Using Adaptive Differential Evolution algorithm to improve parameter estimation in seismic processing

José Ribeiro* (CEPETRO/UNICAMP), Nicholas T. Okita (CEPETRO/UNICAMP), Tiago A. Coimbra (CEPETRO/UNICAMP), Gustavo B. Ignácio (CEPETRO/UNICAMP), and Martin Tygel (CCES/CEPID and CEPETRO/UNICAMP)

Abstract

Since the end of the 1990s, the methods of imaging and inversion seismic have been given systematic attention, through multiparametric traveltimes, such as the Common-Reflection-Surface (CRS) method, in its two versions zero offset (ZO) and finite offset (FO). Despite its superior quality to traditional methods, CRS faces the challenges of additional computation costs, which stem the required multiparameter estimations. Because of their usefulness for many processing and imaging purposes, the problem of estimation of the slope, curvature, and velocity parameters (commonly called CRS parameters) reliably and efficiently has drawn attention in the seismic literature. Mathematically, approaches to solve that problem rely on global optimization techniques. The main challenges are robustness (small relative sensitivity to given initial values) and convergence speed. The Differential Evolution (DE) has shown promising results. That method has a welcome property of robustness, however also the drawback of undesired convergence speed. In this paper, we propose to overcome this problem upon the application of the Adaptive Differential Evolution known as JADE. Qualitative results from synthetic and real datasets show the fast convergence of JADE when compared to that of DE, with similar execution time. Therefore, JADE presents itself as a great alternative to DE, showing even more promising results regarding the estimation of the ZO-CRS and FO-CRS parameters.

Introduction

In seismic data processing, reliable and accurate traveltime approximations to reflection and diffraction events play an important role as stacking operators to generate initial sections or volumes from the given data. In general, a finite number of parameters define such traveltime operators. For each given data sample, a corresponding set of parameters is to be estimated by coherency (semblance) analysis directly applied to the given dataset. More specifically, assuming a given data sample, for each given candidate set of parameters, the corresponding operator is constructed, and the data is stacked along that traveltime. The energy of the stack (semblance value) is computed. We refrain here to provide the mathematical expression of the semblance, which is well known in seismic processing. For a more detailed explanation of the semblance quantity, the reader is referred to, e.g., Neidell and Taner (1971). The semblance plays the role of the objective function. The set of parameters that maximizes the sample is the desired parameter set.

In this paper, two traveltime operators are selected, upon which parameter-estimation experiments are analyzed. These are the hyperbolic ZO-CRS and FO-CRS, namely zero-offset (ZO) and finite-offset (FO) common-reflection-surface (CRS) traveltimes, respectively. The ZO-CRS and FO-CRS are employed for parameter estimations on a given 2D dataset. We also refrain to provide the mathematical expressions of the ZO-CRS and FO-CRS operators, referring the reader to the publications Jäger et al. (2001) and Zhang et al. (2001), respectively. For each data sample in the post-stack domain, the ZO-CRS traveltime depends on three parameters, namely one slope and two curvatures (see, e.g., Jäger et al., 2001). For each data sample in the prestack domain, the FO-CRS traveltime depends on five parameters, namely two slopes and three curvatures (see, e.g., Zhang et al., 2001). Finally, to correct the stretching due to the moveouts used, we use a stretch-free method developed by Faccipieri et al. (2018).

Parameter estimation

Concerning the parameter-estimation problem for the 2D ZO-CRS, which involves three parameters, Jäger et al. (2001) proposed a scheme that consists of a sequence of one-parameter searches in suitable sub-domains of the data, followed by a local, three-parameter search having the previously-obtained parameters as initial values. A generalization of that scheme has been proposed in Garabito et al. (2001), that is based on an auxiliary two-parameter (diffraction) traveltime operator. At first, the simulated annealing (SA) method is applied to provide a simultaneous estimate of these two parameters, upon which a (partial) stacked section is constructed. Next, a one-parameter search applied to that partial stacked section to estimate the third parameter. In the same way as before, a local three-parameter search is carried out using the previous estimations as initial values.

From basic mathematical principles, sequential, few-parameter (cheaper) searches of the parameters in sub-domains are expected followed by a final, full-parameter local search are expected to provide less accurate estimations than the ones provided by simultaneous, full-parameter, global (much more expensive) searches. Ernesto Bonomi and Marchetti (2009) applied the global conjugate direction optimization method (see Powell, 1964) to simultaneously estimate the eight parameters that define the 3D ZO-CRS operator. Estimations based on the global Differential Evolution (DE) method have been recently reported in Barros et al. (2015) and Walda and Gajewski (2017).

In this work, we address the parameter-estimation problem that corresponds to the 2D ZO- and FO-CRS. Our approach is based on the JADE Adaptive Differential Evolution (see, e.g. Zhang and Sanderson, 2009), which combines the good accuracy provided by the DE algorithm, at the same time avoiding its drawback of slow convergence. The examples provided below confirm the expected good results.

Methodology

The methodology employed in our parameter-estimated problem consists of two global optimization methods (or meta heuristics) *Differential Evolution (DE)* and its variant *Adaptive Differential Evolution (JADE)*.

Differential Evolution (DE):

Differential Evolution (Storn and Price, 1997), or simply DE, falls into the category of nature-inspired, evolution meta heuristics (i.e., global optimization algorithmic methods for which there is no mathematical convergence proof). DE tries to formulate optimization problems as evolutionary processes: a population (made up by solution-candidate individuals) is subjected to an iteration process (with Ng iterations), upon which it transforms itself (evolves), according to preassigned rules, to an optimal population (that consists of the solution individuals). The approach is divided into two main steps. First, an initial population (or initial generation) P with Np individuals is created. Each individual is a D-dimension vector, for which each entry is a randomly selected number of a given real interval. Second, an iterative process is applied, such that each iteration represents a new (evolved) population (or generation). The evolution that takes place at each iteration follow the preassigned rules of mutation, crossover and selection described below:
Mutation: At a given iteration, we use the notation $P[j][i]$ for the i-th element of the j-th individual. Three (mutually different) random indexes $r_1, r_2, r_3 \in [1, Np]$, are chosen, based on a uniform integer distribution. In addition, a scale factor $F \in (0,2)$ is fixed for the whole optimization process. The mutant element $V[j][i]$ (i-th entry of the j-th mutant individual $V[j]$) is then given by

$$V[j][i] = P[r_1][i] + F * (P[r_2][i] - P[r_3][i]) \quad \begin{matrix} i \in [1, D] \\ j \in [1, Np] \end{matrix} \quad , \tag{1}$$

Crossover: At the given iteration, the crossover operation involves the elements $P[j][i]$ and $V[j][i]$, from the j-th individual and its corresponding mutant, respectively. Likewise the mutation, a crossover factor $CR \in [0,1]$ is fixed for the whole process. We now introduce the index i_{rand} extracted from a uniform random integer distribution of $\{1, \cdots, D\}$ and $frand$ a uniform random generator of real numbers in the real interval $[0,1]$. The element $U[j][i]$, of crossover individual $U[j]$ is given by

$$U_j[i] = \begin{cases} V[j][i] & \text{if } (frand < CR) \ or \ (i == i_{rand}) \\ P[j][i] & \text{otherwise.} \end{cases} \quad , \tag{2}$$

Selection: That consists of the choice between $P[j]$ and $U[j]$, as the one for which the largest value of objective function f is achieved. In symbols,

$$P[j] = \begin{cases} U[j] & \text{if } f(P[j]) < f(U[j]), \\ P[j] & \text{otherwise.} \end{cases} \quad , \tag{3}$$

Adaptive Differential Evolution (JADE):

Adaptive Differential Evolution (Zhang and Sanderson, 2009), or JADE, can be understood as a DE variant, since initialization of population P is the same, and generation's stages are similar. However, three new changes are introduced: these are an adaptation of control parameters F and CR, a new mutation strategy called $/de/current-to-pbest/1$ and an optional archive population. Such modifications appear in the mutation and crossover stages, in which the F and CR parameters are not anymore fixed, as previously. Instead, they are generated at each iteration for each individual j in the population, being accordingly denoted F_j and CR_j, respectively given by

$$F_j = cauchy((\mu_F, 0.1), \quad \text{and} \quad CR_j = normal(\mu_{CR}, 0.1). \tag{4}$$

In the above equation, $cauchy(\mu_F, 0.1)$ means the Cauchy distribution with scale and location values 0.1 (fixed here for simplicity) and mean value μ_F. In the same way, $normal(\mu_{CR}, 0.1)$ means the normal distribution with standard deviation 0.1 (also fixed) and mean value μ_{CR}. The two adaptive parameters μ_F and μ_{CR} are initially set to 0.5 and 0.9, respectively, and then updated at the end of each generation by:

$$\mu_F = 0.9 \cdot \mu_F \ + \ 0.1 \cdot mean_L(S_F), \tag{5}$$

$$\mu_{CR} = 0.9 \cdot \mu_{CR} \ + \ 0.1 \cdot mean_A(S_{CR}), \tag{6}$$

Here, S_F and S_{CR} denote the sets composed by the successful parameters F_j and CR_j, respectively, both associated with the $U[j]$ individual when the first statement of equation 3 is applied. Moreover, $mean_L(S_F)$ and $mean_A(S_{CR})$ designate the Lehmer and arithmetic mean values of the sets S_F and S_{CR}, respectively. We recall that expression of the Lehmer mean reads

$$mean_L = \frac{\sum\limits_{F \in S_F} F^2}{\sum\limits_{F \in S_F} F}, \tag{7}$$

Based on the above modifications, a new mutation strategy is proposed called $/de/current-to-pbest/1$: For a given percentage number p, with $0 \le p \le 1$, the $100p\%$ best individuals are selected to generate the mutated individual (vector) $V[j]$ according to the expression

$$V[j] = P[j] + F_j * m(P[j_{best}] - P[j]) + F_j * (P[r_1] - P[r_2]). \tag{8}$$

Here, the three indexes $j_{best} \in [1, \lfloor Np \cdot p \rceil]$, $r_1, r_2 \in [1, Np]$, are chosen according to an integer uniform distribution. The population is then sorted so that the j_{best} individual, for which the objective function achieves the maximum value, occupies the position 1, the remaining ones (until j_{best} position) occupying successive positions according decreasing values of the objective function.
Remark: The population diversity of the algorithm can be improved upon the introduction of an archive population A, populated with the j-th individual $P[j]$, for which failed to provide an objective function greater than the one achieved by the individual $U[j]$, defined by equation 3. With the help of the archive A, the individual $P[r_2]$ in equation 8 is such that its index r_2 now ranges within the interval $[1, Np + |A|]$ and moreover, that individual is replaced by $\tilde{P}[r_2]$, given by

$$\tilde{P}[r_2] = \begin{cases} P[r_2] & \text{if } r_2 \le Np, \\ P[r_2 - Np] & \text{otherwise.} \end{cases} \tag{9}$$

The maximum established size of the archive is set to be Np and always when this limit is reached an individual is randomly removed from its population, using for that an uniform integer distribution. The implementation of JADE is described in algorithm 1. In it, it is assumed a vector notation for the population. Further, mutation, crossover and boundary verification are collapsed in one block (lines 31-39). Also, the used term $irand(a,b)$ represents an integer random generator chosen in the interval $[a,b]$.

Experiments

We consider three 2D seismic datasets DIFRAT, SOL, and JEQUITI in which the first two are synthetic and the third a real dataset. Their specifications are displayed in Table 1. The midpoint and offset distances shown for JEQUITI dataset are the mean value of all midpoint and offset distances because it is a real dataset not regular. Besides, JEQUITI dataset is a marine two-dimensional (2D) line from the Jequitinhonha basin at the coast of Bahia, Brazil.

The estimation processes were executed on an Amazon Web Services Elastic Computing Cloud (AWS EC2) instance with 72 Xeon Platinum 8124 3GHz vCores and 144GB of RAM, running on Ubuntu 18.04. The ZO-CRS and FO-CRS code implementations, both for DE and JADE, were compiled with the *gcc* compiler in its version *7.3.0*, using the *-O3* optimization flag. Furthermore, the programs were implemented following the Scalable Partially Idempotent Task System (SPITS) programming model by Borin et al., 2016, then executed using the PY-PITS run time by Benedicto et al., 2017. Moreover, the parameter p from line 3 of the algorithm 1 was set to be 0.2.

Results

For the given datasets, the parameter estimations provided by JADE and DE algorithms are compared and discussed. The cases of ZO CRS and FO CRS are treated separately.

ZO CRS: Tables 2 and 3 show the average execution times of JADE and DE for the different datasets. We can observe that both JADE and DE complete the same number of iterations in very similar times. However, from the coherence panels shown in Figures 1 and 2, it becomes apparent the advantage of JADE over DE in terms of quality. Consider, for instance, the ZO semblance section of the DIFRAT dataset (Figure 1). Comparison of Figures 1c and 1d (refer to generation 21 and population 21 obtained by JADE and DE) shows that JADE has better quality than DE. On the other hand, the difference between Figures 1e and 1f (which represent generation 31 and population 31 obtained by JADE and DE, respectively) is negligible, the reason is that both heuristics were able to converge with that many iterations.

Furthermore, take the JEQUITI dataset as an example (Figure 2). Differently, from the DIFRAT dataset, the convergence between JADE and DE is not that much apparent, being both pretty similar. Nonetheless, some areas of Figure 2a are slightly better than in the same areas in Figure 2b, as illustrated by the red box in both images. Besides that, the same comparison applies to the stacked JEQUITI dataset, shown in Figure 3.

Therefore, JADE performs, in ZO case, in terms of coherence, equally or better than DE for both synthetic and real datasets. Further, when taking run time into

Algorithm 1 Pseudo-code for JADE

```
 1: $\mu_{CR} = 0.9$; $\mu_F = 0.5$;
 2: $pBest = \lfloor p \times Np \rceil$; $sizeA = 0$;
 3:
 4: Create an empty population $A$ of size Np
 5: Create a random initial population $P$ of size Np
 6:
 7: // Iterates through all Ng generations
 8: for $g = 1$ to $Ng$ do
 9:     $meanSF = 0.0$; $meanSF2 = 0.0$;
10:     $meanSCR = 0.0$; $meanSCR2 = 0.0$;
11:
12:     // Sorts the pBest first individuals
13:     for $j = 1$ to $pBest$ do
14:         for $k = j+1$ to $Np$ do
15:             if $f(P[k]) > f(P[j])$ then
16:                 Swap $P[k]$ with $P[j]$
17:
18:     // Updates each individual
19:     for $j = 1$ to $Np$ do
20:         do $F_j = min(1.0,\ cauchy(\mu_F, 0.1))$;
21:         while $F_j \leq 0$
22:
23:         $CR_j = min(1.0,\ max(0.0,\ normal(\mu_{CR}, 0.1)))$;
24:
25:         $\boldsymbol{u} = P[j]$;
26:         $\boldsymbol{x}_1 = P[irand(1, Np)]$;
27:         $\boldsymbol{x}_{best} = P[irand(1, pBest)]$;
28:         $r2 = irand(1, Np + sizeA)$;
29:
30:         if $r2 \leq Np$ then $\boldsymbol{x}_2 = P[r2]$;
31:         else $\boldsymbol{x}_2 = A[r2 - Np]$;
32:
33:         $i_{rand} = irand(1, D)$
34:         for $i = 1$ to $D$ do
35:             if $i == i_{rand}$ or $frand(0,1) < CR_i$ then
36:                 $\boldsymbol{u}[i] += F_j \times (\boldsymbol{x}_{best}[i] - \boldsymbol{u}[i] + \boldsymbol{x}_1[i] - \boldsymbol{x}_2[i])$;
37:
38:                 // Verify lower boundary
39:                 if $\boldsymbol{u}[i] < low[i]$ then
40:                     $\boldsymbol{u}[i] = (low[i] + P[j][i])/2$;
41:
42:                 // Verify higher boundary
43:                 if $\boldsymbol{u}[i] > high[i]$ then
44:                     $\boldsymbol{u}[i] = (high[i] + P[j][i])/2$;
45:
46:         if $f(\boldsymbol{u}) > f(P[j])$ then
47:             $P[i] = \boldsymbol{u}$;
48:
49:             $meanSF += F_j$; $meanSF2 += F_j \times F_j$;
50:             $meanSCR += CR_j$; $meanSCR2 += 1.0$;
51:
52:             if $sizeA < Np$ then
53:                 $sizeA += 1$; $A[sizeA] = \boldsymbol{u}$;
54:             else
55:                 $A[irand(1, sizeA)] = \boldsymbol{u}$;
56:
57:     if $meanSCR2 > 0.0$ then
58:         $\mu_F = 0.9 \times \mu_F + 0.1 \times (meanSF2\ /\ meanSF)$;
59:         $\mu_{CR} = 0.9 \times \mu_{CR} + 0.1 \times (meanSCR\ /\ meanCR2)$;
```

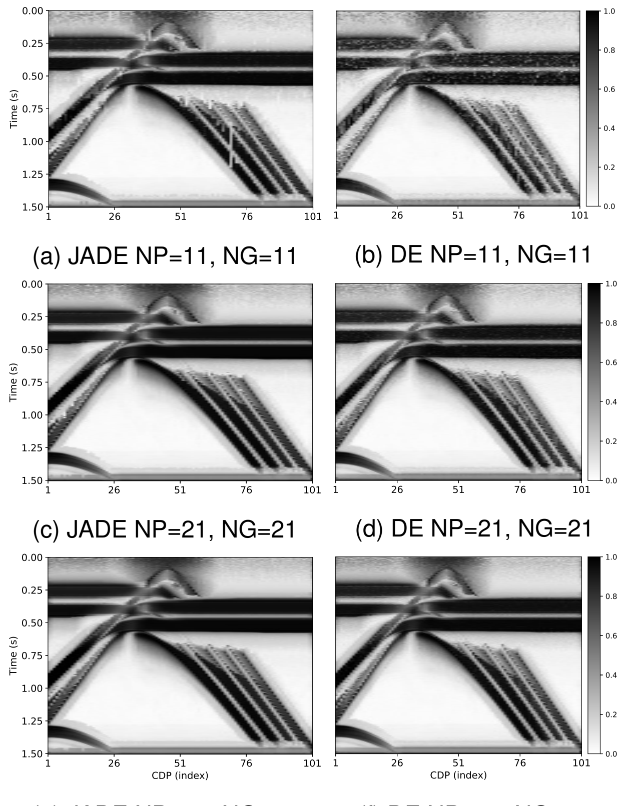
Table 1 Dataset specifications

Name	Midpoint Distance (m)	Offset Distance (m)	Number of Traces	Number of time samples per trace	Sampling interval (ms)
DIFRAT	20	40	2626	376	4
SOL	20	80	4623	700	4
JEQUITI	11.17	90.18	58189	1751	4

consideration, both heuristics are very close to each other. Hence, there is nothing to justify the use of the latter method over the former.

Table 2 ZO-CRS execution time: DIFRAT synthetic dataset

Population	Number of	Execution Time		
Size	Generations	JADE (m)	DE (m)	JADE / DE (%)
11	11	0.21	0.21	100.01
21	21	0.68	0.66	102.85
31	31	1.46	1.42	102.67



(a) JADE NP=11, NG=11 (b) DE NP=11, NG=11

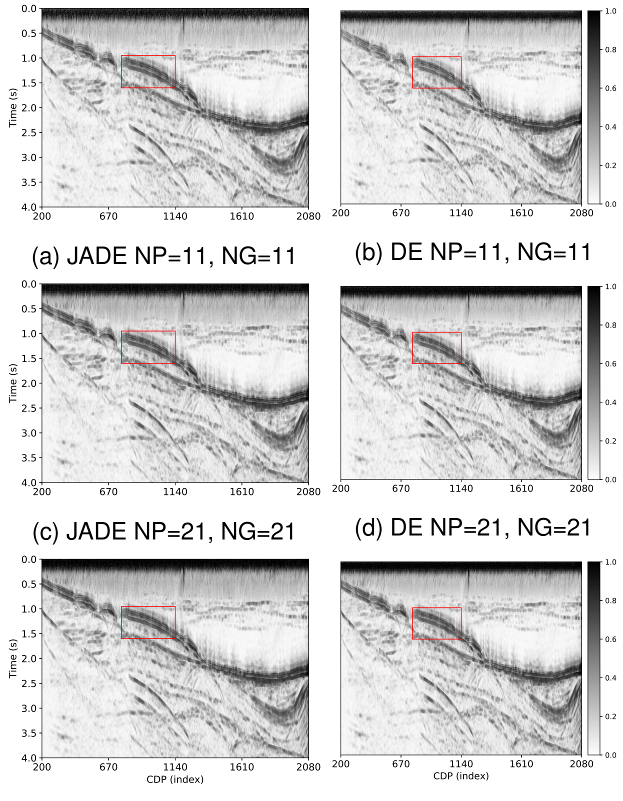
(c) JADE NP=21, NG=21 (d) DE NP=21, NG=21

(e) JADE NP=31, NG=31 (f) DE NP=31, NG=31

Figure 1: ZO-CRS semblance comparison between JADE and DE in DIFRAT dataset, related with table 2

Table 3 ZO execution time: JEQUITI real dataset

Population	Number of	Execution Time		
Size	Generations	JADE (m)	DE (m)	JADE / DE (%)
11	11	2.66	2.67	99.44
21	21	8.67	8.60	100.82
31	31	18.32	18.26	100.29



(a) JADE NP=11, NG=11 (b) DE NP=11, NG=11

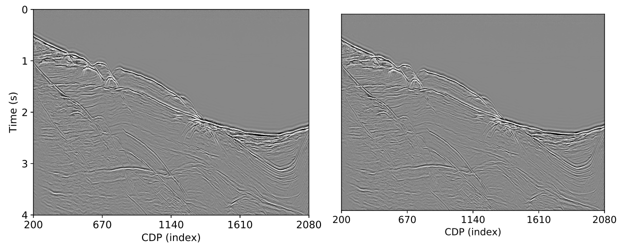
(c) JADE NP=21, NG=21 (d) DE NP=21, NG=21

(e) JADE NP=31, NG=31 (f) DE NP=31, NG=31

Figure 2: ZO-CRS semblance comparison between JADE and DE in JEQUITI dataset, related with table 3

FO CRS: Now, we present and analyze the parameter-estimation results obtained from both JADE and DE for the case of FO-CRS. Tables 4 and 5 show the execution times for the SOL and JEQUITI datasets. Here, due to the longer running period, the run time results were extracted from a single execution (and not two as in the ZO tables). Despite of this, as shown in Table 4, the ratio JADE/DE is still close to 100%. In other words, the run time performance difference between the techniques is, as in ZO part, almost negligible.

Meanwhile, based in Table 5, it is noticeable that JADE offered faster execution times than DE for lower iteration counts (i.e., 21 population individuals and 21 generations), while had about the same performance with higher iterations (i.e., 31 population individuals and 31



(a) JADE NP=31, NG=31 (b) DE NP=31, NG=31

Figure 3: ZO-CRS stack comparison between JADE and DE in JEQUITI dataset, related with table 3

generations). Nevertheless, despite this last result, still 93.3% of all executions had run times very similar using both heuristics.

Regarding the qualitative results, Figure 4 shows the objective function values for the first three offsets in SOL dataset, while Figure 6 shows these values for the first three offsets in JEQUITI dataset.

Considering the first image (Figure 4), notice that the increase in population and generation numbers improve the objective function values using both DE and JADE heuristics, but, again, JADE provided a much better convergence even in a simple synthetic data set, such as SOL (Figure 4), where a 21×21 JADE is extremely superior than the 21×21 DE. Here, the time difference is only a few seconds.

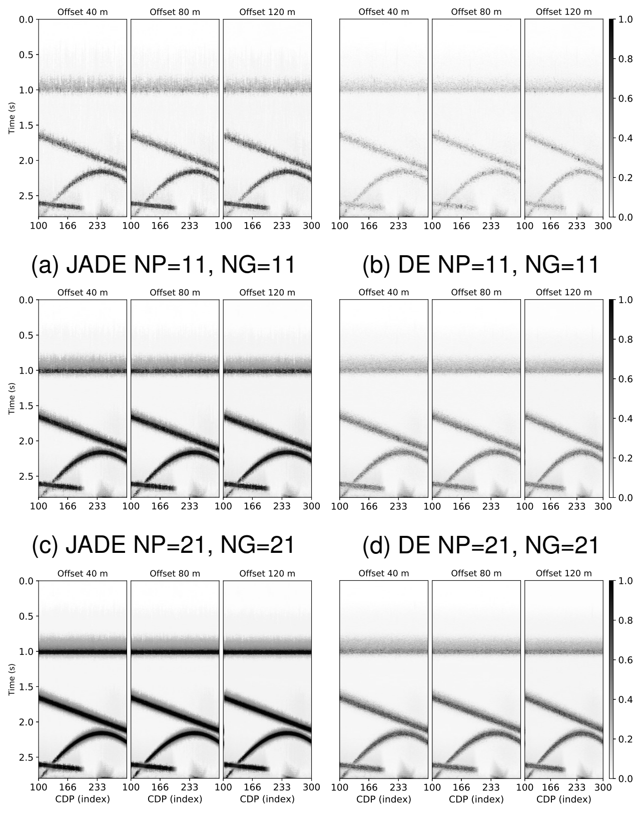
In accordance with the aforementioned, the same pattern repeats in the second image (Figure 6), where a DE with 21 individuals evolved during 21 generations (Figure 6d) shows a result much worst than a JADE with 21 individuals mutated during 21 iterations (Figure 6c). In this case, the execution time was smaller in JADE, being around 3.8% faster. This wider gap in results quality is due to the amount of parameters to be estimated, in which five are searched using the heuristics against only three in the zero-offset.

Table 4 FO-CRS execution time: SOL synthetic data set

Population Size	Number of Generations	Execution Time		
		JADE (m)	DE (m)	JADE / DE (%)
11	11	4.32	4.27	98.84
21	21	14.26	14.36	100.70
31	31	30.39	30.31	99.73

Table 5 FO-CRS execution time: JEQUITI real dataset

Population Size	Number of Generations	Execution Time		
		JADE (m)	DE (m)	JADE / DE (%)
11	11	305.98	353.26	86.61
21	21	843.59	868.03	97.21
31	31	1739.20	1756.08	99.04

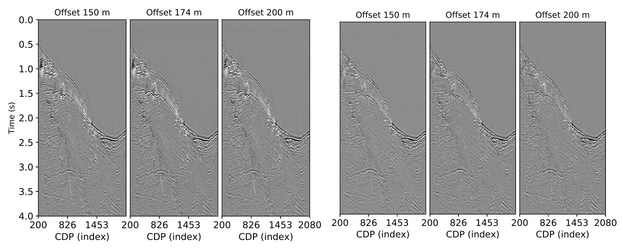


(a) JADE NP=11, NG=11 (b) DE NP=11, NG=11

(c) JADE NP=21, NG=21 (d) DE NP=21, NG=21

(e) JADE NP=31, NG=31 (f) DE NP=31, NG=31

Figure 4: FO semblance comparison between JADE and DE in data set Sol, related with table 4

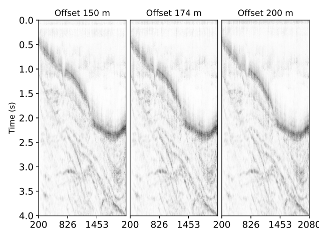


(a) JADE NP=31, NG=31 (b) DE NP=31, NG=31

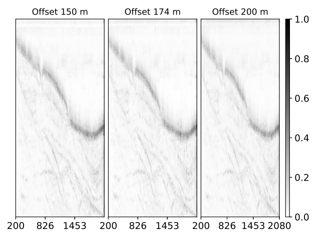
Figure 5: FO-CRS stack comparison between JADE and DE in JEQUITI dataset, related with table 5

Conclusions

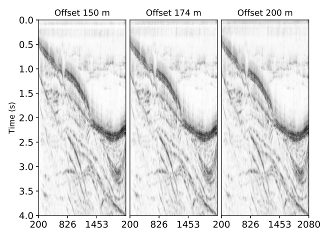
In this work, two heuristics were implemented to maximize the semblance objective function when estimating the parameters for seismic processing algorithms ZO-CRS and FO-CRS, namely DE and JADE. It is well known that DE is a robust solution. However, its convergence can take many iterations, making the final results either unsatisfying or take too long. On the other hand, JADE could be used as an alternative, improving convergence time and quality. The results showed that for the ZO-CRS program both heuristics were able to converge properly, with better objective function values by JADE when using a lower number of population and generations. Nevertheless, in a more complex program, such as FO-CRS that needs



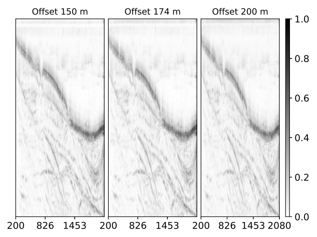
(a) JADE NP=11, NG=11



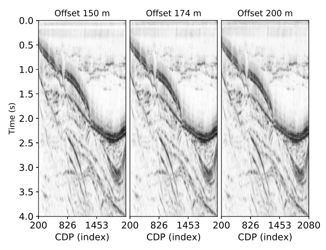
(b) DE NP=11, NG=11



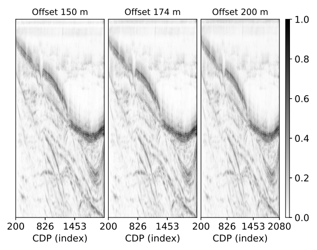
(c) JADE NP=21, NG=21



(d) DE NP=21, NG=21



(e) JADE NP=31, NG=31



(f) DE NP=31, NG=31

Figure 6: FO semblance comparison between JADE and DE in data set Jequiti, related with table 5

to estimate five parameters, JADE offered considerably better convergence, even in simple synthetic data sets and more so in complex real data sets. Furthermore, for the same number of population individuals and generations, the program execution time remained about the same. Finally, it is clear that JADE is an acceptable alternative to the commonly used DE, given its convergence and time performance. Also, it would be useful to have a working version of JADE that takes advantage of accelerators, i.e., graphics processing units (or GPUs), enabling shorter execution times while keeping high convergence.

Acknowledgments

This work was possible thanks to the support of Petrobras and Fapesp/Cepid #2013/08293-7-Brazil. The authors also thank the High-Performance Geophysics (HPG) team for technical support.

References

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

Benedicto, C., I. L. Rodrigues, M. Tygel, M. Breternitz, and E. Borin, 2017, Harvesting the computational power of heterogeneous clusters to accelerate seismic processing: Presented at the 15th International Congress of the Brazilian Geophysical Society & EXPOGEF, Rio de Janeiro, Brazil, 31 July-3 August 2017, Brazilian Geophysical Society.

Borin, E., C. Benedicto, I. L. Rodrigues, F. Pisani, M. Tygel, and M. Breternitz, 2016, Py-pits: A scalable python runtime system for the computation of partially idempotent tasks: 2016 International Symposium on Computer Architecture and High Performance Computing Workshops (SBAC-PADW), 7–12.

Ernesto Bonomi, Antonio M. Cristini1, D. T., and P. Marchetti, 2009, 3d crs analysis: a data-driven optimization for the simultaneous estimate of the eight parameters: Presented at the SEG Technical Program Expanded Abstracts 2009, Society of Exploration Geophysicists.

Faccipieri, J. H., T. A. Coimbra, and R. Bloot, 2018, Stretch-free generalized normal moveout correction: Geophysical Prospecting, **67**, 52–68.

Garabito, G., J. C. Cruz, P. Hubral, and J. Costa, 2001, Common reflection surface stack: A new parameter search strategy by global optimization: Presented at the SEG Technical Program Expanded Abstracts 2001, Society of Exploration Geophysicists.

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

Neidell, N. S., and M. T. Taner, 1971, SEMBLANCE AND OTHER COHERENCY MEASURES FOR MULTICHANNEL DATA: GEOPHYSICS, **36**, 482–497.

Powell, M. J. D., 1964, An efficient method for finding the minimum of a function of several variables without calculating derivatives: The Computer Journal, **7**, 155–162.

Storn, R., and K. Price, 1997, Storn and price: Journal of Global Optimization, **11**, 341.

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

Zhang, J., and A. C. Sanderson, 2009, Jade: Adaptive differential evolution with optional external archive: IEEE Transactions on Evolutionary Computation, **13**, 945–958.

Zhang, Y., S. Bergler, and P. Hubral, 2001, Common-reflection-surface (CRS) stack for common offset: Geophysical Prospecting, **49**, 709–718.