A Method of Adaptive Coarsening for Compressing Scientific Datasets

Tallat M. Shafaat[1] and Scott B. Baden[2]

[1] Royal Institute of Technology (KTH),
School of Information and Communication, Stockholm, SWEDEN,
tallat@kth.se
[2] Department of Computer Science and Engineering,
University of California, San Diego, La Jolla CA 92093-0404, USA
baden@cs.ucsd.edu
Home page: http://www.cse.ucsd.edu/users/baden

Abstract. We present *adaptive coarsening*, a multi-resolution lossy compression algorithm for scientific datasets. The algorithm provides guaranteed error bounds according to the user's requirements for subsequent post-processing. We demonstrate compression factors of up to an order of magnitude with datasets coming from solutions to time-dependent partial differential equations in one and two dimensions.

1 Introduction

A current challenge in large scale computing is how to compress, without losing valuable information, the prodigious output of simulations of ever increasing fidelity. Compression is traditionally applied to data visualization [11, 5, 9, 6]. However, analysis and post-processing of scientific data often entails taking spatial derivatives, and the required tolerances usually exceed that of visualization itself. Not surprisingly, users are reluctant to compress data in the spatial domain. Instead, they often compress by sub-sampling in the time domain. While such sub-sampling introduces aliasing errors, users seem less concerned about the artifacts introduced in the temporal domain than in the spatial domain. An alternative is to use lossless compression [8, 4]. But owing to the presence of noise in floating point data, compression is modest—often far less than a factor of two.

We present a lossy compression strategy called *adaptive coarsening* which coarsens data selectively and locally according to how the user intends to subsequently process the data. Our algorithm produces a compact representation providing guaranteed error bounds for the designated post-processing operations. An adaptively coarsened dataset includes geometric meta data describing the structure of the multi-resolution mesh. This information may be used to optimize subsequent data access and analysis.

We discuss preliminary results with two data sets coming from solutions to one and two dimensional partial differential equations. Using Adaptive Coarsening, we obtained compression factors of up to 11 and 15, respectively. Like wavelet

compression [9], adaptive coarsening is a multi-resolution technique; however, it does not represent data progressively.

This paper makes two contributions: (1) it offers an adaptive sub-sampling procedure for scientific data represented as uniform arrays, and (2) a framework that takes into account the context in which the data is subsequently post-processed.

2 Adaptive Coarsening

Consider the numerical solution to a time-dependent partial differential equation in one dimension shown in Fig.1, which has been computed on a uniform mesh using a finite difference method. Owing to the irregularity of the solution, some portions of the mesh may be stored at a lower resolution than others without significant loss of accuracy. As a result we may approximate the uniform mesh with a mesh with variable spacing as shown in Fig.2.

Adaptive coarsening works by coarsening a mesh, re-constructing the result back to the original mesh's index domain, and coalescing (coarsening) those points where the re-constructed data approximates the original data with sufficient accuracy. This process is carried out recursively on the newly generated coarse mesh(es) until it is no longer possible to coarsen the data: either further coarsening would violate a specified error bound, or the resultant new coarse meshes fall below some minimal size threshold. As previously mentioned, the notion of "sufficient accuracy" is measured with respect to the designated post-processing operations. For example, if we are interested in accurately reconstructing second derivatives from compressed data, then we measure the accuracy in terms of the second derivative of the input, rather than the input itself.

Adaptive coarsening employs the following error estimation procedure to determine which points may be safely stored at a lower resolution. Let G^h be a set of points defined uniformly over a discrete index space, which is a set of contiguous integers in one dimension. (In general we have a regularly spaced set of points in d dimensions.) Let $u(G^h)$ or, by an abuse of notation, u^h, represent the data on G^h. Define the *sub-sampling* operator $\mathcal{R} : u^h \rightarrow u^{2h}$ which maps values from a mesh defined on G^h onto a coarsened index domain G^{2h}, which, in one dimension, has half as many points.[3] Define the *up-sampling* operator $\mathcal{P} : u^{2h} \rightarrow u^h$, that maps values from a coarse index domain G^{2h} onto the refined domain G^h.

As previously mentioned, compression will be carried out in the context of post-processing. Let Φ be the post-processing operator. We compress the data, post-process it, and then re-construct the original by up-sampling, i.e. interpolation. That is, we compute $\mathcal{P} \circ \Phi \circ \mathcal{R}\ (u^h)$. Noting that $\mathcal{P} \circ \mathcal{R}$ is not the identity operator, we define the *error* operator $\mathcal{E} = \Phi - \mathcal{P} \circ \Phi \circ \mathcal{R}$. Now, for a given relative error threshold $\epsilon << 1$, we may coarsen the mesh without a significant loss of accuracy where $|\mathcal{E}(u^h)/\Phi(u^h)| < \epsilon$, and $|\cdot|$ is the absolute value taken point-

[3] The coarsening factor may be greater than two.

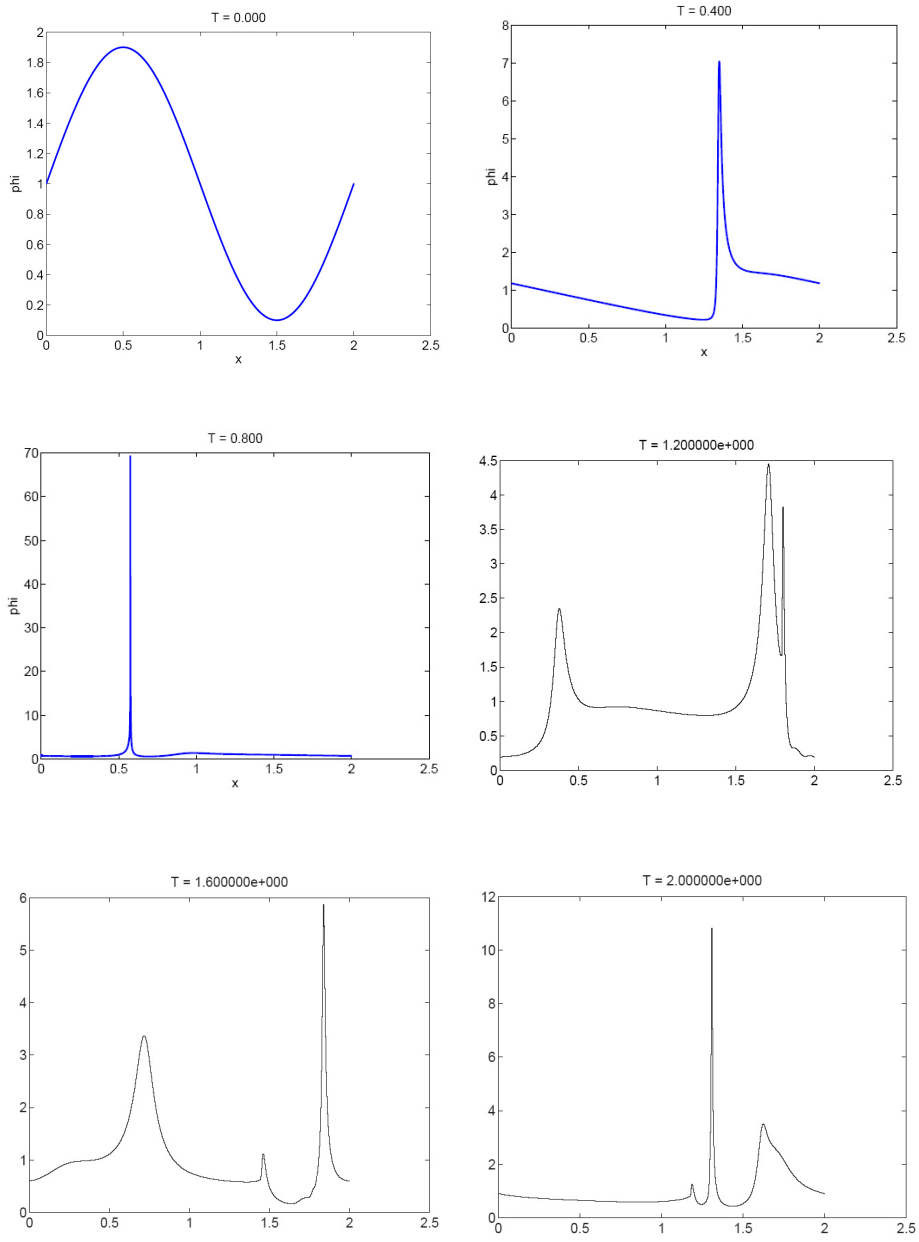

Fig. 1. The solution to a time dependent partial differential equation.

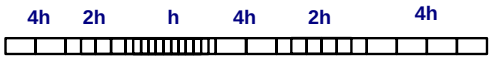

Fig. 2. An adaptively coarsened mesh.

wise[4]. We carry out this procedure recursively on the newly coarsened portions of the mesh until it is no longer possible to coarsen the data.

The adaptive coarsening algorithm appears as pseudo code in Fig. 3. The algorithm proceeds one level at a time (line 2), starting from the initial (finest) level 0. Line (5) estimates the error and coarsens the mesh accordingly, generating a list of intervals (rectangles in 2D, etc.). Since there may be more than one subdomain at the parent's level, we augment the existing coarse grids at the current level with a union $\cup$ operation (6). We also remove from the previous finer level any points that have been coarsened, since each point may appear in at most one level.[5] We perform this operation using set difference (7).

Adaptive coarsening requires rules for coarsening and refining the data, that is, the operations $\mathcal{P}$ and $\mathcal{R}$. These rules may be defined by the user, in particular, to customize them to the data and the postprocessing operator Φ. At present we use simple sub-sampling for $\mathcal{R}$, that is, selecting every Cth point, where C is the sub-sampling rate. Our up-sampling operator $\mathcal{P}$ currently uses a piecewise cubic Hermite interpolating polynomial[6].

3 Results

We present results for 1D and 2D datasets obtained by solving time-dependent partial differential equations.

1D dataset. We solved a variant of the Burger's equation on a mesh with 16,385 points over a series of 163,840 timesteps. The dataset consists of 81 snapshots spaced at equal intervals in time, including the initial data. We take the second derivative as the post-processing operation.

We applied adaptive coarsening to the 81 snapshots, using relative error thresholds of 10^{-3} and 10^{-4}. We obtained compression factors of 11.4 and 3.86, respectively.

[4] Clearly the significance of "loss of accuracy" depends on the choice of an appropriate definition of ϵ.

[5] By comparison, progressive methods would represent values at all levels, up to the coarsest one.

[6] Matlab `pchip`, www.mathworks.com/access/helpdesk/help/techdoc/ref/pchip.html

1. **let** level[0] = u^h
2. **for** i := 1 **to** maxLev
3. **let** h^i := discretization at level i
4. **foreach** subdomain $u_j^{h^{i-1}} \in$ level[i-1]
5. **where** $\mathcal{E}(u_j^{h^{i-1}}) < \epsilon$, create a new coarse domain u^{h^i}
6. level[i] := level[i] $\cup$ u^{h^i}
7. level[i-1] := level[i-1] $\setminus \mathcal{P}u^{h^i}$
8. **end foreach**
9. **end for**

Fig. 3. The adaptive coarsening algorithm. The operations $\setminus$ and $\cup$ are set union and set difference operations.

If we *do not* take into account the context of post-postprocessing, i.e., we set Φ to the identity operation, compression increases significantly: in our case to 53.8 and 19.6, respectively. Thus, we reduce compression effectiveness when we take into account how the data will be subsequently analyzed. However, this tradeoff is essential: it gives us the assurance that we will incur only modest, i.e. acceptable, errors when we take second derivatives of the compressed data.

In order to compensate for the decreased spatial sampling rate in coarsened data, we computed the 2nd derivative post processing operation using a higher order (4th order) 5-point centered-difference formula. Compared with the 3-point 2nd order formula, the higher order formula boosts compression by about 20% [10]. The added cost of this higher order stencil is modest. While the number of floating point operations increases, the number of memory accesses stays the same, and these largely determine the cost of taking spatial derivatives. Moreover, the cost of the I/O dominates the cost of CPU processing,

Adaptive coarsening is similar to adaptive sub-sampling employed in High Definition TV signal processing [1], which splits the index domain into fixed size pieces and sub-samples each piece separately. This strategy results in a more regular structure, and lower software bookkeeping overheads, than adaptive coarsening. Owing to additional constraints on the sub-sampling process, we expect that adaptive sub-sampling will yield a lower compression factor than adaptive coarsening. Indeed, the simplicity of adaptive sub-sampling comes at a cost. Compared with adaptive coarsening, compression drops to 7.56 and 3.33, respectively. At a threshold of 10^{-3}, adaptive coarsening is about 40% more effective than adaptive sub-sampling. The added flexibility offered by adaptive coarsening fits the sub-sampling pattern more tightly to the measured error in the data, leading to a higher degree of compression.

2D data set. The 2D data set was obtained by solving the Navier Stokes equations.[7] There were 51 snapshots, each comprising an array of 1020 $\times$ 1020 floating point numbers. These data carry a differentiated quantity (vorticity), and hence have already been post-processed. We have preliminary results for

[7] `http://www.math.ucla.edu/~anderson/270e.1.04f/Assignment8/Assign8.html`.

just one threshold: 10^{-3}. We applied adaptive sub-sampling in two dimensions, obtaining a compression factor of 14.9.

We then selected the parts of the mesh that could not be compressed at all, unraveling them in row major order into 1D arrays and applying 1D adaptive coarsening to the result. Because the data are smoother in the X direction than the Y, this optimization increased compression by about 10% to 16.4. However, knowing which direction to compress along requires some user intervention. We are currently extending adaptive coarsening to multiple dimensions, and pursuing automated strategies for anisotropic compression.

4 Conclusions and Future Work

We have demonstrated a multi-resolution compression technique called adaptive coarsening. Our approach is based on the philosophy that technological factors present an opportunity to employ plentiful processing cycles to reduce the space required to store massive data sets. We have been able to achieve an order of magnitude in compression for uniform mesh data sets. The effect is to significantly reducing the time to transfer data from off-line storage, and to process the data in subsequent analysis. Moreover, this level of compression was obtained while retaining engineering precision. Compression drops rapidly as the number of digits increases. To this end we are investigating more sophisticated sampling and interpolation procedures. Our ultimate target is three dimensional data, and work is currently in progress.

Although Adaptive Coarsening was applied to data arranged on a regular array of points, the technique is applicable to irregular data sets, e.g. arising in finite element methods, so long as there exist appropriate sub-sampling and up-sampling procedures [7].

Adaptive coarsening parallelizes readily. Sub-sampling and up-sampling operations require nearest neighbor communication only. Some collective communication is required to ensure that the irregular mesh structure makes balanced use of parallel I/O, but the cost is modest compared with that of the I/O.

Multiresolution compression techniques have been employed for several years in computer graphics, signal processing, and efficient mesh generation [11, 5, 9, 6, 9]. A complete list of references is too lengthly to include here. Hierarchical triangulations have been applied to the visualization of geophysical data [6]. Adaptive coarsening is similar in spirit to this technique, though the goal is to conserve space rather than bandwidth. By conserving space, adaptive coarsening also conserves bandwidth.

Our approach permits the direct manipulation of the compressed data sets; that is, without the need to reconstruct the original data. This capability is desirable because it enables data analysis to proceed within the reduced footprint of the compressed data and promotes the reuse of existing analysis code bases, simplifying post-processing application development. A framework may be constructed to apply the analysis code to each mesh component. This frees the user from having to manage the details. In general, the user needs to provide ap-

plication dependent coarsening and refinement modules along with appropriate metrics for estimating the error. While such operations are often application-dependent, strategies based on Richardsonian extrapolation, for example, are robust in Structured Adaptive Mesh Refinement [3, 2]. Thus, there is hope that they may prove equally useful in adaptive coarsening. These issues await further study.

5 Acknowledgments

This research was carried out while the second author was on sabbatical leave at Kungliga Tekniska Högskolan (KTH) in Stockholm, Sweden. The authors gratefully acknowledge the assistance of Bjorn Sjögren (KTH) and Chris Anderson (UCLA) for providing code and advice for the 1D and 2D PDE solvers, respectively. We are also grateful to Jesper Oppelstrup for many lively discussions. Scott Baden dedicates his portion of the work done in this paper to the memory of Olav Beckmann (1972-2006). This work was supported in part by NSF contract ACI9619020 (National Partnership for Advanced Computational Infrastructure (NPACI)) and the University of California, San Diego.

References

1. R.A.F. Belfor, M.P.A. Hesp, R.L. Lagendijk, and J. Biemond. Spatially adaptive subsampling of image sequences. *IEEE Trans. Image Proc.*, 3(5):492–500, 1994.
2. Marsha J. Berger and Phillip Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82(1):64–84, May 1989.
3. Marsha J. Berger and Joseph Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53(3):484–512, March 1984.
4. E. Engelson, D. Fritzson, and P. Fritzson. Lossless compression of high-volume numerical data from simulations. In *Proc. Data Compression Conf.*, pages 574–586, 2000.
5. L. A. Freitag and R. M. Loy. Adaptive, multiresolution visualization of large data sets using a distributed memory octree. In *Proc. SC99*, November 1999.
6. T. Gerstner. Multiresolution visualization and compression of global topographic data. *GeoInformatica*, 2001.
7. Benjamin F. Gregorski, David E. Sigeti, J. J. Ambrosiano, G. Graham, M. Wolinsky, Mark A. Duchaineau, Bernd Hamann, and Kenneth I. Joy. Multiresolution representation of datasets with material interfaces. In Gerald Farin, Bernd Hamann, and Hans Hagen, editors, *Hierachical and Geometrical Methods in Scientific Visualization*, pages 99–117. Springer-Verlag, Heidelberg, Germany, 2003.
8. P. Ratanaworabhan, J. Ke, and M. Burtscher. Fast lossless compression of scientific floating-point data. In *Proc. Data Compression Conf.*, 2006.
9. J. Roerdink and Michel Westenberg. Wavelet-based volume visualization. Technical Report 98-9-06, Univeristy of Groningen, 1998.
10. Tallat Mahmood Shafaat. Context dependent compression using adaptive subsampling of scientific data. *M.S Thesis, KTH, Stockholm, Sweden*, 2006.
11. Jane Wilhelms and Allen Van Gelder. Octrees for faster isosurface generation. *ACM Trans. Graph.*, 11(3):201–227, 1992.