CUDA-Accelerated ODETLAP: A Parallel Lossy Compression Implementation

[Extended Abstract]

Daniel N. Benedetti* W. Randolph Franklin* Wenli Li*

1. INTRODUCTION

We present an implementation of Overdetermined Laplacian Partial Differentiation Equations (ODETLAP) that uses CUDA directly. This lossy compression technique approximates a solution to an overdetermined system of equations in order to reconstruct gridded, correlated data. ODETLAP can be used to compress a dataset or to reconstruct missing data. Parallelism in CUDA provides speed performance improvements over other implementation methods.

ODETLAP is inspired by the Laplacian Partial Differntial Equation, though it is capable of preserving local extrema. Typical compression techniques are limited to viewing data as being one dimensional. The ODETLAP algorithm instead utilizes the autocorrelation of data in multiple dimensions to perform compression. This allows for improved compression of higher dimensional datasets, as can be found in geographical and environmental data[1, 2].

ODETLAP requires the construction and solution approximation of a sparse over-determined linear system of equations. As a result, the algorithm is quite compute-intensive. Parallelization techniques offer means of improving performance of this algorithm. The CUDA libraries, Thrust and CUSP, will be utilized in this parallel implementation of ODETLAP The Thrust library provides means for construction of the overdetermined system matrix, while the CUSP library contains solvers for such systems.

2. BACKGROUND

Compression techniques can be either lossy or lossless. Lossy schemes cannot precisely reconstruct the original data, but do provide greater compression ratios than lossless schemes. Lossy compression algorithms utilize correlation within the dataset in order eliminate nonessential data. Compression techniques such as JPEG2000, [4], and SPIHT, [5], have been used for higher dimensional data compression. These techniques, unfortunately, do not utilize correlation in more than two dimensions.

The earliest incarnation of ODETLAP sought to both lossily compress 2D terrain data and reconstruct surfaces using scattered elevation points [2]. Performance benefits of utilizing parallelism and the handling of very large datasets were explored by means of parallel ODETLAP runs on a supercomputer [3]. ODETLAP was shown to efficiently compress three dimensional oceanographic data by utilizing autocorrelation in all three dimensions [2]. As a means of improving performance of the compute-intensive ODETLAP algorithm, CUDA is utilized to efficiently compress five dimensional data in parallel [1, 2].

3. ODETLAP STRUCTURE

For a two dimensional dataset, the Laplacian PDE $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$ is applied and an averaging equation is added to the system for each non-boundary point within the dataset, where each point is the average of its four immediately adjacent neighboring points. A number of points are selected to be "known" and equations are added to the system, where each known point is set equal to its actual value from the original data. The solution is approximated, and known points are iteratively added until a stopping criteria is met. A smoothness coefficient is specified to determine the weighting of the known point equations compared to the averaging equations.

The structure for the ODETLAP implementation varies little when expanding into higher dimensions. For example, the three dimensional Laplacian PDE $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = 0$, which means that each non-boundary point will be the average of its six immediately adjacent neighbors. The iterative selection of worst error points and approximating the solution for the new system does not vary for higher dimensioned data.

4. IMPLEMENTATION

The CUDA-implementation utilizes the Thrust and CUSP libraries. Thrust is a C++ template library, providing a higher level interface for CUDA without negatively affecting performance. The Thrust library provides STL-compliant device vectors and associated operations for CUDA-enabled GPUs. Most general functionality of the ODETLAP implementation utilizes Thrust, including system matrix construction and error calculation.

CUSP is a parallel sparse matrix linear algebra library, based on Thrust. Several sparse matrix formats are available in CUSP, with the Coordinate format utilized for the A matrix in the overdetermined system in ODETLAP. CUSP also

*Dept. of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, 110 8th Street, Troy, NY, USA, 12180; benedd@rpi.edu, mail@wrfranklin.org, liw9@rpi.edu

provides several Krylov subspace linear system solvers, providing the capability of approximating the system solution, of which GMRES is used for this implementation.

5. RESULTS

A MATLAB CPU-based implementation was used as a benchmark. Table 1 shows comparisons between the MATLAB and CUDA implementations of 2D-ODETLAP for various sized data sets with regard to total computation time of the algorithm. These tests were all performed using an average error of less than 0.1% as a stopping condition for the algorithm. Likewise, all tests specified the smoothness coefficient as $R = 0.1$, which weighs the known point equations more heavily than the averaging equations when approximating the solution.

Table 1: A timing comparison table, showing performance improvements when using CUDA over MATLAB. All ODETLAP parameters remain constant between the MATLAB and CUDA tests.

dataset	MATLAB	CUDA
400x400	16.4 s	2.3 s
1024x1024	425.6 s	16.7 s
2048x2048	1556.8 s	49.8 s

The results show that the largest tested set shown was computed greater than thirty times faster when using the CUDA implementation. Due to overhead associated with CUDA setup, smaller datasets benefit less from the acceleration due to GPU parallelism. Notably, the values obtained from the resulting CUDA-based ODETLAP implementation were comparable in both relative average error and relative maximum error to the results from the MATLAB implementation.

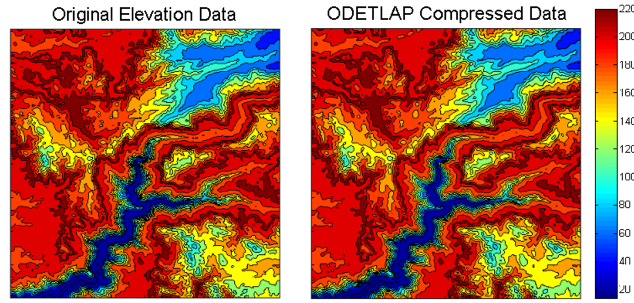
Figure 1 shows example results produced by the CUDA implementation of 2D-ODETLAP. This test set is a 400x400 elevation map, which is a data set used by the GeoStar project [6]. The stopping condition was set to a threshold of 0.5% average error, and the smoothness coefficient was set to $R = 0.25$. The final result is compressed to 20.29% of the number of points from the original data, with an average error of 0.442% and a maximum error of 8.668%. This test completed after five iterations with a time of 1.89 seconds.

6. CONCLUSION AND FUTURE WORK

The technique described in this paper provides a method for the construction and approximation portions of the ODETLAP algorithm, fully utilizing the capabilities of the GPU. This differs from previous techniques, which relied on MATLAB and the CPU for the construction of the overdetermined system equations [1, 2]. This paper also showed the large performance improvement achieved by utilizing CUDA over a CPU-based implementation, such as MATLAB.

There are several areas to explore for future work. Implementations of higher dimensional ODETLAP versions using this technique will be created. Techniques for the better

Figure 1: 2D-ODETLAP 400x400 example, with stopping condition threshold set to 0.5% average error and smoothness coefficient $R = 0.25$.



handling of boundary points will be investigated. Alternative initial and iterative point selection methods will be tested. Techniques for overcoming the limitations of memory within the GPU will be explored.

This research was partially supported by NSF grant IIS-1117277.

7. REFERENCES

[1] W. Randolph Franklin, You Li, Tsz-Yam Lau, and Peter Fox. *CUDA-accelerated HD-ODETLAP: lossy high dimensional gridded data compression.* In: 2012 International Workshop on Modern Accelerator Technologies for GIScience (MAT4GIScience 2012), Columbus, OH, 18 Sep 2012.

[2] You Li. *CUDA-accelerated HD-ODETLAP: a high dimensional geospatial data compression framework.* PhD thesis. Rensselaer Polytechnic Institute, 2011.

[3] Jared Stookey, Zhongyi Xie, Barbara Cutler, W. Randolph Franklin, Daniel M. Tracy, and Marcus V.A. Andrade. *Parallel ODETLAP for terrain compression and reconstruction.* In: 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL GIS 2008), Ed. by Walid G. Aref et. al., Irvine, CA, 5-7 Nov 2008.

[4] Lluis Pesquer, Xavier Pons, Ana Cortes, and Ivette Serral. *Spatial pattern alterations from JPEG2000 lossy compression of remote sensing images: massive variogram analysis in high performance computing.* In: Journal of Applied Remote Sensing 7(1), Jan 17 2013.

[5] S.H. Dabhole, V.A. Gundale, and J. Potgieter. *An efficient modified structure of CDF 9/7 Wavelet based on adaptive lifting with SPIHT for lossy to loseless image compression.* In: 2013 International Conference on Signal Processing Image Processing and Pattern Recognition (ICSIPR), Feb 7-8 2013.

[6] W. Randolph Franklin, Metin Inanc, Zhongyi Xie, Daniel M. Tracy, Barbara Cutler, Marcus V. A. Andrade, and Franklin Luk. *Smugglers and border guards – the GeoStar project at RPI.* In: 15th CM International Symposium on Advances in Geographic Information Systems (ACM GIS 2007), Seattle, WA, USA Nov 2007.