Submission code: D9RGAPJRB6

See this and other abstracts on our website: https://home.sbgf.org.br/Pages/resumos.php

Automatic NMO velocity estimation via Recurrent Neural Network (RNN)

Joan Arruda (UNICAMP), Joerg Schleicher (UNICAMP), Amélia Novais (UNICAMP)

Automatic NMO velocity estimation via Recurrent Neural Network (RNN)

Please, do not insert author names in your submission PDF file.

Abstract Summary

Manual picking on Semblance spectra is the standard process to determine normal moveout (NMO) velocities. Stacking of NMO corrected CMP gathers allows to approximate traces in a zero-offset section. To speedup this process and increase its reliability, we use a Recurrent Neural Network. The input to the proposed network is a time window of a CMP section with fixed duration, with its starting time being interpreted as a zero-offset time. Output of the RNN is the classification of the initial time of the window as a zero-offset time corresponding to a reflection event or not. Our network started to work after little training with very few models. The first numerical tests demonstrate the potential of the proposed methodology to detect events and determine their correct zero-offset traveltimes.

Introduction

Primary reflections from subsurface interfaces recorded in a Common Midpoint (CMP) geometry exhibit approximately hyperbolic traveltime curves. Semblance analysis is used to determine whether the recorded traveltimes follow a hyperbolic trajectory for a given zero-offset time t_0 and normal moveout (NMO) velocity v_{nmo}. Semblance values range between $[0, 1]$, and the objective is to select optimal points based on peak semblance values (see, e.g., Yilmaz, 1987). However, this approach presents several challenges: Multiple-reflection events may exhibit high semblance values, more than one primary event can exist for a given t_0 creating a conflicting-dip scenario and selection of appropriate picking requires some expertise to avoid systematical errors (bias).

In traditional manual picking, analysts simultaneously select (t_0, v_{nmo}) pairs. However, manual picking is a time-consuming, high-cost task. Our approach starts at a RNN predicting only the zero-offset times t_0 associated with a primary refection event. The NMO velocity can then be estimated, in a second step, based on the associated semblance values.

Zero-offset traveltime estimation using an RNN

Our choice of Recurrent Neural Networks (RNNs) in NMO velocity analysis is inspired by the foundational work of Biswas et al. (2019). Rather than using a simpler RNN architecture, we employ a variation known as a Gated Recurrent Units (GRU) (Chung et al., 2014), which we refer to generically as our RNN for simplicity. For our application, each input sequence consists of seismic traces extracted from a Common Midpoint (CMP) gather (see an example in Figure 1). We extract a sliding window of dimensions $(N_t$, $N_h)$ from a CMP section. In our examples, we choose

- N_t = 100 (time samples, given a temporal sampling rate of dt = 0.004 s),
- N_h = 61 (traces, starting from 600 m and offset spacing of dh= 20 m).

Before entering the RNN architecture, the input data are normalized.

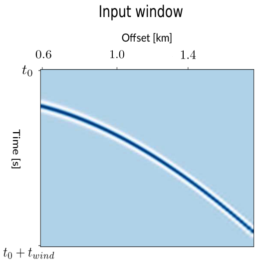

Figure 1: Example input window used in training, with temporal duration t_{wind}, containing a single reflection event.

The RNN architecture follows a many-to-one structure, as illustrated in Figure 2. In this framework, the input data are treated as an ordered sequence. Each column in this window represents an element of the sequential input. The RNN processes these elements while maintaining hidden states that depend on both the current input and previous sequence elements. These hidden states are composed of neurons in $\mathbb{R}^{100}$. The final output is a classification, indicating whether the window's starting time t_0 corresponds to a zero-offset event or not.

For the RNN to provide results at a desirable scale, we used a sigmoid as the final activation function in the classification RNN. This choice is standard for classification tasks as it outputs values between 0 and 1, which can be interpreted as a probability. For our purpose, we then impose a threshold of 0.5 to classify the starting time of the window as a zero-offset event time or not.

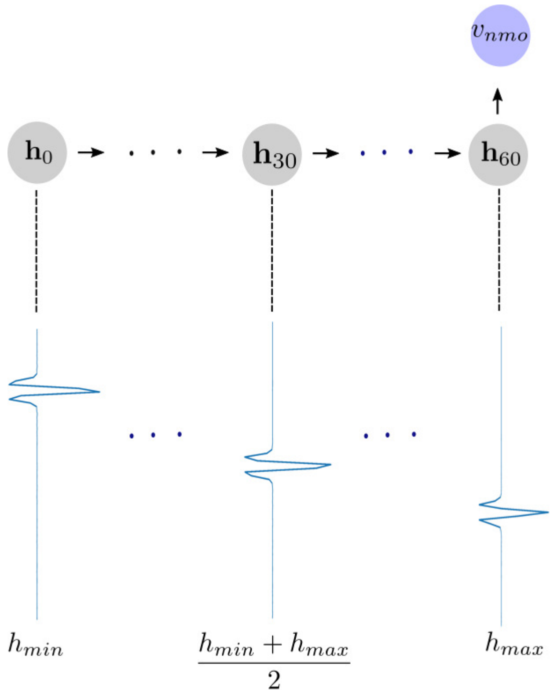

Figure 2: RNN architecture. Output is a classification (0 or 1) that t_0 (associated with the input window) corresponds to the zero-offset traveltime of a primary reflection. If t_0 is classified as an event, the associated v_{nmo} can be estimated in a second step.

Let $W(t_0, m_i)$ denote the selected CMP window with start time t_0 and midpoints m_i, $C(W(t_0, m_i))$ the output of the classification RNN, $\hat{t}_{0,i} \in \{0, 1\}$ expected label. The loss function used is cross entropy, given by

$$\mathcal{L}_c(\Theta) = \frac{1}{N_b} \sum_{i=1}^{N_b} - \log(C(W(t_0, m_i); \Theta_c))\hat{t}_{0,i} \ , \tag{1}$$

where Θ_c represents the classifier network weights and N_b the batch size for stochastic optimization.

We modeled CMP data for training the networks via SU's finite difference modelling, using a Ricker wavelet with peak frequency of $f_p = 10$ Hz. Five different types of constant velocity models with $v = 1.5, 2.0, 2.5, 3, 3.5$ km/s and $N_R = 12$ dipping reflectors were used. One typical models is shown in Figure 3. For all velocity models, velocity in each subsequent layer increases by 0.1 km/s. The training dataset covers all zero-offset times of interest.

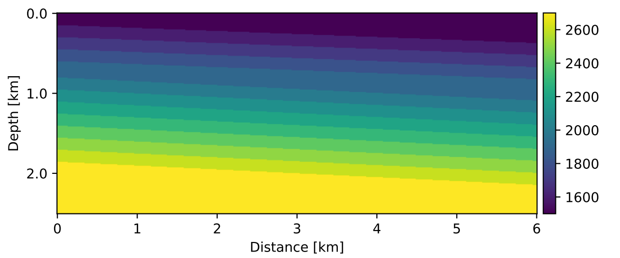

Figure 3: One of the five velocity models used to generate CMP data for training the RNN network. Velocity starts at $v_0 = 1.5$ km/s and increases by 0.1 km/s at each subsequent layer.

Implementation and training of the RNN network were done with Tensorflow API. Network optimization utilized the Adam Optimizer with a batch size of $N_b = 100$ and 879.400 pairs of CMP input slices and their corresponding scalar outputs. We trained the classifier RNN using 15 epochs and learning rate $\alpha = 10^{-4}$.

Results

Primaries only

After training, we tested our RNN on a model with starting velocity $v = 2\ \mathrm{km/s}$ and a constant vertical gradient of $dv/dz = 0.4/\mathrm{s}$ (see Figure 4). Aquisition was done using SU's Kirchhoff modelling in the CMP's $4 - 10\ \mathrm{km}$ with spacing $dm = 0.05\ \mathrm{km}$ betweem each CDP and Ricker wavelet with $f_{peak} = 10\ \mathrm{Hz}$. Therefore, the data contain only primary reflection events. Results of the prediction of the RNN in the central part of the model are shown in Figure 5. We can see that almost all events are correctly predicted. In spite of their weak amplitudes, even the bow-ties from the synclinal structures of the first and second reflectors are partially detected by the network.

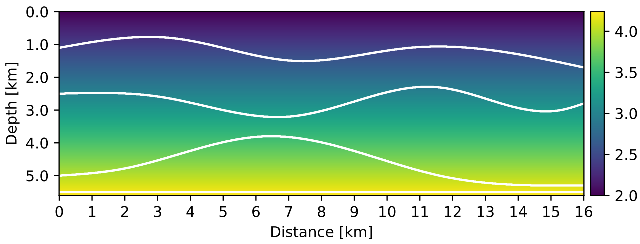

Figure 4: Velocity model of constant vertical gradient, velocity starts at the surface at $v_0 = 2\ \mathrm{km/s}$ and the vertical gradient is $dv/dz = 0.4/\mathrm{s}$. White lines are the artificial reflectors used in Kirchhoff modelling.

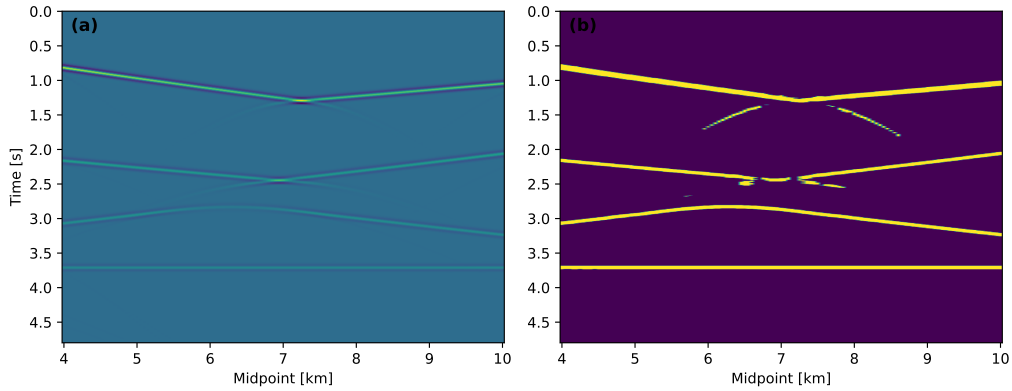

Figure 5: a) Zero-offset section for the gradient model. b) Result of the predicted zero-offset time mask of the RNN.

Full wavefield

To further test the RNN capacity, we used a more realistic velocity model (Figure 6) and SU's Finite difference modelling to generate CMP data containing the full wavefield with all primaries and multiples. The aquisition was done between 1.23 and $7.77\ \mathrm{km}$ with a spacing of $dm = 0.06\ \mathrm{km}$ between CMPs. Prediction of the

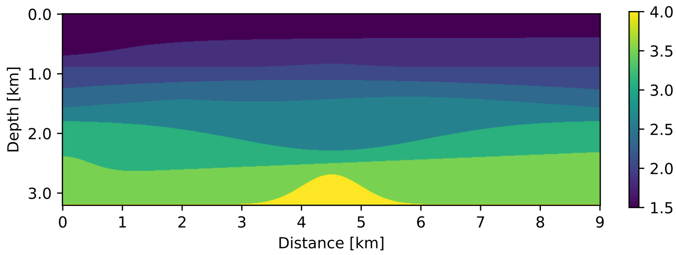

Figure 6: Velocity model used to test the RNN with finite difference data.

RNN is presented in Figure 7. The first impression is acceptable, all events have been detected by the network. However, at some positions failures are visible, possibly due to an imperfect window size. Moreover, the first event is rather weak and followed by a spurious event. It is to be expected that more extensive training with a larger variety of models will further improve these results.

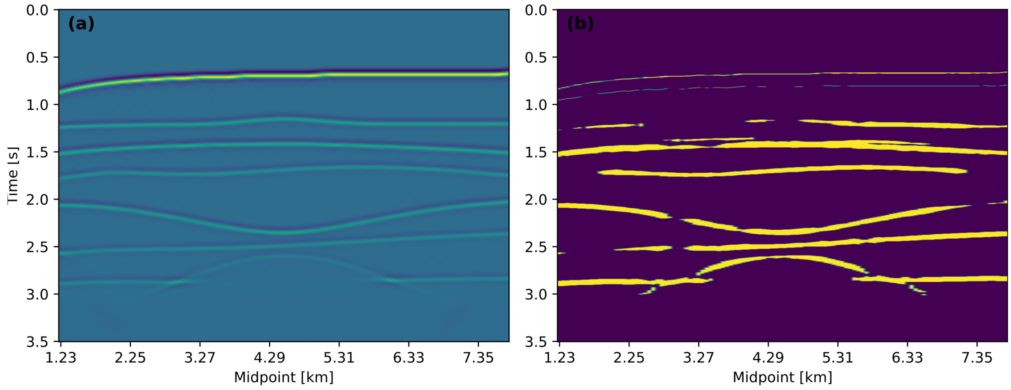

Figure 7: a) Zero-offset section for the velocity model of Figure 6. b) Result of the predicted zero-offset time mask of the RNN.

Acknowledgments

I gratefully acknowledge the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES), Finance Code 001, and Petrobras-CENPES for sponsoring this research.

Conclusions

We have designed a recurrent neural network to determine zero-offset traveltimes from CMP sections. Our first results are promising, demonstrating the potential of the method. To train our network we need only a small number of simples with plane dipping reflectors. Further training with additional, more complex models, should help to improve the network accuracy. Given that convergence of the network happens at just 15 epochs, probably more data can still help.

References

Biswas, R., A. Vassiliou, R. Stromberg, and M. K. Sen, 2019, Estimating normal moveout velocity using the recurrent neural network: Interpretation, **7**, T819–T827.

Chung, J., Ç. Gülçehre, K. Cho, and Y. Bengio, 2014, Empirical evaluation of gated recurrent neural networks on sequence modeling: CoRR, **abs/1412.3555**.

Yilmaz, Ö., 1987, Seismic data processing: Investigations in Geophysics No. 2:SEG.