Using Machine Learning to Understand Uncertainty in Subsurface Exploration

Joshua Bae and Jonathan Sheng

Rice University, Houston TX 77005, USA

1 Background

In the energy industry, it is important to have a solid understanding of subsurface characteristics in order to discover untapped natural resources. Before drilling down to acquire these resources, engineers will conduct a thorough seismic analysis of the area to determine whether it is worth the risk to do so. The process of generating an understanding of the unknown subsurface structures includes the following:

1. Seismic data collection/surveying
2. Seismic data pre-processing
3. Seismic migration & velocity model construction
4. Seismic interpretation

Seismic surveys are typically conducted using powerful sound waves being emitted deep into the earth. These sound waves bounce off boundaries between subsurface layers and are recorded at the surface. During this collection of seismic data, one or more sources of sound energy transmit waves while one or more receivers record the reflections of these waves. The locations of these layers can be identified knowing the source location, receiver location, and measurement of how much time has elapsed between the transmission and reception of the signal. Ideally, one can label subsurface boundary layers based on peaks found in the seismic trace (amplitude over time of the signal received). However, noise can often distort the signal, making peaks much more difficult to identify, so signal-processing techniques are employed to denoise the signal as much as possible.

To evaluate every seismic trace, offset pair gathers are used to reduce the uncertainty. Offset pair gathers are produced by many pairs of sources and receivers that record reflections off the same reflector in the given subsurface. This redundancy helps validate the accuracy of a velocity model. These many pairs collect depth estimates for reflectors and plot them so that depth is depicted on the y-axis and offset pairs along the x-axis. Gathers with mostly horizontal lines indicate an accurate velocity model.

In this paper, we analyze subsurface densities and incorporate different methods to quantify and visualize areas of uncertainty. Given how difficult it can be to identify points of uncertainty as there is no certain way to verify our results, we employed a variety of many different statistical and imaging methods to arrive at our most confident conclusion.

2 Related Work

In determining what methods to use for quantifying and visualizing uncertainty, we drew ideas from medical imaging and image comparison measurements. The Kullback-Leibler Divergence approach was inspired by [1], which explains the uses of K-L Divergence in determining uncertainty within medical images.

Another metric we used, Structural Similarity Index [3], is a common image comparison method that compares the similarity between two entities. SSIM uses luminance, contrast, and structure to check image quality, returning a number between -1 to 1 (complete opposite to exact replica, respectively).

As for the use of canny filters [2], academic papers on computer vision provided insight into how we could use edge detection techniques to focus on horizontal striping present in gathers. The use of this popular edge detection technique primarily comes from image processing in the field of artificial intelligence.

3 Contributions

In this paper, we showcase various statistical and computer vision techniques utilized for analyzing subsurface characteristics and the uncertainty around them. We developed a set of metrics that seek to identify exactly where uncertainty is present and then techniques for visually presenting those metrics overlaid on raw data. Specifically, we delve into seismic uncertainty, seeking to further understand subsurface characteristics.

In particular, we evaluated the following methods/metrics for quanitifying uncertainty in seismic realizations:

1. Standard Deviation, a common method to determine a metric of uncertainty
2. K-L Divergence, a metric to quantify the statistical distance difference between an actual and observed probability distribution
3. Structural Similarity Index, an image comparison technique to determine the similarities and differences between partitions of the realizations
4. Canny Filtering, an edge detection operator used to detect edges in images based on the intensity of the gradients

All code developed as part of this project is made available open source on Github at https://github.com/agrippa/geo-owl-ogy.

4 Methods

In this section, we expand on the dataset and algorithms used in this work.

4.1 Dataset

There were two datasets available for this project: one small dataset (3GB) and one large dataset (49GB). The small dataset is a subset of the files contained

in the large dataset. The datasets consist of a set of models of the subsurface density (called density models, realizations, or stacks). For every realization, the plausible density at a given point is generated based on the full seismic survey.

These 2-D realizations were randomly generated and based on a single known ground truth, and can be used to visualize the structures in the subsurface. These realizations are commonly loaded into a 2-D Numpy array (depth (z) dimension of 400 and a horizontal (x) dimension of 1058, with x as your leading dimension and z as your innermost dimension. Figure 1 shows an example of what a stack looks like when visualized in Python.

For each realization, the dataset also includes a file of gathers produced using the same velocity model. A gather is basically an estimate of the density at a given point for a source-receiver pair in the seismic survey. Every single realization is mapped to one of these offset pair gather files, and each file is conveyed in 3-D form as a Numpy array with 39 offset pairs in the survey. The x and z dimensions are the same as the realizations, with an added y-dimension indicating the offset pair. These gathers basically store values measured at the same physical coordinate in the subsurface using different sources and receivers during a seismic survey. Figure 2 shows an example of what several gathers look like when visualized.

All of these files were stored in industry-standard SEGY format, and a Python module was used to load and visualize them.

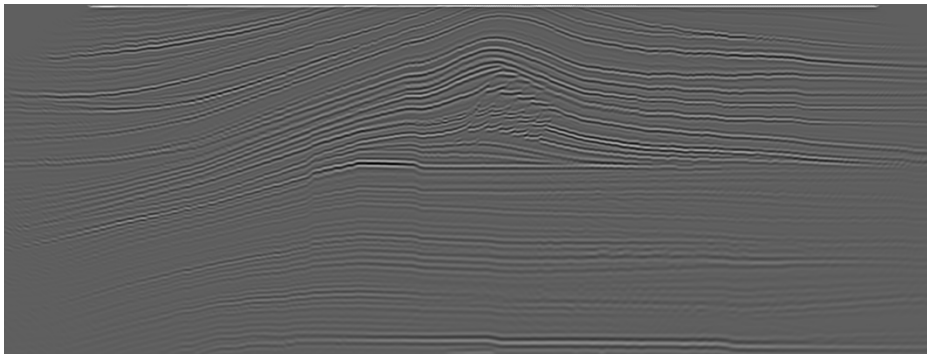

Fig. 1: 2-D realization of subsurface

4.2 Standard Deviation

The goal of our Standard Deviation method was to take the standard deviation across all realizations for each pixel, and use that as a simple uncertainty metric.

First, the standard deviations were calculated through collecting every pixels' values from every realization in the small data set, storing these values in a 1058 $\times$ 400 (dimensions of each realization) $\times$ number of realizations matrix. The

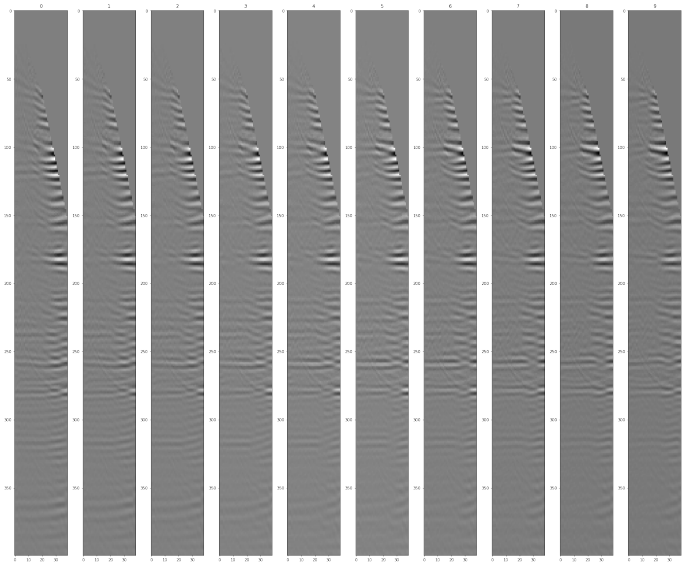

Fig. 2: 3-D gathers based on realization

first iteration over the realizations was to collect the means for each pixel; the second iteration was to apply the standard deviation formula.

After the computation, the standard deviations were visualized across every pixel to see where the points of high variation existed on a sample realization. The points of high standard deviation were displayed as varying shades of red on top of a random realization, where the darker shade represented more uncertainty. This method was conducted on the large data set and the canny-filtered realizations (later discussed).

4.3 Kullback-Leibler Divergence

We also explored the application of K-L Divergence (also called relative entropy) across realizations to highlight areas of general uncertainty by computing the distance between two probabilistic distributions.

For every pixel across all realizations, the set of possible values were first binned into a frequency vector. Then a "true" distribution was computed by finding the most common binned value among the realizations; it was then converted into a probability vector by giving the most common bin a value of 1.0 and all others 0.0. The K-L score for each pixel was then computed by taking the K-L divergence between this "true" distribution and the observed distribution across realizations. We were then able to view an image of the K-L scores superimposed on a sample realization.

To offer some intuition in to this approach, what we are essentially doing is finding the distance between the observed distribution of values and a "true"

distribution of values, where the "true" distribution assumes that the most common value bin is also the correct one. For pixels where the possible values are focused in a single bin, the K-L divergence between our fake "true" distribution and the observed distribution will be small. For pixels where there is a wide range of possible pixel values/bins, the K-L divergence will be high because no one bin will be much more frequent than the others.

4.4 Structural Similarity

The goal of this method was to calculate the structural similarity of a sliding window across every pair of realizations.

At first, we used a sliding window in which there were no overlap, across all pairs. However, this resulted in blocky visuals, not accurately showing the SSIM values at a fine granularity as large blocks of pixels were being assigned a single value (see Figure 3). Thus, we tweaked our approach to instead use overlapping windows to gather more fine grain information and more accurately assign pixels an SSIM value.

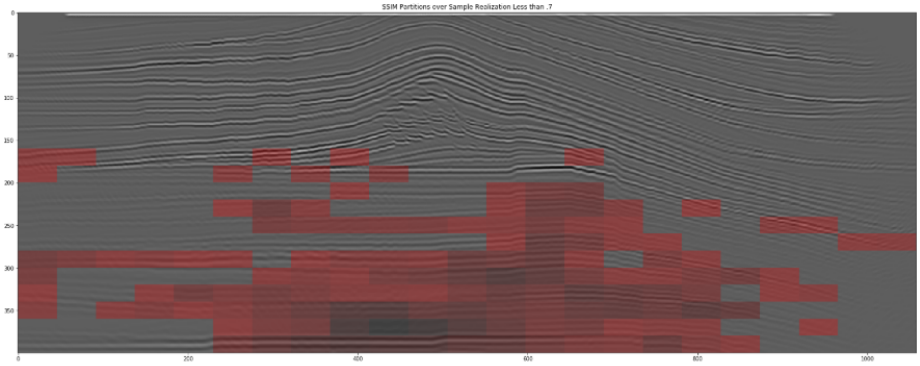

Fig. 3: Areas where mean SSIM values ≤ 0.6 on a sample realization (deeper red, lower SSIM value). Block-like visuals are evident.

When using overlapping windows, we ran into performance issues with a step size of 1. As a result, we decided to reduce the time by a factor of 16 by making the horizontal and vertical step size 4, trading off granularity for speed. With every iteration of the sliding window, we calculated the SSIM value of the window between the two files and added each of those values to its respective pixel. Finally, we took the mean of every pixel's SSIM across all realizations and plotted the result. Due to the nature of overlapping, the corners and edges receive less coverage than elements towards the middle of a realization.

4.5 Canny Filtering/Gather Image Quality

The techniques previously described have focused entirely on the stacks/realizations available in this dataset. However, an entirely separate collection of data is also available to us in the gathers of the dataset. These gathers break down the measured values at each physical coordinate by source-receiver pairs, and so they can offer more fine grain information. From the challenge problem description, we know that "in a gather of an accurate velocity model, geophysicists expect to mostly see horizontal lines, indicating that the depth estimate for a layer is the same across all offset pairs" [4]. Therefore, we explored techniques for finding high quality velocity models by finding horizontal lines in gathers. This can enable us to then focus our techniques on only realizations that are most realistic.

The Canny Filtering method proved to be especially useful. After normalizing the gathers' data from the small data set to ensure good illumination, Canny Filter was applied to the images. A Canny Filter is an edge detection operator used to detect edges in images based on the intensity of their gradients

With the application of the Canny Filter, the Gaussian-smoothed gather images were reduced to just a few white lines edges highlighting the most obvious edges. The smoothing threshold was set for the purpose of noise reduction and to showcase just the most prominent edges. Figure 4 shows an example of what a gather looks like after applying a Canny Filter.

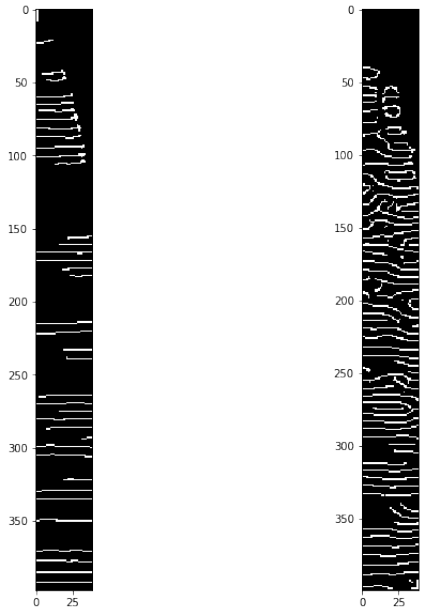

Fig. 4: Example gathers after applying Canny Filters

We could then narrow down the number of "good" realizations based on the horizontal consistency of the gathers after applying the Canny Filter. To do this, the 2-norm (Frobenius norm) was calculated between neighboring columns of each gather and summed up. By this metric, a gather containing the greatest number of similar columns of pixels would be given the lowest scores and qualify as a "good" gather.

The Structural Similarity and K-L Divergence techniques mentioned in the previous sections were then re-applied to a tuned dataset consisting of only "good" realizations.

5 Results

5.1 Standard Deviation

Figure 5 shows a sample result from our Standard Deviation method, applied to the large dataset for this challenge problem. In this visualization, overlay red on top of pixels whose standard deviation exceeds a certain threshold (2.0). Deeper reds indicate higher standard deviations, lighter/brighter reds indicate lower standard deviations. In general, this method appears to be highlighting areas where the layers in the subsurface are more angled – or very deep portions of the volume.

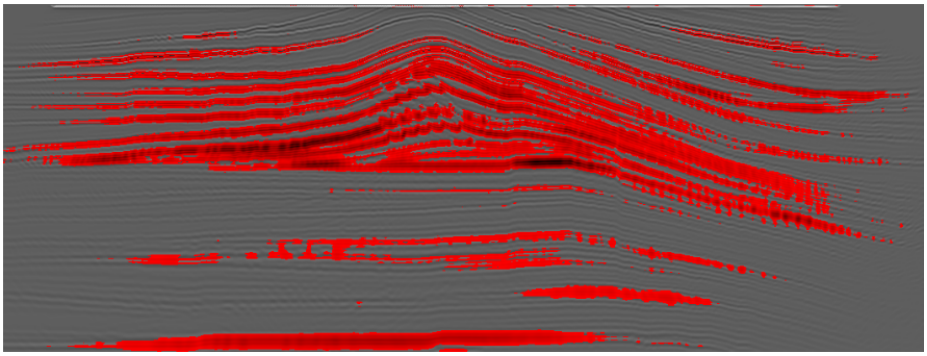

Fig. 5: Areas where standard deviation ≥ 2.0 overlaid on a sample realization.

5.2 K-L Divergence

Figure 6 shows a sample output of our K-L Divergence method applied to the small dataset. It is clear that the uncertainties picked up by this method differ from those highlighted by the standard deviation metric. K-L appears to be picking up on more fine grain uncertainties around the edges of layers in the subsurface as they shift around under different velocity models.

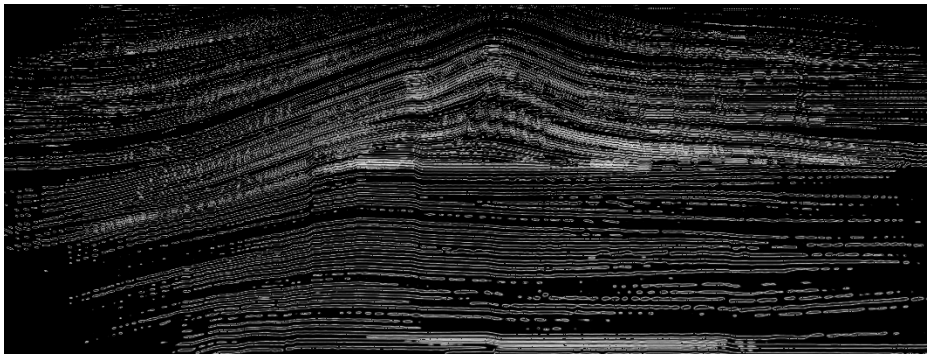

Fig. 6: Superimposed image of the good realizations (small data set) after being processed using K-L Divergence

5.3 Structural Similarity

Figure 7 shows a sample output of the Structural Similarity method. In these images the brighter the red, the higher the mean SSIM values while not crossing above a given threshold (0.6). The deeper the red, the lower the mean SSIM value and therefore the more uncertainty. Again, we find this method seems to be highlighting a different area of uncertainty than the previous two methods – particularly focusing on the deeper regions of the volume.

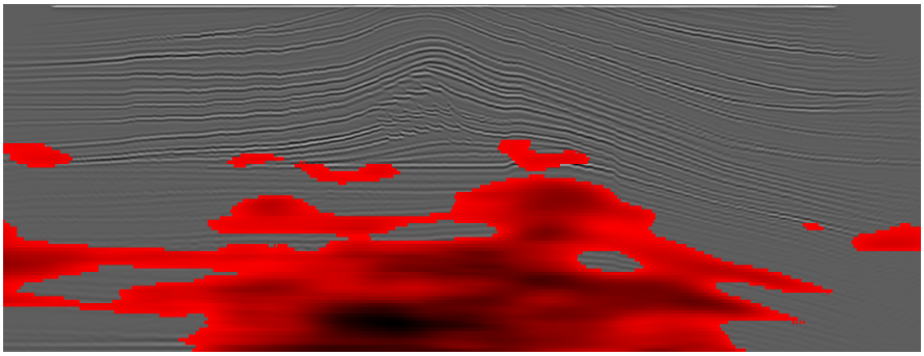

Fig. 7: Areas where mean SSIM values ≤ 0.6 on a sample realization

5.4 Canny-Filtered Dataset

Finally, we also consider how using a filtered and tuned dataset with the three above methods (Standard Deviation, K-L, SSIM) changes the outputs generated by those respective methods on the challenge's small dataset.

Figure 8 shows the distribution of quality scores across the small data computed using the Canny method applied to gathers files. Based on this distribution, we labeled all realizations/gathers in the small dataset with a score ¡= 286,000 as "good". This reduced the size of our dataset from 59 realizations to 29 realizations. Figures 9 and 10 shows an example of a quick spot check to validate that the scores match our intuition about what a "good" and "bad" gather looks like.

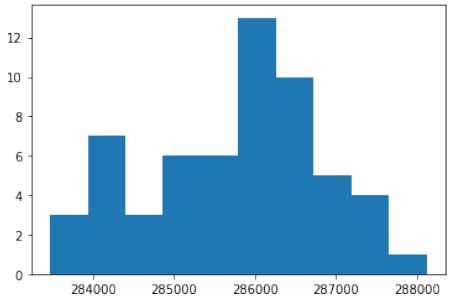

Fig. 8: Histogram visualizing the distributions of realization scores (small dataset) based on gather distances

Figures 11, 12, and 13 show the outputs of our Standard Deviation, K-L Divergence, and SSIM methods applied to the high quality dataset, respectively. We can see that the trimmed dataset has a major impact on the output of the Standard Deviation and SSIM methods, but don't observe much change in the K-L Divergence output.

The Standard Deviation metric is now highlighting less of the image, suggesting that uncertainty has been reduced by focusing on "good" realizations.

The SSIM metric now appears to be mostly highlighting uncertainty in the left side of the volume – the region of highest uncertainty appears to have shifted. At the moment, we do not have an explanation for this change.

6 Discussion & Conclusions

Based on the visualizations delivered by each strategy, each technique seems to be highlighting the image in different ways. Standard deviation indicates broader strokes of uncertainty by coloring in whole regions of the image, while K-L divergence focuses on outlining horizons where the distribution is showing high uncertainty. Meanwhile, the SSIM approach is similar to Standard Deviation in that it is highlighting broad strokes of the image but seems to be focusing on a different region.

When focusing on the realizations that were deemed to be "good" by our simple distance metric, we saw that Standard Deviation and SSIM were significantly

Fig. 9: Example of a gather with a good quality score

impacted in their outputs. In general, both seemed to demonstrate lower uncertainties. In the case of Standard Deviation, the same regions are highlighted but to a lesser extent. In the case of SSIM, its focus seems to have shifted entirely.

Given these results, it is difficult to objectively say that a single metric stands out above the rest to serve as basis for correctly quantifying the uncertainty into a certain area. Since the data is sparse on how accurate density models have been in identifying desirable subsurface characteristics, in addition to not having a ground truth, it is difficult to select a clear winner. Instead, it is likely some combination of these techniques (and potentially other unexplored ones) would be the best option as each is able to highlight different types or regions of uncertainty that might be interesting to seismic interpreters.

References

1. Al-Taie, Ahmad A. Uncertainty Estimation and Visualization in Segmenting Uni- and Multi-modal Medical Imaging Data. Ulm University, Ulm, Germany. 2015.
2. Canny, John. "A computational approach to edge detection." IEEE Transactions on pattern analysis and machine intelligence 6 (1986): 679-698.

Fig. 10: Example of a gather with a poor quality score

3. Wang, Zhou, et al. "Image quality assessment: from error visibility to structural similarity." IEEE transactions on image processing 13.4 (2004): 600-612.
4. Keith Gray, Max Grossman, Anar Yusifov. "Using Machine Learning to Understand Uncertainty in Subsurface Exploration". Smokey Mountain Data Challenge Problem Descriptions. 2020.

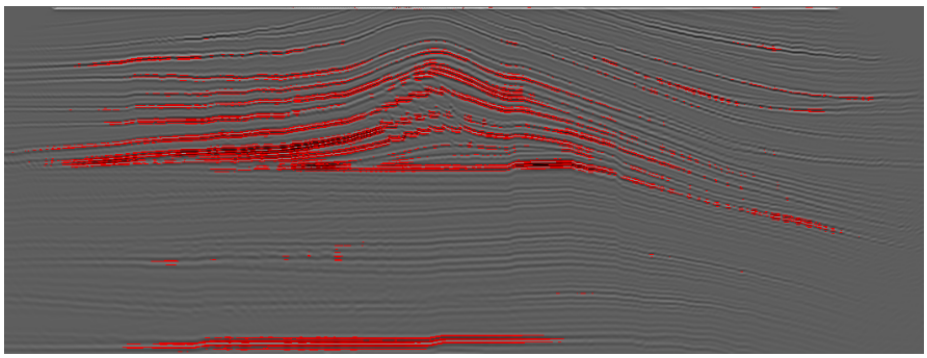

Fig. 11: Areas where standard deviation ≥ 2.0 on a sample realization

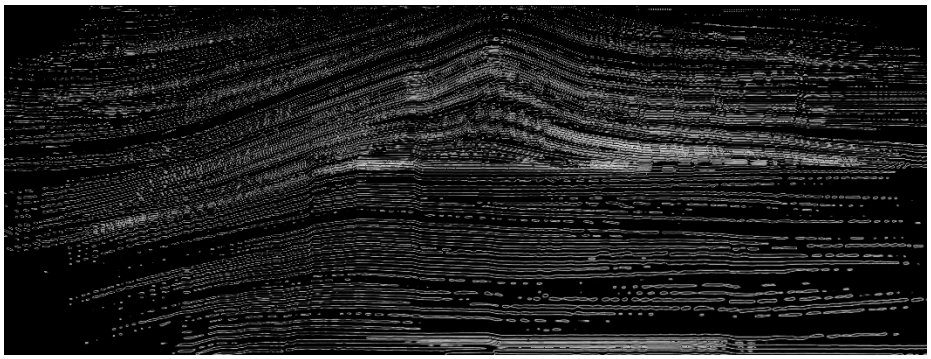

Fig. 12: Superimposed image of all realizations (small data set) after being processed using K-L Divergence

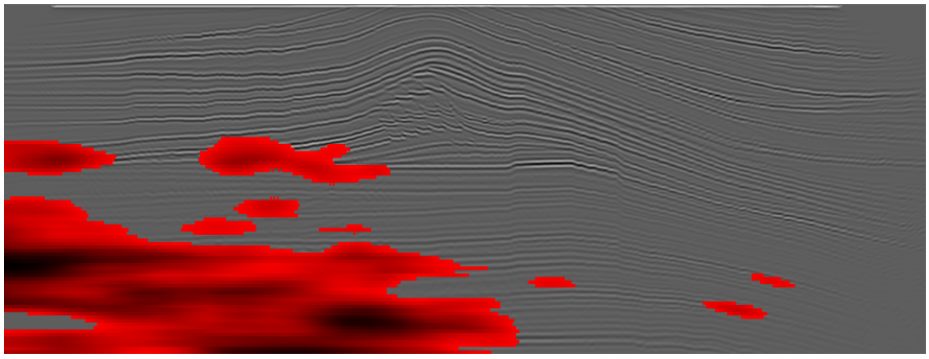

Fig. 13: Areas where mean SSIM values ≤ 0.6 on a sample realization