Computers and Geosciences 113 (2018) 81–93

Contents lists available at ScienceDirect

Computers and Geosciences

journal homepage: www.elsevier.com/locate/cageo

Research paper

A parallel competitive Particle Swarm Optimization for non-linear first arrival traveltime tomography and uncertainty quantification

Keurfon Luu [a,*], Mark Noble [a], Alexandrine Gesret [a], Nidhal Belayouni [b], Pierre-François Roux [b]

[a] MINES ParisTech, PSL – Research University, Centre de Géosciences, 35, rue Saint-Honoré, 77305 Fontainebleau, France
[b] Baker Hughes, a General Electric Company, France

ARTICLE INFO

Keywords:
Microseismic
Traveltime tomography
Particle swarm optimization
Uncertainty quantification
High performance computing

ABSTRACT

Seismic traveltime tomography is an optimization problem that requires large computational efforts. Therefore, linearized techniques are commonly used for their low computational cost. These local optimization methods are likely to get trapped in a local minimum as they critically depend on the initial model. On the other hand, global optimization methods based on MCMC are insensitive to the initial model but turn out to be computationally expensive. Particle Swarm Optimization (PSO) is a rather new global optimization approach with few tuning parameters that has shown excellent convergence rates and is straightforwardly parallelizable, allowing a good distribution of the workload. However, while it can traverse several local minima of the evaluated misfit function, classical implementation of PSO can get trapped in local minima at later iterations as particles inertia dim. We propose a Competitive PSO (CPSO) to help particles to escape from local minima with a simple implementation that improves swarm's diversity. The model space can be sampled by running the optimizer multiple times and by keeping all the models explored by the swarms in the different runs. A traveltime tomography algorithm based on CPSO is successfully applied on a real 3D data set in the context of induced seismicity.

1. Introduction

Tomographic inversion schemes aiming at reconstructing the subsurface structures from seismic traveltime data are widely used (e.g. Rawlinson et al. (2010)). The obtained wave propagation velocity distribution is usually a starting point for further analysis at various scales, from near surface to global scale. For a reliable and quantitative interpretation of the tomographic solution, an accurate velocity model with its associated uncertainties are required.

In spite of the fact that the inversion for the velocities is a totally non-linear problem, very often it is solved with iterative linearized approaches that minimize a misfit function. The misfit function usually measures the difference between observed and computed traveltimes as a function of the velocity model parameters. The linearization makes the implicit assumption of a unique solution which is chosen thanks to a regularization procedure that reduces the solution non-uniqueness (Menke, 2012). This is generally achieved by imposing the solution to be somehow similar or close to an initial a priori model.

The data-model (traveltimes-velocities) relationship can be highly non-linear and requires the use of global optimization methods. In addition, the linearized approaches are not really adapted to provide reliable uncertainties. From a theoretical point of view, to address these two issues, methods based on Markov Chain Monte Carlo (MCMC) that sample the velocity model parameter space are required, such as reversible-jump MCMC (Green, 1995; Bodin and Sambridge, 2009), Parallel Tempering (Sambridge, 2014), or Interactive MCMC (Bottero et al., 2016). These global optimization methods can be applied on non-smooth and non-convex functions as they are derivative-free and produce results independent of the initial model. However, they cannot be parallelized and turn out to be prohibitive in terms of computation time.

Another class of global optimization methods has shown growing interest in the last decades. These methods, known as evolutionary algorithms (EA), are inspired by the natural evolution of species and have demonstrated very good convergence rates. While MCMC methods sample the model parameter space by perturbing iteratively a single model, EA work with a population of simultaneous models that evolve toward better models through stochastic processes. This simultaneous evaluation of independent models implies that it is straightforward to parallelize and thus can significantly reduce the computation time. EA

* Corresponding author.
E-mail addresses: keurfon.luu@mines-paristech.fr (K. Luu), mark.noble@mines-paristech.fr (M. Noble), alexandrine.gesret@mines-paristech.fr (A. Gesret), nidhal.belayouni@bhge.com (N. Belayouni), pierre-francois.roux@bhge.com (P.-F. Roux).

https://doi.org/10.1016/j.cageo.2018.01.016
Received 11 August 2017; Received in revised form 18 January 2018; Accepted 24 January 2018
Available online 31 January 2018
0098-3004/

include Genetic Algorithm (Sambridge and Drijkoningen, 1992; Whitley, 1994), Differential Evolution (Storn and Price, 1997; Barros et al., 2015), and Covariance Matrix Adaptation Evolution Strategy (Hansen and Ostermeier, 1996; Grayver and Kuvshinov, 2016).

In this work, we propose to overcome the non-linearity using a rather new EA known as Particle Swarm Optimization (PSO) for its ease of implementation and the low number of tuning parameters required. PSO has been introduced to study birds flocking and fish schooling (Kennedy and Eberhart, 1995). While it has been extensively used in other engineering domains (e.g. biomedical, signal processing …) for years, PSO has been fairly ignored by the geophysical community until recently. In seismics, PSO has been applied in history matching for reservoir characterization (Mohamed et al., 2010; Fernández Martínez et al., 2012), traveltime tomography (Tronicke et al., 2012; Rumpf and Tronicke, 2015; Poormirzaee et al., 2015), and surface waves inversion (Wilken and Rabbel, 2012; Poormirzaee, 2016). Yet, PSO may suffer from premature convergence, in particular for functions with complex shapes. Therefore, we propose and describe a simple modification of PSO to tackle premature convergence and improve its robustness. Although PSO is mainly used as a global optimization method, we show that our implementation not only demonstrates better convergence rates, but also samples correctly the model parameter space, allowing more reliable uncertainty quantification. We apply the method on a real 3D microseismic example and sample the model parameter space which allows us to derive reliable velocity model uncertainties.

2. Theory and method

Geophysical inverse problems are underdetermined optimization problems that can be solved by either linear or non-linear techniques (Tarantola and Valette, 1982). Let us define the discrete data vector $\mathbf{d} = [d_1, \dots, d_n]^\top$, where n is the number of data points. Calculated data are generated by applying the forward modeling operator g, most often non-linear, on the model vector $\mathbf{m} = [m_1, \dots, m_p]^\top$, with p the number of parameters defining the model

$$\mathbf{d}^{calc} = g(\mathbf{m}). \tag{1}$$

Inverse problems consist in determining the model vector $\mathbf{m}$ that minimizes the misfit between the observed data and the calculated data

$$\mathbf{e}(\mathbf{m}) = \mathbf{d}^{obs} - \mathbf{d}^{calc} = \mathbf{d}^{obs} - g(\mathbf{m}). \tag{2}$$

Given an error vector $\mathbf{e}$ (Equation (2)), the misfit function is usually defined with an ℓ_p -norm. In geophysical inverse problems, even though other norms can be found in the literature, the ℓ_2 -norm is often used

$$||\mathbf{e}(\mathbf{m})||_2 = \left[\left(\mathbf{d}^{obs} - g(\mathbf{m})\right)^\top \left(\mathbf{d}^{obs} - g(\mathbf{m})\right)\right]^{\frac{1}{2}}. \tag{3}$$

The non-linearity can be addressed by global optimization methods that explore the model parameter space. In this section, we first describe the PSO algorithm before introducing a more robust implementation based on PSO that tackles its shortcomings.

2.1. Particle Swarm Optimization

For consistency in the notation, the so-called position vector usually denoted by $\mathbf{x}$ in the literature will be denoted by $\mathbf{m}$. Consequently, we will only speak in terms of models instead of position vector.

In PSO, the first step is to generate a swarm composed of several models in the model parameter space. The initial models can either be defined a priori or generated given a random distribution (usually uniform). Each model is represented by a particle that interacts with its neighborhood to find the global minimum of the misfit function. Kennedy (1999) has studied several neighborhood topologies and concluded that the global best topology (all the particles are connected to each other) performed better than the others. Thus, we here only consider the global best topology where the neighborhood of each particle is the entire swarm.

At iteration k, a particle i is defined by a model vector $\mathbf{m}_i^k$ and a velocity vector $\mathbf{v}_i^k$ and is adjusted according to its own personal best model and the global best model of the whole swarm. The velocity vector controls how a particle moves in the model parameter space and is initialized to zero (Engelbrecht, 2012). The velocity and the position of each particle are updated following

$$\mathbf{v}_i^k = \omega \mathbf{v}_i^{k-1} + \phi_p \mathbf{r}_p^k \left(\mathbf{m}_{p,i} - \mathbf{m}_i^{k-1}\right) + \phi_g \mathbf{r}_g^k \left(\mathbf{m}_g - \mathbf{m}_i^{k-1}\right) \tag{4}$$

$$\mathbf{m}_i^k = \mathbf{m}_i^{k-1} + \mathbf{v}_i^k \tag{5}$$

where $\mathbf{m}_{p,i}$ and $\mathbf{m}_g$ are respectively the personal best model of particle i and the global best model of the swarm, $\mathbf{r}_p^k$ and $\mathbf{r}_g^k$ are uniform random numbers vectors drawn at iteration k, ω is an inertia weight, ϕ_p and ϕ_g are two acceleration parameters that respectively control the cognition and social interactions of the particles.

The inertia weight ω has been introduced by Shi and Eberhart (1998) to help the particles to dynamically adjust their velocities and refine the search near a local minimum. Another formulation using a constriction coefficient based on Clerc (1999) to insure the convergence of the algorithm can be found in the literature. However, Eberhart and Shi (2000) showed that the inertia and constriction approaches are equivalent since the parameters are connected.

Empirical studies have concluded that the performance of PSO is sensitive to its control parameters, namely the swarm size s, the maximum number of iterations $k_{\max}$, ω, ϕ_p and ϕ_g. Yet, these studies have provided some insights on the initialization of some parameters (Van Den Bergh and Engelbrecht, 2006). Eberhart and Shi (2000) empirically found that $\omega = 0.7298$ and $\phi_p = \phi_g = 1.49618$ are good parameter choices that lead to convergent behaviour. Although these parameters have shown good results in previous studies, be aware that they can also be tuned according to the optimization problem. The sensitivity of PSO to these parameters is analyzed in Section 3.1. Unless explicitly stated, we will set $\omega = 0.7298$ and $\phi_p = \phi_g = 1.49618$.

The swarm size and the maximum number of iterations have to be carefully chosen dependently of the problem and the computer resources available. These two parameters are related since a smaller swarm will require more iterations to converge, while a bigger swarm will converge more rapidly. In real optimization problems, the computation cost is mainly dominated by the forward modeling. Therefore, the optimization is usually stopped when a predefined number of forward modelings (i.e. computation of misfit functions) is performed. The desired number of forward modelings is controlled by both the swarm size and the maximum number of iterations. Trelea (2003) has studied the effect of the swarm size on several benchmark test functions in 30 dimensions. He found that a medium number of particles (≈ 30 particles) gives the best results in terms of number of misfit function evaluations. Too few particles (≈ 15 particles) gives a very low success rate while too many particles (≈ 60 particles) results in much more misfit function evaluations than needed although it increases the success rate. Piccand et al. (2008) came to the same conclusion with problems of higher dimensions (up to 500).

In the original PSO, the swarm's global best position is updated in a synchronous fashion. In other words, $\mathbf{m}_g$ is updated at the end of an iteration once the misfit functions of the entire swarm have been evaluated. Carlisle and Dozier (2001) has shown that PSO yields better performance when the particles are evaluated asynchronously (i.e. $\mathbf{m}_g$ is evaluated after each individual misfit evaluation). Synchronous PSO is intrinsically parallelizable and performs well if the individual misfit function evaluations require the same amount of time, while a parallel asynchronous PSO is not straightforward but could reduce wasted CPU cycles (Schutte et al., 2004; Koh et al., 2006). This work only deals with