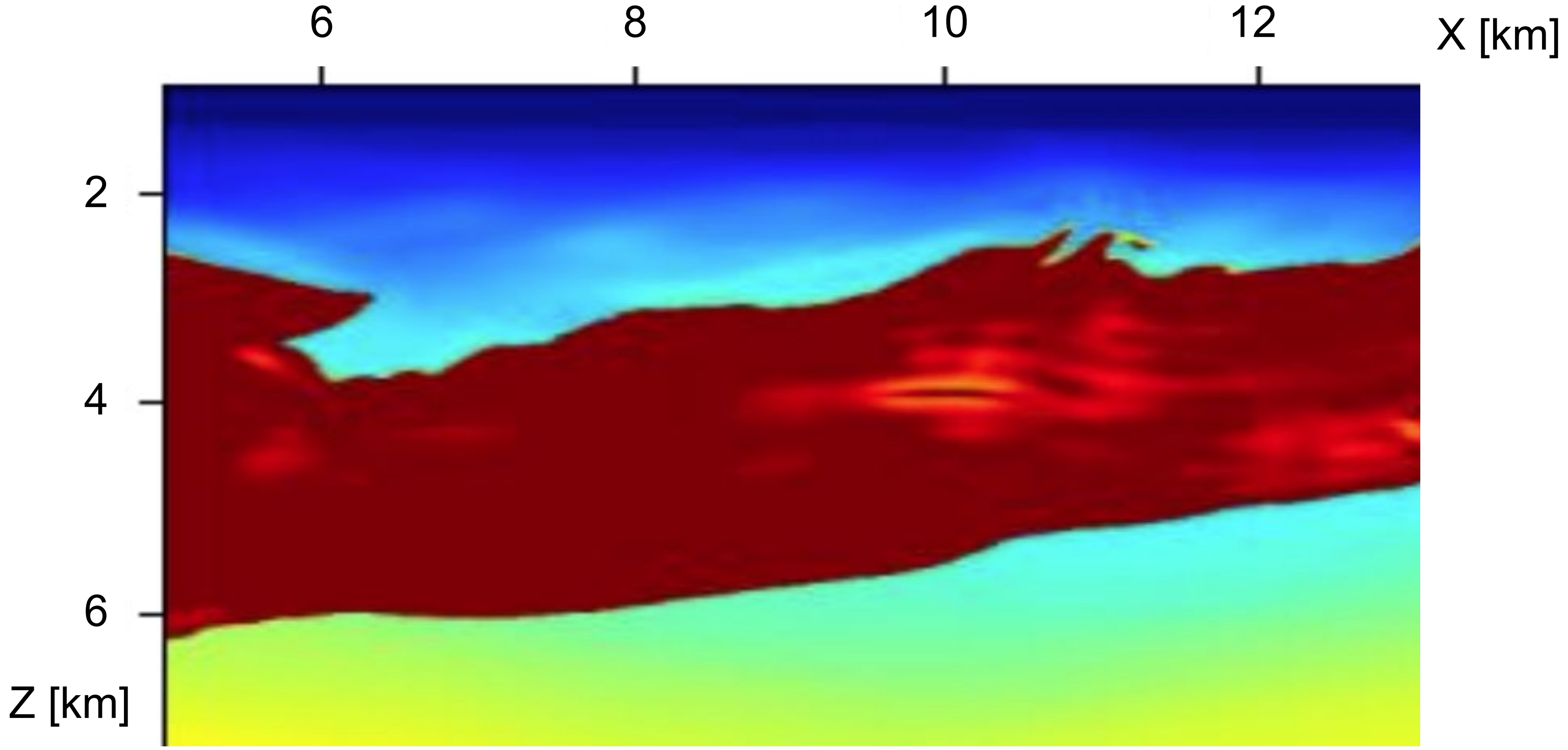


Figure: Xukai Shen, “Salt model building at Atlantis with Full Waveform Inversion“, SEG 2017

Treat salt as a cohesive body

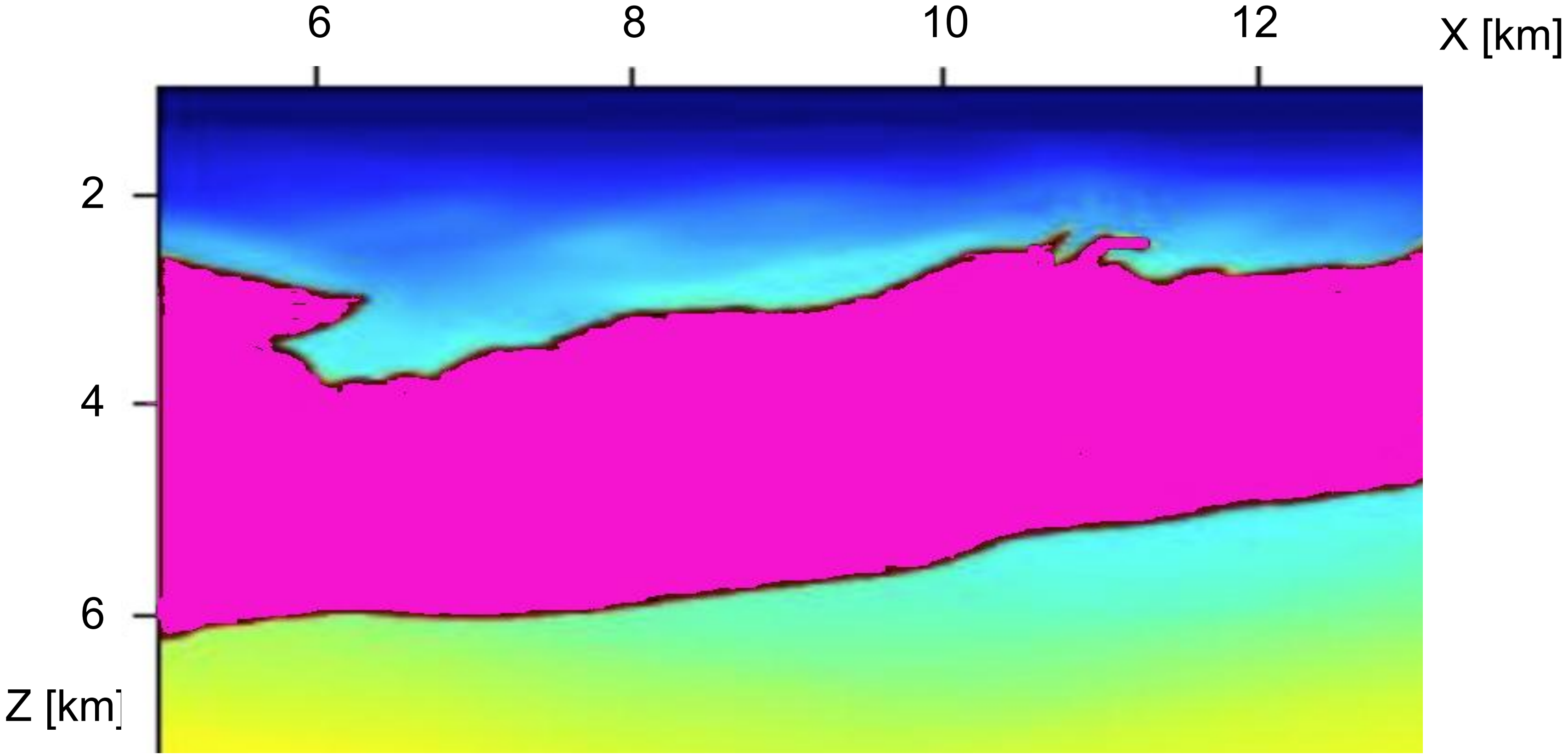


Figure: Xukai Shen, “Salt model building at Atlantis with Full Waveform Inversion“, SEG 2017

How do we keep track of these sharp boundaries?

Track every point on the salt boundary?

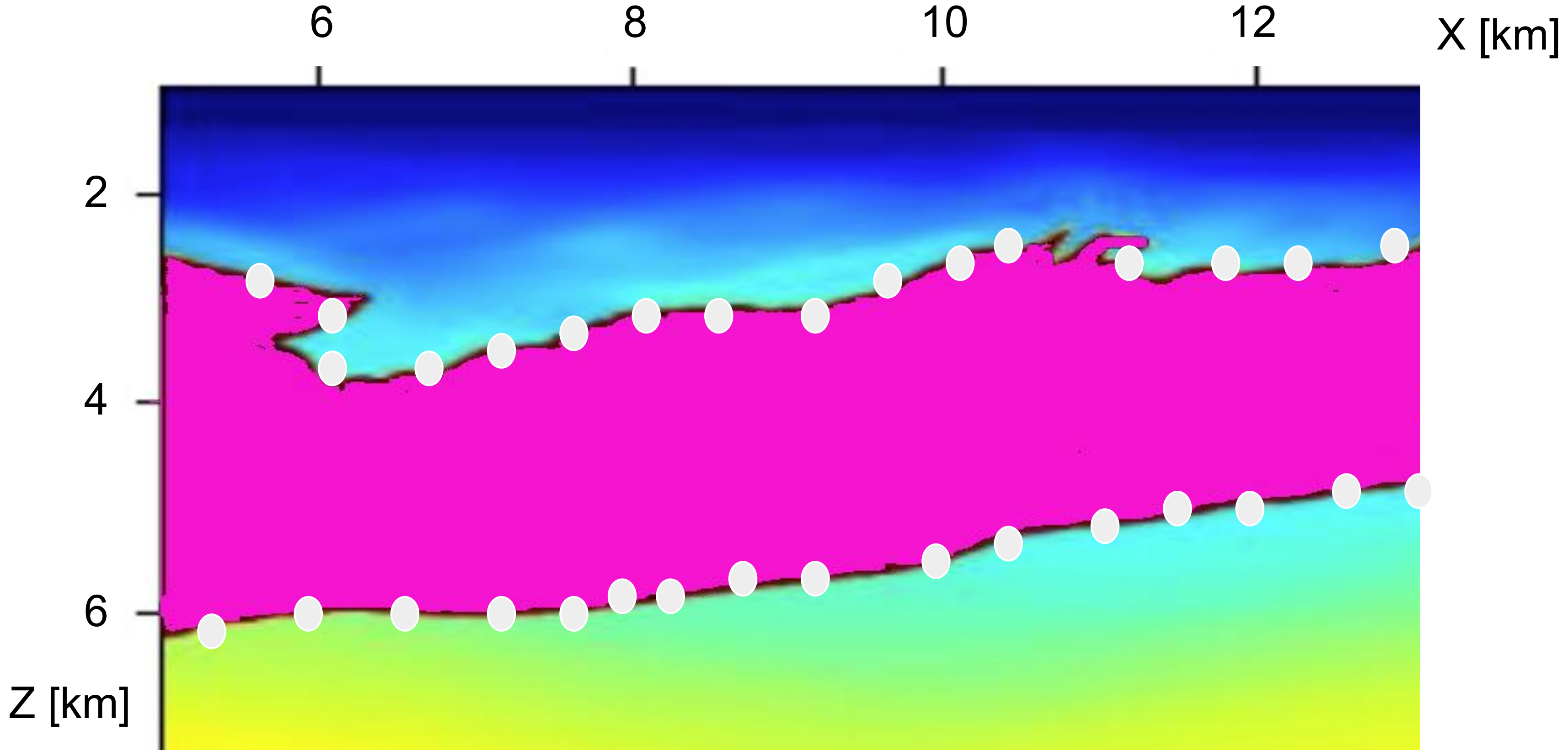
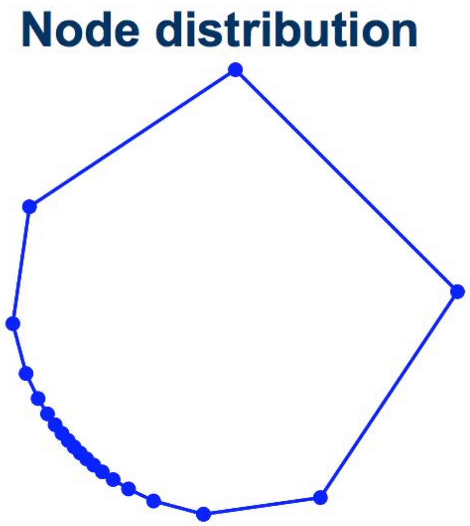


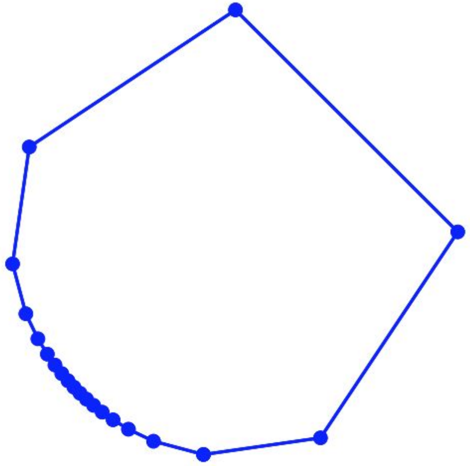
Figure: Xukai Shen, “Salt model building at Atlantis with Full Waveform Inversion“, SEG 2017

Drawback: Distributing points is not intuitive

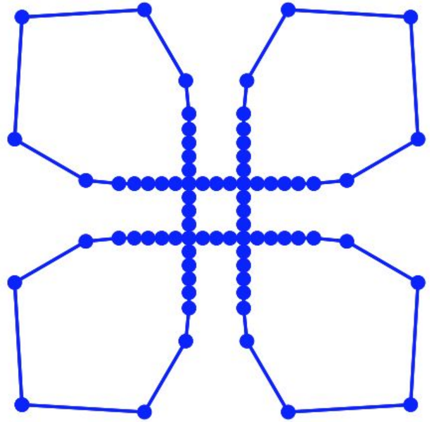


Drawback: Sharp corners become tricky

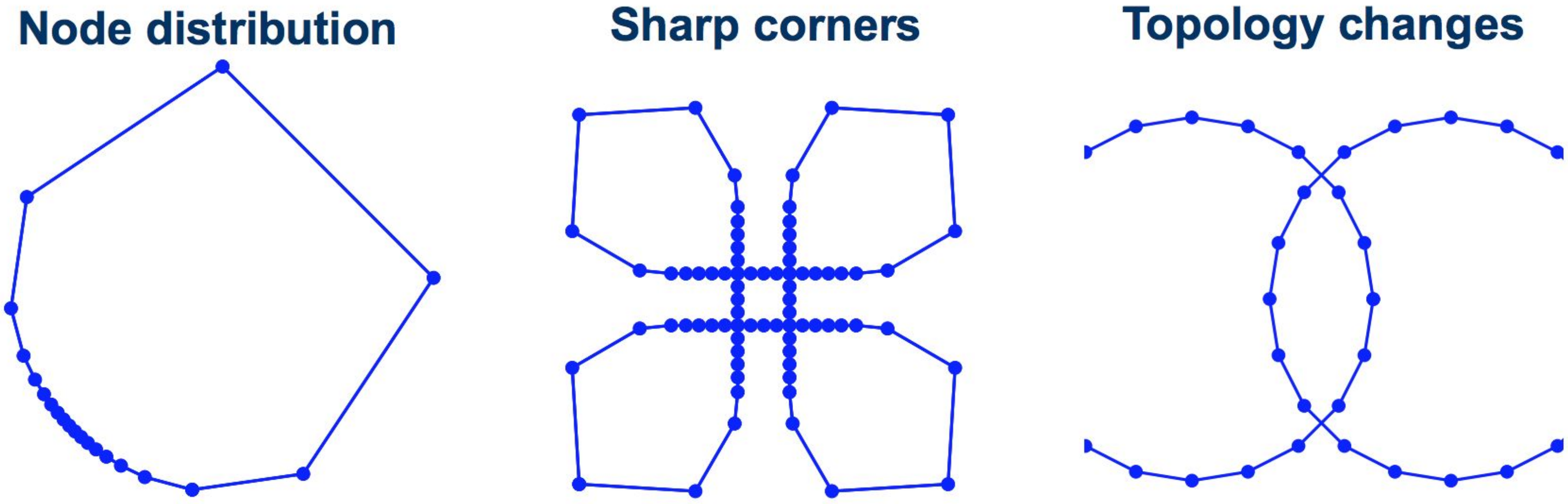
Node distribution



Sharp corners



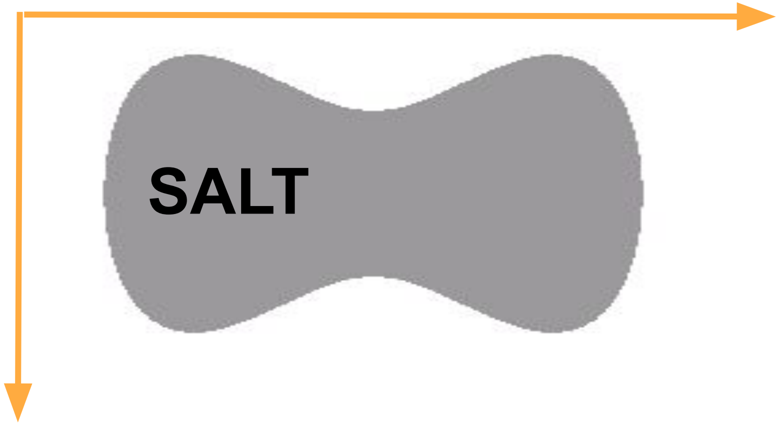
Drawback: Merging/separating bodies is difficult

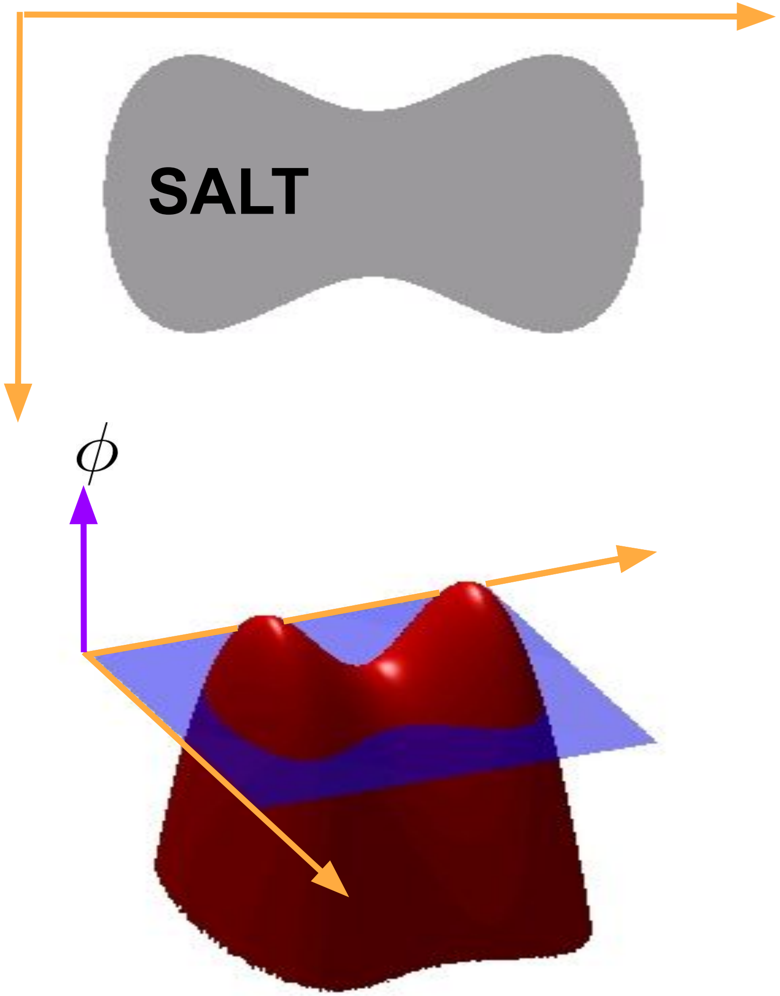


I am going to use **level sets**

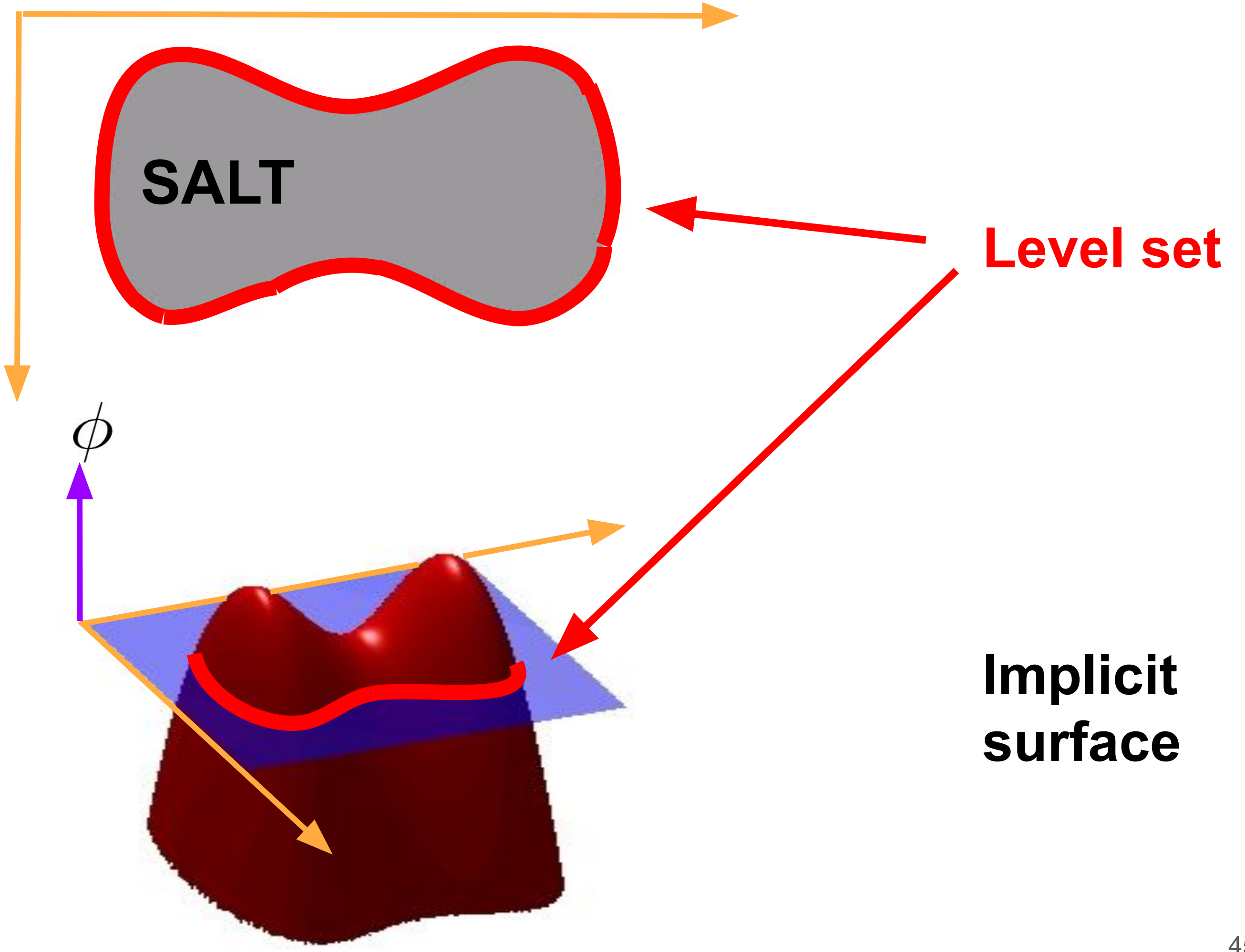
I am going to use **level sets**

What are level sets?

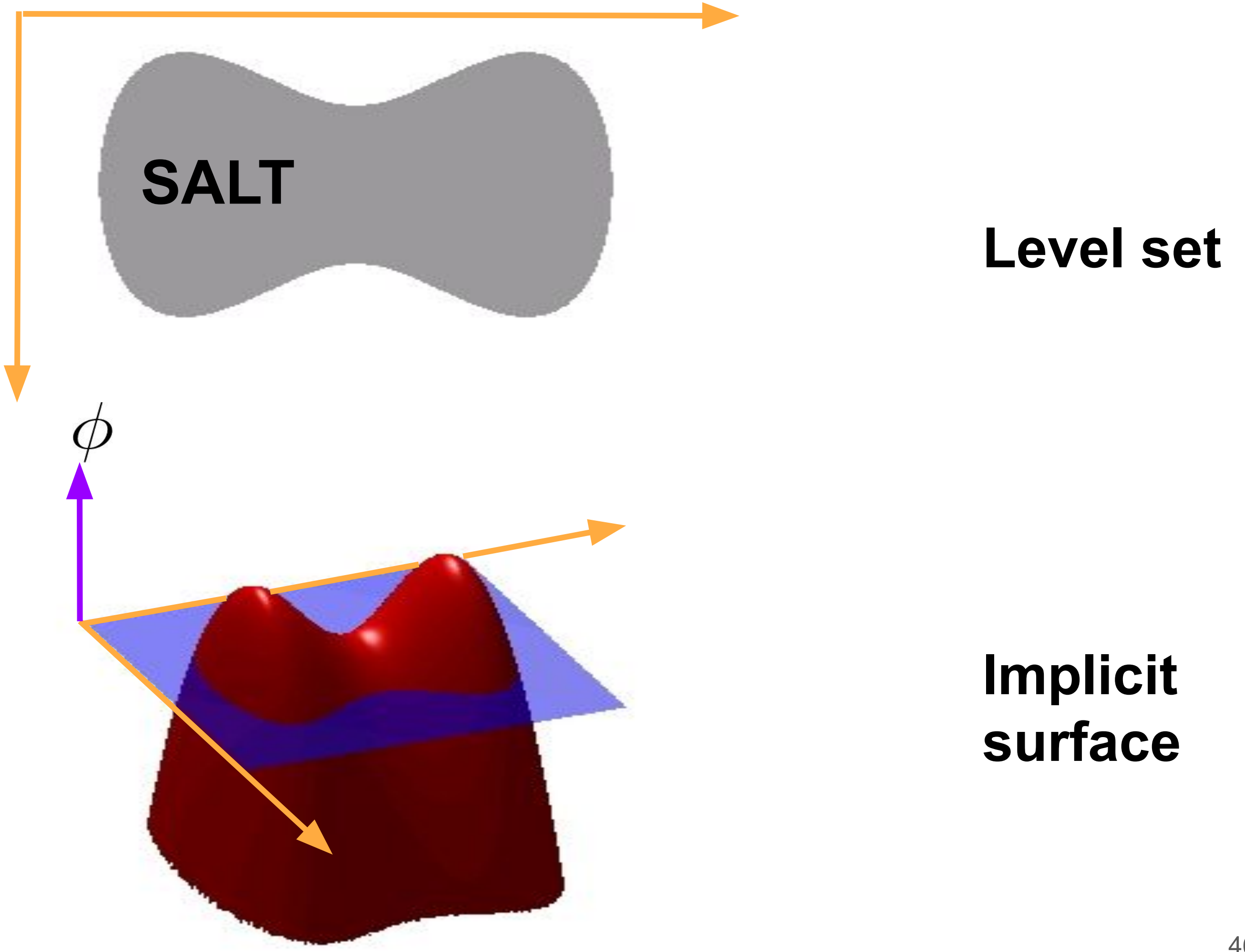

SALT

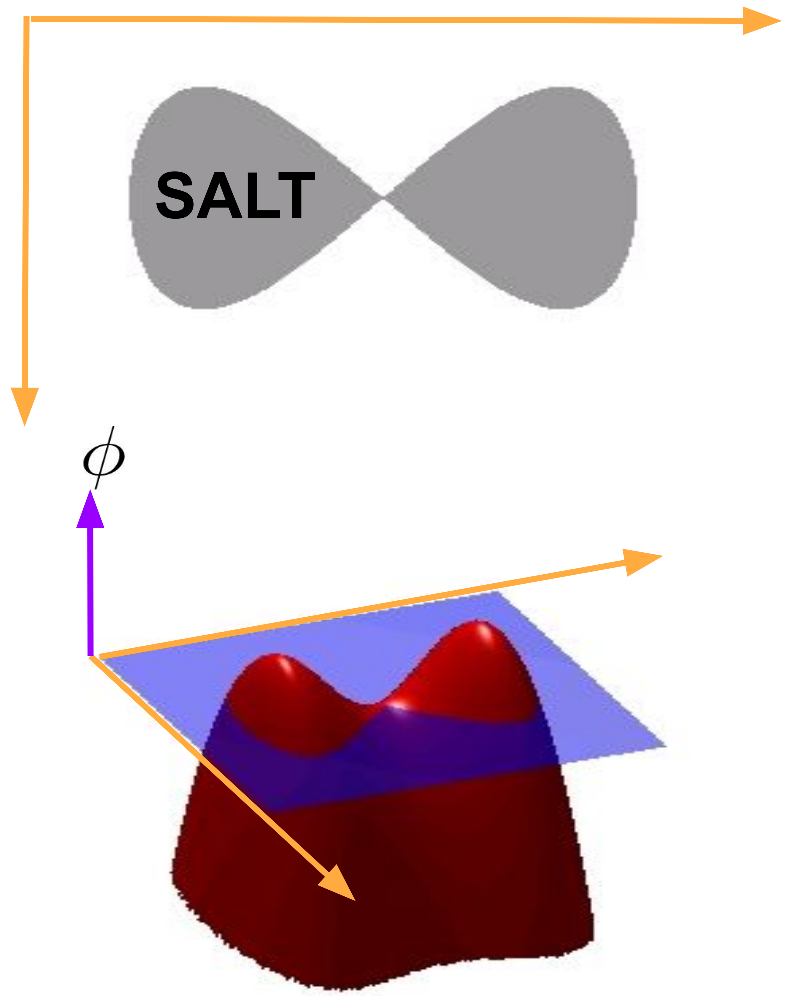


Implicit surface



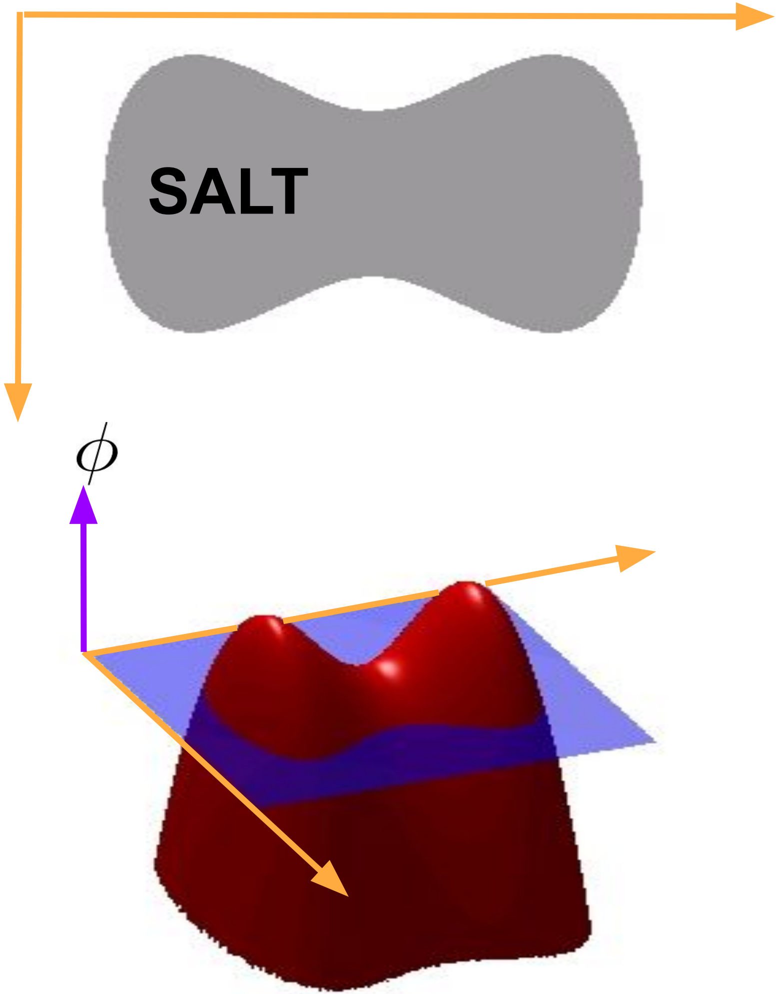
Implicit surface


SALT
ϕ
Level set
Implicit surface



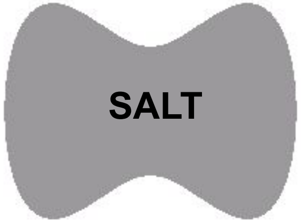
Level set

Implicit surface

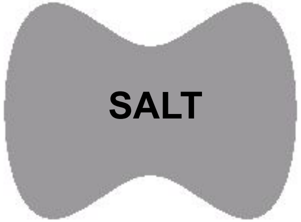


Level set

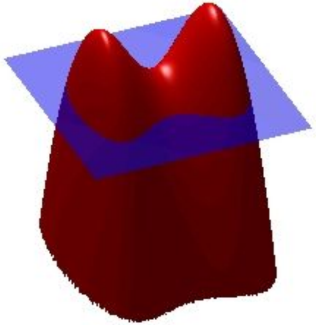
Implicit surface



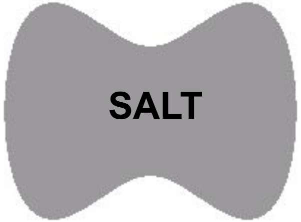
Level set



Level set

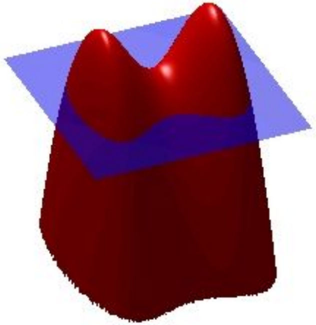


Implicit surface

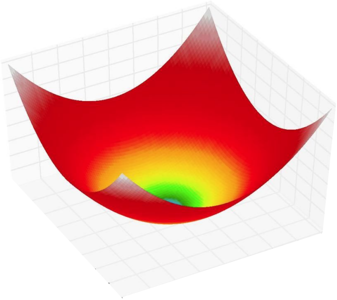


Level set

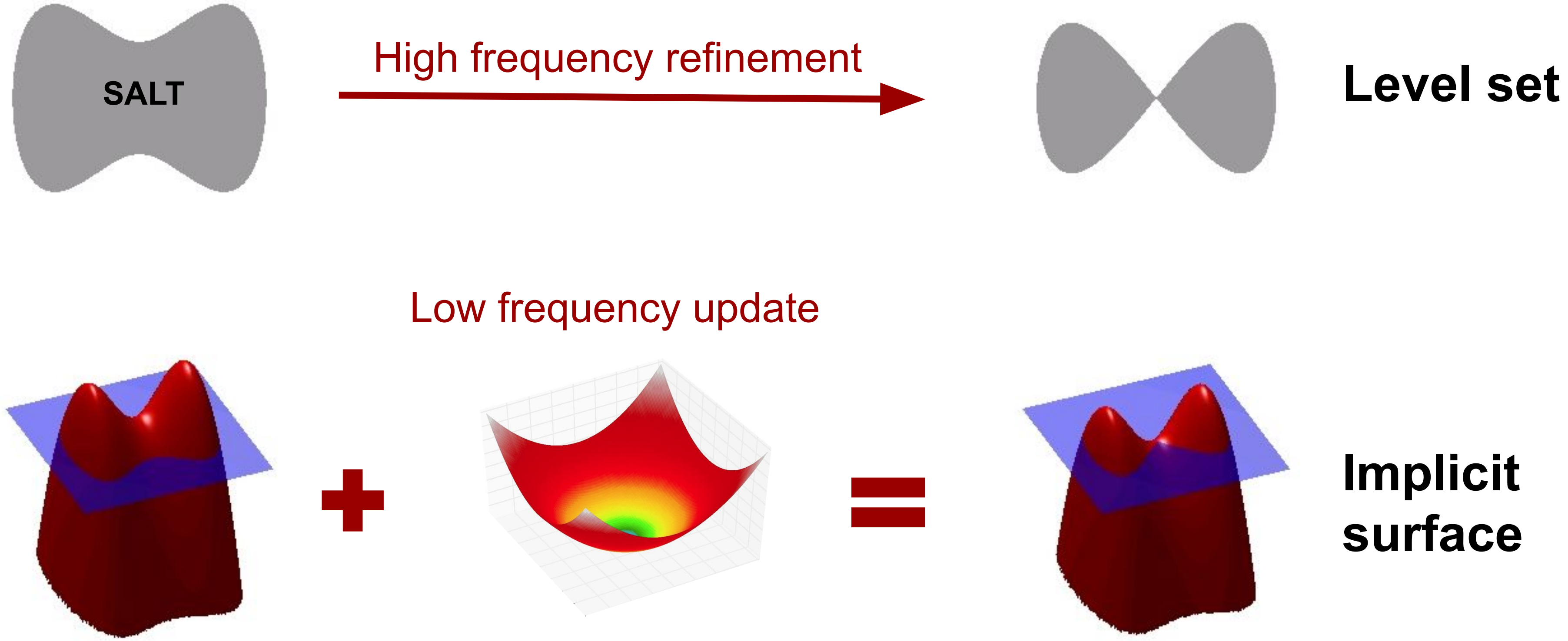
Low frequency update

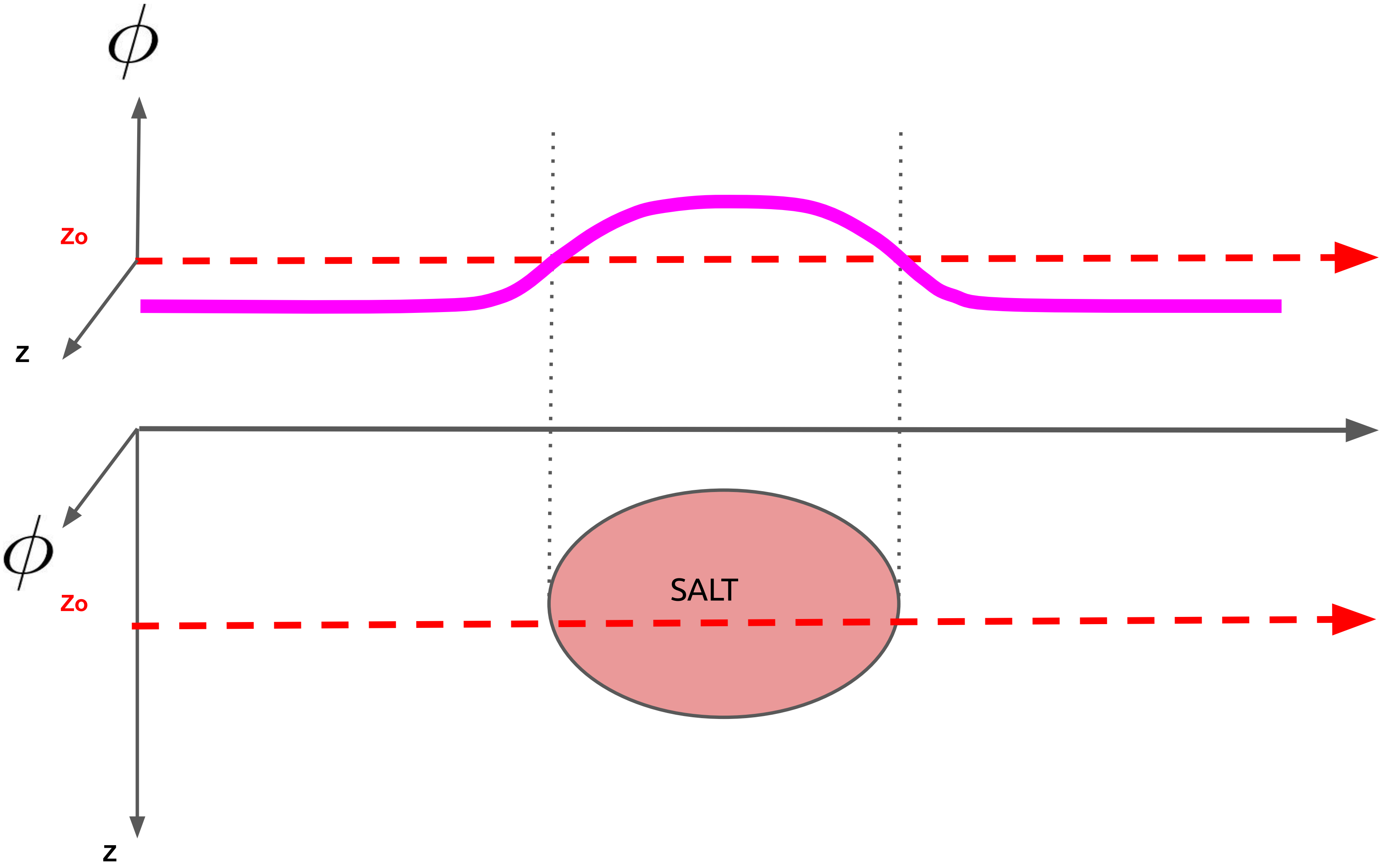


+

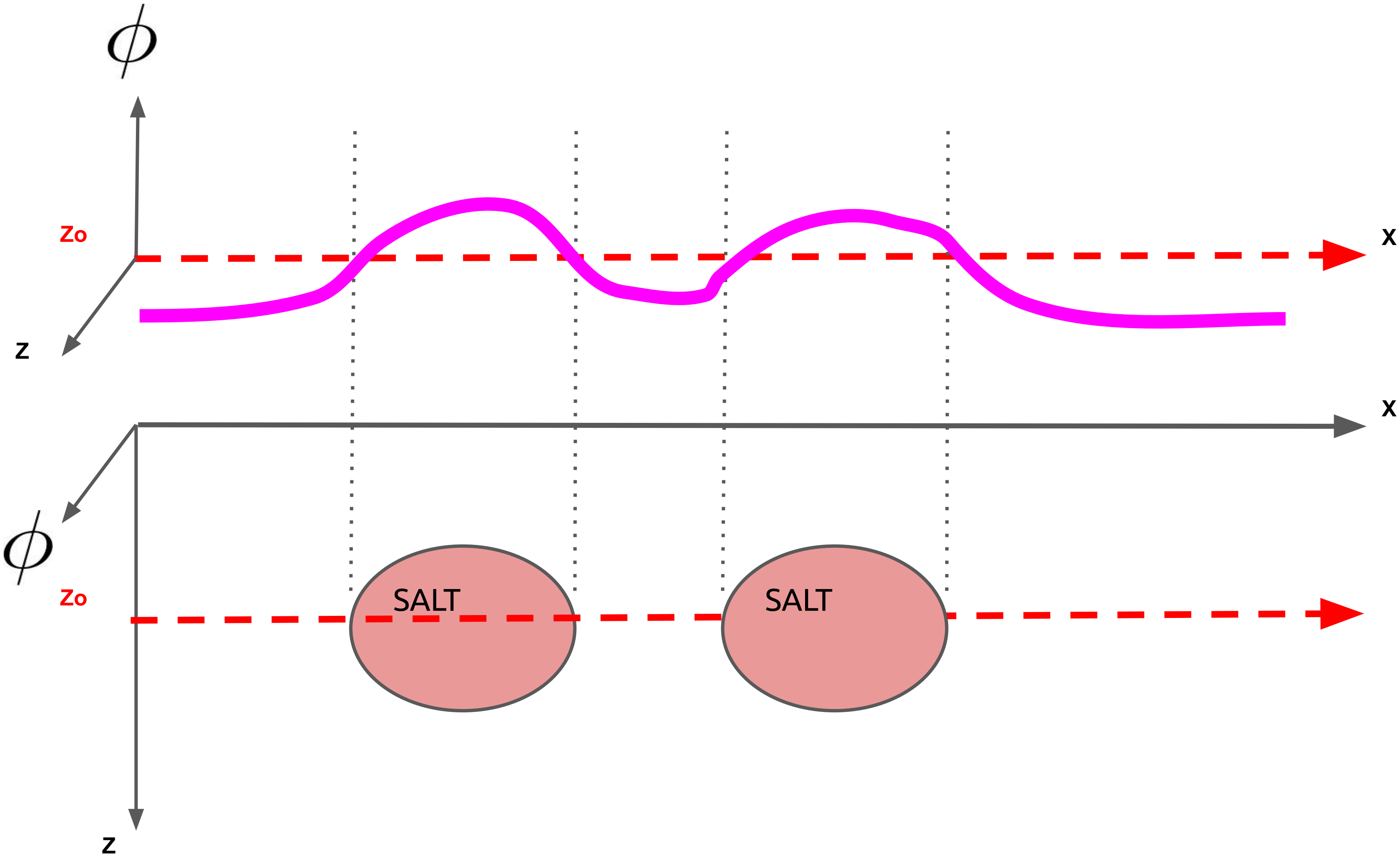


Implicit surface

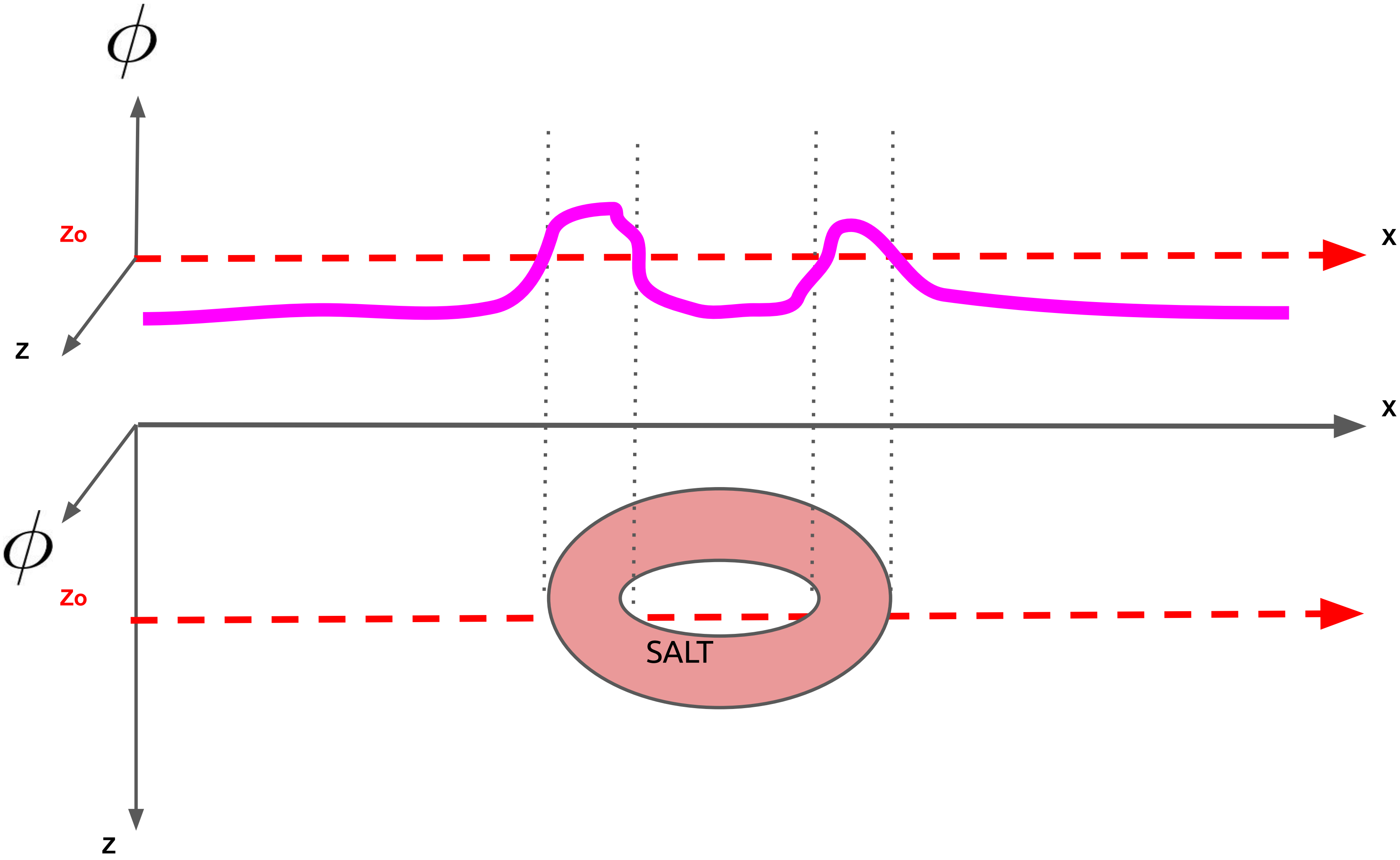

SALT
High frequency refinement
Level set
Low frequency update
Implicit
surface



ϕ
Zo
x
z
x
ϕ
SALT
Zo
z



ϕ
Zo
X
Z
X
ϕ
Zo
SALT
SALT
Z



ϕ
Zo
X
Z
X
ϕ
Zo
SALT
Z

How do we update
the level set so that it
becomes like the **real salt** ?

How do we update
the level set so that it
becomes like the **real salt** ?

Derivation: Objective function

Typical FWI:

$$||F(m)-d_{obs}||_2^2$$

Derivation: Objective function

Typical FWI:

$$||F(m)-d_{obs}||_2^2$$

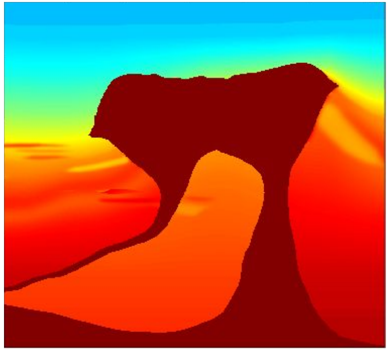
L2 Norm

Derivation: Objective function

Typical FWI:

$$||F(m) - d_{obs}||_2^2$$

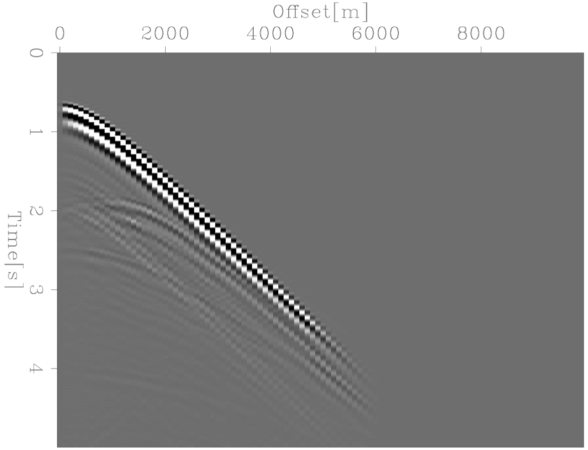
Acoustic wavespeed model



Derivation: Objective function

Typical FWI:

$$||F(m) - d_{obs}||_2^2$$

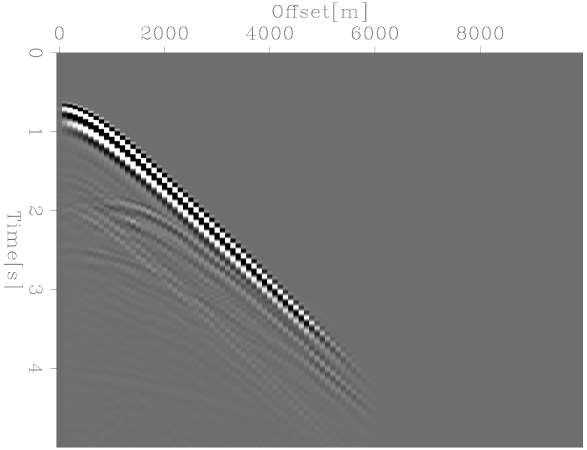


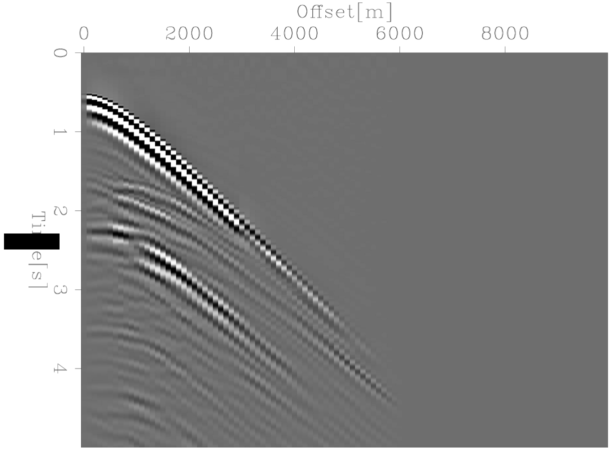
Synthetic data modeling

Derivation: Objective function

Typical FWI:

$$\|F(m)-d_{obs}\|_2^2$$



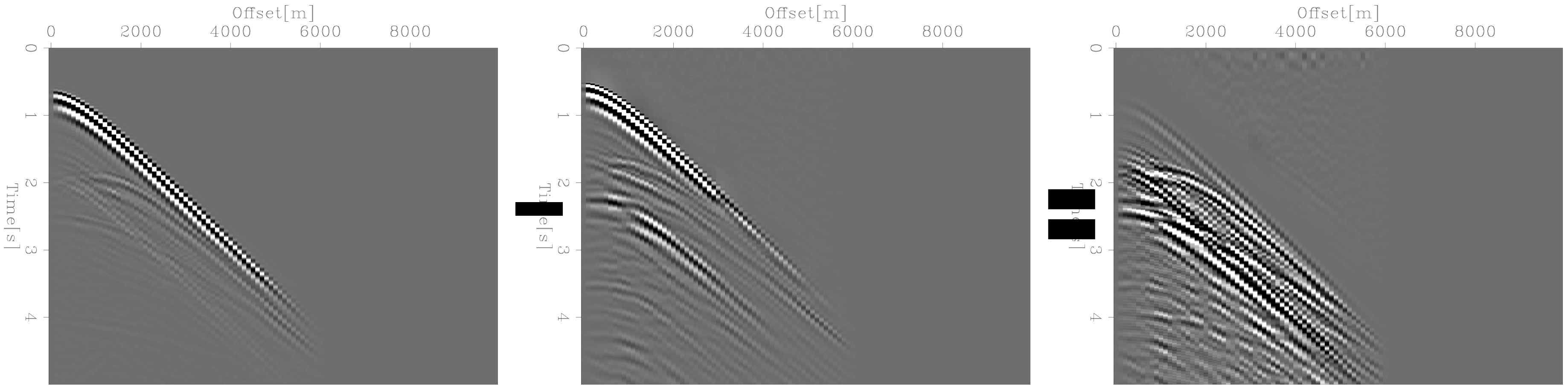


Observed acoustic data

Derivation: Objective function

Typical FWI:

$$\| F(m) - d_{obs} \|_2^2$$



Data residual

Derivation: Objective function

Typical FWI:

$$||F(m)-d_{obs}||_2^2$$

First derivative = Gradient

Derivation: Objective function

Typical FWI:

$$||F(m)-d_{obs}||_2^2$$

First derivative = Gradient

Second derivative = Hessian

Derivation: Objective function

Typical FWI:

$$||F(m)-d_{obs}||_2^2$$

First derivative = Gradient

Second derivative = Hessian

Derivation: Objective function

Level set FWI:

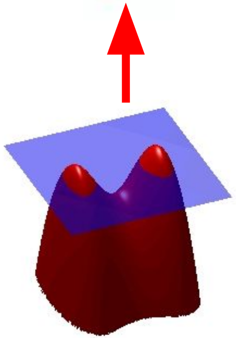
$$||F(m(\phi,b))-d_{obs}||_2^2$$

Derivation: New model space

$$m(\phi,b)=H(\phi)(c_s-b)+b$$

Derivation: New model space

$$m(\phi, b) = H(\phi)(c_s - b) + b$$

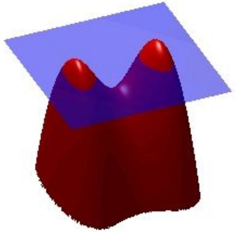


Implicit surface

Derivation: New model space

Salt velocity

$$m(\phi,b)=H(\phi)(c_s-b)+b$$

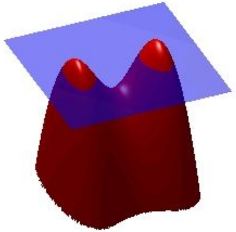


Implicit surface

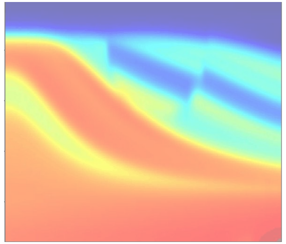
Derivation: New model space

Salt velocity

$$m(\phi, b) = H(\phi)(c_s - b) + b$$



Implicit surface

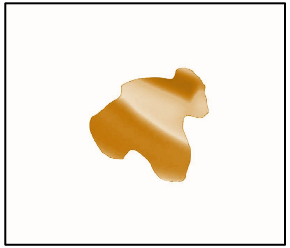


Background velocity

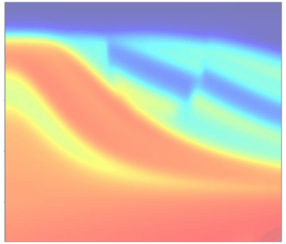
Derivation: New model space

(Approximate)
Heaviside
function

$$m(\phi, b) = H(\phi)(c_s - b) + b$$



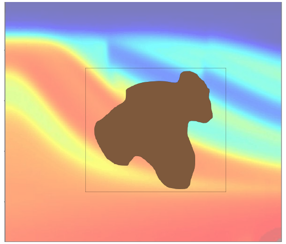
Salt body overlay



Background velocity

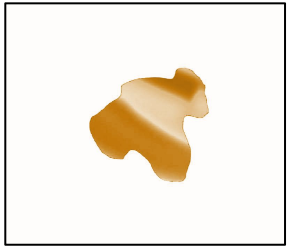
Derivation: New model space

$$m(\phi,b)=\underbrace{H(\phi)(c_s-b)}+b$$



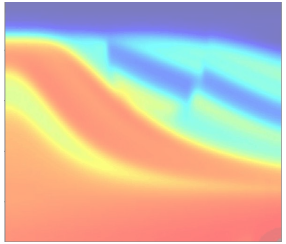
Full acoustic velocity

=



Salt body overlay

+



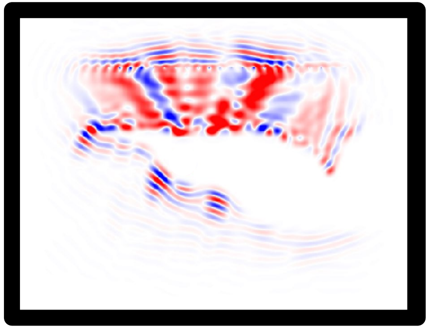
Background velocity

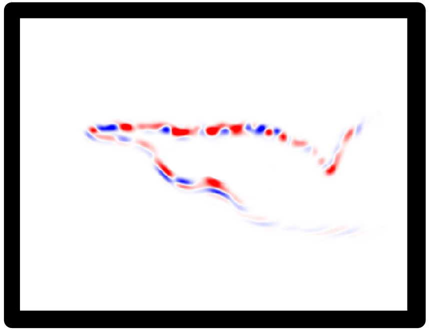
Derivation: Gradient

$$\begin{bmatrix} g_b \\ \\ g_\phi \end{bmatrix}$$

Derivation: Gradient

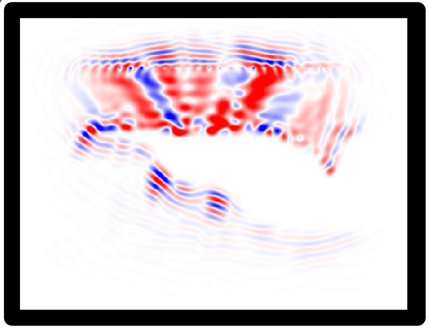
$$\begin{pmatrix} g_b \\ g_\phi \end{pmatrix} =$$

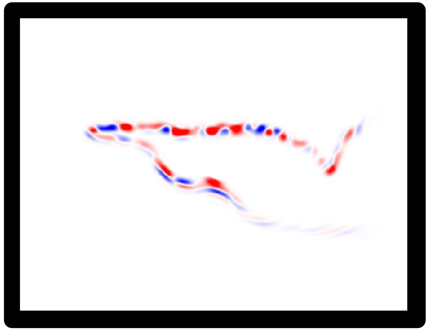




Derivation: Gradient

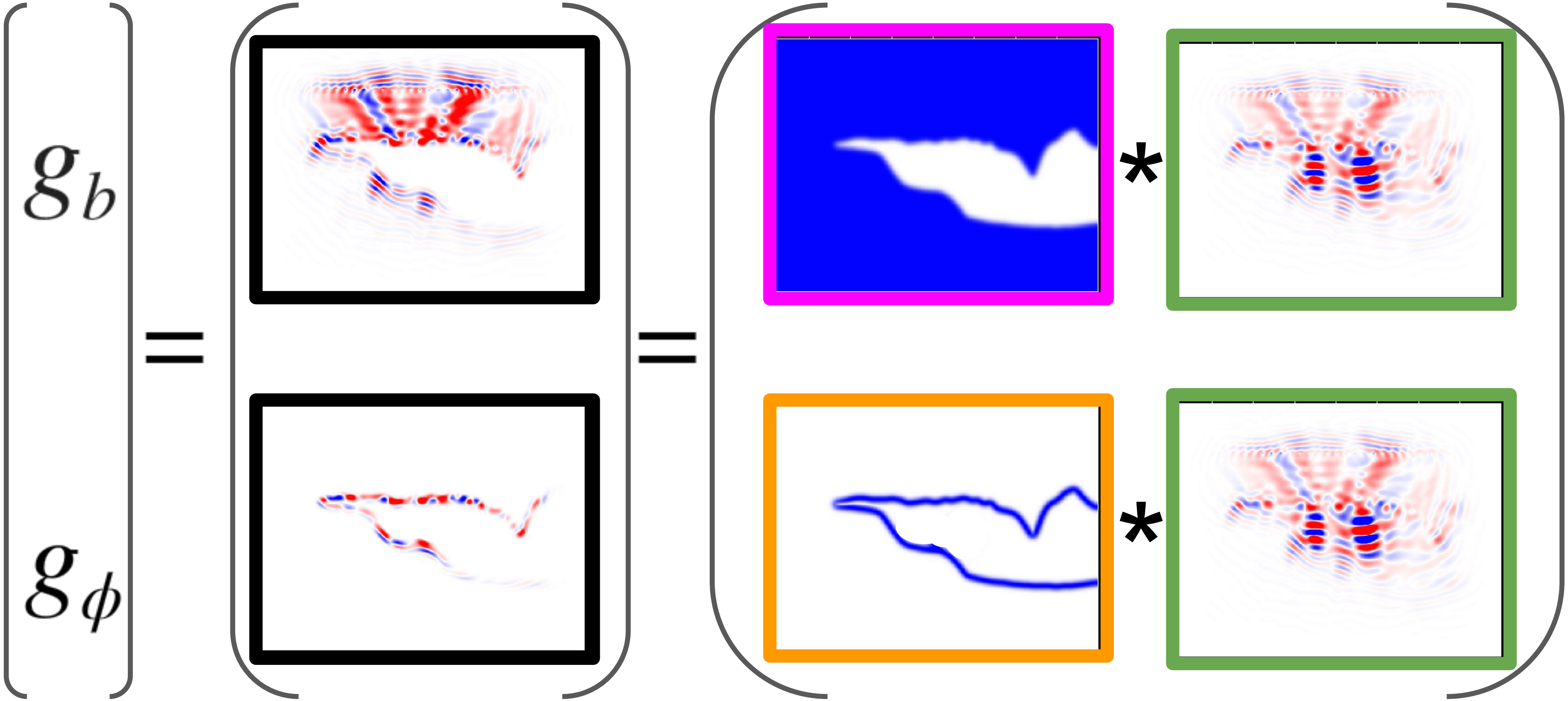
$$\begin{pmatrix} g_b \\ g_\phi \end{pmatrix} =$$





$$= \begin{pmatrix} \left(\frac{\delta m}{\delta b}\right)^T * \left.\left(\frac{\delta F(m)}{\delta m}\right)^T\right|_{m_o} \triangle d \\ \left(\frac{\delta m}{\delta \phi}\right)^T * \left.\left(\frac{\delta F(m)}{\delta m}\right)^T\right|_{m_o} \triangle d \end{pmatrix}$$

Derivation: Gradient



Isn’t this new model space
twice as big now?

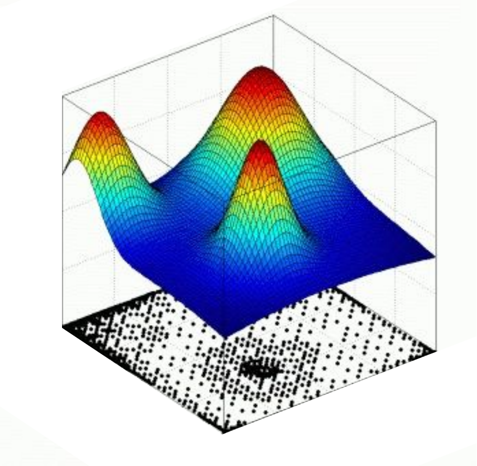
$$m(\phi, b) = H(\phi)(c_s - b) + b$$

Yes, more model parameters ....

$$m(\phi, b) = H(\phi)(c_s - b) + b$$

Yes, more model parameters ....
But we can use less!

Build implicit surface with Radial Basis Functions (RBFs)



$$= \sum_{i=1}^{3} \lambda_i$$

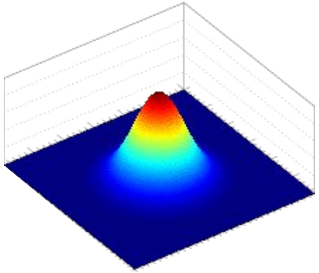
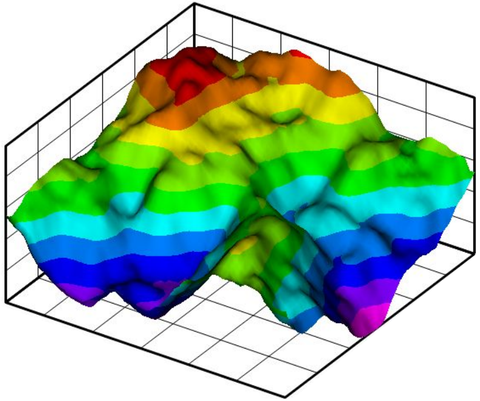


Image credit: http://www.it.uu.se/research/scientific_computing/project/rbf/rbfpde

Build implicit surface with Radial Basis Functions (RBFs)



$$= \sum_{i=1}^{100} \lambda_i$$

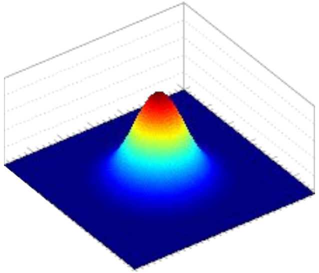
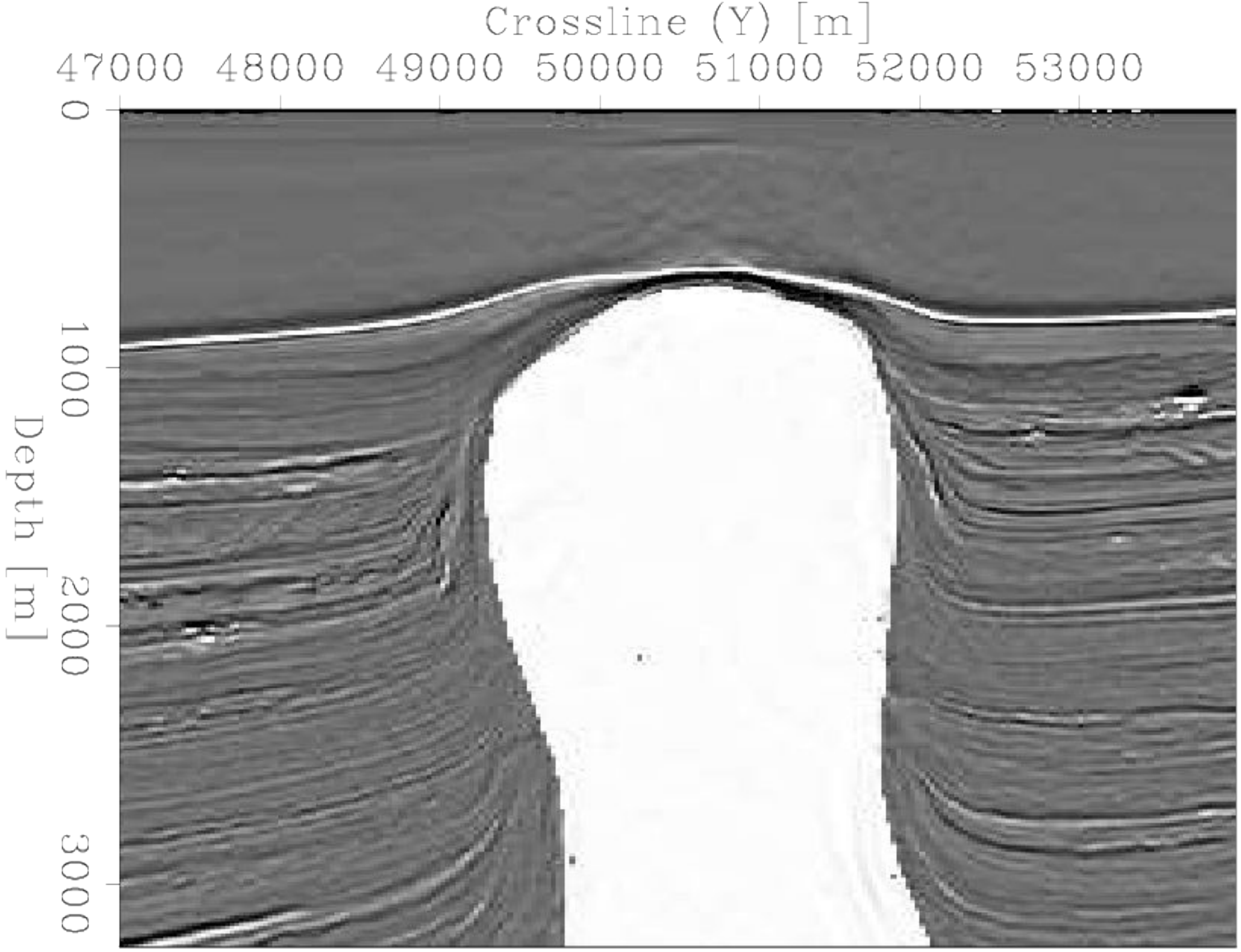
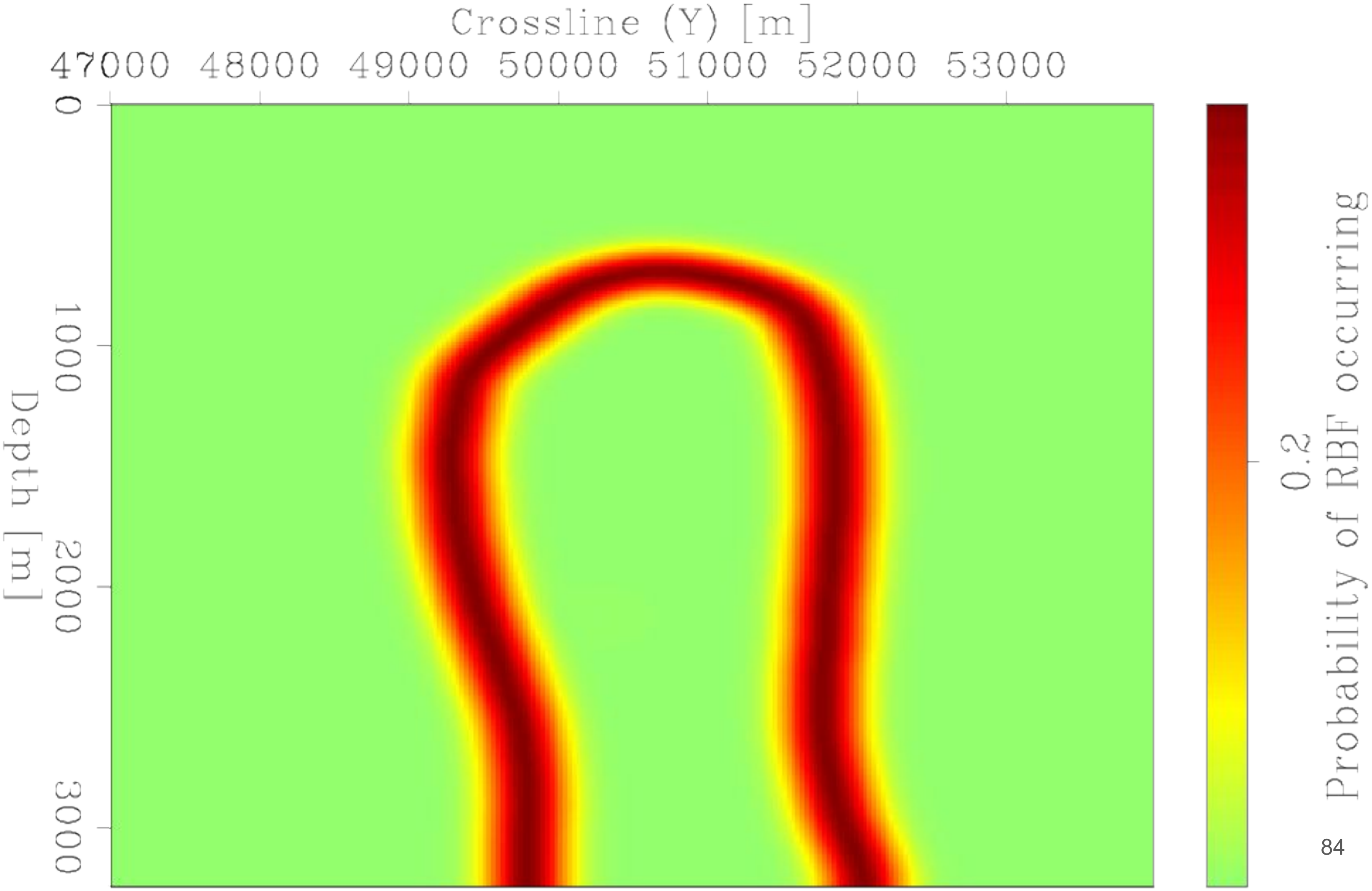
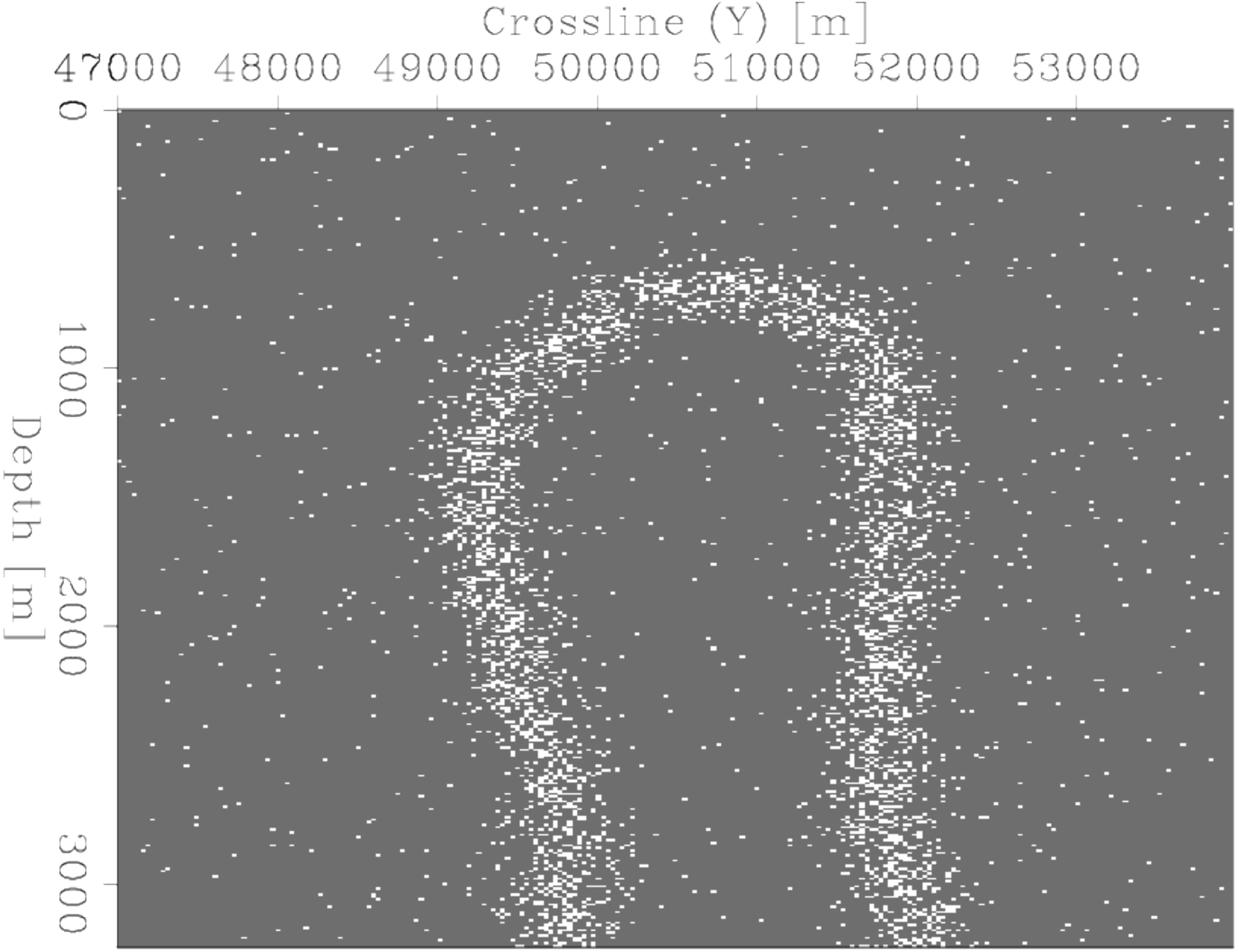
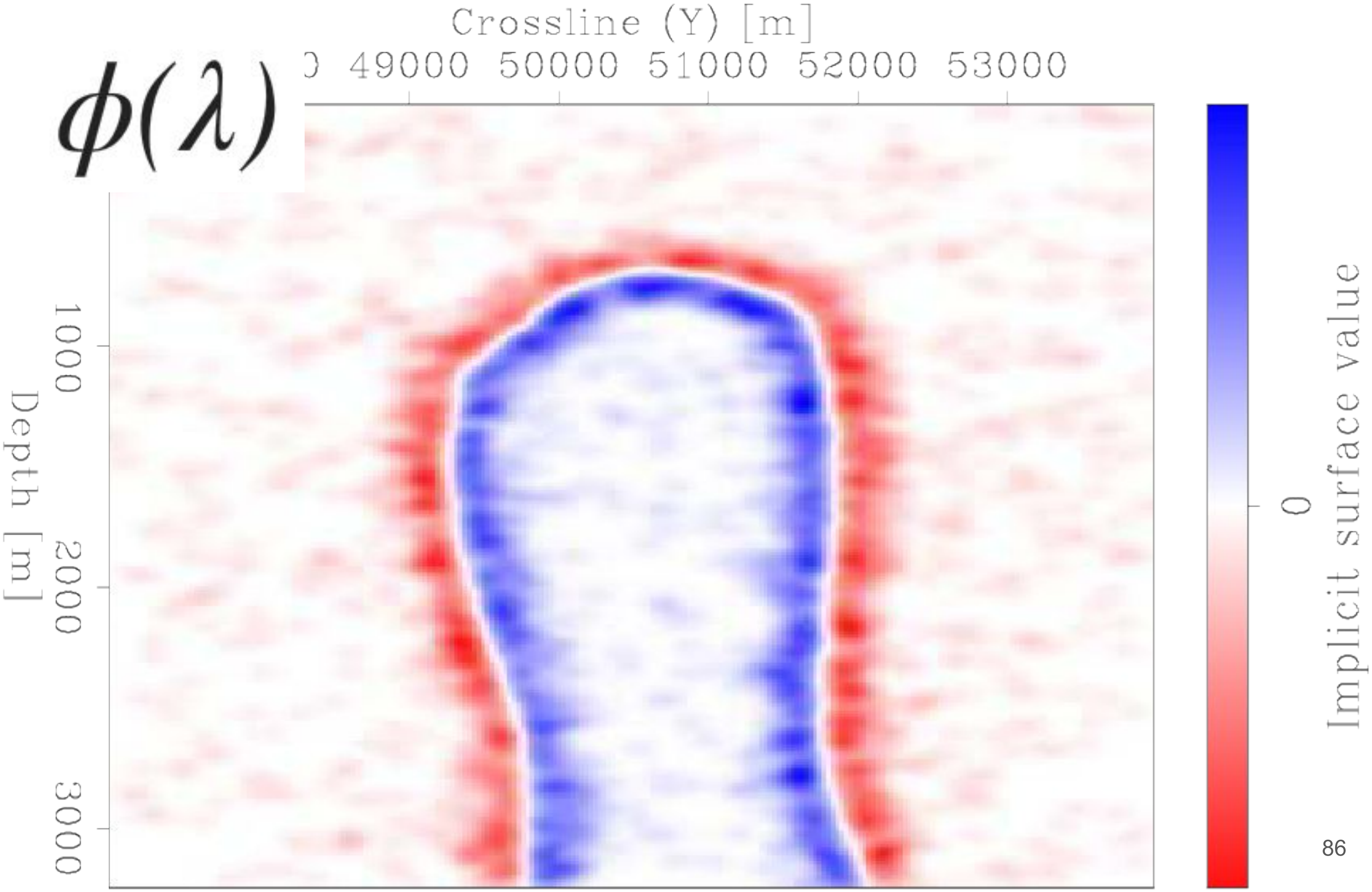


Image credit: http://www.dplot.com/features.htm


Crossline (Y) [m]
47000 48000 49000 50000 51000 52000 53000
Depth [m]
0
1000
2000
3000

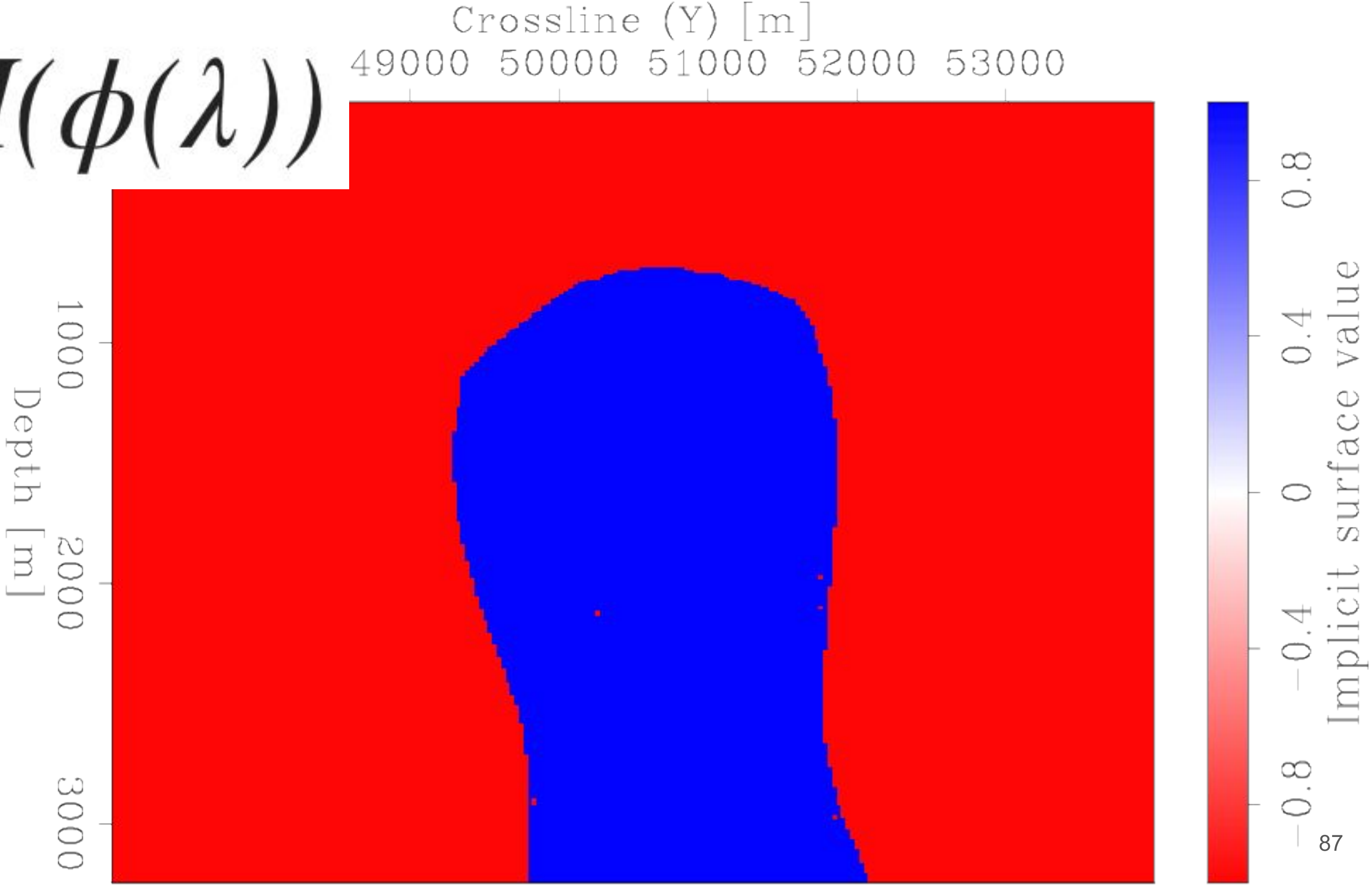

Crossline (Y) [m]
47000
48000
49000
50000
51000
52000
53000
0
1000
2000
3000
Depth [m]
0.2
Probability of RBF occurring


Crossline (Y) [m]
47000 48000 49000 50000 51000 52000 53000
Depth [m]
0
1000
2000
3000

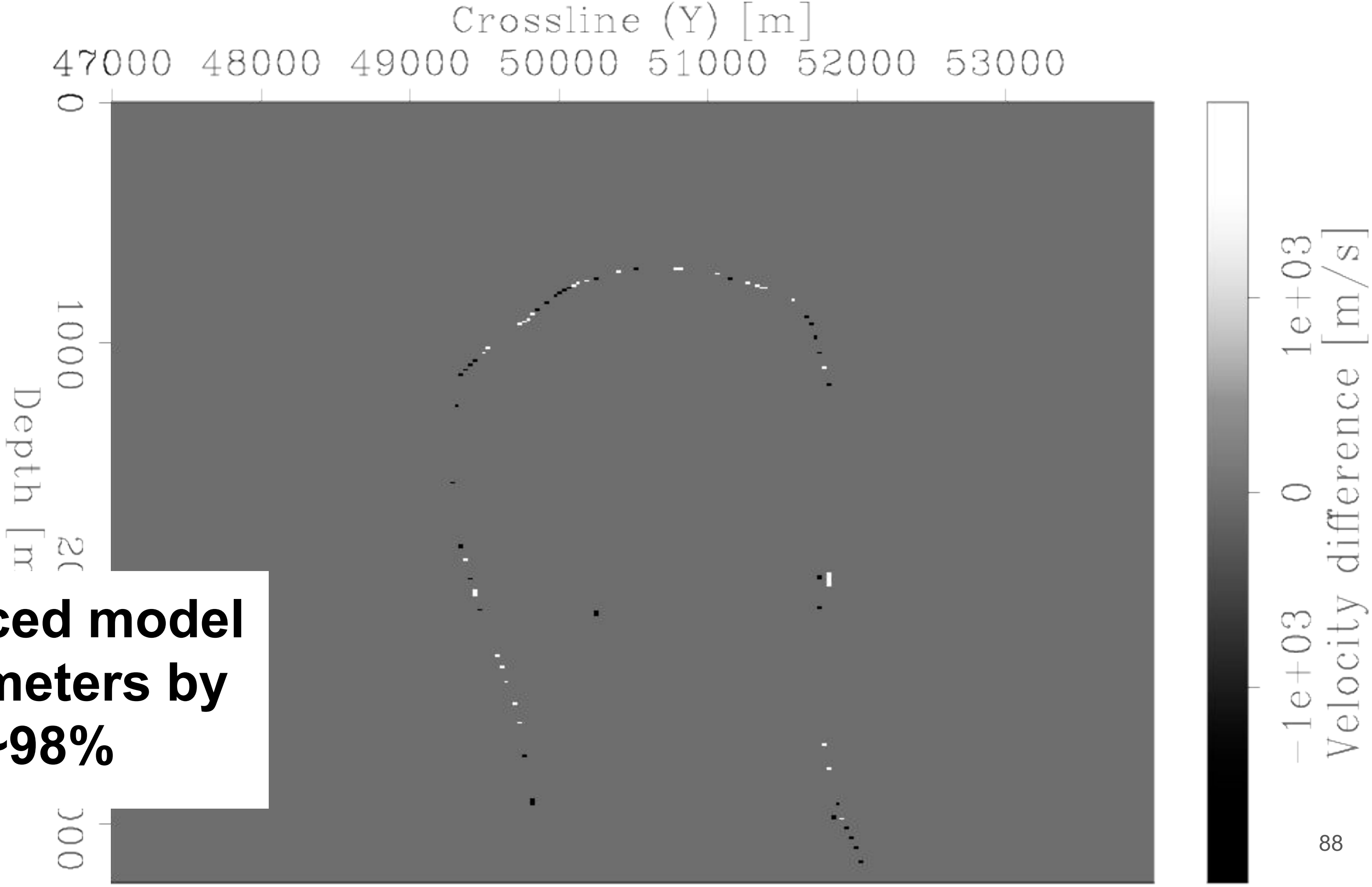


$\phi(\lambda)$
Crossline (Y) [m]
49000
50000
51000
52000
53000
Depth [m]
1000
2000
3000
0
Implicit surface value

$$H(\phi(\lambda))$$



Crossline (Y) [m]
49000
50000
51000
52000
53000
Depth [m]
1000
2000
3000
Implicit surface value
0.8
0.4
0
−0.4
−0.8



Reduced model parameters by ~98%

Do RBFs help improve the inversion outcome?

First-order descent is okay sometimes ….

$$\triangle \boldsymbol{m} = -\boldsymbol{g}$$

...But Newton's method allows us to converge faster

$$H \triangle m = -g$$

...But Newton's method allows us to converge faster

$$H \triangle m = -g$$

Solve Newton equation

$$\triangle m = -H^{-1} g$$

...But Newton's method allows us to converge faster

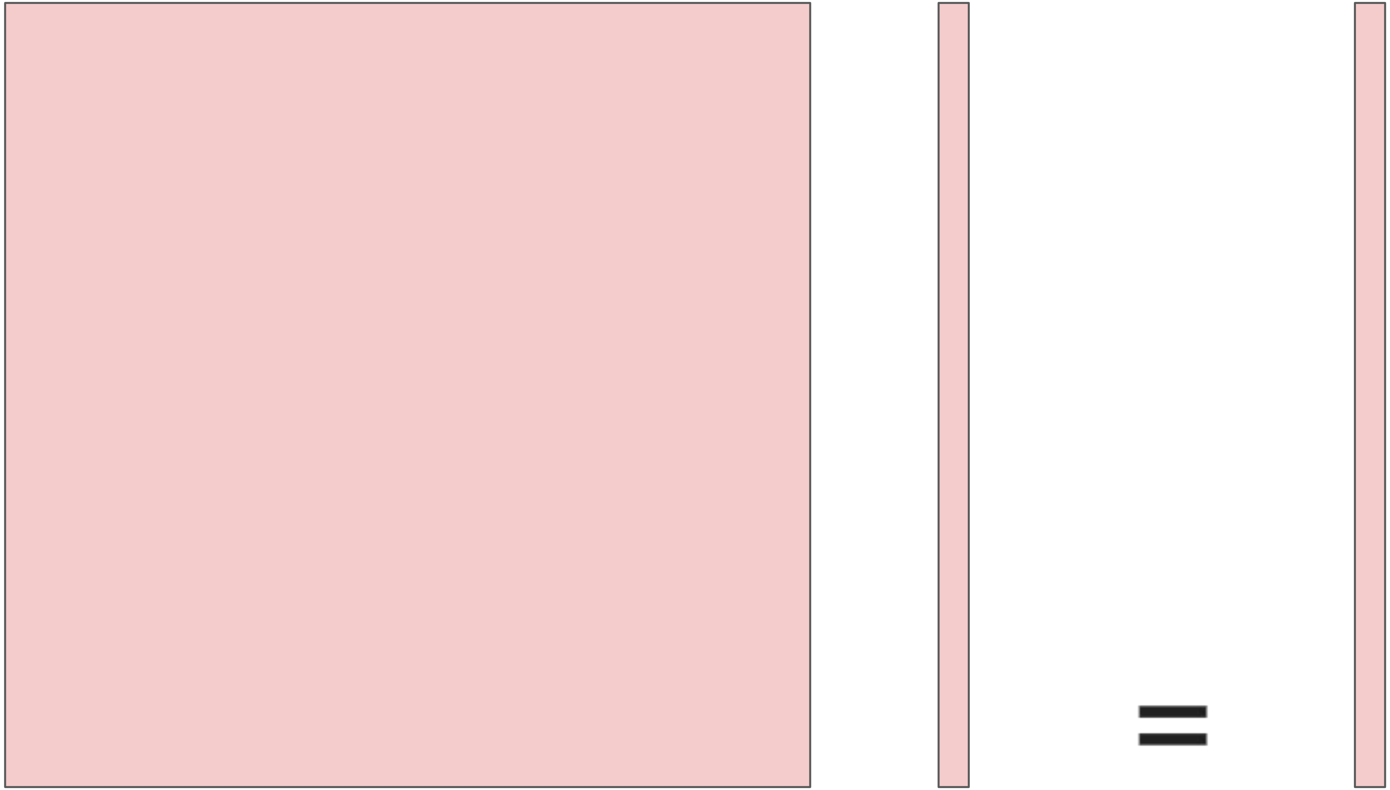
$$H \triangle m = -g$$

$$\triangle m = -H^{-1} g$$

Almost always use iterative methods

$$H_\phi \triangle\phi = -g_\phi$$

$$H_\phi \triangle\phi = -g_\phi$$

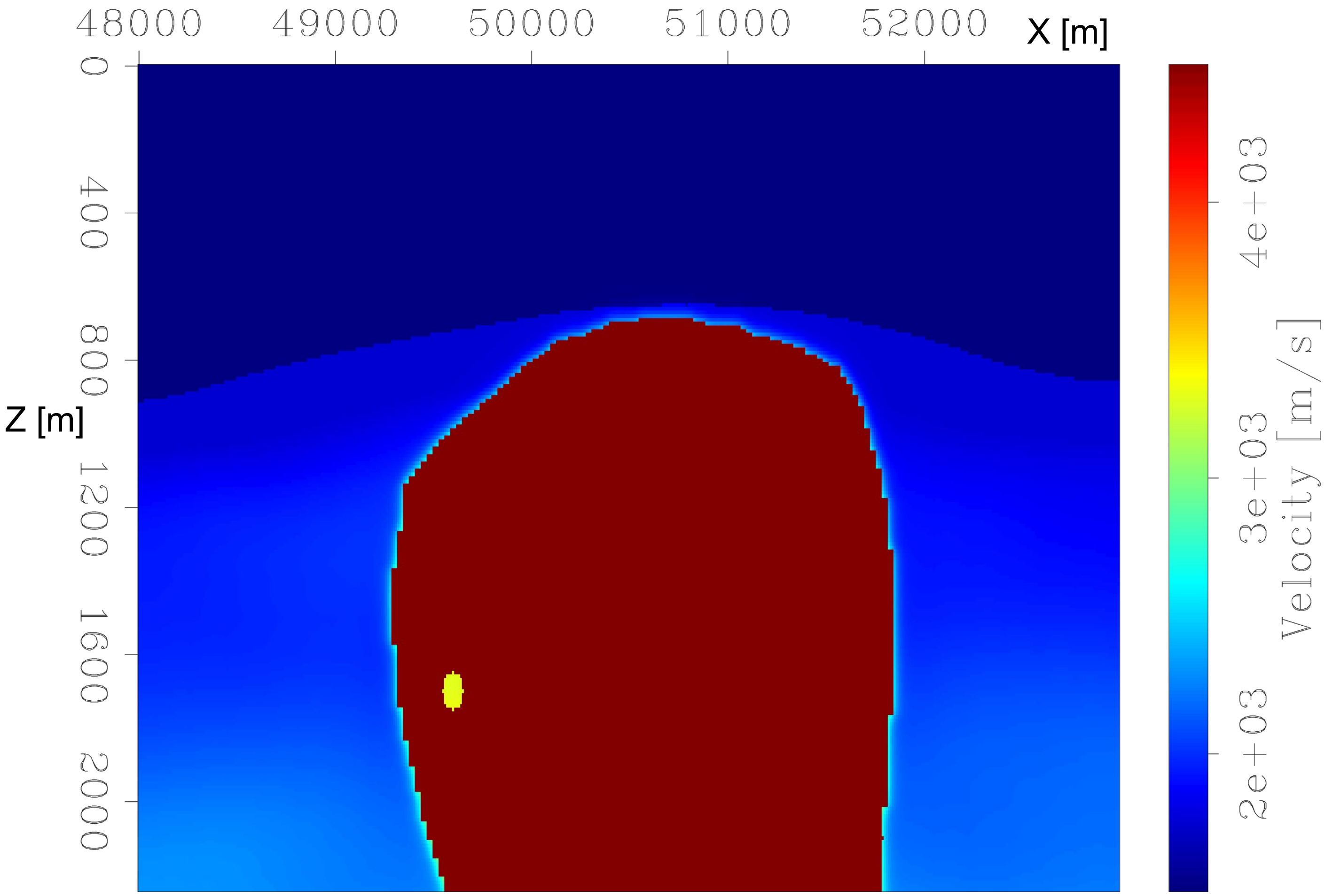


$$\boldsymbol{H}_\lambda \triangle\lambda = -\boldsymbol{g}_\lambda$$

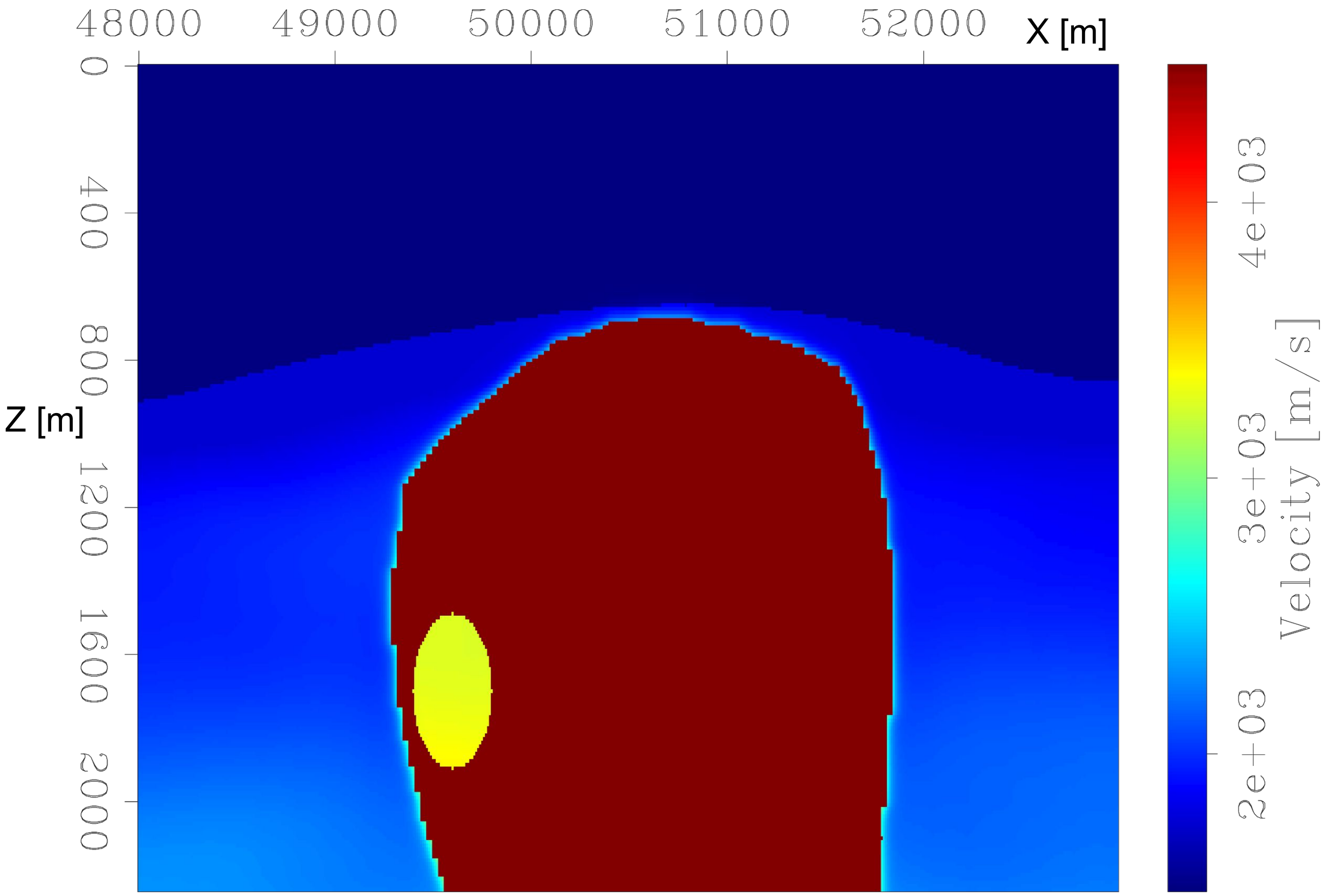
Smaller system solves faster!

$$\boldsymbol{H}_\lambda \triangle\lambda = -\boldsymbol{g}_\lambda$$

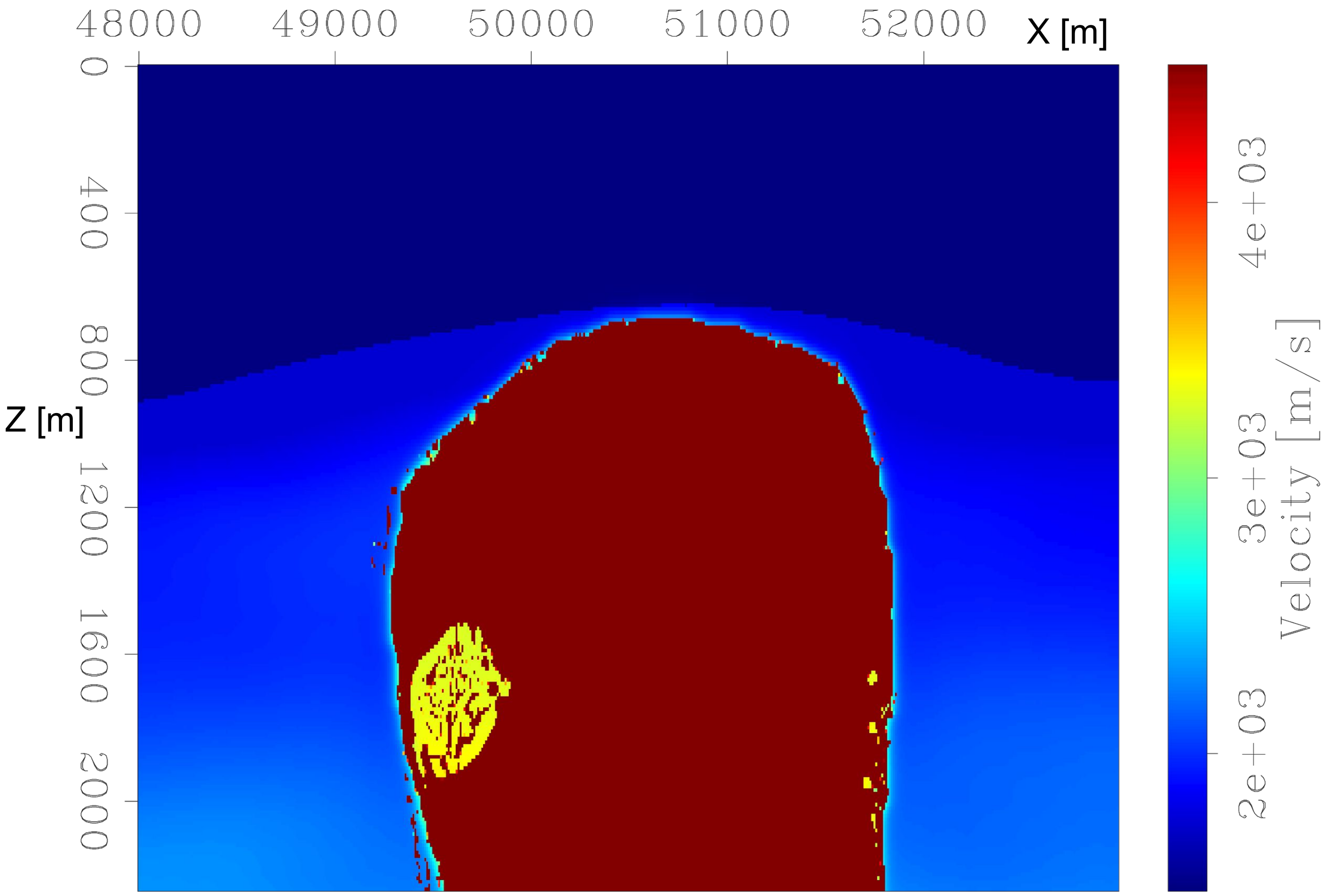
INITIAL MODEL



TRUE MODEL

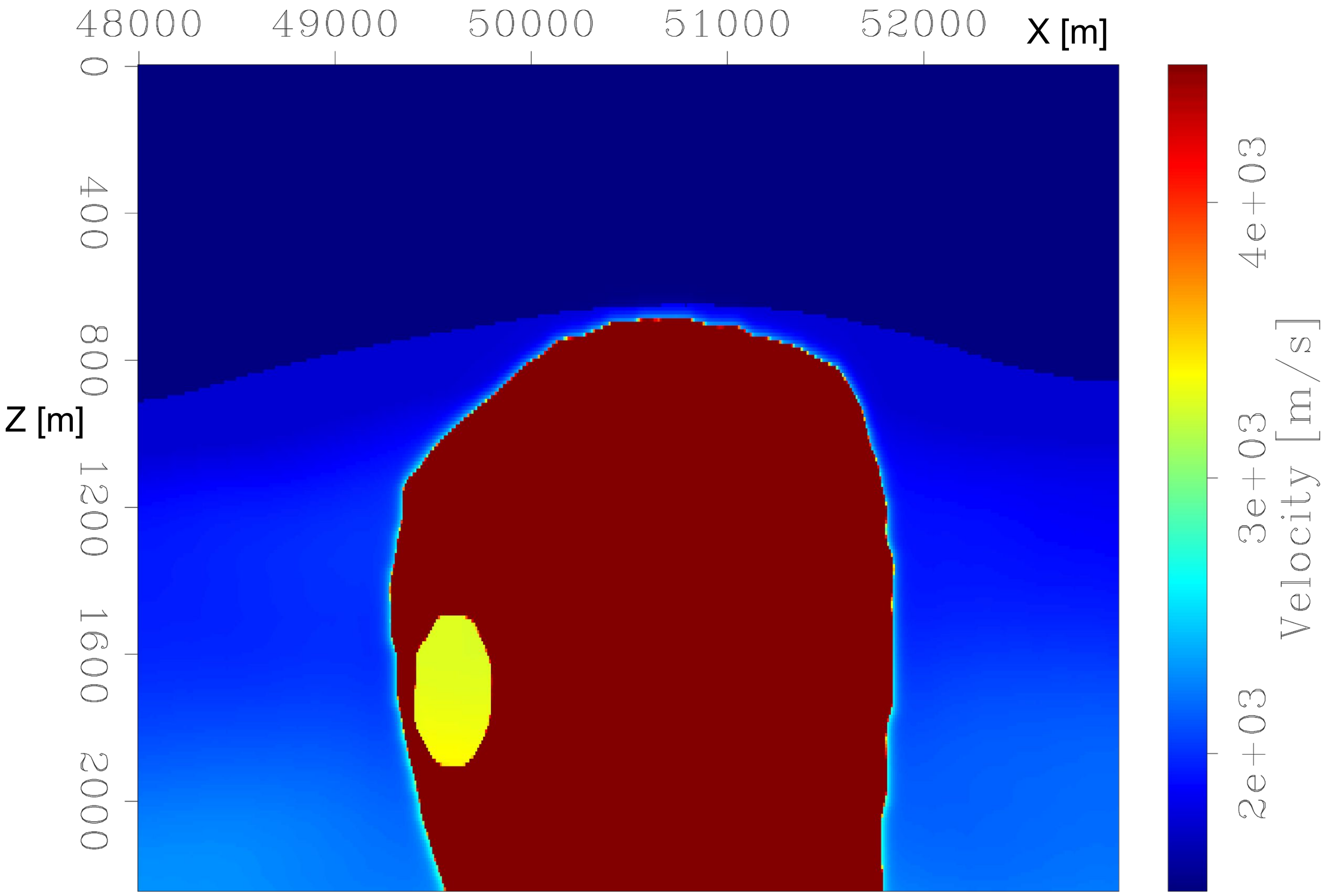

48000
49000
50000
51000
52000
X [m]
0
400
800
1200
1600
2000
Z [m]
2e+03
3e+03
4e+03
Velocity [m/s]

INVERTED MODEL it=30



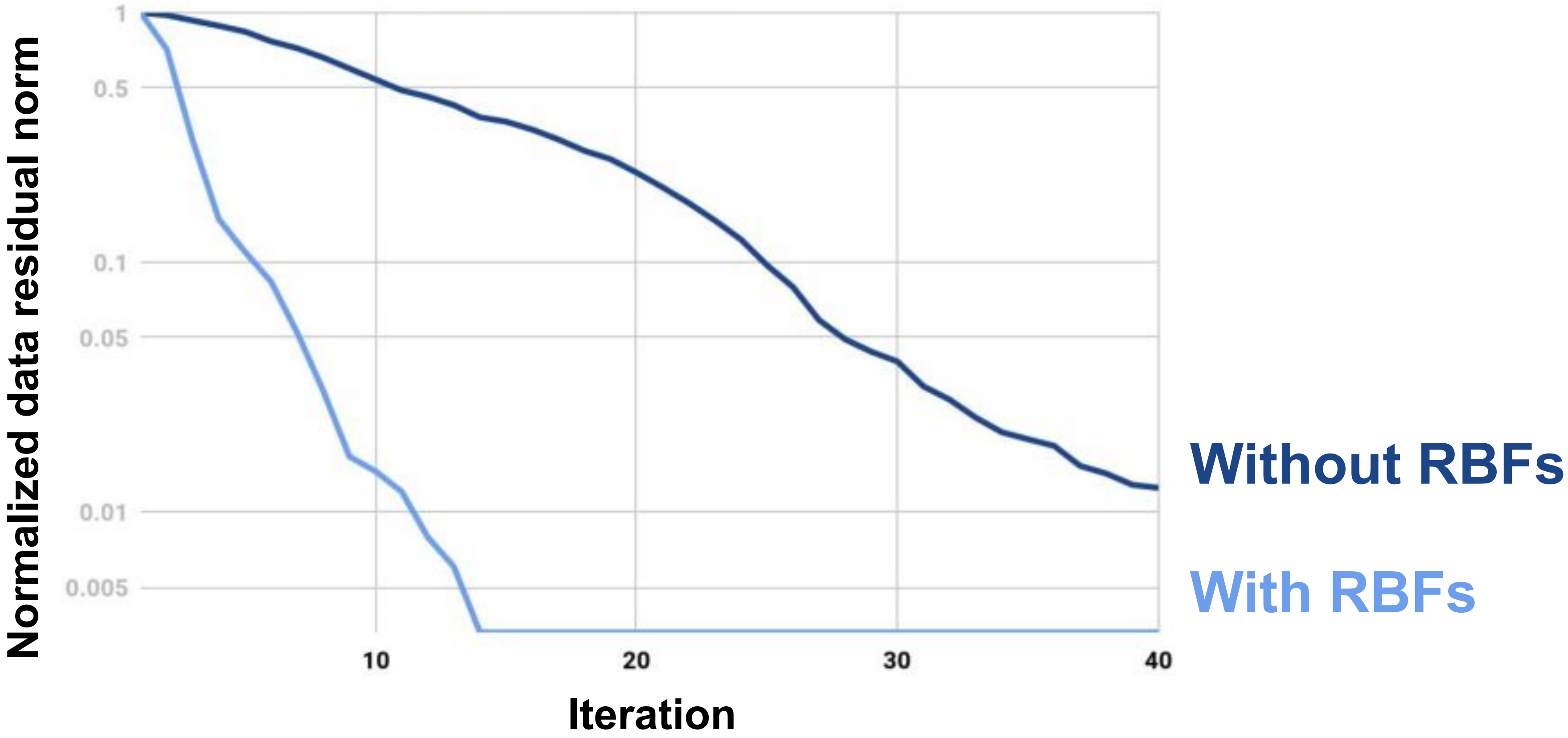
Without RBFs

INVERTED MODEL it=30

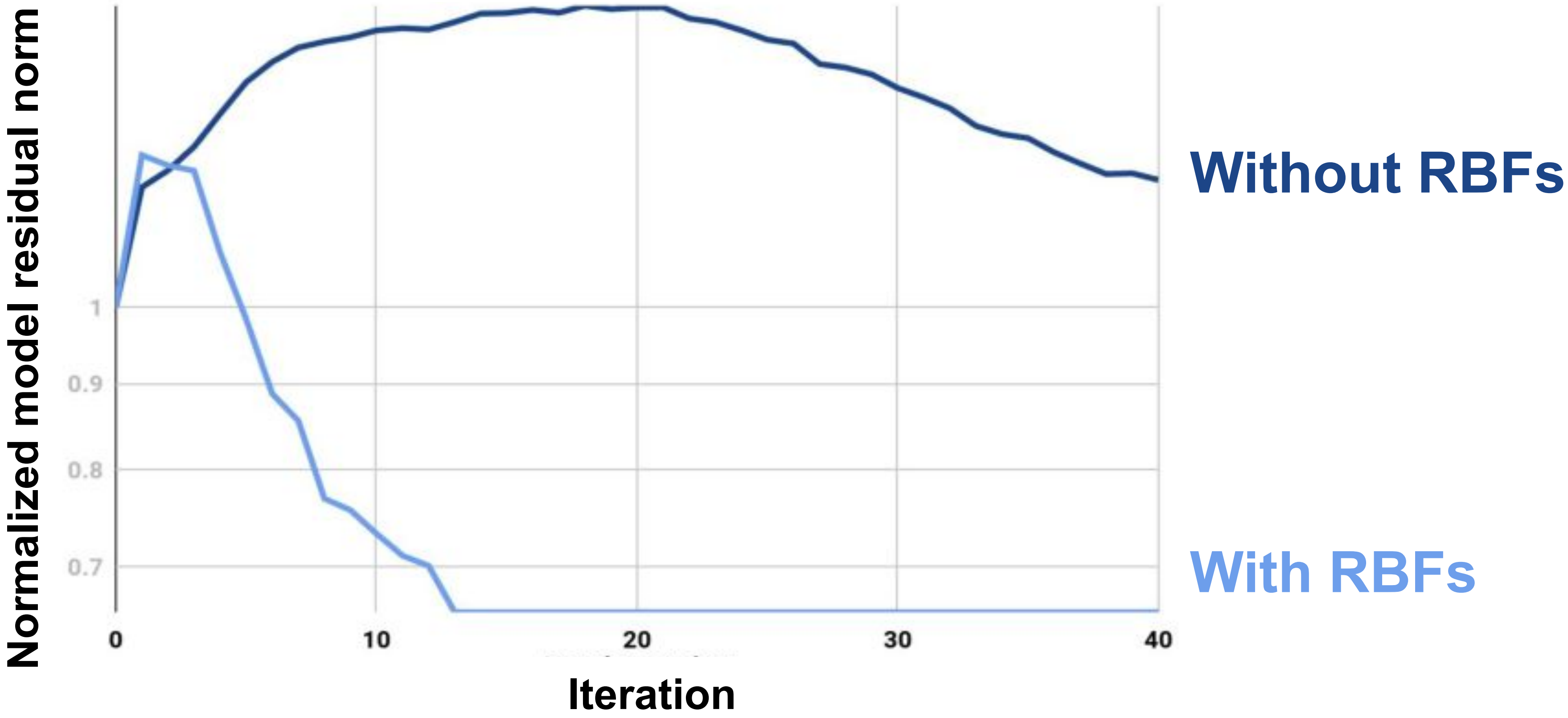


With RBFs

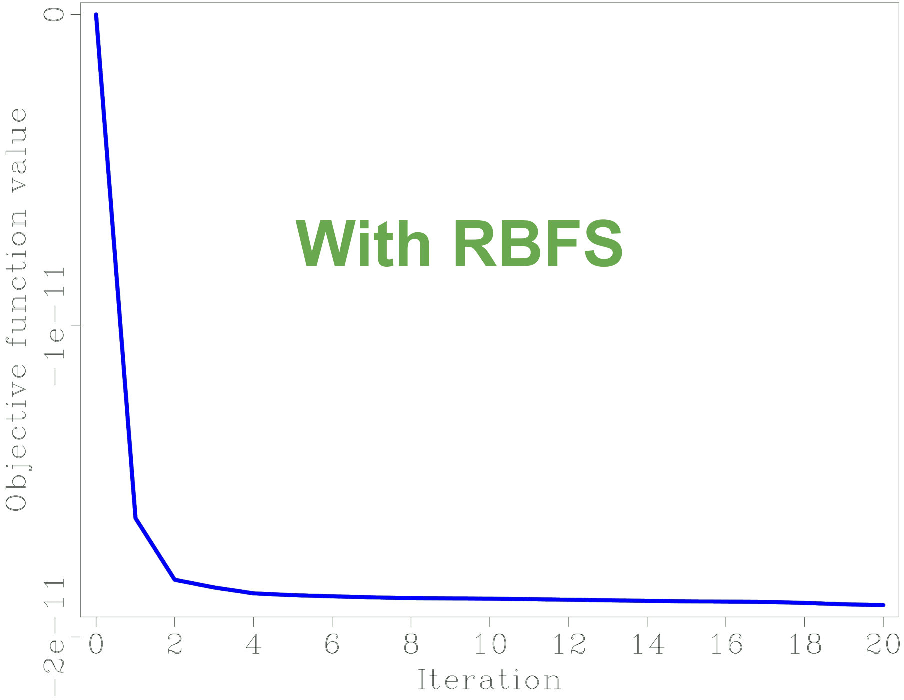
DATA RESIDUAL NORM

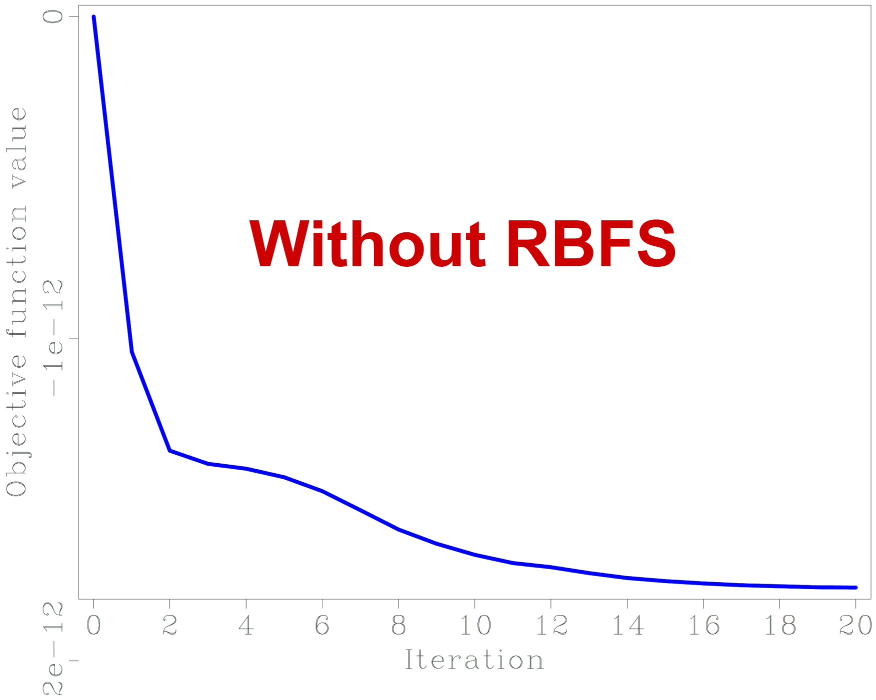


MODEL RESIDUAL NORM

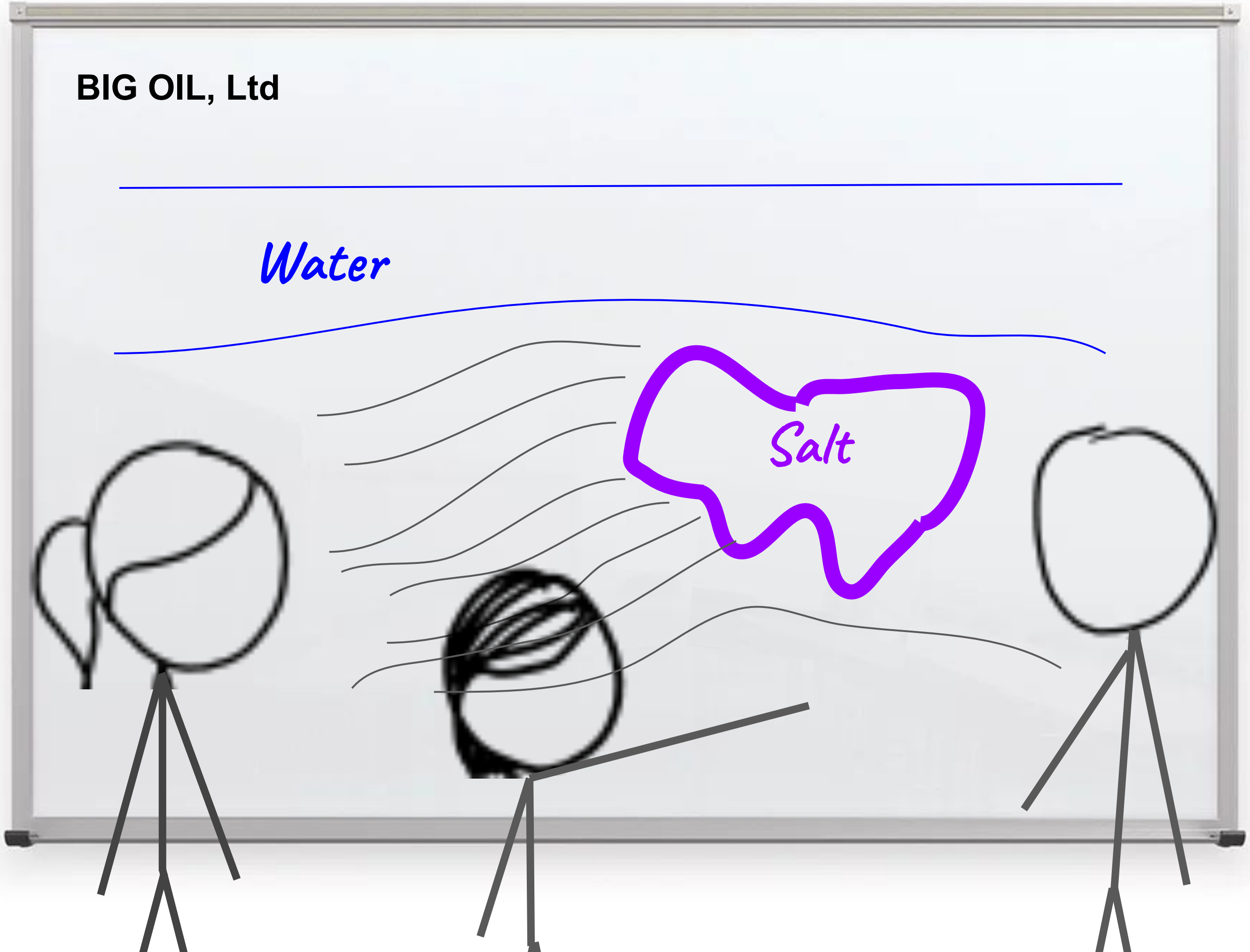


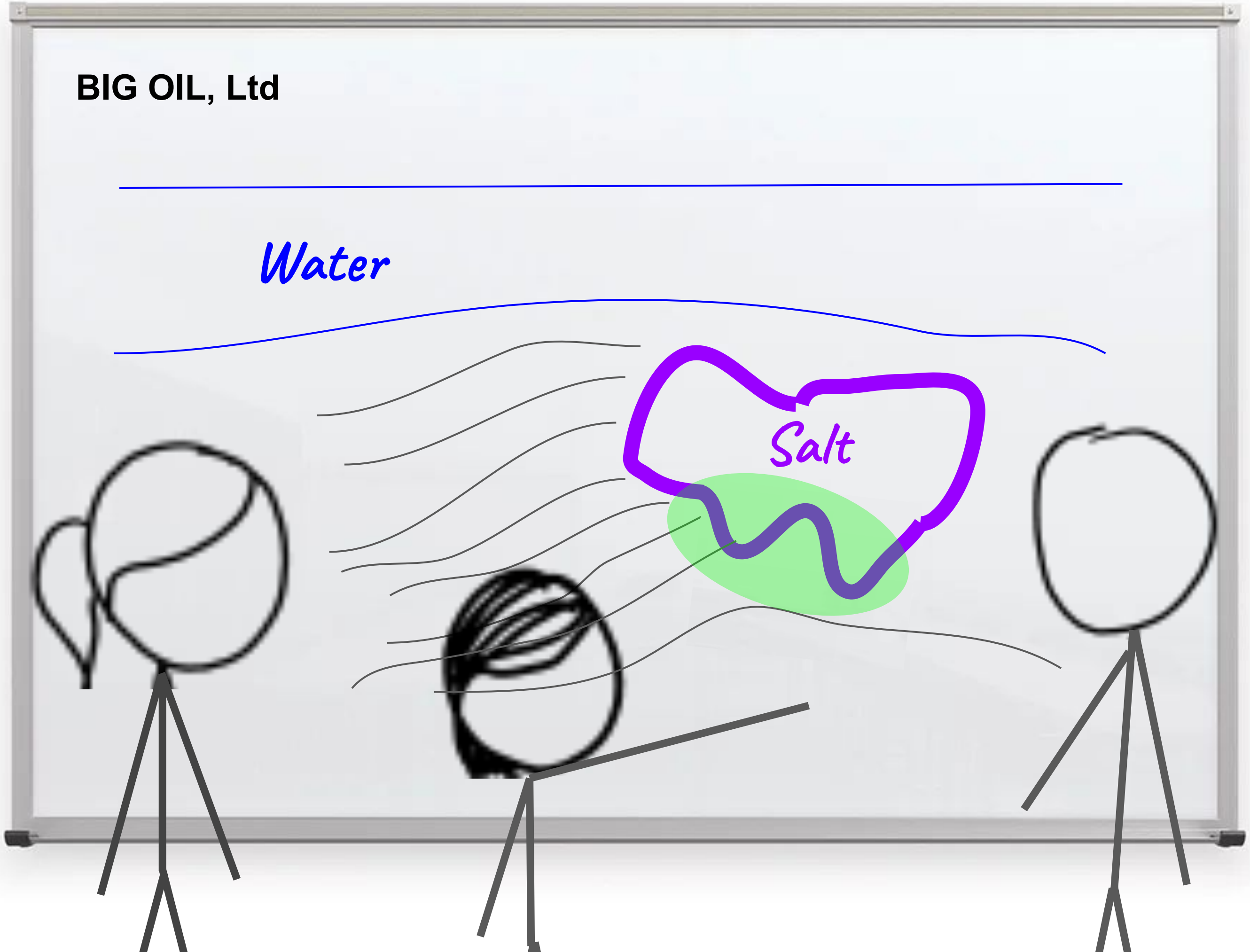
Search direction inversion is better! $\triangle m = -H^{-1}g$

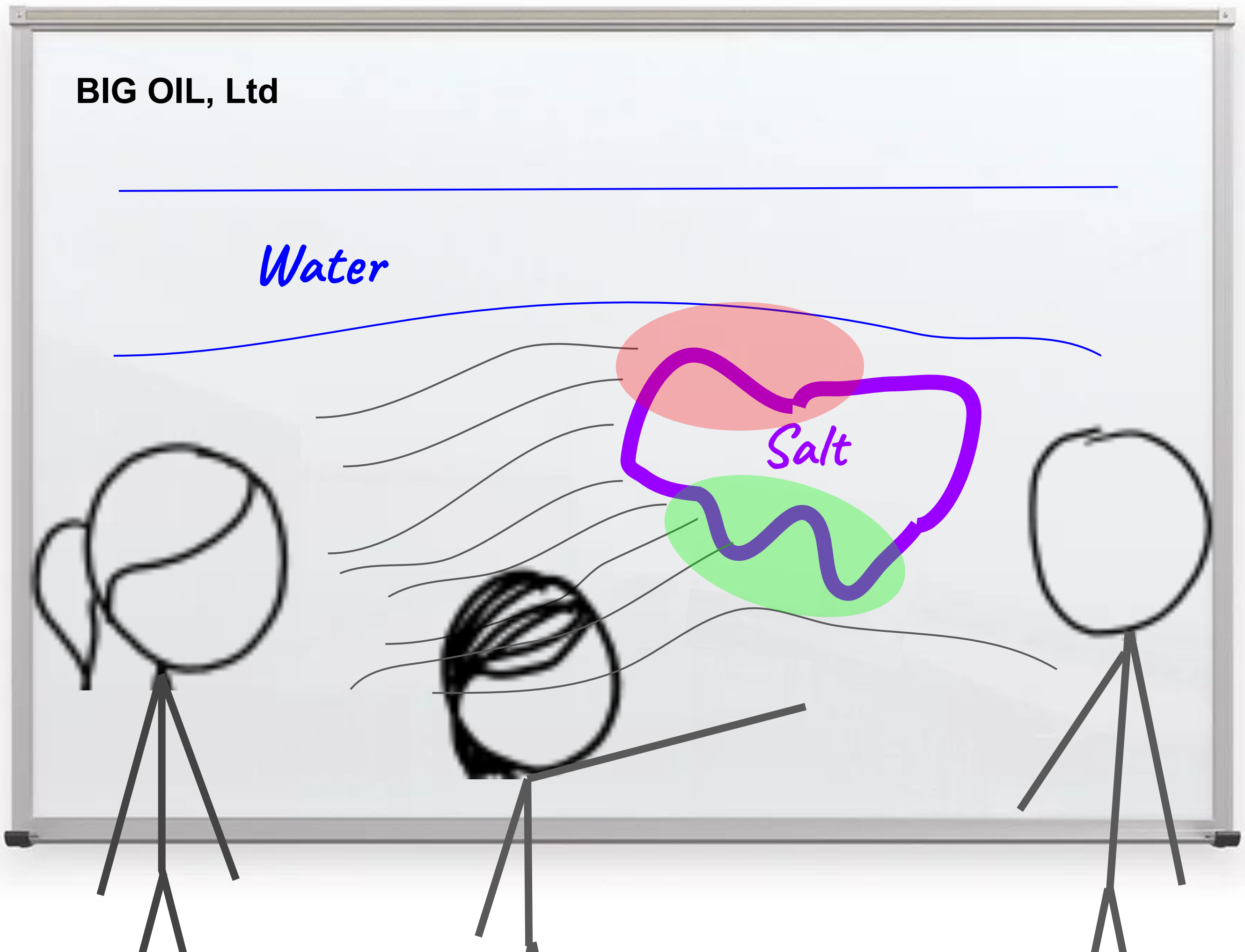


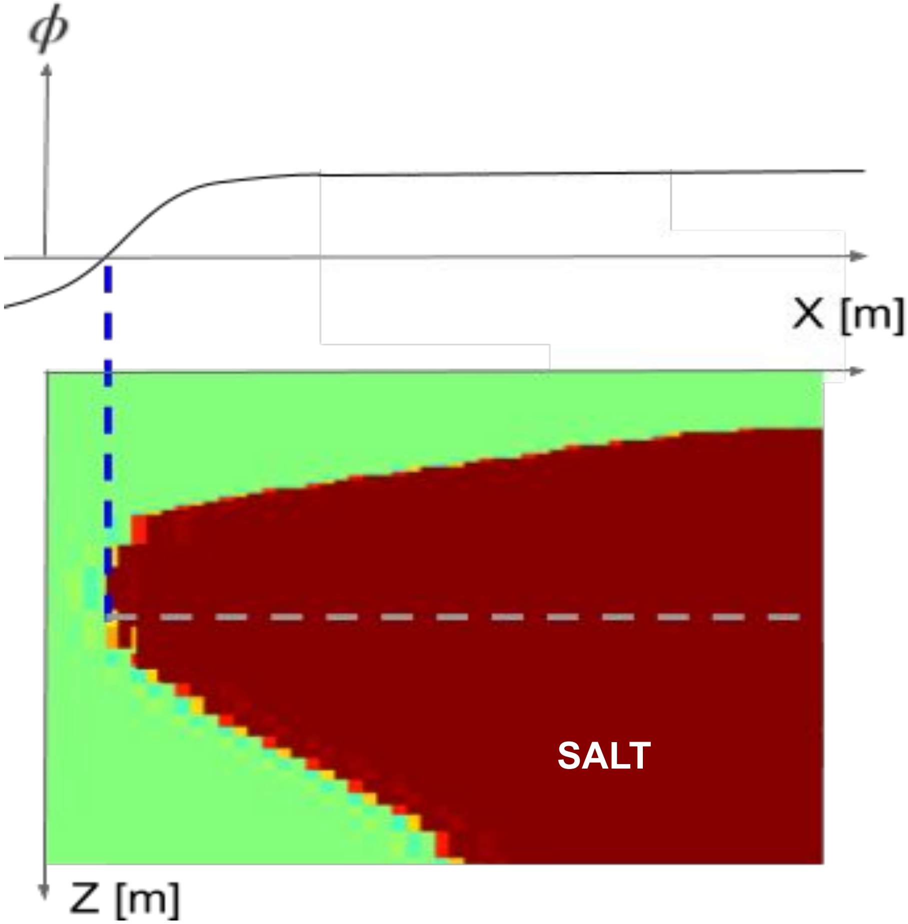


Is there any way we can include human input into our inversion?


BIG OIL, Ltd
Water
Salt

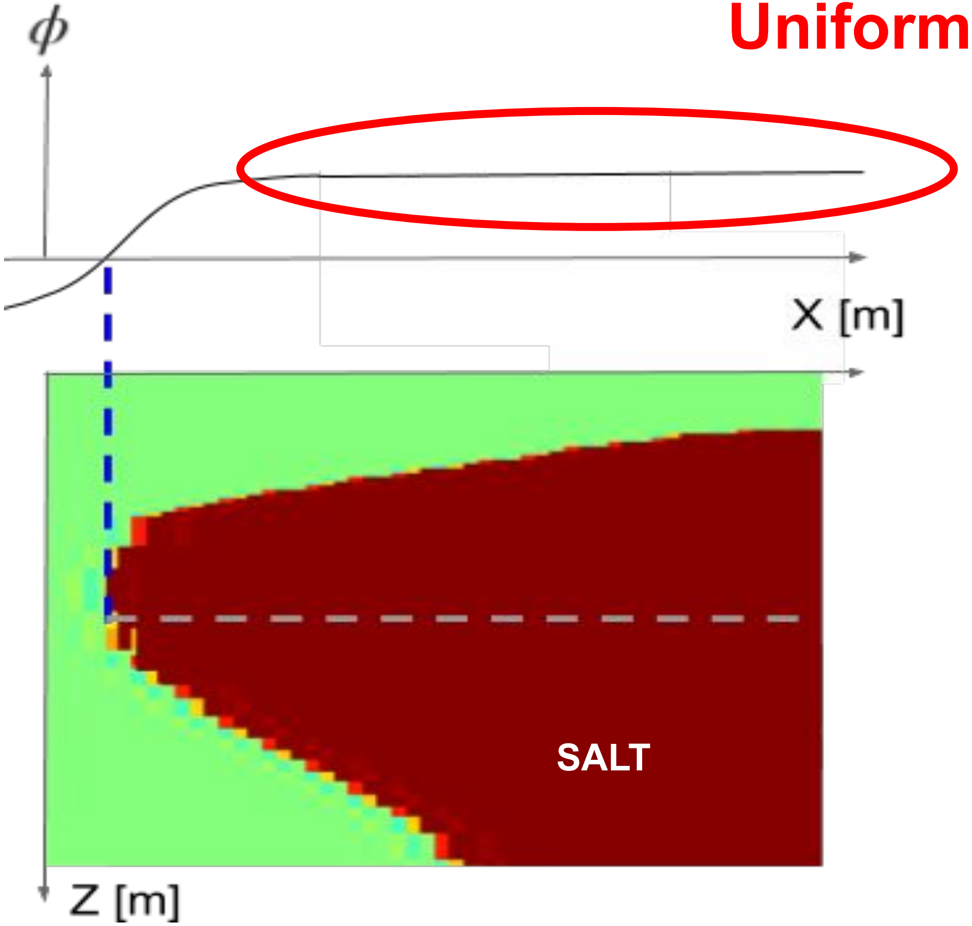

BIG OIL, Ltd
Water
Salt

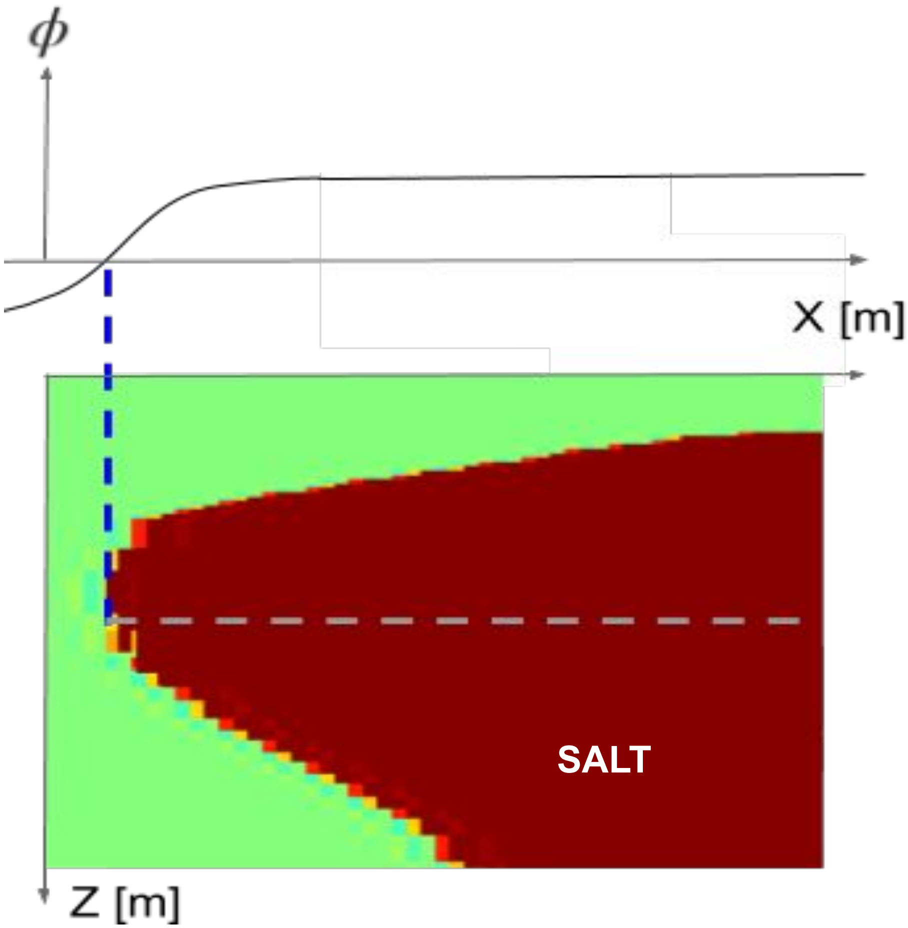

BIG OIL, Ltd
Water
Salt



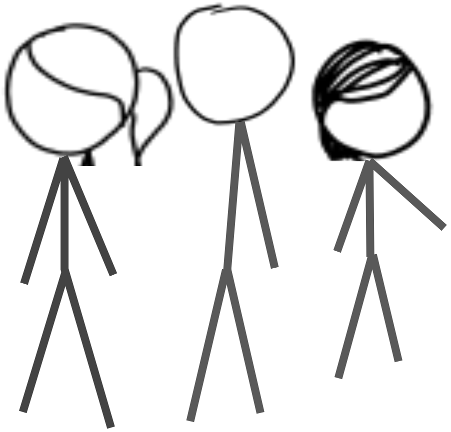
ϕ
X [m]
SALT
Z [m]

Uniform sensitivity

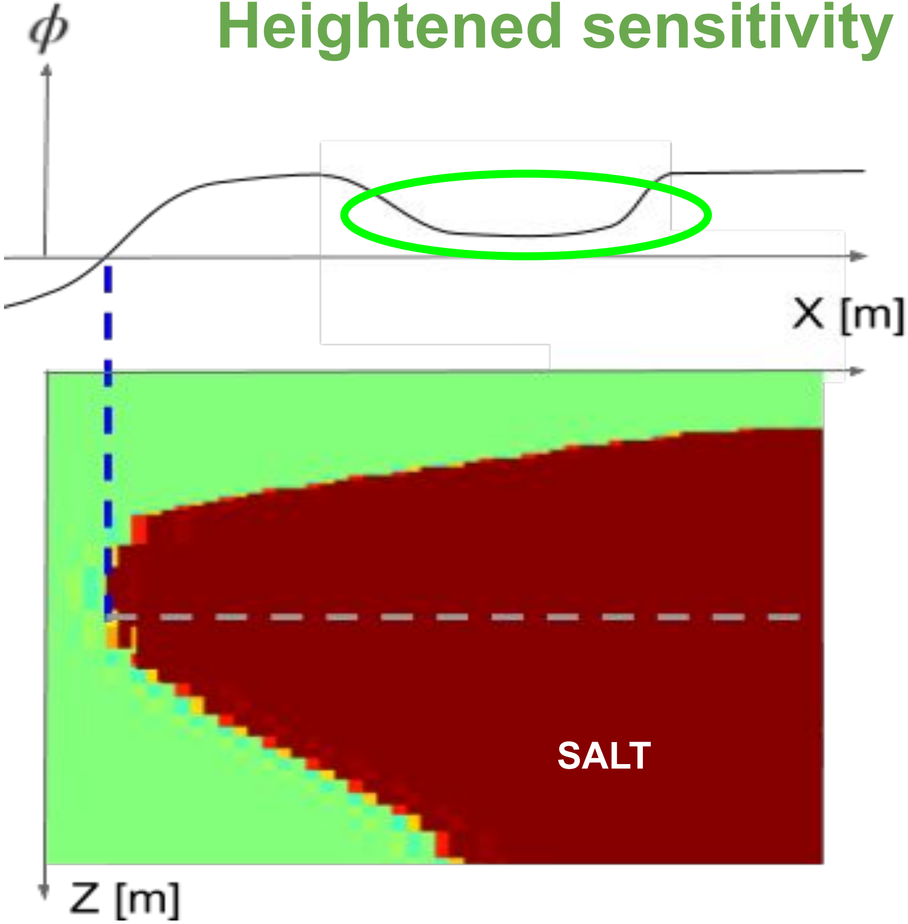


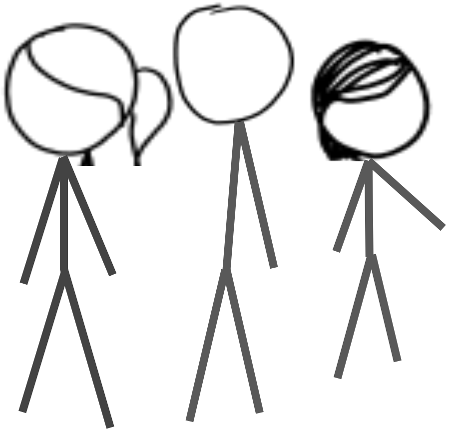


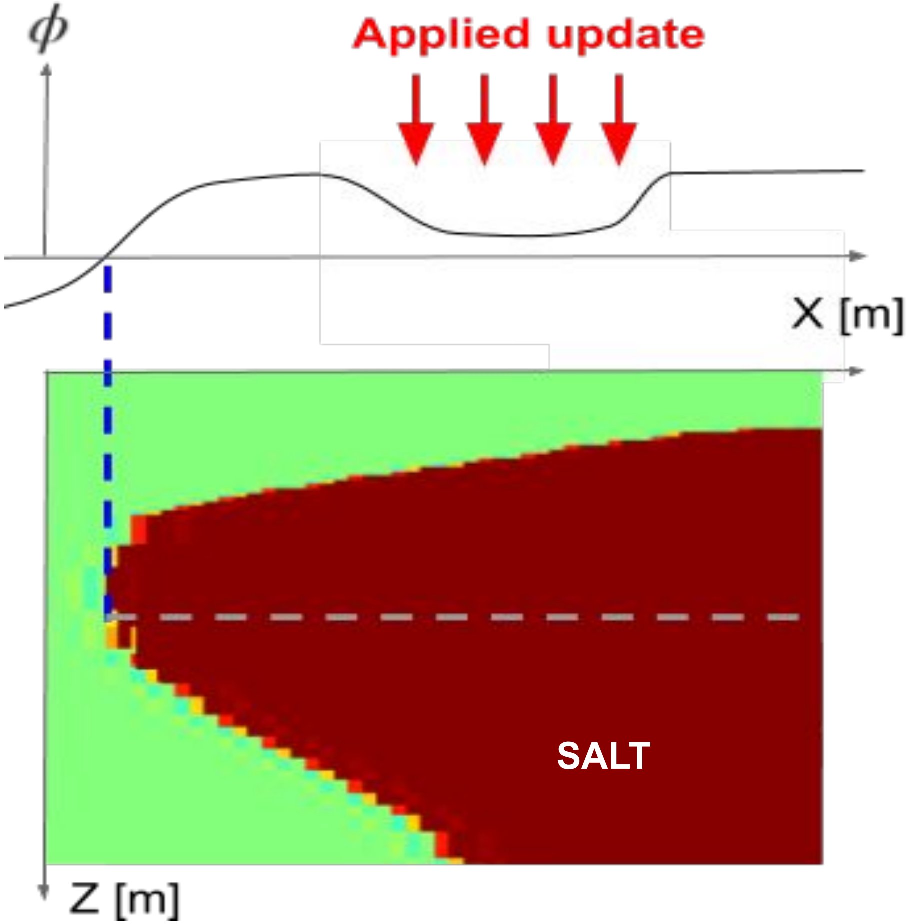
ϕ
X [m]
SALT
Z [m]



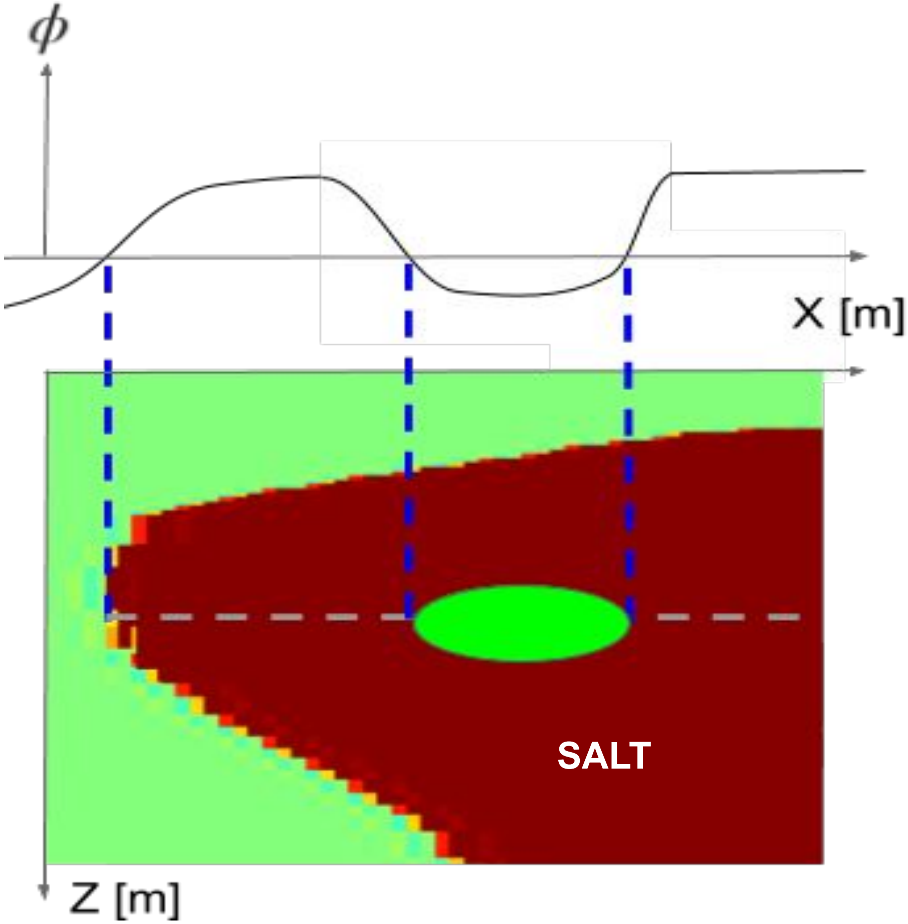
Heightened sensitivity



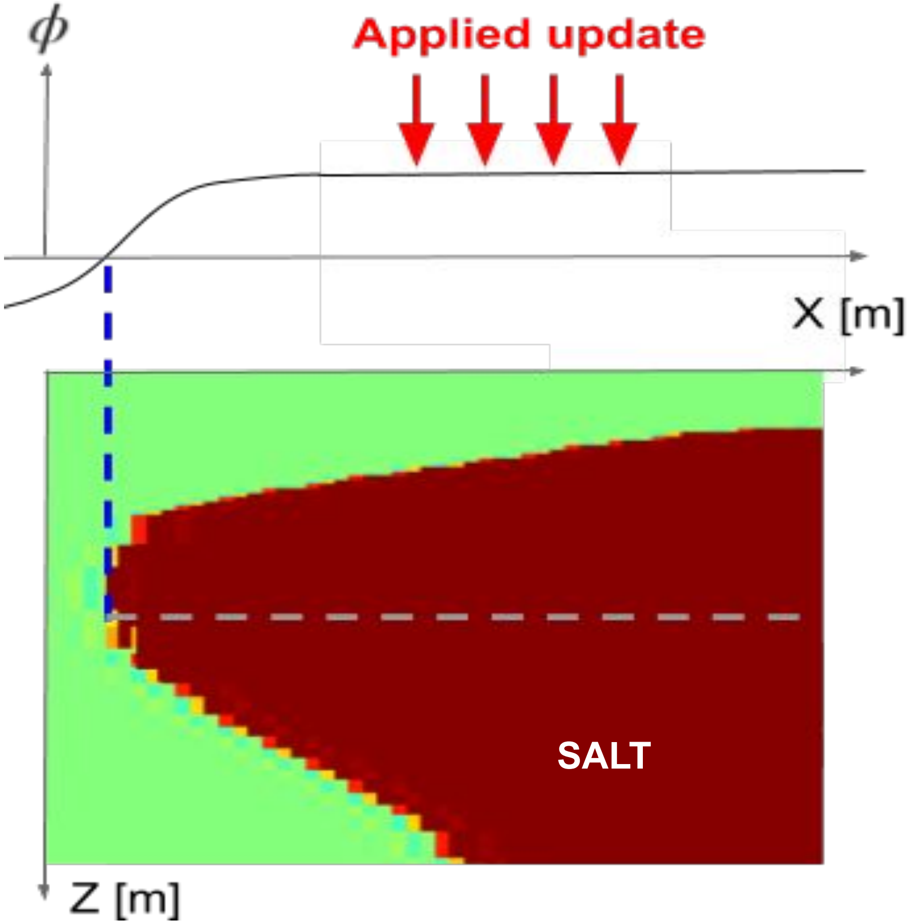




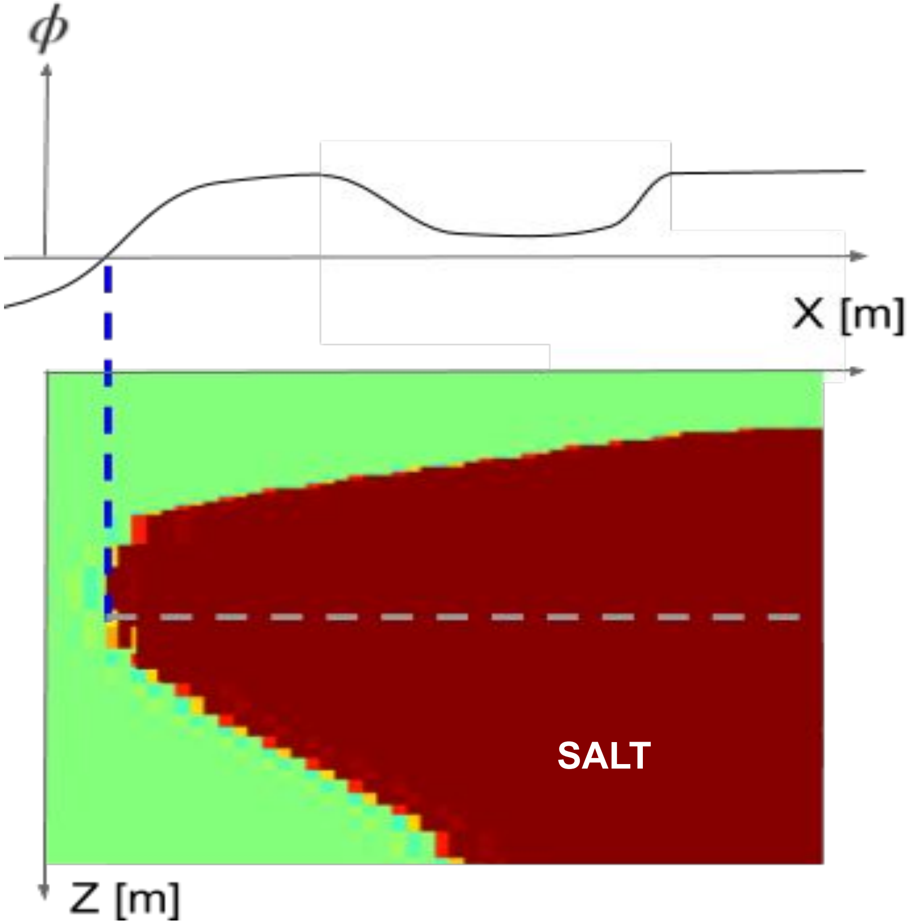
ϕ
Applied update
X [m]
SALT
Z [m]



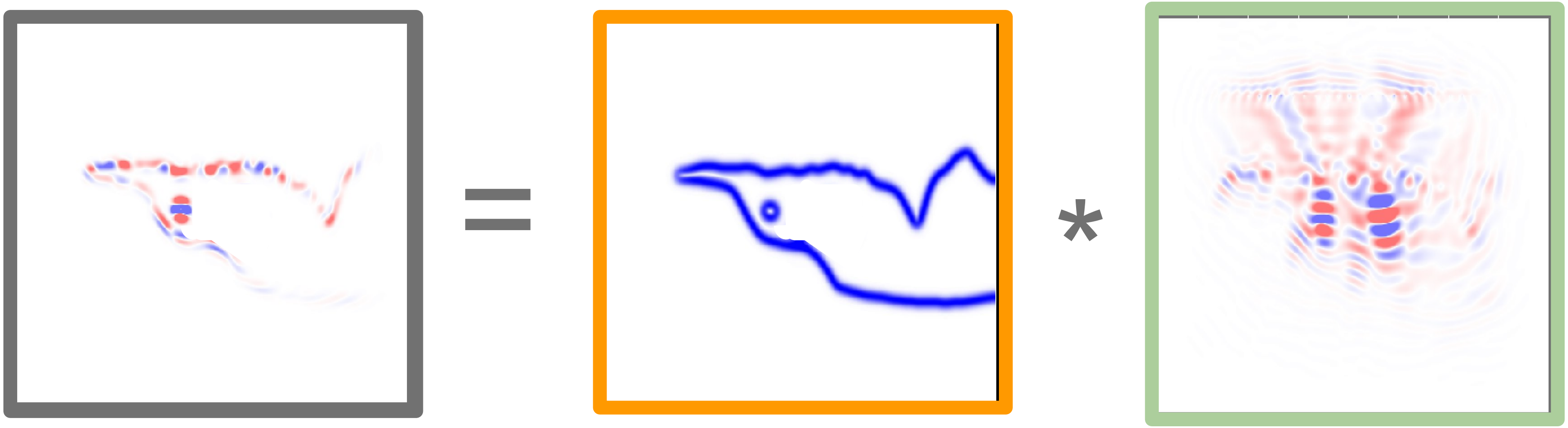
ϕ
X [m]
SALT
Z [m]



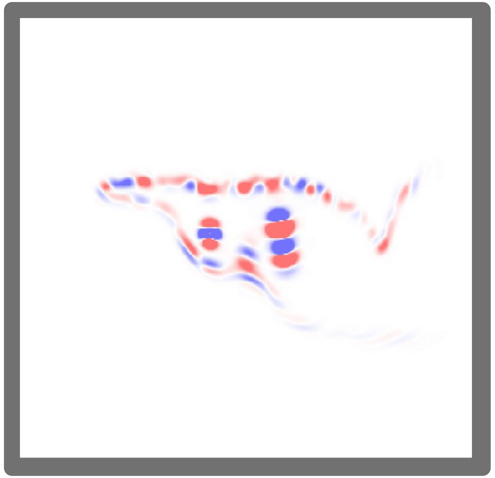
ϕ
Applied update
X [m]
SALT
Z [m]


ϕ
X [m]
SALT
Z [m]

Wait …. can we make changes inside the salt, or only along the boundaries?

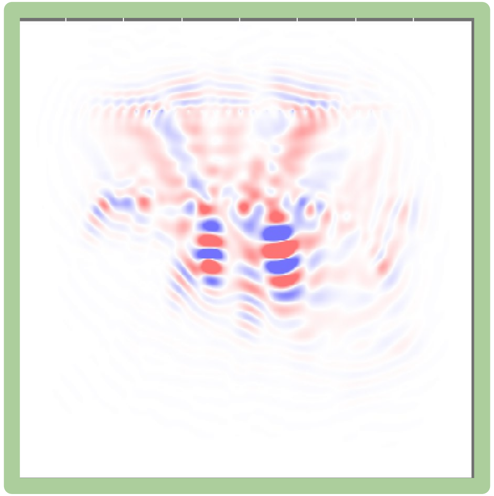


$$\delta(\phi)(c_s - b)$$



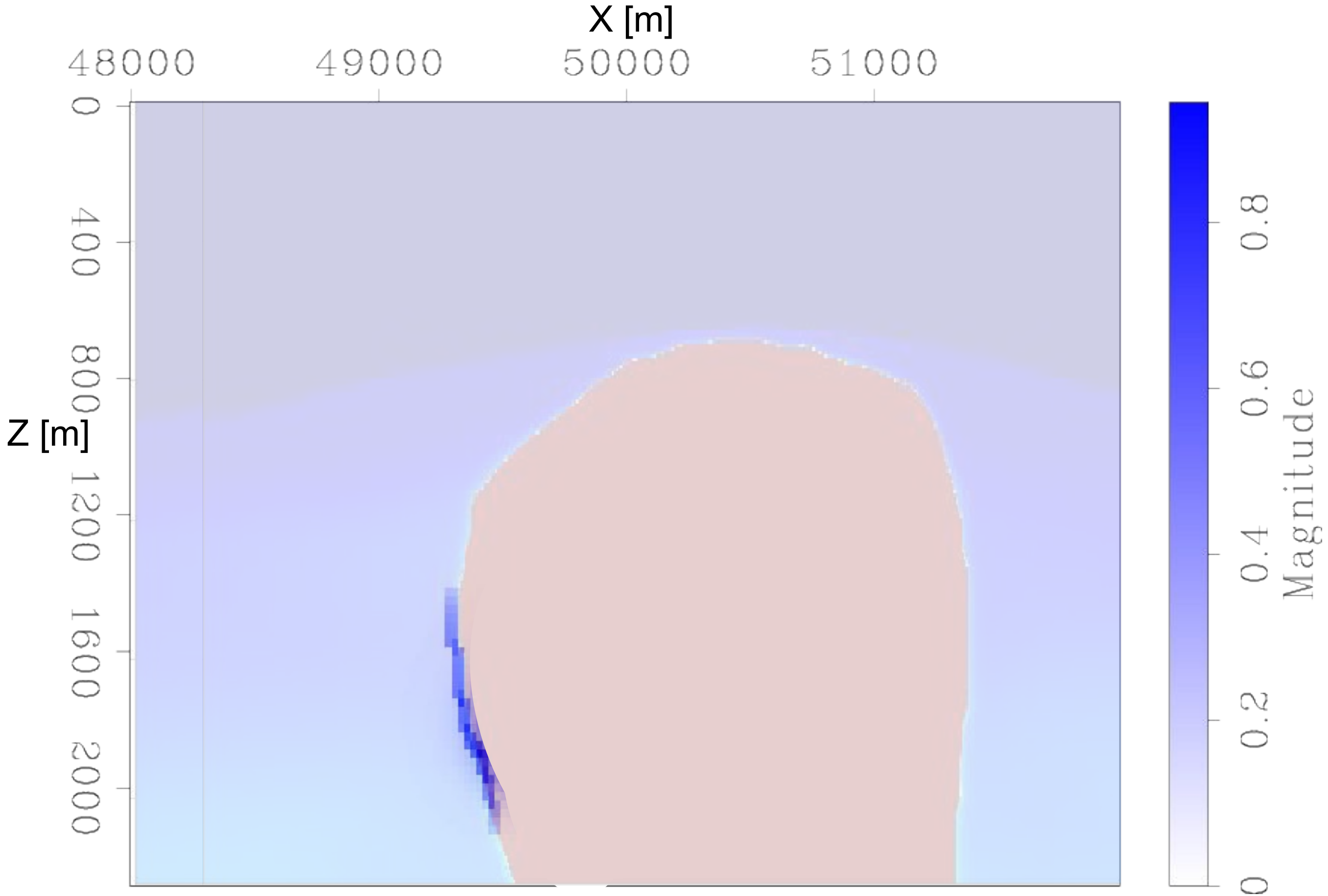
=

*

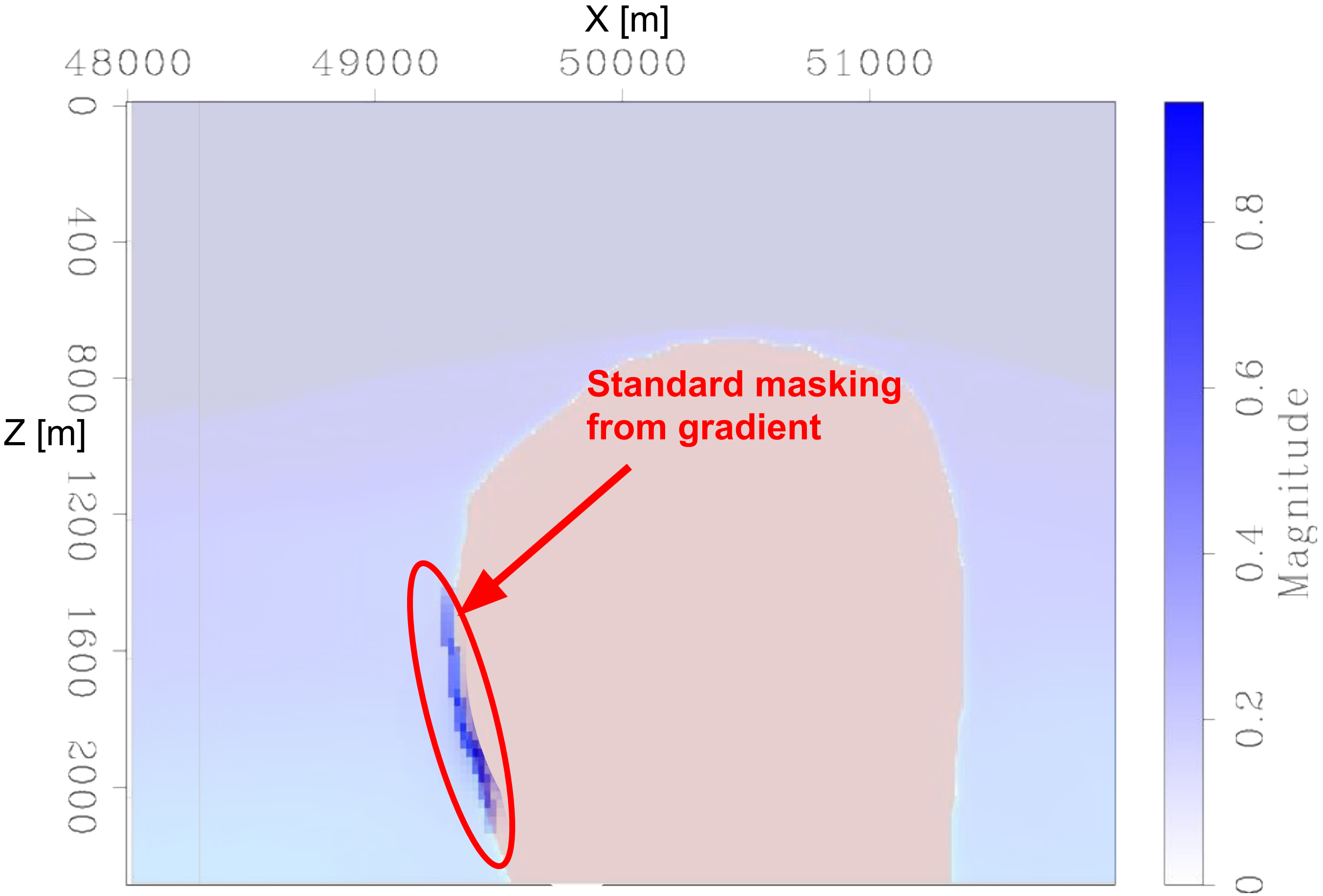


$$\hat{\delta}(\phi,G)(c_s-b)$$

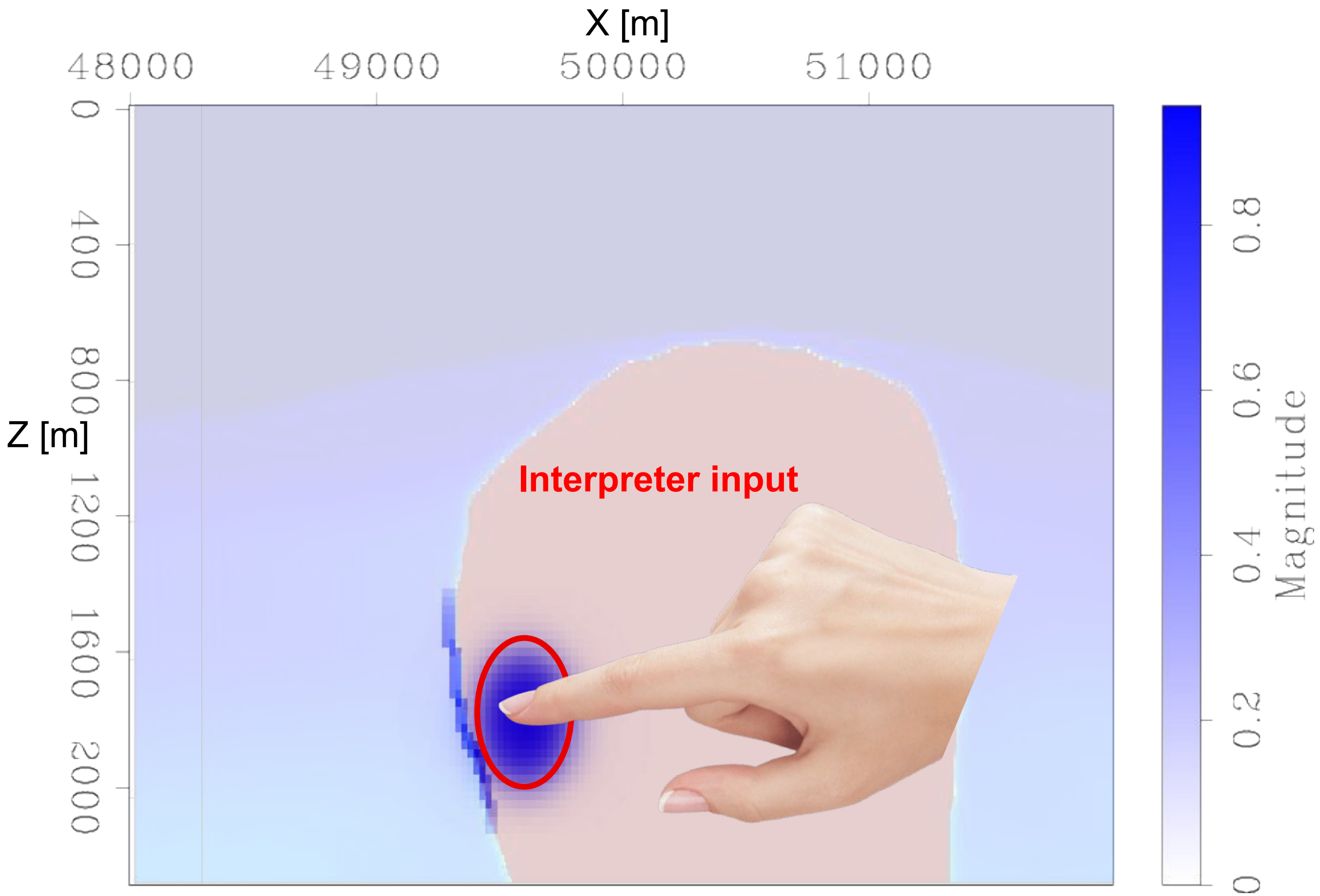
Expand the gradient footprint



Expand the gradient footprint

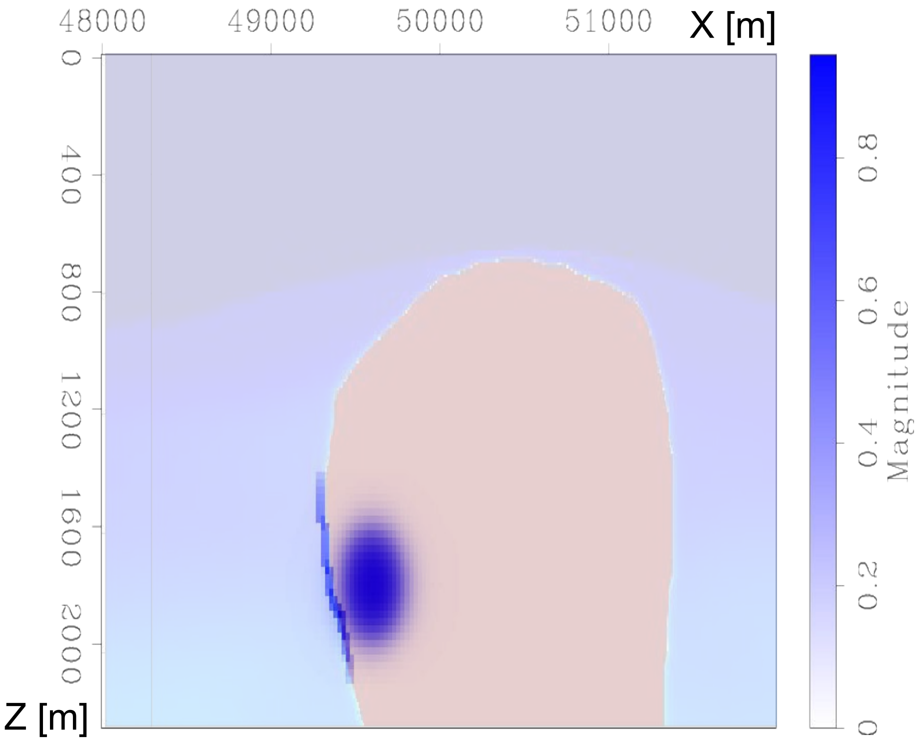


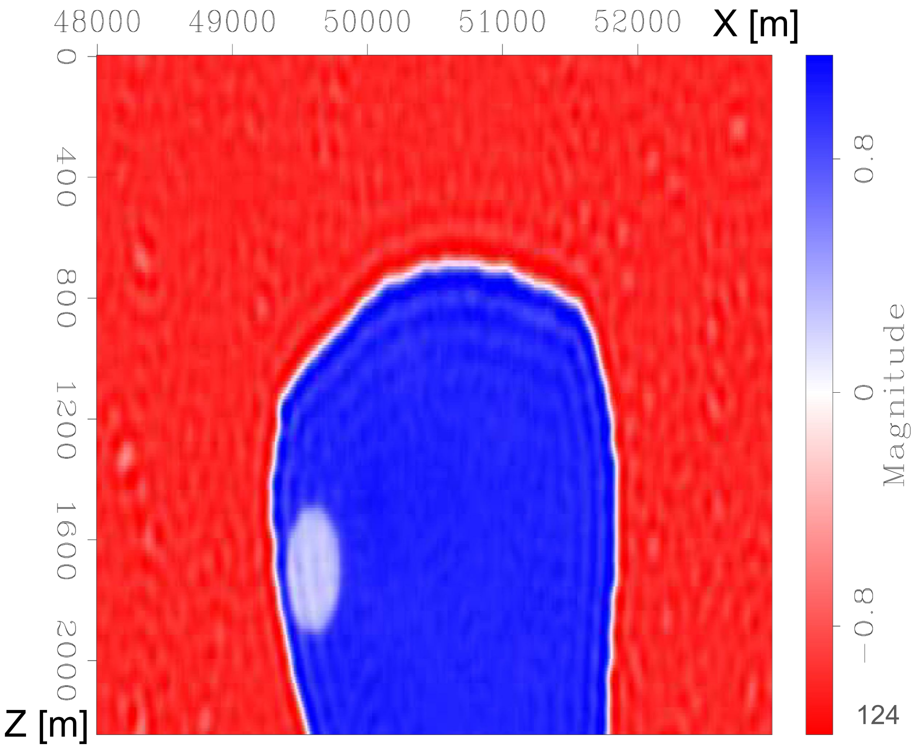
Expand the gradient footprint



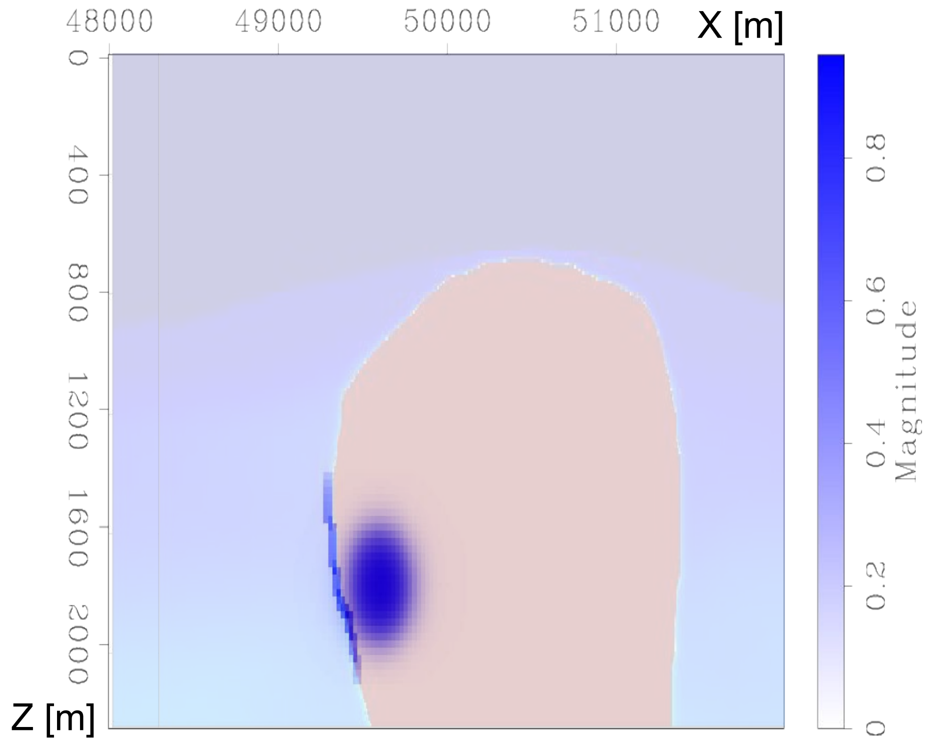
How much do expanded gradients and expert guidance **actually help**?

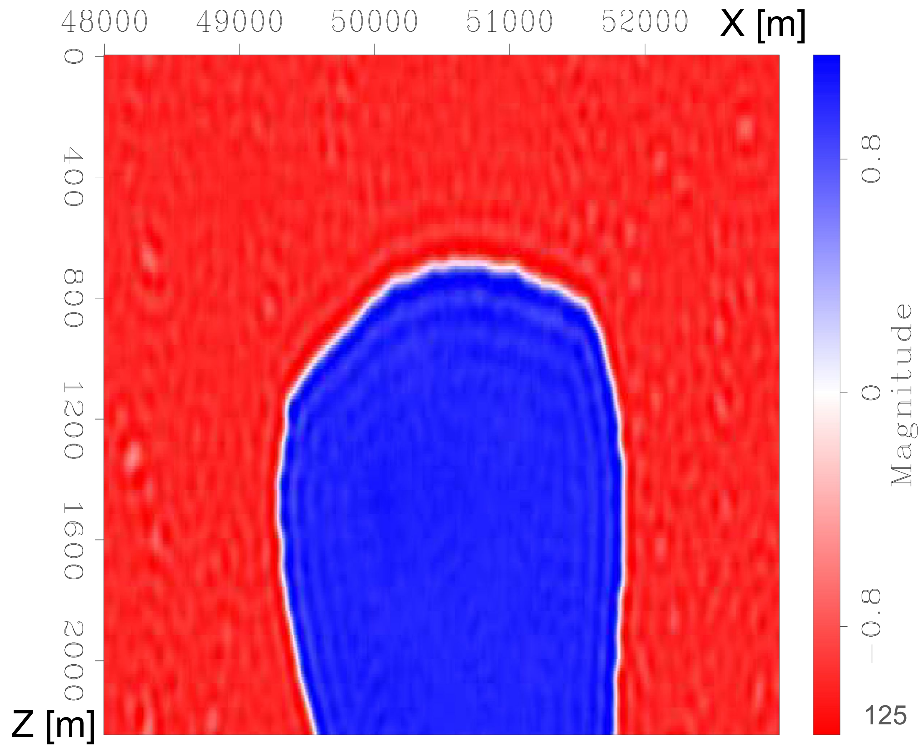
Fully guided inversion



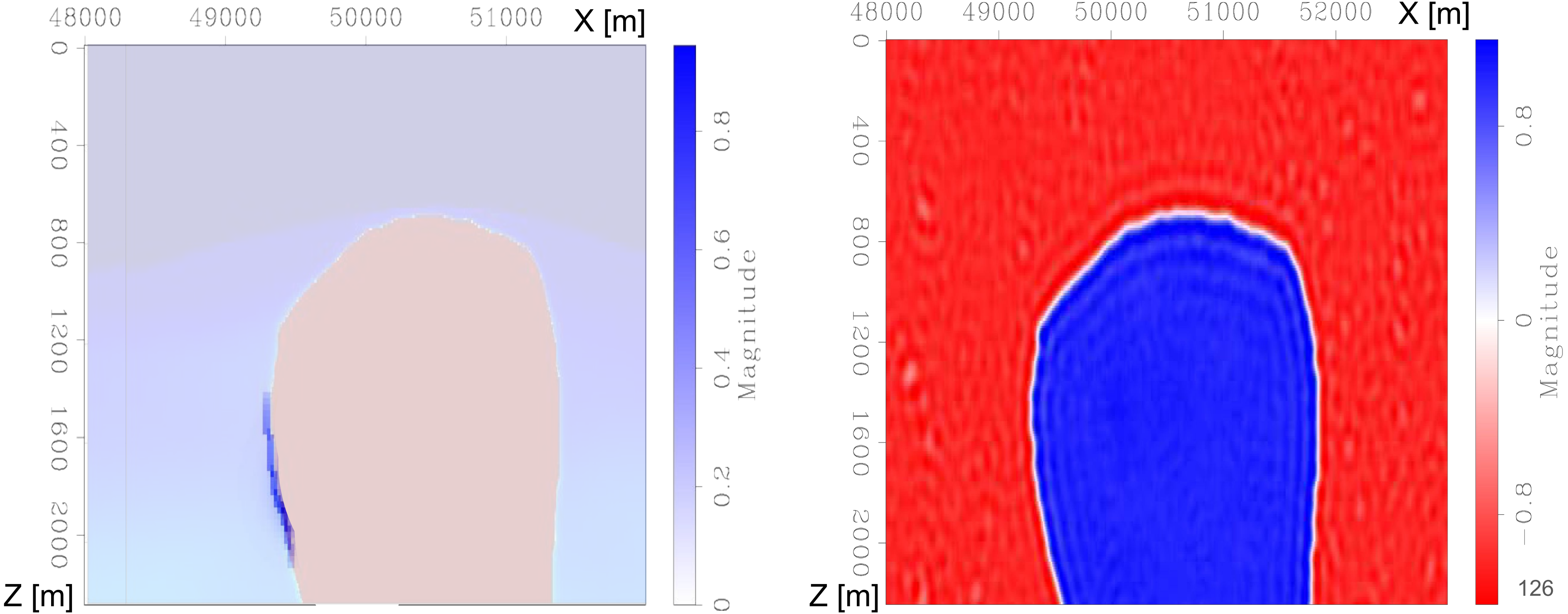


Partially guided inversion

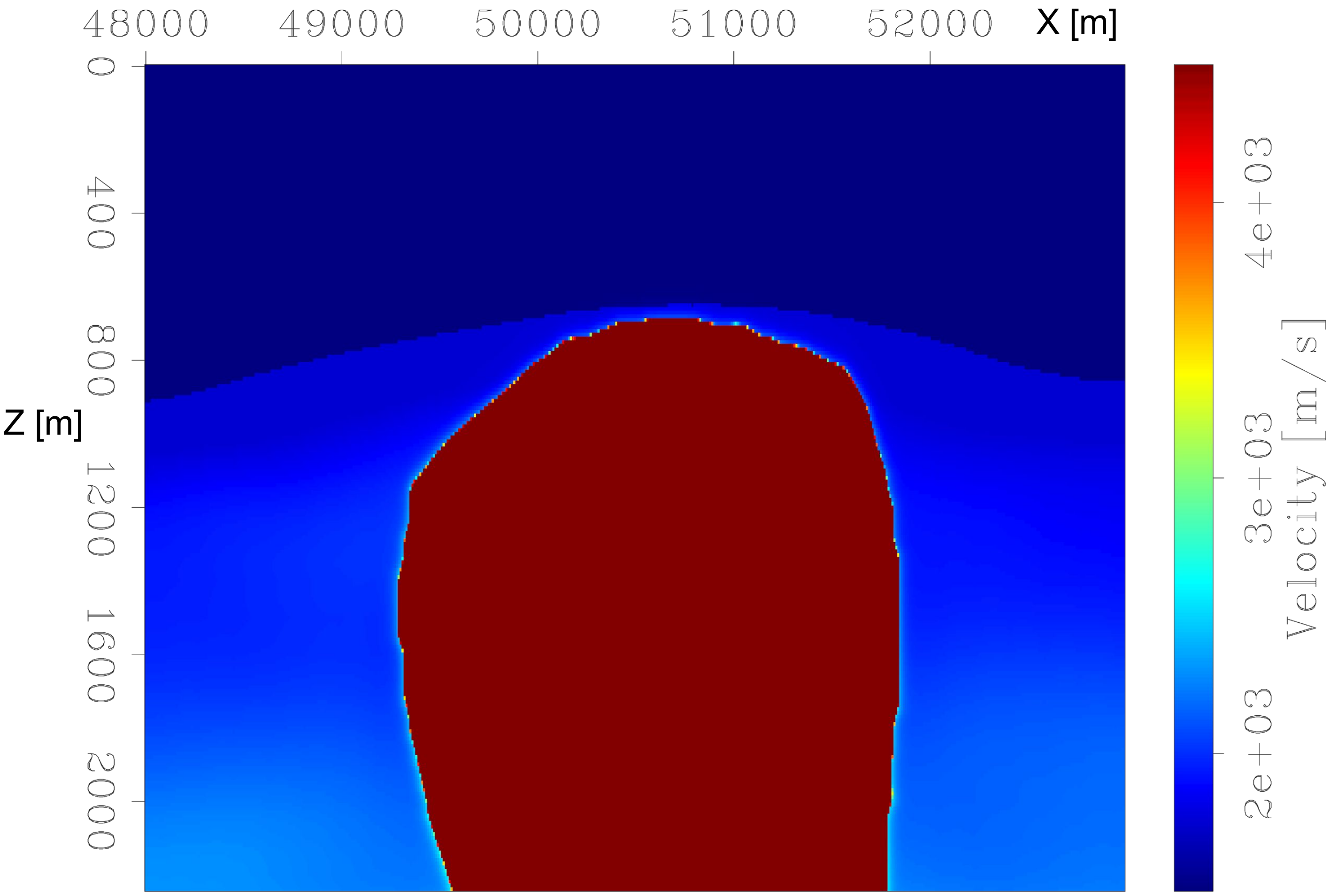




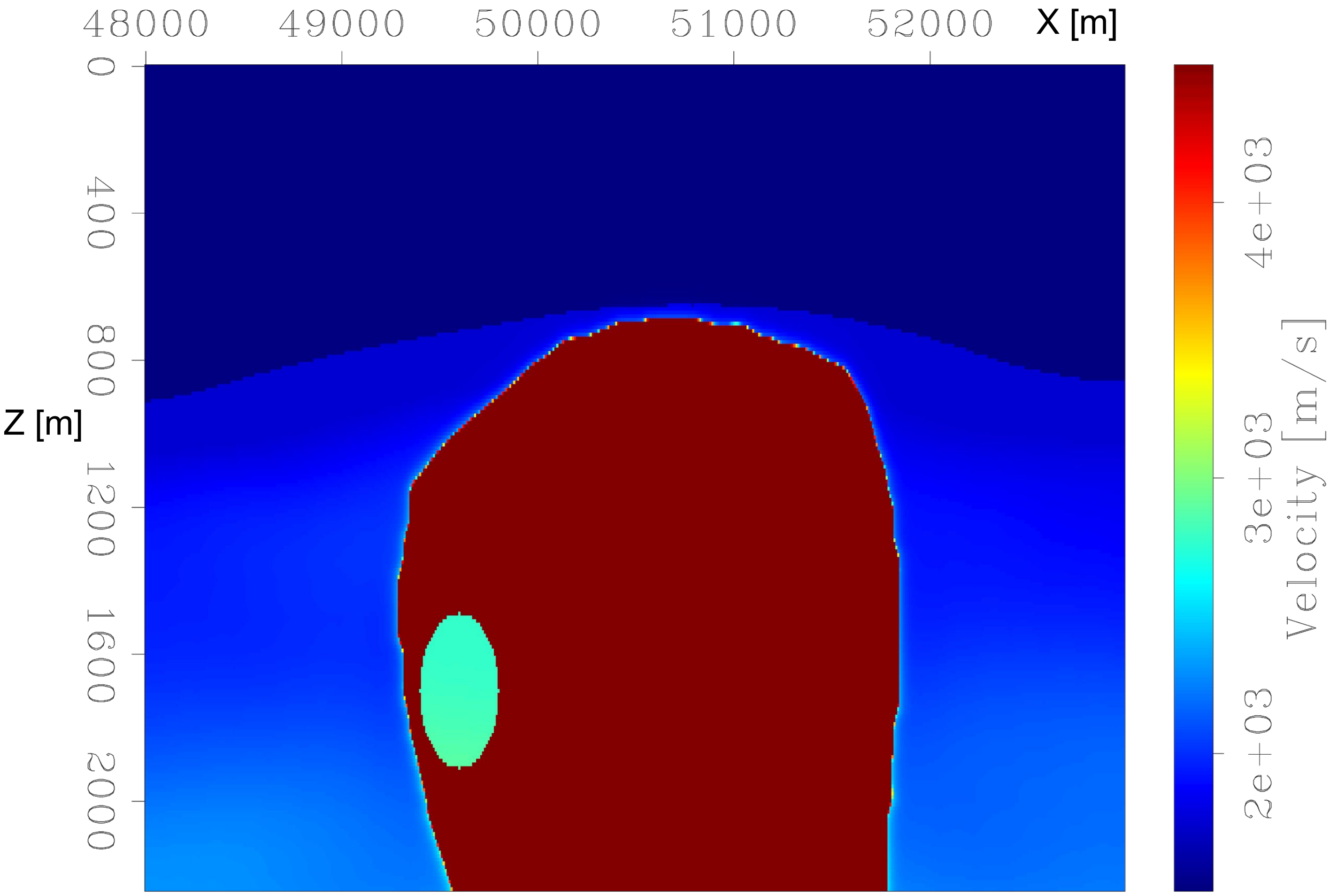
Unguided inversion



INITIAL MODEL

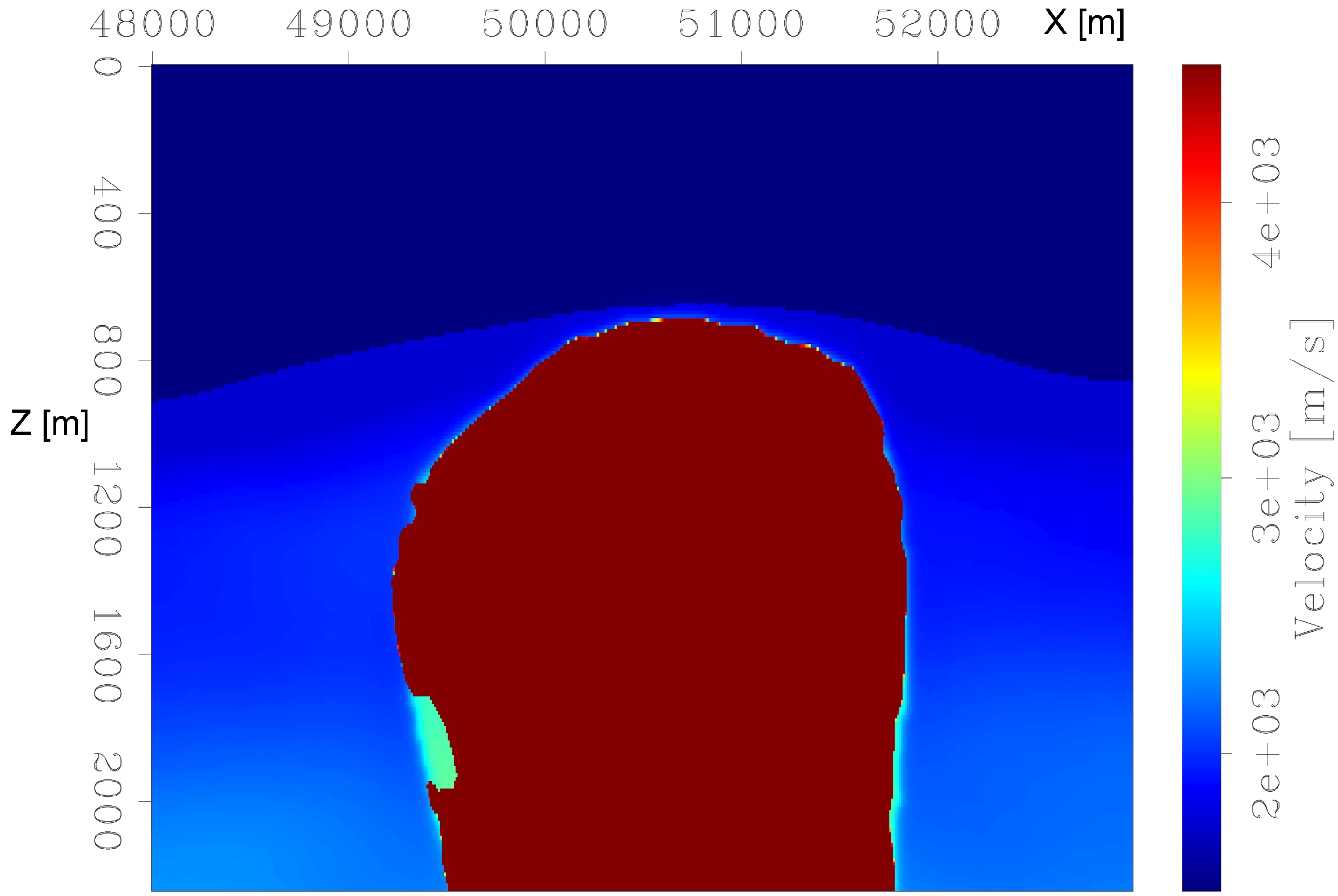


TRUE MODEL



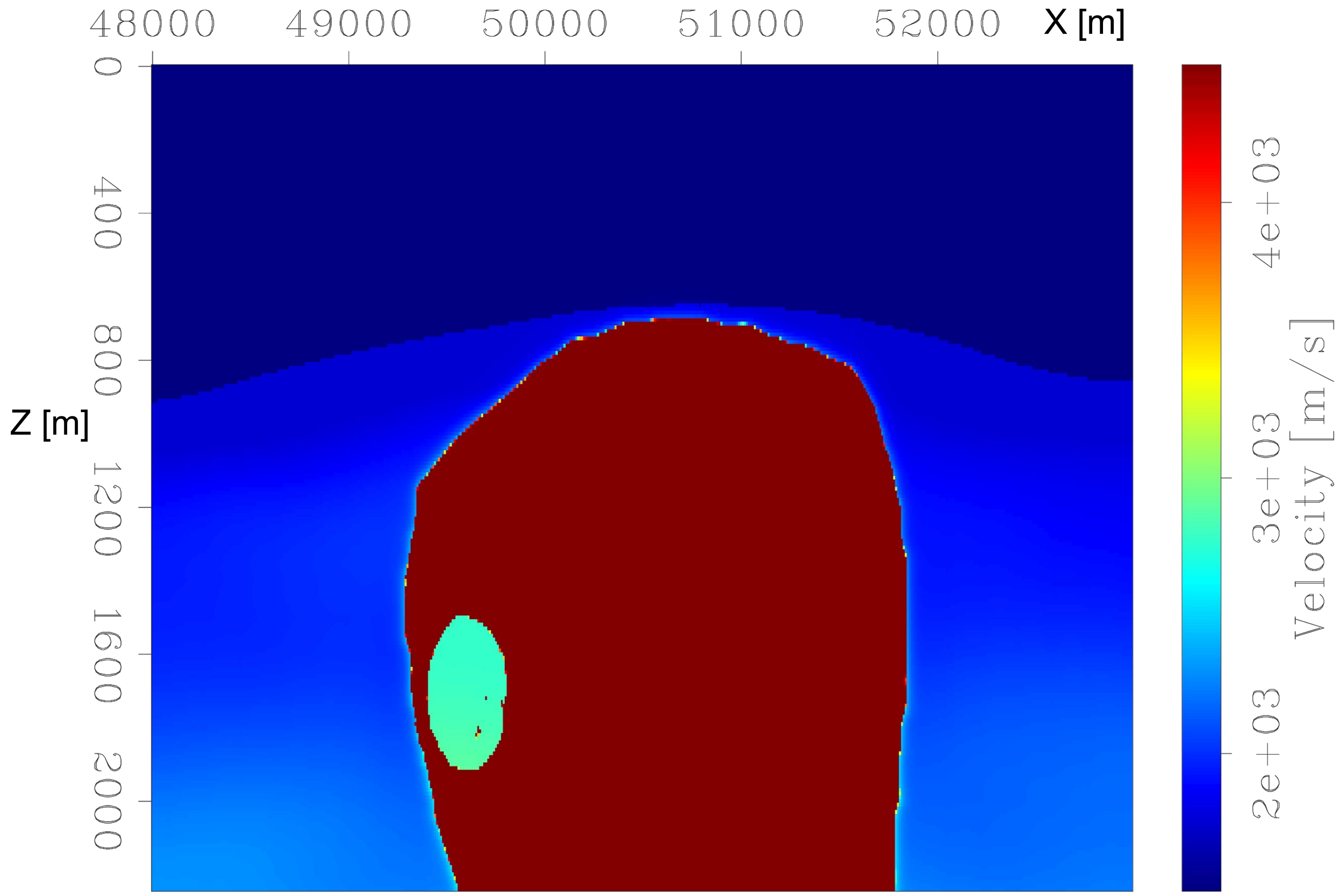
48000
49000
50000
51000
52000
X [m]
0
400
800
1200
1600
2000
Z [m]
2e+03
3e+03
4e+03
Velocity [m/s]

UNGUIDED Inversion result



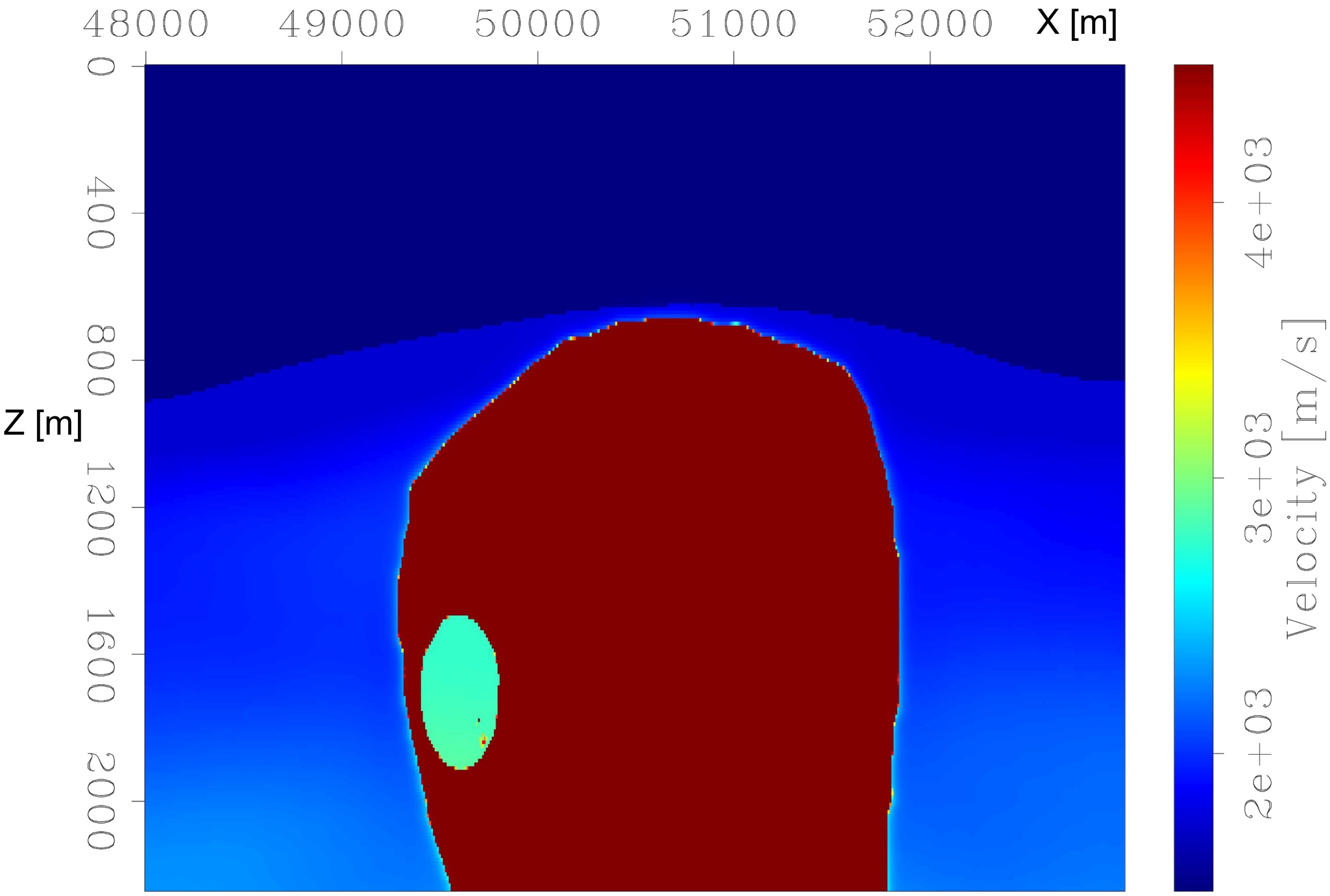
Iteration =15

PARTIALLY GUIDED Inversion result



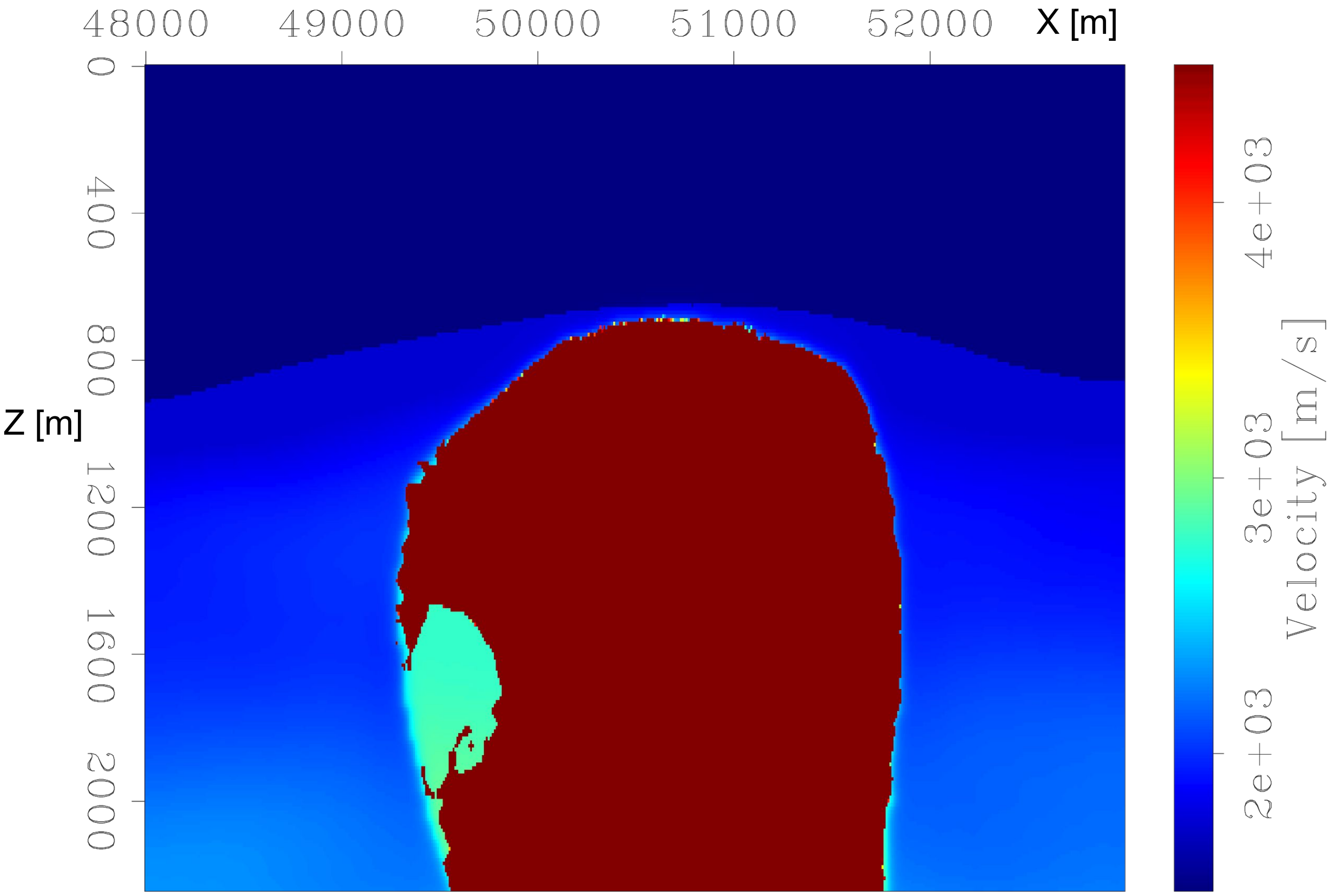
Iteration =15

FULLY GUIDED Inversion result



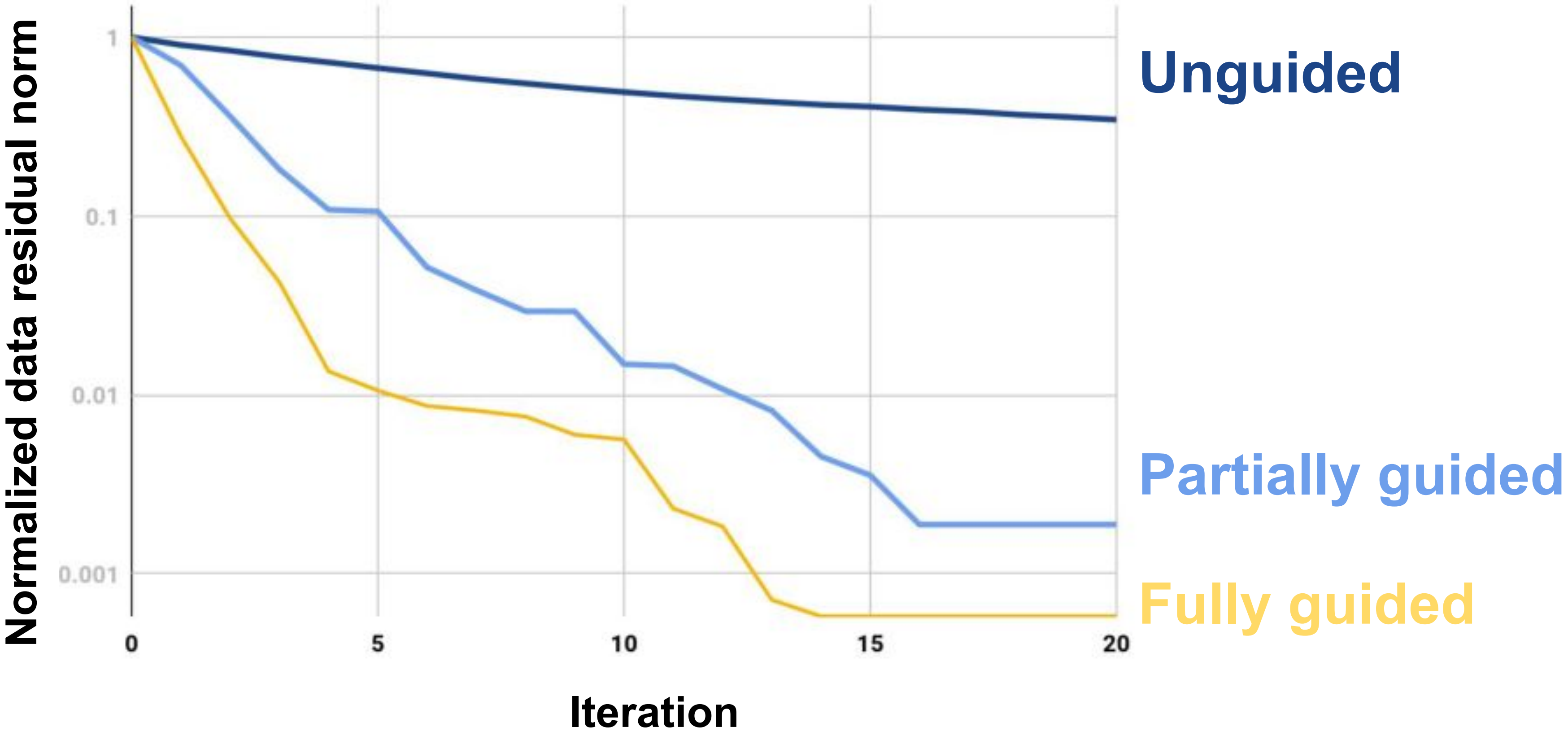
Iteration =15

UNGUIDED Inversion result

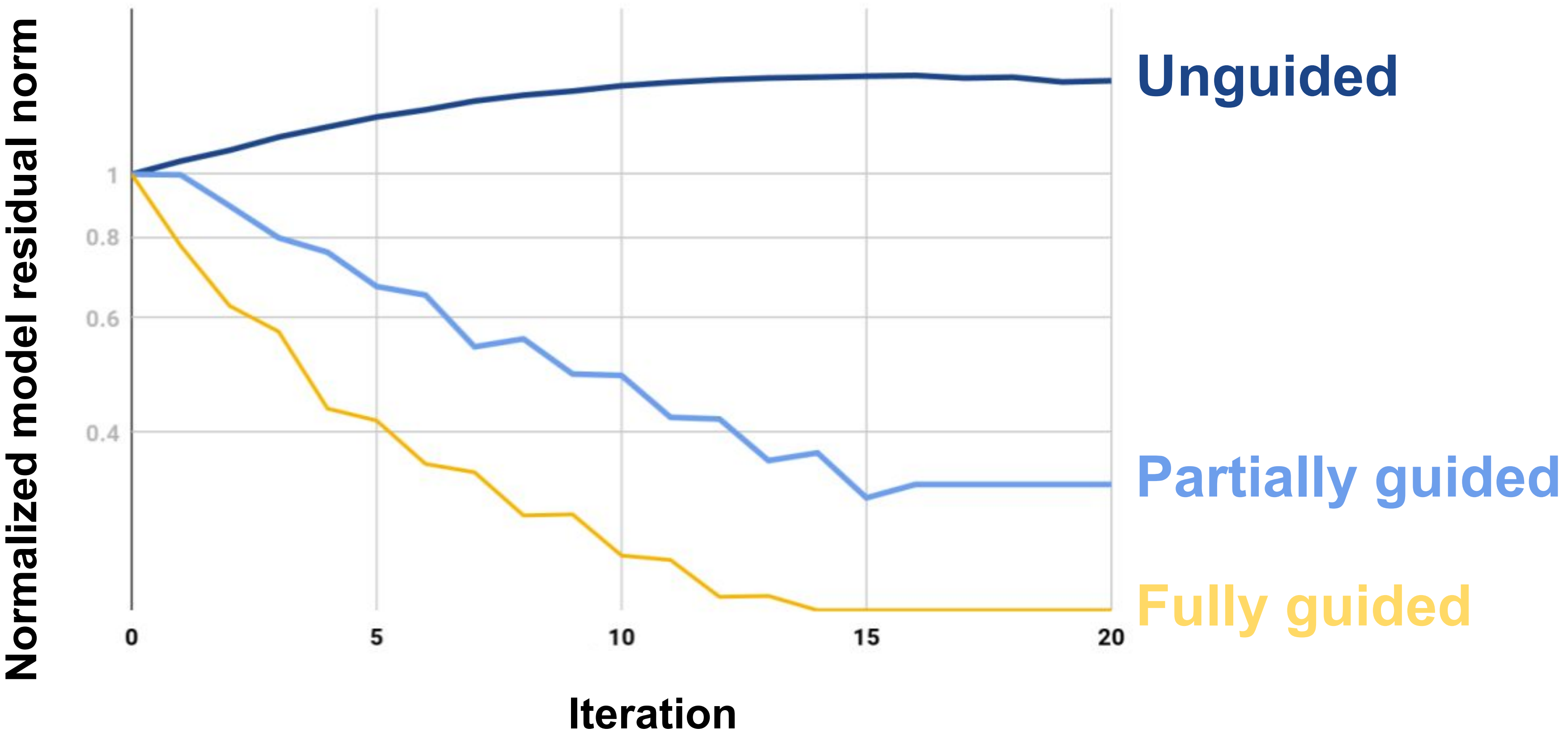


Iteration =65

DATA NORM


Normalized data residual norm
1
0.1
0.01
0.001
0
5
10
15
20
Iteration
Unguided
Partially guided
Fully guided

MODEL NORM


Normalized model residual norm
1
0.8
0.6
0.4
0
5
10
15
20
Iteration
Unguided
Partially guided
Fully guided

How well does any of this work on **real data**?

Application to 3D field data

Provided by Shell Exploration & Production Company

Application to 3D field data

Provided by Shell Exploration & Production Company

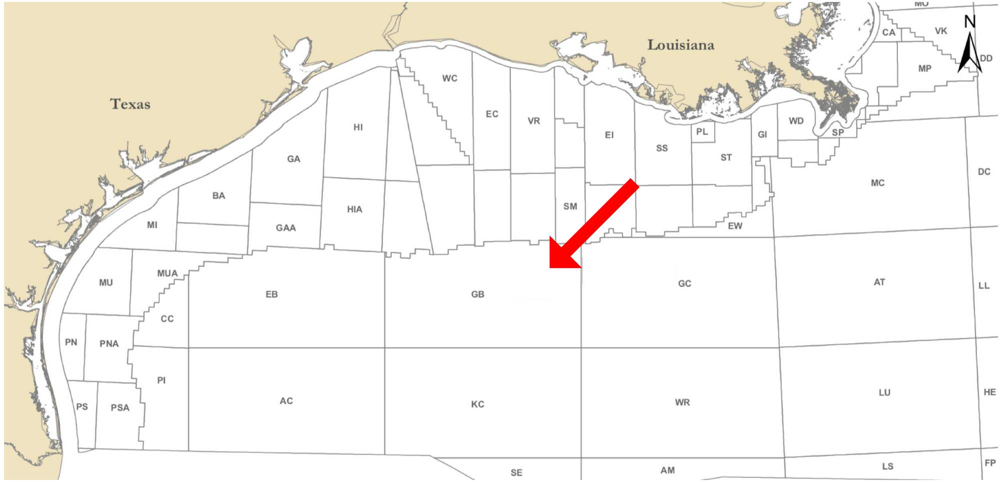
OBN (ocean bottom-node) survey (2010)

Application to 3D field data

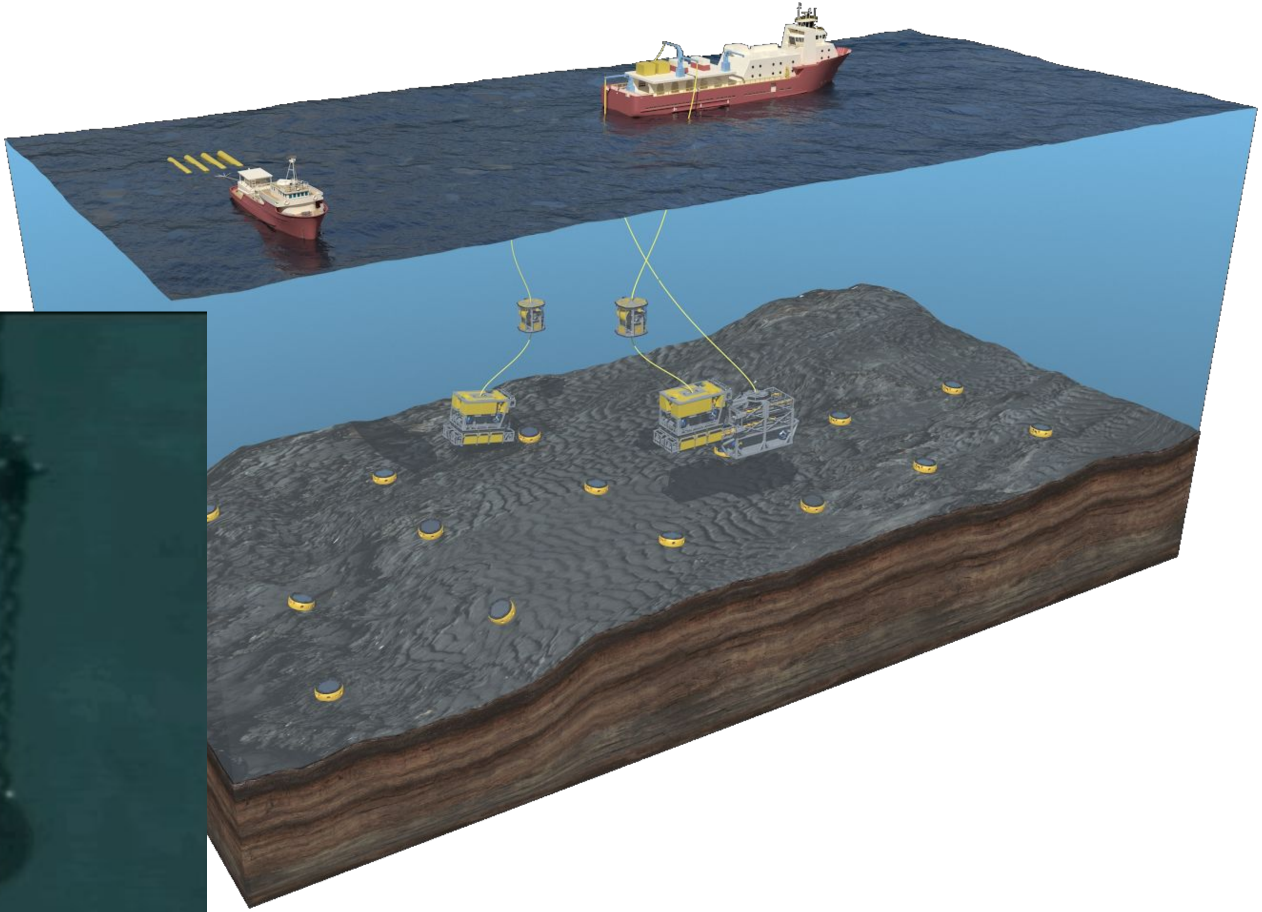
Provided by Shell Exploration & Production Company

OBN (ocean bottom-node) survey (2010)

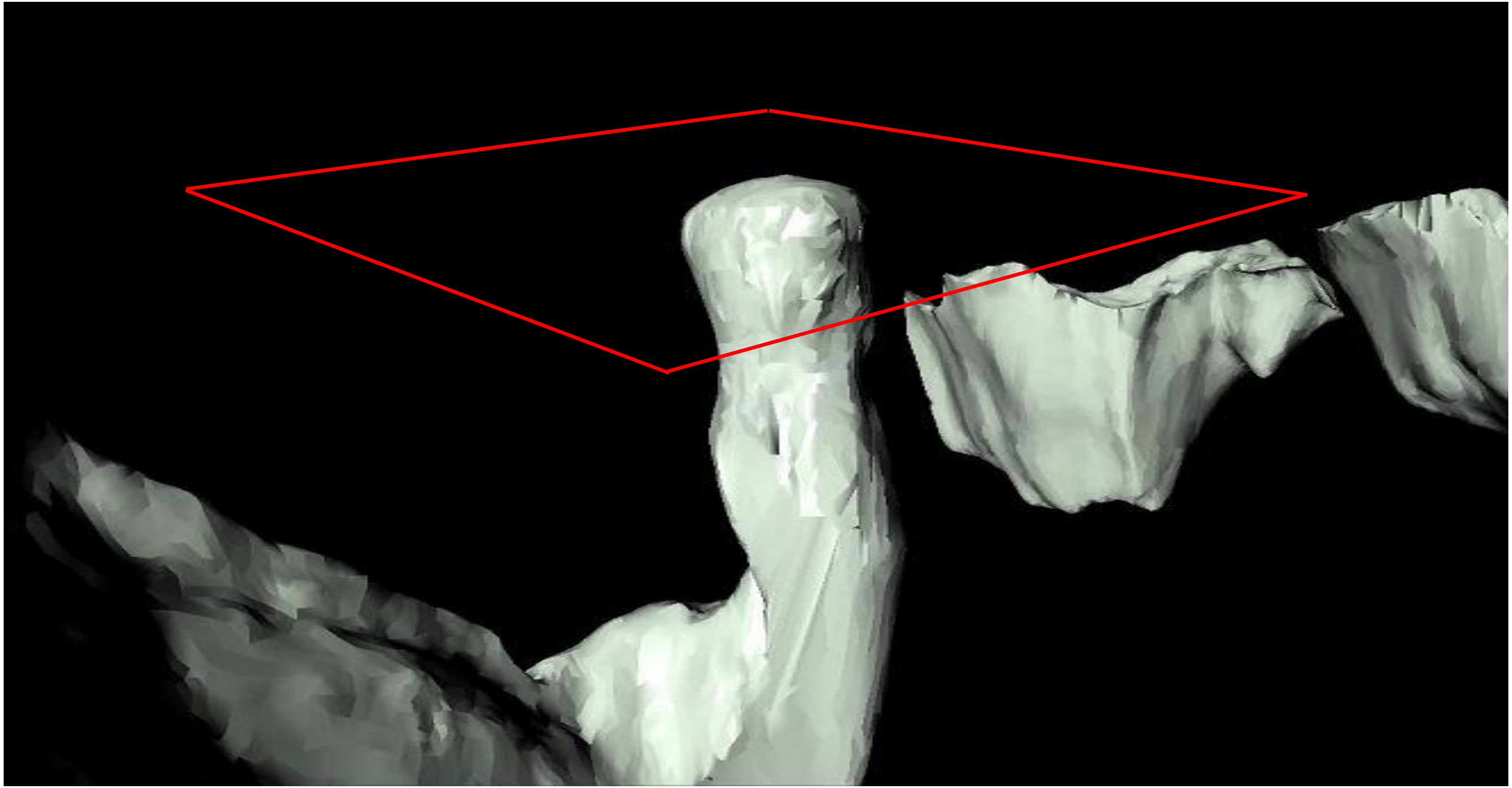
Gulf of Mexico, offshore Louisiana



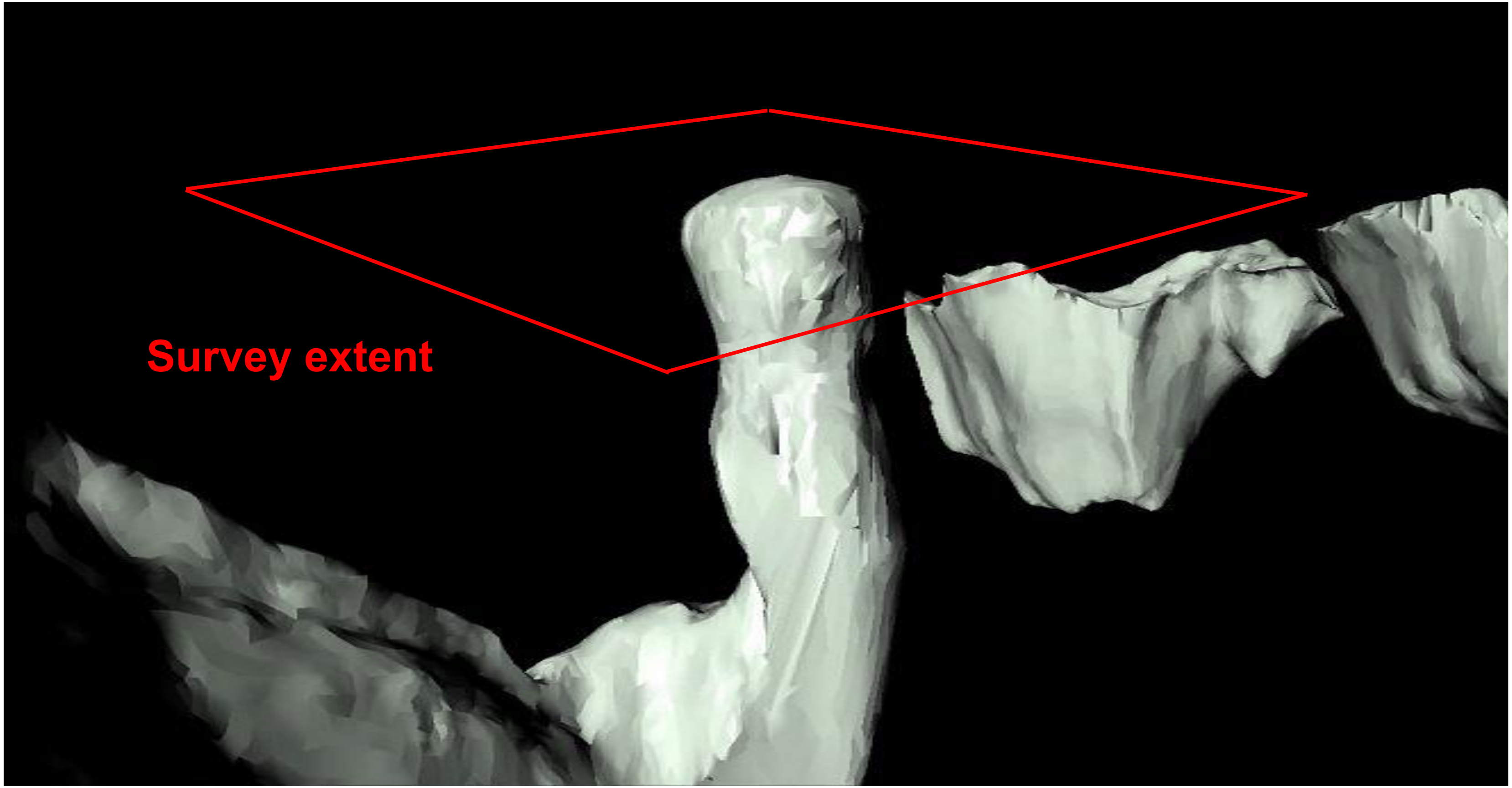
AIRGUN SOURCE



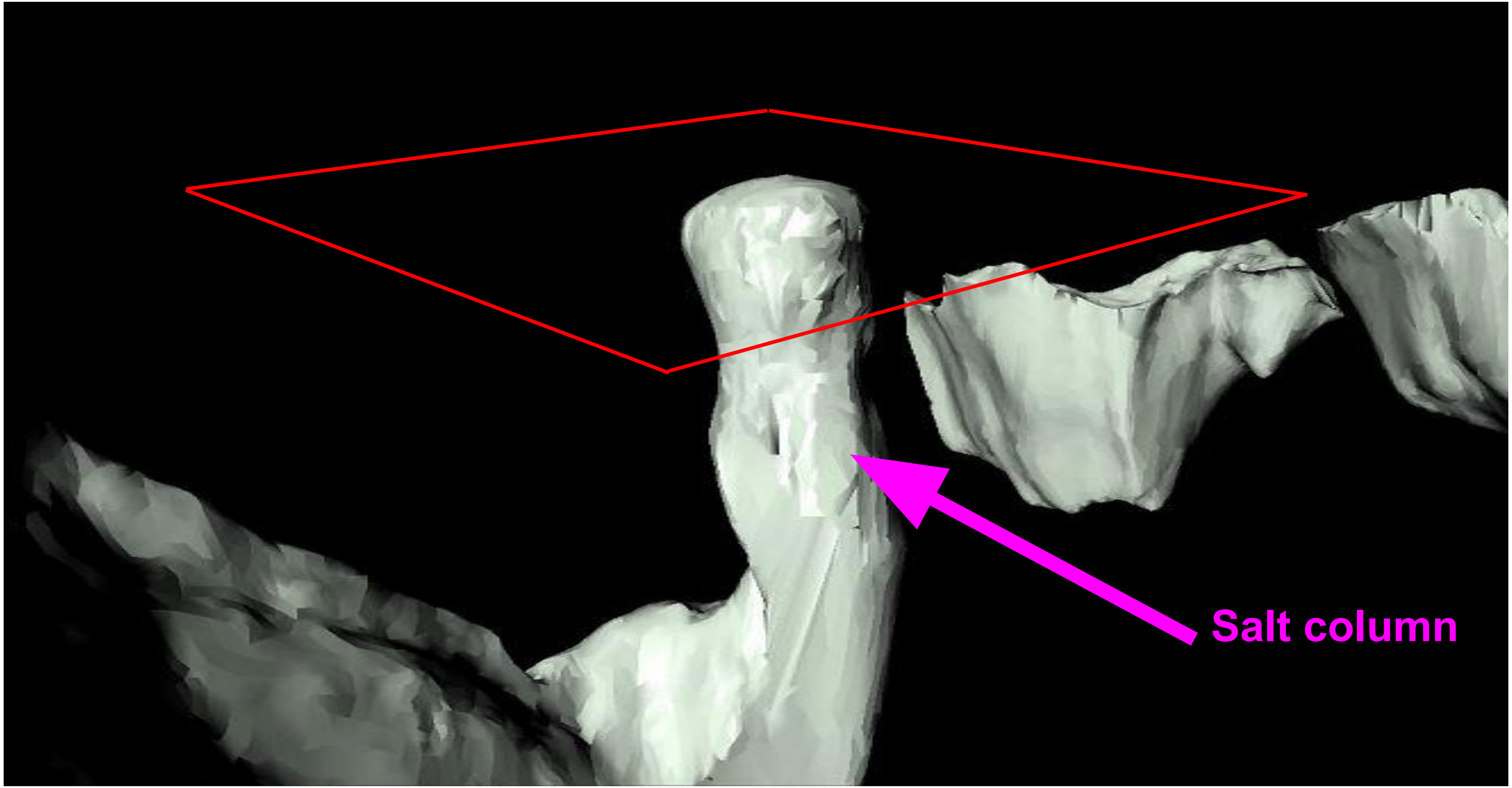
Oblique view of survey area

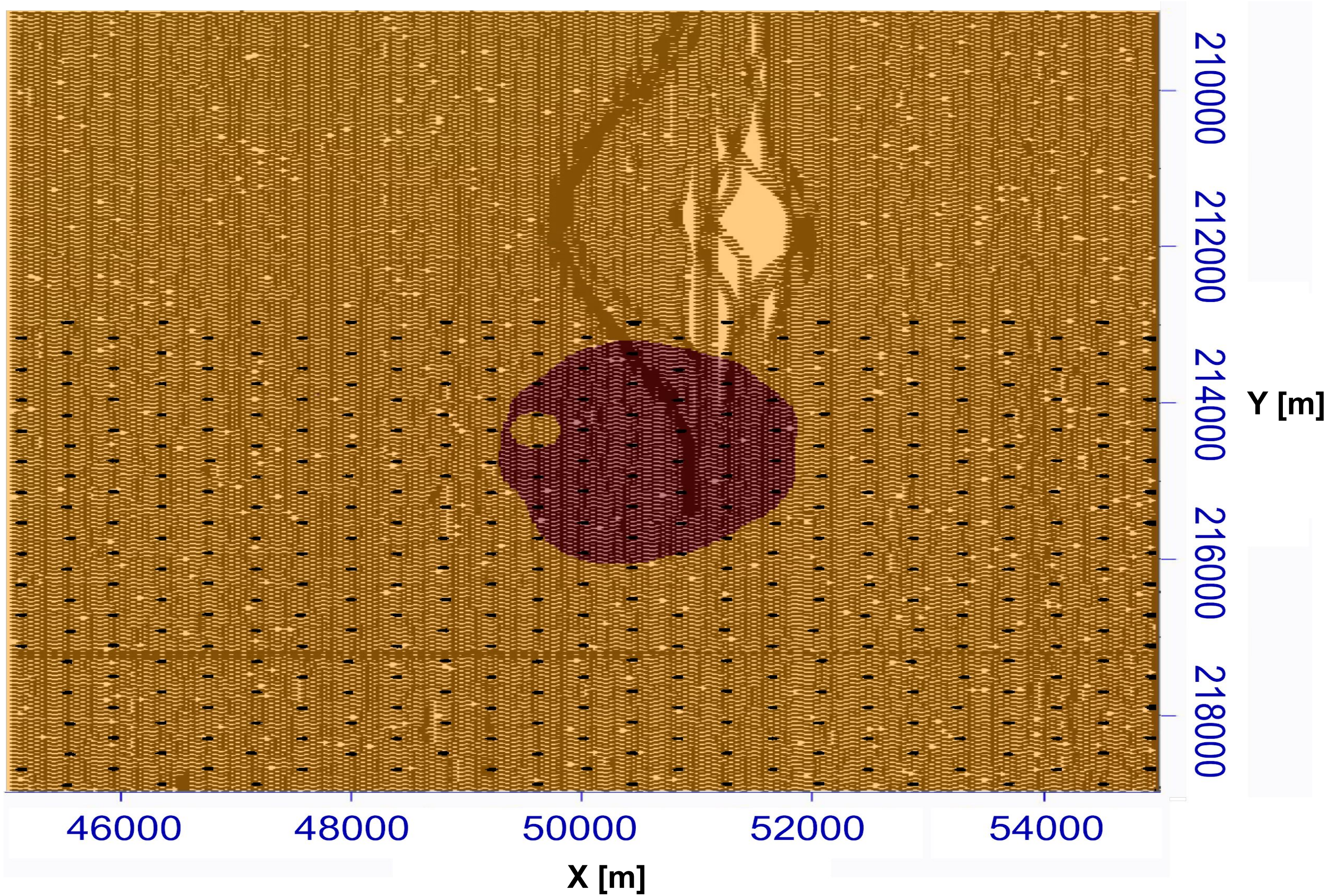


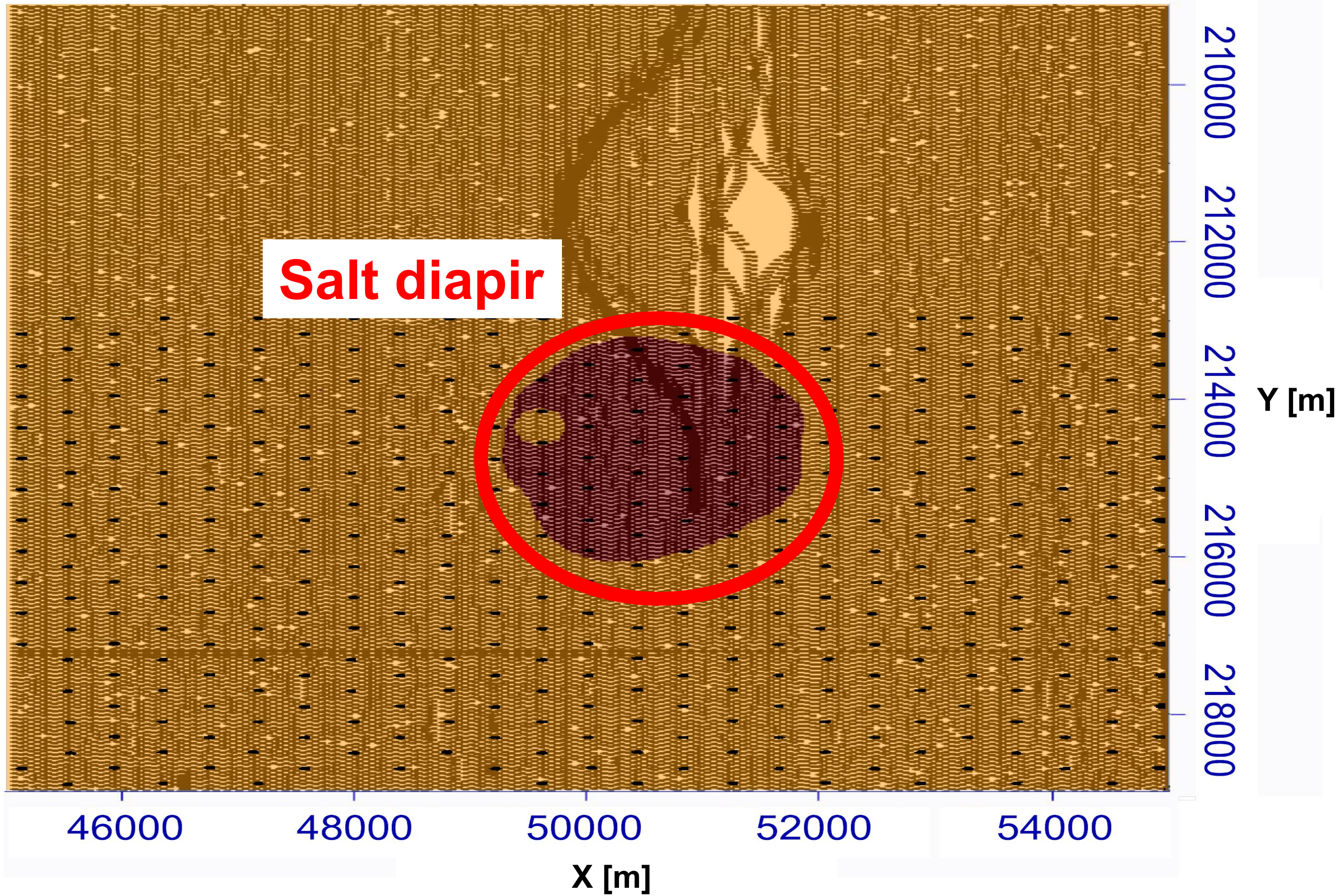
Oblique view of survey area



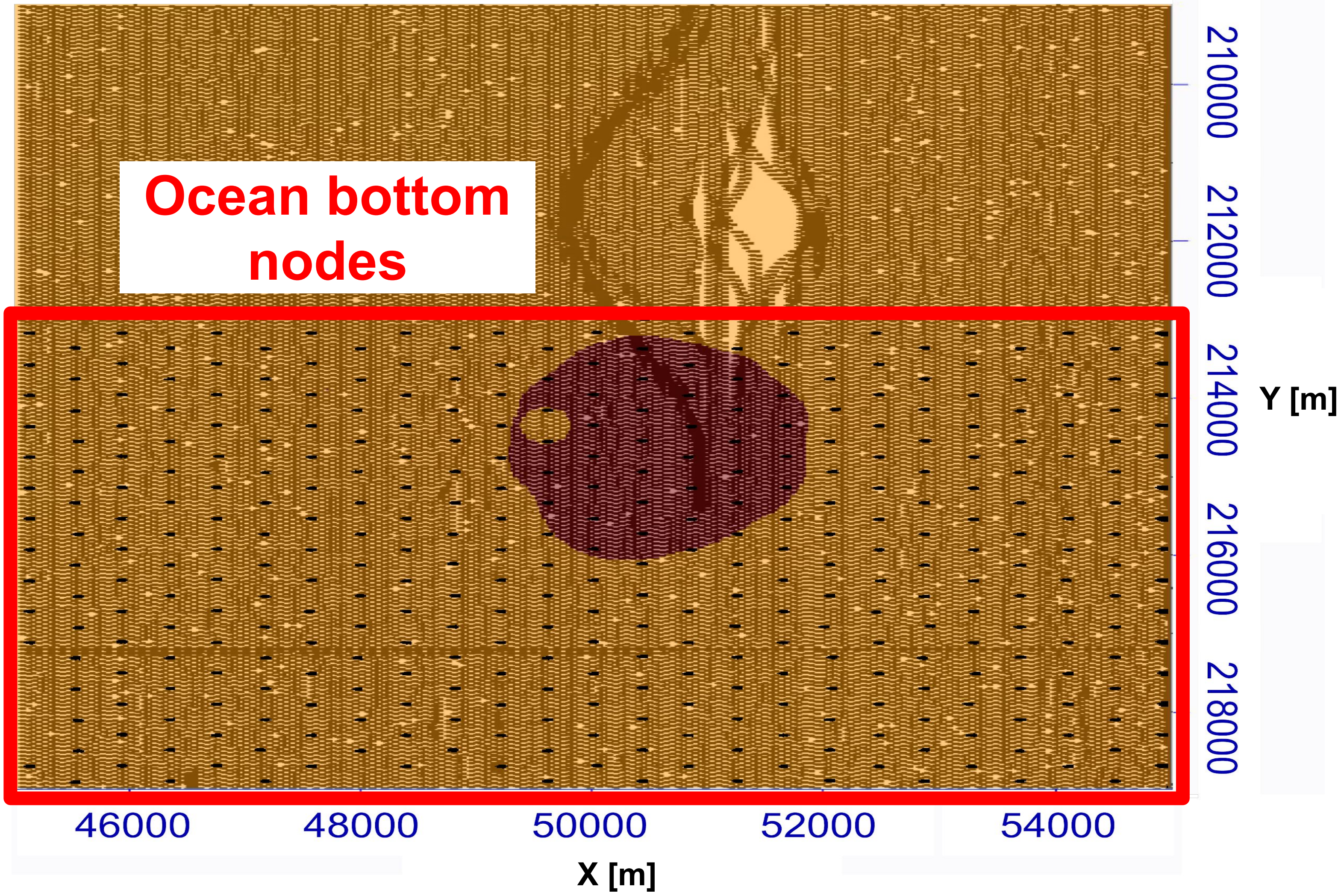
Oblique view of survey area



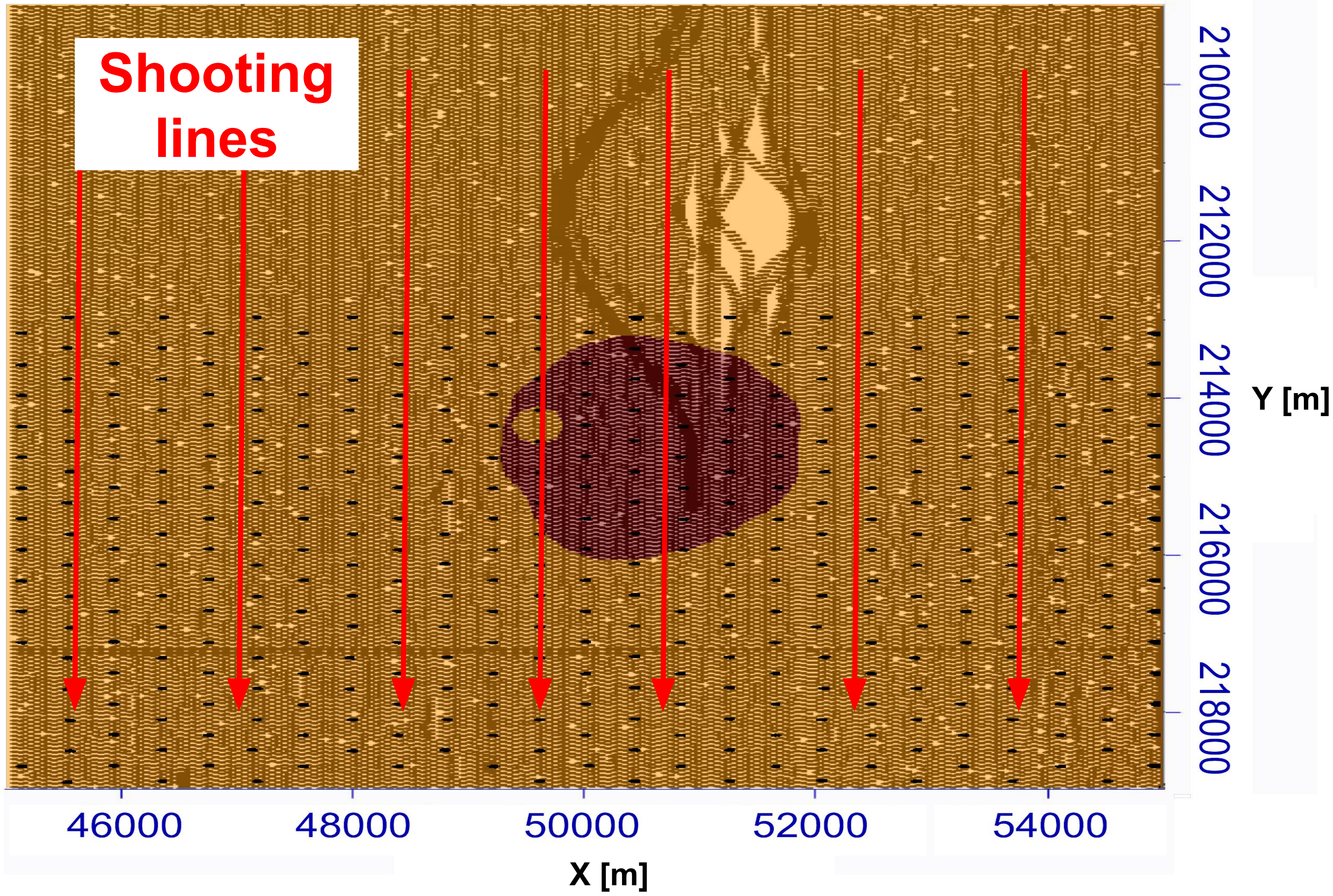

210000
212000
214000
216000
218000
Y [m]
46000
48000
50000
52000
54000
X [m]



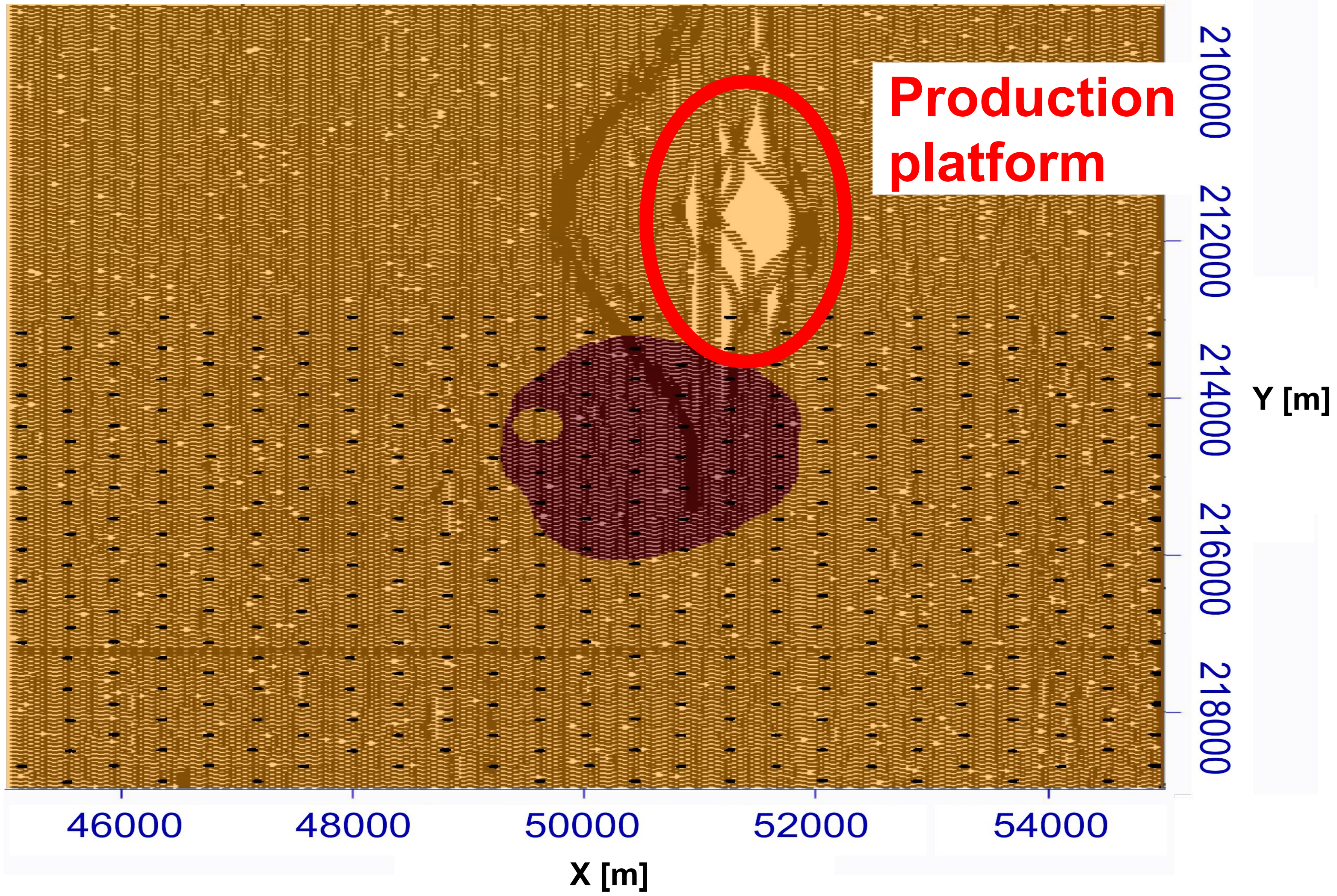
Salt diapir
210000
212000
214000
216000
218000
Y [m]
46000
48000
50000
52000
54000
X [m]



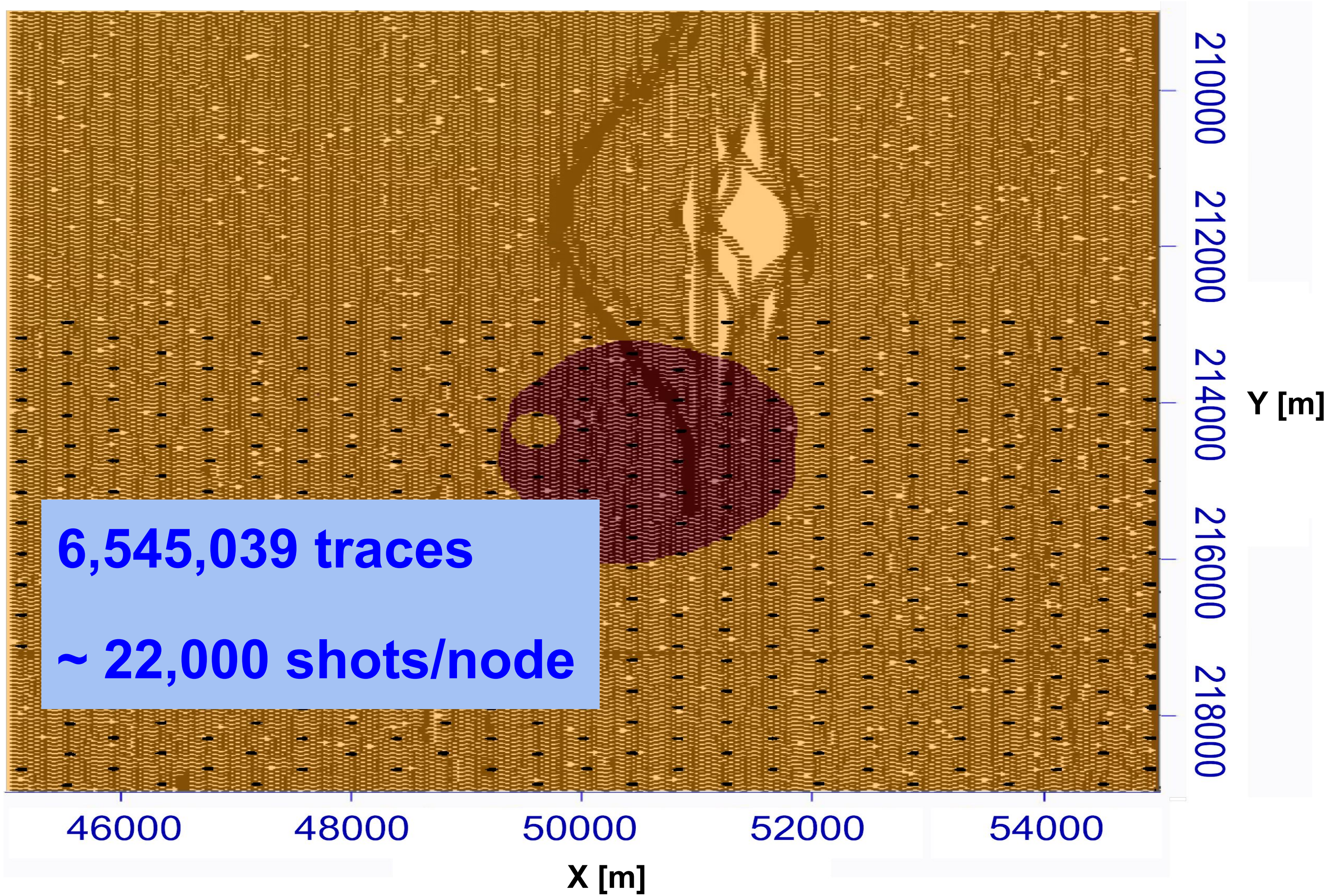
Ocean bottom nodes
210000
212000
214000
216000
218000
Y [m]
46000
48000
50000
52000
54000
X [m]



Shooting lines
210000
212000
214000
216000
218000
Y [m]
46000
48000
50000
52000
54000
X [m]

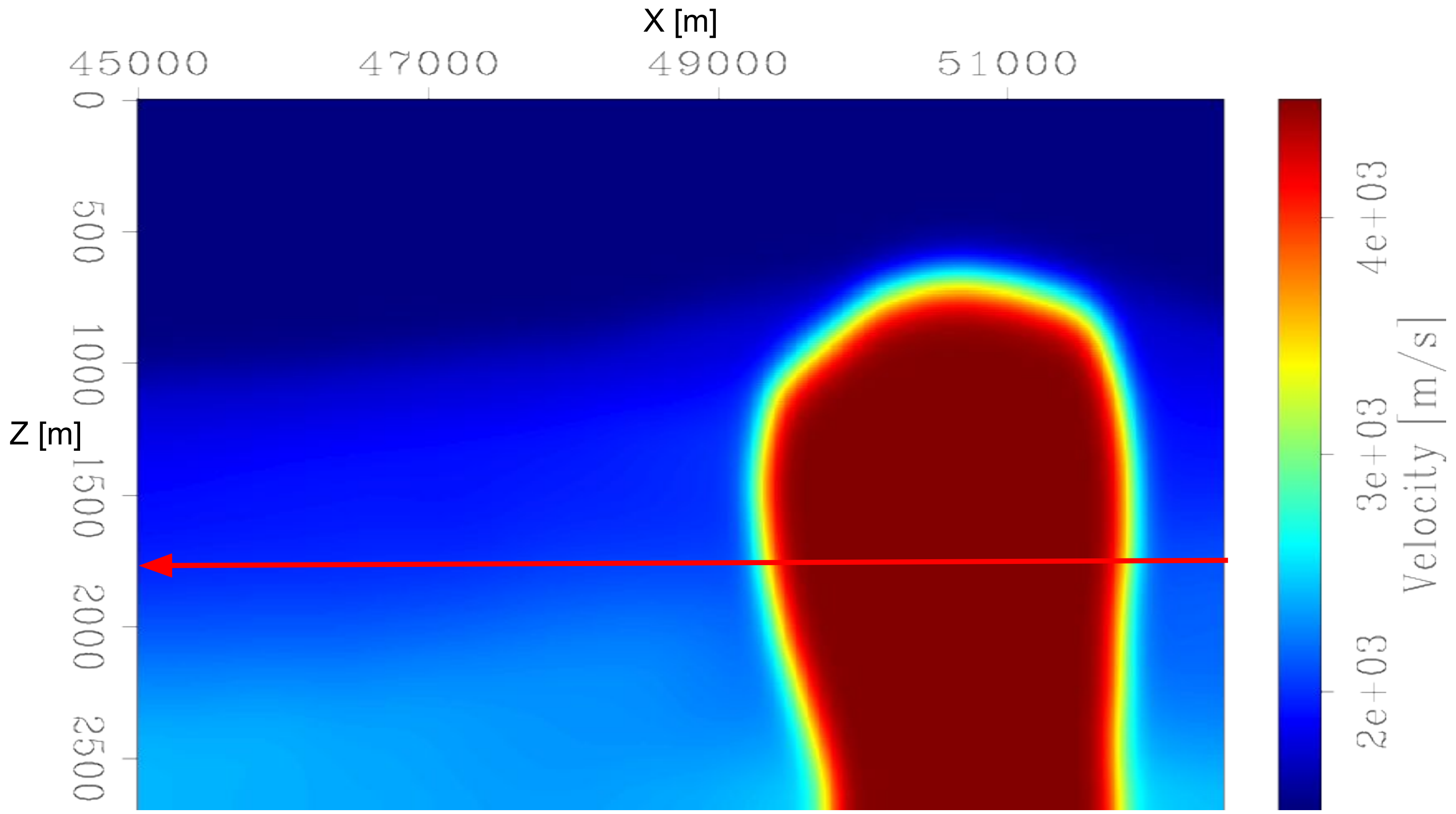


Production platform
210000
212000
214000
216000
218000
Y [m]
46000
48000
50000
52000
54000
X [m]

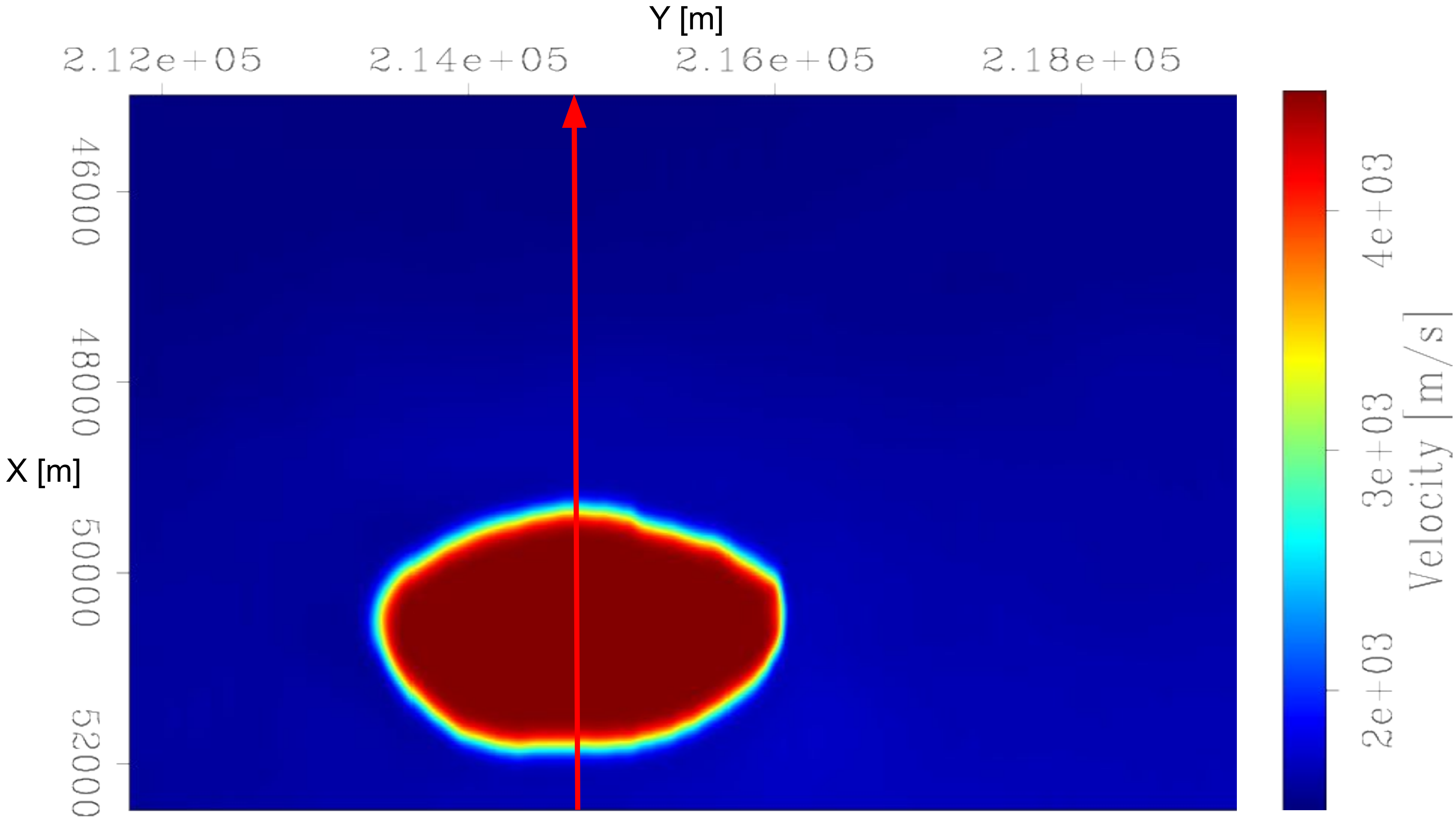

6,545,039 traces
~ 22,000 shots/node
210000
212000
214000
216000
218000
Y [m]
46000
48000
50000
52000
54000
X [m]

STEP 1: **Image the data**

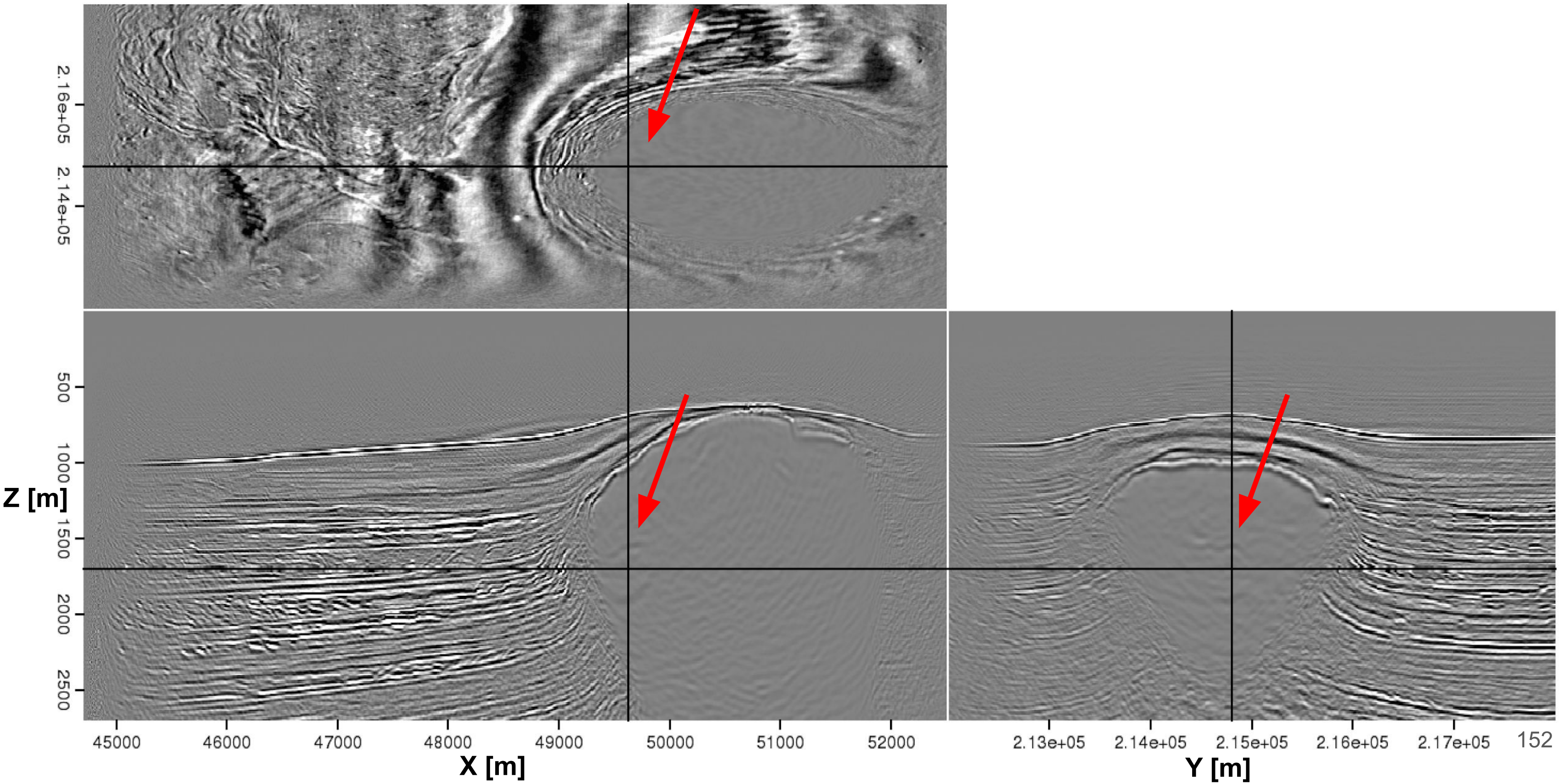
Migration velocity model



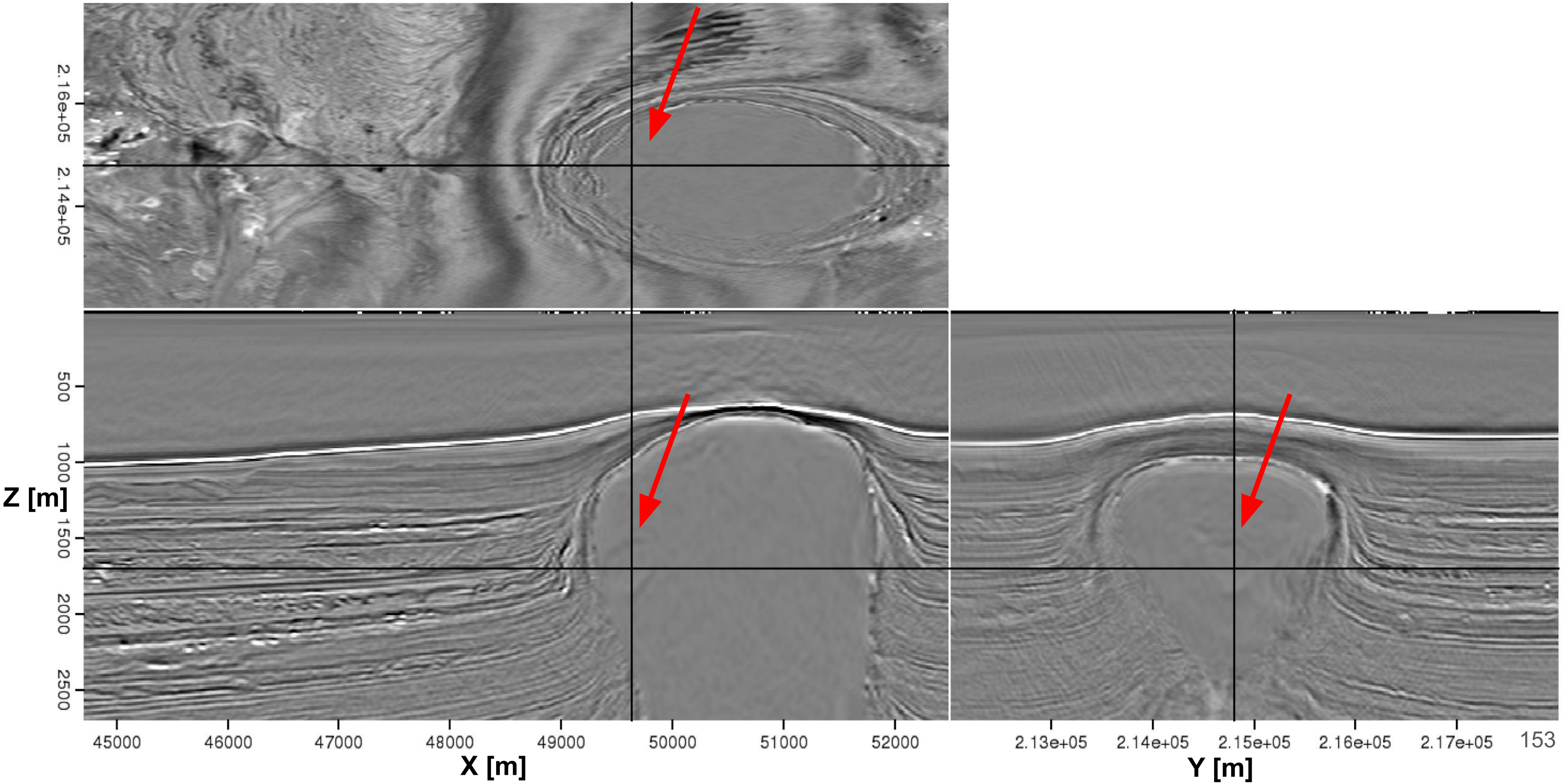
Migration velocity model



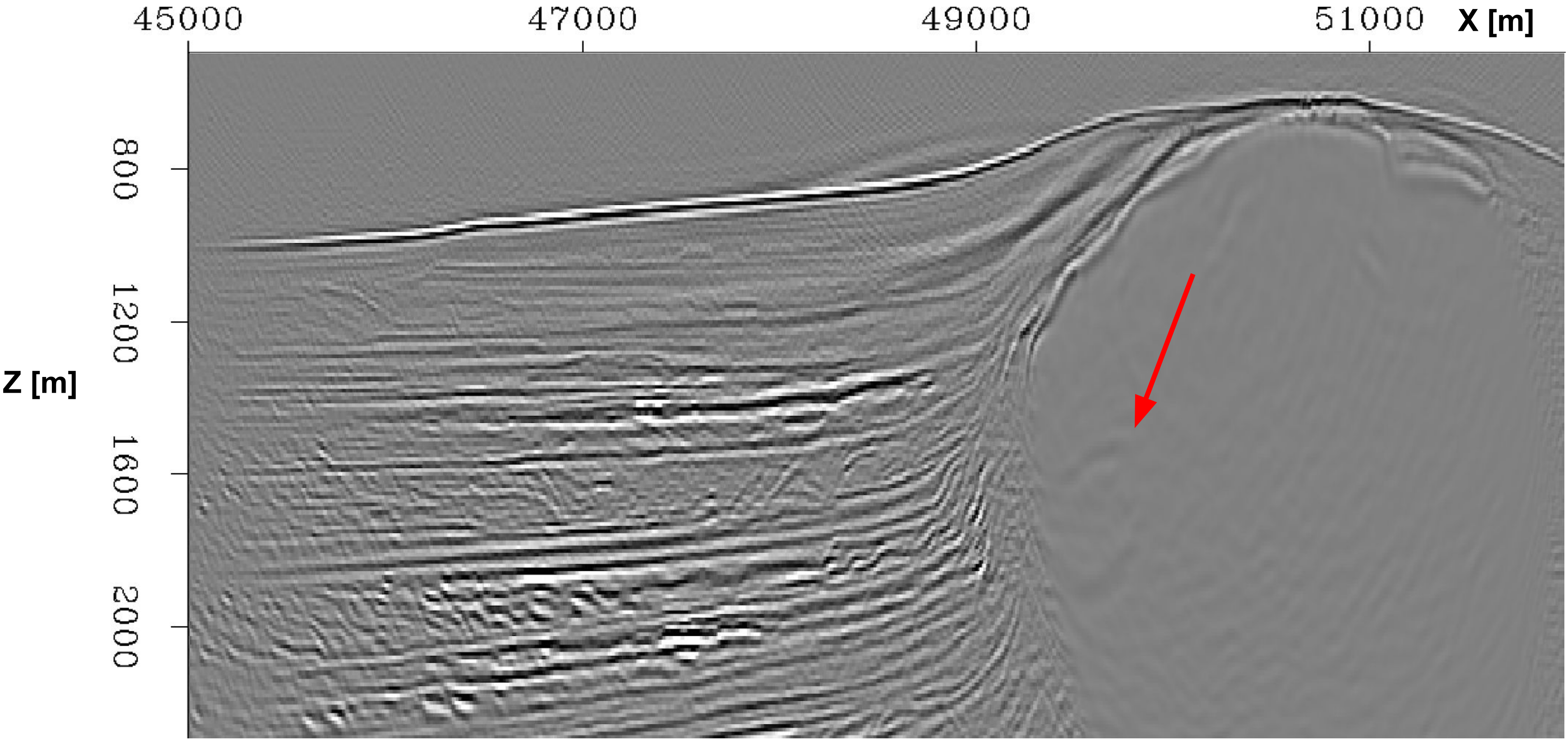
Reverse-Time Migration (RTM) (**Stanford**)



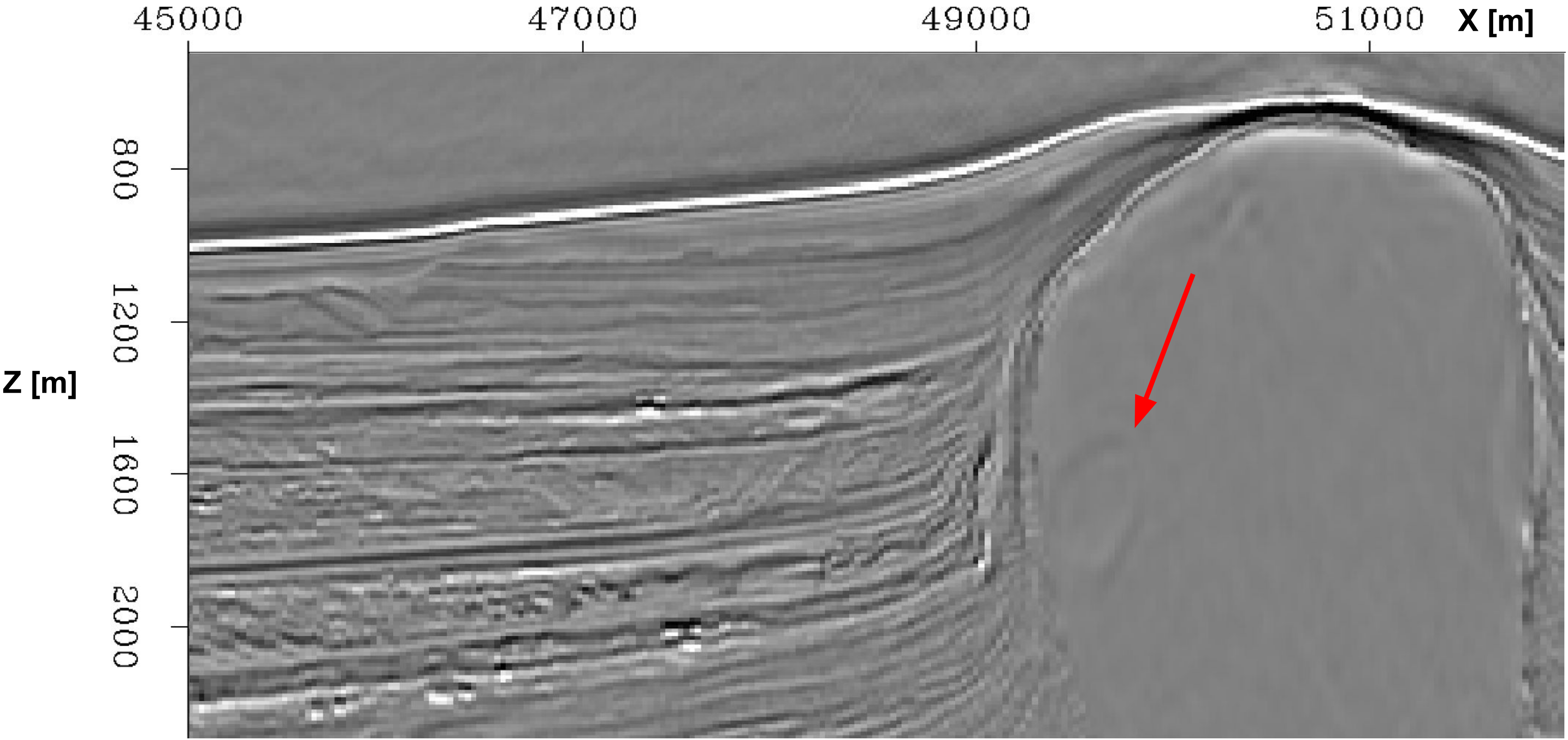
Reverse-Time Migration (RTM) (**Shell**)



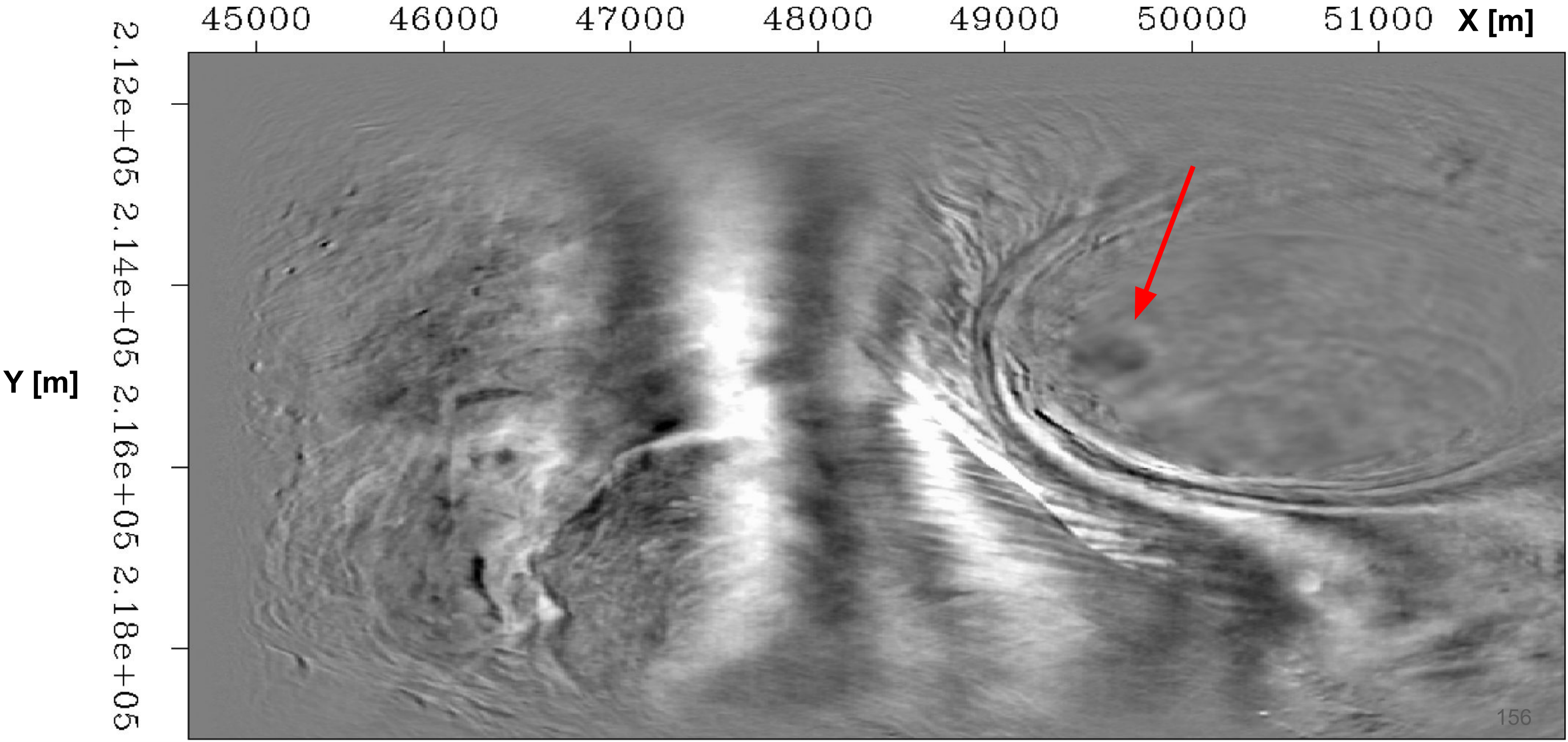
Reverse-Time Migration (RTM) (**Stanford**)



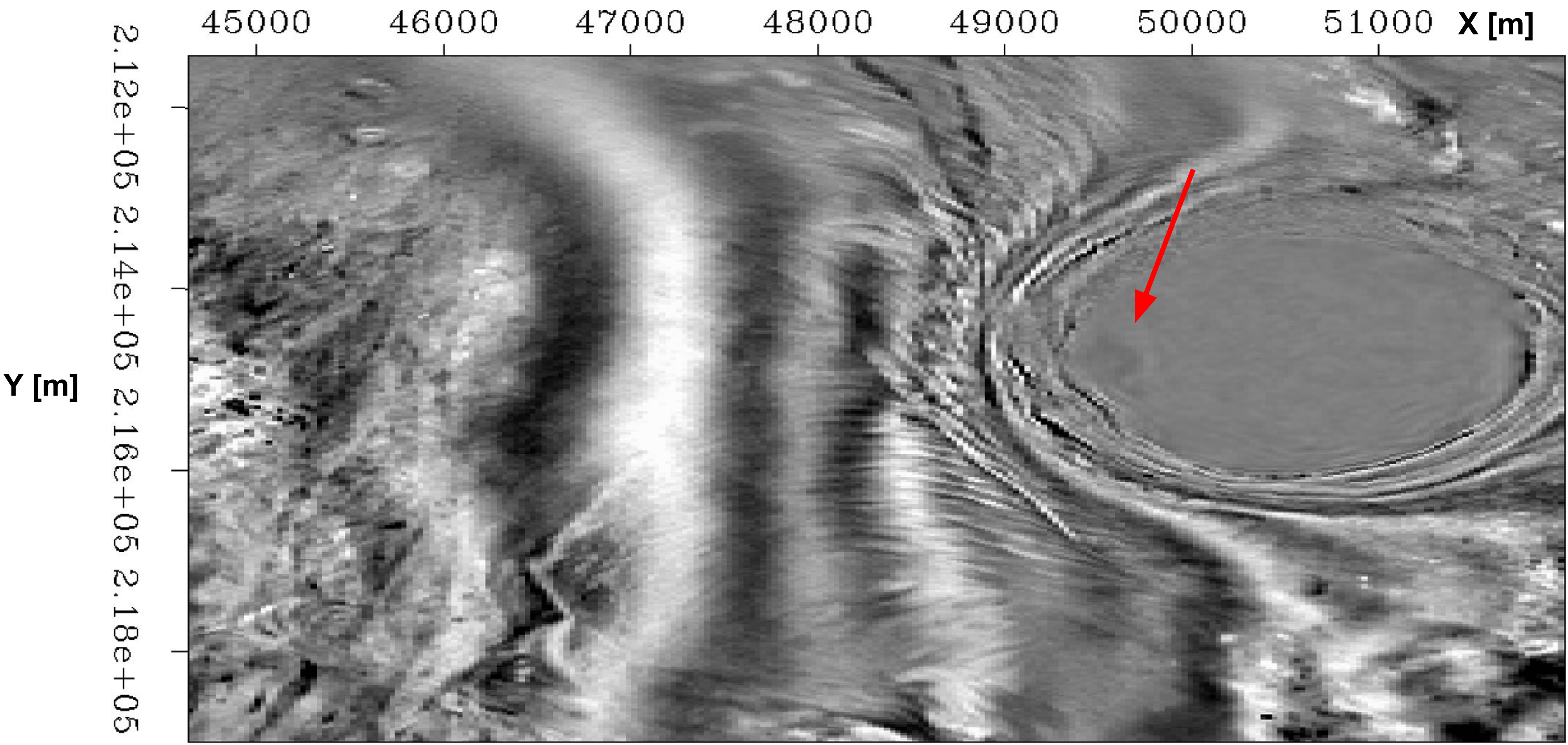
Reverse-Time Migration (RTM) (**Shell**)



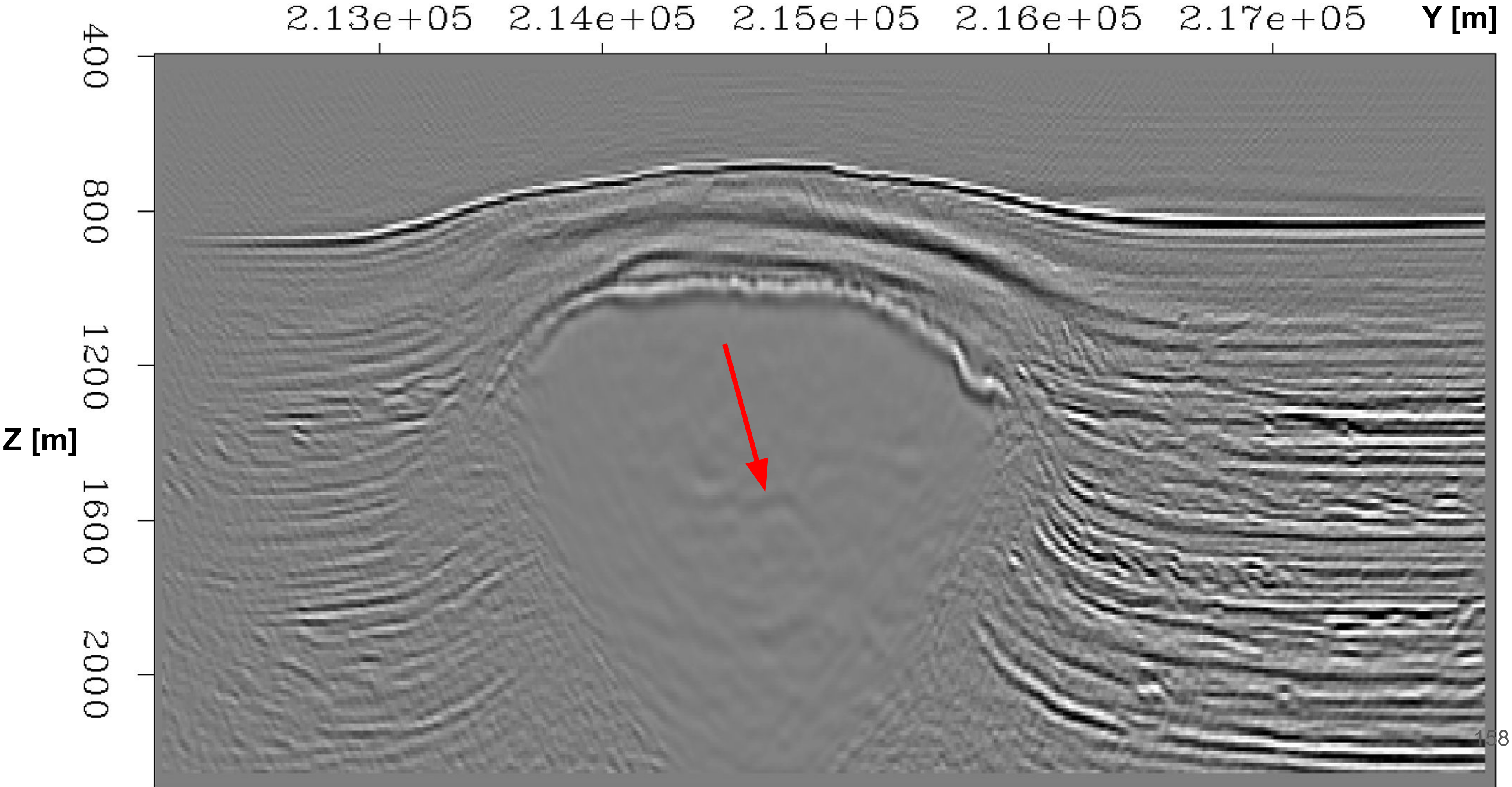
Reverse-Time Migration (RTM) (**Stanford**)



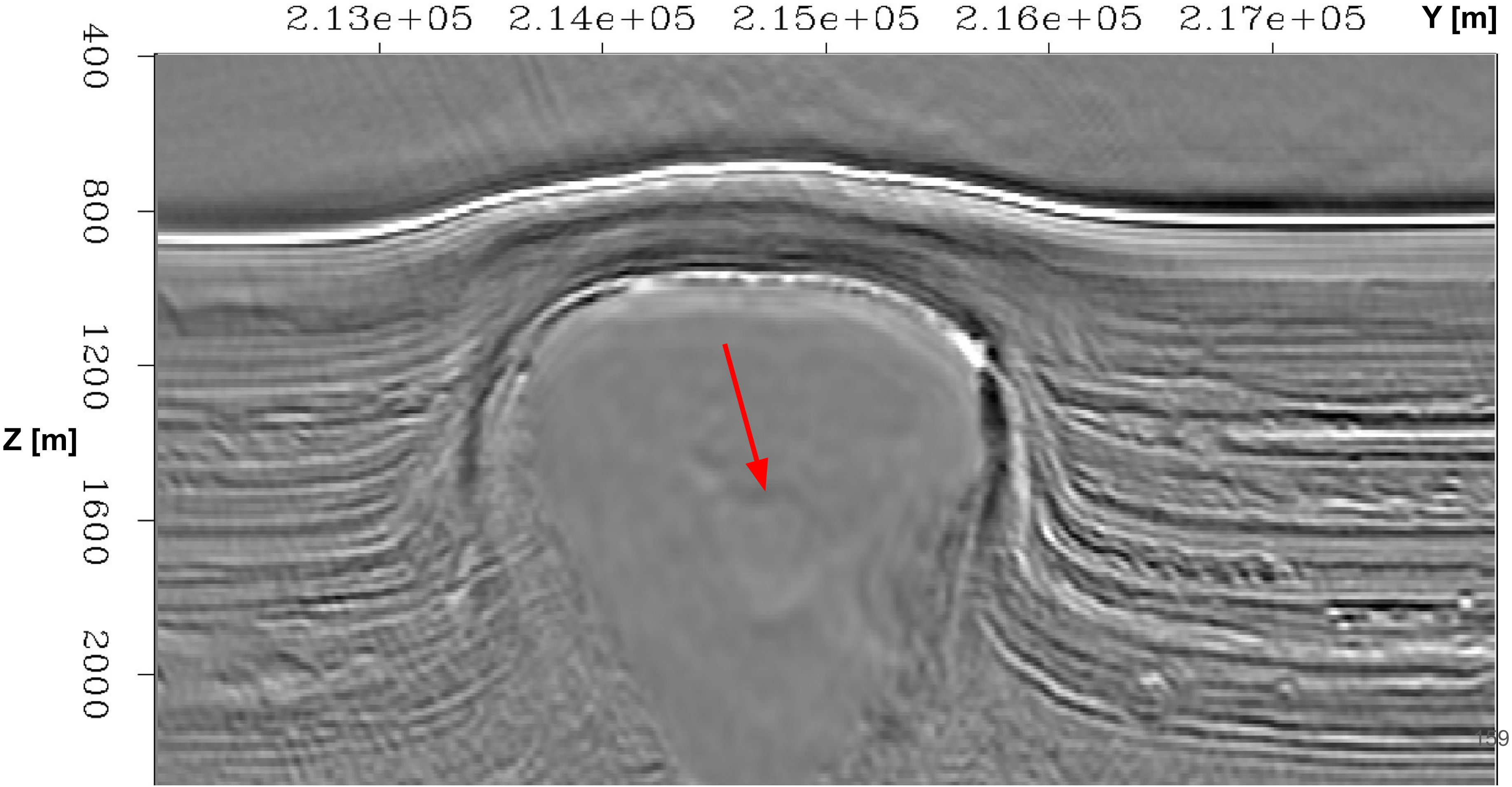
Reverse-Time Migration (RTM) (**Shell**)



Reverse-Time Migration (RTM) (**Stanford**)



Reverse-Time Migration (RTM) (**Shell**)

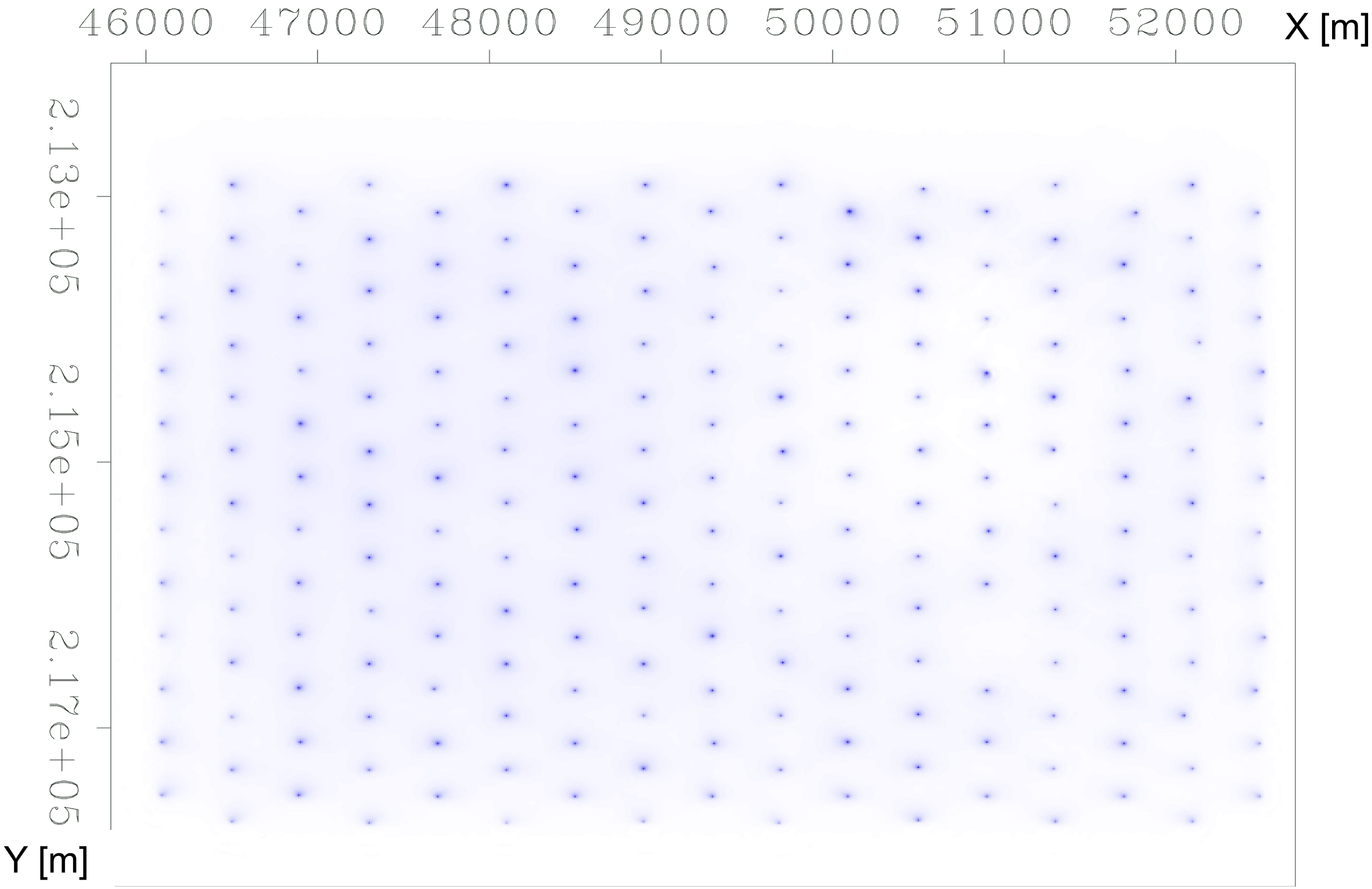


STEP 2: **Run inversion**

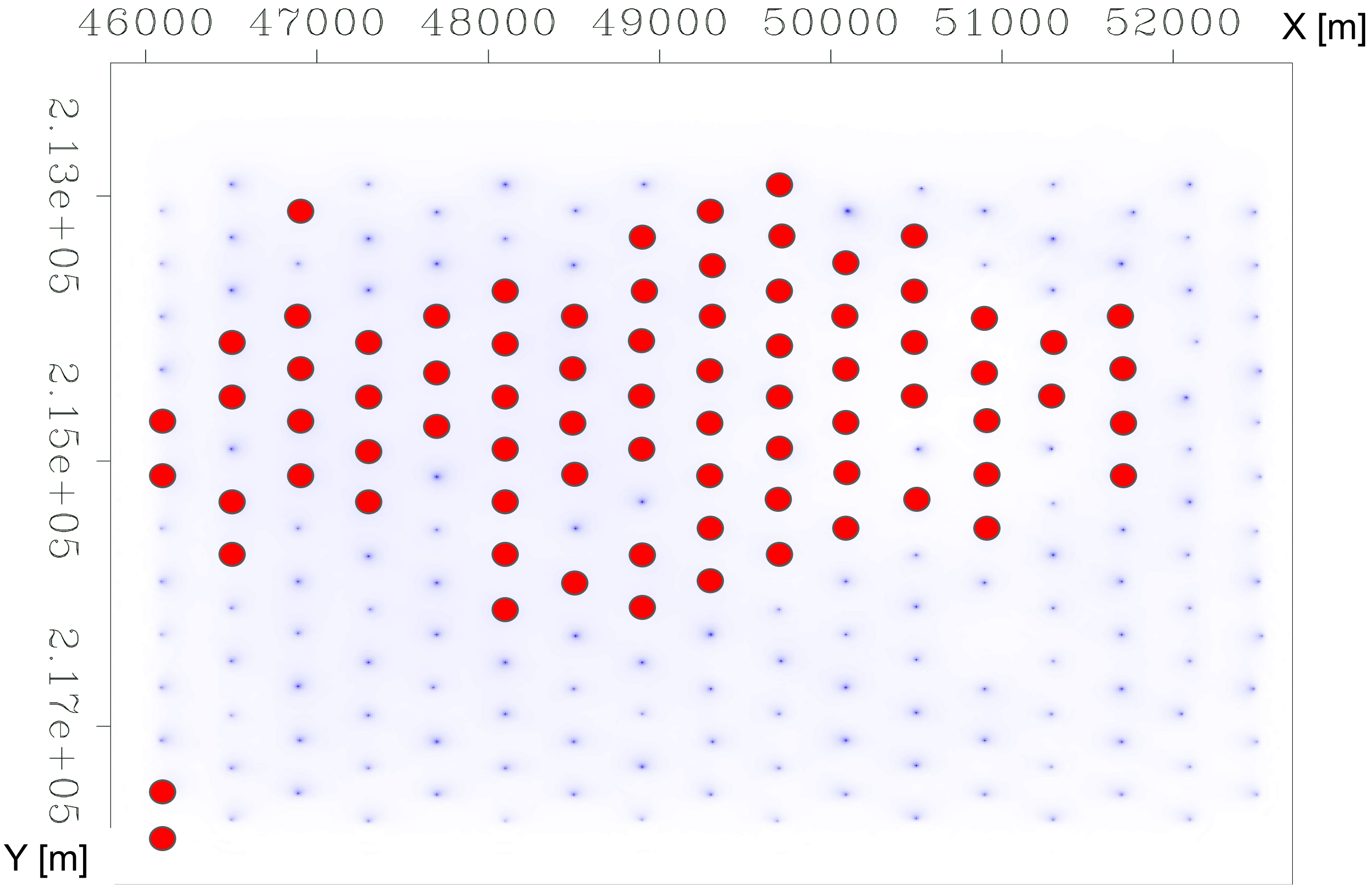
3D inversion results

- Parameterized salt boundary using radial basis functions.
- Inner-loop inversion used Gauss-Newton Hessian.
- Alternated updating between background velocity and level set (salt boundary).
- 77 nodes used.

Nodes used for RTM



Nodes used for inversion



for i in $(1,N)$ **do**

$d_{\text{syn}}(i) = \mathrm{F}(\phi_i, \phi_{i-1})$

$\triangle d_i = d_{\text{obs}} - d_{\text{syn}}(i)$

$g_i = D^T B^T \triangle d_i$

if EvenNumberedIteration **then**

$\triangle\lambda_i = \textbf{CGHessianInv}(g_i)$

$\triangle\phi_i = \mathrm{D}(\triangle\lambda_i)$

$\triangle b_i = 0$

$\alpha = \textbf{linesearch}(\triangle\phi_i)$

$\beta = 0$

else

$\triangle\phi_i = 0$

$\triangle b_i = \textbf{CGHessianInv}(g_i)$

$\alpha = 0$

$\beta = \textbf{linesearch}(\triangle b_i)$

end if

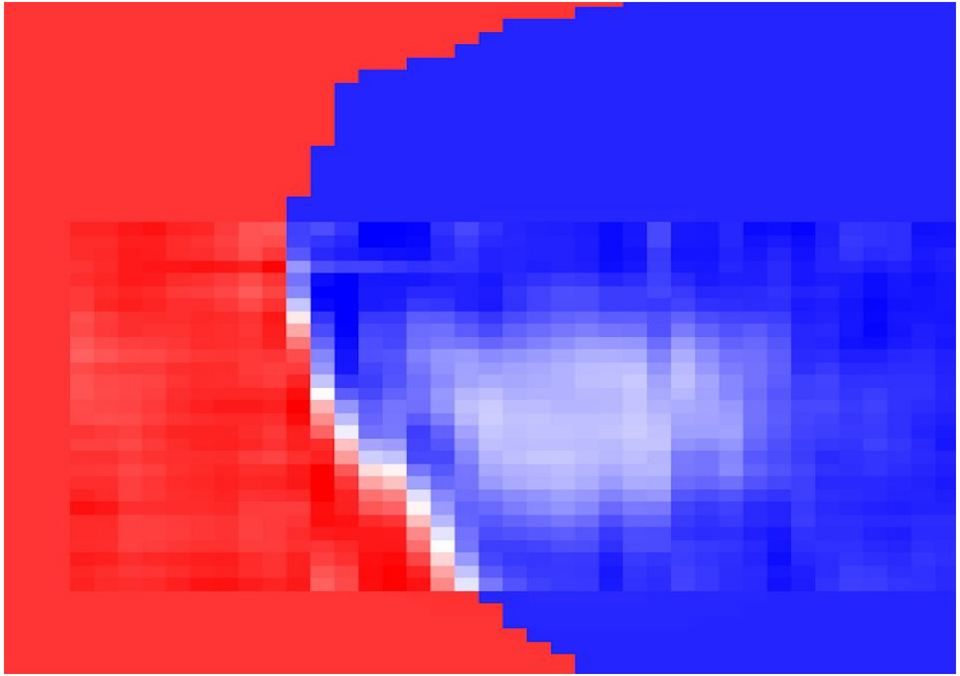
$\phi_i = \phi_{i-1} - \alpha \cdot \triangle\phi_i$

$b_i = b_{i-1} - \beta \cdot \triangle b_i$

end for

Return $m(\lambda N, b_N)$

Implicit surface



Implicit surface

for i in $(1,N)$ **do**

$d_{\text{syn}}(i) = \mathrm{F}(\phi_i, \phi_{i-1})$

$\triangle d_i = d_{\text{obs}} - d_{\text{syn}}(i)$

$g_i = D^T B^T \triangle d_i$

if EvenNumberedIteration **then**

$\triangle \lambda_i = \mathbf{CGHessianInv}(g_i)$

$\triangle \phi_i = \mathrm{D}(\triangle \lambda_i)$

$\triangle b_i = 0$

$\alpha = \mathbf{linesearch}(\triangle \phi_i)$

$\beta = 0$

else

$\triangle \phi_i = 0$

$\triangle b_i = \mathbf{CGHessianInv}(g_i)$

$\alpha = 0$

$\beta = \mathbf{linesearch}(\triangle b_i)$

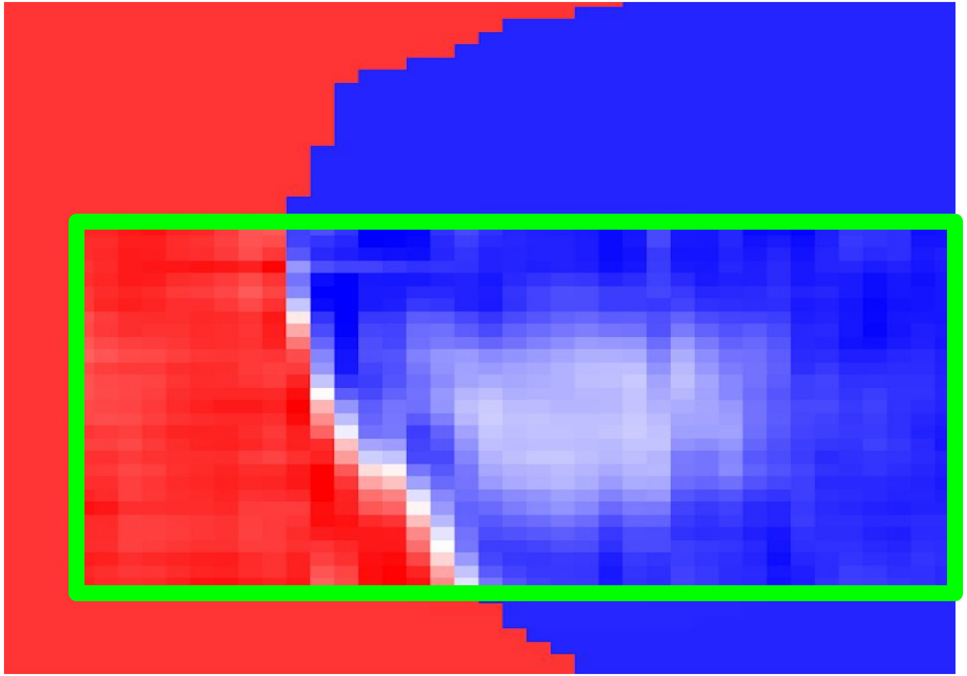
end if

$\phi_i = \phi_{i-1} - \alpha \cdot \triangle \phi_i$

$b_i = b_{i-1} - \beta \cdot \triangle b_i$

end for

Return $m(\lambda N, b_N)$



for i in $(1,N)$ **do**

$d_{\text{syn}}(i) = \text{F}(\phi_i, b_{i-1})$

$\triangle d_i = d_{\text{obs}} - d_{\text{syn}}(i)$

$g_i = D^T B^T \triangle d_i$

if EvenNumberedIteration **then**

$\triangle\lambda_i = \textbf{CGHessianInv}(g_i)$

$\triangle\phi_i = \text{D}(\triangle\lambda_i)$

$\triangle b_i = 0$

$\alpha = \textbf{linesearch}(\triangle\phi_i)$

$\beta = 0$

else

$\triangle\phi_i = 0$

$\triangle b_i = \textbf{CGHessianInv}(g_i)$

$\alpha = 0$

$\beta = \textbf{linesearch}(\triangle b_i)$

end if

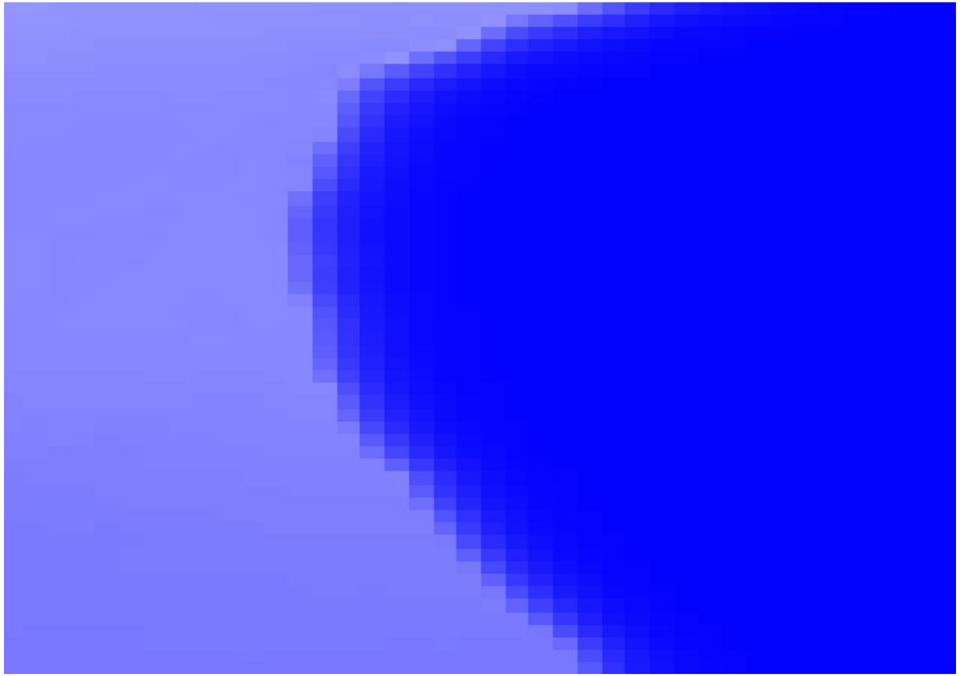
$\phi_i = \phi_{i-1} - \alpha \cdot \triangle\phi_i$

$b_i = b_{i-1} - \beta \cdot \triangle b_i$

end for

Return $m(\lambda N, b_N)$

Background velocity model



for i in $(1,N)$ **do**

$d_{\text{syn}}(i) = \mathrm{F}(\phi_i, b_{i-1})$

$\triangle d_i = d_{\text{obs}} - d_{\text{syn}}(i)$

$g_i = D^T B^T \triangle d_i$

if EvenNumberedIteration **then**

$\triangle\lambda_i = \textbf{CGHessianInv}(g_i)$

$\triangle\phi_i = \mathrm{D}(\triangle\lambda_i)$

$\triangle b_i = 0$

$\alpha = \textbf{linesearch}(\triangle\phi_i)$

$\beta = 0$

else

$\triangle\phi_i = 0$

$\triangle b_i = \textbf{CGHessianInv}(g_i)$

$\alpha = 0$

$\beta = \textbf{linesearch}(\triangle b_i)$

end if

$\phi_i = \phi_{i-1} - \alpha \cdot \triangle\phi_i$

$b_i = b_{i-1} - \beta \cdot \triangle b_i$

end for

Return $m(\lambda N, b_N)$

Synthetic modeled data

for i in $(1,N)$ **do**

$d_{\text{syn}}(i) = \mathrm{F}(\phi_i, b_{i-1})$

$\triangle d_i = d_{\text{obs}} - d_{\text{syn}}(i)$

$g_i = D^T B^T \triangle d_i$

if EvenNumberedIteration **then**

$\triangle\lambda_i = \mathbf{CGHessianInv}(g_i)$

$\triangle\phi_i = \mathrm{D}(\triangle\lambda_i)$

$\triangle b_i = 0$

$\alpha = \mathbf{linesearch}(\triangle\phi_i)$

$\beta = 0$

else

$\triangle\phi_i = 0$

$\triangle b_i = \mathbf{CGHessianInv}(g_i)$

$\alpha = 0$

$\beta = \mathbf{linesearch}(\triangle b_i)$

end if

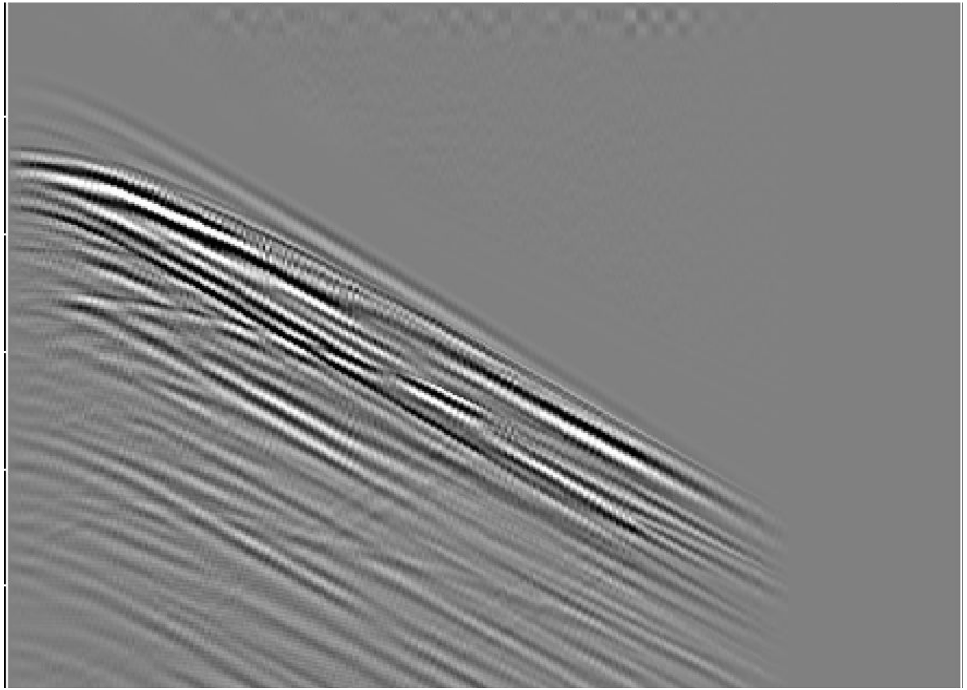
$\phi_i = \phi_{i-1} - \alpha \cdot \triangle\phi_i$

$b_i = b_{i-1} - \beta \cdot \triangle b_i$

end for

Return $m(\lambda N, b_N)$

Data residual



for i in $(1,N)$ **do**

$d_{\text{syn}}(i) = \text{F}(\phi_i, b_{i-1})$

$\triangle d_i = d_{\text{obs}} - d_{\text{syn}}(i)$

$g_i = D^T B^T \triangle d_i$

if EvenNumberedIteration **then**

$\triangle \lambda_i =$ **CGHessianInv**(g_i)

$\triangle \phi_i = \text{D}(\triangle \lambda_i)$

$\triangle b_i = 0$

$\alpha =$ **linesearch**$(\triangle \phi_i)$

$\beta = 0$

else

$\triangle \phi_i = 0$

$\triangle b_i =$ **CGHessianInv**(g_i)

$\alpha = 0$

$\beta =$ **linesearch**$(\triangle b_i)$

end if

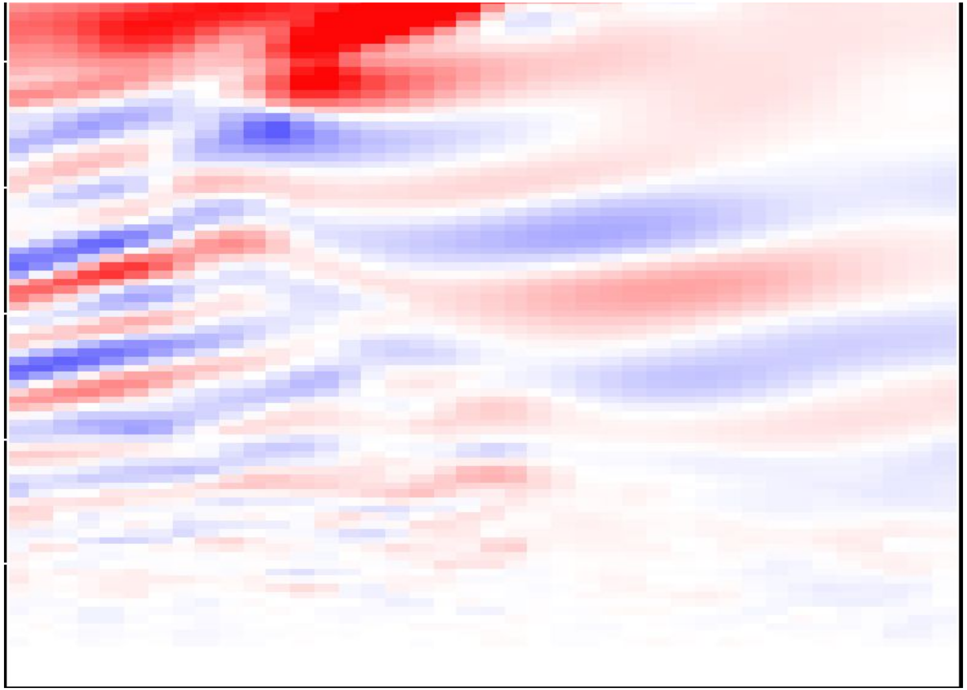
$\phi_i = \phi_{i-1} - \alpha \cdot \triangle \phi_i$

$b_i = b_{i-1} - \beta \cdot \triangle b_i$

end for

Return $m(\lambda N, b_N)$

Gradient

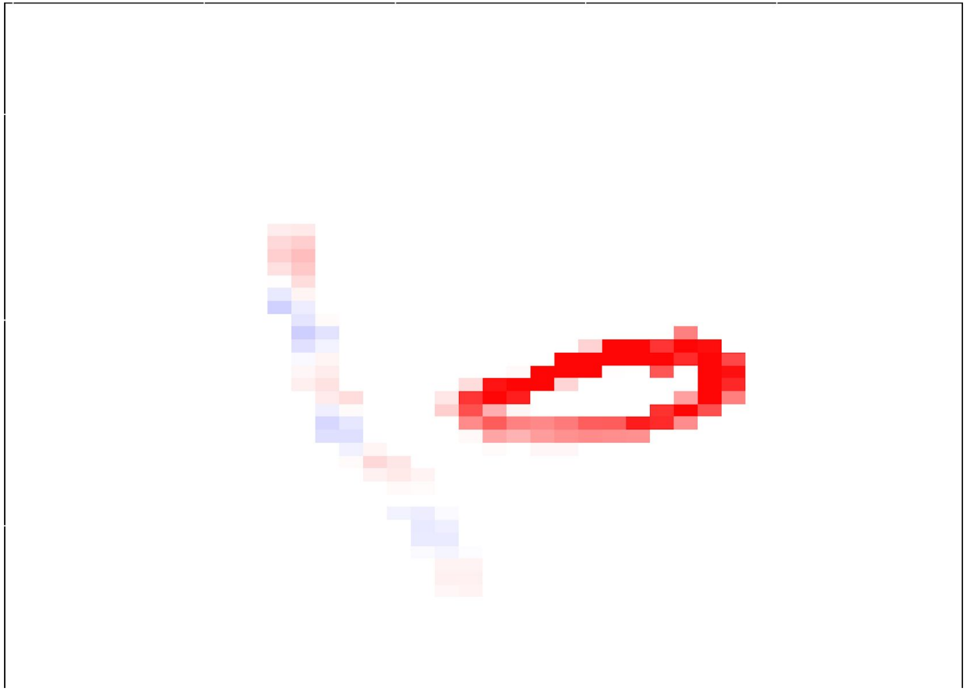


Search direction: Implicit surface

for i in $(1,N)$ **do**
- $d_{\text{syn}}(i) = \text{F}(\phi_i, b_{i-1})$
- $\triangle d_i = d_{\text{obs}} - d_{\text{syn}}(i)$
- $g_i = D^T B^T \triangle d_i$
- **if** EvenNumberedIteration **then**
  - $\triangle\lambda_i = \textbf{CGHessianInv}(g_i)$
  - $\triangle\phi_i = \text{D}(\triangle\lambda_i)$
  - $\triangle b_i = 0$
  - $\alpha = \textbf{linesearch}(\triangle\phi_i)$
  - $\beta = 0$
- **else**
  - $\triangle\phi_i = 0$
  - $\triangle b_i = \textbf{CGHessianInv}(g_i)$
  - $\alpha = 0$
  - $\beta = \textbf{linesearch}(\triangle b_i)$
- **end if**
- $\phi_i = \phi_{i-1} - \alpha \cdot \triangle\phi_i$
- $b_i = b_{i-1} - \beta \cdot \triangle b_i$

end for

Return $m(\lambda N, b_N)$



for i in $(1,N)$ **do**

$d_{\text{syn}}(i) = \text{F}(\phi_i, b_{i-1})$

$\triangle d_i = d_{\text{obs}} - d_{\text{syn}}(i)$

$g_i = D^T B^T \triangle d_i$

if EvenNumberedIteration **then**

$\triangle\lambda_i = \textbf{CGHessianInv}(g_i)$

$\triangle\phi_i = \text{D}(\triangle\lambda_i)$

$\triangle b_i = 0$

$\alpha = \textbf{linesearch}(\triangle\phi_i)$

$\beta = 0$

else

$\triangle\phi_i = 0$

$\triangle b_i = \textbf{CGHessianInv}(g_i)$

$\alpha = 0$

$\beta = \textbf{linesearch}(\triangle b_i)$

end if

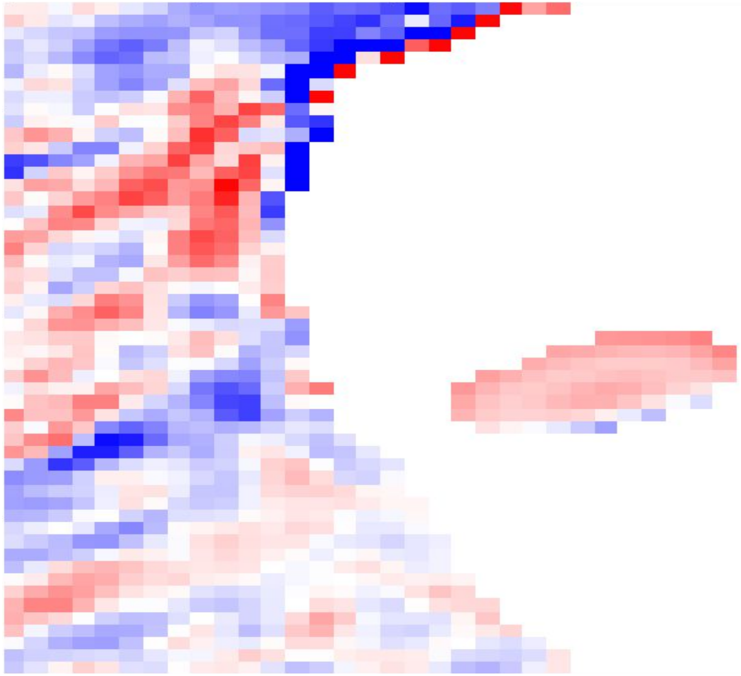
$\phi_i = \phi_{i-1} - \alpha \cdot \triangle\phi_i$

$b_i = b_{i-1} - \beta \cdot \triangle b_i$

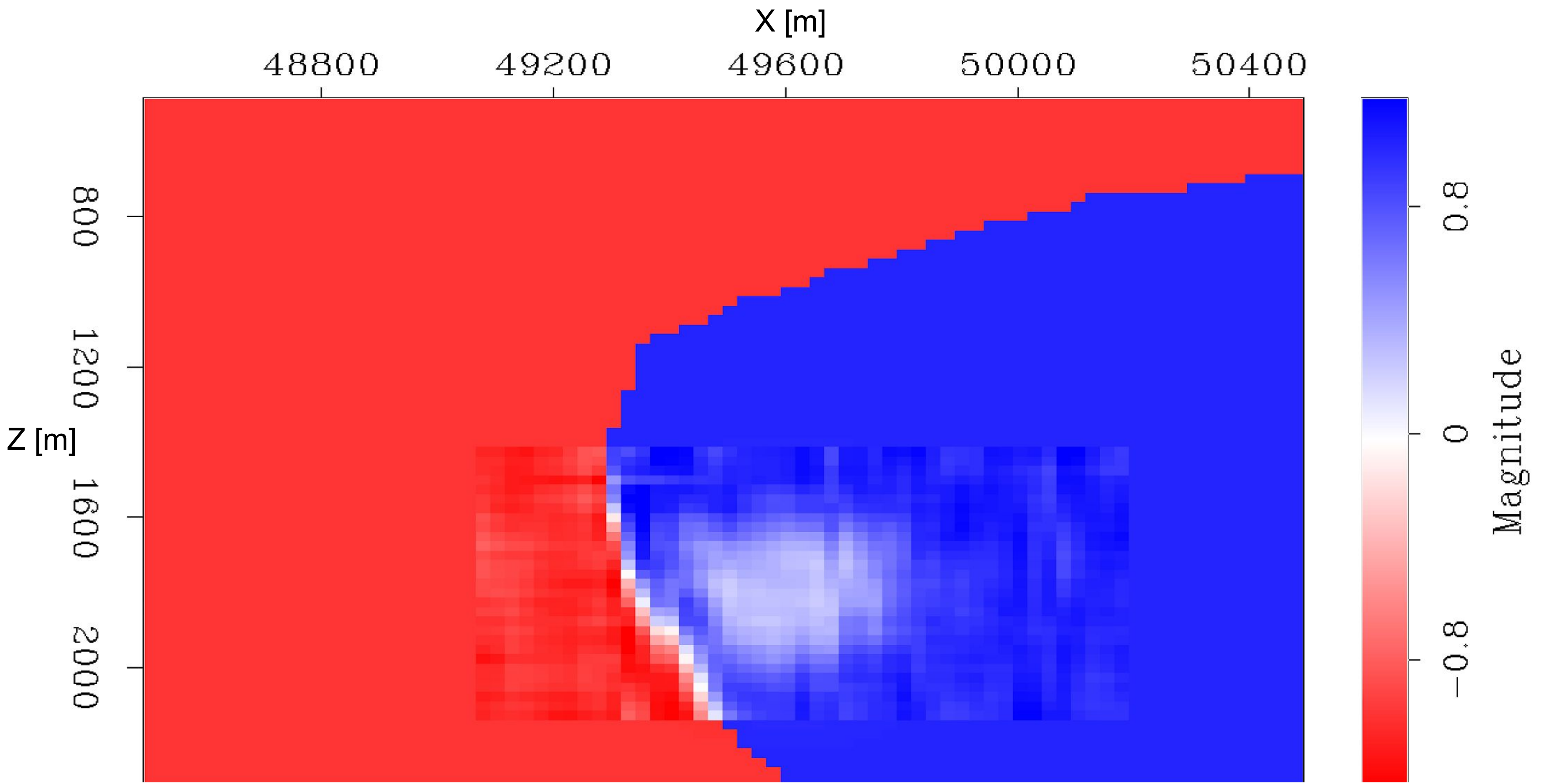
end for

Return $m(\lambda N, b_N)$

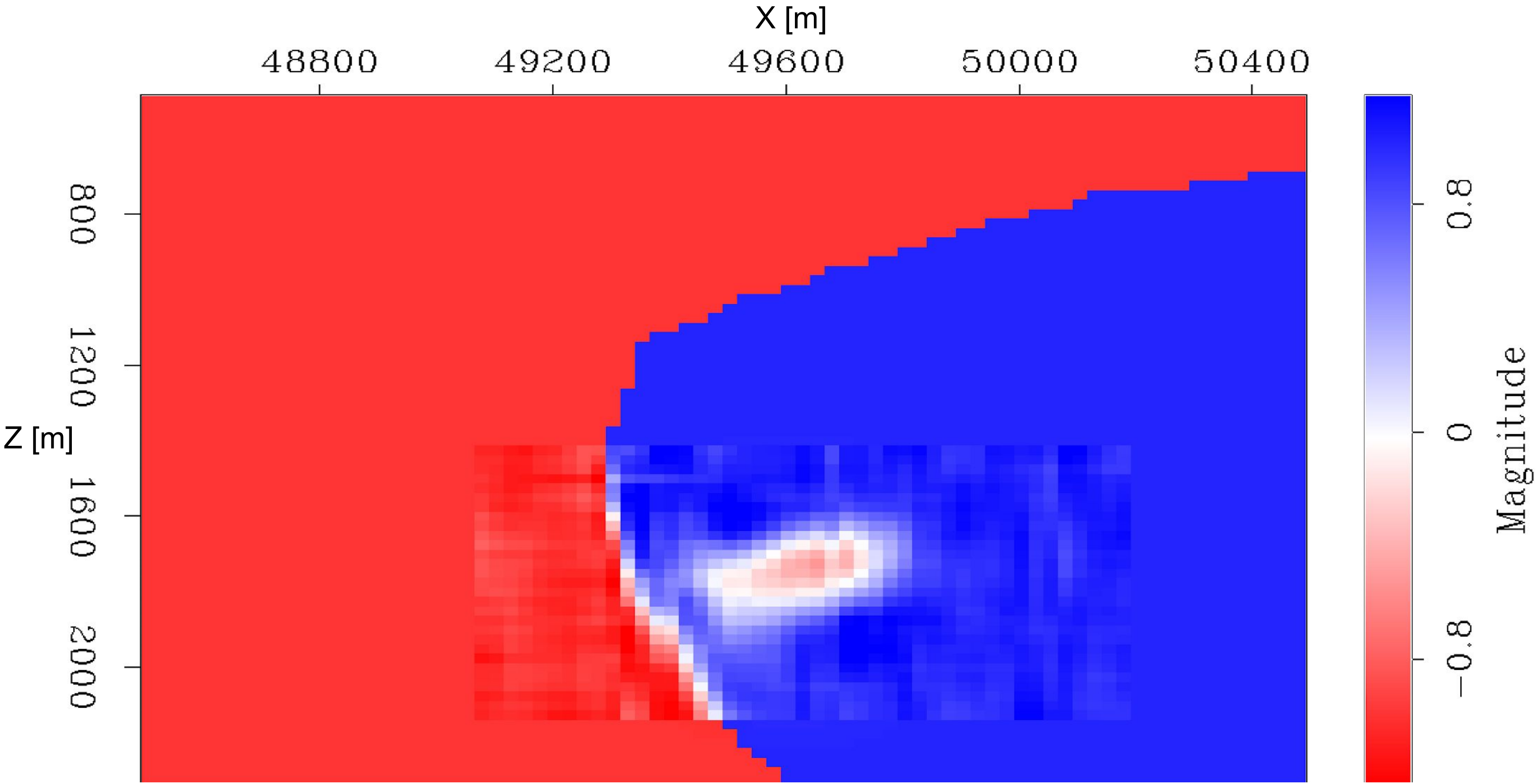
Search direction: Background velocity



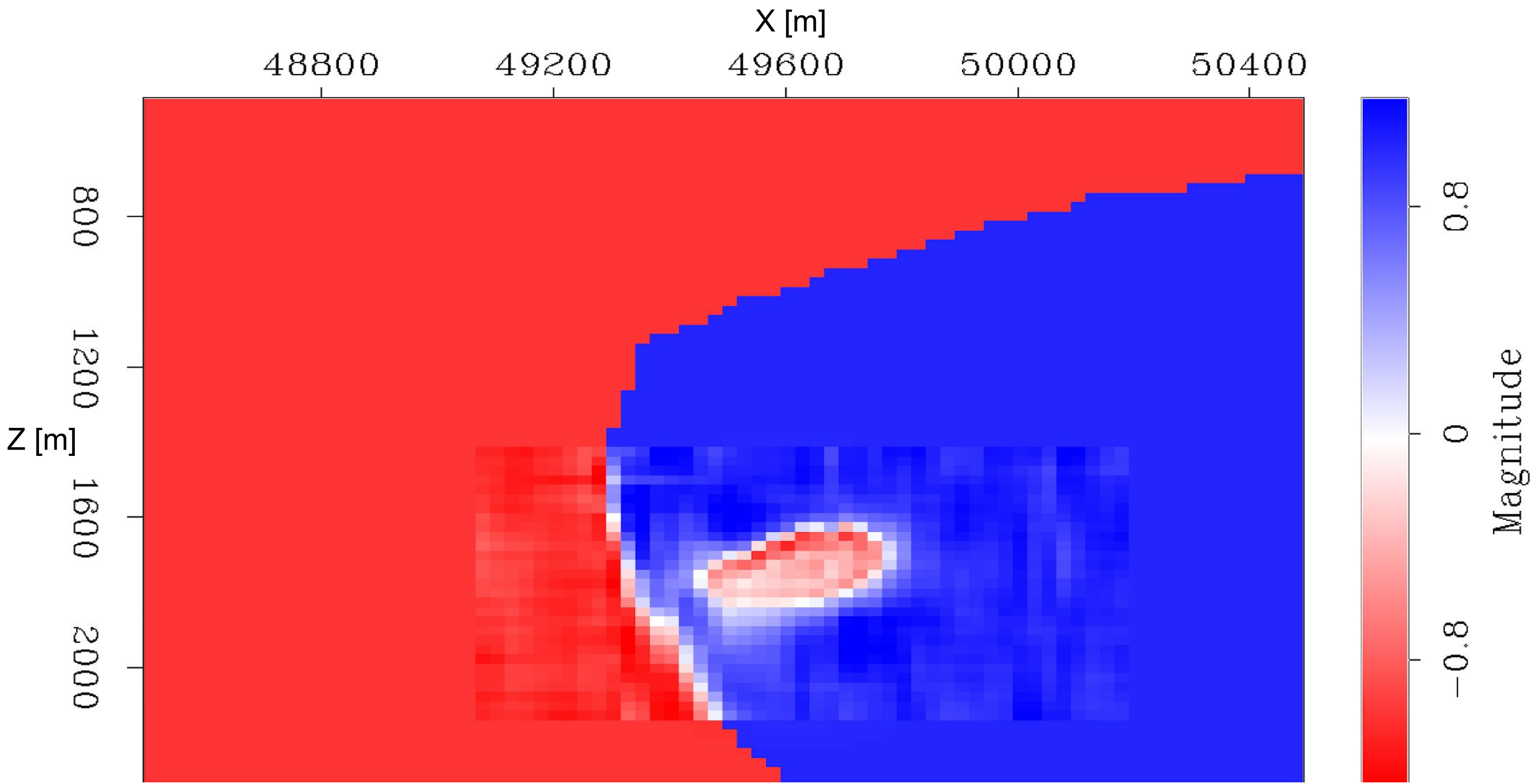
Implicit surface iteration=0



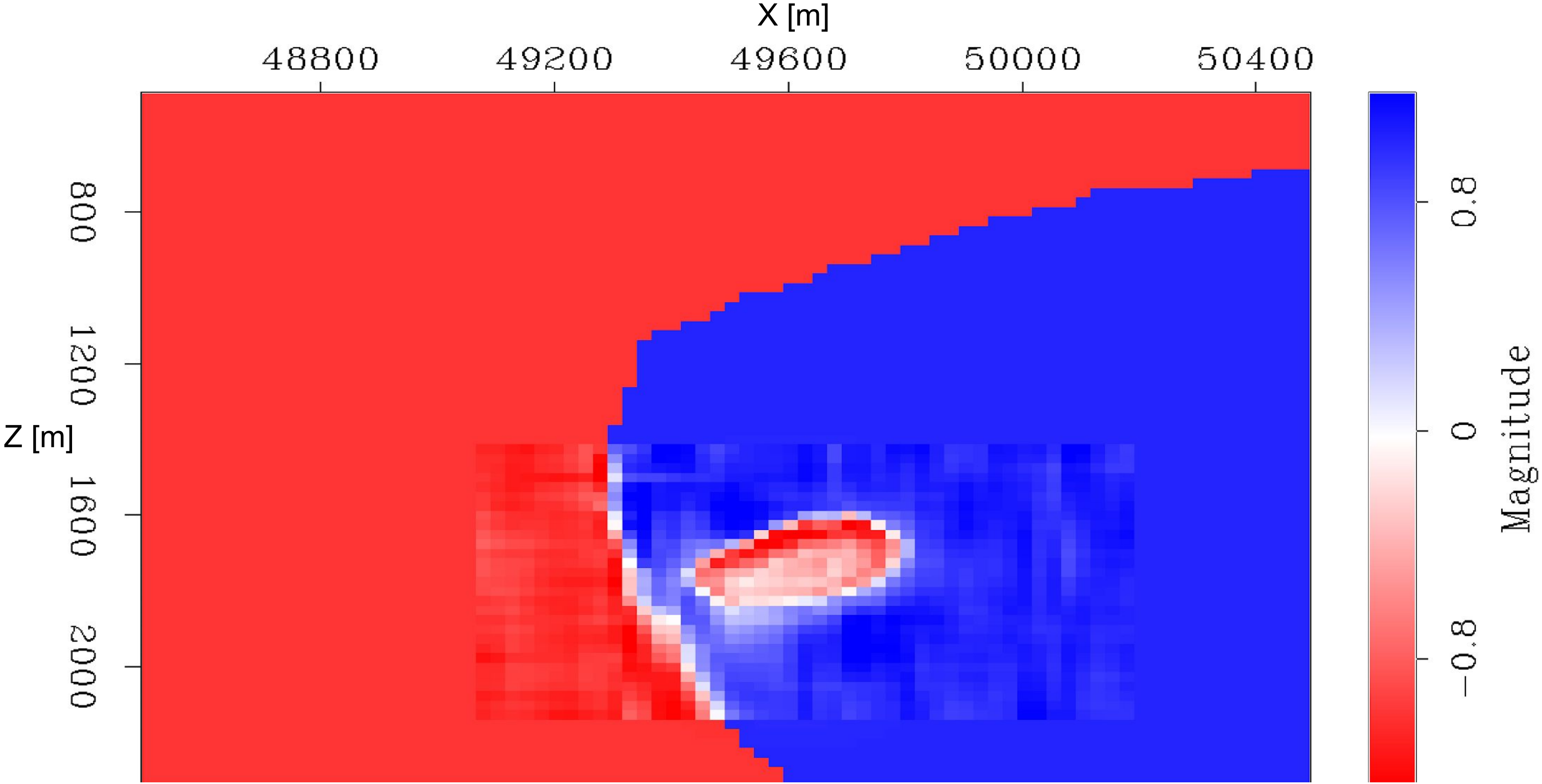
Implicit surface iteration=1



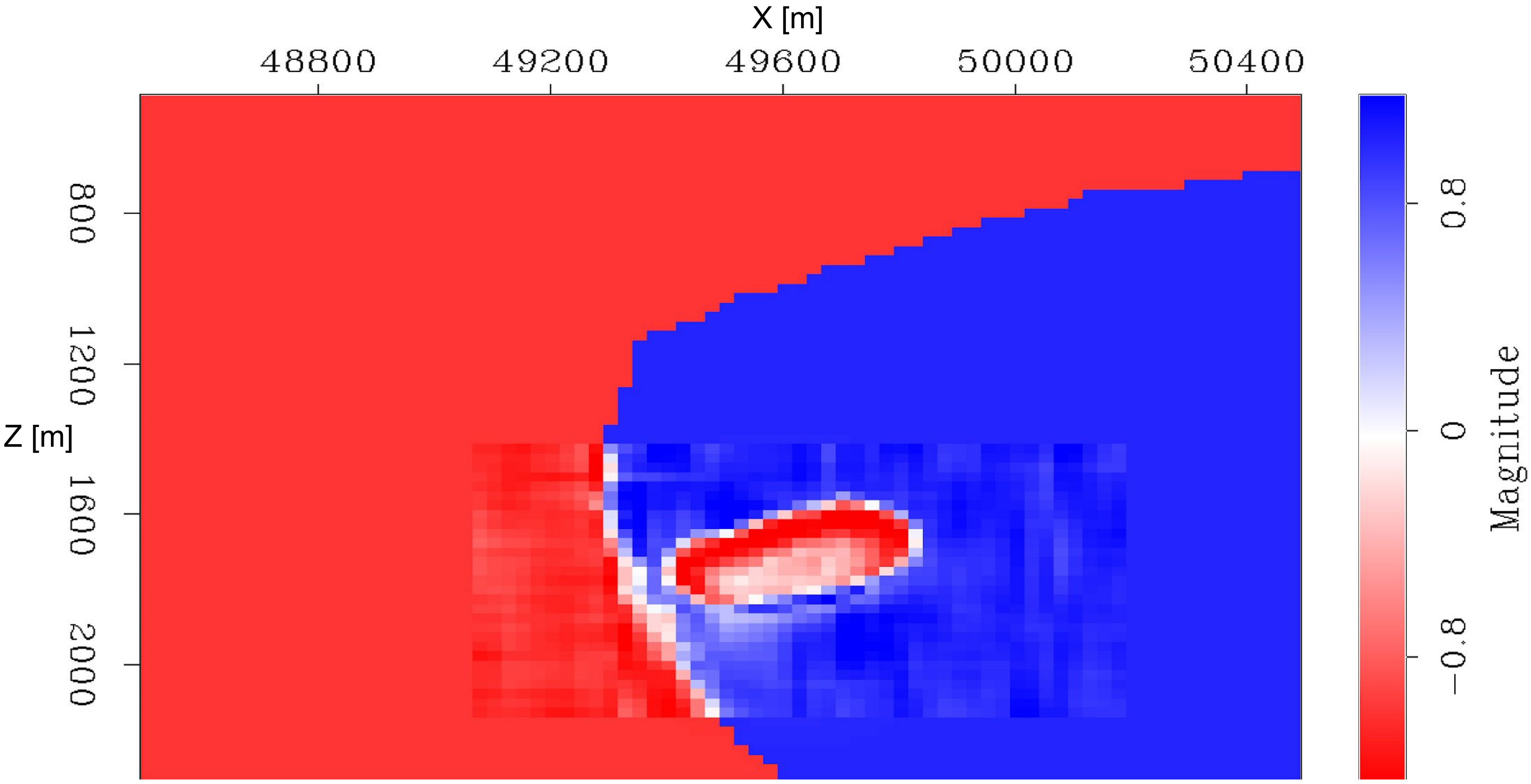
Implicit surface iteration=5



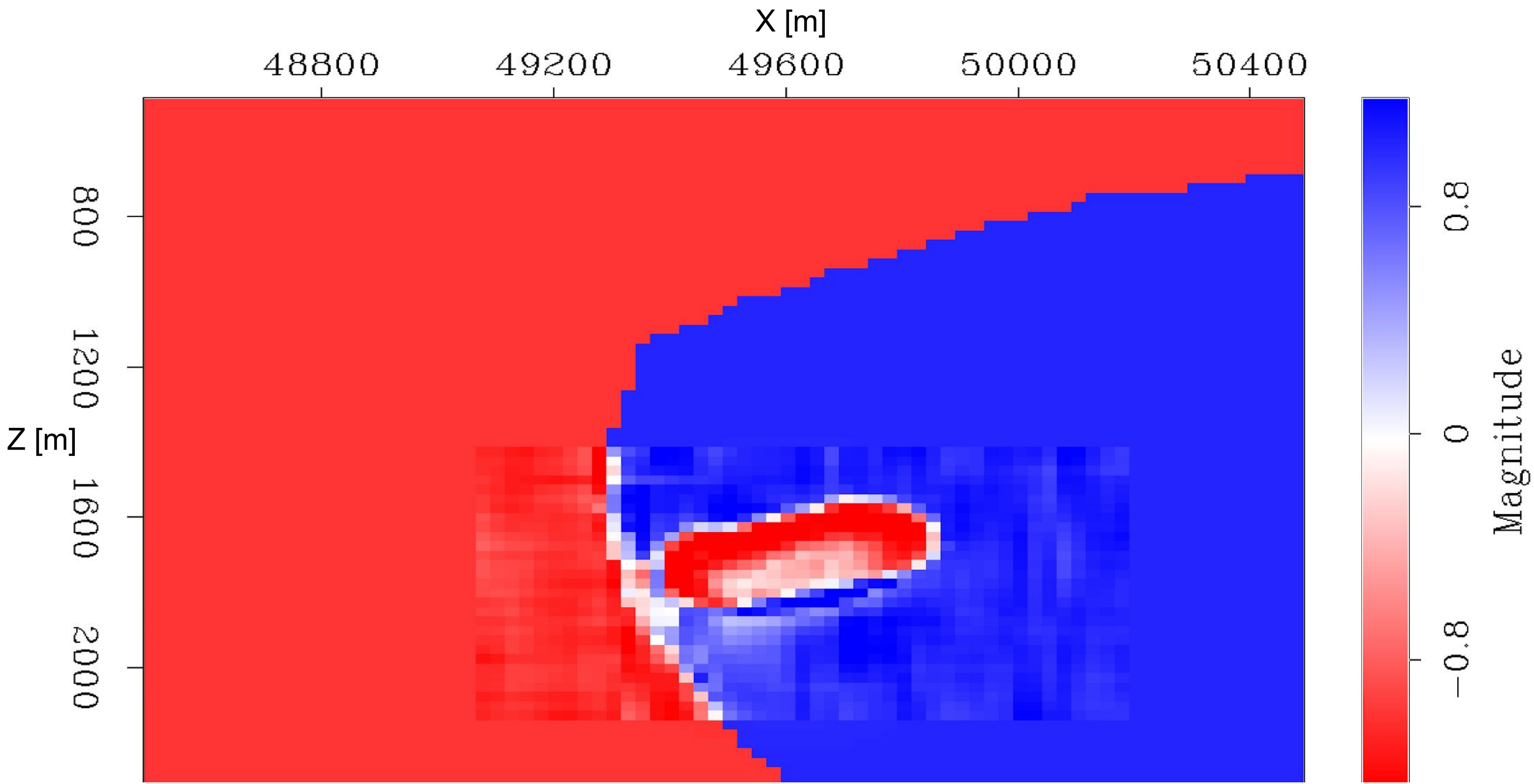
Implicit surface iteration=10



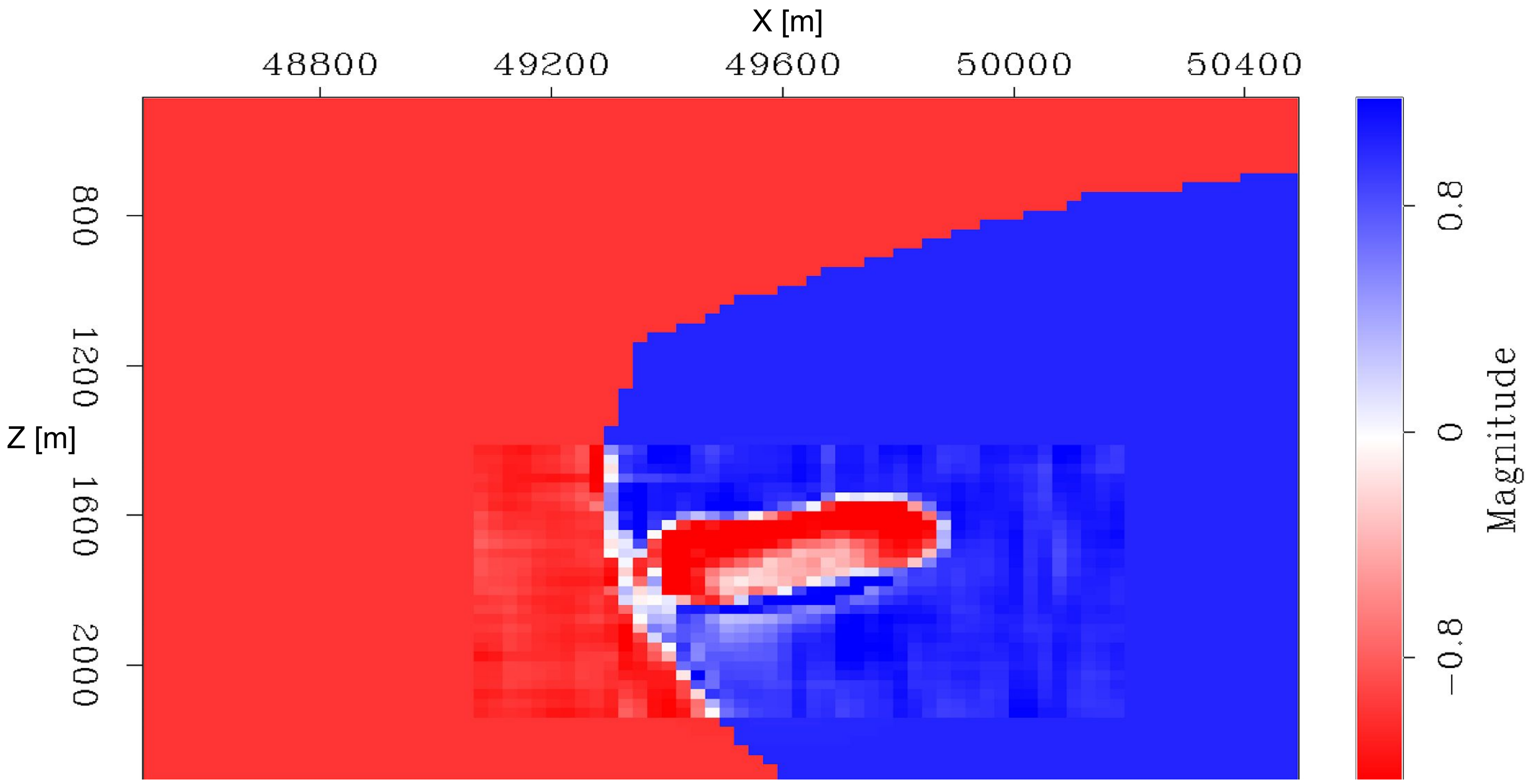
Implicit surface iteration=20



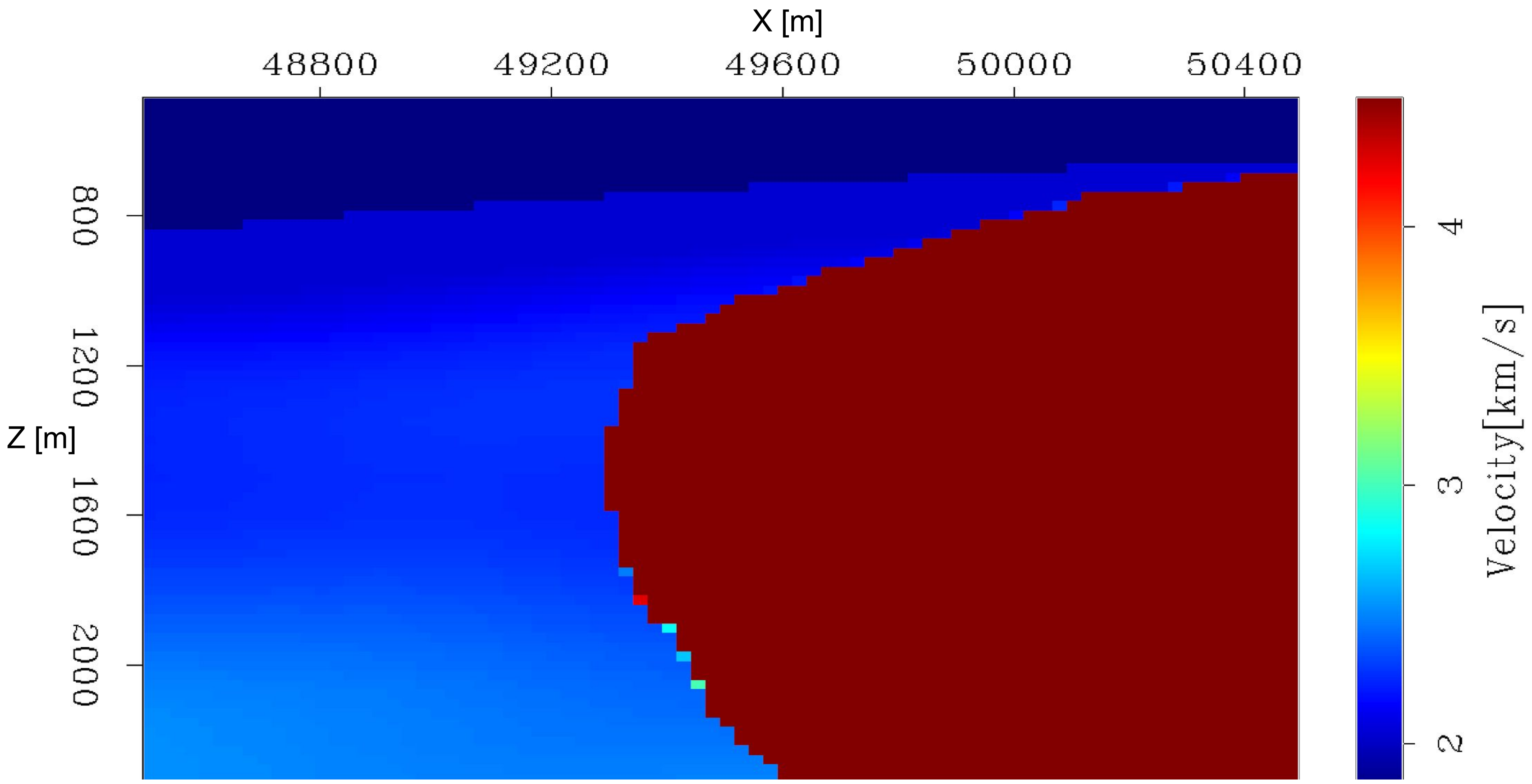
Implicit surface iteration=30



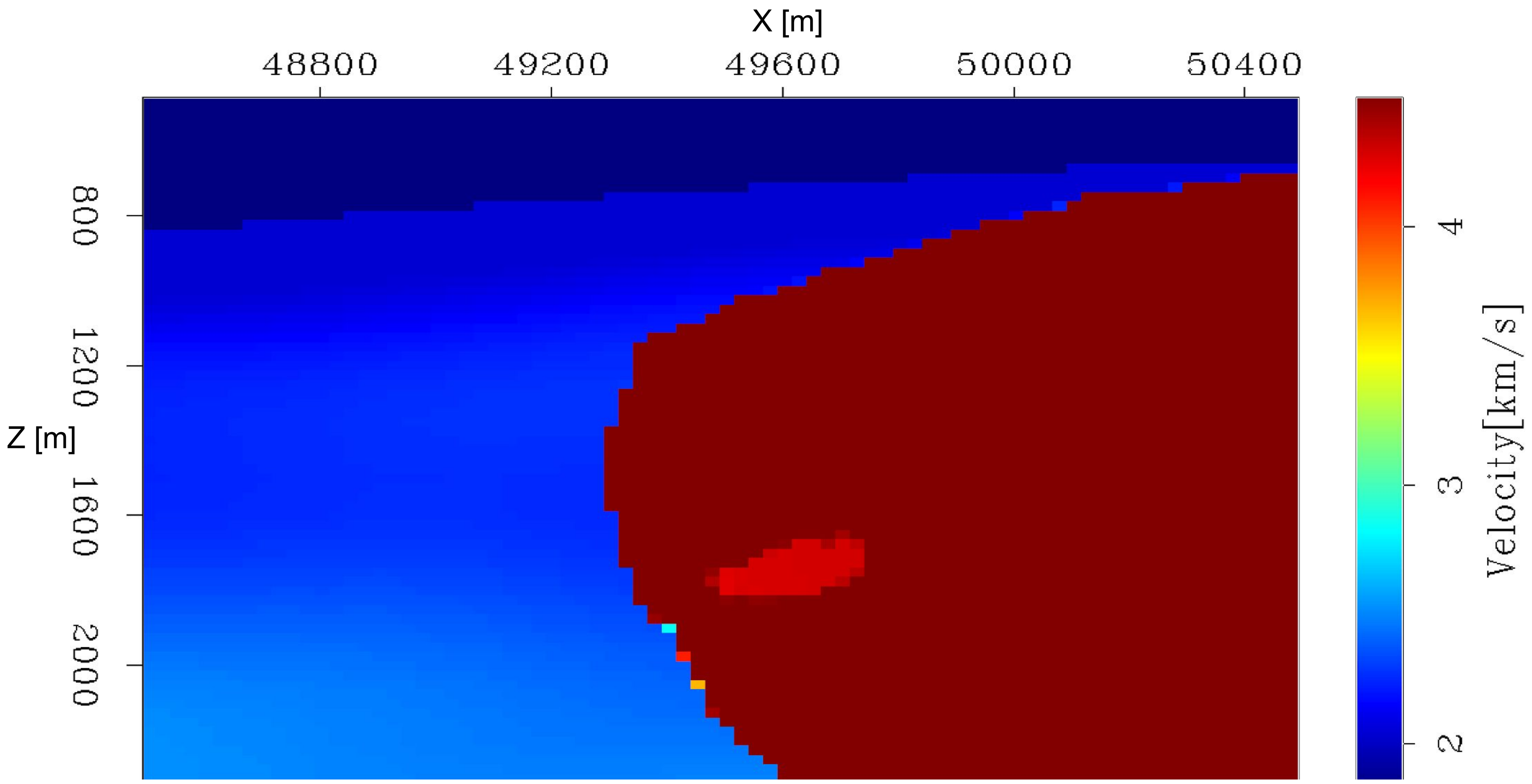
Implicit surface iteration=35



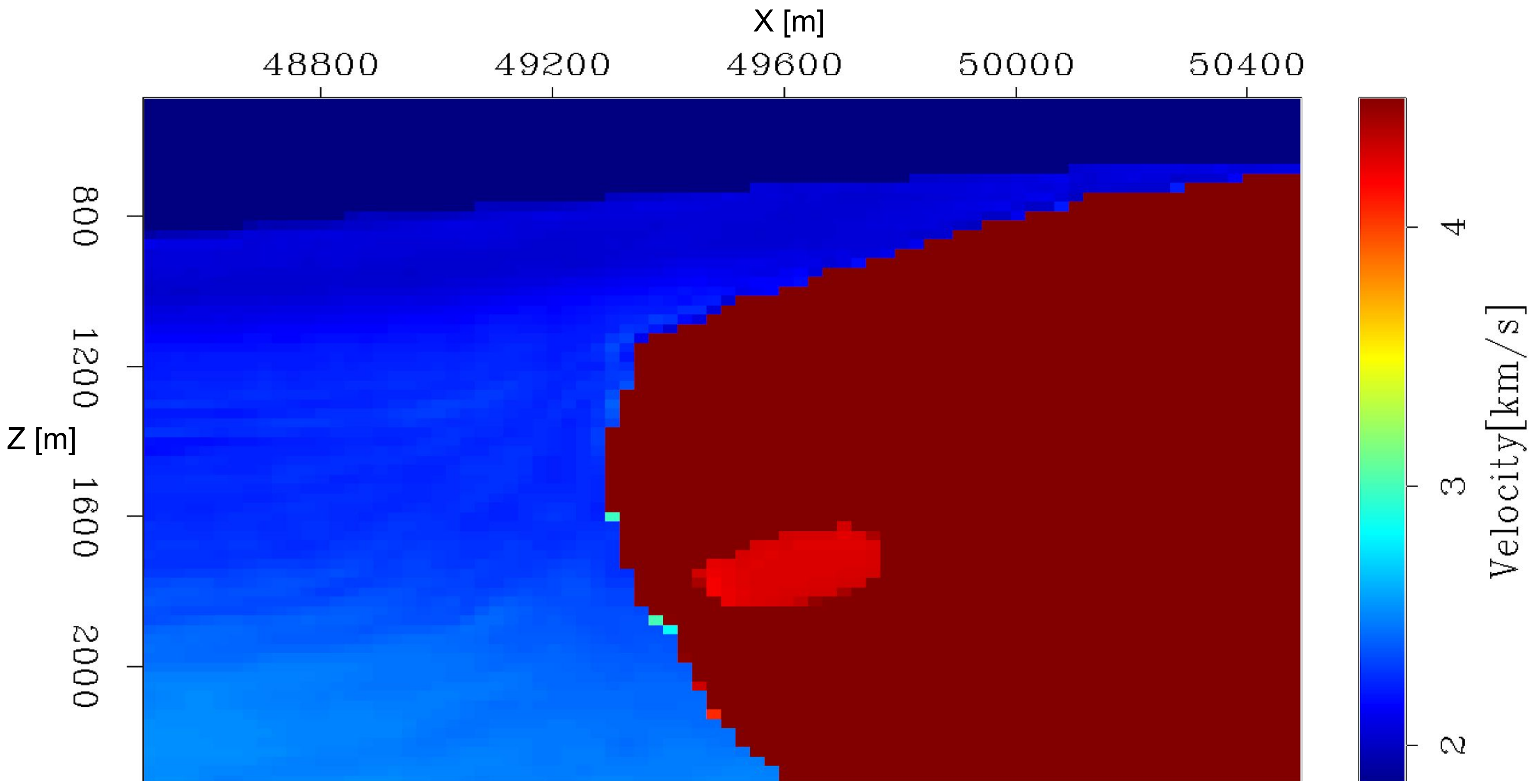
Velocity model iteration=0



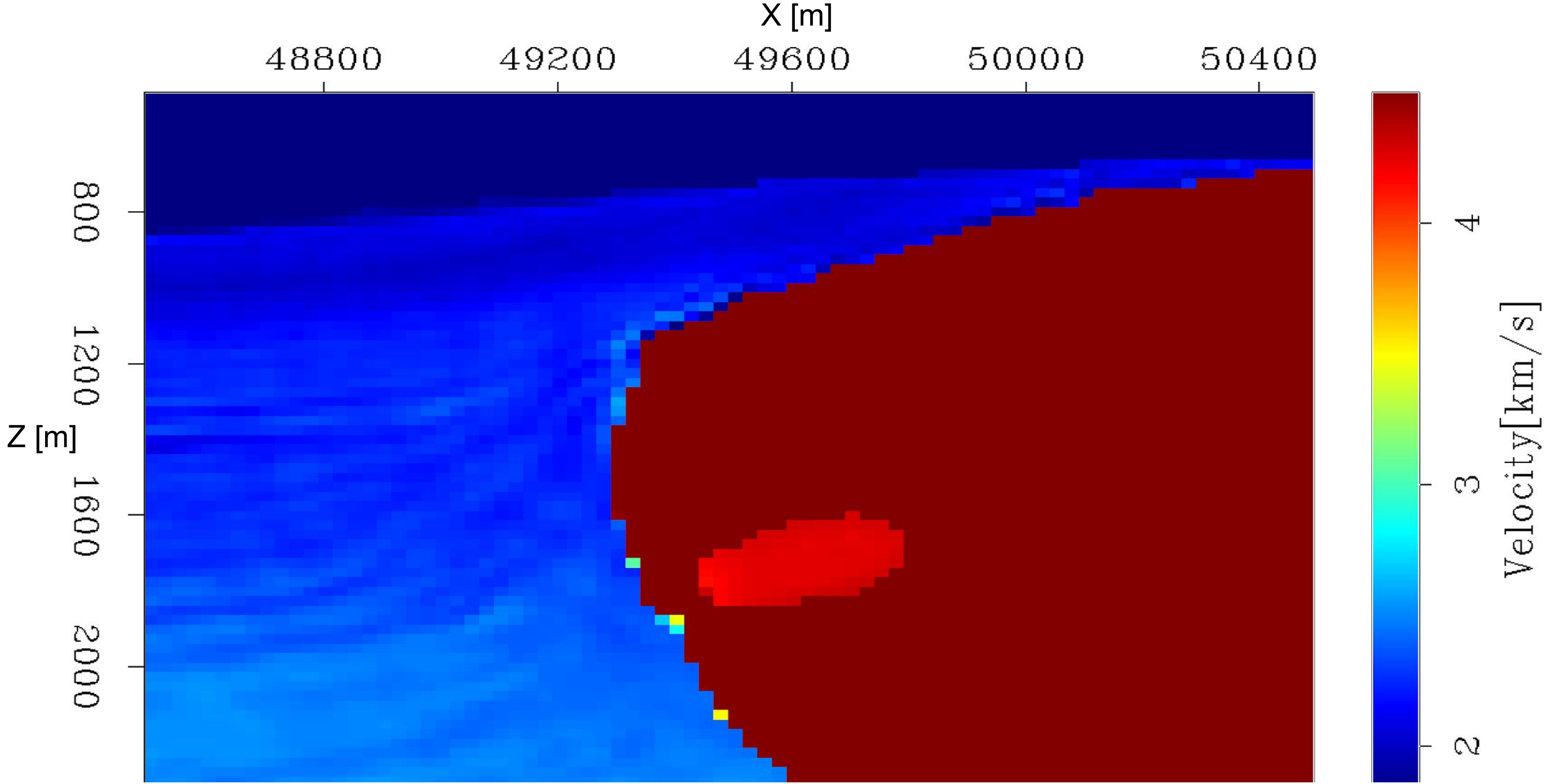
Velocity model iteration=1



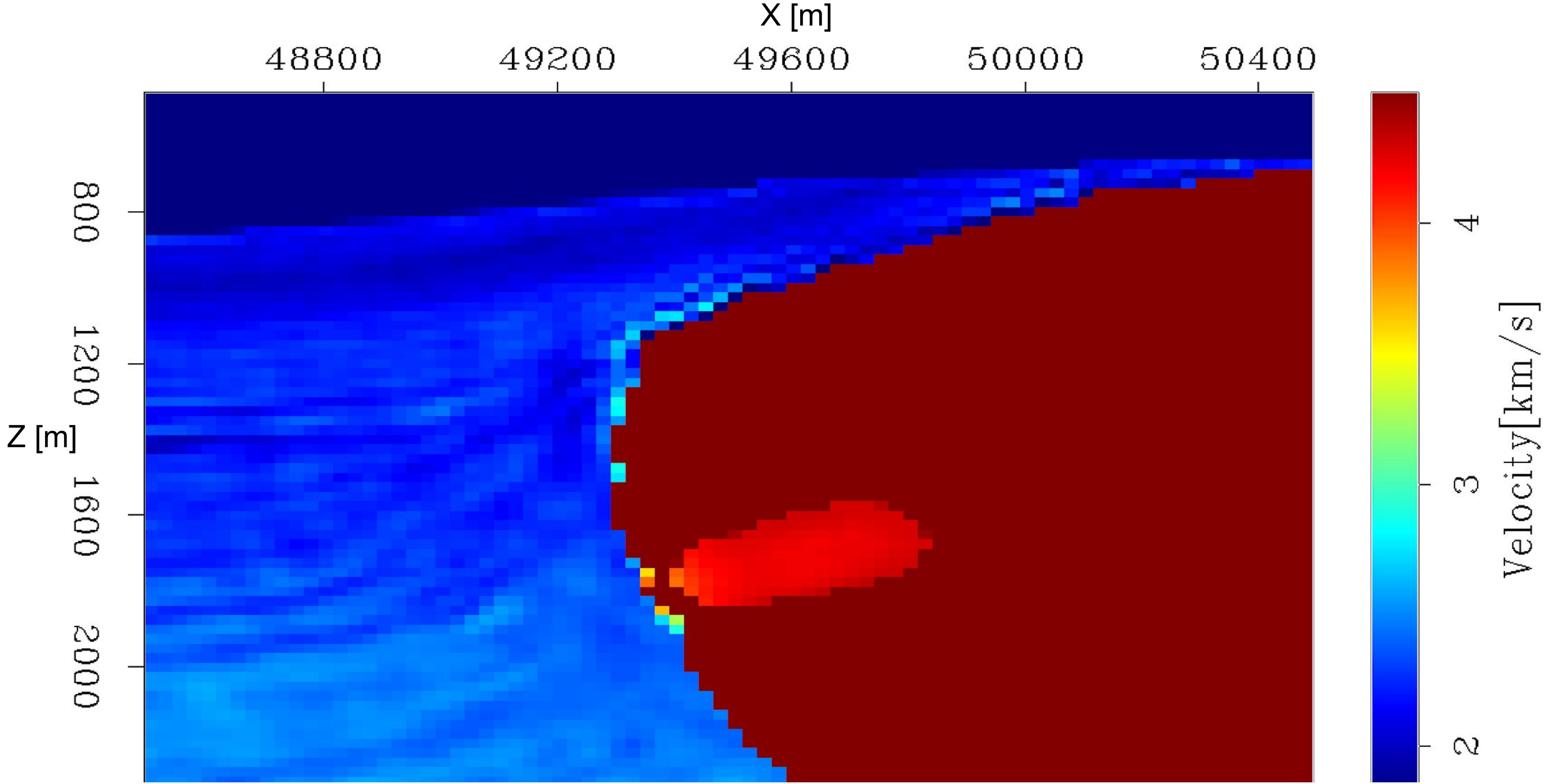
Velocity model iteration=5



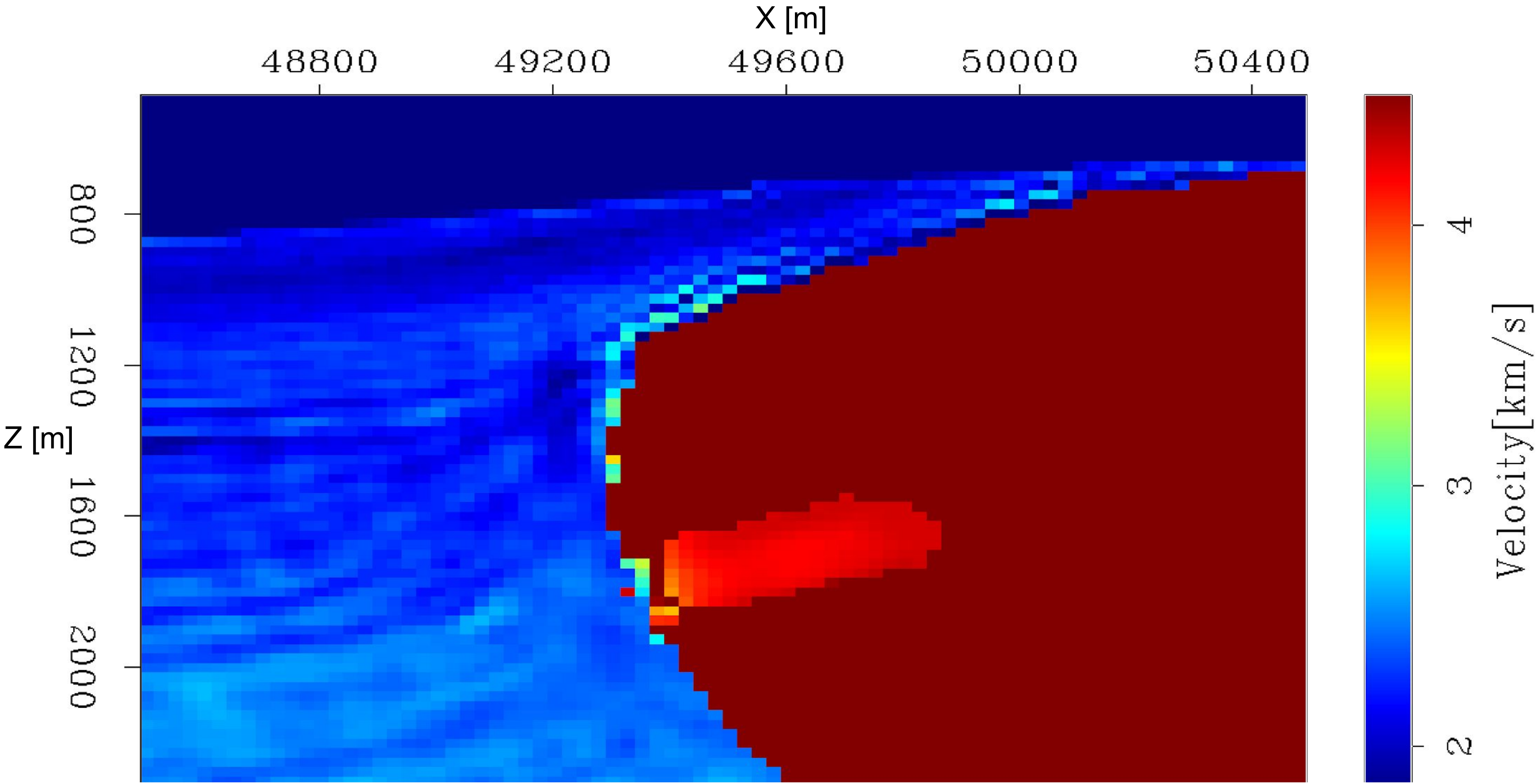
Velocity model iteration=10



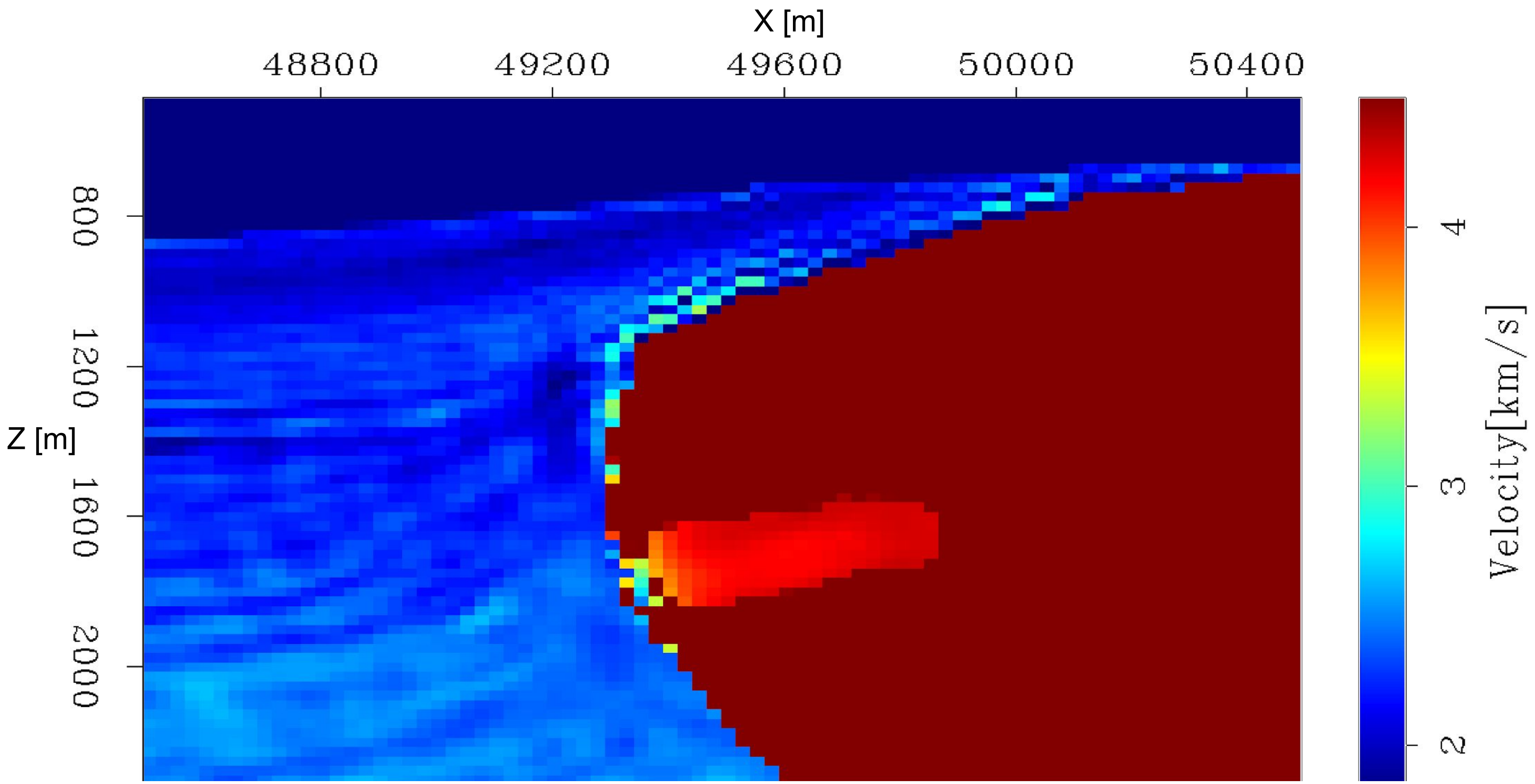
Velocity model iteration=20



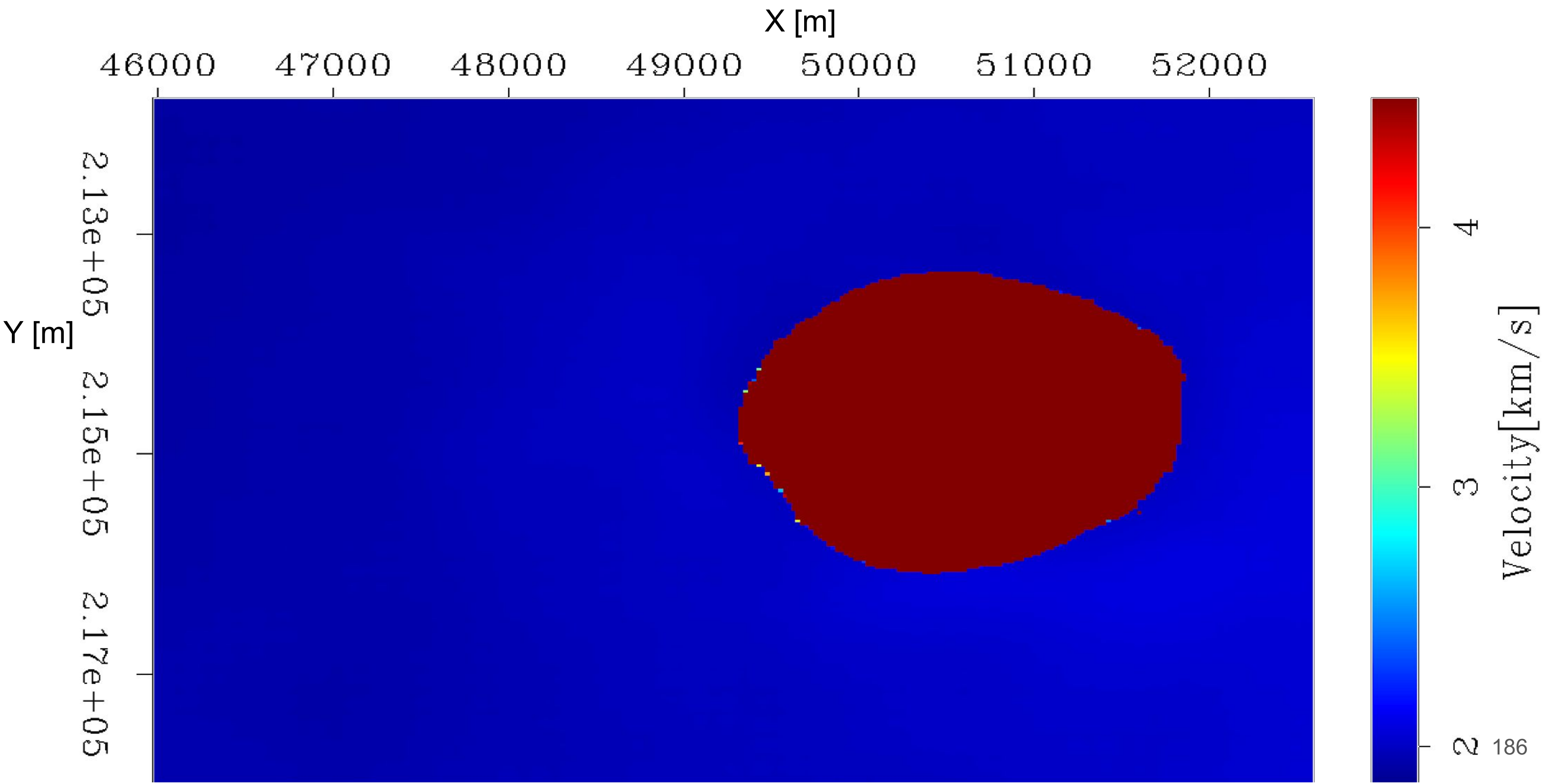
Velocity model iteration=30



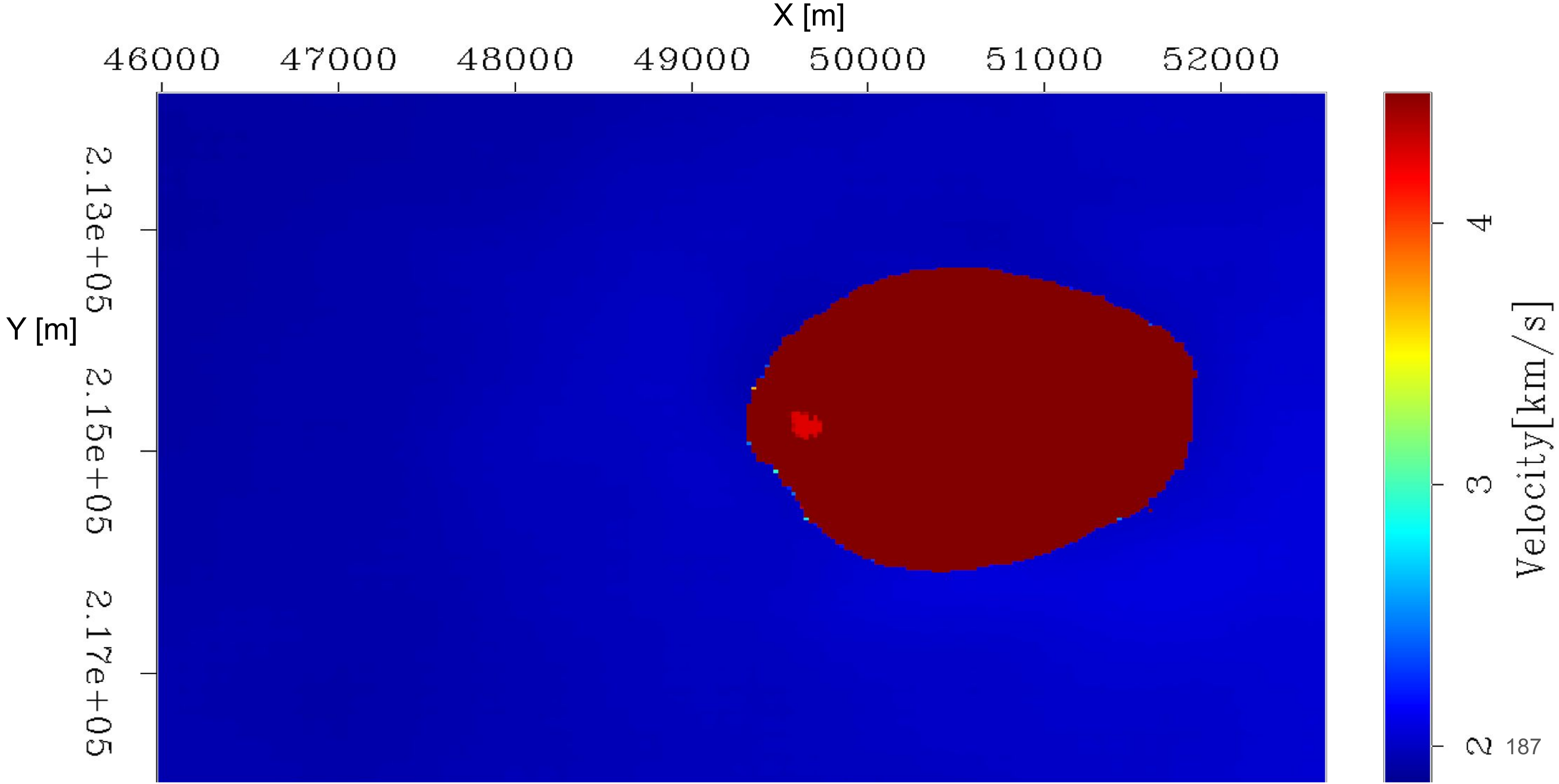
Velocity model iteration=35



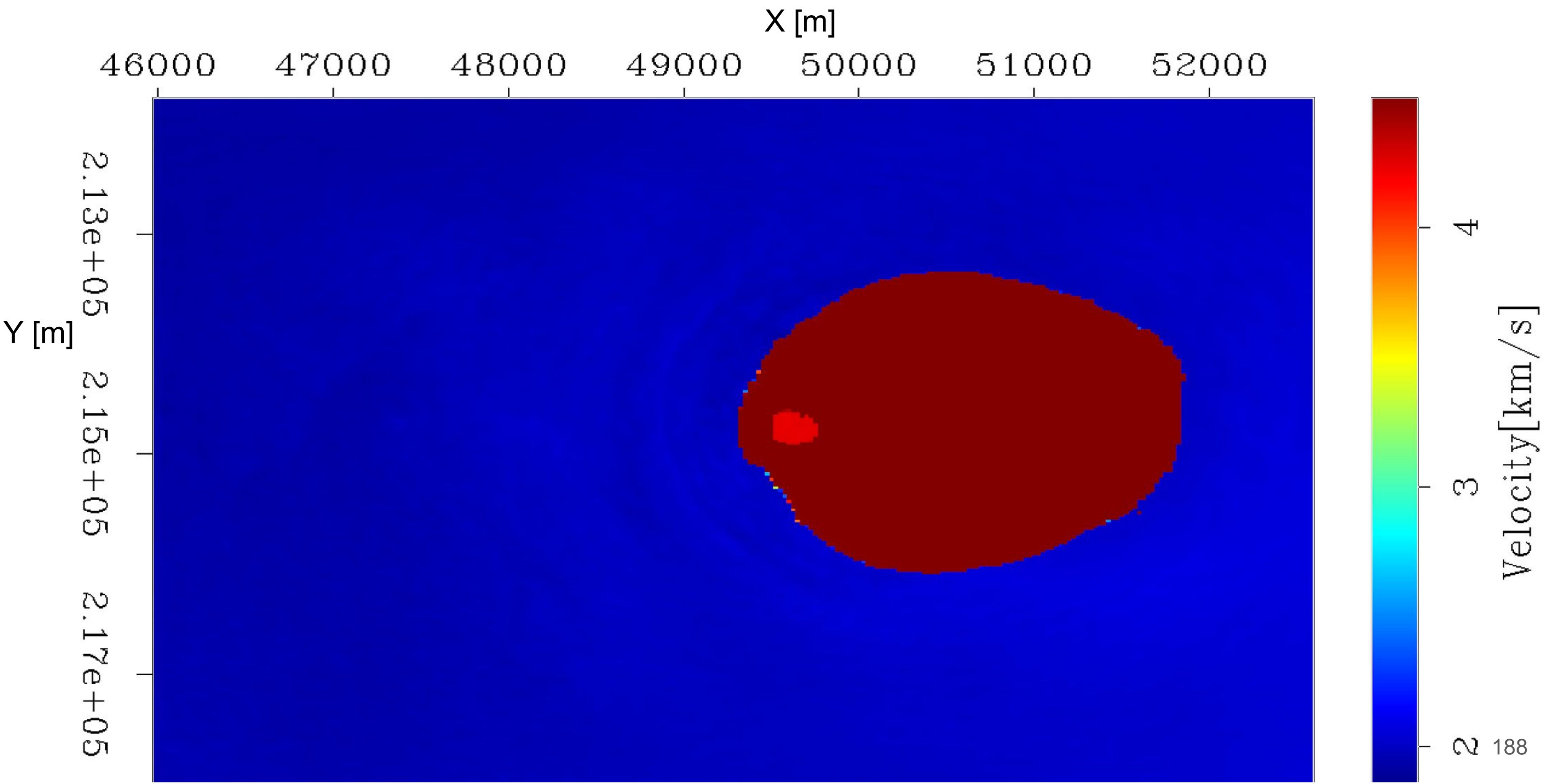
Velocity model iteration=0



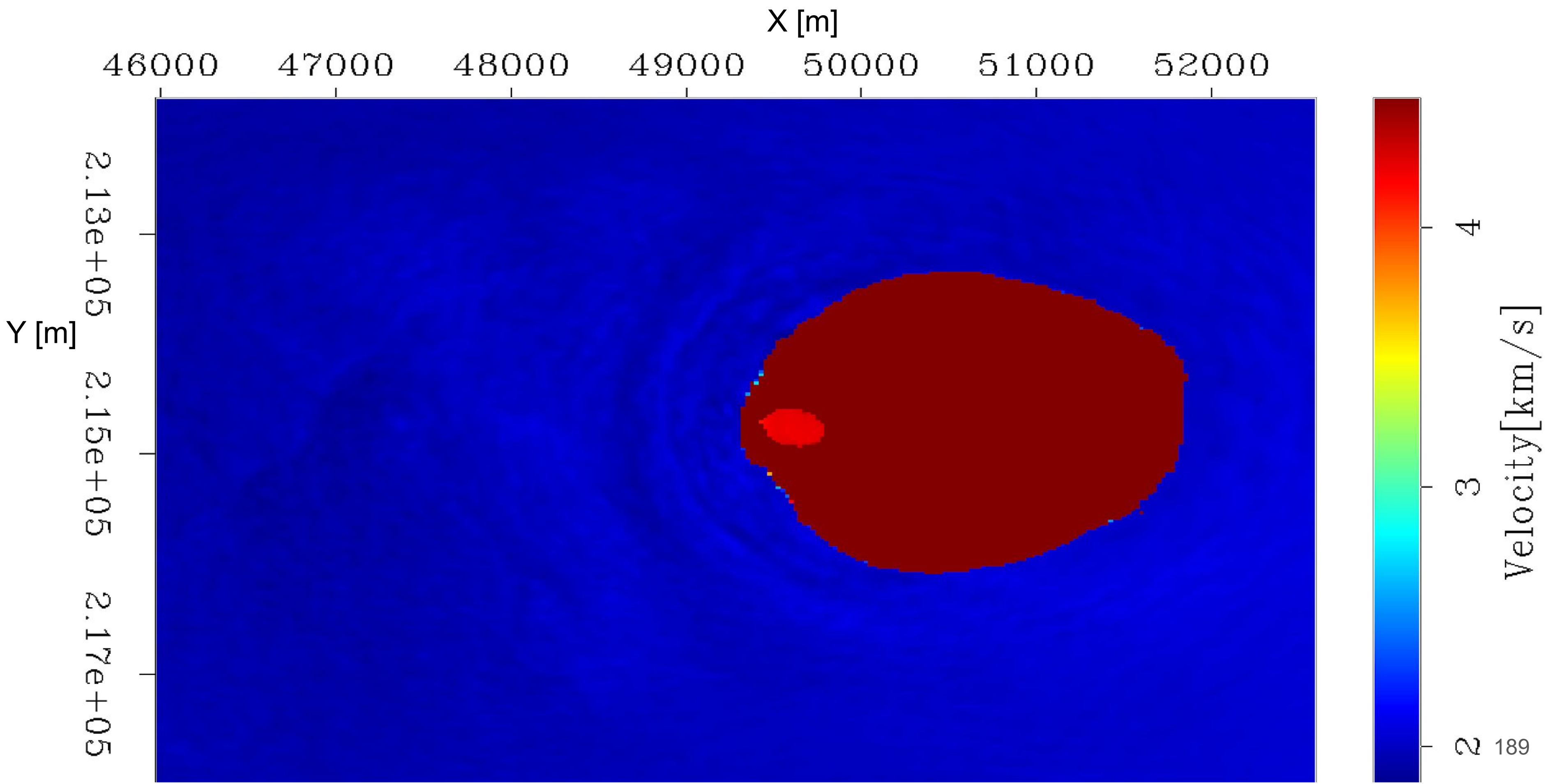
Velocity model iteration=1



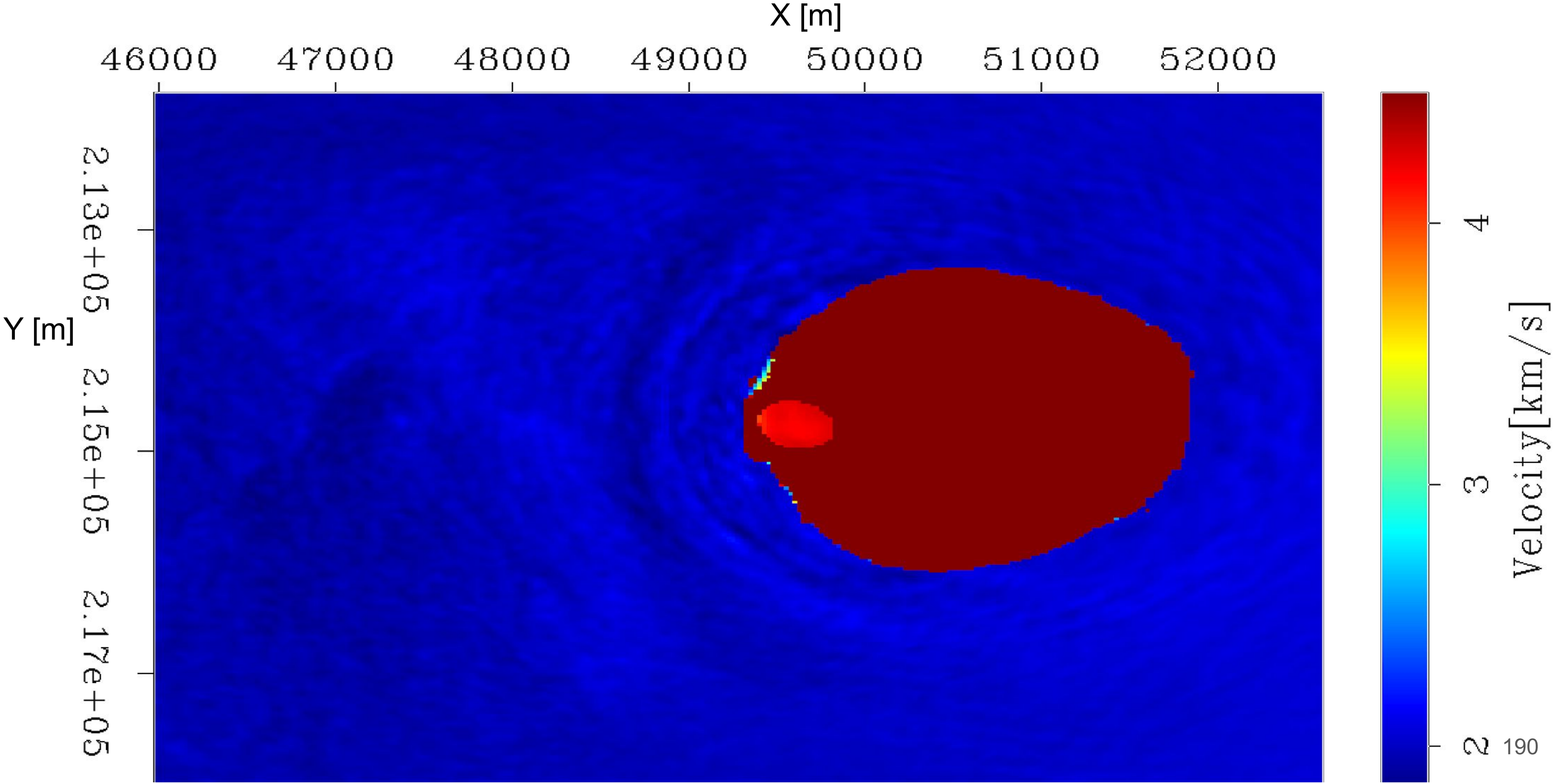
Velocity model iteration=5



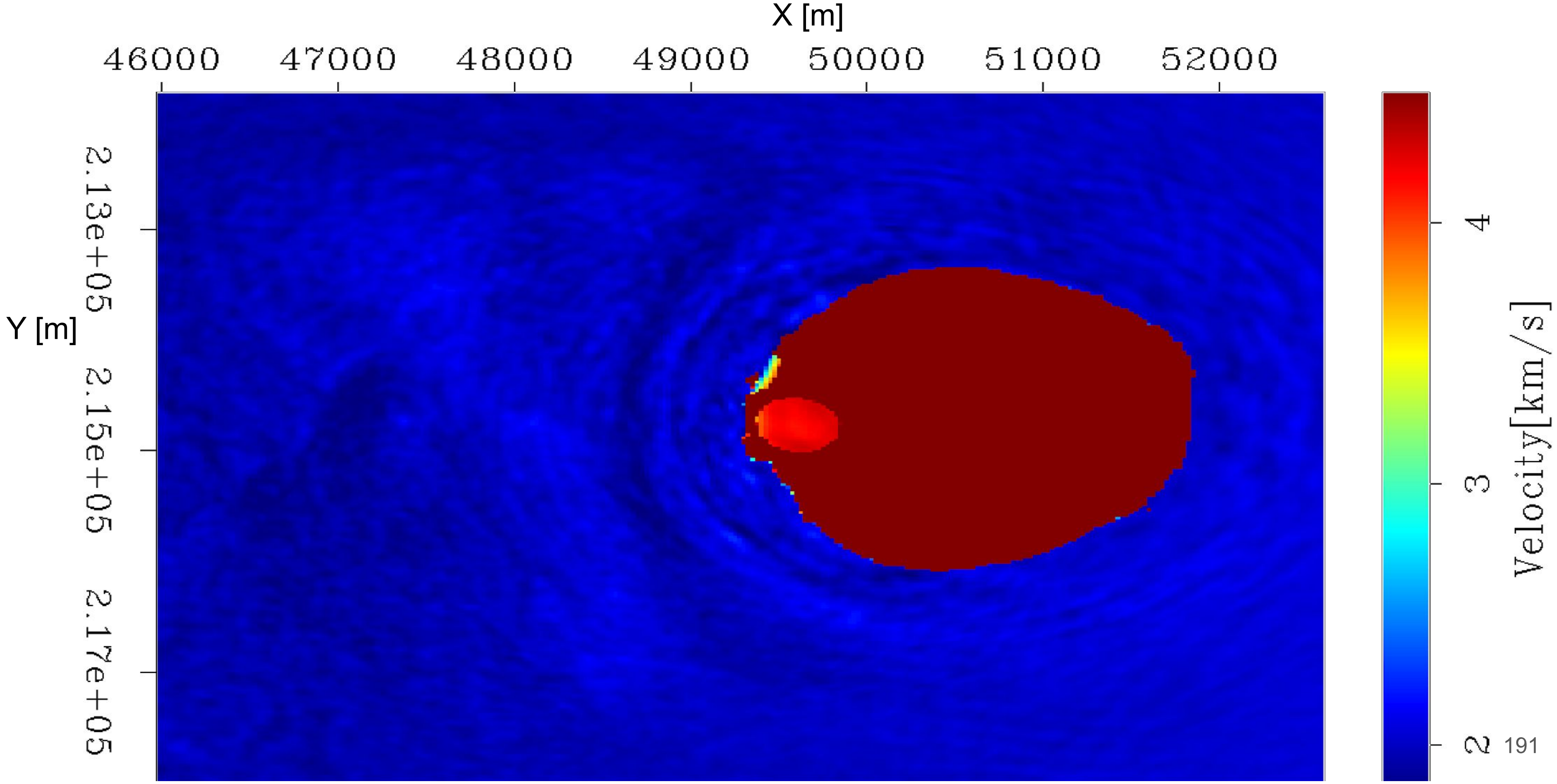
Velocity model iteration=10



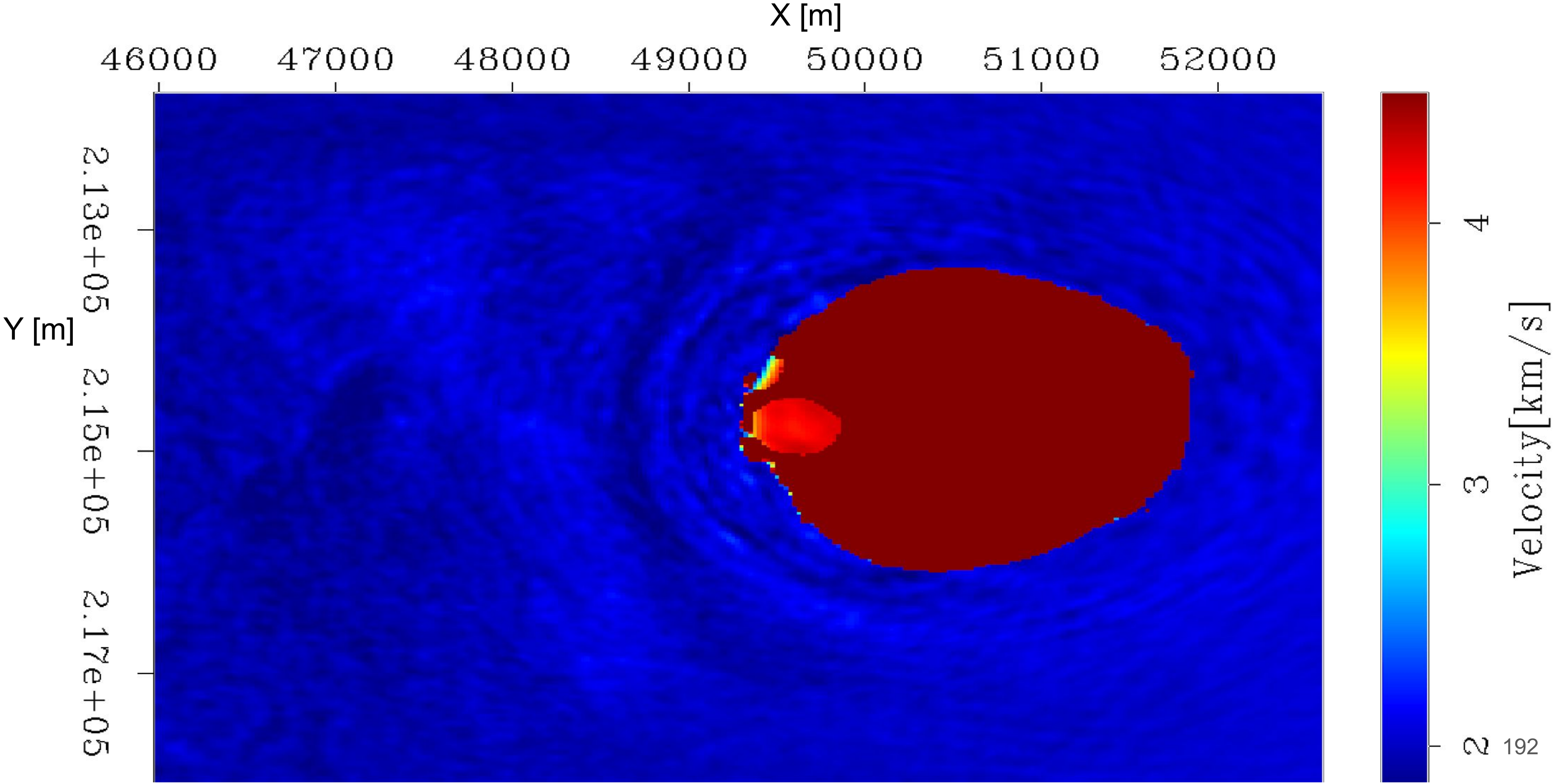
Velocity model iteration=20



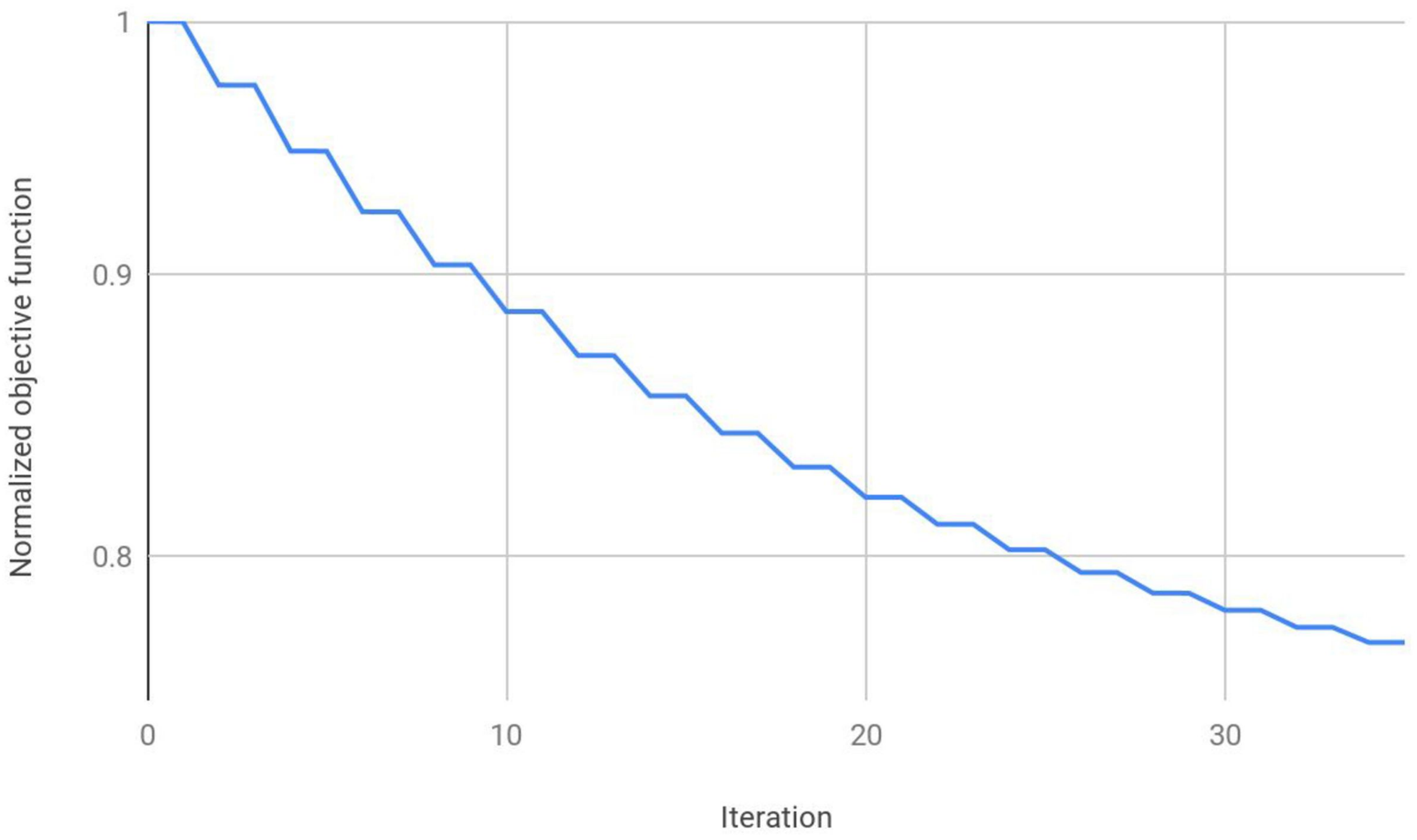
Velocity model iteration=30



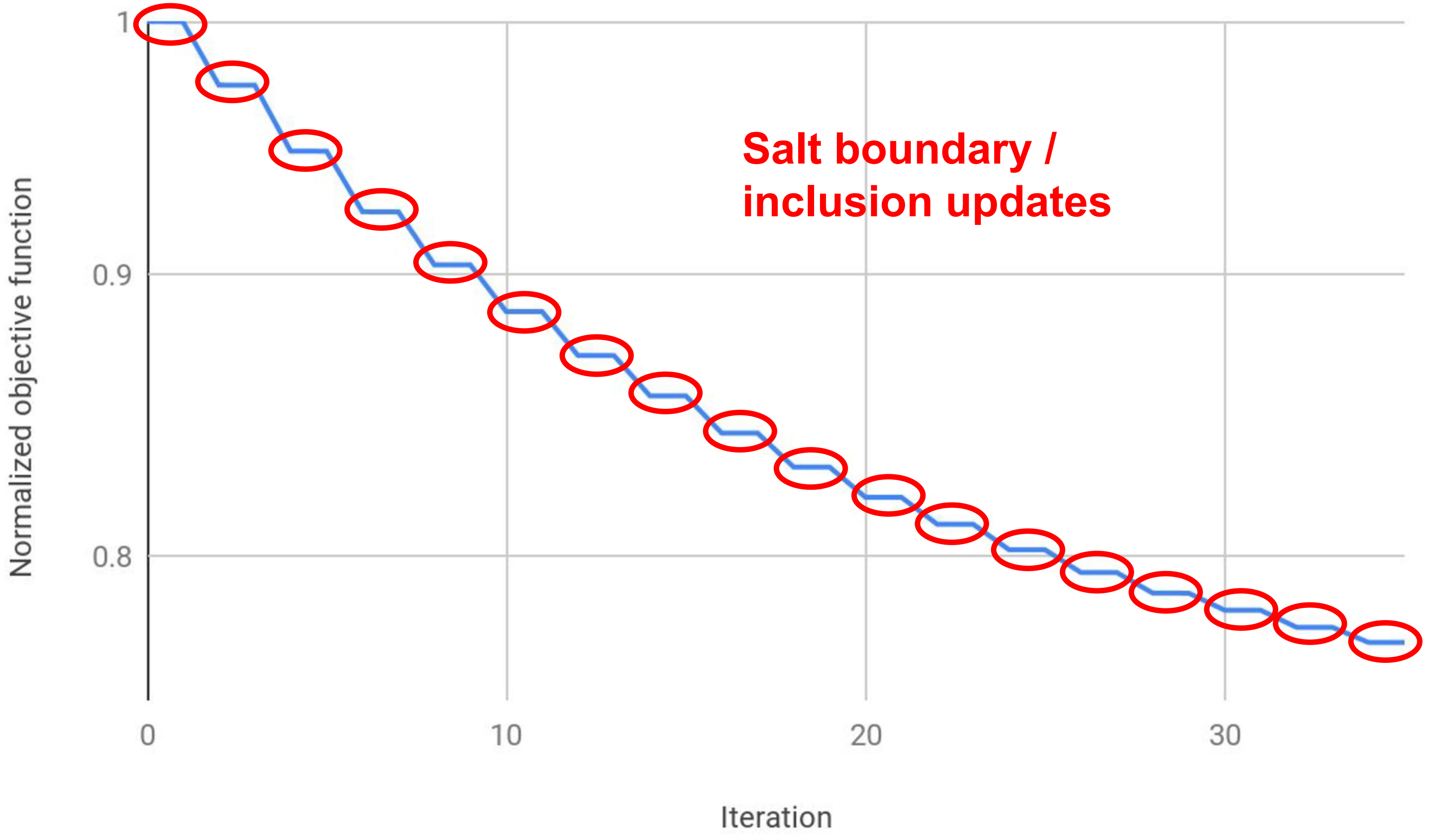
Velocity model iteration=35



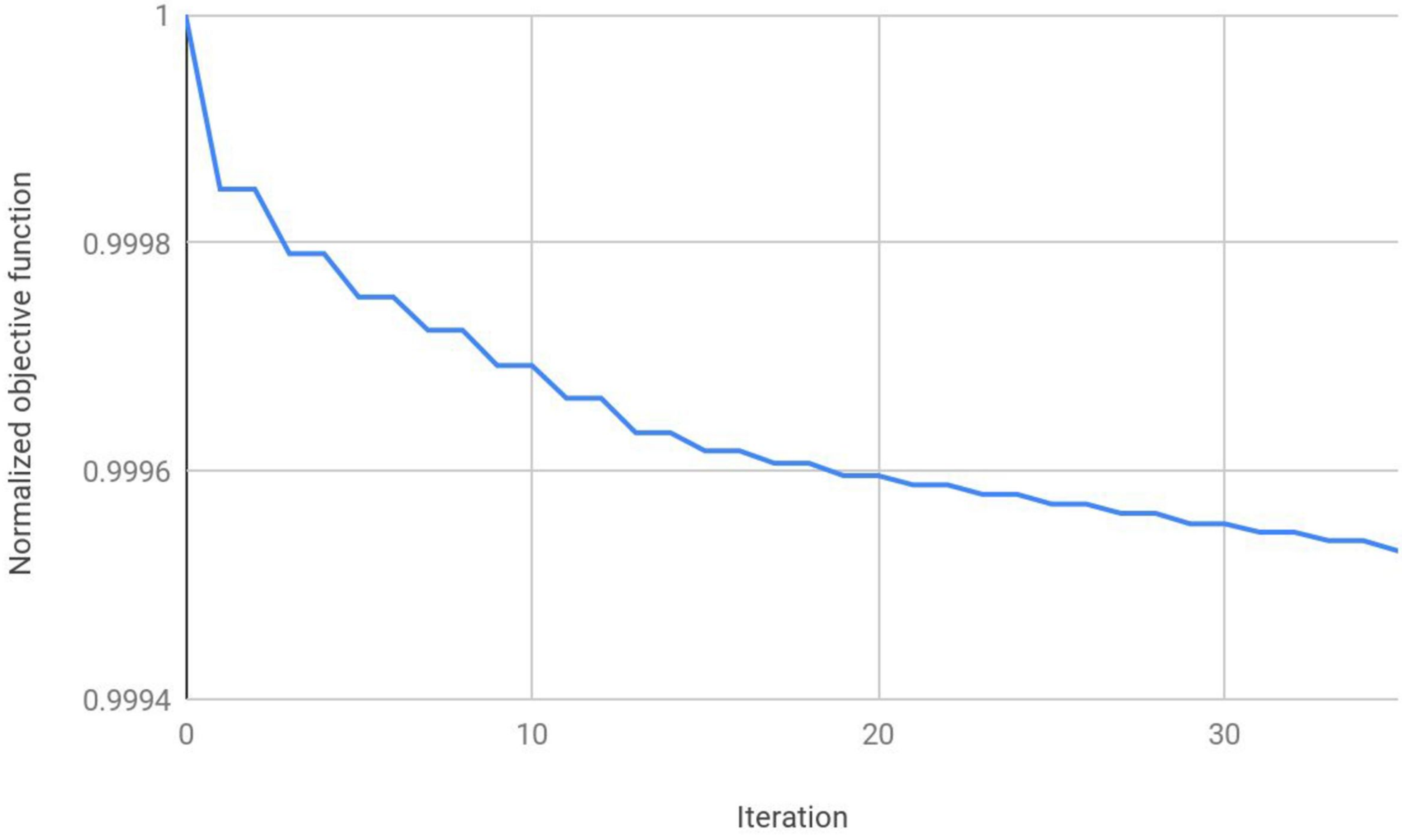
TOTAL OBJECTIVE FUNCTION



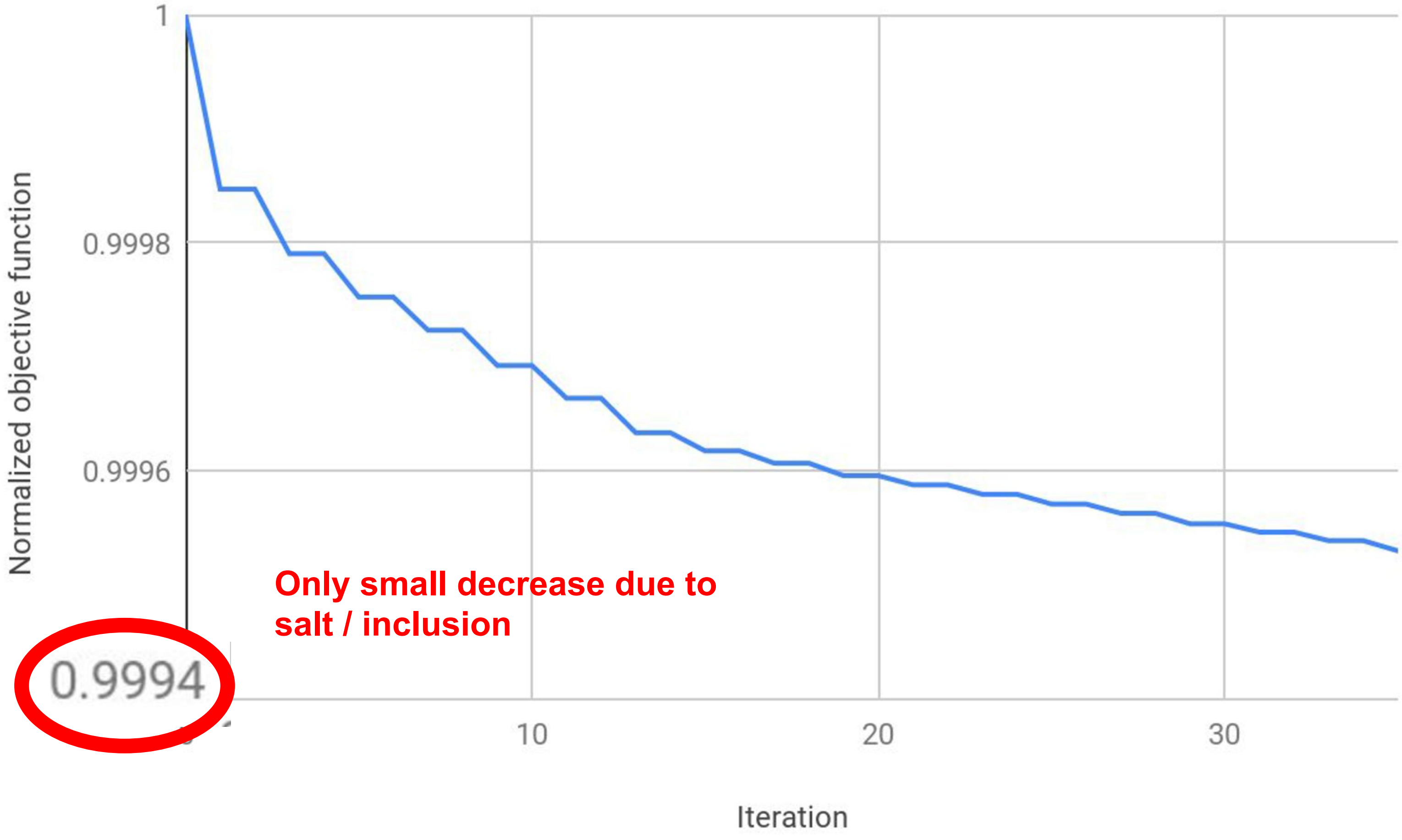
TOTAL OBJECTIVE FUNCTION

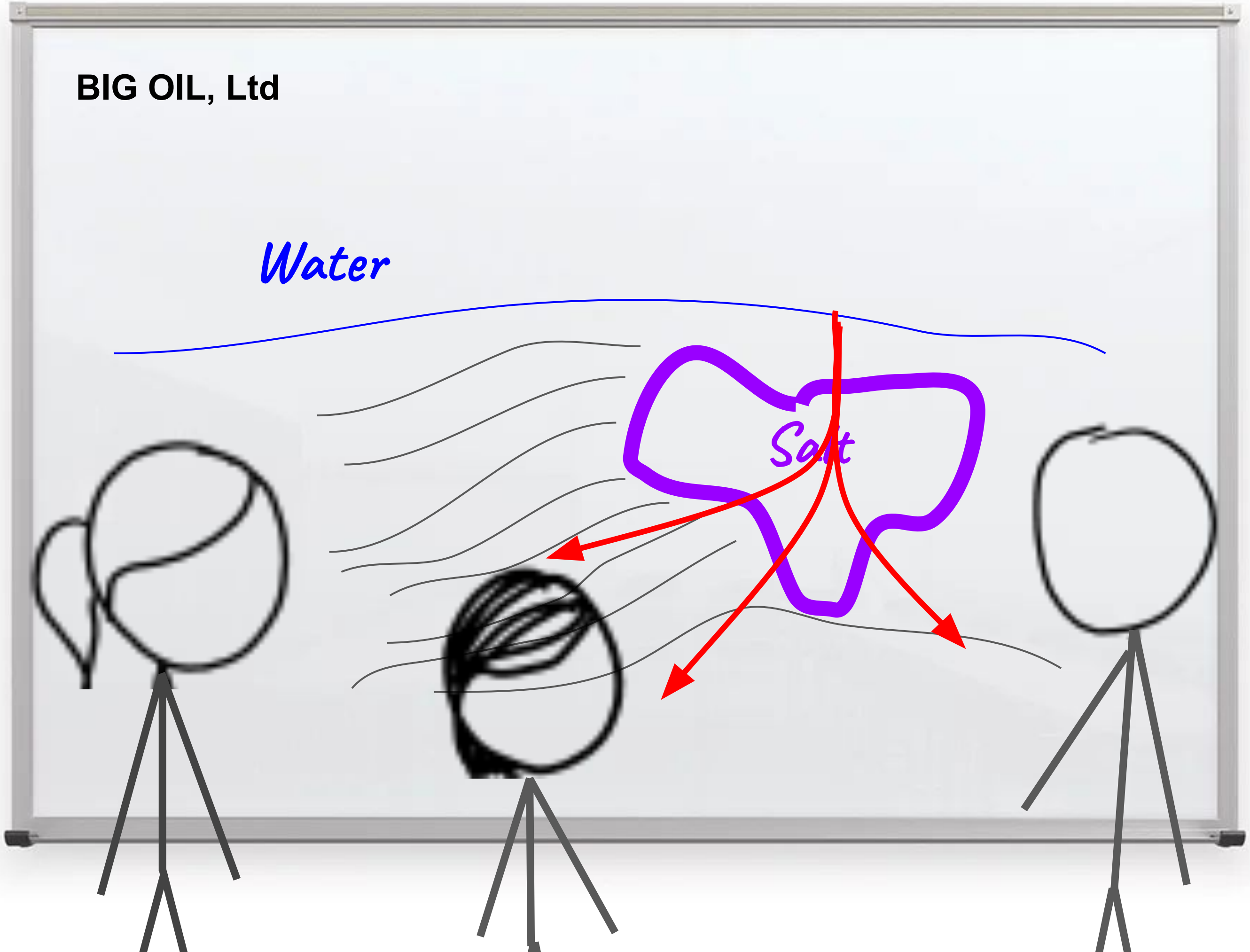


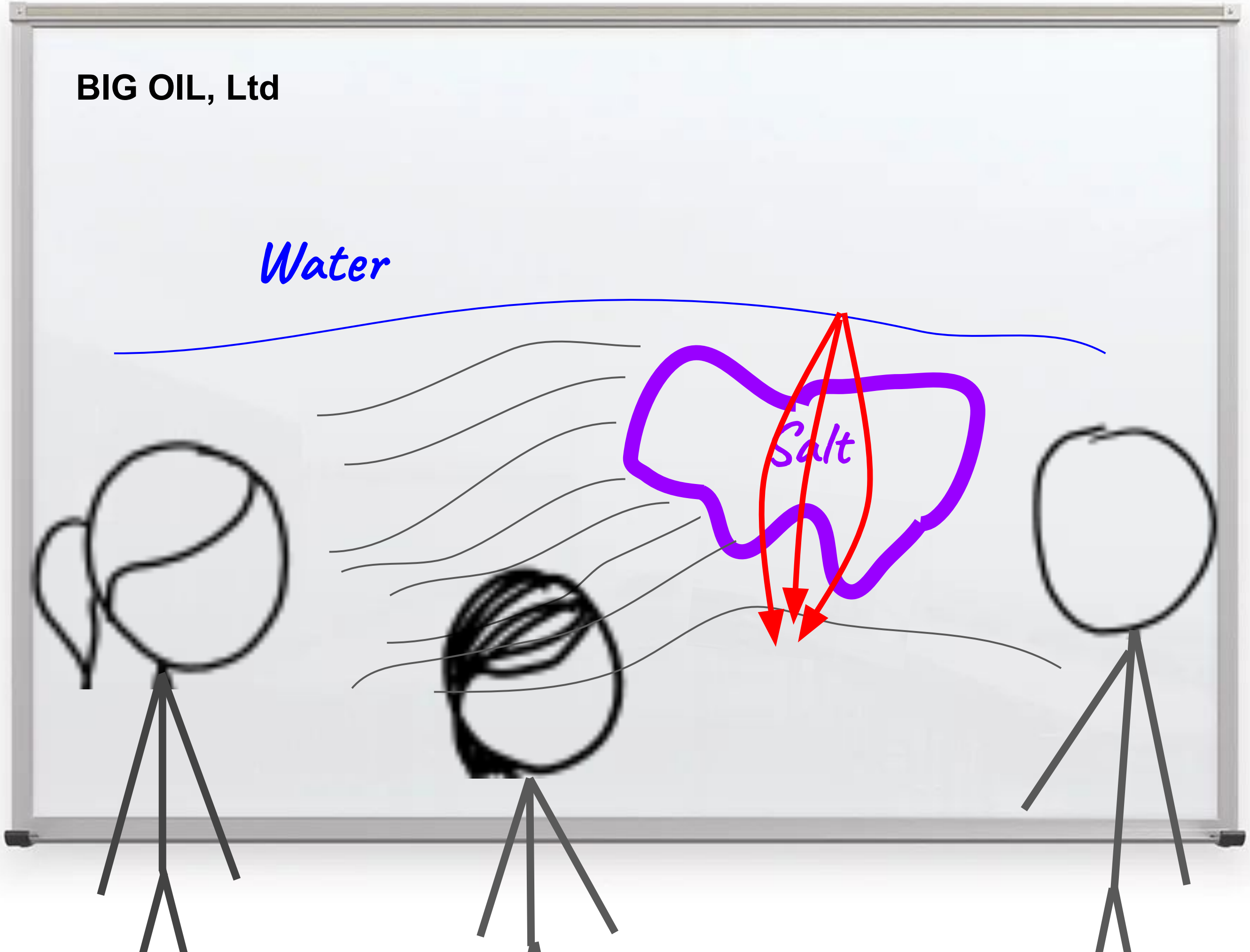
SALT COMPONENT OF OBJECTIVE FUNCTION



SALT COMPONENT OF OBJECTIVE FUNCTION

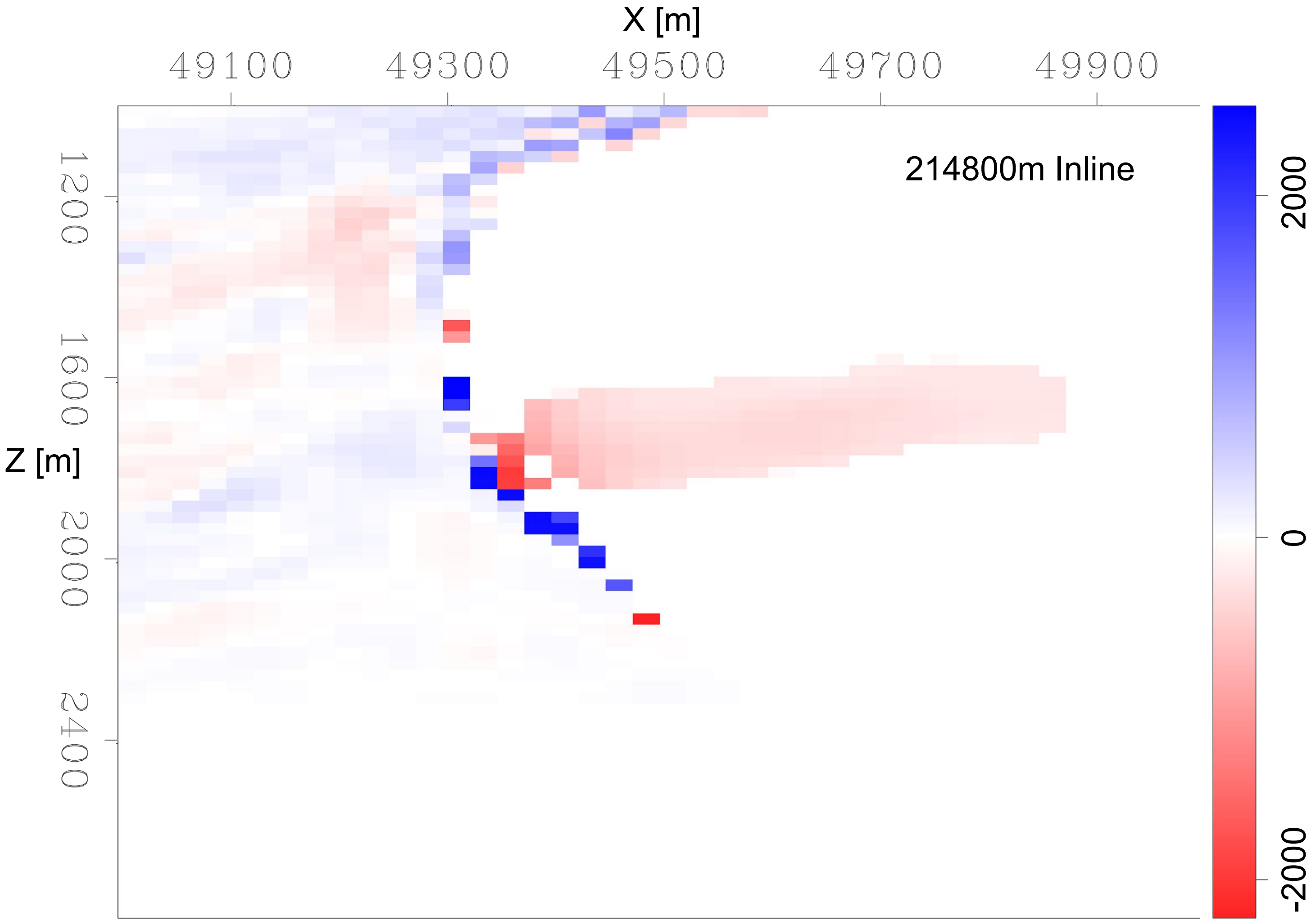



BIG OIL, Ltd
Water
Salt

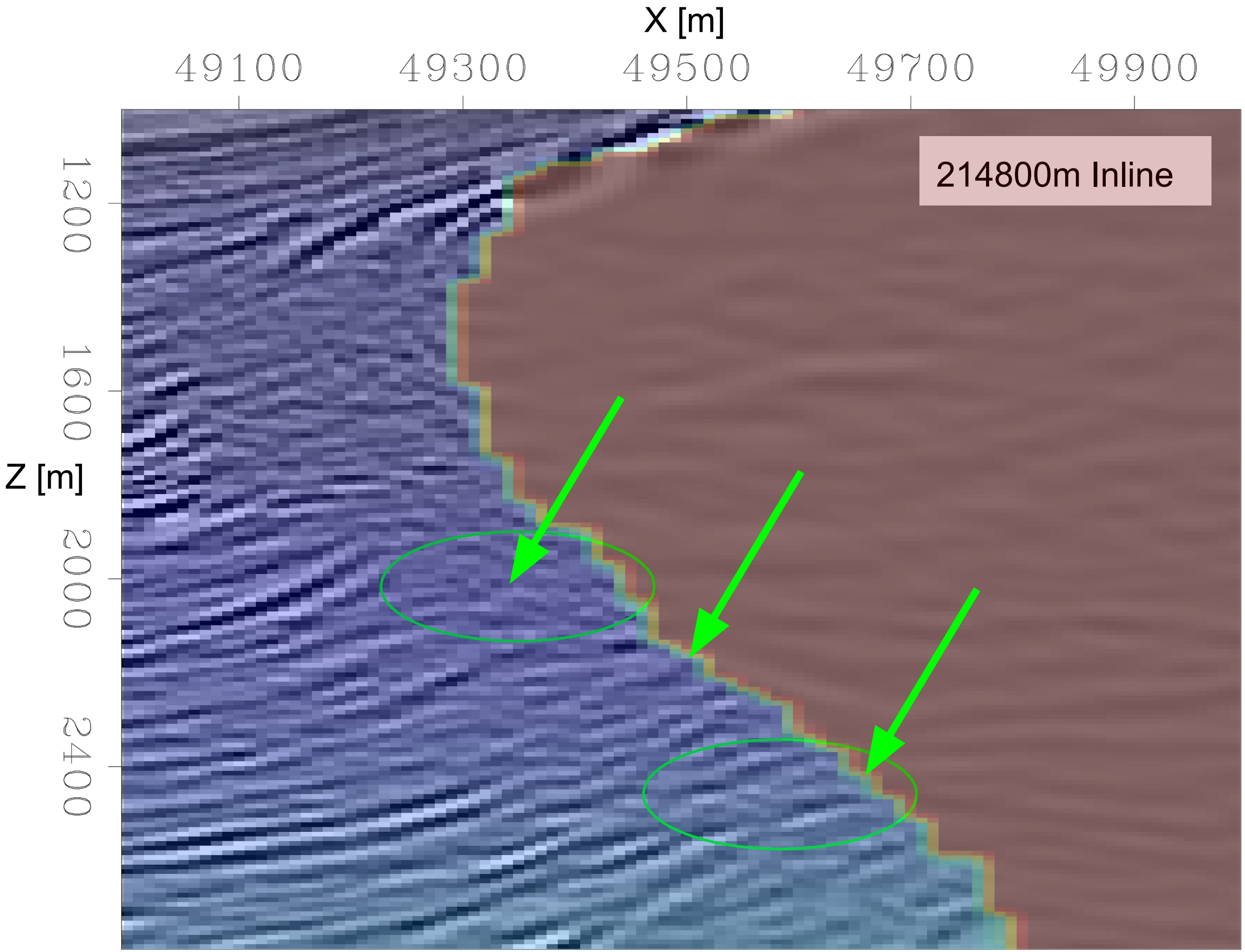

BIG OIL, Ltd
Water
Salt

STEP 3: **Compare new & old RTM images**

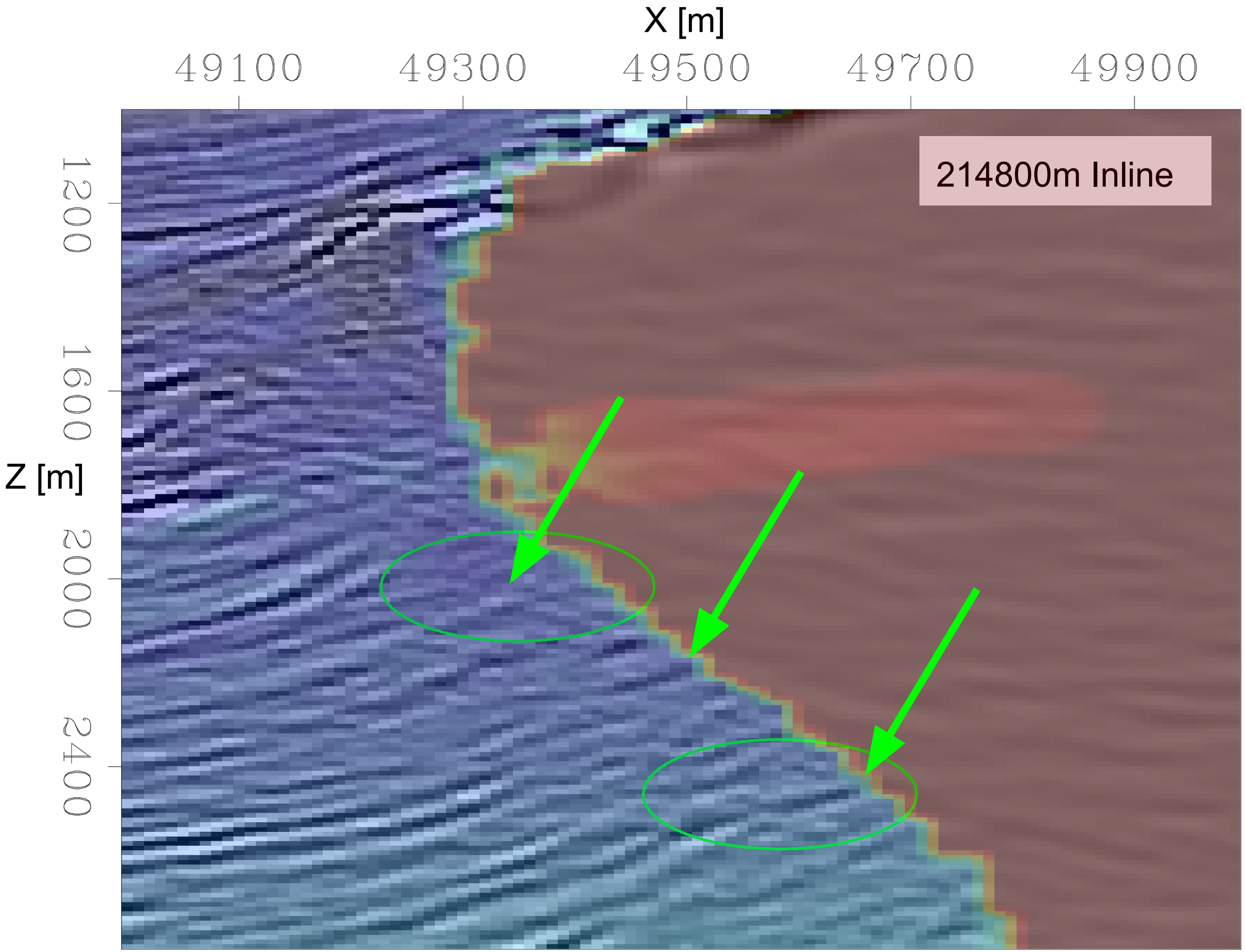
214800m Inline

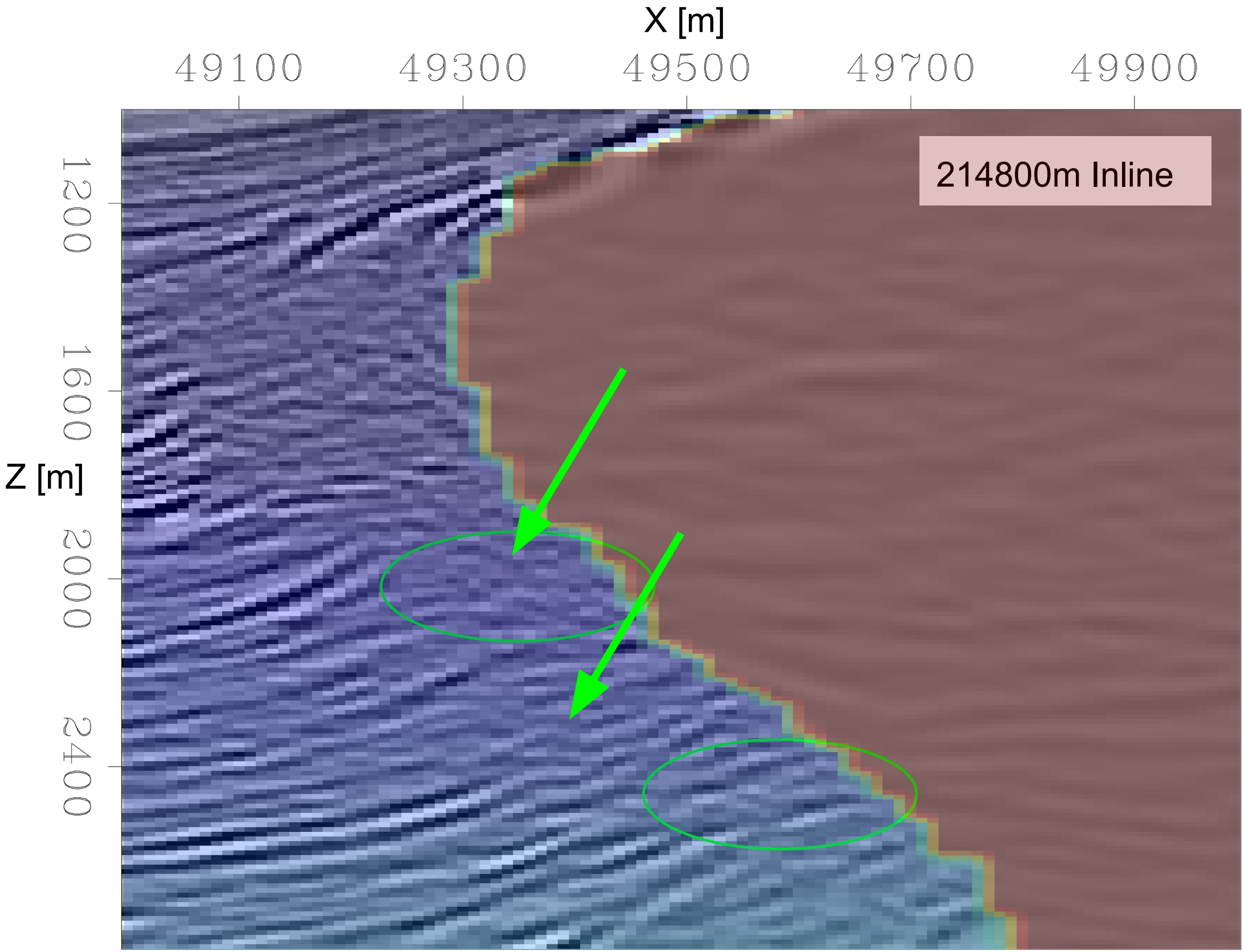


BEFORE UPDATES



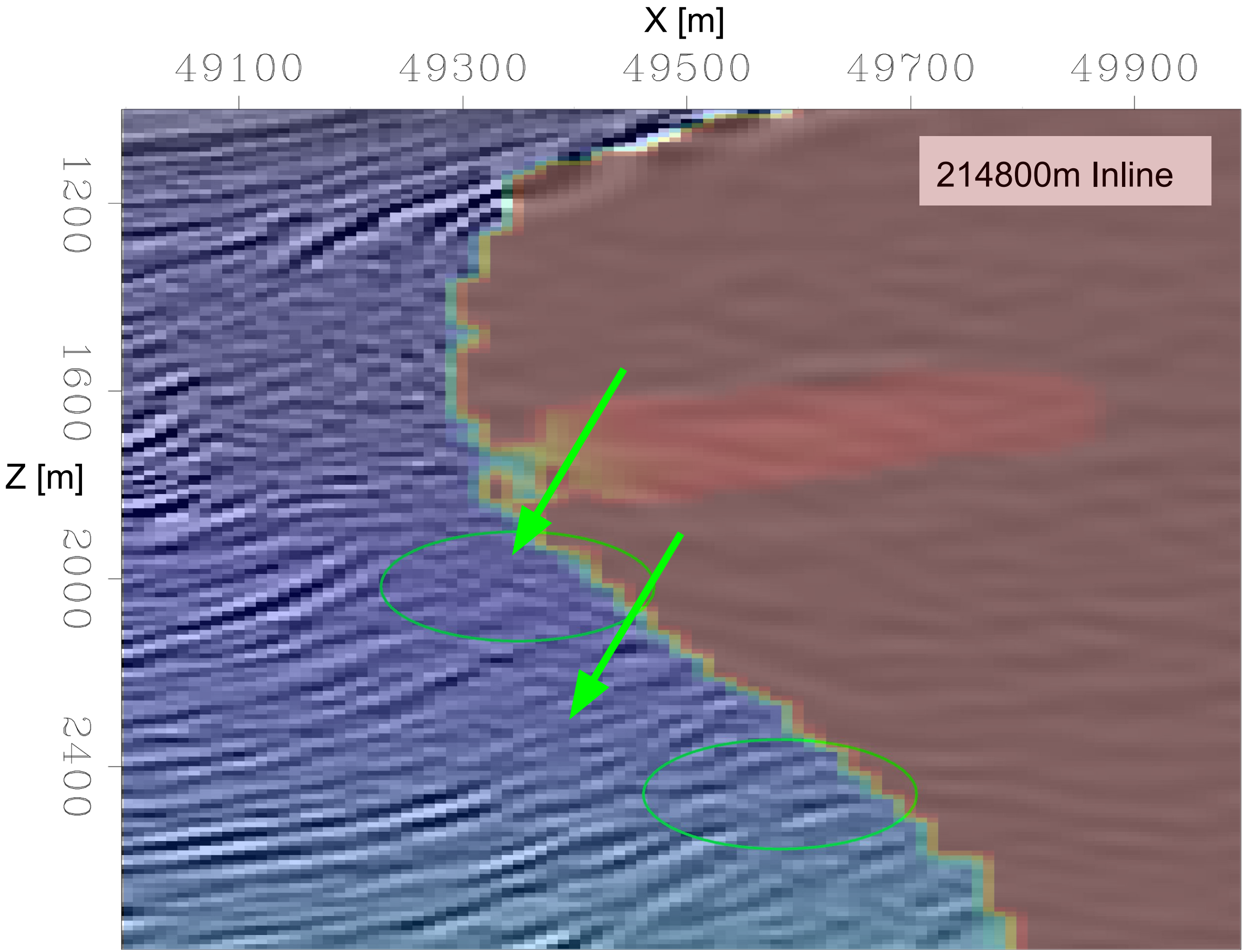
AFTER SALT + BACKGROUND UPDATES

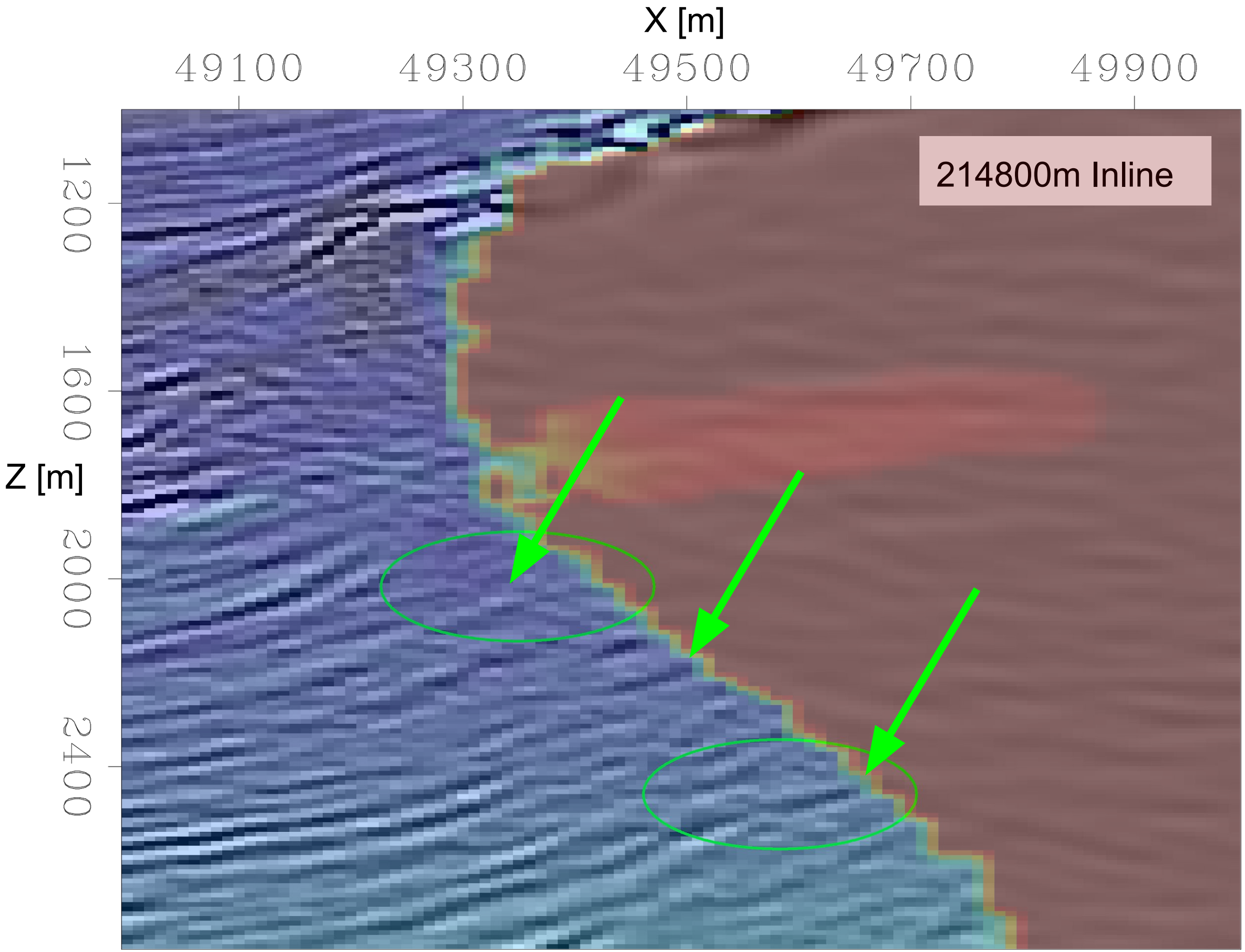




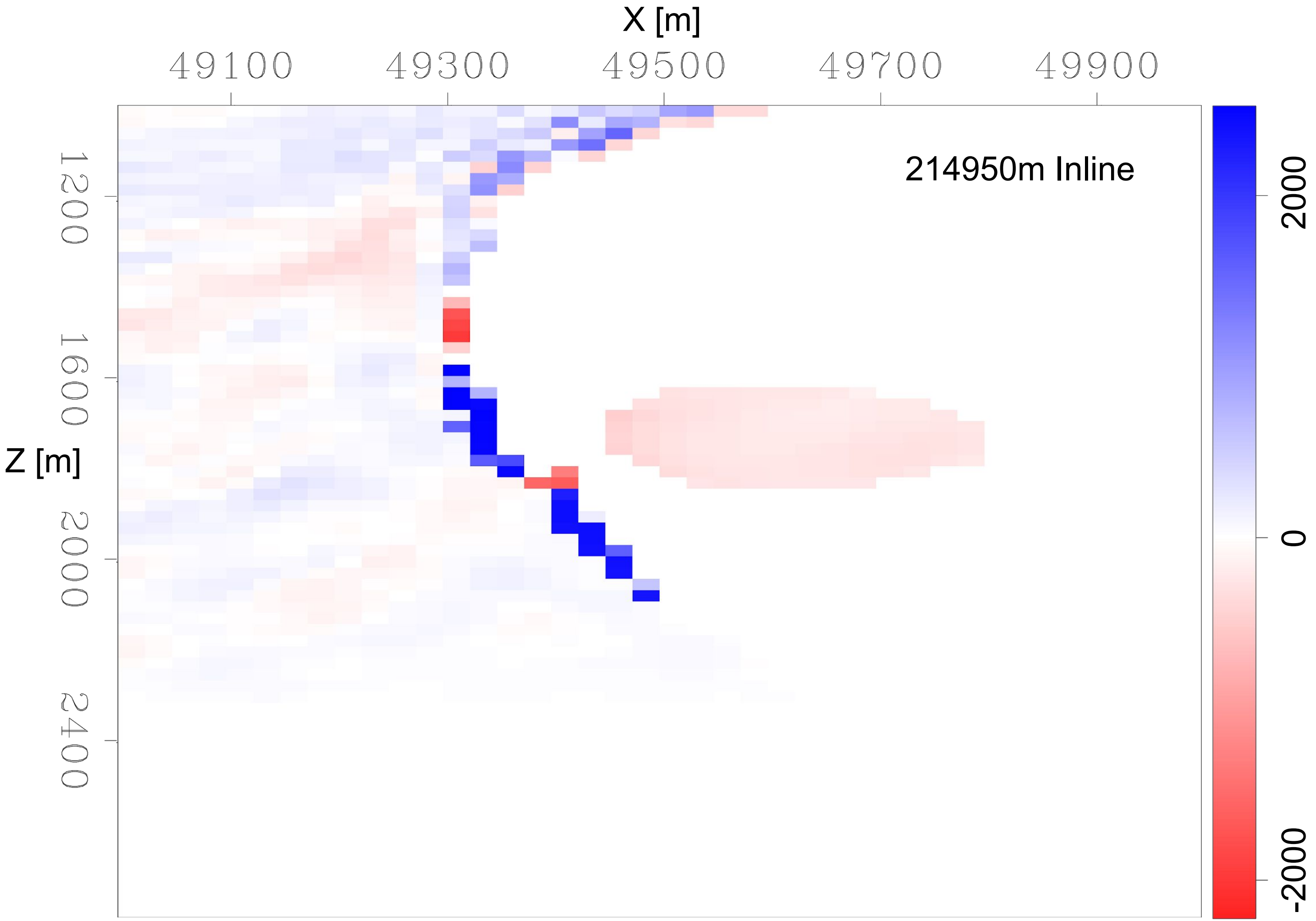
BEFORE UPDATES

AFTER SALT UPDATES

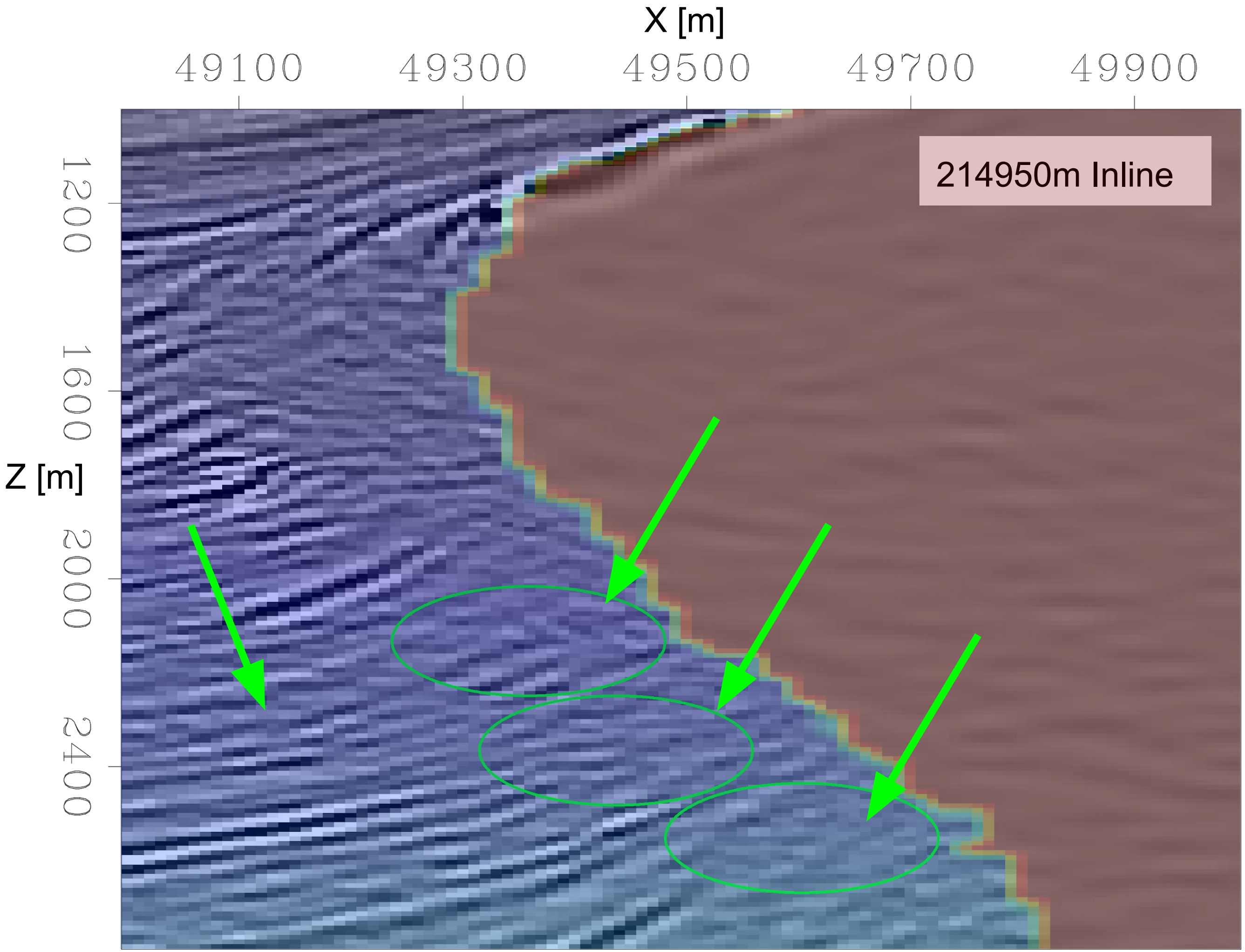




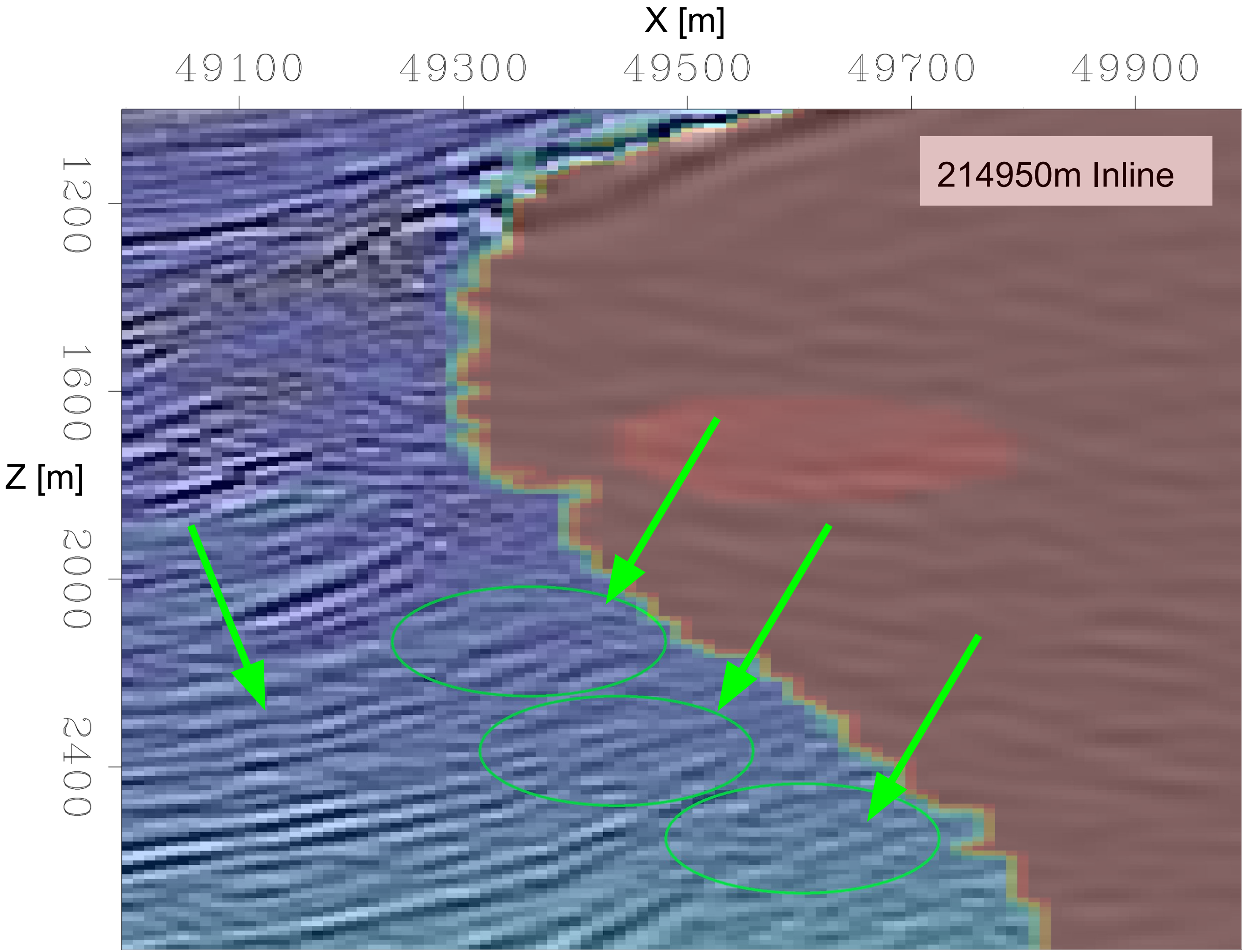
AFTER SALT + BACKGROUND UPDATES


X [m]
49100
49300
49500
49700
49900
Z [m]
1200
1600
2000
2400
214950m Inline
2000
0
-2000

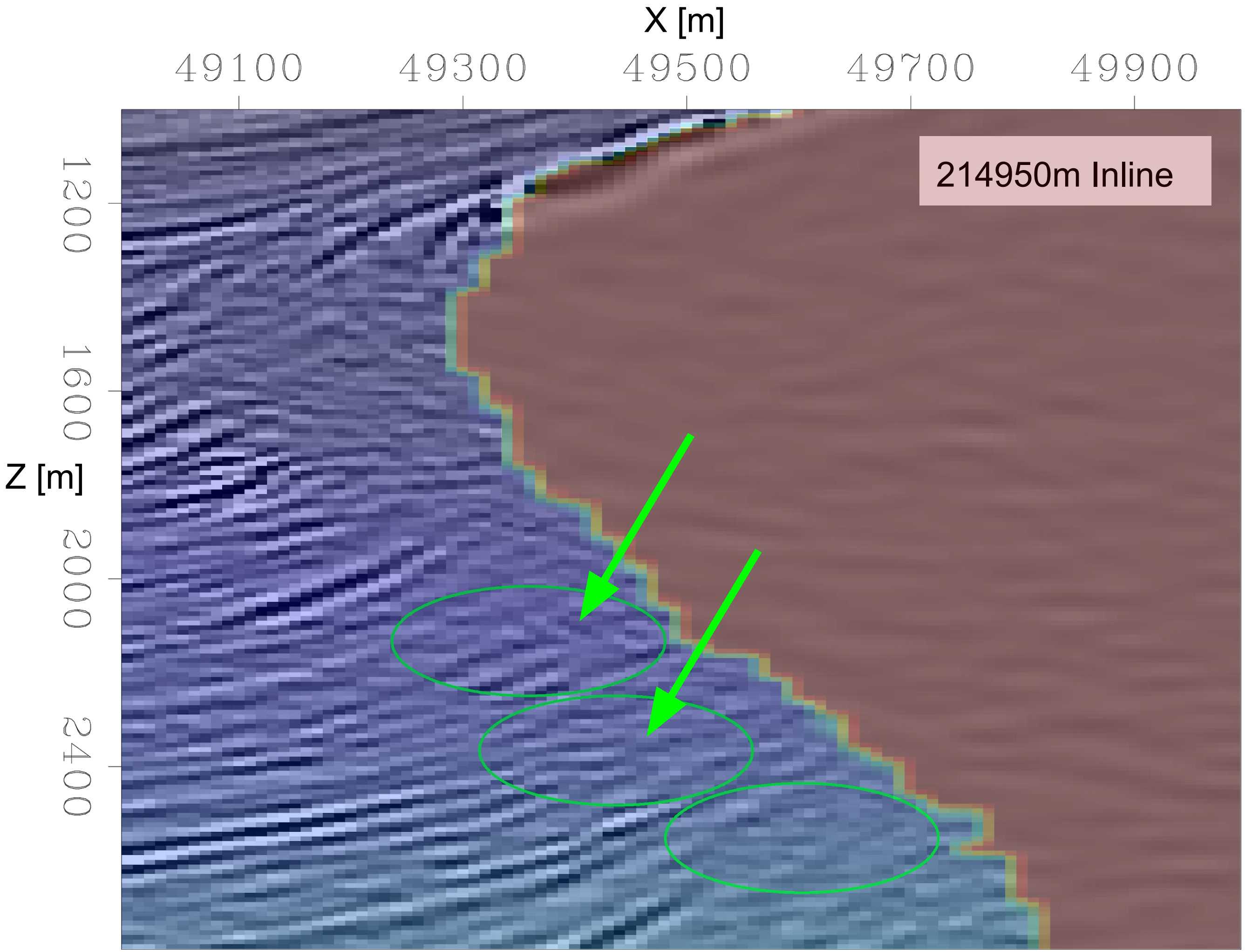
BEFORE UPDATES



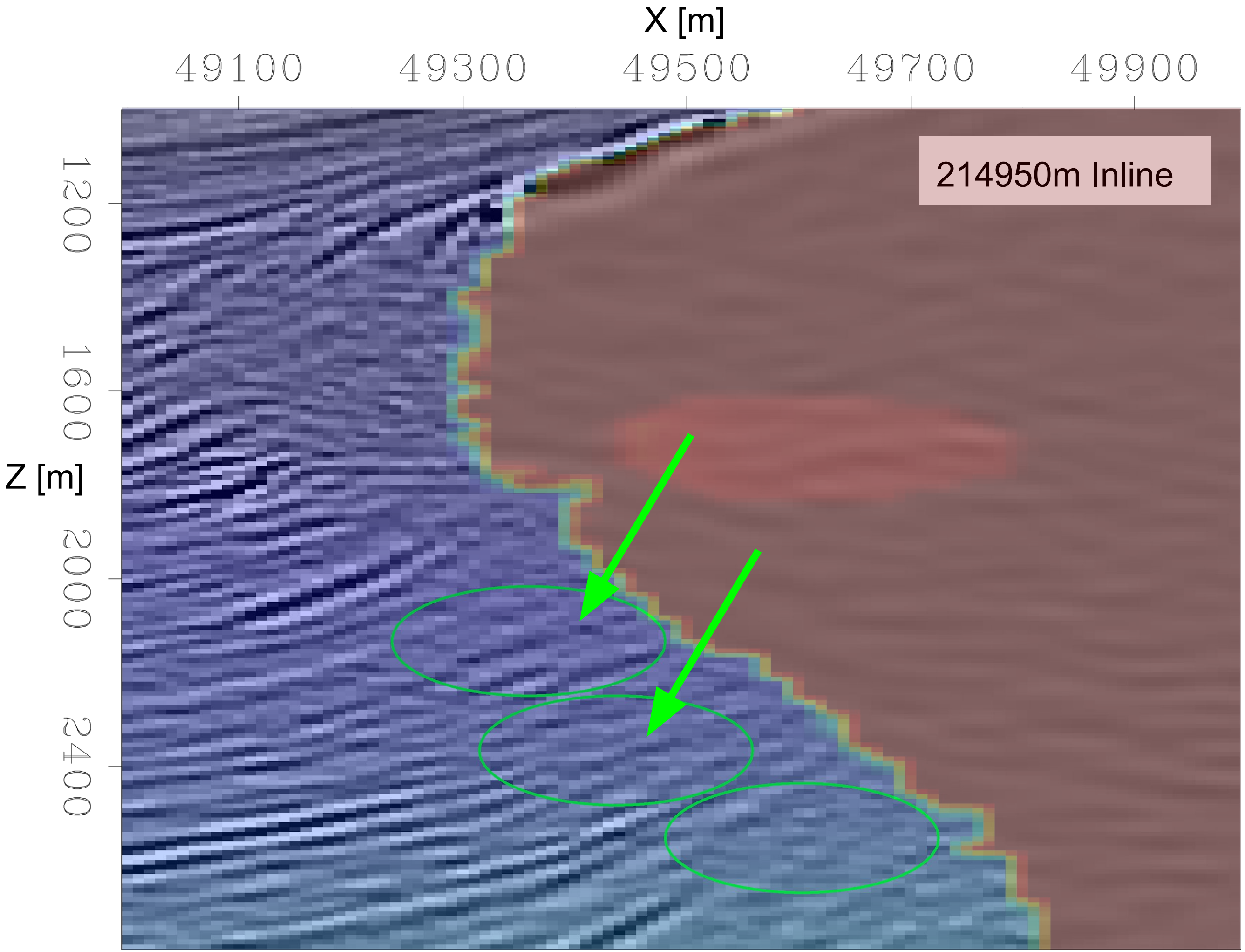
AFTER SALT + BACKGROUND UPDATES



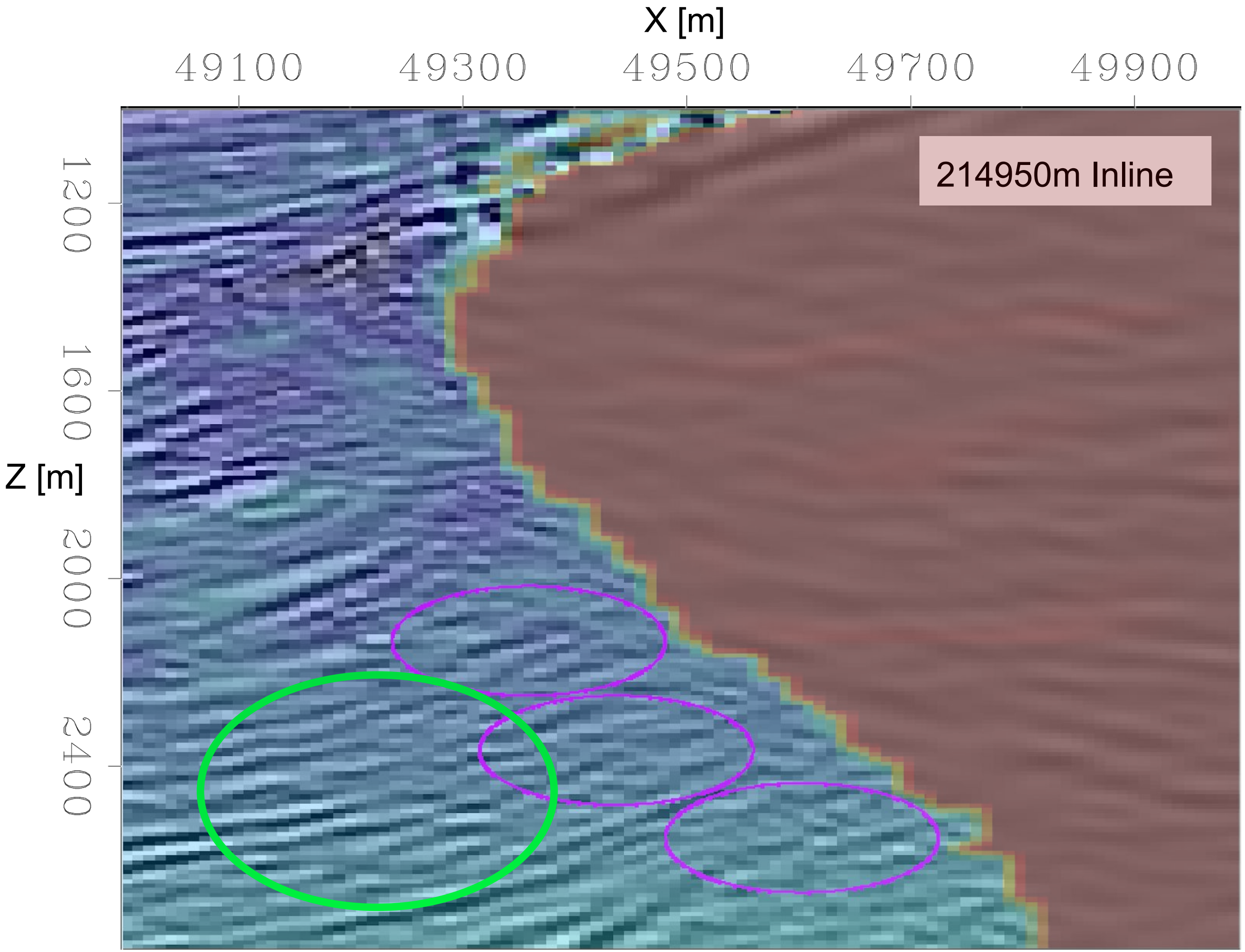
BEFORE UPDATES

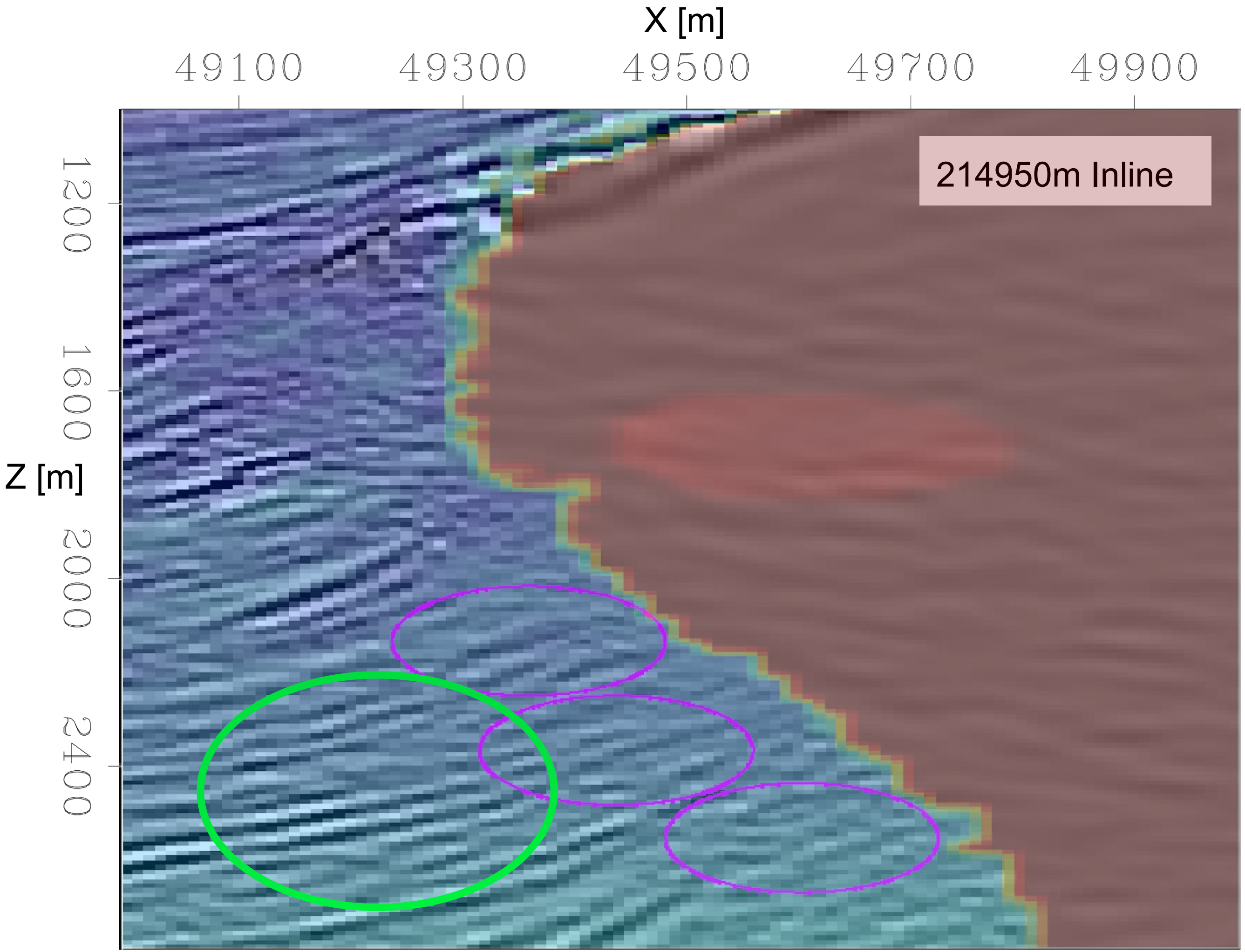


AFTER SALT UPDATES

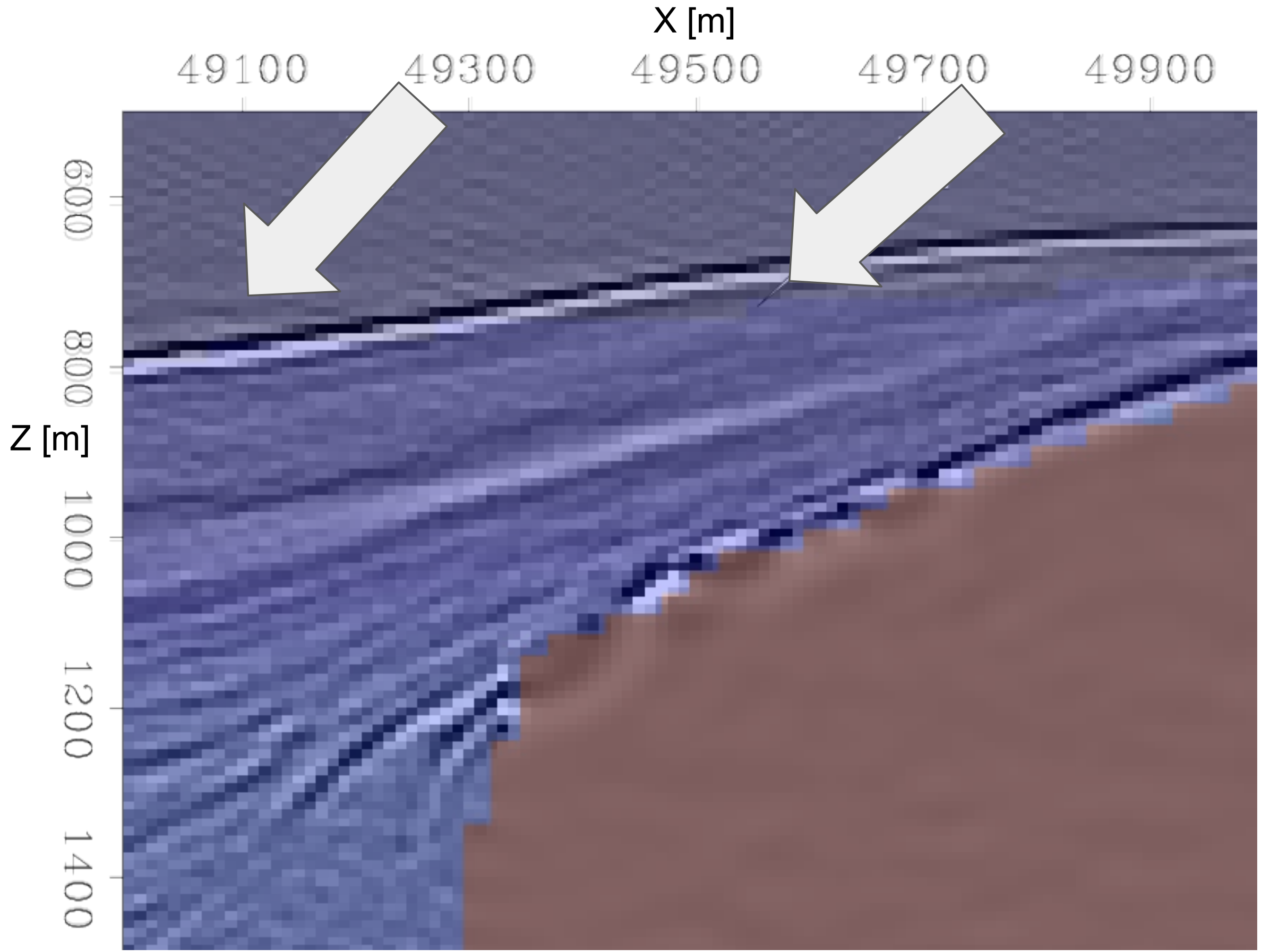


AFTER PLAIN FWI

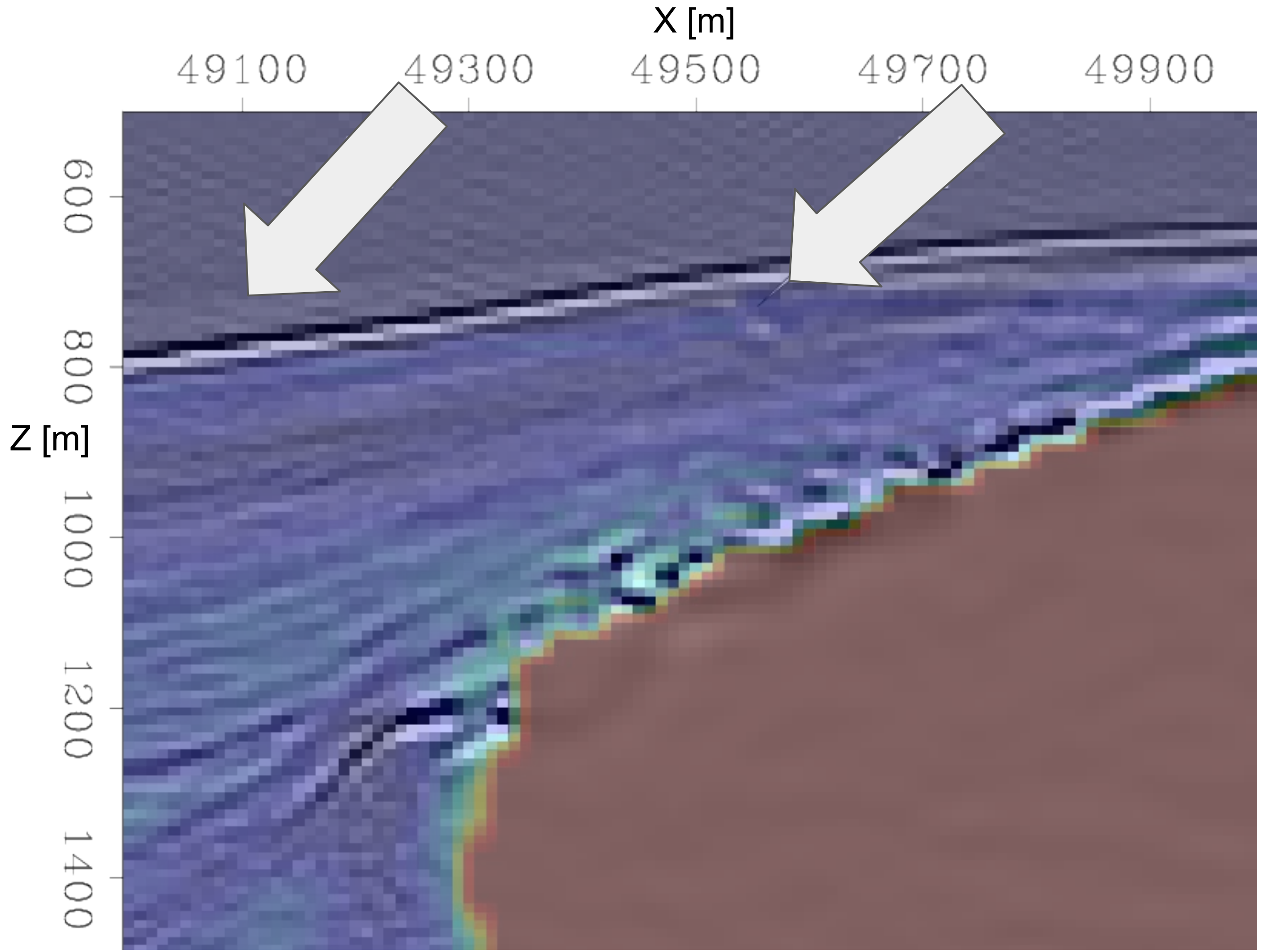




AFTER SALT + BACKGROUND UPDATES

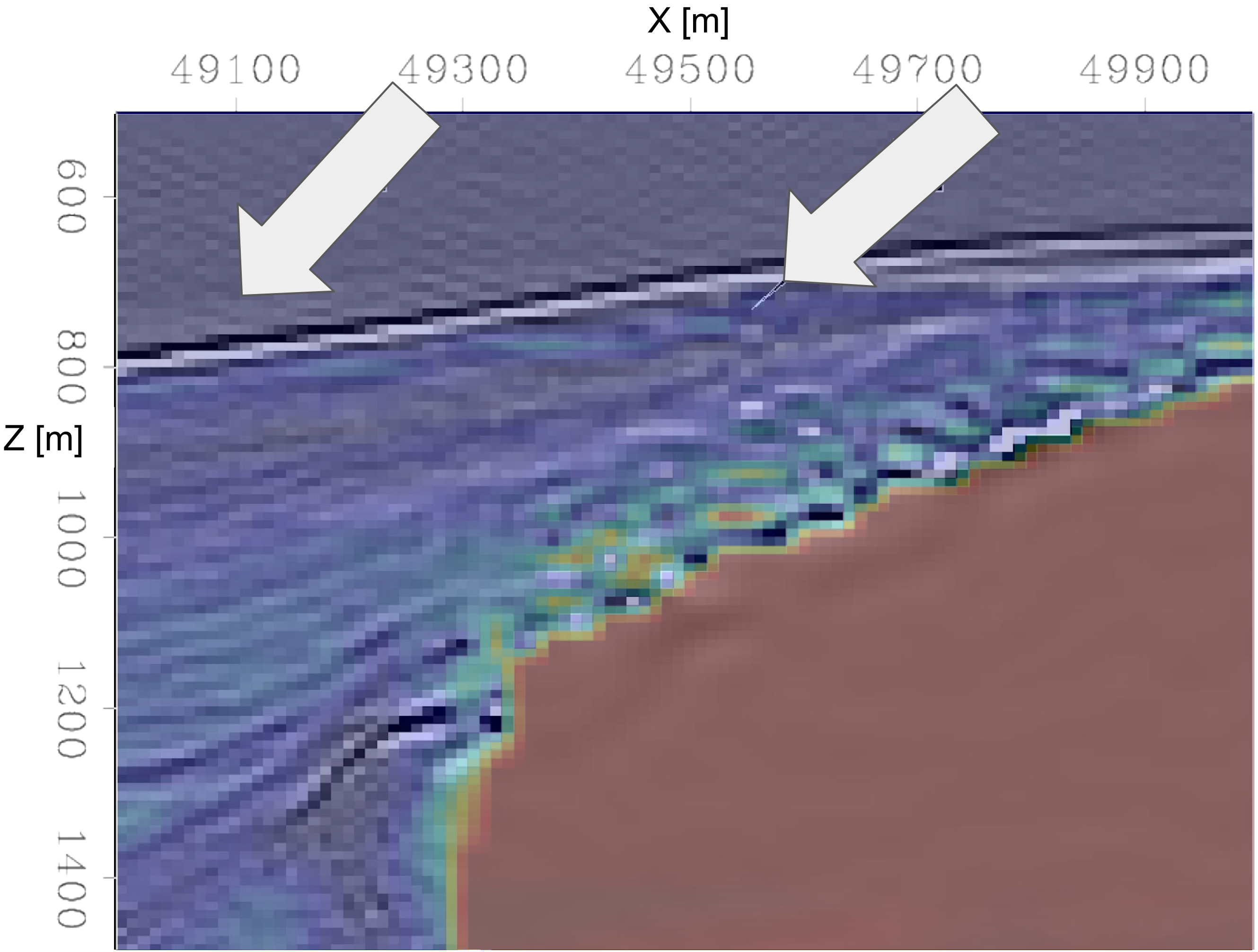


BEFORE UPDATES



AFTER SALT + BACKGROUND UPDATES

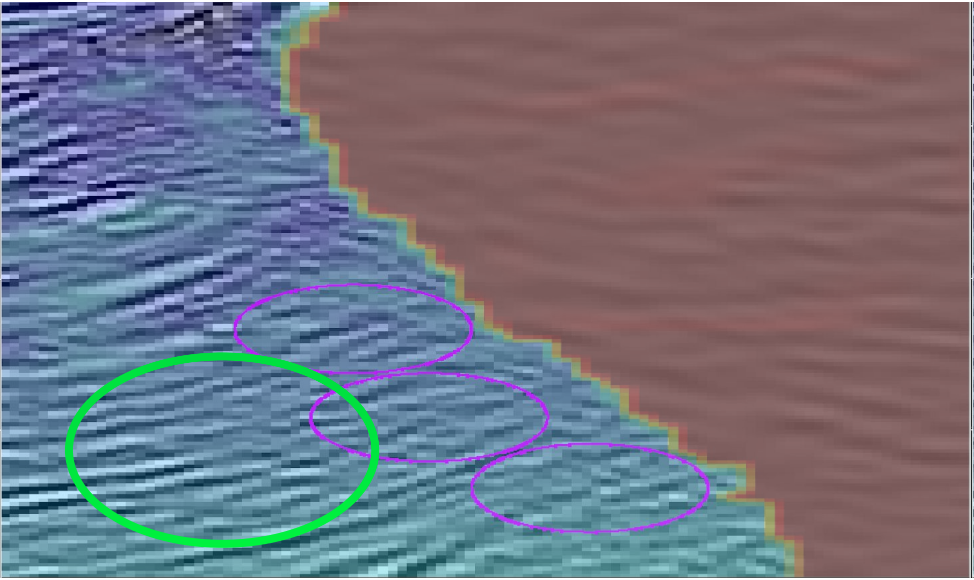
AFTER PLAIN FWI



Conclusions

- Level sets can be used to track sharp salt boundaries in a velocity model inversion context.
- The implicit surface can be sparsely represented, making inversion of the Hessian more computationally feasible.
- The implicit surface offers an elegant means of including expert guidance into the inversion workflow, and can hasten convergence.
- The full method can be used on 3D datasets to find improved salt models, even with inclusions.

STANDARD FWI



SHAPE OPTIMIZATION