Software Documentation for Time-Domain Convolutional-neural-network-based Elastic Reflection Full Waveform Inversion

Yulang Wu

Thursday, September 09, 2021

1. Introduction

Elastic Full waveform inversion (EFWI) is a powerful imaging algorithm in exploration seismology to provide high-resolution elastic models (P-wave velocity, S-wave velocity, and density models), by fitting the simulated seismic data to the observed seismic data. The EFWI mainly fits the first arrival in the observed data to update the shallow part of the elastic models. However, the observed reflected events which contain information of the deep part of the elastic models, cannot be effectively matched by the EFWI. Besides, as the elastic parameters are more or less coupled (e.g., overlapping radiation patterns of P-wave velocity, S-wave velocity, and density models), the EFWI is more ill-posed than the monoparameter (P-wave velocity) acoustic FWI. Therefore, the EFWI faces a cross-talk issue, especially when it is applied to reconstruct the density model.

This document describes software for the time-domain convolutional-neural-network-based elastic reflection full-waveform inversion (CNN-ERFWI). Instead of fitting the first arrival in the observed data, the CNN-ERFWI effectively utilizes the starting elastic models and the corresponding migration images, obtained from the observed reflected events in the observed data, to invert for both the shallow and deep parts of the elastic models. CNN-ERFWI not only reduces the cross-talk in elastic full-waveform inversion but also accurately inverts for both the shallow and deep parts of the elastic models.

2. Summarized Theory

CNN-ERFWI is an iterative inversion to apply three mutually independent CNNs to predict the P-wave velocity, S-wave velocity, and density models, respectively, from the corresponding starting model and the elastic migration images. CNN-ERFWI mainly contains two parts at each iteration: training dataset preparation, CNN training, and prediction. Figure 1 shows the workflow of the CNN-ERFWI.

1) Training dataset preparation (platform: MATLAB and FORTRAN90)

The preparation of the training dataset includes the preparation of the training elastic models and the preparation of the training starting elastic models and the corresponding RTM images.

[1] The preparation of the training elastic models (in the purple box in Figure 1). To create the training elastic models, a spatially-constrained divisive hierarchical k-means method is applied, to partition the starting (at the first

iteration) or the latest CNN-predicted (at the following iterations) P-wave velocity model into disjoint features. Then, the training P-wave velocity, S-wave velocity, and the density model sets are obtained by assigning random values, drawn from their distributions in the disjoint features, to themselves, respectively.

[2] The preparation of the training starting elastic models and the corresponding RTM images (the upper parts in the red box in Figure 1). The starting elastic models are obtained by smoothing the training elastic models. The migration images are obtained by elastic reverse-time migration (RTM).

2) CNN training and prediction (platform: Tensorflow 1.0 and Python 3)

After the training dataset is prepared, the training starting elastic models and the RTM images are the input data to train CNNs to accurately reproduce the corresponding training elastic models (the lower part in the red box in Figure 1). Then, the trained CNNs are applied to predict the unknown true elastic models from the corresponding starting elastic models and the RTM images (in the black box in Figure 1). Then, the CNN-predicted elastic models are used to create the training dataset for the next iteration.

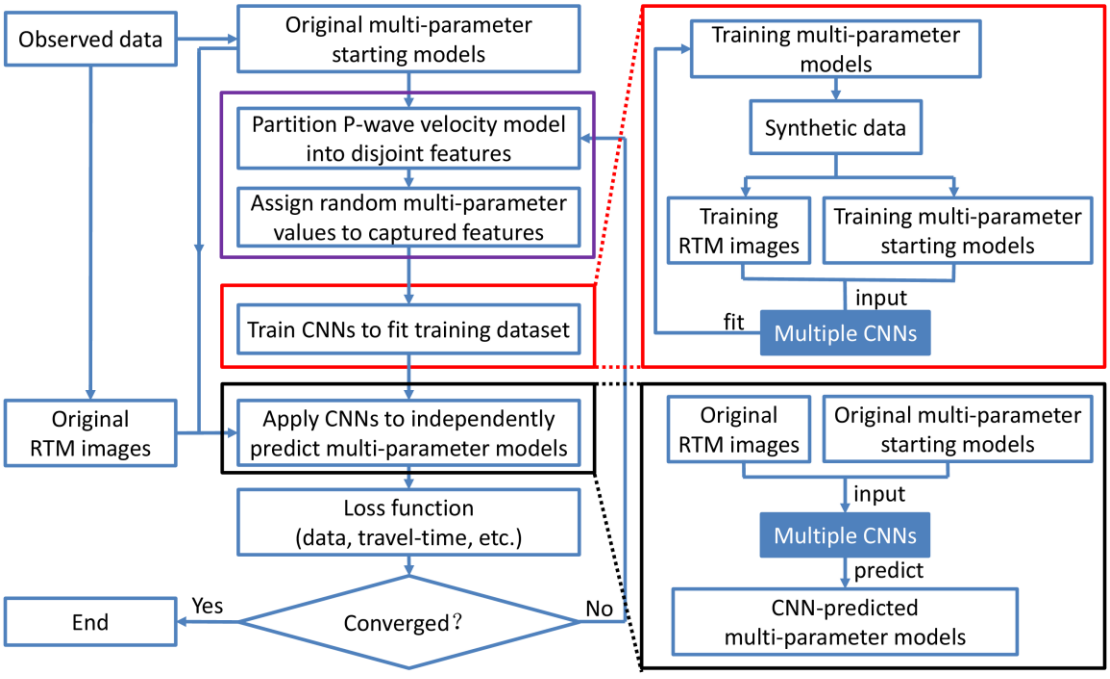

Figure 1 Multi-parameter CNN-based reflection full-waveform inversion

3. User guide with examples

Environment:

[1]Python 3: Anaconda3-5.3.0-Linux-x86_64.sh

[2]Tensorflow: tensorflow-gpu==1.13.2

[3]MATLAB

[4]FORTRAN90

1) Modify the parameters for inversion:

[1] Modify the parameters in the shell script (all_in.csh) under the directory (./)

```
#parameters ================================================================================
nz=256
nx=256
num_threads=4 #number of threads to run RTM images
iteration=0    #The first iteration of the CNN-RFWI starts at 0 (th iteration)
iter=50 #The last iteration of the CNN-RFWI
num_samples=24 #The number of samples for CNN training
vel_dir='models/' #The velocity directory to store the predicted and smooth models
fortran_dir='fortran1.0/'
matlab_dir='matlab1.0/'
CNN_dir='CNN_approximation1.0/'
CNN_result_dir="CNN_results/"
CNN_max_epochs=1600
store_weights_freq=1600
CNN_last_epochs=`expr $CNN_max_epochs - 1`
CNN_max_epochs_vs=2800
store_weights_freq_vs=2800
CNN_last_epochs_vs=`expr $CNN_max_epochs_vs - 1`
postfix=""
sleep_time=60 #unit is second
sh_num_smooth_iteration=800
sh_filter_size=3
sh_water_depth=13 #Marmousi230z6500x
#remember to manually set total time step for Fortran
#============================================================================================
```

[2] Modify the parameters in the parameter.txt file under the directory (./fortran1.0) (reference: ./fortran1.0/common/globalvar.f90).

```
npml =           10
nx =            256
dx =     12.50000
nz =            256
dz =     12.50000
f0 =     20.00000
t0 =    0.1650000
factor =    1.0000000E+07
izs =             5
ixs =            64
nt =           4000
dt =    1.0000000E-03
it_display =           100
truevp =
true

initvp =
mig

incre_t =             1
incre_s =            10
num_shot =           100
```

[3] Put the files of the elastic models under the directory (./matlab1.0/models)

[4] Change the corresponding filenames in
./ matlab1.0/models/prepare_corresponding_smooth_starting_mig_model.m

```
vp = dlmread(['Sigsbee_vp350x1.dat']);
vs = dlmread(['Sigsbee_vs350x1.dat']);
rho = dlmread(['Sigsbee_rho350x1.dat']);
```

2) **Run CNN-ERFWI** (directory: ./)
source all_in.csh

3) Plot the data (MATLAB platform) (directory: ./fortran1.0)
[1] Calculate the data on the elastic models at each iteration
source batch_inversion_data.csh

[2] Plot the data
run plot_data.m

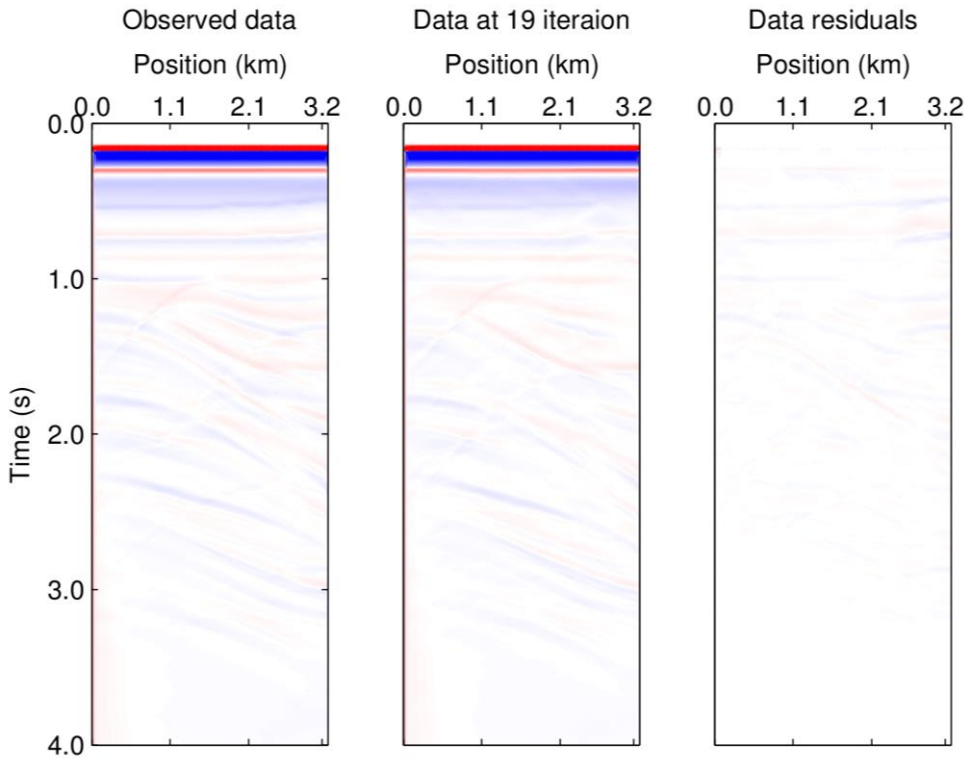

Figure 2 Comparison between the observed data and the final synthetic data.

4) Plot the model (MATLAB platform) (directory: ./ matlab1.0/Marmousi)
run SCA_plot_3x3portions_velocity.m

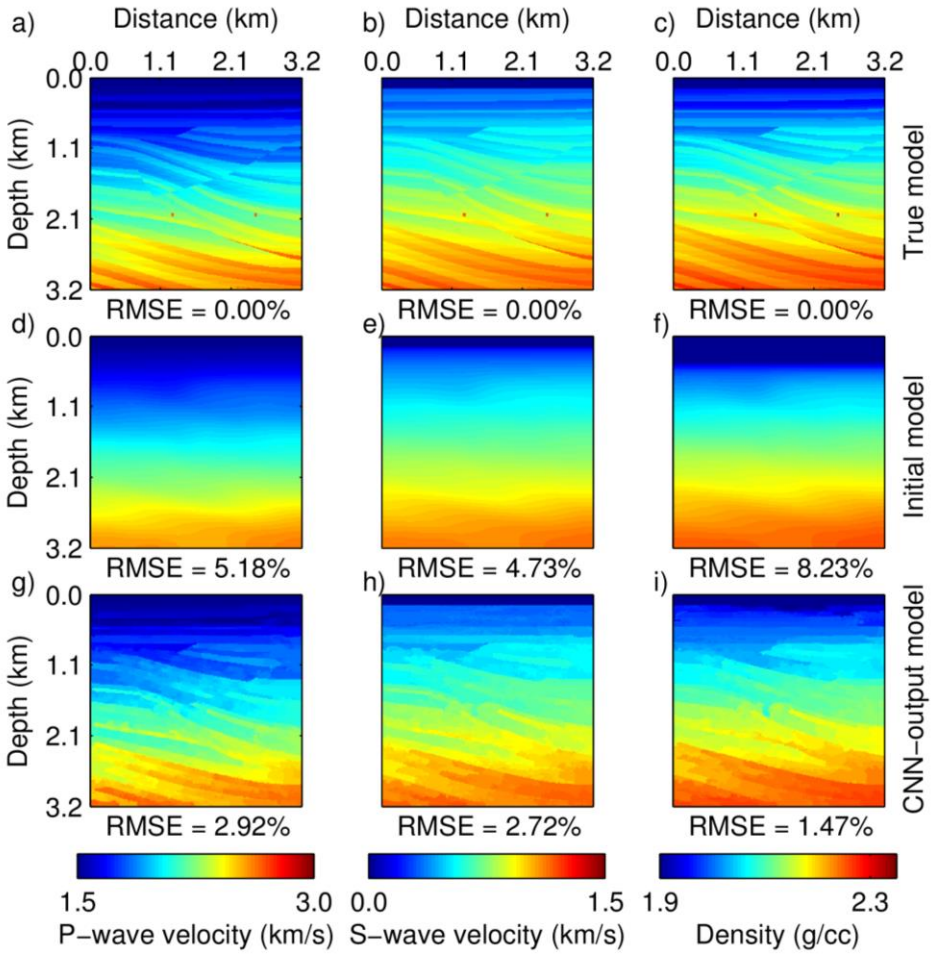

Figure 3 Comparison of the elastic models

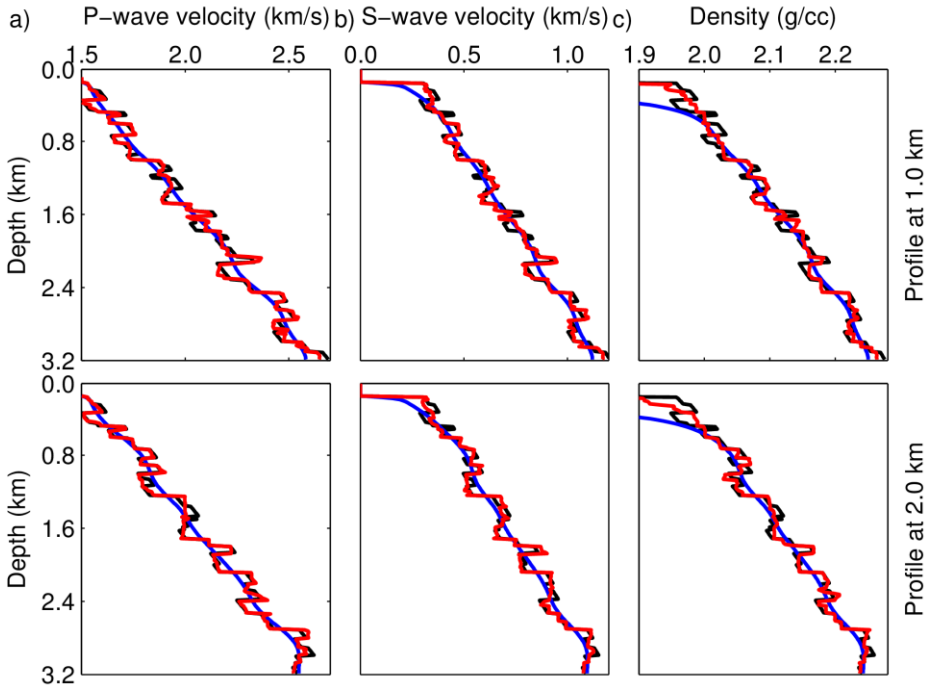

Figure 4 Comparison of the profiles in the elastic models