ACCELERATION OF FULL WAVEFORM INVERSION BASED ON RANDOM SUPER-SHOT AND GPU

BAOLI WANG, JINGHUAI GAO and WENCHAO CHEN

Institute of Wave and Information, School of Electronics and Information Engineering, Xi'an Jiaotong University, 28 Xiannig West Road, Xi'an, Shaanxi 710049, P.R. China. pooly1981@163.com

(Received December 13, 2010; revised version accepted July 17, 2011)

ABSTRACT

Wang, B, Gao, J. and Chen, W., 2011. Acceleration of Full Waveform Inversion based on Random Super-shot and GPU. *Journal of Seismic Exploration*, 20: 331-346.

The full waveform inversion (FWI) method is more and more important in the field of seismic exploration, which can be used to estimate the subsurface model by matching the simulated data with the acquired records. However, due to extreme computer-intensive time, FWI cannot be applied to real data widely. This paper describes a new strategy for the acceleration of FWI algorithm. Firstly of all, by generating the super-shots, the numbers of shots are reduced effectively, and thus, the computational cost is also reduced. Besides, the super-shots are regenerated randomly for suppressing the crosstalk noises and the artifacts caused by the summation during iterations. Furthermore, FWI is accelerated by the introduction of the hardware of graphics processing units (GPU) and the parallel programming based on GPU. Then, the synthetic records forwarded by finite-difference method in the time domain based on the Marmousi velocity model are employed to examine the proposed algorithm. Test results show that, as far as a same inversion level is concerned, our algorithm reduces the computational cost by 80 times than the conventional FWI method. Result also indicates that the super-shots regenerating randomly can suppress the crosstalk noise and the artifacts caused by summation than conventional methods. Finally, a marine real dataset is used to prove the superiority of RSSFWI based on GPU.

KEY WORDS: full waveform inversion, GPU, CUDA, velocity analysis, super-shot.

INTRODUCTION

Generally, it is very difficult and time-consuming to directly calculate the gradient of objective function of the full waveform inversion (FWI). Therefore, many good works in FWI have been done by the geophysical researchers. To reduce the computational cost in the time-domain inversion, Tarantola (1984) calculates the gradient of objective function of FWI with the back-propagation algorithm. Furthermore, Pratt (1998) presents the FWI method with the back-propagation algorithm in the frequency domain. However, as far as the FWI methods in both time-domain and frequency-domain are concerned, they are extremely computer intensive because each iteration step of FWI always needs to forward several propagation procedures which are the very time-consuming works.

Therefore, many geophysical researchers have paid their attenuation to reduce the computational cost of FWI. For example, Vigh (2008) introduces the plane-wave gathers in FWI, which can reduce computational time than the traditional shot-gathers. Also, for FWI in frequency-domain, Sirgue and Pratt (2004) define an inversion strategy which only considers a few frequencies. But these methods are all developed for an individual source, so the computational cost is proportional to the number of shots. Unfortunately, the number of shots is typically large for seismic surveys.

It has been tried to accelerate the FWI by the contraction in the number of shots. Mora (1987) reduces the number of shots in FWI through generating some super-shots. However, some cross-talk noises and artifacts always produce together with the inversion result because the super-shots are summed by several individual shots which are not changed between iterations. Moreover, encoded source sums technology is a good idea to reduce the number of shots for seismic acquisition (Neelamani, 2008a) and seismic data processing (Romero et al., 2000). Krebs et al. (2009) present the phase encoding algorithm for two-dimensional FWI in time-domain, and Ben-Hadj-Ali et al. (2009) give the phase encoding technology for three-dimensional FWI in the frequency-domain. These two encoding technologies reduce the computational cost based on super-shot technology.

Except for the algorithm advance in the software, the acceleration of FWI may be achieved by hardware advance such as the graphics processing units (GPU). In recent years, the computing capacity of GPU has been improved enormously. GPU has more processing units and higher memory bandwidth than the central processing units (CPU). As far as the single-precision floating-point is concerned, the computing capacity of GPU can reach 1.35T flops, which means 1.35 trillion floating-point operations per second. Therefore, the introduction of GPU may be a good idea for the acceleration of FWI.

In this paper, we focus on the fast implementation of FWI with the GPU. At the first, in the second part, we will demonstrate the super-shot technique, by which the computational cost can be reduced. Different from the others, the shots, which are assembled to being a super-shot, are selected randomly between iterations. Comparing conventional fixed manner, this selecting manner can help us suppressing the crosstalk noise and the artifacts caused by summation. By using this method, we can achieve efficiency gains by many times for FWI. Secondly, the hardware of GPU and the parallel programming for GPU are described in the third part. Then we employ the Marmousi velocity model to examine our FWI algorithm in the fourth part. And finally, the conclusions are given.

METHOD

It is well known that FWI is generally based on minimizing a objective function that measures the difference between the simulated and the acquired data. The objective function of multi-shot records can be defined as the l_2-norm of residuals between them, which is written as (Gauthier, 1986)

$$\min_{m} S(m) = \sum_{s=1}^{Ns} \int_{0}^{T} dt \sum_{r=1}^{Ng} [tp(\mathbf{x}_r,t;\mathbf{x}_s) - \tilde{p}(\mathbf{x}_r,t;\mathbf{x}_s)]^2 \quad , \tag{1}$$

where:

$S(m)$ = the error energy function,

$\min_{m} S(m)$ = the objective function,

m = the model,

Ns = the number of sources,

Ng = the number of receivers,

T = the total time of data,

$p(\mathbf{x}_r,t;\mathbf{x}_s)$ = the simulated data,

$\tilde{p}(\mathbf{x}_r,t;\mathbf{x}_s)$ = the acquired data.

From the above equation, we can find that the main job of FWI is just the simulation, which is very time-consuming and proportional to the number of shots. So, it is a good idea to reduce the number of simulating shots in each iteration by using super-shot technology (Mora, 1987), which is a shot assemblage. In order to reduce computational cost, all shots are randomly divided into several groups with the same number of shots, and then one super-shot can be derived by stacking the shot gathers in a same group as follows

$$\tilde{P}(\mathbf{x}_r,t;X) = \sum_{i=1}^{M} \tilde{p}(\mathbf{x}_r,t;\mathbf{x}_i) \quad , \tag{2}$$

where $\tilde{P}$ is a super-shot composed of M individual shots selected randomly in each iteration. In this way, for seismic records with individual shots, the number of super-shots is N = Ns/M. Now, instead of simulations of individual shot, we only need to perform N simulations of super-shots. Evidently, the computational cost can be reduced by times. Now, eq. (1) can be rewritten as

$$S(m) = \sum_{s=1}^{N} \int_0^T dt \sum_{r=1}^{Ng} [P(\mathbf{x}_r,t;X_s) - \tilde{P}(\mathbf{x}_r,t;X_s)]^2 \quad , \tag{3}$$

where N is the number of super-shots, and P is simulated super-shots which has M individual sources being equal to the sources of $\tilde{P}$. For each super-shot, it is necessary to satisfy with following wave equation

$$(1/v^2)(\partial^2P/\partial t^2) = (\partial^2P/\partial x^2) + (\partial^2P/\partial z^2) + f(t)\sum_{s=1}^{M} \delta(\mathbf{x} - \mathbf{x}_s) \quad , \tag{4}$$

where f(t) is the source function, and $\mathbf{X}_{s,s=1,2,\ldots,M}$ are the locations of the M individual shots.

Finally, the gradient of velocity can be calculated easily by

$$\gamma = (2/v^3) \int_0^T \lambda(\partial^2P/\partial t^2) \quad , \tag{5}$$

where λ denotes the back-propagated residual wavefield. And the velocity can be updated according to

$$v_{n+1} = v_n + a_n\gamma_n \quad , \tag{6}$$

where a_n denotes the optimal step-length and can be obtain by performing a line search , for which a linear approximation of the forward problem is used (Gauthier et al., 1986; Pica et al., 1990).

In this way, we can simulate super-shots, and then matches them with the real super-shots generated by observed data. For convenience, we call this matching procedure as the random super shot FWI (RSSFWI) for short in this paper.

In consideration of computational efficiency and convergence rate, we choose the conjugate gradient (CG) method (Pratt et al., 1998) as our optimization approach. Although the full Newton method and the Gauss-Newton method have faster convergence rate and more accurate inverted velocity imaging, the second-order term of full Newton method and the inverse of the Hessian matrix are more time-consuming than the conjugate gradient method, which has faster convergence rate than the steepest descent method and small storage requirement. In iteration, the partial derivatives for the velocity gradient are efficiently calculated by using back-propagation technique (Tarantola, 1984). Finally, the inversion algorithm of RSSFWI can be described as follows:

Randomly divides all shots into N groups, and then generates N super-shots by summing the shots;

1) Computes the forward-propagating wavefield by doing N super-shots simulations, and calculates the residual wavefield;
2) Back-propagates the data residuals;
3) Calculates the gradient from the cross-correlation between the back-propagated residual wavefield and the forward-propagated wavefield;
4) Forms the preconditioned gradient;
5) Forms the conjugate gradient;
6) Performs a line search to obtain the step-length;
7) Updates the velocity model;
8) Loops from step 1 to step 8.

To distinguish, the procedure that the super-shots are generated with the fixed manner of group division, may be called the super shot FWI (SSFWI). The RSSFWI changes randomly the places of every super-shot during iteration, therefore the crosstalk noise and the artifacts are incoherent from iteration to iteration and therefore these noises can be suppressed after several iterations.

ACCELERATION WITH GPU AND CUDA

Review of GPU and CUDA

On the one hand, driven by the insatiable market demand for real-time and high-definition 3D graphics, the programmable GPU has evolved into highly parallel, multi-threaded, and multi-core processors with tremendous computational horsepower and very high memory bandwidth.

On the other hand, NVIDIA provides a convenient development environment - the Computer Unified Device Architecture (CUDA) (NVIDIA, 2011), which can help us with GPU solve many complex computational problems in a more efficient way than on a CPU. To simplify program's development, one can use the CUDA C-compiler to combine CPU and GPU code into one continuous program file. Simple additions to the C-programs tell the CUDA compiler that which functions compile for CPU or GPU. Then the program is compiled with the CUDA compiler for the GPU, and the CPU host code is compiled with the developer's standard C compiler. CUDA C extends C by allowing the programmer to define C functions, called kernels, that, when called, are executed $M * N$ times in parallel by M different CUDA blocks, each block has N different CUDA threads, as opposed to only once like regular C functions. The number M of blocks and the number N of threads per block, given by programmer when the kernel is called, affect the program's speed. Generally, the bigger the value of M is, the faster the program accelerates. Actually, M and N are limited by the hardware specifications of GPU. Finally, we test our method and parallel program by applying to the synthetic seismogram of the Marmousi model. This test shows that RSSFWI can also obtain a good inverted velocity but needs less computational cost.

Moreover, GPU has several types of memories, and each type has its own strong points and shortcomings. For example, the global memory has more storage space, but inefficient transmission speed; the shared memory is much faster than the global memory, but its space is smaller. In this study, we use both of them to accelerate the program.

Implementation Details

It is well known that seismic wave modeling is the most time-consuming flow during FWI. Fortunately, modeling using finite-difference in the time domain is well suited to implementation using CUDA on GPU because each computational grid is highly parallel. We test our parallel program implemented with the CUDA on GPU, and then compare its computational cost with CPU.

Table 1 and Table 2 list the specifications of the CPU and GPU in this study. Forward-modeling one shot profiling only need 1.9 s for GPU, while 155 s for CPU, in which the acceleration ratio is about 81 times, as shown in Table 3. Meanwhile, avoiding frequent data communications between CPU and GPU, which can seriously affect the computational efficiency, all computing tasks should be performed on GPU, as specified by the pseudo codes in Fig. 1.

Table 1. CPU specifications.

CPU	Intel(R) Core(TM)2 Quad Q6600 2.40 GHz
Memory	4GB

Table 2. GPU specifications.

GPU Model	NVIDIA GeForce GTX 480
Global Memory	1535MB DDR5
Multiprocessors	15
Threads Per Block	1024
Single-precision Float	1.33e+06 M flop/s
Double-precision Float	1.68e+05 M flop/s
Compute Capability	2.0

Table 3. Comparison of computational cost of forward-modeling a synthetic wavefield between using CPU and GPU.

Model	Marmousi
Grid	741 * 2301
Numbers of time step	7500
Cost on CPU	155 s
Cost on GPU	1.9 s

Fast Full Waveform Inversion of Multi-shot Seismic Data

```
// Read parameters and data; Allocate memory on CPU and GPU;
cudaMemcpy(d_V, h_V)    //upload velocity to Device from Host
Loop over iterations
   <<<d_Vgrad=0>>>    //initialization of gradient
   <<<generate several super-shots randomly>>>
      Loop over super-shots
      <<< Computer the forward-propagating wavefield >>>
      <<< Calculate the residual wavefield >>>
      <<< Back-propagate the data residuals >>>
      <<< Calculate the gradient >>>
      <<< Form the preconditioned gradient >>>
      <<< Form the conjugate gradient >>>
      <<<Simulating to obtain the step-length >>>
      <<< Update the velocity model >>>
      End of loop over super-shots
End of loop over iterations
cudaMemcpy(h_V, d_V)    //download velocity to Host from Device
```

Fig. 1. Pseudo code of the GPU-based RSSFWI. The sign "<<<>>>" denotes the kernel execution on GPU.

NUMERICAL EXAMPLE

To examine the RSSFWI method, we employ the Marmousi model as illustrated in Fig. 2, which has a grid of dimensions 741 $\times$ 2301 with a sampling step size of 4 m for Δx and Δz. For simplicity, we generate the synthetic seismograms using the equation 4 by FD method with the Clayton-Engquist boundary conditions except top (free surface). For the Marmousi model, there are 20 shots in 400-m space and 360 receivers in 25-m space located on the surface. With the source signature simulated by a 10 Hz Ricker wavelet, we generate 3-s records with sampling interval of 0.4 ms. The initial velocity model for FWI are shown in Fig. 3, which are derived by smoothing the Marmousi velocity model with a Gaussian filter with correlation length of 100 m.

In this test, we generate several super-shots by $M = 4$ in eq. (2). Fig. 4 shows a super-shot gather, whose sources are located at 4800 m, 5200 m, 5600 m, and 6000 m. For comparison, the velocity is inverted by three methods. Figs. 5-7 shows the velocity images inverted by the conventional single shot FWI, the SSFWI, and the RSSFWI with $M = 4$, respectively. Each of them is

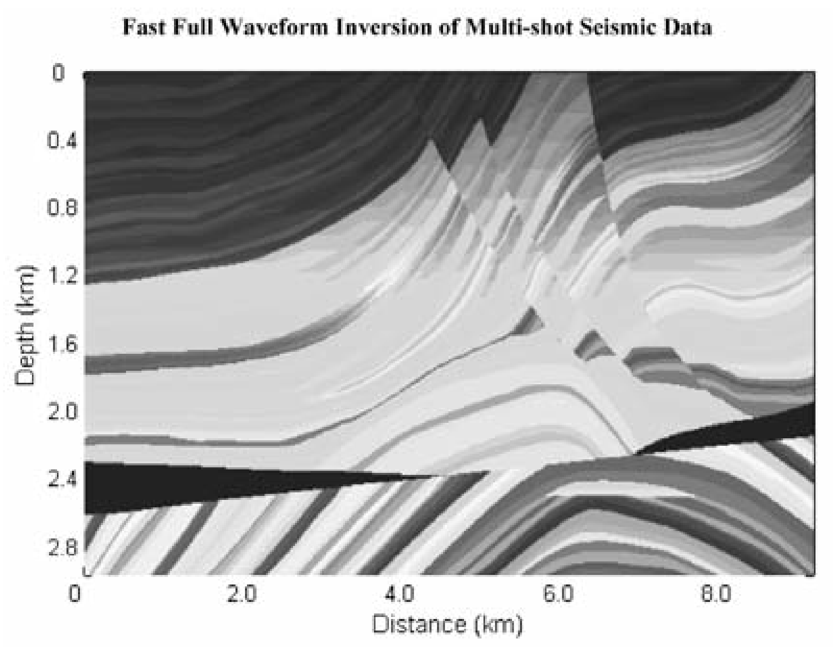

Fig. 2. The true Marmousi velocity model, which has a grid of dimensions 741 × 2301 with a sampling step size of 4 m for Δx and Δz.

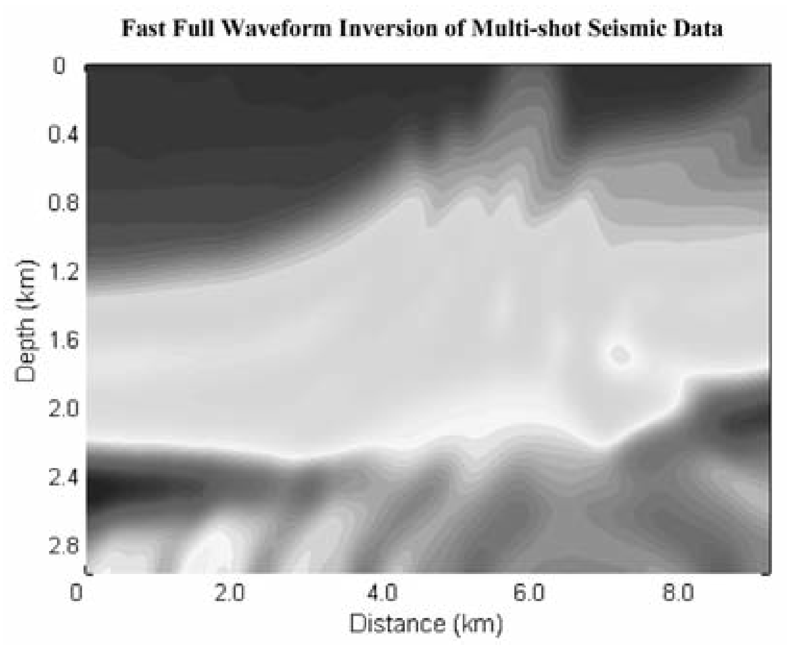

Fig. 3. The starting velocity model obtained by smoothing the real Marmousi velocity model using Gaussian filter with correlation length 100 m.

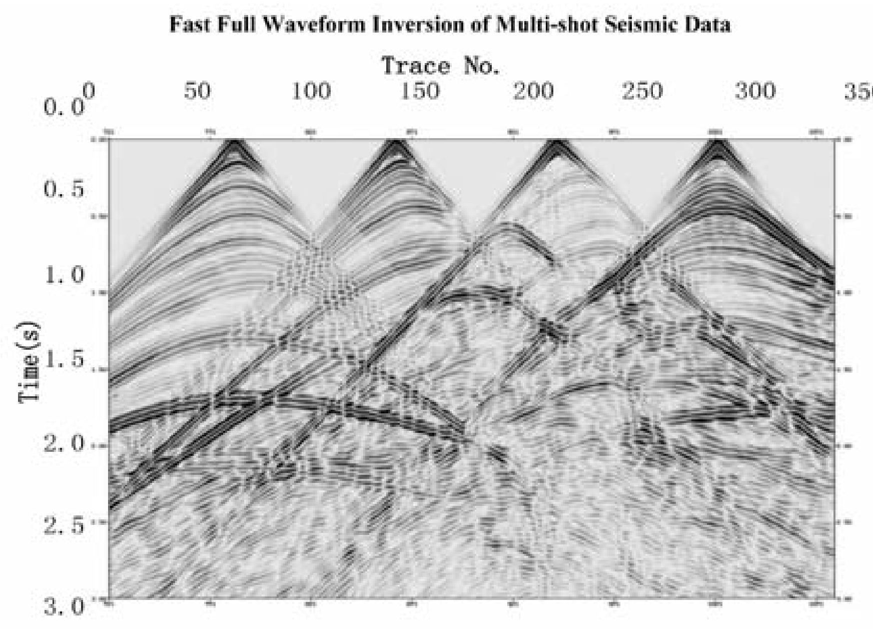

Fig. 4. A super-shot gather, whose sources are located at 4800 m, 5200 m, 5600 m, 6000 m.

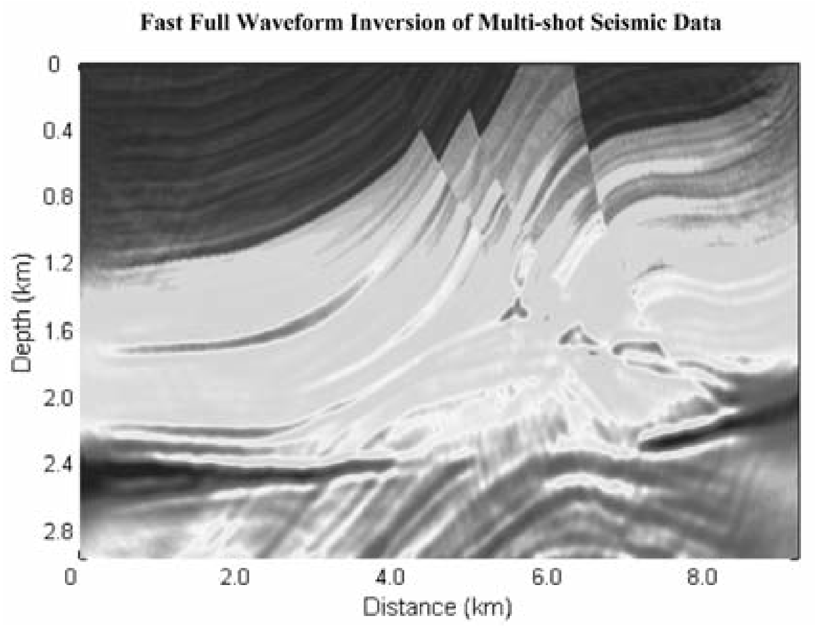

Fig. 5. The velocity model inverted by conventional FWI after 270 iterations (equivalent to M = 1). The code based on GPU costs 10.5 hours.

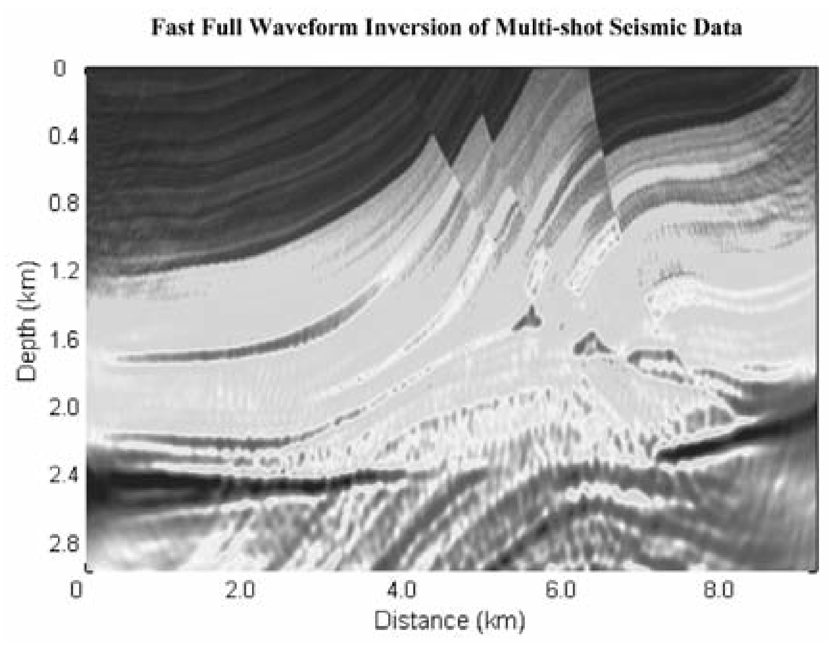

Fig. 6. The velocity model inverted by SSFWI (M = 4) after 270 iterations. The code based on GPU costs 2.6 hours.

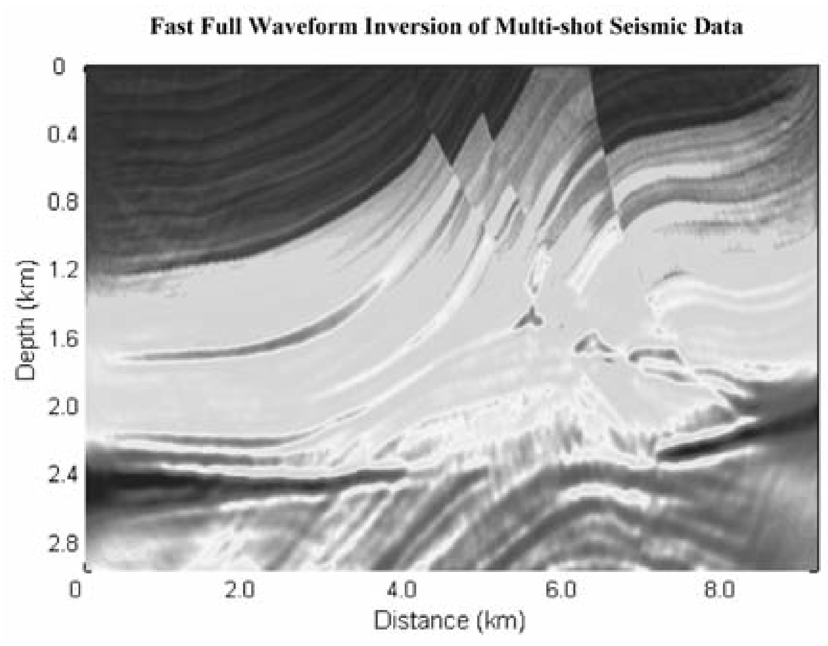

Fig. 7. The velocity model inverted by RSSFWI (M = 4) after 270 iterations. The code based on GPU costs 2.6 hours.

calculated for 270 iterations. The comparison of Fig. 1, Fig. 5, and Fig. 7 shows that the results inverted by the conventional FWI and the RSSFWI are very close to the true velocity model. With M = 4 in eq. (2), the computational efficiency of the RSSFWI is 4 times than that of the conventional FWI. At the same time, comparison between Figs. 6 and 7 shows that the results of SSFWI involves much noise, but the RSSFWI can effectively suppress the crosstalk noise due to the fixed summation manner for the generation of a super-shot.

For this test, the code based on GPU of RSSFWI takes 140 seconds per iteration at M = 1, while only 35 seconds per iteration at M = 4, which means the computational cost has been reduced by more than 75%. For illustration, the normalized cost function versus inversion-time is given in Fig. 8.

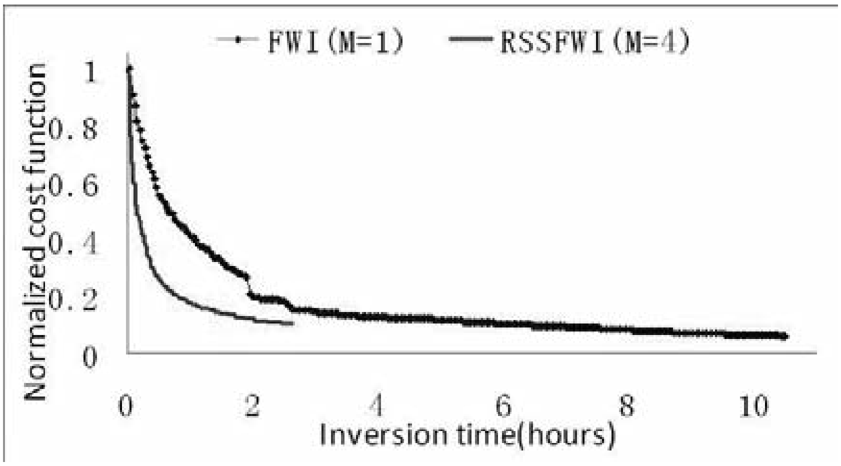

Fig. 8. Cost function versus computational time for SSFWI (M = 4) and conventional SSFWI (M = 1, equivalent to conventional FWI).

FIELD DATASET

In this section, we apply our method to a 2D marine field dataset that has 20 shot records with 50 m interval, a time-sampling interval of 2 ms, and a trace length of 5 s. The 280 receivers were fixed and spaced at 12.5 m intervals with a near offset of 143.75 m.

Before applying our method, the data are transformed from 3D to 2D format by applying the filter $\sqrt{(i/\omega)}$ in the frequency domain and scaling the data by $\sqrt{t}$ to approximate geometric spreading (Bleistein, 1986; Williamson and

Pratt, 1995; Zhou et al., 1995). Then a Band-pass filter (0-15 Hz) and muting first break are used to pre-process the field data.

Fig. 9a shows the initial velocity model. Fig. 9b shows the reconstructed velocity by our method after 500 iterations, which has more details than the initial model. Figs. 10a and 10b show one original shot gather and corresponding synthetic shot gather from our reconstructed velocity. We can see that there is a good match between them. The normalized misfit-function reduction versus iterations is shown in Fig. 11.

This dataset's inversion test cost 3.6 hours for 500 iterations, about 26 seconds per iteration, which shows the superiority of our method.

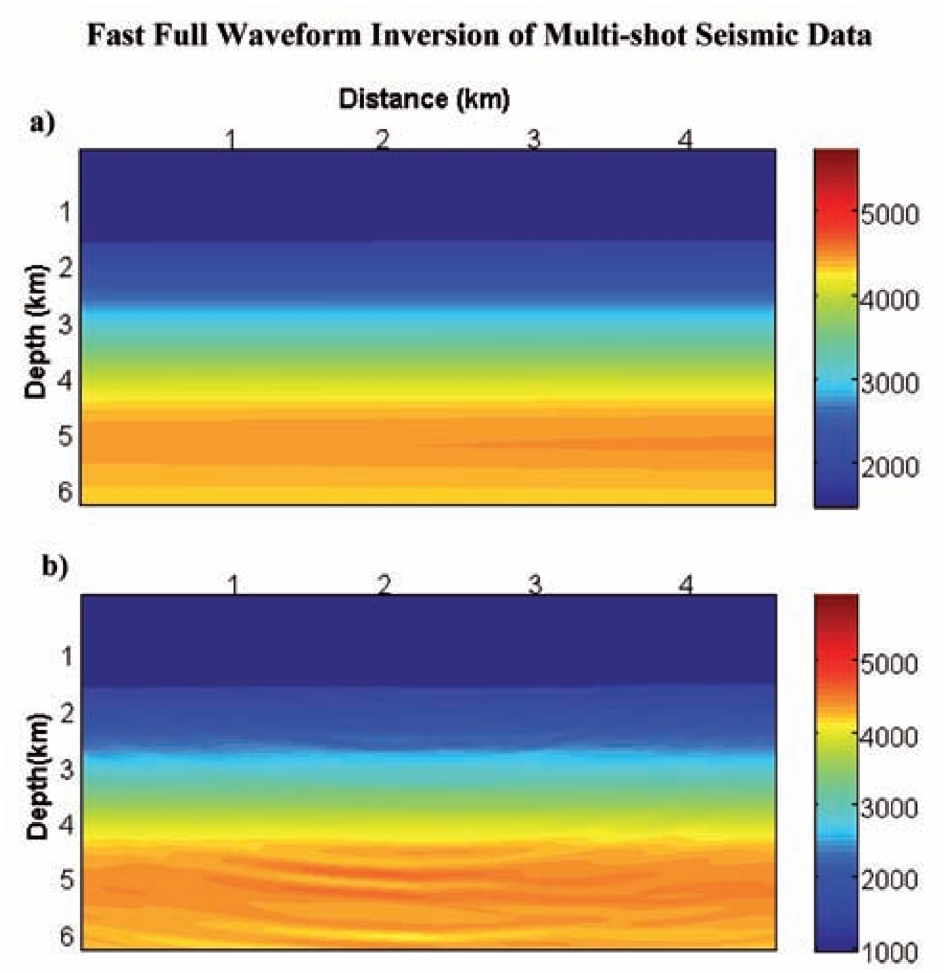

Fig. 9. Field dataset. (a) The initial velocity model. (b) The reconstructed velocity by our method after 500 iterations.

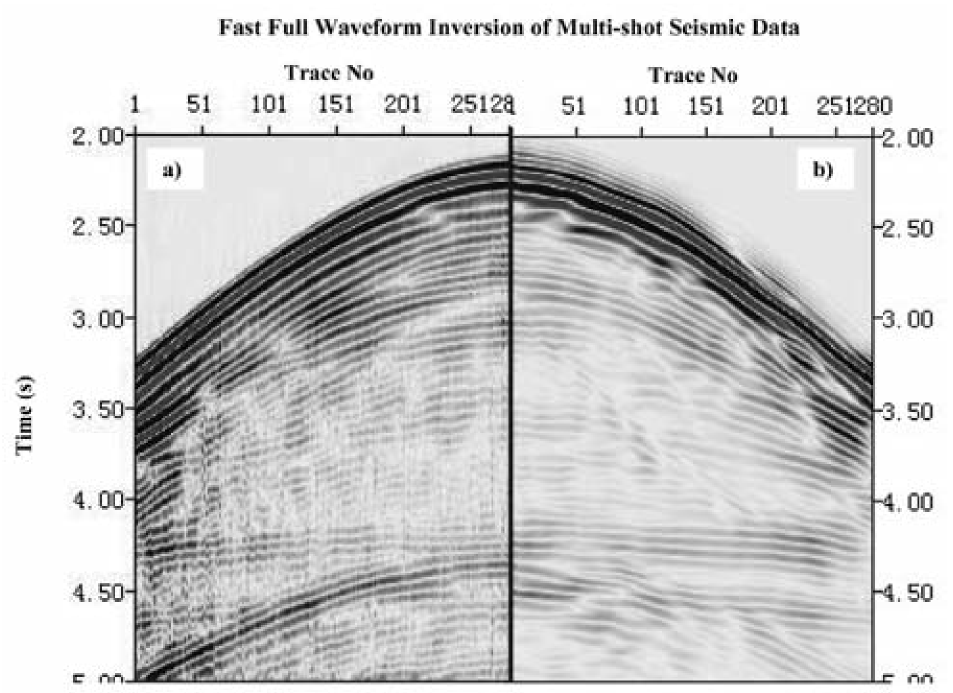

Fig. 10. (a) One original shot gather. (b) Corresponding synthetic shot gather from our reconstructed velocity. We can see that there is a good match between them.

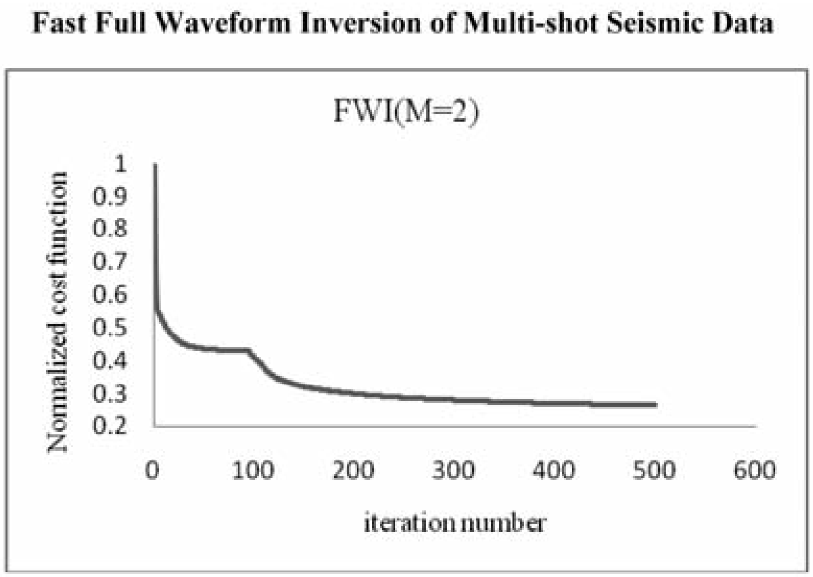

Fig. 11.The normalized misfit-function reduction versus iterations.

CONCLUSIONS

We propose the acceleration of FWI based on the random super-shots and GPU. Tests of the Marmousi model show that the method can increase the computational efficiency of FWI at about 80 times and suppress the crosstalk noise and the artifacts. And a marine field dataset is also used to show the superiority of this method.

Actually, the proposed method accelerates FWI is based both on software and hardware. In software algorithm, generally, the bigger the value of is, the faster the FWI accelerates. The RSSFWI will become the conventional SSFWI when $M = 1$. In hardware, the higher the specification of GPU, the higher the computational efficiency is, meanwhile the more expensive the hardware cost will become. Introduction of GPU increases hardware cost indeed, however, the enhancement in computational efficiency are further more than the rising of hardware cost. In a word, with higher computational efficiency than conventional FWI, the RSSFWI based on GPU may become a strong tool for seismic full wave inversion.

ACKNOWLEDGEMENTS

This work was supported by National Hi-Tech Research and Development Program of China (2006A09A102-11), National Natural Science Funds of China (40730424, 40674064), and Major National Science & Technology Projects (2008ZX05023-005-005, 2008ZX05025-001-009).

REFERENCES

Ben-Hadj-Ali, H., Operto, S. and Virieux, J., 2008. Velocity model building by 3D frequency-domain full-waveform inversion of wide-aperture seismic data. Geophysics, 73: VE101-VE117.

Bleistein, N., 1986. Two-and-one-half dimensional in-plane wave-propagation. Geophys. Prosp., 34: 686-703.

Bunks, C., Saleck, F.M., Zelski, S. and Chaven, G., 1995. Multiscale seismic waveform inversion. Geophysics, 60: 1457-1473.

Gauthier, O., Virieux, J. and Tarantola, A., 1986. Two-dimensional nonlinear inversion of seismic waveforms: Numerical results. Geophysics, 51: 1387-1403.

Krebs, J.R., 2009. Fast full wave seismic inversion suing source encoding. Expanded Abstr., 77th Ann. Internat. SEG Mtg., Houston: 2273-2277.

Mora, P., 1987. Nonlinear two-dimensional elastic inversion of multi-offset seismic data. Geophysics, 52: 1211-1228.

NVIDIA, 2011. NVIDIA CUDA Programming Guide, version 4.0.

NVIDIA, 2011, NVIDIA CUDA C Best Practices Guide, version 4.0.

Pica, A., Diet, J. and Tarantola, A., 1990. Nonlinear inversion of seismic reflection data in a laterally medium. Geophysics, 55: 284-292.

Pratt, R.G., Shin, C. and Hicks, G.J., 1998. Gauss-Newton and full Newton methods in frequency-space seismic waveform inversion. Geophys. J. Internat., 133: 341-362.

Sheen, D.-H., Tuncay, K., Baag, C.-E. and Ortoleva, P.J., 2004. Efficient finite difference calculation of partial derivative seismic wavefield using reciprocity and convolution. Expanded Abstr., 74th Ann. Internat. SEG Mtg. Denver: 1850-1853.

Shin, C., Yoon, K., Marfurt, K.J., Park, K., Yang, D., Lim, H.Y., Chung, S. and Shin, S., 2001. Efficient calculation of a partial-derivative wavefield using reciprocity for seismic imaging and inversion. Geophysics, 66: 1856-1863.

Sirgue, L. and Pratt, R.G., 2004. Efficient waveform inversion and imaging: A strategy for selecting temporal frequencies. Geophysics, 69: 231-248.

Tarantola, A., 1984. Inversion of seismic reflection data in the acoustic approximation. Geophysics, 49: 259-1266.

Tarantola, A., 1986. A strategy for nonlinear elastic inversion of seismic reflection data. Geophysics, 51: 1893-1903.

Tarantola, A., 1987. Inverse Problem Theory: Methods for Data Fitting and Parameter Estimations. Elsevier Science Publishers, Amsterdam.

Vigh, D. and Starr, E.W., 2008. 3D prestack plane-wave, full-waveform inversion. Geophysics, 73: VE135-VE144.

Williamson, P. and Pratt, G., 1995. A critical review of 2.5D acoustic wave modeling procedures. Geophysics, 60: 591-595.

Zhang, J.H., Wang, S.Q. and Yao, Z.X., 2009. Accelerating 3D Fourier migration with graphics processing units. Geophysics, 74: WCA129-WCA139.

Zhou, C., Cai, W., Luo, Y., Schuster, G.T. and Hassanzadeh, S., 1995. Acoustic wave-equation traveltime and waveform inversion of crosshole seismic data. Geophysics, 60: 765-773.