A NEW IMPLEMENTATION OF CONFLICTING DIPS IN THE COMMON-REFLECTION-SURFACE METHOD USING DIFFERENTIAL EVOLUTION

J. Walda and D. Gajewski

email: *jan.walda@uni-hamburg.de, dirk.gajewski@uni-hamburg.de*
keywords: *diffractions, CRS, multiparameter, optimization, interference*

ABSTRACT

The CRS method enhances the signal-to-noise ratio of stacked data significantly. Furthermore, its wavefront attributes can be used in appealing subsequent processing steps, in particular diffraction imaging and velocity model building. Increasing computational facilities research in recent years showed, that a simultaneous parameter estimation is feasible nowadays and leads to more accurate CRS parameters compared to the pragmatic approach, introduced when CRS was published. This also made alternative methods of detecting intersecting events possible, where events are searched for in different dip cluster. Nevertheless, modern methods still are expensive, in particular for 3D acquisitions since the search space was divided into smaller spaces where CRS has to be performed in each of the smaller spaces. In this paper, we use one application of CRS over the full search space and detect clustering at each iteration based on the dip angle or slowness. The application to 2D synthetic and field data shows promising results. The 3D case indicates issues with the acquisition footprint which needs further investigation.

INTRODUCTION

The first idea to account for the problem of conflicting dips in the common-reflection-surface (CRS) method (Jäger et al., 2001) was proposed by Mann (2001), where he used the angle search step within the pragmatic approach, that estimates the CRS parameters in three individual search steps, to identify several local maxima with a grid search. While this approach is quite efficient, the quality of attributes can be low in challenging environments since very few traces contribute in this step. In recent years, several other search algorithms where proposed to estimate the CRS parameters simultaneously (Garabito et al., 2012; Witte, 2014; Barros et al., 2015; Walda and Gajewski, 2017). Those algorithms are designed to find a good solution in a finite amount of time. This solution mostly coincides with the global maximum. However, those algorithms are designed to find one maximum. In conflicting dip situations, we need to find all contributing maxima, which are as many as there are conflicting events. Hence, we need to track passed by local maxima during the search. Current approaches simply estimate CRS parameters multiple times, either for a fixed dip, as in the CDS approach (Soleimani et al., 2009) or dip ranges (Walda and Gajewski, 2017). This is a stable approach but also very expensive, particularly in the 3D or common-offset case. Furthermore, events are sometimes stacked multiple times with partly wrong attributes, which leads to smearing in the stack and inaccurate results in subsequent wavefront attribute-based processing such as wavefront tomography (Duveneck, 2004; Bauer et al., 2017). We propose to use a property of population-based search algorithms, the existence of many starting solutions. While the population of the algorithm converges, those algorithms tend to partly cluster in local maxima on their way to the global maximum. We aim to track those local maxima discovered during the search to account for conflicting dips with just one search. In order to do so, we need to identify if, where and when several population member are close

to each other. We have to quantify close in the search space and account for varying sensitivity of the CRS parameters. We use the first order derivative CRS parameter, which is the slowness, to identify clustering of population members. In 3D an additional problem is the acquisition footprint which can lead to well determined CRS attributes in one direction while the attributes in the perpendicular direction might be of poor quality, e.g. due to a narrow azimuth acquisition. We therefore might need to neglect clustering in one direction for such datasets.

MULTIPARAMETER STACKING

The CMP method stacks data along a traveltime curve in offset direction

$$S(t_0, m_0) = \iint P(t(t_0, h), m_0, h)dh \quad . \tag{1}$$

This method uses only data redundancy in offset direction which becomes problematic if only few traces are available. Challenging data, where CMP processing does not produce desired results, benefit from stacking techniques where the summation is carried out in offset and midpoint direction

$$S(t_0, m_0) = \iint P(t(t_0, \Delta m, h), m, h)dmdh \quad . \tag{2}$$

In Equation 2, the quantity $\Delta m = m_0 - m$ is the midpoint displacement. Since neighboring midpoints m are stacked into the midpoint m_0 as well, a higher amount of traces are summed which results in an increased signal-to-noise ratio. However, the traveltime description depends on more parameters compared to classical CMP stacking. Hubral (1983) introduced wavefront attributes that can be used as a parametrization to describe the required traveltime surface $t(t_0, \Delta m, h)$. Traveltime operators that use these attributes can be classified as common-reflection-surface (CRS) type operators. They differ in their mathematical structure which stem from the underlying assumptions. While CRS itself is a hyperbolic single-square-root traveltime expression, there are other double-square-root operators. In the following, the classical CRS and an efficient non-hyperbolic variant are described.

Common-reflection-surface

The CRS stack is a multiparameter stacking technique (Müller, 1999; Mann et al., 1999; Jäger et al., 2001; Mann, 2002) that considers neighboring midpoints as well as the offset while the CMP method uses only offsets. It describes an event in the vicinity of the zero-offset sample by a second-order traveltime approximation. Since more traces are stacked, the signal-to-noise ratio is improved significantly.

In 2D the CRS operator is formulated in terms of three wavefront attributes, which are related to two hypothetical one-way experiments. The resulting two waves are described by the angle of emergence α of the zero-offset ray and the corresponding radii of curvature: R_N for the normal (N) wave and R_{NIP} for the normal-incidence-point (NIP) wave (Hubral, 1983). The N wave is generated by a fictitious exploding reflector experiment around the normal-incidence-point. The NIP wave is generated by a fictitious point source placed at the normal-incidence-point.

Using the notation by Fomel and Kazinnik (2013), the CRS formula in its hyperbolic expression can be written as

$$t(t_0, \Delta m, h) = \sqrt{F(\Delta m, t_0) + 2t_0 M h^2} \quad , \tag{3}$$

$$F(t_0, \Delta m) = (t_0 + 2p\Delta m)^2 + 2t_0 N \Delta m^2 \quad . \tag{4}$$

The coefficients p, M and N can be related to the wavefront attributes α, R_{NIP} and R_N as

$$\begin{aligned} p &= \frac{\sin\alpha}{v_0} \\ N &= \frac{\cos^2\alpha}{v_0 R_N} \\ M &= \frac{\cos^2\alpha}{v_0 R_{NIP}} \quad . \end{aligned} \tag{5}$$

The quantity v_0 is the near surface velocity. In the 3D case the curvatures R_N and R_{NIP} become symmetric 2x2 matrices and the slowness p becomes a vector with two elements. Therefore, the 3D CRS operator depends on 8 attributes and reads

$$t(t_0, \Delta\boldsymbol{m}, \boldsymbol{h}) = \sqrt{F(\Delta\boldsymbol{m}, t_0) + 2t_0 \Delta\boldsymbol{h}^T \boldsymbol{M}\boldsymbol{h}} \quad , \tag{6}$$

$$F(t_0, \Delta\boldsymbol{m}) = (t_0 + 2\boldsymbol{p}^T \Delta\boldsymbol{m})^2 + 2t_0 \Delta\boldsymbol{m}^T \boldsymbol{N} \Delta\boldsymbol{m} \quad . \tag{7}$$

Non-hyperbolic common-reflection-surface

Fomel and Kazinnik (2013) introduced an extension of the conventional CRS approach. The non-hyperbolic common-reflection-surface (nCRS) method assumes a hyperbolic reflector and reads in 2D

$$t(t_0, \Delta m, h) = \sqrt{\frac{F(t_0, \Delta m) + Ch^2 + \sqrt{F(t_0, \Delta m - h) F(t_0, \Delta m + h)}}{2}} \quad , \tag{8}$$

where F is defined according to Equation 4 and $C = 2t_0(2M - N) + 4p^2$ accounts for the asymmetry of the source-receiver traveltime contributions. From an implementation point of view, nCRS requires very few additional computations compared to CRS (see Equation 3) which simplifies the implementation in existing CRS codes. The 3D extension of nCRS is according to CRS and reads but similar to CMP processing

$$t(t_0, \Delta\boldsymbol{m}, \boldsymbol{h}) = \sqrt{\frac{F(t_0, \Delta\boldsymbol{m}) + \boldsymbol{h}^T \boldsymbol{C}\boldsymbol{h} + \sqrt{F(t_0, \Delta\boldsymbol{m} - \boldsymbol{h}) F(t_0, \Delta\boldsymbol{m} + \boldsymbol{h})}}{2}} \quad , \tag{9}$$

where $\boldsymbol{C} = 2t_0(2\boldsymbol{M} - \boldsymbol{N}) + 4\boldsymbol{p}^T\boldsymbol{p}$.

METHODOLOGY

In our method, we assume the CRS parameters are estimated by a population-based search algorithm. In literature, there are plenty of such algorithms around, especially evolutionary-based algorithms. For demonstration purpose, we use the differential evolution (DE) algorithm here since it provided satisfying results in the past.

Differential Evolution

The DE algorithm starts with a number of initial solutions referred to as starting population. In most cases, they are randomly distributed. It is also possible to provide previously-determined solutions into the starting population, for example obtained CRS attributes of neighboring samples or velocity estimations. In the next step, information obtained from the initial solutions is used to generate a new set of trial solutions. Each solution in DE is called agent. Each iteration, which generates new potential agents, is called a generation. The process of creating new generations is repeated until a satisfying solution is found. In DE, for each agent $\boldsymbol{x}_i$ a trial agent $\boldsymbol{u}_i$ is generated that is only accepted into the next generation if its fitness value (i.e., semblance functional) is better than the original agent. The trial agent is generated by so called mutation and crossover, where the index i denotes the agent in the population. The mutation $\boldsymbol{y}_i$ of an agent $\boldsymbol{x}_i$ is carried out by the formula

$$y_{ij} = x_{aj} + F \cdot (x_{bj} - x_{cj}) \tag{10}$$

where $\boldsymbol{x}_a$, $\boldsymbol{x}_b$ and $\boldsymbol{x}_c$ are randomly selected agents of the population that are different from each other and $\boldsymbol{x}_i$. The subscripts a, b and c are determined randomly for each agent i. Variants of the DE algorithm use the agent with the highest fitness instead of $\boldsymbol{x}_a$. This can lead to a faster convergence but risks premature convergence. The index j denotes one dimension of the problem, here the number of CRS wavefront parameters, and $F \in [0, 2]$ is called the differential weight. The mutation of a dimension is carried out with a probability determined by the crossover parameter CR. The size of the population is determined by a control parameter called NP. These parameters are control parameters of the DE

algorithm and have to be chosen upfront. (Pedersen, 2010) suggest certain choices of parameters based on the dimensionality and desired number of function evaluations. Alternatively, it is also possible to randomly select the control parameter F within a range at each iteration. This adds diversity and usually leads to better coverage of the search space.

Since the CRS method in 3D has an 8D search space, we tested different combinations of control parameters for the 8D Rosenbrock function, which is given by the formula

$$f(x) = \sum_{j=1}^{n-1} \left[100(x_{j+1} - x_j^2)^2 + (x_j - 1)^2\right] \tag{11}$$

where j is the index over dimensions of the search space and n the total number of dimensions. The function minimum for $n > 3$ is $f(1, ..., 1) = 0$. Figure 1 shows the convergence of the best agent at each iteration for 200 realizations. In these tests F changes randomly at each iteration within 0.5 and 1.0. The actual size of the population here is $NP \times n$, where n is the number of dimensions. A low crossover probability of 50% leads to no sufficient convergence. In 99.5% of the cases, the algorithm converged prematurely. A crossover probability of 70% results in the least amounts of premature convergence for 'smaller' populations. However, the convergence rate is also rather slow compared to a crossover probability of 90%. A larger population size on the other hand does not improve the convergence rate. It mostly prevents premature convergence. However, an increase from a total of 96 to 180 agents lead only to a total decrease of premature convergence of 2% for a crossover probability of 70% and 6% for a crossover probability of 90%. Hence, as long as the population is not too small, a high crossover probability of 90% should be chosen to converge efficiently.

So far we used random initialization, which can lead to uncovered areas in the search space as well as unintended initial clustering. An alternative initialization can be done using the latin hypercube sampling (McKay et al., 1979) which splits the search space into segments and seeds one staring point into each segment. Usually, it leads to a better coverage of the search space and prevents clustering as shown in Figure 2 for a 3D search space.

Clustering

Figure 3 illustrates a slice of the 3D search space of the 2D CRS operator for a sample, where two events are present. The most spread red maximum corresponds to the global maximum. The considered sample is extracted from the Sigsbee2A model shown in the following section. The two maxima represent two intersecting diffraction tails originated from two artificially placed diffractions in the deeper part of the model.

In order to account for conflicting dips during the stacking and further CRS-based processing steps such as wavefront tomography (Duveneck, 2004; Bauer et al., 2017), we need to find both maxima. Unfortunately, search algorithms are designed to find, either the global maximum, or the nearest local maximum. Previously, the search space was divided into sub-cubes in which a search was performed. This means for N amount of cubes, N full CRS searches had to be performed. Fortunately, a search algorithm like DE covers large parts of the search space and should have a decent chance to pass the local maxima during the search as seen in Figure 3. Furthermore, the search population tends to cluster at places where good solutions are found until better solutions become available. This means, if we keep areas of clustering in memory, we can assume there is a local maximum. After the search is finished, we might search those local maxima with a fast converging algorithm which runs into the next local maximum it finds.

So far, we are not aware of a method to identify clustering in the search space for the general case, particularly for multidimensional problems. The 3D CRS formula uses eight attributes of different ranges and sensitivities with respect to the influence on the traveltime surface and the acquisition. Therefore, we only use the most stable parameters, in 2D the angle of emergence α, in 3D the two components of the horizontal slowness p_x and p_y. A cluster is considered when m amount of agents are within an user-defined angle/slowness range. The number of required agents to cluster m is also specified by the user. A smaller angle range enables more potential clusters in case of intersecting events with similar dips. A

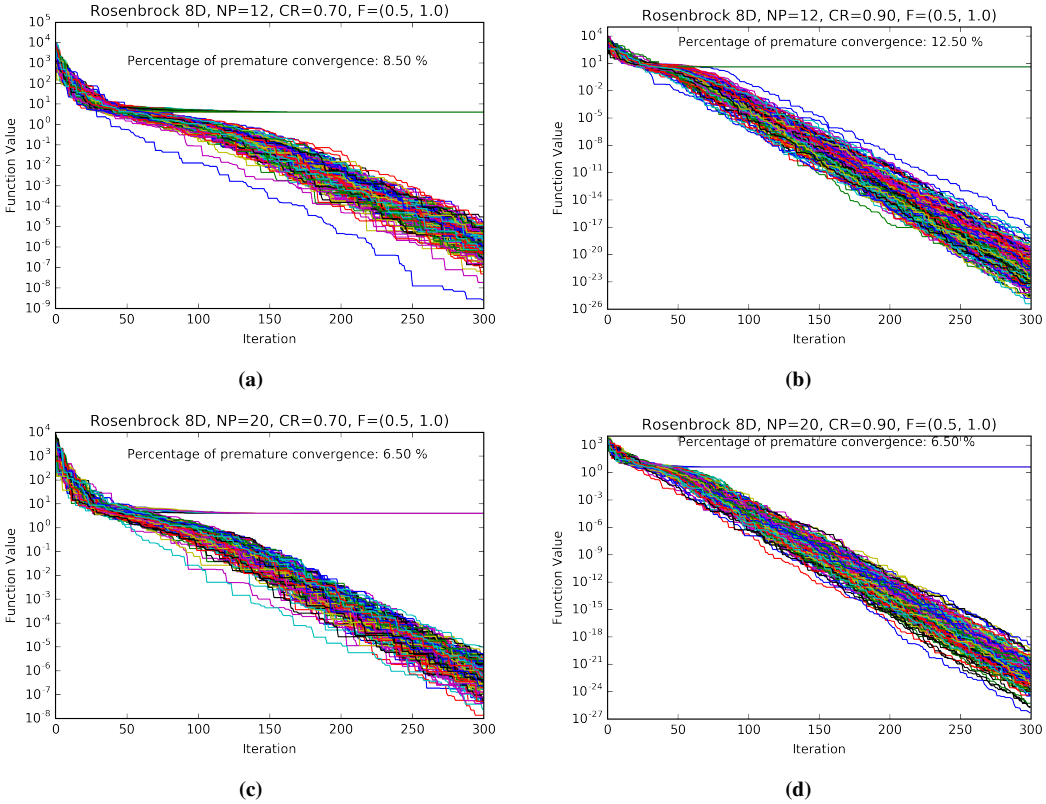


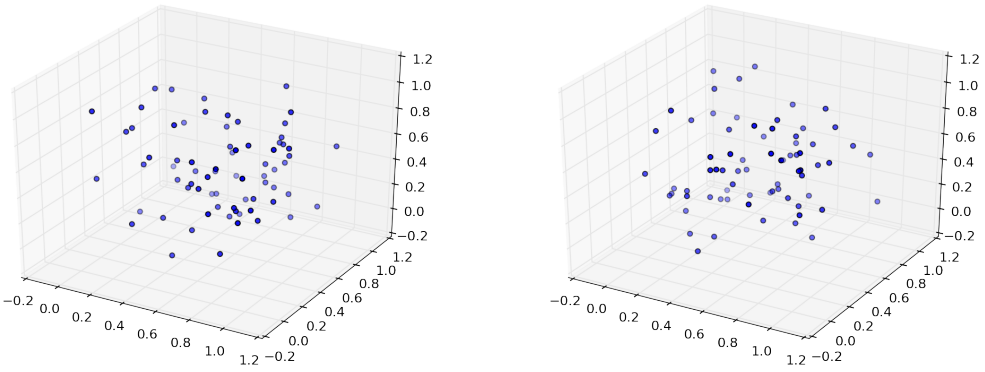
Figure 1: Convergence comparison of different control parameter choices. Please note that the vertical logarithmic scales vary and the actual population size is $NP \cdot 8$ due to latin hypercube sampling.

larger angle range might miss intersecting events but also reduces the risk of falsely detected clustering. In a similar fashion, a high amount of clustered agents m will only find pronounced local maxima while a small value of m leads to found local maxima that might not be actual events. To prevent clustering noise the user can provide a minimal semblance which is required to consider a clustered event as a local maximum. Furthermore, the acquisition in 3D can lead to well determined CRS parameters in one direction and poorly determined CRS parameters in the perpendicular direction. This is the case for narrow azimuth data. In such cases it is advisable to cluster considering only either p_x or p_y.

The objective function shown in Figure 3 shows that clustering also occurs in the R_{NIP} (or $\boldsymbol{M}$ in 3D) dimension. However, this parameter depends on the velocities and depth of the considered sample. Therefore, it is challenging to formulate a stable clustering criterion. Since machine learning became popular in recent years and are particularly well suited to identify clustering in classification problems, a well trained neural network might help to identify clustering during the optimization much more reliably.

SYNTHETIC DATA EXAMPLE

To evaluate whether our new approach is suitable to handle conflicting dip situations, we use the synthetic Sisgbee2A dataset from SMAART JV (Subsalt Multiples Attenuation And Reduction Team Joint Venture consisting of BHP Billiton, BP and ChevronTexaco). The model is shown in Figure 4. It contains a prominent salt body in white as well as layering with some fault structures and artificially placed diffractors. Due to all these features, the model produces tremendous amounts of scattering causing many conflicting dip situations. In the previous implementation we simply divided the search space into smaller cubes along the dip angle α and searched for the CRS attributes in each dip range. Depending on the width of these dip ranges, one can reliably detect almost all apparent conflicting dip situations. However, the approach is computationally very demanding and introduces some artifacts as well as smearing, if one maximum



(a) Random initialization

(b) Latin hypercube

Figure 2: Comparison of a random initialization (a) and an initialization using latin hypercube sampling. In the latin hypercube sampling model the population is better spread over the search space while clustering is prevented as it can appear in the random initialization.

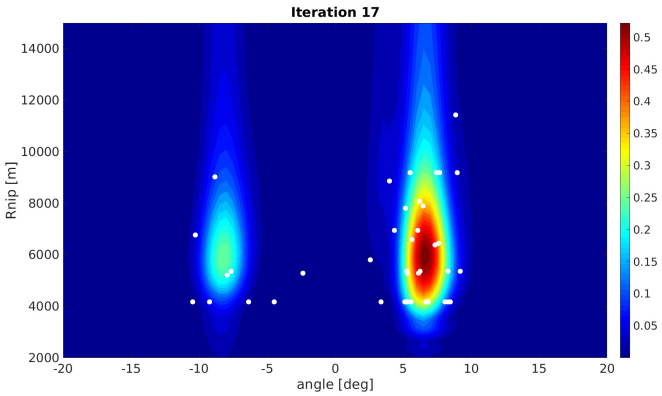


Figure 3: 2D slice of the 3D search space during the parameter estimation using the 2D nCRS operator. Two maxima can be observed that differ in their dip angle. White dots represent the location of an agent at the 17th iteration. Most of the agents are already located in the global maximum while some are located in the local maximum. The colorbar represents the semblance

stretches over several dip ranges. This is mostly the case for prominent reflectors of particularly strong impedance contrast like top and bottom of salt or sea floor.

Figure 5(a) shows a CRS stack of the Sigsbee2A model where even diffractions caused by stair-stepping effects during the modeling, are recovered. In this case, 9 dip intervals were used which increased in size at higher dips. For comparison, Figure 5(b) shows the proposed method to account for conflicting dips. Though a single application of CRS is roughly 50%-100% more expansive as in the previous case, it is still much cheaper than before since the CRS method only has to be performed once here. The additional computational cost mostly stems from a larger required population size which was 48 in this example. The new implementation does not detect as many intersecting events as the former implementation but is more efficient. Most events that are not resolved are rather weak. On the other hand, events are less smeared and the amplitudes are not artificially increased as previously. This is of particular importance for amplitude friendly processing. Since the synthetic example showed promising results, we apply the new approach on a marine field data example.

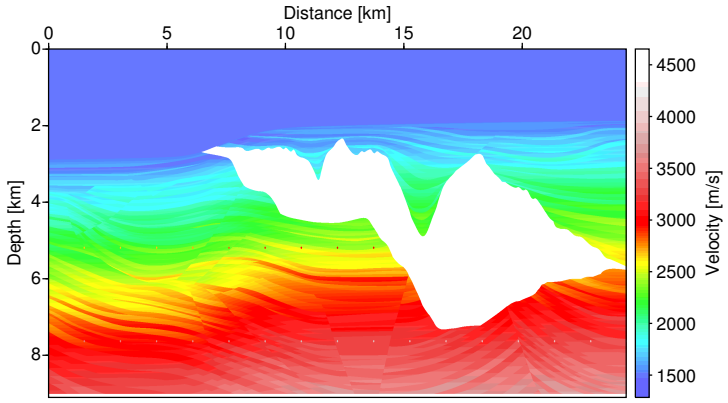


Figure 4: Velocity model of the Sigsbee2A data, which is based on salt structures in the Gulf of Mexico. The model contains a prominent salt body in white and layered structures including faults and artificially placed diffractors around it.

MARINE DATA EXAMPLE

The application of our method to synthetic data showed satisfying results. In order to estimate the applicability to field data we apply our method to a marine data set from the Levantine Basin in the Mediterranean Sea, offshore Israel (Netzeband et al., 2006). A comparison of our approach to the previous implementation is shown in Figure 6. To ensure we get as many conflicting events as possible in our previous implementation (Figure 6(a)), we used 17 dip intervals here which made the processing very expensive but reliable. As a result, we recovered the rich content of diffractions in this data which enables us to perform diffraction imaging. The disadvantage of this approach shows in the sea floor and salt body reflections which are smeared since those event represent wide maxima in the search space which are stacked in several dip cluster with partly wrong attributes. In our new implementation shown in Figure 6(b) we are able to recover most events, apart from very weak ones. Comparing both implementations, it seems that the results of the new implementation shows the found events, particularly diffractions, much more pronounced. The most important advantage however, apart from the efficiency, is the much higher resolution which is evident in the layered structures since we do not stack them multiple times with wrong attributes.

THE 3D CASE

In the previous parts of the paper we showed that our approach is feasible in 2D. The 3D case however, introduces more challenges. We have to consider two dips, in X and Y direction. Additionally, more often than not, the acquisition is not distributed similarly in X and Y direction. In the marine case, most of the time one ship is carrying a source and above 10 streamer that are above 10 km long with a dense hydrophone group spacing, e.g. every 12.5 m or 25 m. The streamer spreading however, often is 50 m to 200 m. This means in sail-direction (inline) the acquisition is longer and denser than in the perpendicular (crossline) direction. This so-called acquisition footprint needs to be considered as well. Finally, the higher amount of involved traces and especially CRS parameters, hamper the search itself and raise the computational cost. Splitting the search space in intervals for both directions rapidly leads to the curse of dimensionality (Corporation, 1957). In case of 5 intervals per direction, we would need to estimate 25 times the 3D CRS operator.

Fortunately, our new approach only needs the CRS parameter estimation once. However, we also need to perform clustering based on the acquisition. In narrow azimuth data, we simply cluster along one component of the slowness. If more azimuths are available, we use both directions. A preliminary result is shown in Figure 7. In this example, no local refinement of the maxima was used while the clustering criteria are quite narrow. This means the algorithm found more clusters compared to the 2D case which are mostly meaningless and only contribute noise. Nevertheless, we can observe intersecting events at numerous places, indicating that the principle idea can work when the method is improved, in particular

the local refinement and treatment of acquisition footprint.

CONCLUSIONS

We introduce a new and efficient method to account for conflicting dips in the CRS method. If several events are intersecting at a sample, weaker events will appear as local maxima in the search space that need to be found additionally. We assume a population-based global optimization method to be used during the parameter estimation. During the parameter estimation a local maximum is considered each time several population members accumulate in a small region. In case this region has been recognized before, the local maximum is updated. This enables us to track local maxima the algorithm has seen during the global optimization. All found local maxima are stacked, together with the global maximum, to provide a stacked section where intersecting events are resolved. We verify the method on 2D synthetic and field data. Most events are efficiently recognized. Furthermore, the frequency content is not harmed as in previous implementations. In the 3D case the acquisition needs to be considered. For narrow azimuth data, the slowness in inline direction is much better determined than in crossline direction. Hence, for such data, only the slowness in inline direction should be considered.

ACKNOWLEDGMENTS

The work was supported by the sponsors of the Wave Inversion Technology (WIT) consortium and the Federal Ministry for Economic Affairs and Energy of Germany (project number 03SX427B). We would like to thank the Applied Seismics Group Hamburg for continuous discussions. Furthermore, we thank the SEG for providing the 3D synthetic dataset, SMAARTJV for the Sigsbee2A model and TGS for the marine data. Finally, we thank TEEC, in particular Marco Diers, Gerald Eisenberg-Klein and Jürgen Prüßmann for fruitful discussions and suggestions.

REFERENCES

Barros, T., Ferrari, R., Krummenauer, R., and Lopes, R. (2015). Differential evolution-based optimization procedure for automatic estimation of the common-reflection surface traveltime parameters. *Geophysics*, 80(6):WD189–WD200.

Bauer, A., Schwarz, B., and Gajewski, D. (2017). Utilizing diffractions in wavefront tomography. *Geophysics*, 82(2):R65–R73.

Corporation, R. E. B. R. (1957). *Dynamic programming*, volume 1. Princeton University Press.

Duveneck, E. (2004). Velocity model estimation with data-derived wavefront attributes. *Geophysics*, 69(1):265–274.

Fomel, S. and Kazinnik, R. (2013). Non-hyperbolic common reflection surface. *Geophysical Prospecting*, 61(1):21–27.

Garabito, G., Stoffa, P. L., Lucena, L. S., and Cruz, J. C. (2012). Part i âĂŤ crs stack: Global optimization of the 2d crs-attributes. *Journal of Applied Geophysics*, 85(Supplement C):92 – 101.

Hubral, P. (1983). Computing true amplitude reflections in a laterally inhomogeneous earth. *Geophysics*, 48:1051–1062.

Jäger, R., Mann, J., Höcht, G., and Hubral, P. (2001). Common-reflection-surface stack: Image and attributes. *Geophysics*, 66:97–109.

Mann, J. (2001). Common-Reflection-Surface Stack and conflicting dips. *71st Annual Internat. Mtg., Soc. Expl. Geophys*, Extended Abstracts:Session SP 5.3.

Mann, J. (2002). *Extensions and Applications of the Common-Reflection-Surface Stack Method*. PhD thesis, Logos Verlag, Berlin.

Mann, J., Jäger, R., Müller, T., Höcht, G., and Hubral, P. (1999). Common-reflection-surface stack - a real data example. *Journal of Applied Geophysics*, 42:301–318.

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21(2):239–245.

Müller, T. (1999). *The Common Reflection Surface Stack Method: Seismic Imaging without explicit knowledge of the velocity model*. PhD thesis, University of Karlsruhe.

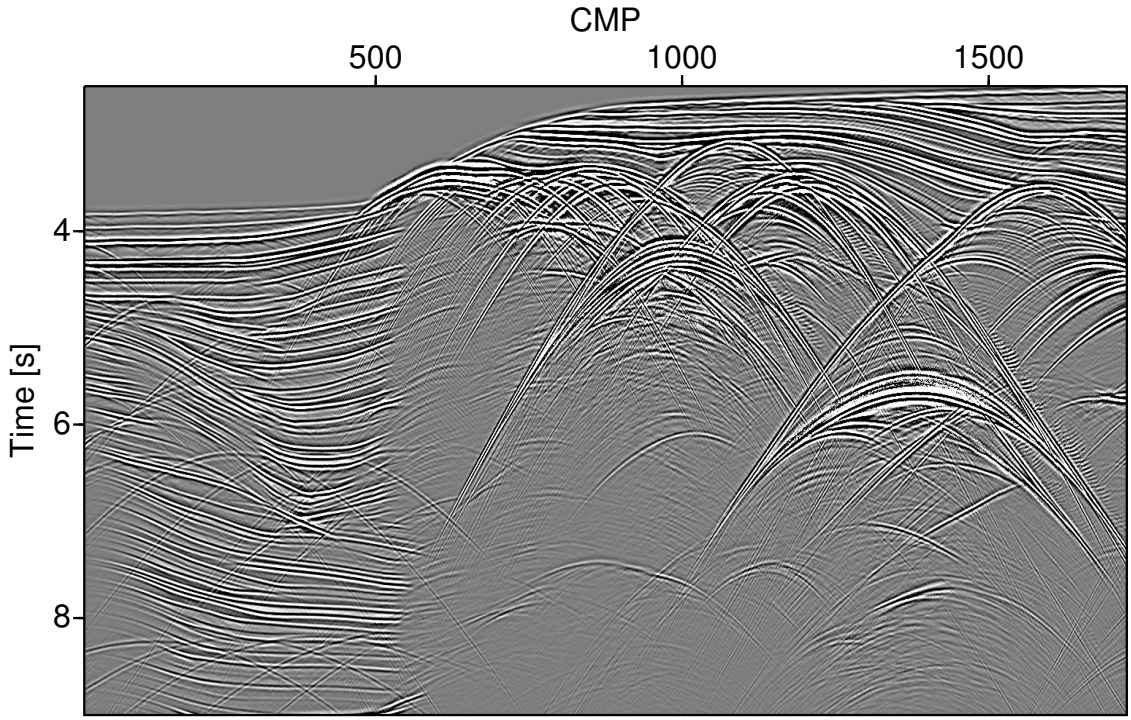
Netzeband, G. L., Gohl, K., Hübscher, C. P., Ben-Avraham, Z., Dehgani, G. A., Gajewski, D., and Liersch, P. (2006). The Levantine Basin-crustal structure and origin. *Tectonophysics*, 418:167–188.

Pedersen, M. E. H. (2010). Good parameters for differential evolution. Technical Report HL1002, Hvass Laboratories.

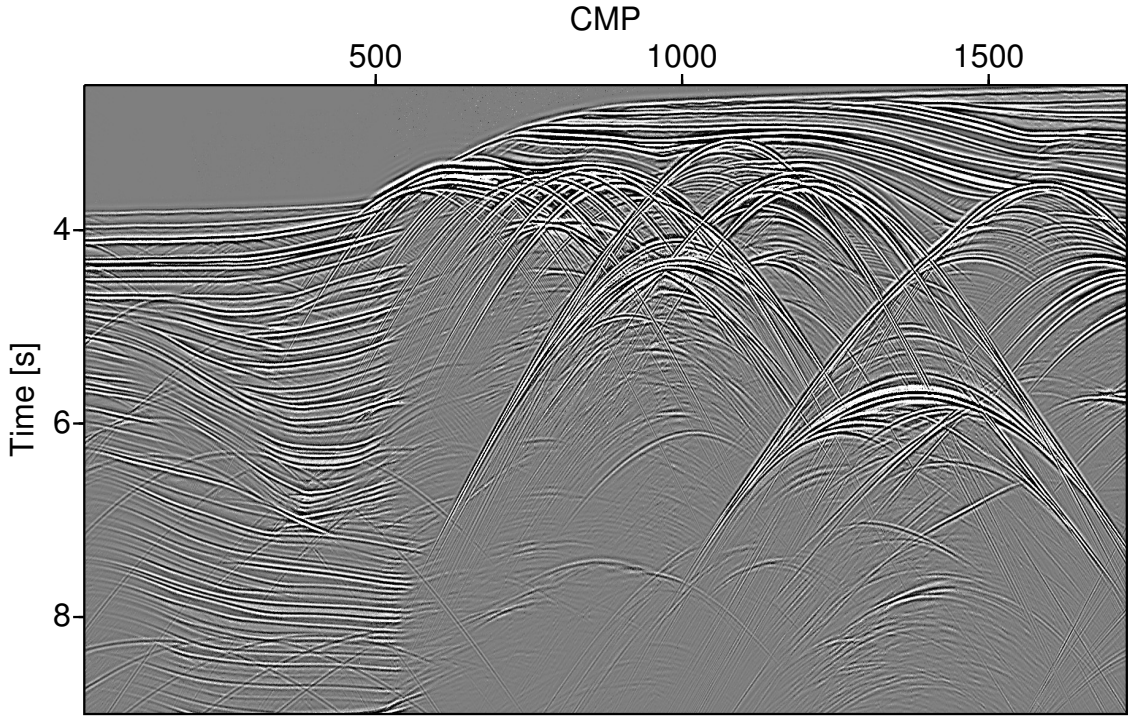
Soleimani, M., Piruz, I., Mann, J., and Hubral, P. (2009). Solving the problem of conflicting dips in common reflection surface (CRS) stack. *1st International Petroleum Conference & Exhibition*, EAGE, expanded abstracts, A39.

Walda, J. and Gajewski, D. (2017). Determination of wavefront attributes by differential evolution in the presence of conflicting dips. *Geophysics*, 82(4):V229–V239.

Witte, P. (2014). Optimization of the Semblance in Seismic Processing. Master's thesis, University of Hamburg.

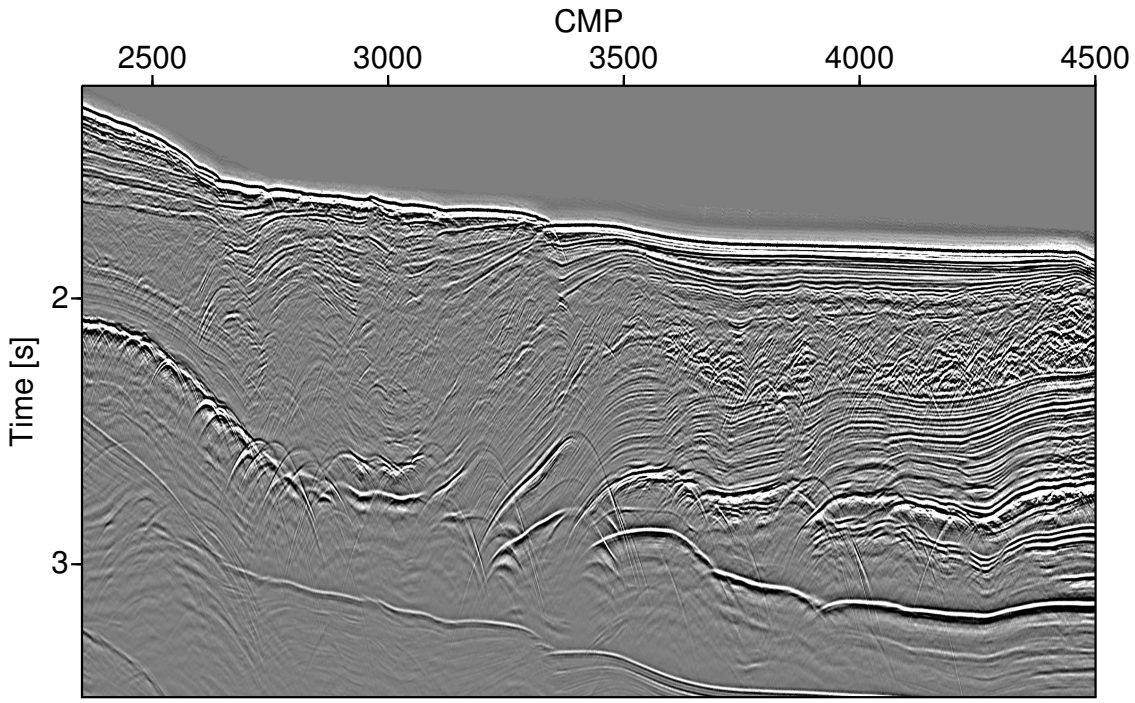


(a) nCRS stack of former implementation

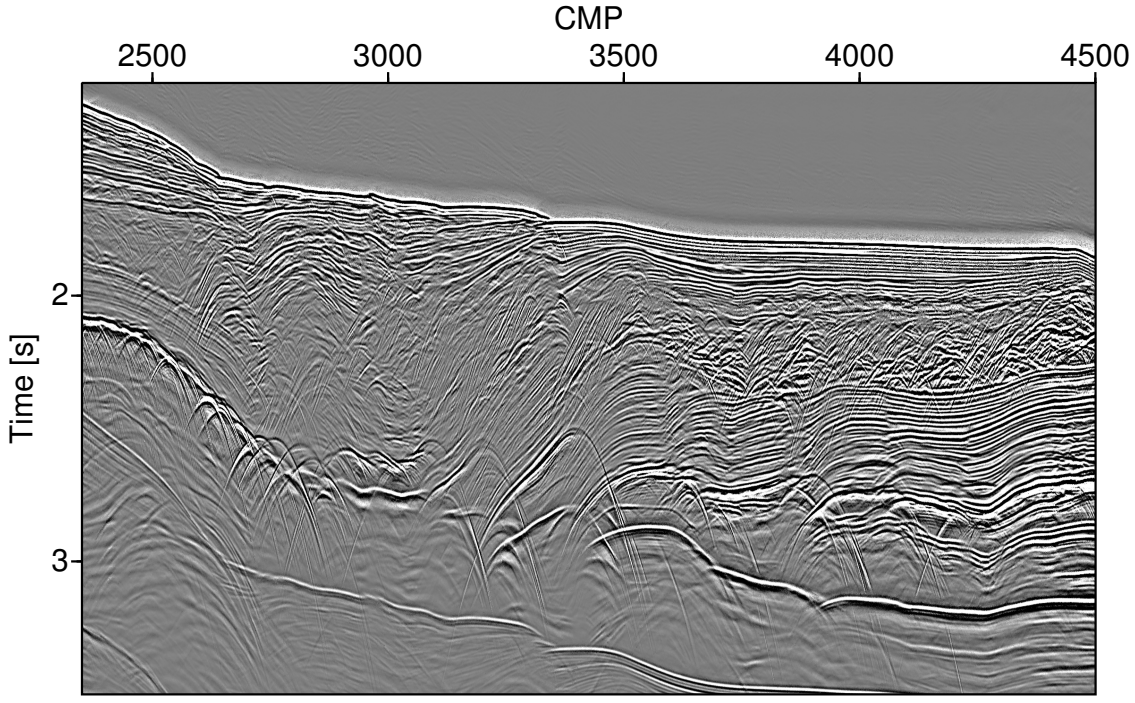


(b) nCRS stack of new implementation

Figure 5: Comparison of the previous implementation in (a) and the new implementation in (b). The new implementation does not recover as much events as the former method. However, CRS needs to be applied only once for each sample in (b), while CRS is applied 9 times in (a).



(a) nCRS stack of former implementation



(b) nCRS stack of new implementation

Figure 6: Comparison of obtained stacks from the previous and new implementation. The data shows offshore Israel in the Mediterranean Sea. Similar to the Sigsbee2A results, less events are recognized and some diffraction tails are interrupted. However, the thin layers near the sea floor are much better resolved. Additionally, the image obtained by the new implementation seems less noisy.

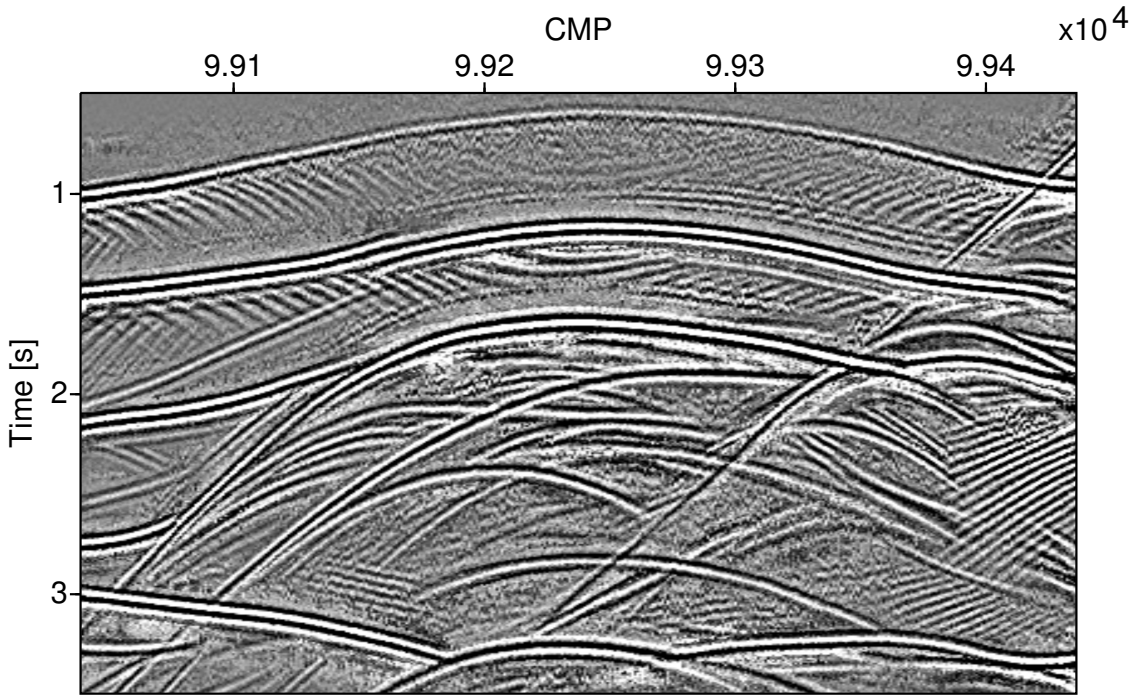


Figure 7: Inline section of the 3D SEG salt model containing many intersecting events. Since this data has very limited azimuth information, we can only cluster using the slowness in inline direction. This might be one reason we do not recover all intersecting events. The poor quality comes from the lack of a final local optimization that we used in the 2D case.