

A Method of Adaptive Coarsening for Compressing Scientific Datasets

Tallat M. Shafaat¹ and Scott B. Baden²

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`,
WWW home page: <http://www.cse.ucsd.edu/users/baden>

Abstract. We present work in progress with *adaptive coarsening*, a multi-resolution lossy compression algorithm for scientific datasets providing 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.

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. Strategies for compressing data for subsequent visualization are too numerous to enumerate here [7, 3, 6, 4]. However, analysis and post-processing of scientific data often entails taking spatial derivatives, and the required tolerances usually exceed that of visualization itself. An alternative is to use lossless compression [5, 2], but compression is generally limited to a factor of two at best.

We present work in progress with a lossy compression strategy that coarsens data selectively and locally according to how the user plans to subsequently process the data. This *adaptive coarsening* algorithm produces a compact representation providing guaranteed error bounds for the designated post-processing operations. Like wavelet compression [6], adaptive coarsening is a multiresolution technique; however, it does not represent data progressively.

We present preliminary results with two data sets obtained by solving one and two dimensional partial differential equations, and obtained compression factors of up to 11 and 15, respectively.

2 Adaptive Coarsening

Consider the numerical solution to a time-dependent partial differential equation in one dimension, which has been computed on a uniform mesh. Owing to

irregularity in the smoothness of the solution, some portions of the mesh may be stored at a lower resolution than others without a significant loss of accuracy, as shown in Fig.1.

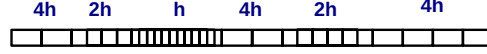


Fig. 1. An adaptively coarsened mesh.

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. Let u^h represent the data on G^h . Define the *restriction* operation $\mathcal{R} : u^h \rightarrow u^{2h}$ 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. Define the *prolongation* operation $\mathcal{P} : u^{2h} \rightarrow u^h$ that maps values on a coarse index domain G^{2h} onto G^h .

As mentioned, compression will be carried out in the context of post-processing. Let Φ be the post-processing operator. We first compress the data, post-process, and then re-construct the result by interpolation: compute $\mathcal{P} \circ \Phi \circ \mathcal{R}(u^h)$. We define the *error* operator $\mathcal{E} = \Phi - \mathcal{P} \circ \Phi \circ \mathcal{R}$. Now, for a given relative error threshold $\epsilon \ll 1$, 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-wise. 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 Fig. 2.

```

( 1) let level[0] =  $u^h$ 
( 2) for i := 1 to maxLev
( 3)   let  $h^i$  := discretization at level i
( 4)   foreach subdomain  $u_j^{h^{i-1}} \in \text{level}[i-1]$ 
( 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. 2. The adaptive coarsening algorithm. The operations $\setminus$ and $\cup$ are set union and set difference operations.

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 appears in at most one level (by comparison, progressive methods would represent values at all levels, up to the coarsest one). We do this using set difference (7).

To fully specify the algorithm we need $\mathcal{P}$ and $\mathcal{R}$. These procedures may be customized to the data but we presently use downsampling for the coarsening procedure and linear interpolation for the refinement procedure.

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 took 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. We settled on these tolerances after consulting users; the consensus was that 3 or 4 decimal digits are sufficient for *derived* quantities.

When we don’t take into account the context of post-postprocessing, that is, setting Φ to the identity operation, we boost compression significantly, in our case to 53.8 and 19.6, respectively. However, unacceptable errors are introduced when taking the second derivatives of this compressed data product³. When we compress data in the context of post-processing, we can differentiate the compressed data with violating accuracy constraints.

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 results in a more regular structure, and lower software bookkeeping overheads, than adaptive coarsening. Owing to additional constraints on the downsampling process, we expect that adaptive subsampling will yield a lower compression factor than adaptive coarsening. Indeed, the simplicity of adaptive subsampling 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 downsampling 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.⁴ There were 51 snapshots, each comprising an array of 1020×1020 floating point numbers. These data carry a differentiated quantity (vorticity), and hence have already been post-processed. We have preliminary results for just one threshold: 10^{-3} . We applied adaptive sub-sampling in two dimensions,

³ We used a 5-point, centered-difference formula which is 4th order accurate.

⁴ <http://www.math.ucla.edu/~anderson/270e.1.04f/Assignment8/Assignment8.html>.

obtaining a compression factor of 14.9. Encouraged by this result, we plan to implement adaptive coarsening, as we believe that the improvements shown in the 1-dimensional experiments will increase in higher dimensions.

4 Conclusions and Future Work

We have presented work in progress with a compression strategy called adaptive coarsening. This paper makes two contributions: it offers an improved adaptive subsampling procedure for scientific data that trades off greater irregularity for improved compression; it establishes the need to compress in the context of how data will be used in subsequent post-processing. Up to an order of magnitude in compression was obtained while retaining engineering precision. Compression drops rapidly as the number of digits increases, and to this end we are investigating more sophisticated sampling and interpolation procedures. We expect higher compression in three dimensions, which is currently under investigation.

Adaptive coarsening parallelizes readily. Coarsening and refinement procedures 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, since the data block sizes are large.

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. This work was supported in part by NSF contract ACI9619020.

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. 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.
3. L. A. Freitag and R. M. Loy. Adaptive, multiresolution visualization of large data sets using a distributed memory octree. In *Proc. SC99*, November 1999.
4. T. Gerstner. Multiresolution visualization and compression of global topographic data. *GeoInformatica*, 2001.
5. P. Ratanaworabhan, J. Ke, and M. Burtscher. Fast lossless compression of scientific floating-point data. In *Proc. Data Compression Conf.*, 2006.
6. J. Roerdink and Michel Westenberg. Wavelet-based volume visualization. Technical Report 98-9-06, Univeristy of Groningen, 1998.
7. Jane Wilhelms and Allen Van Gelder. Octrees for faster isosurface generation. *ACM Trans. Graph.*, 11(3):201–227, 1992.