A GPU-enabled Black-box Optimization in Application to Dispersion-based Geoacoustic Inversion

Vadim Bulavintsev
Matrosov Institute for System Dynamics and Control Theory SB RAS
Irkutsk, Russia
v.g.bulavintsev@gmail.com
Pavel Petrov
Il'ichev Pacific Oceanological Institute FEB RAS, Far Eastern Federal University
Vladivostok, Russia
petrov@poi.dvo.ru
Mikhail Posypkin
Dorodnicyn Computing Centre FRC CS RAS
Moscow, Russia
mposypkin@gmail.com
Oleg Zaikin
Matrosov Institute for System Dynamics and Control Theory SB RAS
Irkutsk, Russia
zaikin.icc@gmail.com

Abstract

In this study, a GPU-enabled implementation of an algorithm for the solution of a real-life optimization problem arising in the geoacoustic inversion is proposed. In the inversion algorithm, a single-hydrophone recording of a pulse acoustic signal is used for the estimation of the waveguide parameters from the dispersion data. We devised and implemented a hierarchical optimization scheme where and efficient GPU-enabled implementation of the objective function calculation is combined with the CPU implementation of the search algorithm. The GPU-enabled implementation turned out to be several dozen times faster than the pure CPU one. Exploiting this advantage we were able to successfully solve the considered problem in reasonable time.

1 Introduction

Black-box optimization methods can be employed to solve real-life problems from different areas (e.g., see [Evtushenko et al., 2016, Gornov et al., 2016]). In this study, we apply black-box optimization for solving one problem arising in geoacoustic inversion. The notion of geoacoustic inversion refers to a collection of techniques that can be used for the reconstruction of geoacoustical waveguide structure from the sound pressure

In: Yu. G. Evtushenko, M. Yu. Khachay, O. V. Khamisov, Yu. A. Kochetov, V.U. Malkova, M.A. Posypkin (eds.): Proceedings of the OPTIMA-2017 Conference, Petrovac, Montenegro, 02-Oct-2017, published at http://ceur-ws.org

measurements [Katsnelson et al., 2012]. A typical geoacoustical waveguide in a shallow sea is comprised by a water column and several layers of bottom sediments. Normally measurements for the geoacoustic inversion are performed using expensive receiver arrays. However, recently it was shown that single-hydrophone recording of a broadband pulse signal can be also successfully used for estimating the acoustical parameters of sea bottom [Bonnel & Chapman, 2011, Bonnel et al., 2012, Bonnel et al., 2013]. The implementation of the method from [Bonnel & Chapman, 2011] in practice can be thought of as a solution of an minimization problem, where every evaluation of the objective function requires numerous solutions of an acoustic spectral problem [Petrov, 2014, Zaikin & Petrov, 2016]. In the present paper we use GPU-enabled black-box optimization to solve the inversion problem mentioned above.

Let us give a brief outline of the paper. In Section 2 we describe an black-box optimization algorithm, used to solve the problem. In Section 3 we describe a problem of geoacoustic inversion. In Section 4 we describe two implementations of the proposed algorithm, and show the results of computational experiments on the test problem. In the rest of the paper we draw conclusions.

2 New Black-box Optimization Algorithm

In black-box optimization problems derivatives are either unavailable or very hard to compute. Thus, solution methods should not rely on derivatives. There are a lot of local search techniques which can be used for this purpose: Hooke-Jeeves method [Hooke & Jeeves, 1961], pseudo-gradient approaches [Evtushenko et al., 2016] and a variety of coordinate descent techniques. We developed a simple modification of the coordinate descent method presented in Fig. 1.

```
while max_{i=1,...,n} δ_i > ε do
    for i = 1 to n do
        x' = x + δ_i e_i
        if f(x') < f(x) then
            x := x'
            δ_i := α δ_i
        else
            x' = x − δ_i e_i
            if f(x') < f(x) then
                x := x'
                δ_i := α δ_i
            else
                δ_i := β δ_i
            end
        end
    end
end
```

Algorithm 1: ASN Search

The proposed algorithm (ASN search) uses asymmetric neighborhood adaptation. It tries to improve the objective function $f(\mathbf{x})$ by shifting by some value δ_i along each coordinate vector $\mathbf{e_i}$ in both directions. In the case of success (failure) the respective step size is increased (decreased) by multiplying it on $\alpha > 1 (\beta < 1)$. When approaching minimum values of δ_i usually decrease. The method terminates when the maximal component of the vector δ becomes less that the predefined $\varepsilon > 0$.

The choice of initial δ and ε significantly affects the performance. For the problem under consideration the best performance were obtained with $\delta_i = 10^{-1}, i = 1 \ldots, n$, $\varepsilon = 10^{-4}$.

3 Test Problem

The algorithm suggested in Section 2 is now applied to the solution of the following (synthetic) problem of geoacoustic inversion.

Consider a homogeneous 2D geoacoustic waveguide $\{(r, z)|z > 0\}$ [Jensen et al., 2011], where the sound speed and density depend on the depth z but do not depend on r (the range of the receiver [Jensen et al., 2011]). The layer $0 \leq z \leq h$ represents the water colum, where the sound speed $c(z)$ is a continuous function of depth, while the density is $\rho_w = 1$ g/cm^3. The halfspace $x > h$ is a penetrable (liquid) bottom, where the sound speed

Table 1: Search space and the true values of the unknown parameters.

Parameter	True value	Min value	Max value
c_b, m/s	1700	1550	1850
ρ_b, g/cm^3	1.7	1.1	2
R, m	7000	6850	7150

and density are c_b and ρ_b respectively. In this study, the value c_b, ρ_b are considered unknown media parameters (i.e. we have no data about the acoustic properties of the bottom). Our goal is to estimate these quantities by analyzing some acoustic data.

Assume that a pulse acoustic signal is emitted by a point source located at $R = 0, z = z_s$ and received by a hydrophone at the distance R from the source. The exact value of the receiver range R is also considered unknown. Performing some time-frequency analysis of the received signal (see e.g. [Bonnel & Chapman, 2011, Bonnel et al., 2012, Petrov, 2014, Zaikin & Petrov, 2016]), we can obtain the arrival times of different modal components of the signal $\tau^{\exp}(f, m)$ as the functions of frequency f (here m is the mode number [Jensen et al., 2011, Bonnel & Chapman, 2011]).

Using the mode theory of sound propagation in shallow-water waveguides we can also compute the theoretical arrival times $\tau^{\mathrm{th}}(f, m)$ for any given set $\mathbf{A} = (c_b, \rho_b, R)$ of the values of the unknown parameters. Introducing a fitness that quantifies the agreement between the theoretical and experimental arrival times

$$E(\mathbf{A}) = \frac{\sum_{m=1}^{M} \sum_{n=1}^{Nm} \left|\tau_m^{\mathrm{th}}(f_n, \mathbf{A}) - \tau_m^{\exp}(f_n)\right|^2}{\sum_{m=1}^{M} N_m}, \tag{1}$$

we turn the geoacoustic inversion problem into a global minimization problem (note that in practice we use the discrete set of frequencies $\{f_1, f_2 \ldots, f_{N_m}\}$ and take into account M trapped modes [Jensen et al., 2011]). Indeed, the vector $\mathbf{A}_{\min}$ which denotes the solution of the problem $E(\mathbf{A}) \to \min$ for certain search space, contains such values of unknown parameters that make the best match with experimental data.

For our test case we simulated the signal at the receiver using the mode theory, then we applied a warping-based algorithm to the resulting time series, and subsequently used the resulting syntetic dispersion data $\tau^{\exp}(f, m)$ as the input for our inversion algorithm. The sound speed profile in the water column and the pulse signal emitted by the source was the same as in [Zaikin & Petrov, 2017]. The true values of the unknown parameters and the search space (a cuboid) are presented in Table 1.

4 Computational Experiments

For many classes of computational problems, graphics processing units (GPUs) show significant speedup over central processing units (CPUs). In particular, GPUs can help to solve global optimization problems (e.g., see [Barkalov & Gergel, 2016]). However, to actually obtain this speed-up, the implementation of an algorithm should be thoroughly adapted for GPU architecture [Bulavintsev, 2015]. Fortunately, in our approach to geoacoustic inversion problem the base algorithms are well fit for a GPU. The complete process of solving the geoacoustic inversion problem could be represented as a hierarchy of the subproblems solved by the corresponding sub-algorithms (see Table 2).

Table 2: Hierarchy of algorithms used in the process of solving the geoacoustic inversion problem.

Level	Problem	Algorithm
1	Search for lowest residue point	ASN Search
2	Calc. residue for a point	Calc. residue over individual frequencies
3	Calc. modal group velocities for a single frequency	Numerical differentiation
4	Sturm-Liouville problem on a mesh	Calc. eigenvalues of a tridiagonal symmetric matrix (bisection algorithm [Demmel, 1997])

In our computational experiments we considered the geoacoustic inversion problem described in Section 3. So, the ASN search was applied for minimizing the objective function (1).

The GPU programming model is based on data parallelism. To achieve peak efficiency, modern GPUs should simultaneously run about 10 000 computational threads [Corporation, 2017, Bulavintsev, 2015]. In the GPU-based implementation of our search procedure, we exploit data parallelism found in the parallel calculation of

Table 3: Algorithm performance for various platforms and floating-point precisions formats.

Platform	Precision	Residue	Time, s.	Num. points	points/s.
CPU (alglib [Bochkanov & Bystritsky, 2016])	FP64	0.0128206	4678	800	0.1710
CPU	FP64	0.0132451	1212	639	0.5272
	FP32	0.0209403	487	517	1.0616
GPU	FP64	0.0131659	109	717	6.5779
	FP32	0.0215154	46	701	15.2391

Table 4: Mixed-mode algorithm performance.

Stage I			Stage II				
Platform	Final residue	Time, s.	Platform	Initial residue	Final residue	Time, s.	Total time, s.
GPU (FP32)	0.0215154	46	CPU (alglib)	0.0137885	0.0130473	1446	1492
			CPU	0.0133772	0.0131325	1099	1145
			GPU	0.0134315	0.0130635	81	127

modal group velocities for a single point. Level 1 sub-algorithm executes on the CPU, while level 2-4 sub-algorithms execute on the GPU (Table 2). A typical residue calculation for a single search space point requires calculation of modal group velocities for thousands of frequencies. Thus, data parallelism exposed in this way should be enough to efficiently exploit modern GPUs.

Traditionally, geoacoustics computations are performed in double precision floating point arithmetics (FP64). However, modern consumer-grade GPUs suffer a great (for some devices, up to 32 times [Corporation, 2017]) drop in performance when using FP64 instructions instead of their single-precision (FP32) counterparts. This fact led us to investigate the consequences of changing the algorithm to FP32. We considered the model problem of geoacoustic inversion with 3 parameters (R, ρ_b, c_b) (see Section 3). The results are presented in Table 3. All experiments were conducted on the Intel Core i7 930 CPU and the Nvidia GTX 1050 GPU, with the use of CUDA 8.0 SDK and GCC 5.4.0 compiler. The NVCC compiler was forced to produce native Compute Capability 6.1 code. Both NVCC and GCC compilers were run with -O3 optimization flag.

The most computationally intensive part of our algorithm is the bisection sub-algorithm at level no. 4 in Table 2. We implemented it as described in [Demmel, 1997]) and [Corporation, 2017], producing single- and double-precision versions, algorithmically identical on both GPU and CPU platforms. For reference, we provide the double-precision version based on the AlgLib library [Bochkanov & Bystritsky, 2016], that we used in our previous work [Zaikin et al., press]. The AlgLib library includes a state-of-the-art implementation of the bisection algorithm, thoroughly tuned to produce the most accurate results possible with the modern CPUs floating point units (FPUs). Our CPU-based implementation of bisection algorithm can not boast such accuracy. However, it is notably faster than AlgLib due to its simplicity. That is why, our and AlgLib-based algorithm's results are different. The discrepancy between the outputs (residue) of the same algorithm on the CPU and GPU is the result of the different implementations of floating-point units on these platforms.

Table 3 clearly indicates the superiority of the GPU performance, even in FP64. The FP32 mode of the GPU is 150 times faster than the FP64 mode, though the former's final residue (0.0215154) is much worse than the latter one's (0.0131284). However, recomputation of this final point in FP64 gives the result (0.0134315, see Table 4) that is on par with that of the FP64 CPU implementation.

This leads us to suggest the mixed-mode, two-stage algorithm, that uses the "coarse" FP32 computation mode during Stage I, and then refines its result in Stage II by resuming the search in the "fine" FP64 computation mode. The performance of this algorithm is presented in Table 4. For reference, the "Initial residue" column of Table 4 shows residue of the best point found by Stage I recomputed by various Stage II algorithms. It is easy to see that the GPU-based FP32 algorithm for Stage I is best complemented for Stage II by the GPU-based FP64 algorithm. The total runtime of this mixed-mode algorithm is 127 seconds. This is faster than runtime of the pure GPU-based FP64 algorithm (777 seconds), and its final residue (0.0130635) is almost equal to that of the pure CPU-based FP64 (alglib [Bochkanov & Bystritsky, 2016]) algorithm (0.0130471). The coordinates of the points found in our experiments are available in Table 5.

The FP64 performance of our GPU-based algorithm is only 2,5 times lower than the performance of our FP32 algorithm on the same GPU (see Table 3). However, according to [Corporation, 2017] this performance drop should be at least 4x times for the GTX1050 GPU used in our experiments. This means that the GPU

Table 5: Coordinates of the points found by the examined search algorithms.

Platform	Precision	Residue	R	ρ_b	c_b
Single-Stage algorithm					
CPU alglib	FP64	0.0128206	7016.07	1.74473	1736.05
CPU	FP64	0.0132451	7020.62	1.68634	1734.21
	FP32	0.0209403	7018.21	1.44446	1742.41
GPU	FP64	0.0131659	7019.37	1.70663	1735.22
	FP32	0.0215154	7022.99	1.71911	1720.55
Two-Stage algorithm					
GPU (FP32) ->CPU alglib (FP64)		0.0130473	7015.68	1.70976	1734.97
GPU (FP32) ->CPU (FP64)		0.0131325	7020.25	1.71881	1731.43
GPU (FP32) ->GPU (FP64)		0.0130635	7019.3	1.76232	1730.56

performance is likely to be bottlenecked by suboptimal multiprocessor occupancy [Corporation, 2017] and/or excessive memory transfers, rather than the FP64 FPUs of the device. The higher GPU performance we perceived in our previous experiments with the parallelization scheme based on the simultaneous computing of residues for many search space points supports this theory. This means that a thorough optimization of the GPU code could increase the GPU performance of our geoacoustic inversion algorithm at least 2-3 times.

The source code of our application is available online[1].

5 Conclusions

In the present paper, we suggested a modification of a coordinate descent algorithm. This algorithm was implemented in two versions. The first one is designed for launching on a CPU. In the second one, the algorithm itself works on a CPU, while the calculation of the objective function is performed on a GPU. Both versions were applied for solving one synthetic problem of geoacoustic inversion. Being little less accurate (in the sense of found solutions), the GPU-based version turned out to be much faster, than the CPU one.

Acknowledgements

This study was partially supported by the Council for Grants of the President of the Russian Federation (grants No. MK-2262.2017.5, No. NSh-8081.2016.9, No. NSh-8860.2016.1), the Russian Foundation for Basic research (grants No. 16-05-01074_a, No. 16-07-00155_a and No. 17-07-00510_a), Presidium of RAS programs I.33, I.5, and the POI FEB RAS Program "Nonlinear dynamical processes in the ocean and atmosphere".

References

[Barkalov & Gergel, 2016] Barkalov, K. and Gergel, V. (2016). Parallel global optimization on GPU. *Journal of Global Optimization*, 66(1):3–20.

[Bochkanov & Bystritsky, 2016] Bochkanov, S. and Bystritsky, V. (2016). Alglib - a cross-platform numerical analysis and data processing library. *ALGLIB Project. Novgorod, Russia.*

[Bonnel & Chapman, 2011] Bonnel, J. and Chapman, N. R. (2011). Geoacoustic inversion in a dispersive waveguide using warping operators. *The Journal of the Acoustical Society of America*, 130(2):EL101–EL107.

[Bonnel et al., 2013] Bonnel, J., Dosso, S. E., and Chapman, N. R. (2013). Bayesian geoacoustic inversion of single hydrophone light bulb data using warping dispersion analysis. *The Journal of the Acoustical Society of America*, 134(1):120–130.

[Bonnel et al., 2012] Bonnel, J., Gervaise, C., Nicolas, B., and Mars, J. I. (2012). Single-receiver geoacoustic inversion using modal reversal. *The Journal of the Acoustical Society of America*, 131(1):119–128.

[Bulavintsev, 2015] Bulavintsev, V. (2015). An evaluation of CPU vs. GPU performance of some combinatorial algorithms for cryptoanalysis. *Vestnik Yuzhno-Ural'skogo Gosudarstvennogo Universiteta. Seriya "Vychislitel-naya Matematika i Informatika"*, 4(3):67–84.

[1] https://github.com/mposypkin/acouwater/tree/f_gpu

[Corporation, 2017] Corporation, N. (2017). Cuda c programming guide.

[Demmel, 1997] Demmel, J. W. (1997). *Applied numerical linear algebra.* SIAM.

[Evtushenko et al., 2016] Evtushenko, Y. G., Lurie, S. A., Posypkin, M. A., and Solyaev, Y. O. (2016). Application of optimization methods for finding equilibrium states of two-dimensional crystals. *Computational Mathematics and Mathematical Physics*, 56(12):2001–2010.

[Gornov et al., 2016] Gornov, A. Y., Zarodnyuk, T. S., Finkelstein, E. A., and Anikin, A. S. (2016). The method of uniform monotonous approximation of the reachable set border for a controllable system. *Journal of Global Optimization*, 66(1):53–64.

[Hooke & Jeeves, 1961] Hooke, R. and Jeeves, T. A. (1961). " direct search" solution of numerical and statistical problems. *J. ACM*, 8(2):212–229.

[Jensen et al., 2011] Jensen, F. B., Kuperman, W. A., Porter, M. B., and Schmidt, H. (2011). *Computational ocean acoustics.* Springer, New-York et al.

[Katsnelson et al., 2012] Katsnelson, B. G., Petnikov, V. G., and Lynch, J. (2012). *Fundamentals of Shallow Water Acoustics.* Springer US, New-York et al.

[Petrov, 2014] Petrov, P. S. (2014). A method for single-hydrophone geoacoustic inversion based on the modal group velocities estimation: Application to a waveguide with inhomogeneous bottom relief. In *Proceedings of the International Conference Days on Diffraction 2014*, pages 186–191.

[Zaikin & Petrov, 2016] Zaikin, O. and Petrov, P. (2016). Algorithm of reconstruction of the sound speed profile in a shallow-water geoacoustic waveguide from modal dispersion data. *Optoelectronics, Instrumentation and Data Processing*, 52(3):259–265.

[Zaikin & Petrov, 2017] Zaikin, O. and Petrov, P. (2017). Application of iterative hill climbing to the sound speed profile inversion in underwater acoustics. In *Proc. of The Fifth International Workshop on Mathematical Models and their Application*, volume 173 of *IOP Conf. Series: Materials Science and Engineering*, pages 1–8.

[Zaikin et al., press] Zaikin, O., Petrov, P., Posypkin, M., Bulavintsev, V., and Kurochkin, I. (in press). Using volunteer computing for sound speed profile estimation in underwater acoustics. In *Proc. of The Third International conference on BOINC: Fundamental and Applied Science and Technology.*