A deep learning perspective of the forward and inverse problems in exploration geophysics

Jian Sun*, Zhan Niu, Kris Innanen, Junxiao Li*, Daniel Trad

Nov 30, 2018

Banff, AB

Outline

- Motivation

Outline

- Motivation
- The forward problem: a deep learning perspective
  - Forward modeling of wave propagation
  - Recurrent Neural Network (RNN)

Outline

- Motivation
- The forward problem: a deep learning perspective
  - Forward modeling of wave propagation
  - Recurrent Neural Network (RNN)
- The inverse problem: a deep learning perspective
  - The gradient derivation in a RNN framework
  - Connections with FWI?

Outline

- Motivation
- The forward problem: a deep learning perspective
  - Forward modeling of wave propagation
  - Recurrent Neural Network (RNN)
- The inverse problem: a deep learning perspective
  - The gradient derivation in a RNN framework
  - Connections with FWI?
- Numerical analysis & tuning of hyperparameters
  - Best learning rate selection for gradient-based algorithms
  - Comparisons (GD, Momentum, Adagrad, RMSprop, Adam, CG, L-BFGS)

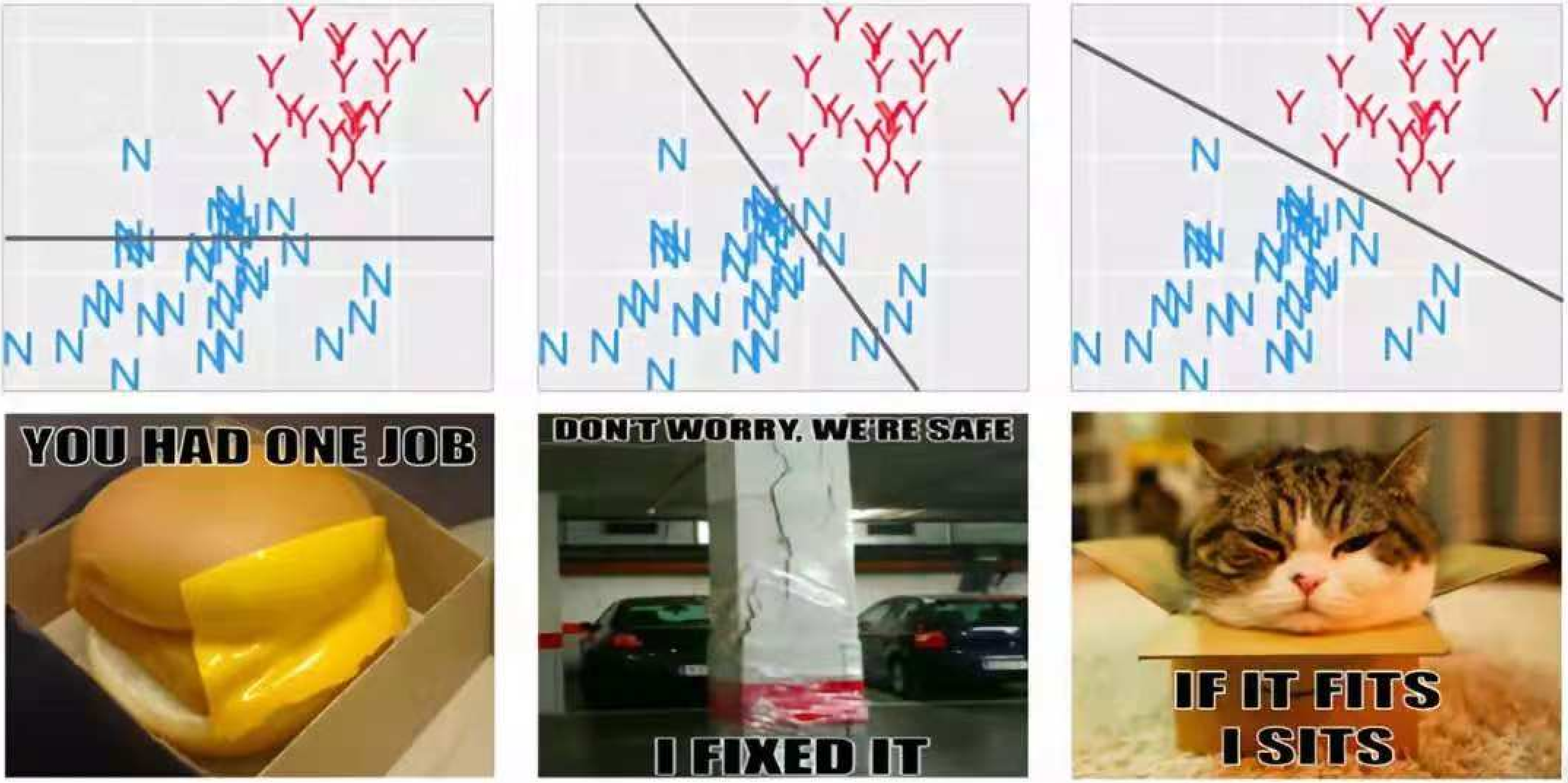
Outline

- Motivation
- The forward problem: a deep learning perspective
  - Forward modeling of wave propagation
  - Recurrent Neural Network (RNN)
- The inverse problem: a deep learning perspective
  - The gradient derivation in a RNN framework
  - Connections with FWI?
- Numerical analysis & tuning of hyperparameters
  - Best learning rate selection for gradient-based algorithms
  - Comparisons (GD, Momentum, Adagrad, RMSprop, Adam, CG, L-BFGS)
- Synthetic test on Marmousi

Outline

- Motivation
- The forward problem: a deep learning perspective
  - Forward modeling of wave propagation
  - Recurrent Neural Network (RNN)
- The inverse problem: a deep learning perspective
  - The gradient derivation in a RNN framework
  - Connections with FWI?
- Numerical analysis & tuning of hyperparameters
  - Best learning rate selection for gradient-based algorithms
  - Comparisons (GD, Momentum, Adagrad, RMSprop, Adam, CG, L-BFGS)
- Synthetic test on Marmousi
- Conclusions & Future works

Motivation



(Picture from internet)

Motivation

- Deep learning (DL):
  - ✓ Widely used: speech recognition, computer vision, auto-driving, machine translation, medical imaging, etc.
  - ✓ Fast: only trained once.
  - ✗ Large and well-labelled training datasets.
  - ✗ Not theory-guided (data-determined).

- Full waveform inversion (FWI):
  - ✓ Theory-guided: small datasets.
  - ✓ Full wavefield information used.
  - ✗ Computationally expensive.
  - ✗ Cycle-skipping.

The forward problem: a deep learning perspective

- Forward modeling of wave propagation

$$\nabla^2 \mathbf{u}(\mathbf{r},t) = \frac{1}{v^2(\mathbf{r})}\frac{\partial^2 \mathbf{u}(\mathbf{r},t)}{\partial t^2} + \boldsymbol{s}(\mathbf{r},t)\delta(\mathbf{r}-\mathbf{r_s})$$

- The second-order finite-difference:

$$\mathbf{u}(\mathbf{r},t+\Delta t) = v^2(\mathbf{r})\Delta t^2\left[\nabla^2\mathbf{u}(\mathbf{r},t) - \boldsymbol{s}(\mathbf{r},t)\delta(\mathbf{r}-\mathbf{r_s})\right] + 2\mathbf{u}(\mathbf{r},t) - \mathbf{u}(\mathbf{r},t-\Delta t)$$

The forward problem: a deep learning perspective

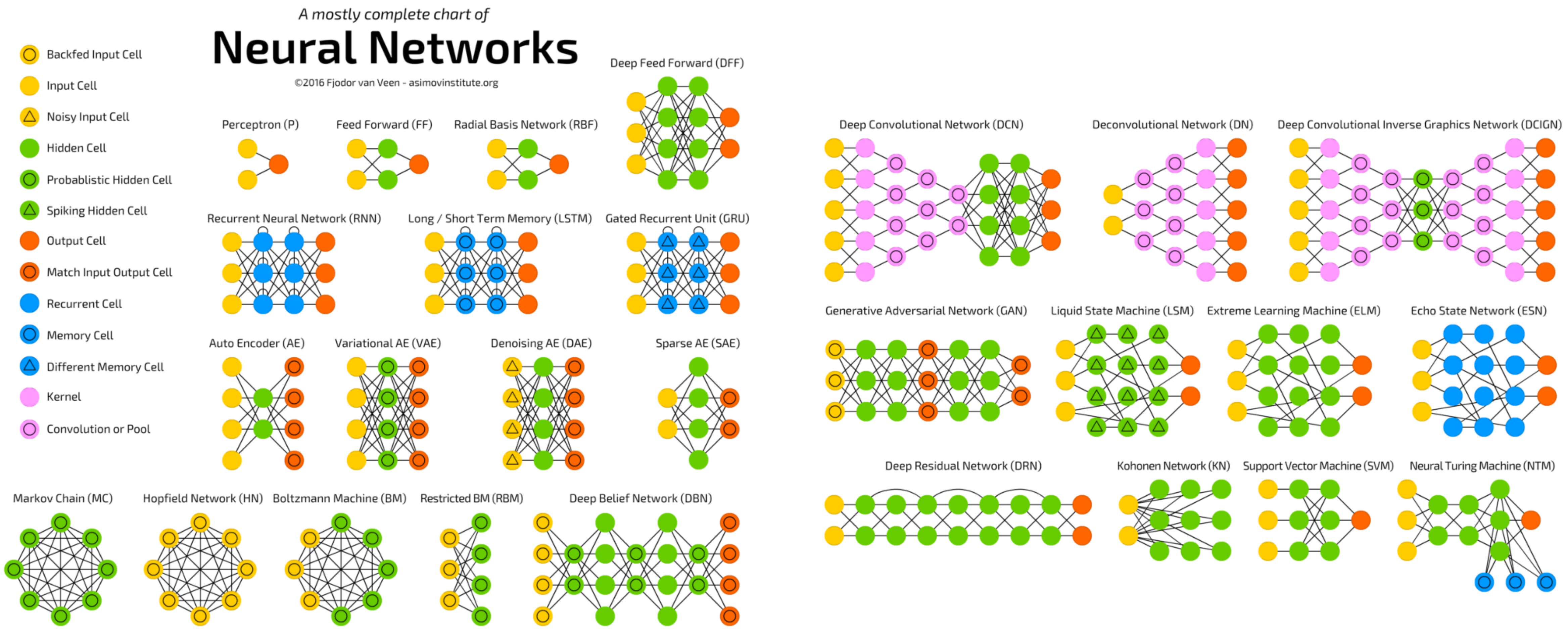
- Forward modeling of wave propagation

$$\nabla^2 \mathbf{u}(\mathbf{r}, t) = \frac{1}{v^2(\mathbf{r})} \frac{\partial^2 \mathbf{u}(\mathbf{r}, t)}{\partial t^2} + \boldsymbol{s}(\mathbf{r}, t)\delta(\mathbf{r} - \mathbf{r_s})$$

- The second-order finite-difference:

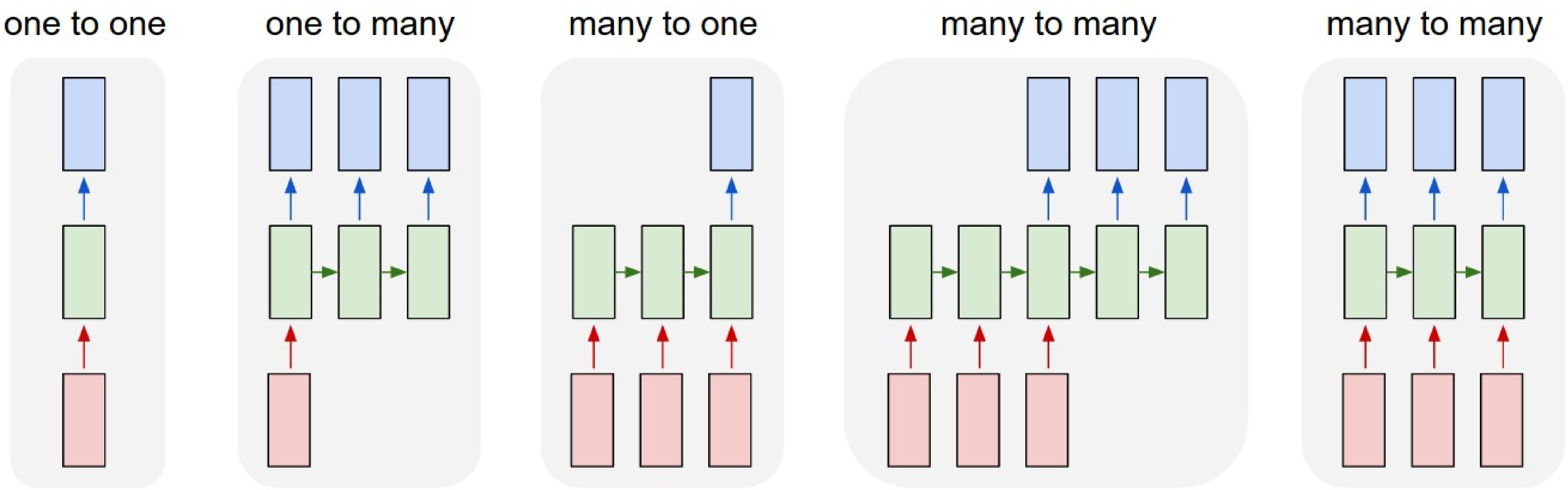
$$\mathbf{u}(\mathbf{r}, t + \Delta t) = v^2(\mathbf{r})\Delta t^2 \left[\nabla^2 \mathbf{u}(\mathbf{r}, t) - \boldsymbol{s}(\mathbf{r}, t)\delta(\mathbf{r} - \mathbf{r_s})\right] + 2\mathbf{u}(\mathbf{r}, t) - \mathbf{u}(\mathbf{r}, t - \Delta t)$$

Neural Networks (NN)

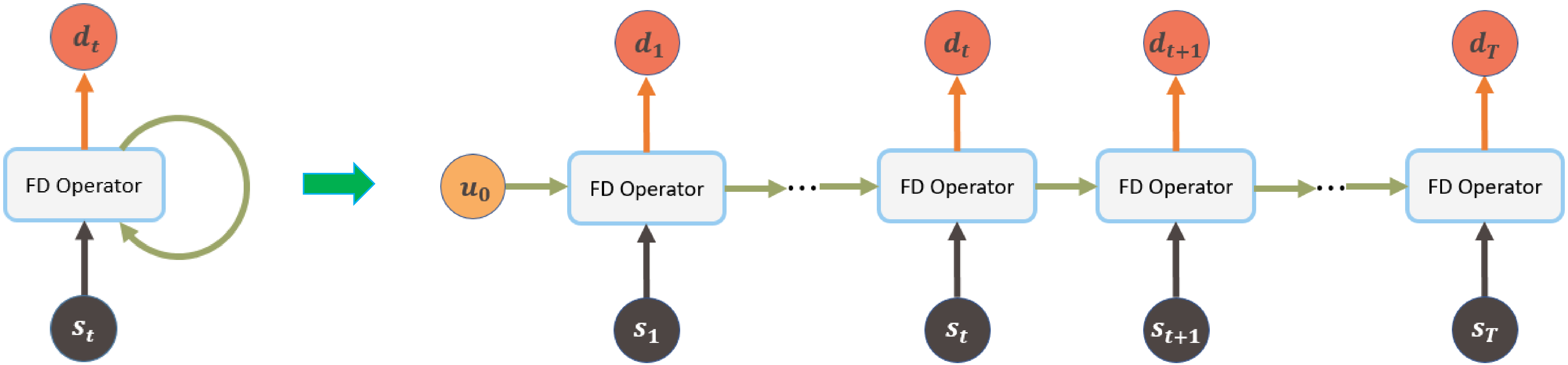


(Fjodor van Veen, 2017)

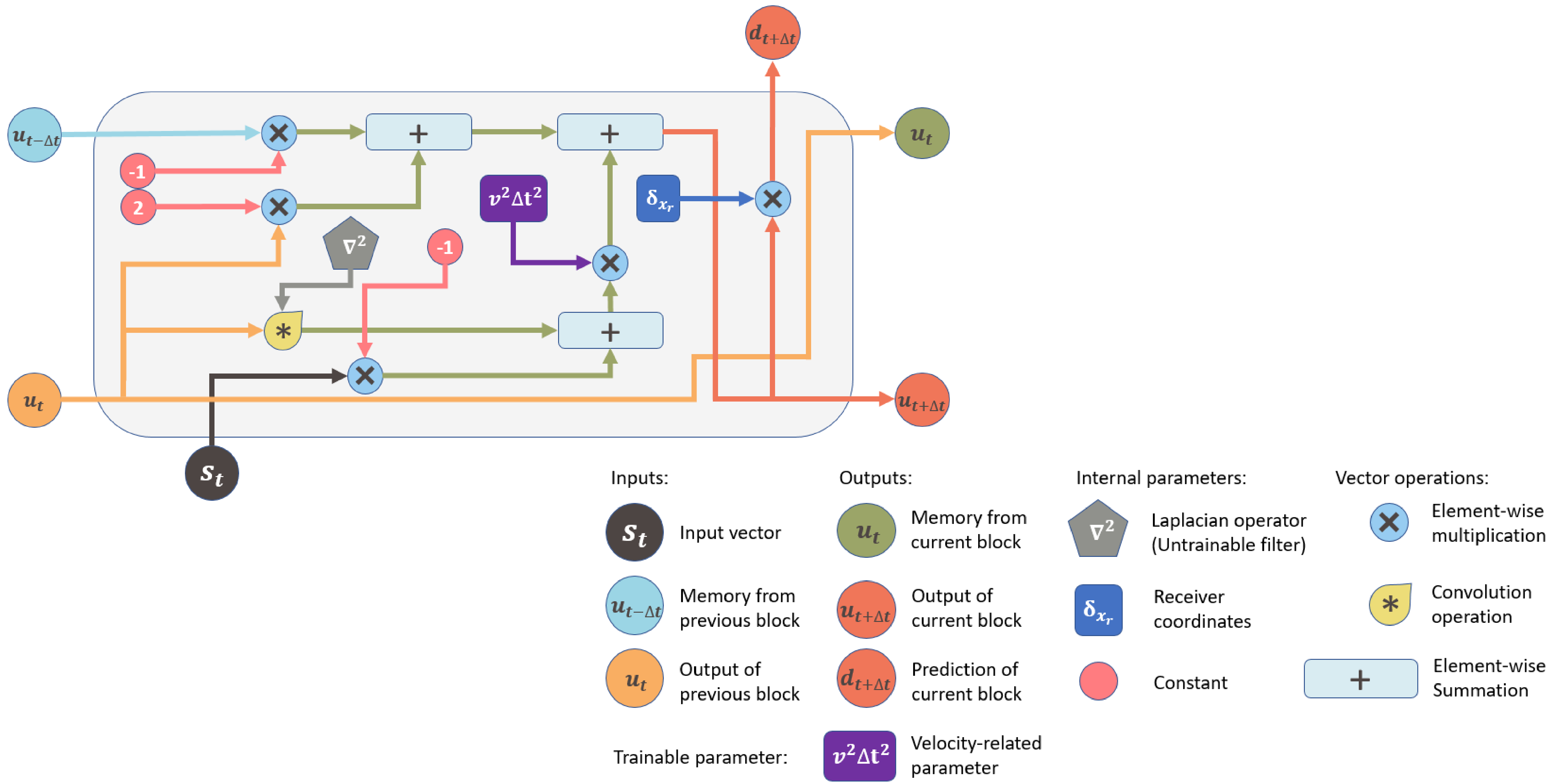
Recurrent Neural Network (RNN)



Recurrent Neural Network (RNN)

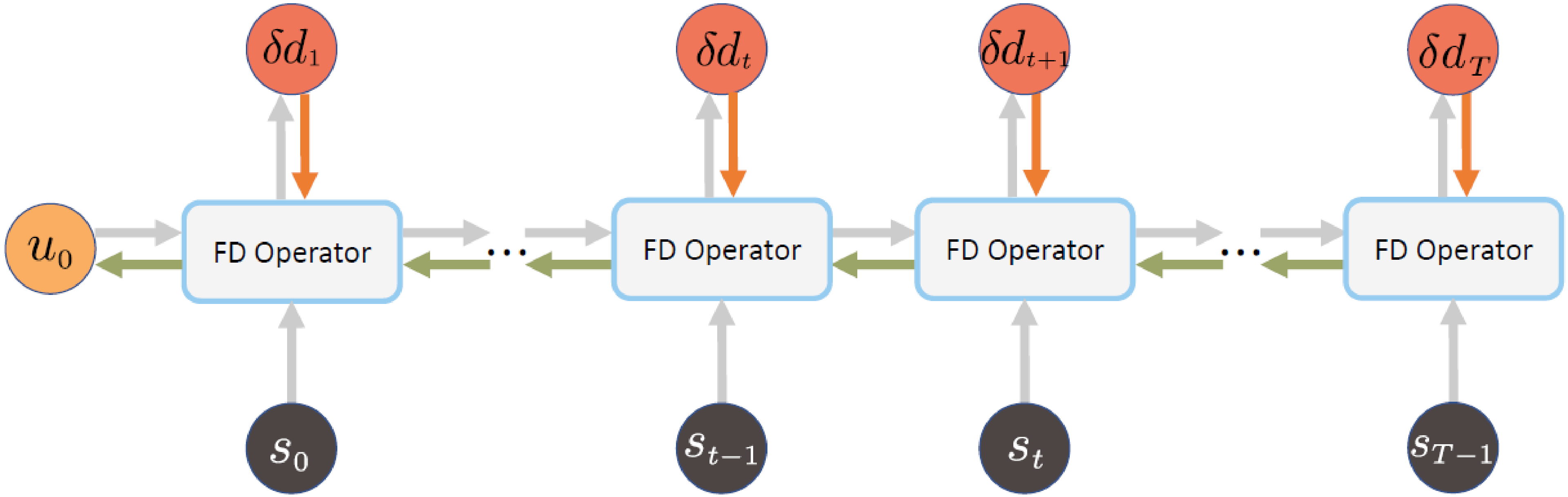


Recurrent Neural Network (RNN)



The inverse problem: a deep learning perspective

- The gradient derivation in a RNN framework



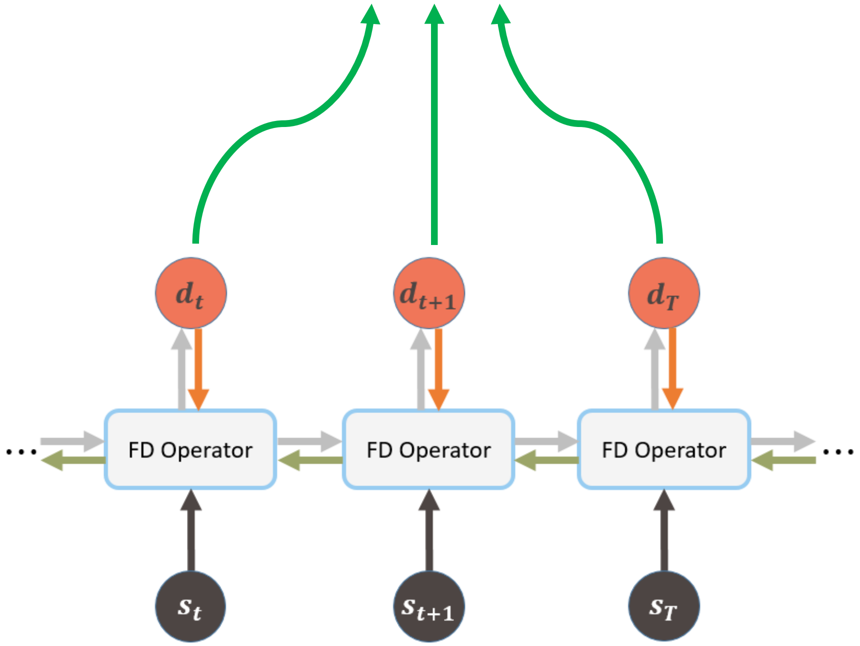
➢ Objective function: $J(v) = \frac{1}{2n_s}\sum_{\mathbf{r_s}}\sum_{\mathbf{r_g}}\sum_{\mathbf{t}}(\mathbf{d_t} - \tilde{\mathbf{d}}_\mathbf{t})^2 = \frac{1}{2n_s}\sum_{\mathbf{r_s}}\sum_{\mathbf{r_g}}\sum_{\mathbf{t}}(\mathbf{d_t} - \boldsymbol{\delta}_{\mathbf{r_g}}\tilde{\mathbf{u}}_\mathbf{t})^2$

➢ Gradient: $\frac{\partial J}{\partial \boldsymbol{v}} = \sum_{\mathbf{t}}^{\mathbf{T}}\left[\frac{\partial J}{\partial \tilde{\boldsymbol{u}}_t}\right]\frac{\partial \tilde{\boldsymbol{u}}_t}{\partial \boldsymbol{v}}$

The inverse problem: a deep learning perspective

- The gradient derivation in a RNN framework

$$J(\boldsymbol{v}) = \frac{1}{2n_s}\sum_{\mathbf{r_s}}\sum_{\mathbf{r_g}}\sum_{\mathbf{t}}(\mathbf{d_t} - \boldsymbol{\delta}_{\mathbf{r_g}}\tilde{\mathbf{u}}_{\mathbf{t}})^2$$



➢ Gradient at time-step t: $\left[\frac{\partial J}{\partial \boldsymbol{v}}\right]_t = \left[\frac{\partial J}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t}}}\right]\frac{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t}}}{\partial \boldsymbol{v}}$

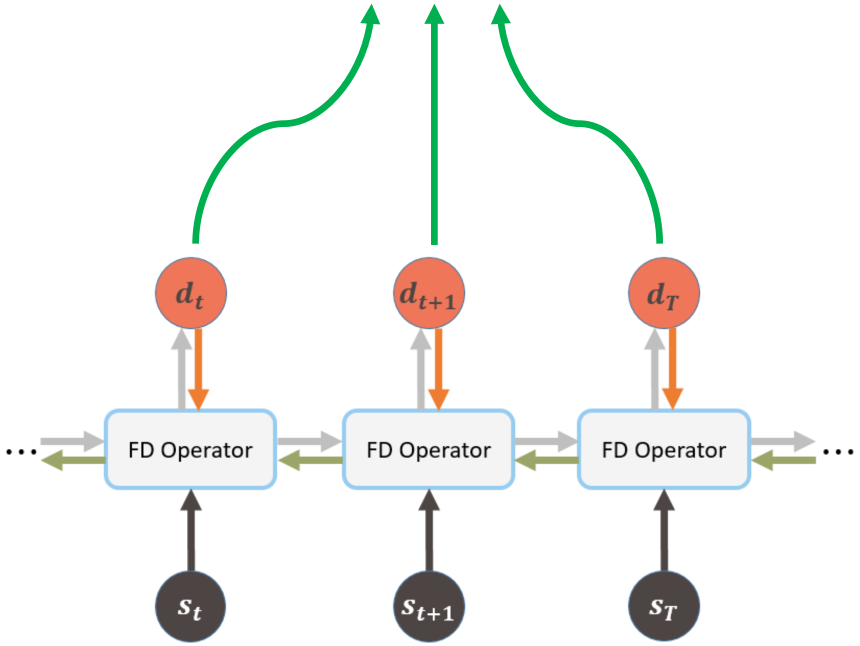
$$\begin{aligned}\left[\frac{\partial J}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t}}}\right] &= \left[\frac{\partial J}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t+2}}}\right]\frac{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t+2}}}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t}}} + \left[\frac{\partial J}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t+1}}}\right]\frac{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t+1}}}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t}}} + \frac{\partial J}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t}}}\\ &= \boldsymbol{v}^2\Delta t^2\left(\nabla^2\left[\frac{\partial J}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t+1}}}\right] - \frac{1}{n_s\boldsymbol{v}^2\Delta t^2}\sum_{\mathbf{r_s}}\sum_{\mathbf{r_g}}\boldsymbol{\delta}\mathbf{d_t}\right)\\ &\quad + 2\left[\frac{\partial J}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t+1}}}\right] - \left[\frac{\partial J}{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t+2}}}\right]\end{aligned}$$

$$\frac{\partial \tilde{\boldsymbol{u}}_{\boldsymbol{t}}}{\partial \boldsymbol{v}} = \frac{2\Delta t^2}{\boldsymbol{v}}\boldsymbol{v}^2(\nabla^2\tilde{\mathbf{u}}_{\mathbf{t-1}} - \boldsymbol{s}_{\boldsymbol{t-1}}) \approx \frac{2\Delta t^2}{\boldsymbol{v}}\frac{\partial^2\tilde{\boldsymbol{u}}_{\boldsymbol{t}}}{\partial t^2}$$

The inverse problem: a deep learning perspective

- Connections with FWI?

$$J(\boldsymbol{v}) = \frac{1}{2n_s}\sum_{\mathbf{r_s}}\sum_{\mathbf{r_g}}\sum_{\mathbf{t}}(\mathbf{d_t} - \boldsymbol{\delta}_{\mathbf{r_g}}\tilde{\mathbf{u}}_{\mathbf{t}})^2$$



➢ Gradient at time-step t:

$$\boldsymbol{g}_t = \left[\frac{\partial J}{\partial \boldsymbol{v}}\right]_t$$

$$= BP(-\frac{1}{n_s \boldsymbol{v}^2 \Delta t^2}\sum_{\mathbf{r_s}}\sum_{\mathbf{r_g}}\boldsymbol{\delta}\mathbf{d_t})\frac{2\Delta t^2}{\boldsymbol{v}}\frac{\partial^2 \tilde{\boldsymbol{u}}_{t-1}}{\partial t^2}$$ RNN

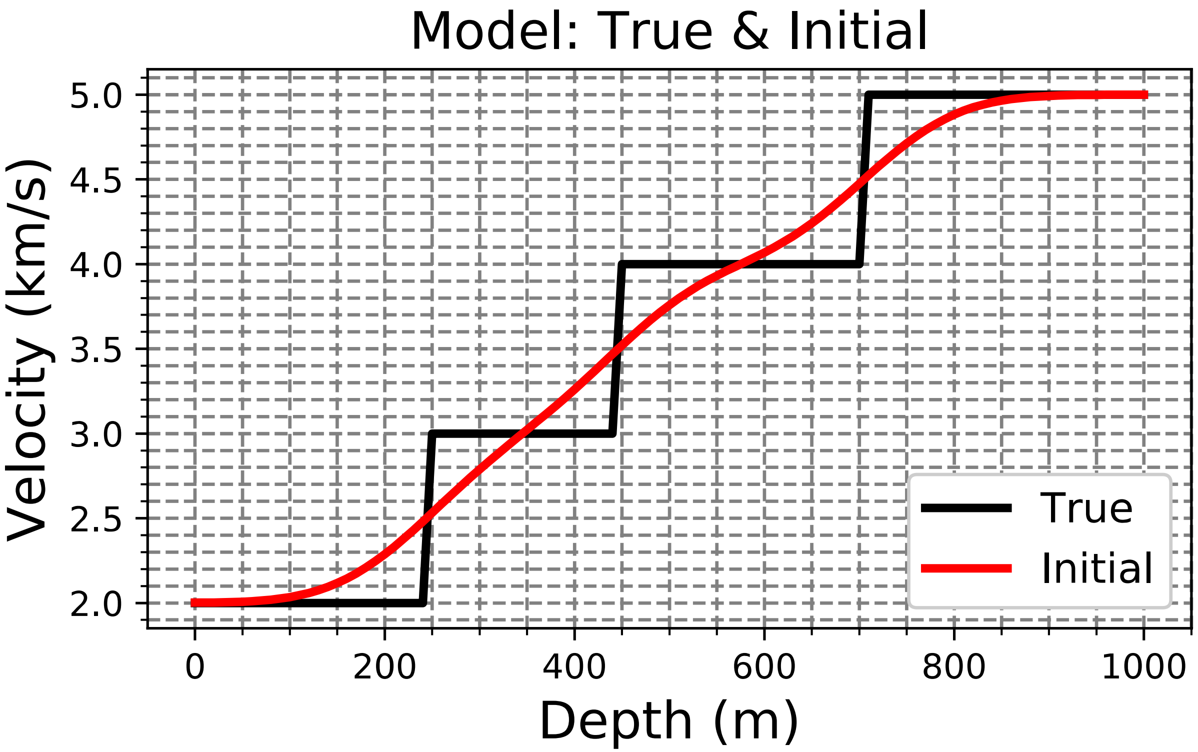
$$\approx BP(-\frac{1}{n_s}\sum_{\mathbf{r_s}}\sum_{\mathbf{r_g}}\boldsymbol{\delta}\mathbf{d_t})\frac{2}{\boldsymbol{v}^3}\frac{\partial^2 \tilde{\boldsymbol{u}}_t}{\partial t^2}$$ FWI

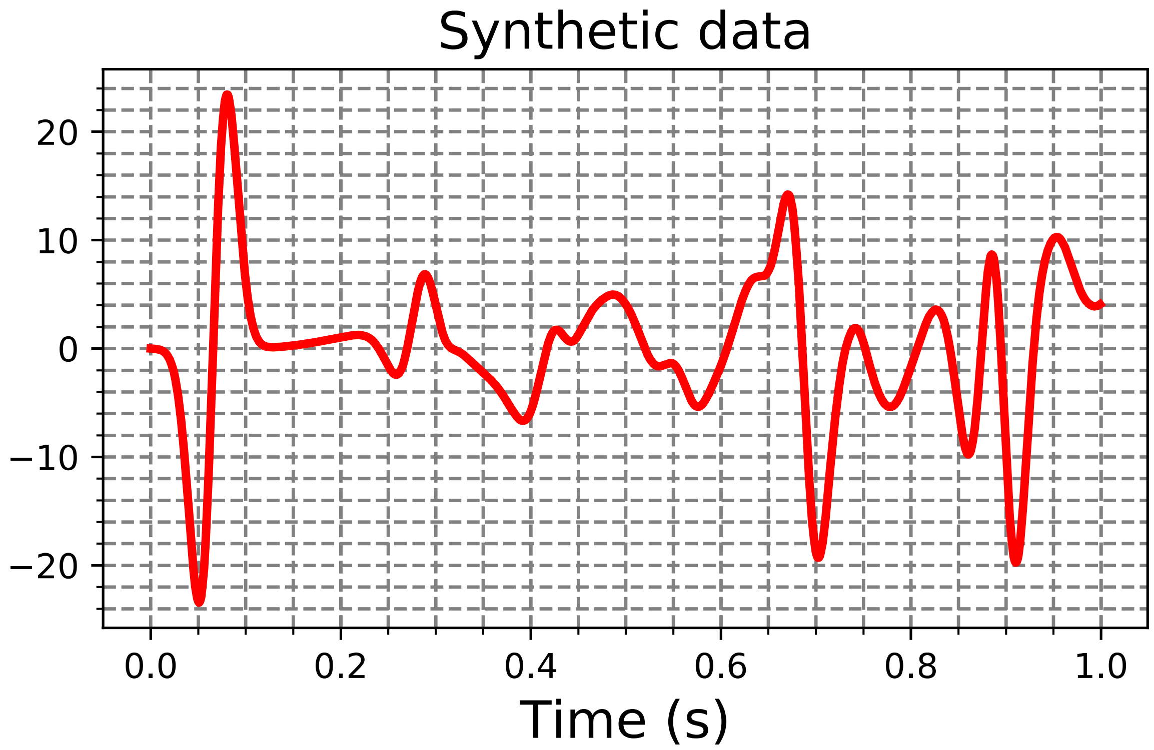
✓ Which means, training this self-designed RNN is approximately equivalent to the FWI process. In other words, FWI is also a special case of machine learning task.

Numerical analysis & tuning of hyperparameters

- Find the best learning rate ranges for gradient-based algorithms including GD, Momentum, Adagrad, RMSprop, Adam
- Several inter-comparisons

1D numerical example

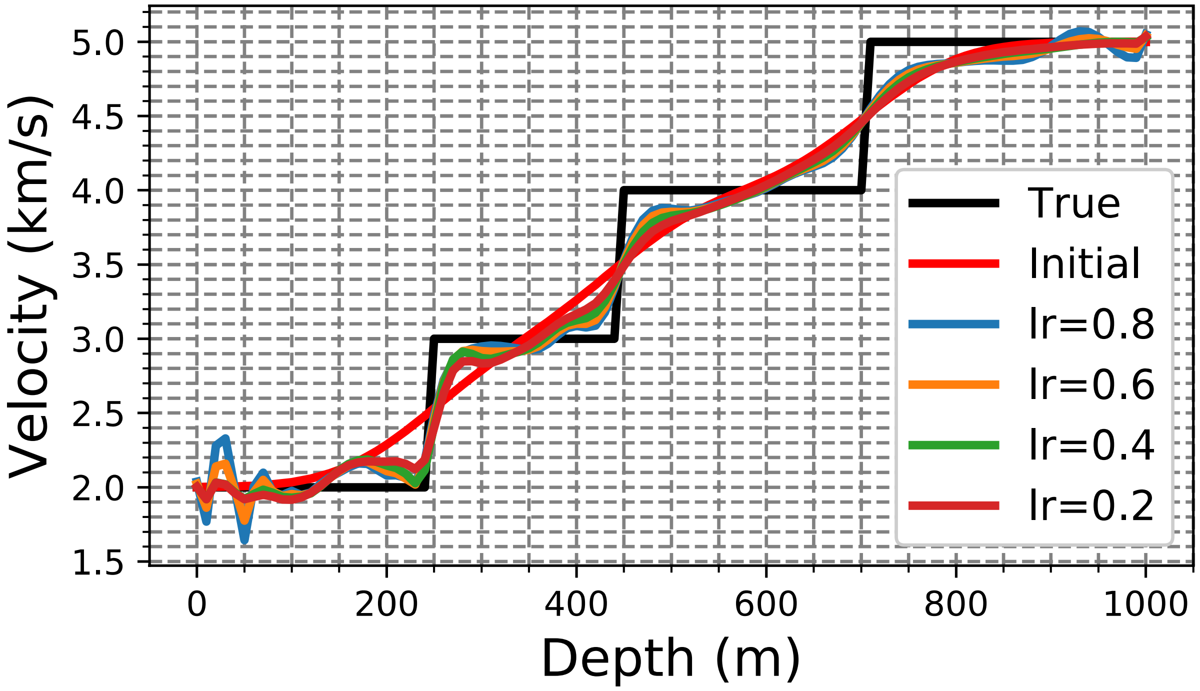


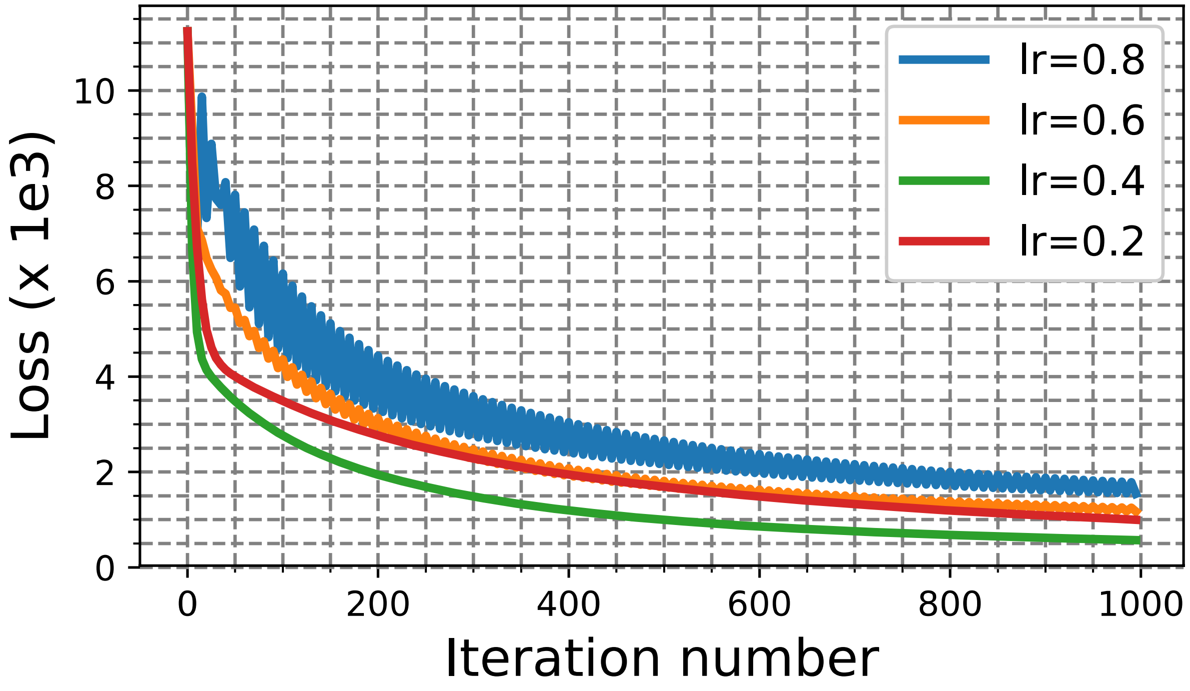


Gradient descent

$$\boldsymbol{v}_k = \boldsymbol{v}_{k-1} - \alpha \cdot \boldsymbol{g}_k$$

$\alpha \in (0,1]$





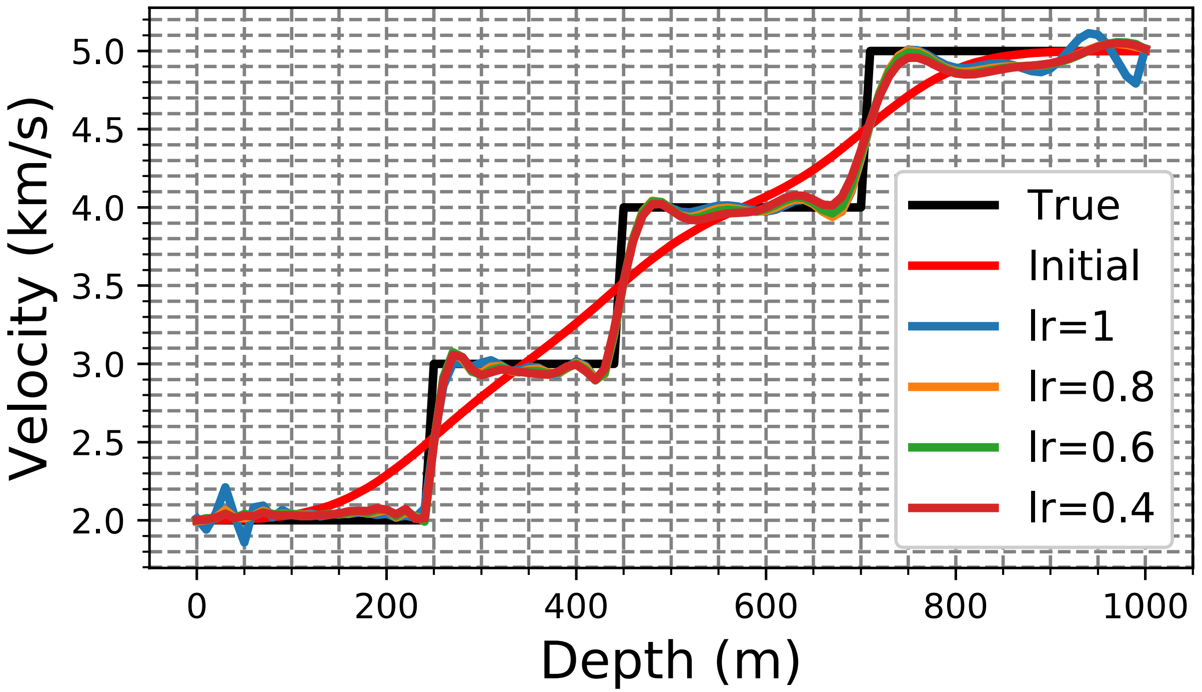
Momentum

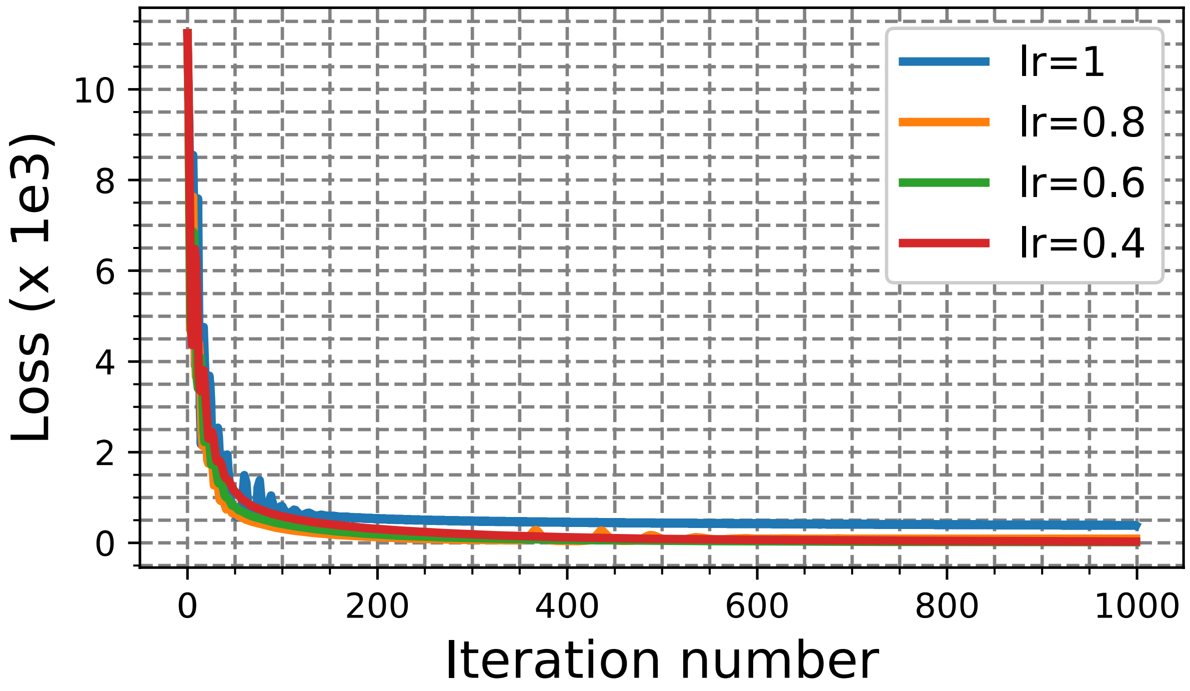
$$\boldsymbol{m}_k = \beta \cdot \boldsymbol{m}_{k-1} + (1-\beta) \cdot \boldsymbol{g}_k$$

$$\tilde{\boldsymbol{m}}_k = \boldsymbol{m}_k / (1-\beta^k)$$

$$\boldsymbol{v}_k = \boldsymbol{v}_{k-1} - \alpha \cdot \tilde{\boldsymbol{m}}_k$$

$\alpha \in (0,1]$

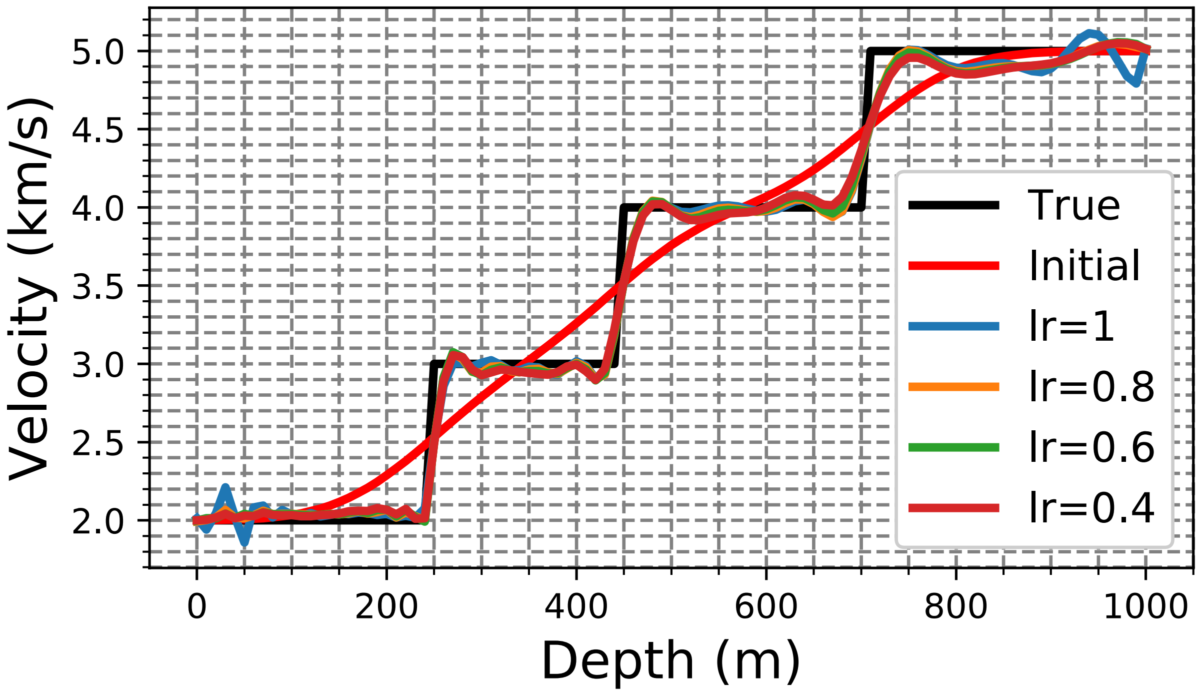


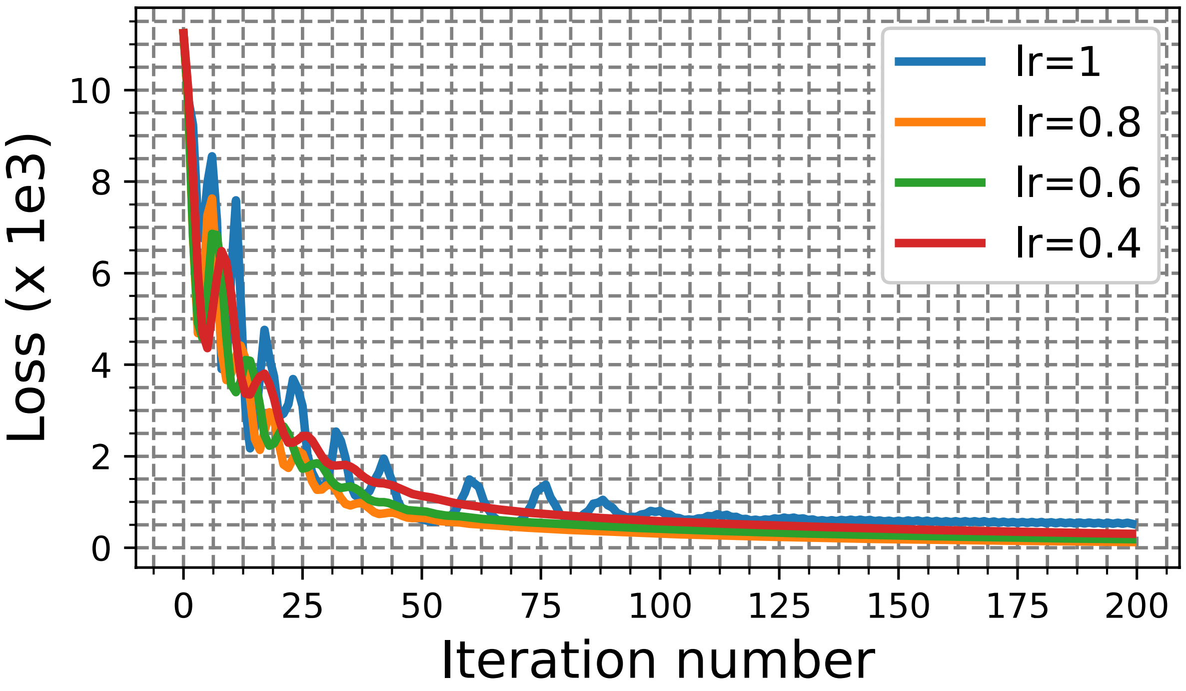


Momentum

$$\boldsymbol{m}_k = \beta \cdot \boldsymbol{m}_{k-1} + (1-\beta) \cdot \boldsymbol{g}_k$$
$$\tilde{\boldsymbol{m}}_k = \boldsymbol{m}_k / (1-\beta^k)$$
$$\boldsymbol{v}_k = \boldsymbol{v}_{k-1} - \alpha \cdot \tilde{\boldsymbol{m}}_k$$

$\alpha \in (0,1]$



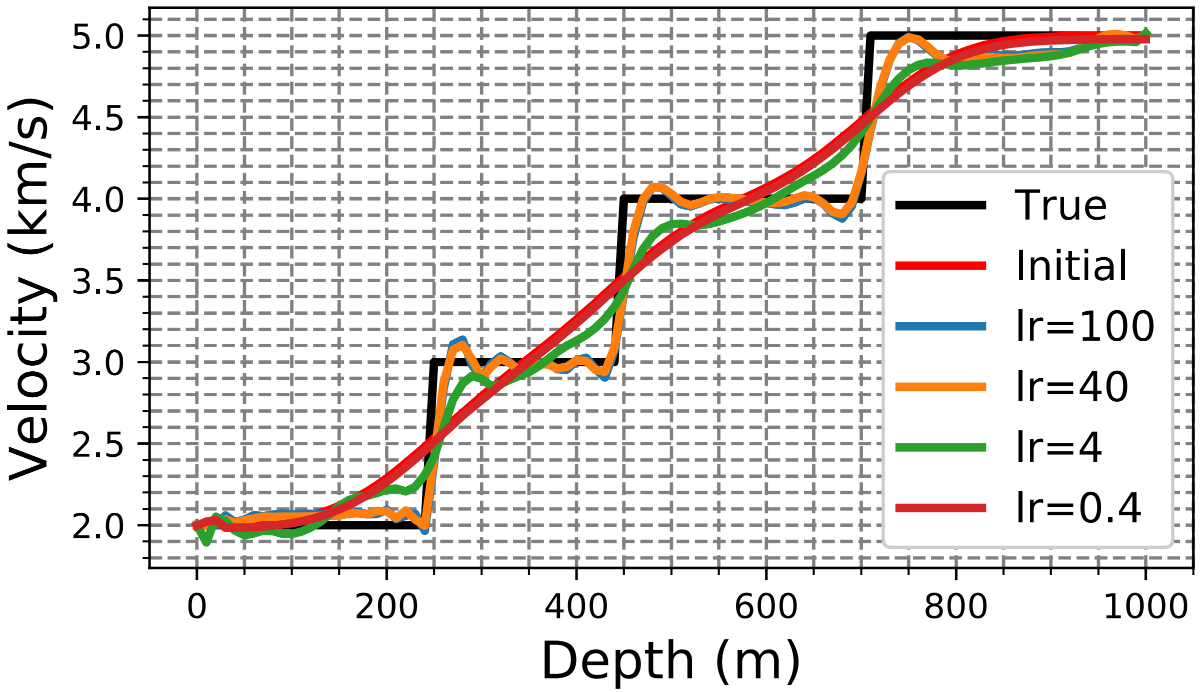


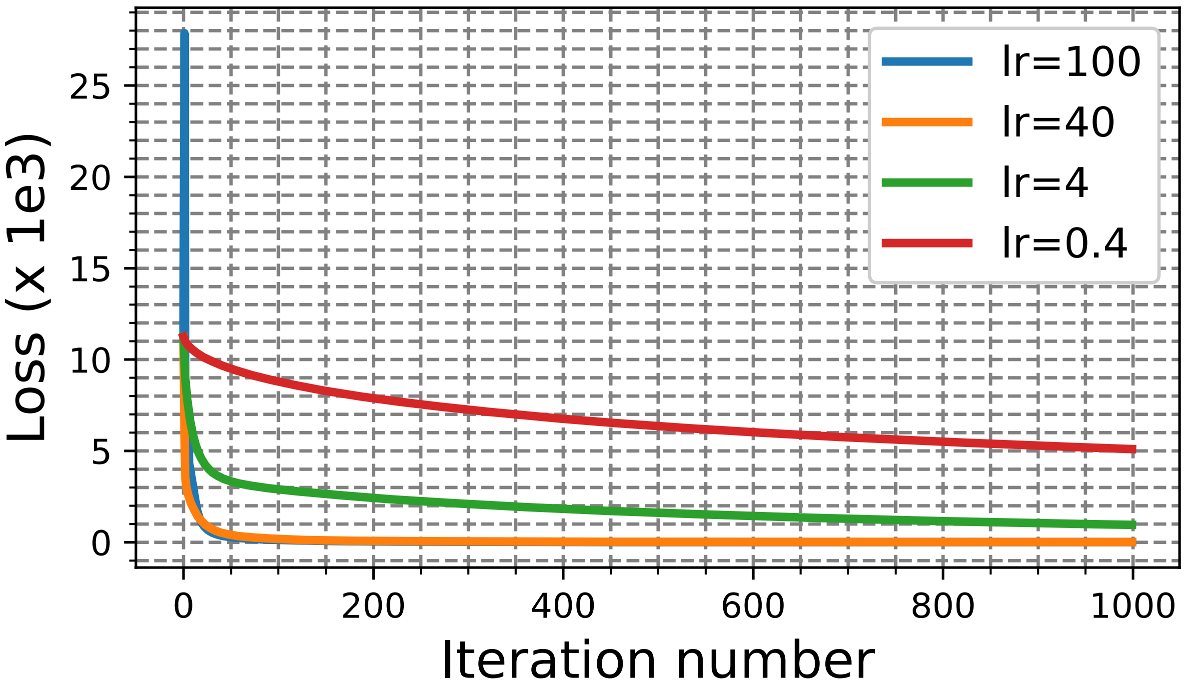
Adaptive gradient (Adagrad)

$$G_{k,ii} = G_{k-1,ii} + g_{k,i}^2$$

$$v_{k,i} = v_{k-1,i} - \boxed{\frac{\alpha}{\sqrt{G_{k,ii} + \epsilon}}} \cdot g_{k,i}$$

$\alpha \in (10,100)$





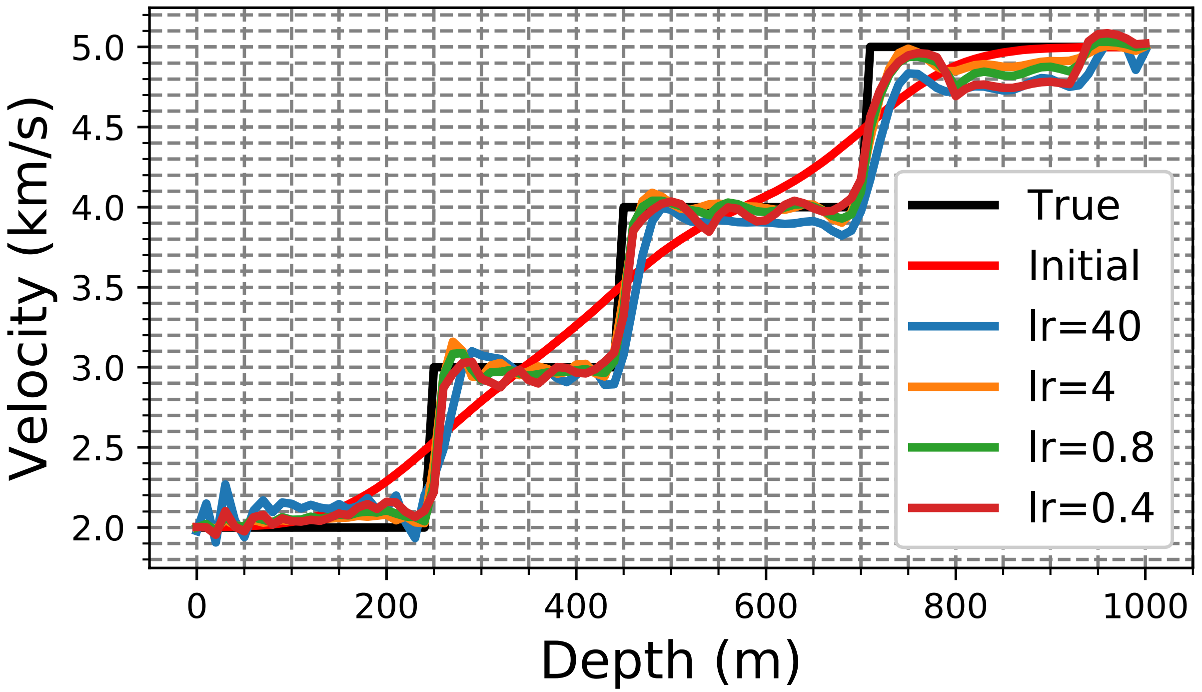
RMSprop

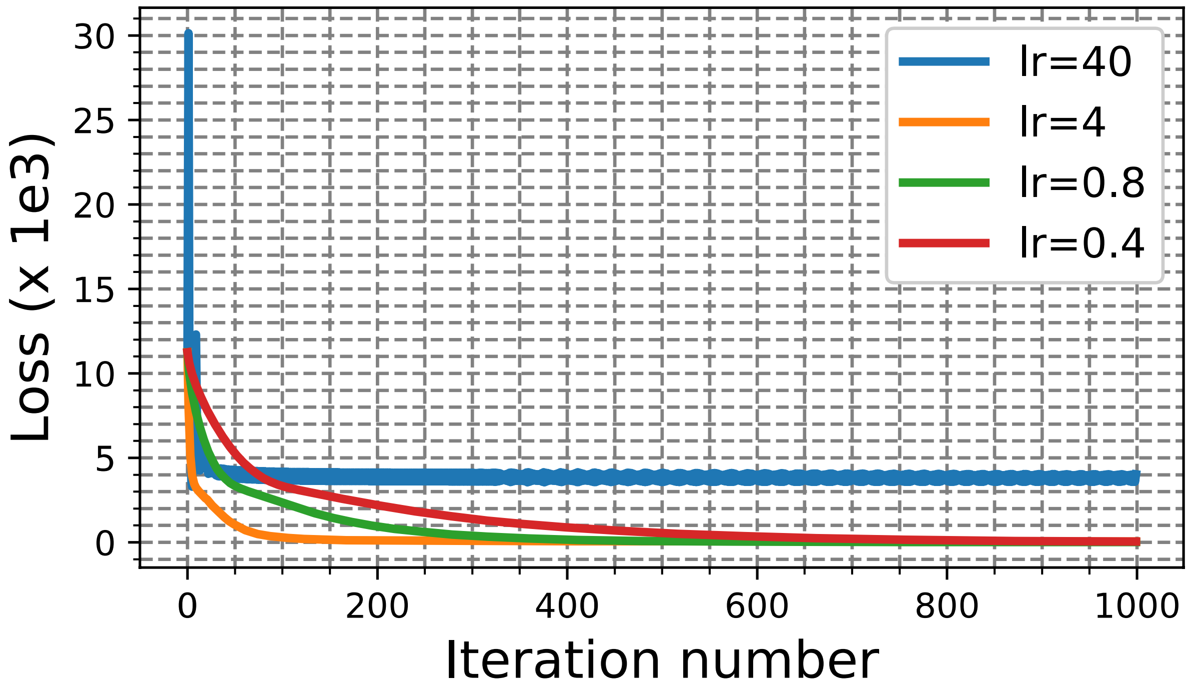
$$\boldsymbol{r}_k = \beta \cdot \boldsymbol{r}_{k-1} + (1-\beta) \cdot \boldsymbol{g}_k^2$$

$$\tilde{\boldsymbol{r}}_k = \boldsymbol{r}_k / (1-\beta^k)$$

$$\boldsymbol{v}_k = \boldsymbol{v}_{k-1} - \boxed{\frac{\alpha}{\sqrt{\tilde{\boldsymbol{r}}_k} + \epsilon}} \cdot \boldsymbol{g}_k$$

$\alpha \in (1,10)$





Adaptive moment (Adam)

$$\boldsymbol{m}_k = \beta_1 \cdot \boldsymbol{m}_{k-1} + (1-\beta_1)\cdot \boldsymbol{g}_k$$

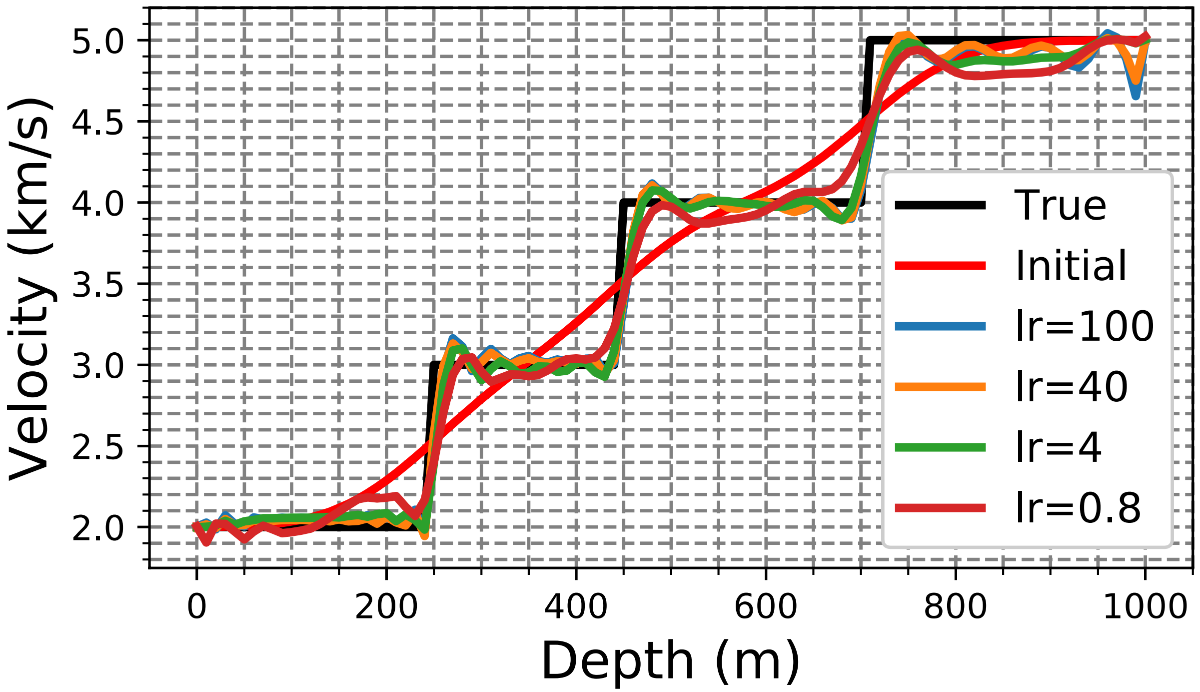
$$\boldsymbol{r}_k = \beta_2 \cdot \boldsymbol{r}_{k-1} + (1-\beta_2)\cdot \boldsymbol{g}_k^2$$

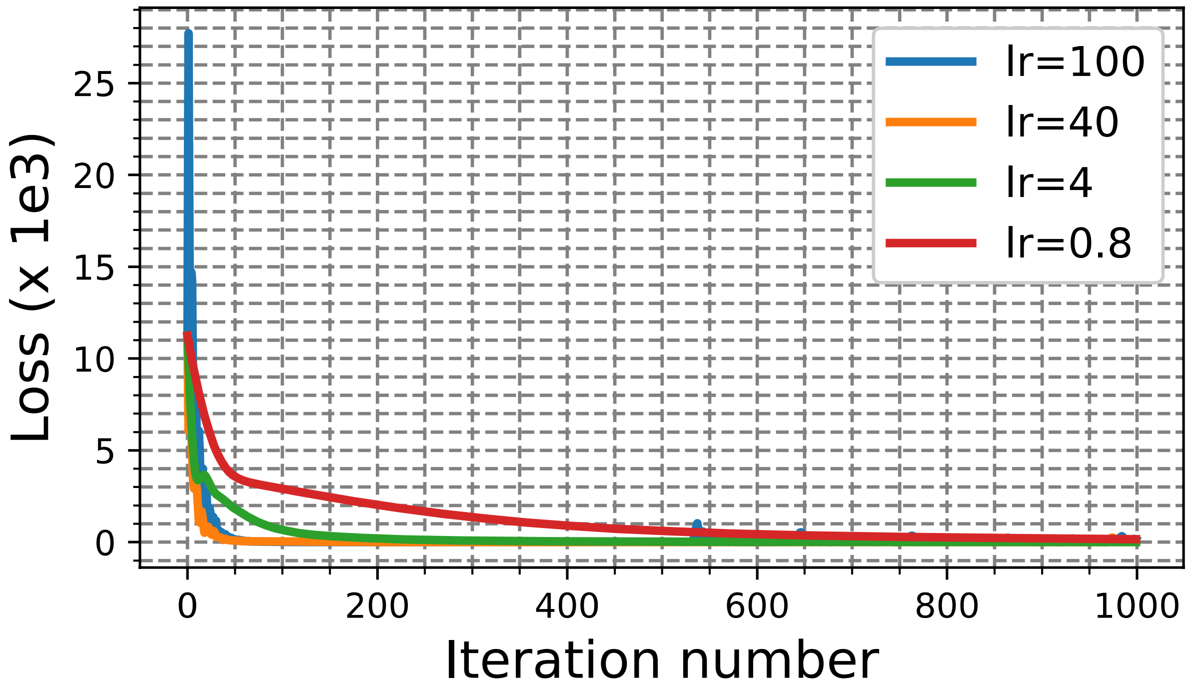
$$\tilde{\boldsymbol{m}}_k = \boldsymbol{m}_k/(1-\beta_1^k)$$

$$\tilde{\boldsymbol{r}}_k = \boldsymbol{r}_k/(1-\beta_2^k)$$

$$\boldsymbol{v}_k = \boldsymbol{v}_{k-1} - \boxed{\frac{\alpha}{\sqrt{\tilde{\boldsymbol{r}}_k}+\epsilon}} \cdot \tilde{\boldsymbol{m}}_k$$

$\alpha \in (10{,}100)$





Adaptive moment (Adam)

$$\boldsymbol{m}_k = \beta_1 \cdot \boldsymbol{m}_{k-1} + (1-\beta_1)\cdot \boldsymbol{g}_k$$

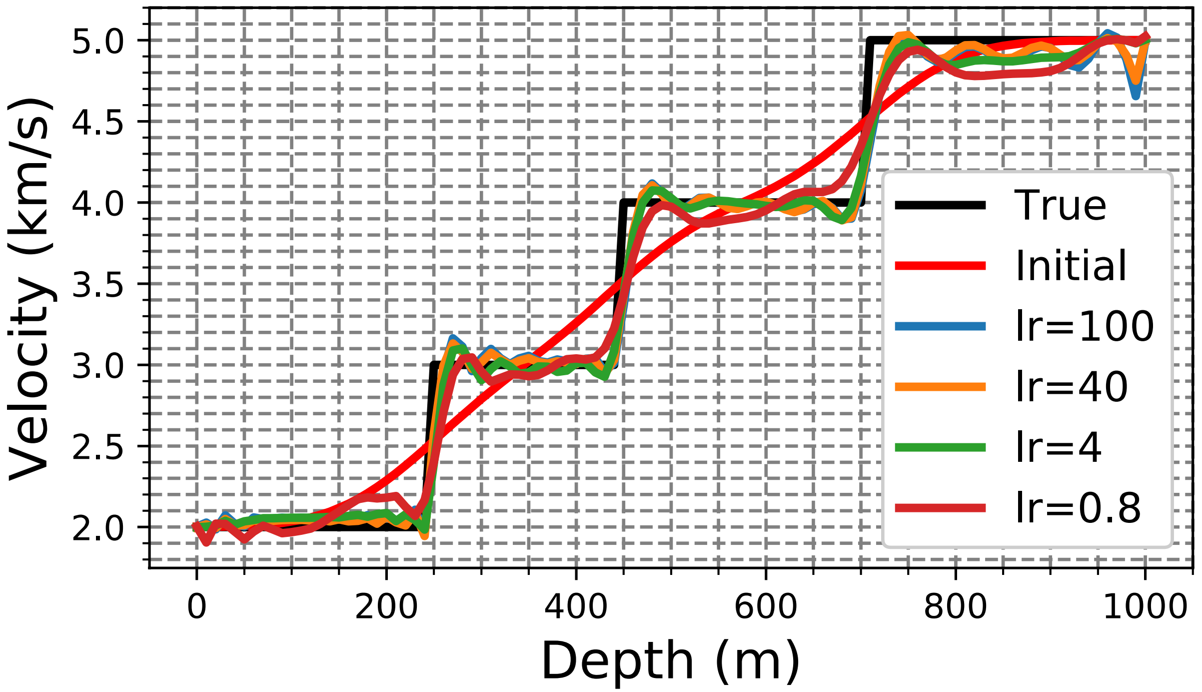
$$\boldsymbol{r}_k = \beta_2 \cdot \boldsymbol{r}_{k-1} + (1-\beta_2)\cdot \boldsymbol{g}_k^2$$

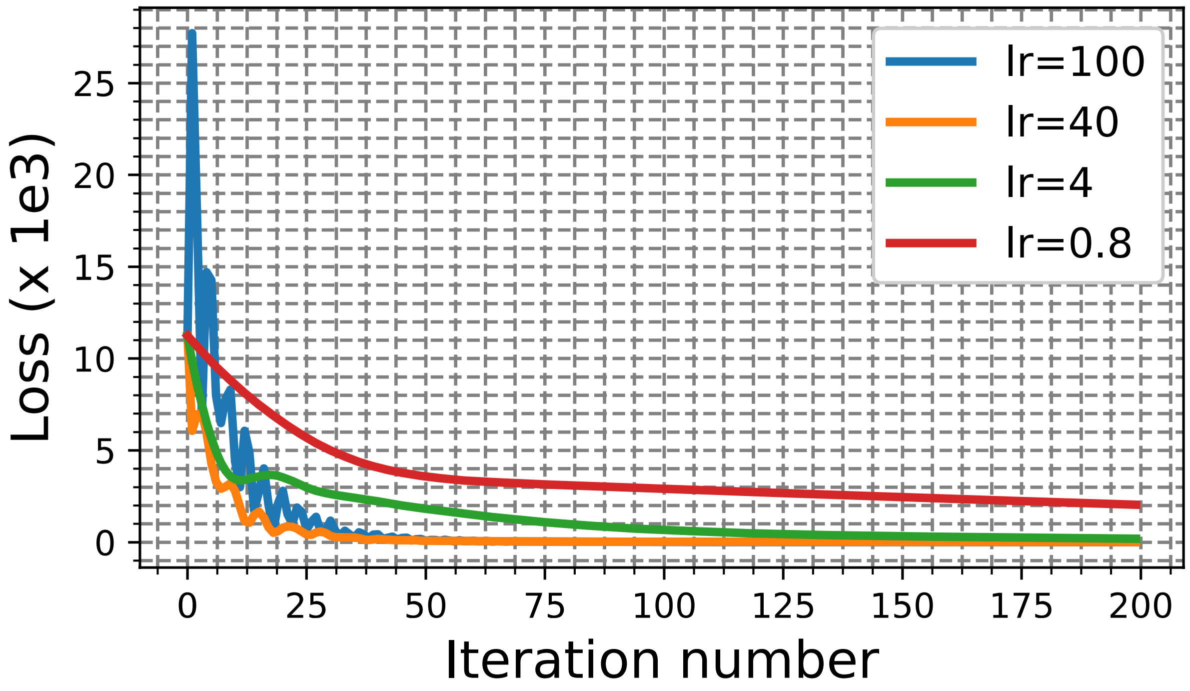
$$\tilde{\boldsymbol{m}}_k = \boldsymbol{m}_k/(1-\beta_1^k)$$

$$\tilde{\boldsymbol{r}}_k = \boldsymbol{r}_k/(1-\beta_2^k)$$

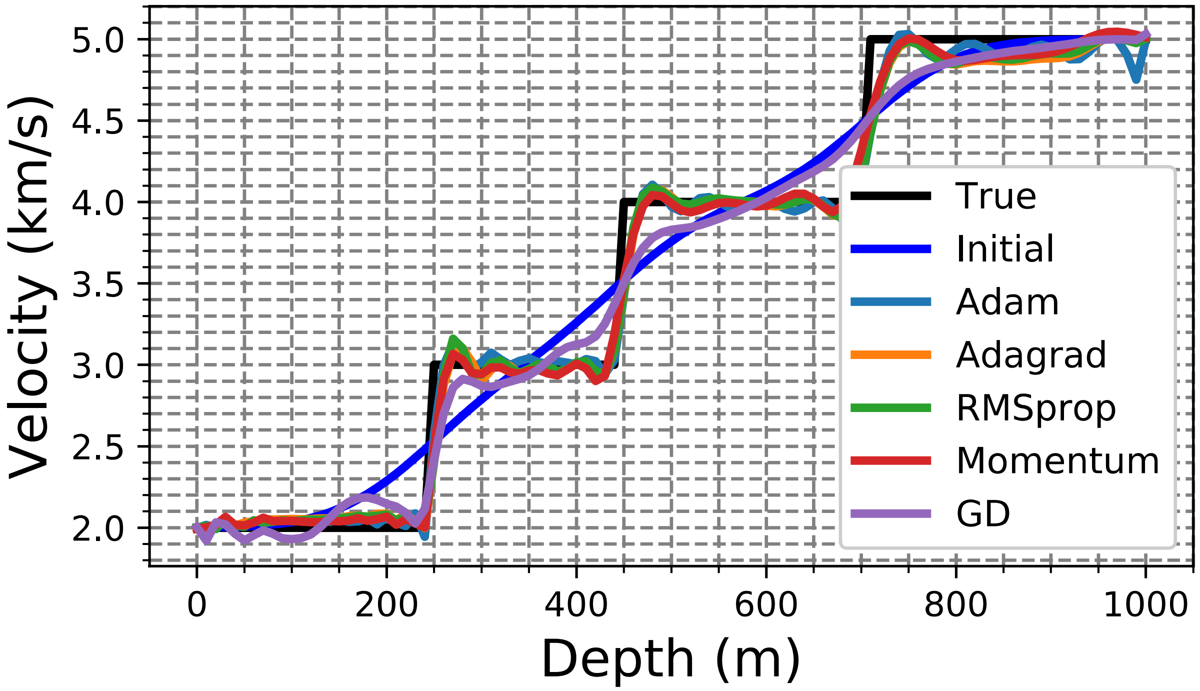
$$\boldsymbol{v}_k = \boldsymbol{v}_{k-1} - \boxed{\frac{\alpha}{\sqrt{\tilde{\boldsymbol{r}}_k}+\epsilon}} \cdot \tilde{\boldsymbol{m}}_k$$

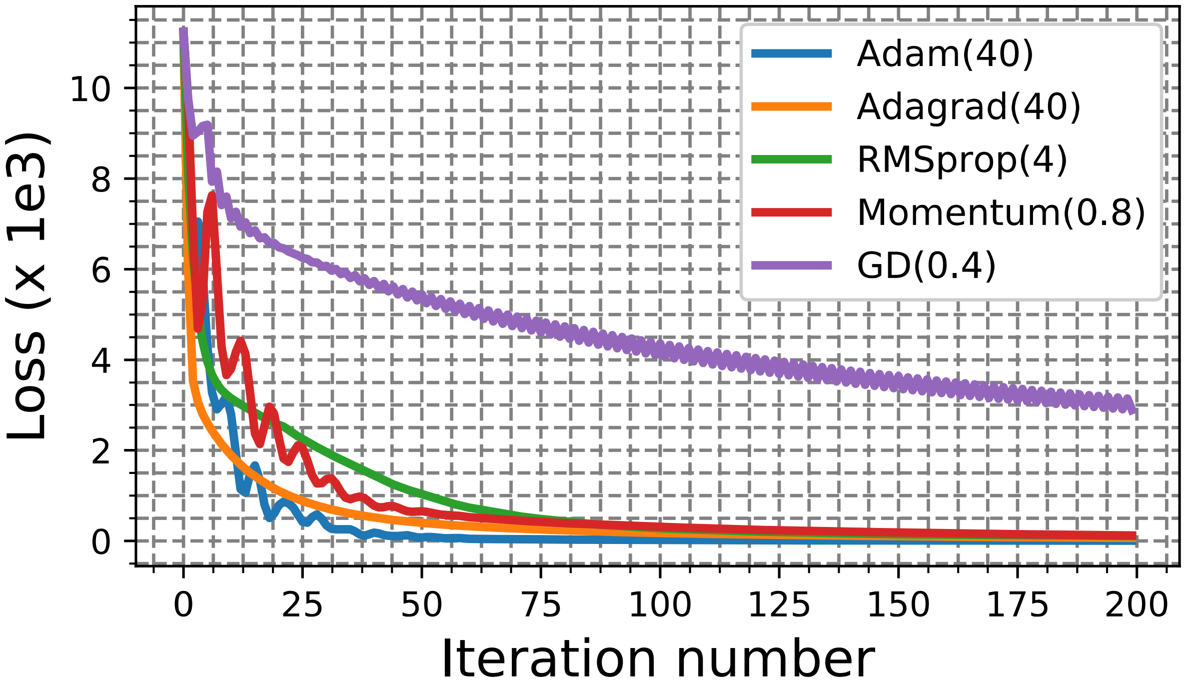
$\alpha \in (10{,}100)$



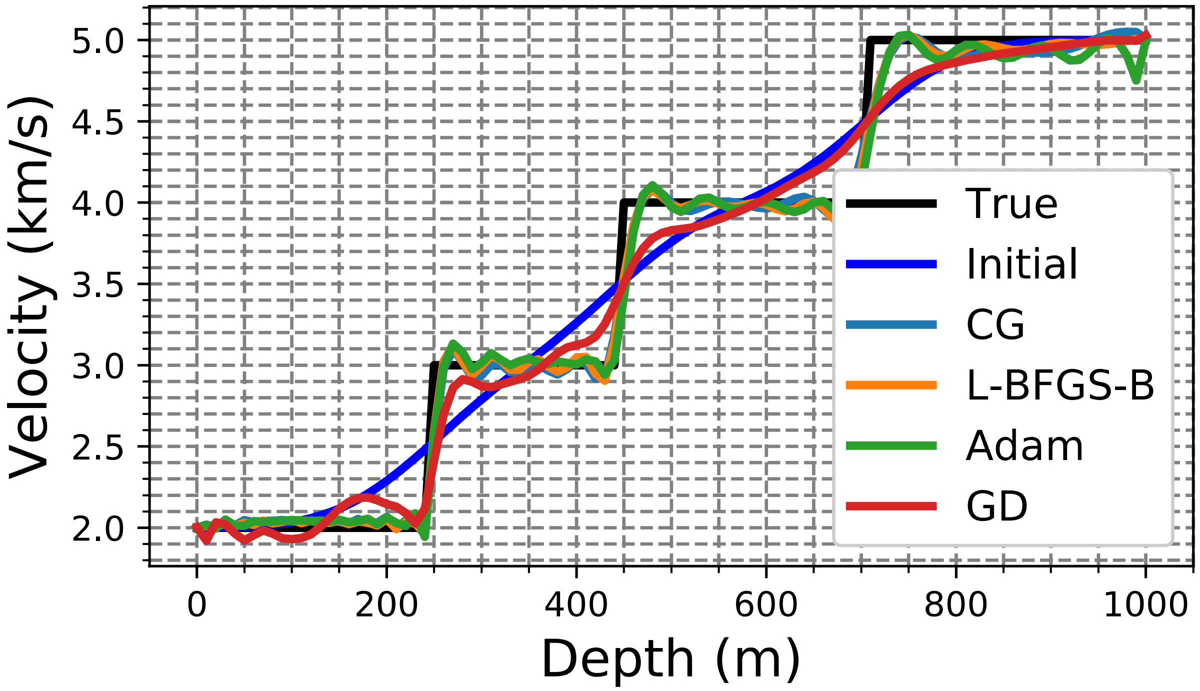


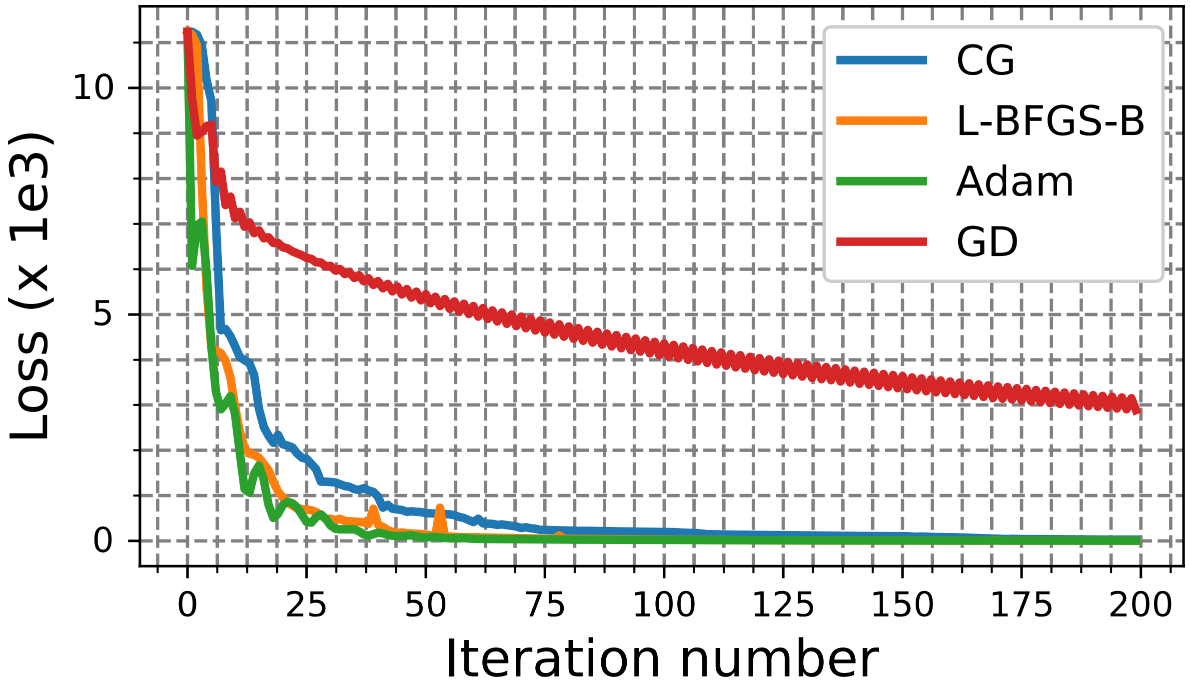
Comparisons of gradient-based algorithms





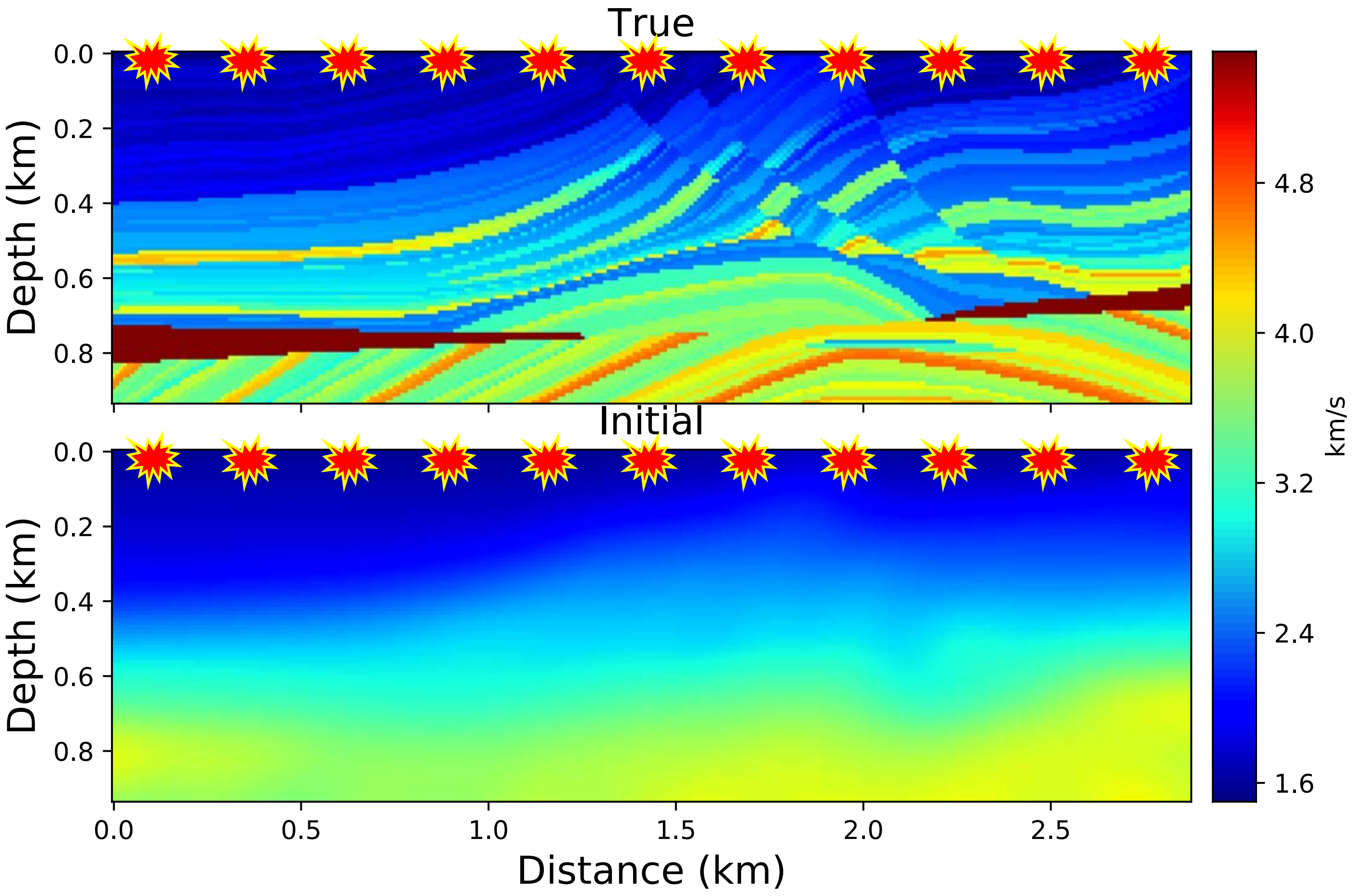
Comparison of Adam, CG, L-BFGS



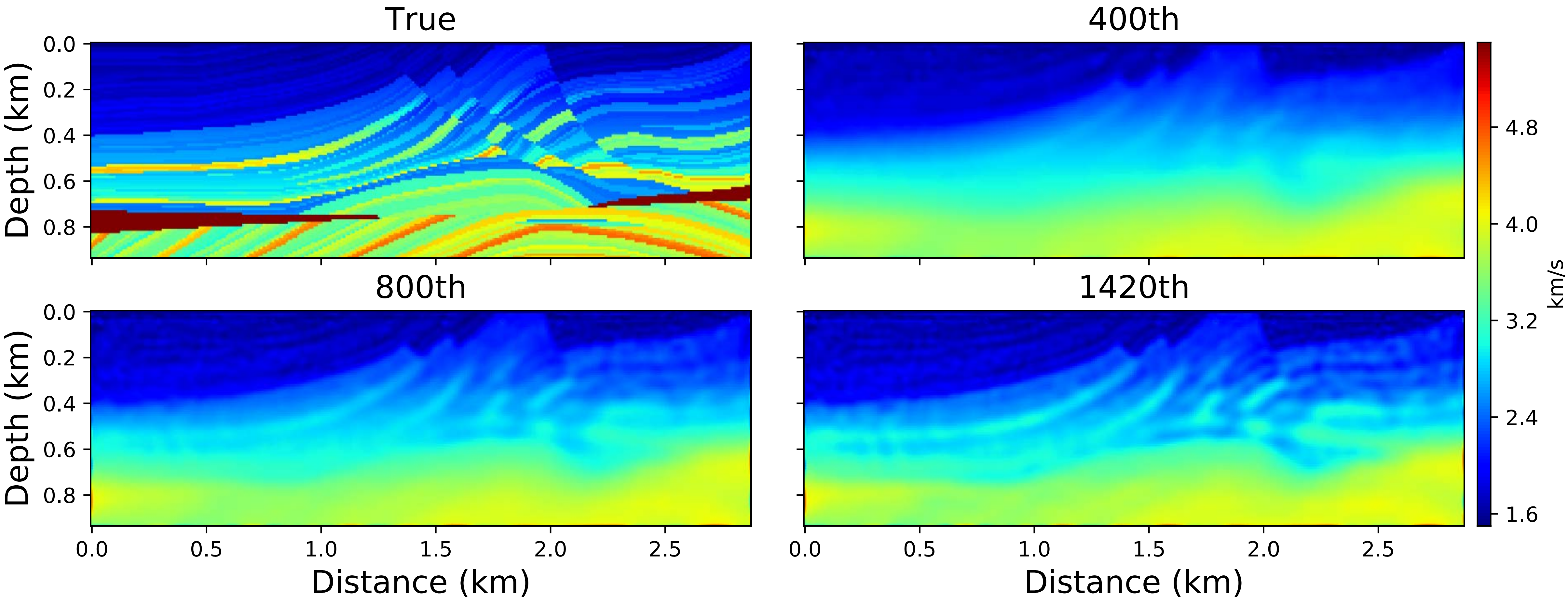


Synthetic test on Marmousi

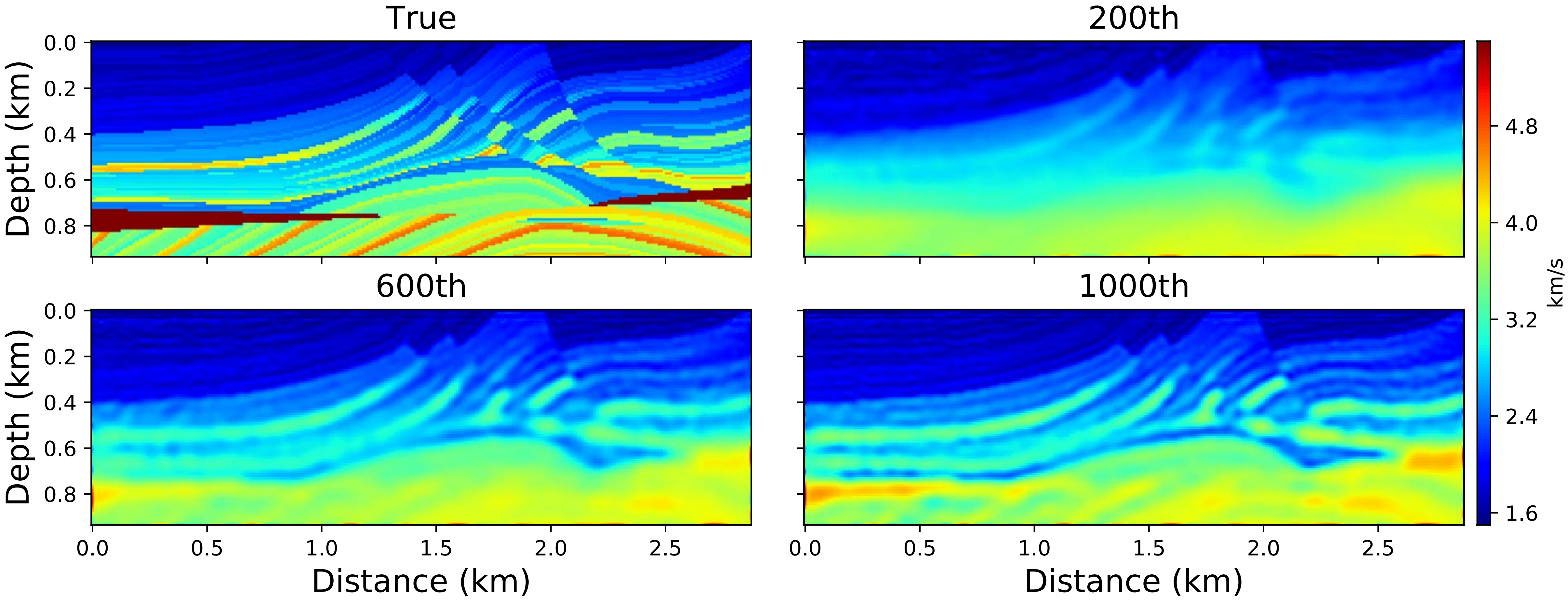
- Marmousi: 11 shots (12Hz)



Non-linear CG

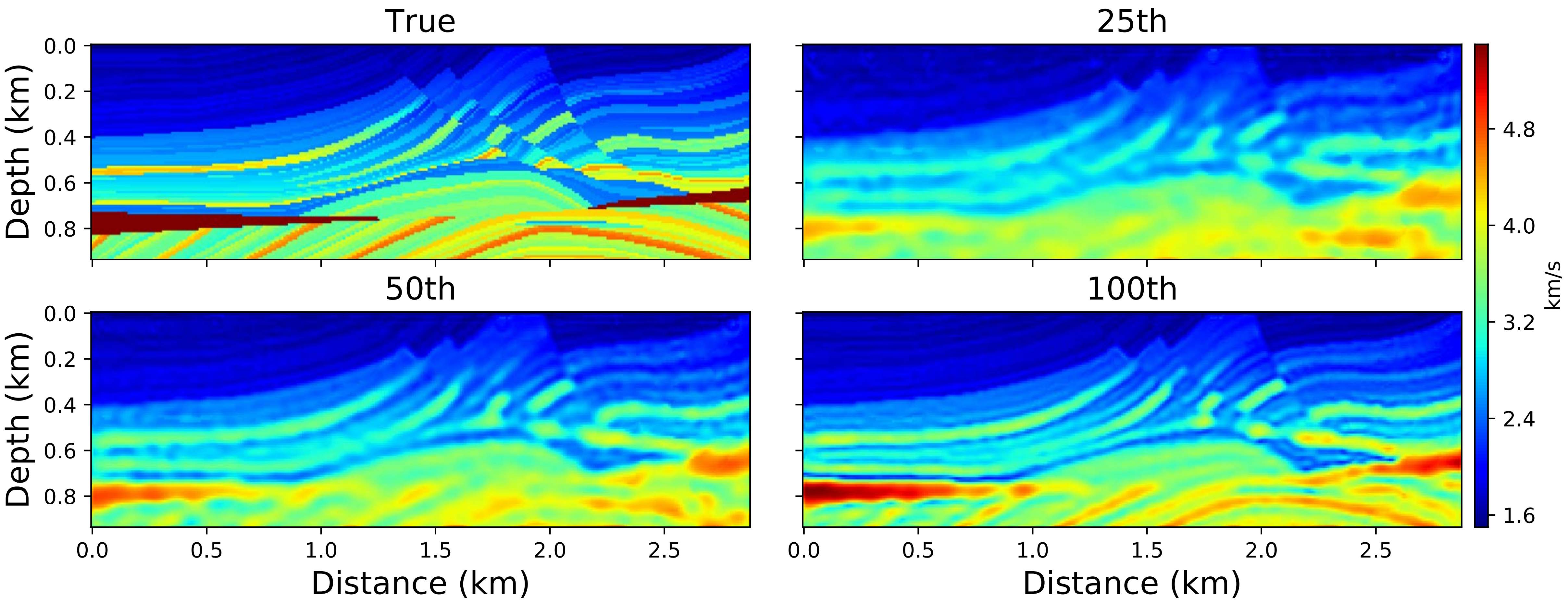


L-BFGS

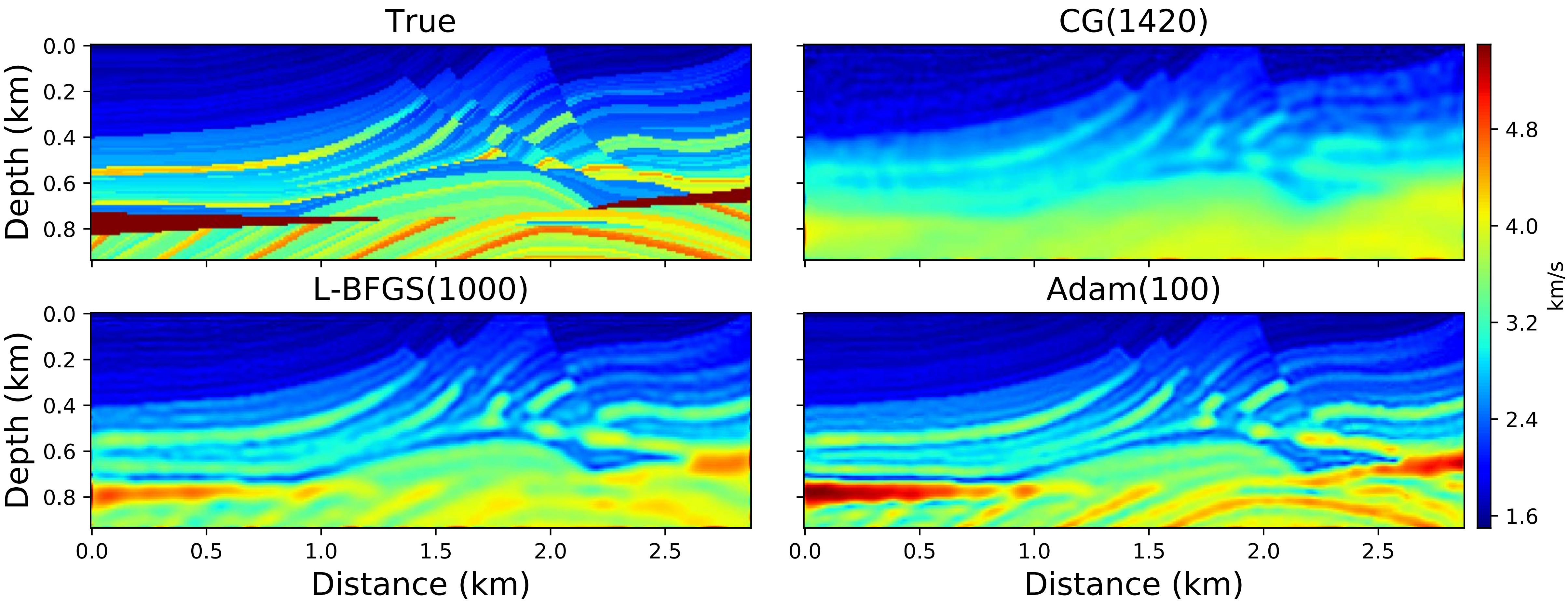


True
200th
600th
1000th
Depth (km)
0.0
0.2
0.4
0.6
0.8
Distance (km)
0.0
0.5
1.0
1.5
2.0
2.5
km/s
4.8
4.0
3.2
2.4
1.6

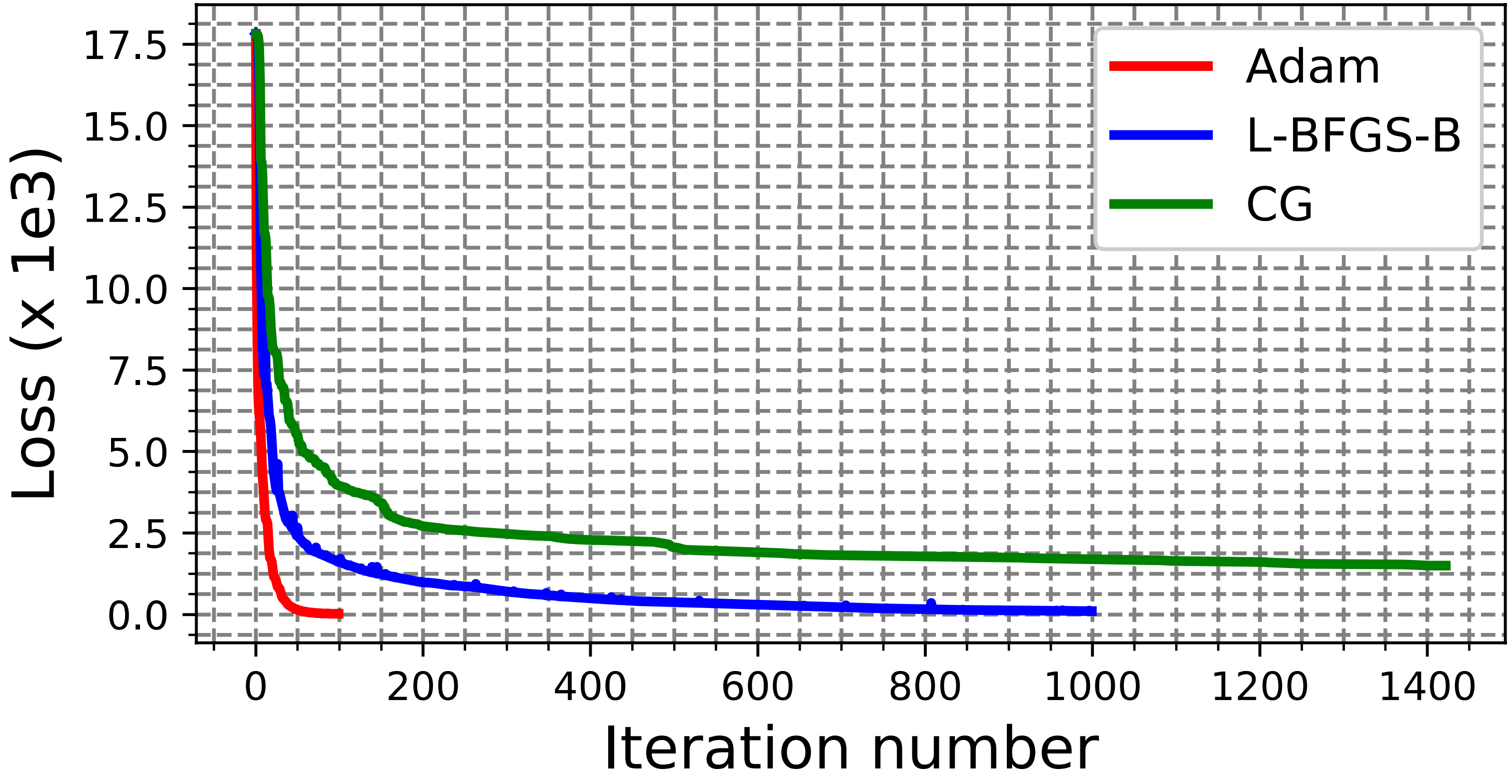
Adam (α=40)



Comparison


True
CG(1420)
L-BFGS(1000)
Adam(100)
Depth (km)
0.0
0.2
0.4
0.6
0.8
Distance (km)
0.0
0.5
1.0
1.5
2.0
2.5
km/s
1.6
2.4
3.2
4.0
4.8

Comparison: loss



Conclusions & Future works

- Conclusions:
  - We illustrated a self-designed RNN framework for forward and inverse seismic modelling.
  - Proved that FWI is a special case of a machine learning process.
  - Best learning rate ranges of gradient-based algorithms were analyzed and investigated for velocity model building.
  - The efficiency of gradient-based and non-linear optimization algorithms are compared and discussed.

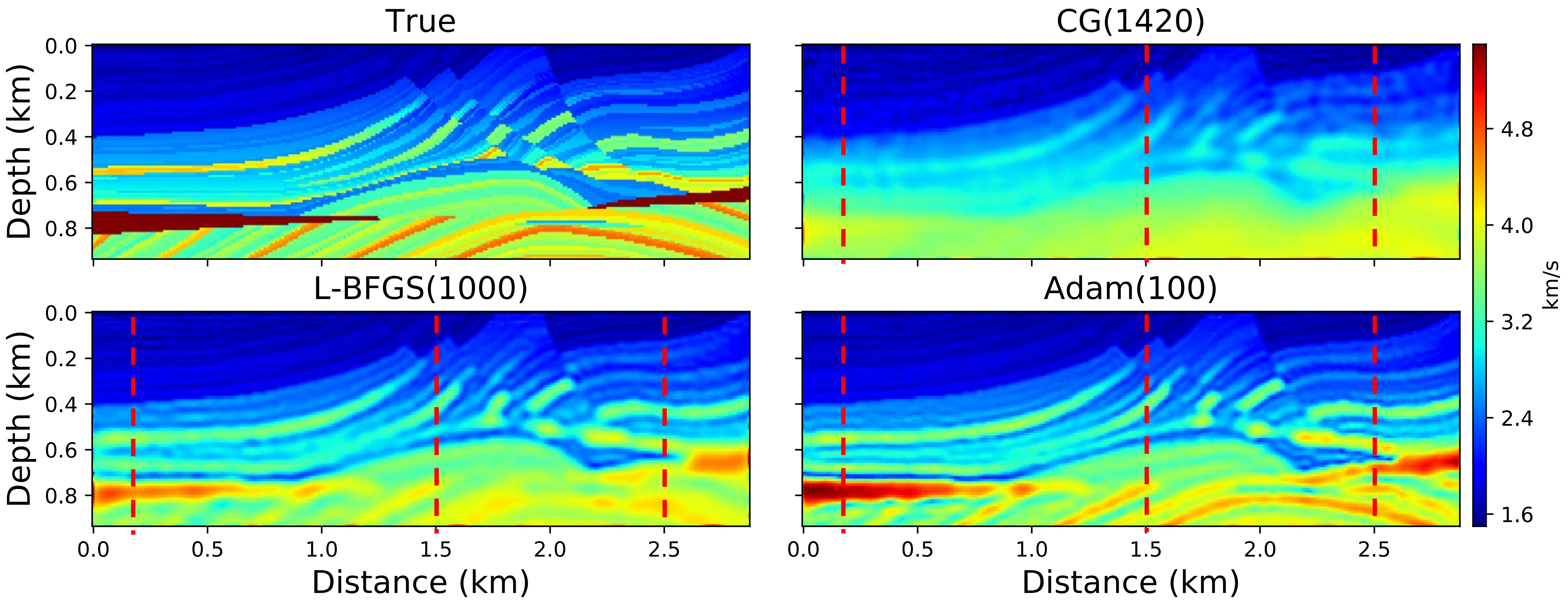
- Future works:
  - A theory-guided neural network.
  - A physical-teaching training process.

Acknowledgements

- All CREWES sponsors
- NSERC (CRDPJ 46117913)
- Kiki Xu, Sergio Romahn, Lei Yang, Xin Fu
- All CREWES staff and students

Comments & Questions ?

Comparison: v-model



Comparison: v-profiles