UCRL-CONF-226261

LAWRENCE LIVERMORE NATIONAL LABORATORY

Spectral Predictors

L. Ibarria, P. Lindstrom, J. Rossignac

November 21, 2006

Data Compression Conference
Snowbird, UT, United States
March 27, 2007 through March 29, 2007

Spectral Predictors

Lorenzo Ibarria
Georgia Institute of Technology
redark@cc.gatech.edu

Peter Lindstrom
Lawrence Livermore National Laboratory
pl@llnl.gov

Jarek Rossignac
Georgia Institute of Technology
jarek@cc.gatech.edu

Abstract

Many scientific, imaging, and geospatial applications produce large high-precision scalar fields sampled on a regular grid. Lossless compression of such data is commonly done using predictive coding, in which weighted combinations of previously coded samples known to both encoder and decoder are used to predict subsequent nearby samples. In hierarchical, incremental, or selective transmission, the spatial pattern of the known neighbors is often irregular and varies from one sample to the next, which precludes prediction based on a single stencil and fixed set of weights. To handle such situations and make the best use of available neighboring samples, we propose a local spectral *predictor that offers optimal prediction by tailoring the weights to each configuration of known nearby samples. These weights may be precomputed and stored in a small lookup table. We show that predictive coding using our spectral predictor improves compression for various sources of high-precision data.*

1. Introduction

The acquisition or computation of scientific data sets [1], high dynamic range images [2], and geospatial data [3] usually requires a significant amount of effort and computing resources. Yet, their exploitation is often hindered by the mismatch between the size of the files in which they are stored and the available bandwidth for downloading or visualizing them. Although the loss of precision resulting from controlled quantization or lossy compression may be acceptable for visualization purposes, lossless compression of integer or floating-point values is required in many settings to guarantee the integrity of the data, e.g. when saving state in "restart dumps" for checkpointing numerical simulations [1].

Whereas traditional image compression techniques are capable of lossless compression [4, 5], they were developed for the media industry which usually deals with low-precision data and tolerates trading some accuracy for increased compression. In contrast, we focus on the lossless compression of high-precision data sets represented for example as 32-bit integers or floating-point numbers. The standard approach to lossless compression of such data is based on *predictive coding* [6–9], and several prediction schemes for structured data sets have been proposed [4, 10–13]. These prior schemes work best when the traversal over the data is simple, e.g. scanline order, so that each sample can be predicted from a single spatial configuration (stencil) of nearby, previously coded samples. When more general traversals are desired or when a nontrivial subset of samples is requested, the configuration of nearby known samples is often irregular and changing, which normally requires falling back on simpler predictors involving fewer samples. In this paper, we address how to make predictions from such irregularly populated neighborhoods that better take advantage of the known samples, and show that such predictors lead to improved compression of high-precision data. Using Fourier analysis, we develop optimal *spectral predictors* for small neighborhoods. While the derivation of the weights for these predictors is somewhat involved, the weights may be precomputed and stored in a small lookup table (available at `http://www.cc.gatech.edu/~lindstro/data/spectral/`), and are straightforward to use in a compression scheme.

The compression and streaming approach investigated here follow a simple paradigm: compute the prediction $p_{i,j}$ of the scalar value $f_{i,j}$ as a weighted combination of previously processed samples in the neighborhood $N_{i,j}$; compress the corrections, $c_{i,j} = f_{i,j} - p_{i,j}$, e.g. using entropy coding; and stream them. The paradigm leads to simplicity of implementation, small memory footprint, and excellent compression.

Although our framework is general enough to handle larger neighborhoods and unstructured and higher-dimensional data, we limit our attention in this paper to prediction within 3×3 neighborhoods. (While using larger neighborhoods may improve compression, precomputing and storing the $n(n-1)2^{n-1}$ weights for all possible combinations of known samples in an n-sample neighborhood is impractical for large n.) When the predicted sample is at a corner of a full neighborhood (all eight neighbors known), our spectral predictor reduces to the extrapolating *bi-Lorenzian* predictor; an extension of the previously proposed Lorenzo predictor suited for scanline transmission. When the predicted sample is at the center of a full neighborhood, we obtain the *radial* interpolating predictor, which is four times more accurate than the bi-Lorenzian and is useful in hierarchical transmission. We show that the spectral predictor leads to smaller correctors than other predictors that use a 3×3 neighborhood for lossless compression of high-precision floating-point or integer data. We also explain how to select a priori the best of the nine possible 3×3 neighborhoods that contain the sample to be predicted.

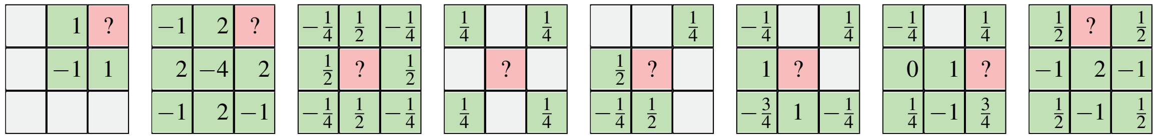
2. Extrapolating bi-Lorenzian predictor, L^2

Before we derive our spectral predictor, we begin by considering the L^1 Lorenzo predictor [13]. Let f be a one-dimensional function regularly sampled at $\{\ldots, f_{i-1}, f_i, f_{i+1}, \ldots\}$, and let Δ^x be the finite difference $\Delta^x_i = f_i - f_{i-1}$. That is, Δ^x is an approximation of the differential $\frac{\partial f}{\partial x} dx$. Setting $\Delta^x_i = 0$, solving for f_i, and substituting L^1_i for f_i, we have as 1D Lorenzo predictor $L^1_i = f_{i-1}$. The Lorenzo predictor extends to 2D via composition of derivatives: $\Delta^{xy}_{i,j} = \Delta^x_{i,j} - \Delta^x_{i,j-1} = f_{i-1,j-1} - f_{i,j-1} - f_{i-1,j} + f_{i,j}$. As the sampling rate of f increases, Δ^{xy} approaches $\frac{\partial^2 f}{\partial x \partial y} dx\,dy$ in the limit. Setting $\Delta^{xy}_{i,j} = 0$, we can now express the 2D Lorenzo predictor as

$$L^1_{i,j} = f_{i,j-1} + f_{i-1,j} - f_{i-1,j-1} \tag{1}$$

Thus, in the limit, L^1 correctly predicts all continuous functions f with $\frac{\partial^2 f}{\partial x \partial y} = 0$. In the discrete setting, L^1 recovers linear polynomials, or equivalently bilinear polynomials without highest order term xy. Figure 1(a) shows how the 2D Lorenzo predictor estimates the sample indicated by '?' as a weighted sum of three of its neighbors. We have successfully used L^1 in higher dimensions to predict regular grids [13].

It is natural to ask whether the Lorenzo predictor can be extended to higher-order polynomials that have vanishing higher-order derivatives. To accomplish this, we take finite differences once



(a) L^1 (b) L^2 (c) R (d) B (e) H (f) S_f (g) S_v (h) S_h

Figure 1. Weights for several spectral predictors used in our experiments: (a) Lorenzo, (b) bi-Lorenzian, (c) radial, (d) bilinear, (e) hybrid linear and radial, (f–h) full spectral.

more and obtain

$$\begin{aligned}\Delta^{xxyy}_{i,j} &= \Delta^{xy}_{i,j} - \Delta^{xy}_{i+1,j} - \Delta^{xy}_{i,j+1} + \Delta^{xy}_{i+1,j+1} \\ &= 2f_{i,j-1} + 2f_{i-1,j} + 2f_{i+1,j} + 2f_{i,j+1} - 4f_{i,j} - f_{i-1,j-1} - f_{i+1,j-1} - f_{i-1,j+1} - f_{i+1,j+1}\end{aligned}$$

where we define Δ^{xxyy} using central differences. Setting $\Delta^{xxyy}_{i,j} = 0$ and solving for $f_{i+1,j+1}$ we obtain the *bi-Lorenzian* predictor

$$L^2_{i+1,j+1} = 2f_{i,j-1} + 2f_{i-1,j} + 2f_{i+1,j} + 2f_{i,j+1} - 4f_{i,j} - f_{i-1,j-1} - f_{i+1,j-1} - f_{i-1,j+1} \tag{2}$$

In the limit, L^2 reproduces functions f with $\frac{\partial^4 f}{\partial x^2 \partial y^2} = 0$, and in the discrete setting interpolates biquadratic polynomials without highest order term x^2y^2. Whereas Δ^{xxyy} relates to Δ^{xy} as Δ^{xy} relates to f, L^2 is usually not the successive application of L^1, i.e. in general $L^2_{i,j} \neq L^1_{i,j-1} + L^1_{i-1,j} - L^1_{i-1,j-1}$. Instead, L^2 may be derived by setting to zero the L^1 correction of the L^1 corrections at (i,j). The L^2 weights are shown in Figure 1(b).

The L^1 predictor has been widely used in the image and geometry compression communities [4–6, 13]. We are, however, not aware of its extension L^2 having been used for compression of 2D and higher-dimensional data.

3. Interpolating radial predictor, *R*

In the previous section we presented an extrapolating predictor, L^2, for a corner $f_{i+1,j+1}$ of a 3×3 neighborhood of samples. This predictor arose from the constraint $\Delta^{xxyy}_{i,j} = 0$, a central difference evaluated at the *center* sample of this neighborhood. A more effective predictor is obtained by solving this equation for the function value at the center sample $f_{i,j}$ (the "face sample"), which results in the *radial* interpolating predictor

$$R_{i,j} = \frac{1}{4}\left(2f_{i,j-1} + 2f_{i-1,j} + 2f_{i+1,j} + 2f_{i,j+1} - f_{i-1,j-1} - f_{i+1,j-1} - f_{i-1,j+1} - f_{i+1,j+1}\right) \tag{3}$$

We use the term "radial" to describe this predictor because its weights are radially dependent on the distance to neighboring samples (Figure 1(c)). The prediction $R_{i,j}$ is $2E - C$, where E is the mean of the four edge neighbors $\{f_{i\pm1,j}, f_{i,j\pm1}\}$ and C is the mean of the four corner neighbors $\{f_{i\pm1,j\pm1}\}$. $R_{i,j}$ also equals the mean of the four possible L^1 predictions of $f_{i,j}$.

R has the same predictive power as L^2, i.e. it reproduces biquadratics with no x^2y^2 term, but typically yields better predictions due to the symmetric configuration of its neighborhood. Using Taylor expansion of f one can show that the prediction error of L^2 is $\frac{\partial^4 f}{\partial x^2 \partial y^2}$ (plus higher order terms), which is four times larger than the prediction error for R. Note that to use R, we either must know all eight surrounding neighbors or must estimate them via alternative predictors.

$$
\underset{B_0}{\begin{array}{|c|c|c|}\hline 1&1&1\\\hline 1&1&1\\\hline 1&1&1\\\hline\end{array}}\quad
\underset{B_1^x}{\begin{array}{|c|c|c|}\hline -1&0&1\\\hline -1&0&1\\\hline -1&0&1\\\hline\end{array}}\quad
\underset{B_1^y}{\begin{array}{|c|c|c|}\hline 1&1&1\\\hline 0&0&0\\\hline -1&-1&-1\\\hline\end{array}}\quad
\underset{B_2}{\begin{array}{|c|c|c|}\hline -1&0&1\\\hline 0&0&0\\\hline 1&0&-1\\\hline\end{array}}\quad
\underset{B_3^x}{\begin{array}{|c|c|c|}\hline -1&2&-1\\\hline -1&2&-1\\\hline -1&2&-1\\\hline\end{array}}\quad
\underset{B_3^y}{\begin{array}{|c|c|c|}\hline -1&-1&-1\\\hline 2&2&2\\\hline -1&-1&-1\\\hline\end{array}}\quad
\underset{B_4^x}{\begin{array}{|c|c|c|}\hline 1&0&-1\\\hline -2&0&2\\\hline 1&0&-1\\\hline\end{array}}\quad
\underset{B_4^y}{\begin{array}{|c|c|c|}\hline -1&2&-1\\\hline 0&0&0\\\hline 1&-2&1\\\hline\end{array}}\quad
\underset{B_6}{\begin{array}{|c|c|c|}\hline 1&-2&1\\\hline -2&4&-2\\\hline 1&-2&1\\\hline\end{array}}
$$

Figure 2. Basis functions for the 2D discrete cosine transform (not normalized).

4. Spectral predictor, S

Our spectral predictor S generalizes L^2 and R to all possible configurations of 0 to 8 known samples and locations of the predicted sample in a 3×3 neighborhood. As in image compression methods based on discrete wavelet [14] and cosine transforms [15], we capitalize on the fact that the signal power is often heavily skewed towards low frequencies. In frequency transforms, this results in small, compressible high-frequency detail coefficients, whereas in predictive coding "smooth" interpolants recover most of the low-frequency response, leading to small correctors for the missing high-frequency content.

In this section, we design as-smooth-as-possible interpolants for irregular sample configurations. We seek to eliminate or, when not possible, to minimize high-frequency responses in the interpolant. The resulting predictors and their sets of weights can be stored in a lookup table indexed by the mask of known and unknown values and the location of the predicted sample.

We build upon the work by Isenburg et al. [16], who use the Fourier transform to predict the geometry of n-sided polygons to be "as regular as possible" given $m < n$ known vertices. They express the vertex coordinates of the polygon in the complex plane, apply the discrete Fourier transform (DFT) to this n-vector of consecutive vertex coordinates, set the highest $n-m$ frequencies to zero, and compute the inverse transform to obtain the complex coordinates of the predicted vertices. Because the Fourier transform is linear, the unknown vertices can be expressed as a linear combination of the known vertices. By working out the mathematics of the forward and inverse Fourier transforms, one can a priori establish a set of weights for a given configuration (m,n) of known and unknown number of vertices (i.e. the weights are not dependent on the geometry of the known samples). Because Fourier frequencies come in pairs, this approach works well when m is odd as then the resulting weights are guaranteed to be real. One can show that the discrete cosine transform (DCT) can instead be used when m is even. Lifting the DFT to higher dimensions, Isenburg et al. further showed that the L^1 predictor from Section 2 is in the spectral sense the optimal predictor (i.e. smoothest interpolant) for hypercube-like neighborhoods with one unknown sample.

We begin by extending the general approach of Isenburg et al. to 3×3 neighborhoods to re-derive the bi-Lorenzian and radial predictors and show that they are optimal. We will make use of the two-dimensional (orthonormal) discrete cosine basis

$$\{u(x)u(y),\ s(x)u(y),\ u(x)s(y),\ s(x)s(y),\ c(x)u(y),\ u(x)c(y),\ s(x)c(y),\ c(x)s(y),\ c(x)c(y)\}$$

where x and y vary over the domain $\{-1,0,+1\}$ of our 3×3 neighborhood, and where

$$u(x) = \sqrt{\tfrac{1}{3}}, \quad s(x) = \sqrt{\tfrac{1}{3}}\sin\left(\tfrac{1}{3}\pi x\right), \quad c(x) = \sqrt{\tfrac{2}{3}}\cos\left(\tfrac{2}{3}\pi x\right)$$

The cosine basis is shown un-normalized in Figure 2. We unfold the 3×3 matrix into a single 9-dimensional vector $b = [f_{i-1,j-1} \quad f_{i,j-1} \quad f_{i+1,j-1} \quad \cdots \quad f_{i+1,j+1}]^T$ of sample values, and write the cosine basis as a 9×9 orthogonal matrix B, where the columns of B are the basis functions. Then the forward discrete cosine transform is simply $x = B^T b$, with x being the DCT coefficients in order of increasing frequency.

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \qquad P^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

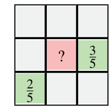


Figure 3. Example mask matrix M, interpolation matrix P, and predictor weights.

To extend the ideas of Isenburg et al. from 1D to 2D, we must rank the basis functions by increasing frequency. The cosine basis formulation gives us pairs of frequencies (ν_x, ν_y) for the horizontal and vertical direction, which must be consolidated into single frequencies. We approach this by deriving the cosine basis through eigenanalysis of the symmetric combinatorial graph Laplacian $\mathcal{L}$ (also called the Kirchoff matrix [17])

$$l_{ij} = \begin{cases} \deg(i) & \text{if } i = j \\ -1 & \text{if } i \text{ and } j \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where we consider the graph formed by the 3×3 neighborhood in isolation, with vertical and horizontal edges between adjacent samples. Here $\deg(i)$ denotes the degree or number of neighbors of a sample i, e.g. $\deg(i)$ is 2 for corner samples, 3 for edge samples, and 4 for face samples. As noted by Taubin [18], the eigenbasis of the normalized (asymmetric) Laplacian coincides with the Fourier basis, and the eigenvalues $\{\lambda_i\}$ of $\mathcal{L}$ correspond to frequencies. The above un-normalized (symmetric, positive semidefinite) Laplacian $\mathcal{L}$ has real non-negative eigenvalues $\{0,1,1,2,3,3,4,4,6\}$ and the cosine basis as eigenbasis. We will use B_λ to denote the eigenvector (i.e. basis function) with corresponding eigenvalue λ, and B^x_λ and B^y_λ to distinguish pairs of eigenvectors with equal eigenvalues (Figure 2).

Our formulation shows that there is a unique highest frequency $\lambda = 6$ with associated basis function B_6. Given the eight known samples in the bi-Lorenzian and radial predictors, similarly to Isenburg et al., we set the highest frequency response x_6 to zero and solve for the unknown sample as a linear combination of the $m = 8$ known samples, which results in the weights given in Equations 2 and 3 for corner and center predictions. When $m < 8$, a similar strategy is possible by zeroing $9-m$ of the highest frequencies. However, we may need to resolve two issues: (1) The $9-m$ first basis functions may not form a basis for the set of known samples, e.g. $\{B_0, B^x_1, B^y_1\}$ is not a basis for $b = [f_{i-1,j-1} \quad f_{i,j-1} \quad f_{i+1,j-1} \quad 0 \quad \cdots \quad 0]^T$. (2) In situations when only one of B^x_λ and B^y_λ is needed (e.g. when exactly two samples are known, as in Figure 3), we may reduce the total frequency response by choosing a linear combination of B^x_λ and B^y_λ.

Let M be an $m \times n$ mask matrix that extracts the m known samples Mb from b, i.e. each row of M has a single one entry and remaining zeros. We wish to solve the underconstrained system $MBx = Mb$ for x with as many high frequencies of x zeroed as possible. This can be done via linearly constrained least-squares methods, which involves symbolic inversion of an $(m+n) \times (m+n)$ matrix. We show here how to accomplish the same goal via inversion of a smaller $m \times m$ matrix.

We first must find an m-dimensional basis for Mb by selecting from or combining the $n > m$ column vectors MB; any excluded vector from MB will implicitly have its corresponding frequency response zeroed. Our approach is to incrementally construct an $n \times m$ interpolation matrix P that linearly combines vectors from MB such that $MBPy = Mb$ is a fully constrained system of m equations, with $Py = x$. We achieve this by adding to P columns that select basis functions from MB in order of increasing frequency λ. If a basis function projected onto the space of known samples is redundant (linearly dependent) with respect to the partially constructed basis, we exclude it and consider the next basis function. When we encounter an eigenspace, i.e. two basis functions with

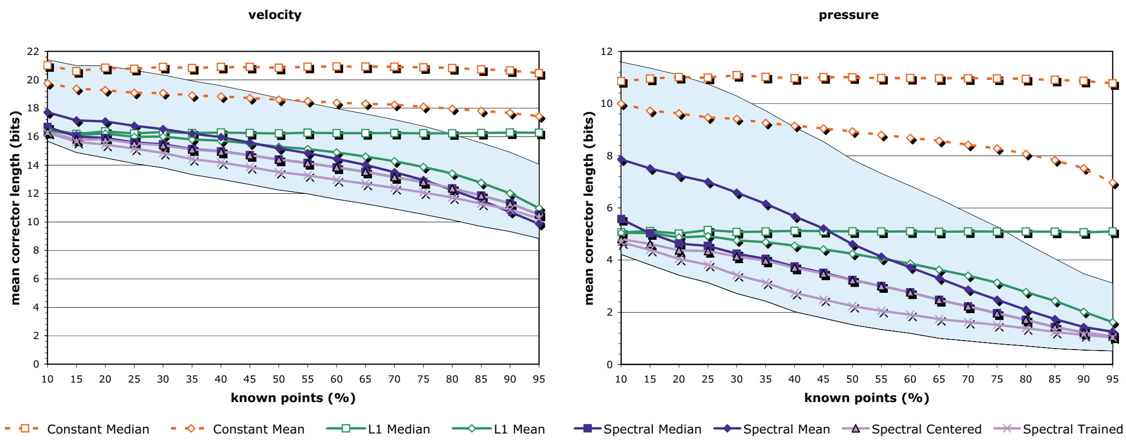


Figure 4. Predictor quality as a function of number of known points in the 3×3 neighborhood. The shaded area indicates the range between best and worst spectral prediction.

the same eigenvalue, one of three situations arises: (1) The whole eigenspace is redundant, and we exclude it. (2) The whole eigenspace is nonredundant, and we include it. (3) The eigenspace is partially redundant, in which case we first "rotate" the eigenspace by an angle θ to make one of the rotated and projected basis functions redundant. (Note that any rotation of an eigenspace preserves eigenvalues and orthogonality with the rest of the basis.) This leaves a nonredundant basis function $B_\lambda^\theta = \cos(\theta)B_\lambda^x + \sin(\theta)B_\lambda^y$ and we add to P a column that has $\cos(\theta)$ and $\sin(\theta)$ in the rows corresponding to B_λ^x and B_λ^y. The effect of this rotation is to "align" the basis function with the spatial configuration of known samples. One can show that this rotation leads to the minimal total frequency response $||x||$.

We may now compute $x = P(MBP)^{-1}Mb$ using matrix inversion. We are, however, not interested in the frequency response x but in the weights of the known samples Mb. Hence we apply the inverse DCT to x and compute $Bx = Wb$, where W is the $n \times n$ weight matrix $W = BP(MBP)^{-1}M$, which is then used to predict unknown samples in b from known ones.

We have implemented this method symbolically in Mathematica and computed exact weights W for all neighborhood configurations, resulting in 41 unique weights in the range $[-4, +4]$ that are predominantly integers and otherwise rationals. The complete list of weights can be found at `http://www.cc.gatech.edu/~lindstro/data/spectral/`. Note that our weights always add to unity, making our predictor affine invariant.

4.1. Choosing a neighborhood

Via translation we can form nine 3×3 neighborhoods around each predicted sample p. Depending on the configuration of known samples it is not immediately clear which neighborhood to predict from. We propose training the compressor on the given data set: each of the 9×2^8 predictors is exercised on each sample and receives a ranking based on the mean error it makes. This short ranking is transmitted before compression begins and determines the choice of predictor. In Figure 4 we show using random sampling of two data sets that our approach improves upon several alternatives that we have explored, including the neighborhood centered at p and the mean or the median of all nine predictions. For calibration, we also report the results for the best (lowest residual) of the nine neighborhoods (which unfortunately is not available to the decoder), as well as the mean and median of constant (single-value) and L^1 prediction.

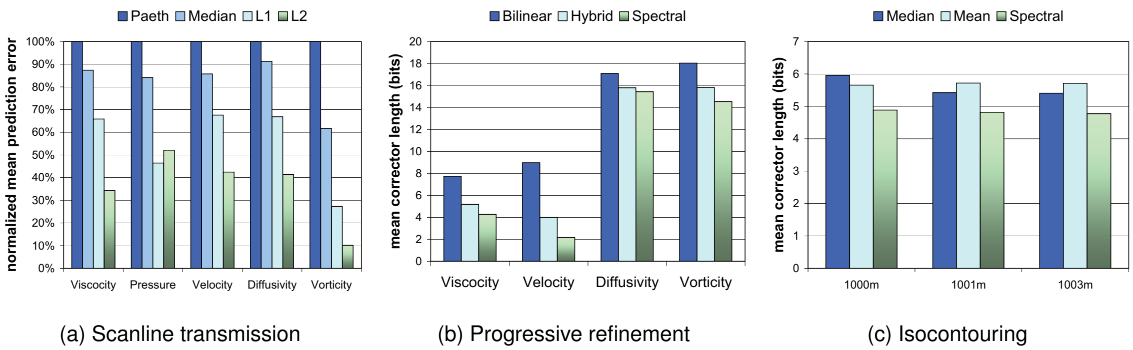


Figure 5. Prediction results for three different applications.

5. Applications of spectral prediction

Our spectral predictor is particularly useful in applications where standard compression techniques, e.g. based on wavelets, are not practical, such as for encoding data sets with irregular domains due to manual or automatic extraction, inpainting, selective updates, adaptive sampling, or range queries that extract those samples whose values fall within an interval. Irregular sample configurations also arise when the data is traversed in other than scanline order, or in mesh compression, where the domain connectivity is inherently irregular. For lack of space, we here consider only a few of these applications.

We evaluate predictor performance in terms of the number of significant corrector bits, which is the dominating cost in predictive coders for high-precision data [7, 8], including our own [9]. For floating-point data, we compute an integer corrector that measures the number of distinct floating-point values between the actual and predicted value (see [9]).

5.1. Scanline transmission

The most straightforward method to compress regularly gridded data is to make a scanline traversal, e.g. row-by-row from bottom to top and from left to right within each row. We here compare our bi-Lorenzian L^2 predictor with other scanline predictors proposed for image compression: the Paeth predictor [12] used in the PNG image format [5], the median predictor used in JPEG-LS [4], and the L^1 Lorenzo predictor [13] used by Lindstrom and Isenburg [9]. All except L^2 predict a sample from the same set of three neighbors (Figure 1(a)).

In order to apply L^2 in a scanline traversal, two rows of previously coded samples must be maintained (Figure 6(a)). To bootstrap the predictor, one may use lower-dimensional Lorenzo prediction to first recover domain boundaries. Alternatively, one may use the spectral predictor for partially known neighborhoods described in Section 4.

Figure 5(a) shows the results of predicting multiple 2D slices of the single-precision floating-point scalar fields shown in Figure 7 obtained from a fluid dynamics simulation [1]. On high-precision data like this, L^2 often offers substantially better prediction than predictors that use smaller stencils. The benefit of a larger stencil comes at the expense of higher sensitivity to quantization, however, due to accumulation of per-sample errors and larger (in magnitude) weights. Analysis shows that the prediction error due to quantization is three times larger for L^2 than for L^1. Hence L^2 generally performs worse than L^1 on low-precision data such as 8-bit images.

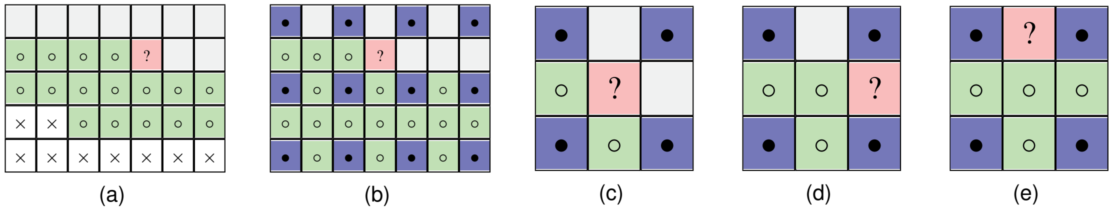


Figure 6. (a) L^2 footprint (circles) maintained during scanline traversal. (b) Coarse-resolution (solid) and fine-resolution (hollow) processed samples in a hierarchical traversal. Within each level of resolution, scanline traversal is used, resulting in three predictor stencils: (c) face, (d) vertical edge, and (e) horizontal edge sample.

5.2. Progressive refinement

Often, data sets are transmitted progressively, doubling the resolution in x and y after each refinement. The missing values within a refinement level may be transmitted in scanline order, as shown in Figure 6(b), which results in three 3×3 neighborhood configurations from which samples are predicted (Figure 6(b–d)).

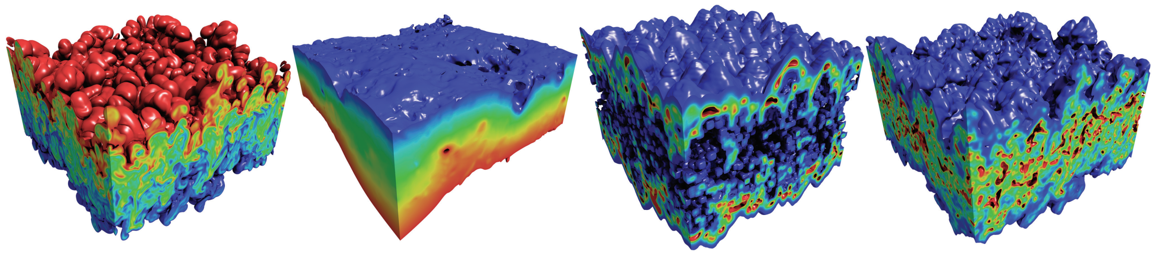
We consider three predictors for the face sample (Figure 6(c)): bilinear interpolation B of corner samples (Figure 1(d)), spectral prediction S_f (Figure 1(f)), and a hybrid predictor H (Figure 1(e)) that first linearly interpolates the unknown neighbors at the vertical and horizontal edges from their immediate neighbors to fill the neighborhood and then predicts the face point using radial prediction R. Note that both B and H are instances of spectral prediction that simply ignore some of the known neighbors. For the edge points, B and H resort to linear interpolation of corner points for prediction (since no other reasonable non-spectral predictor is available), while our spectral predictor is able to make use of all decoded samples (Figures 1(g) and 1(h)).

Figure 5(b) illustrates the advantage of using all known neighboring samples in the prediction. S_f offers in all cases superior prediction over B and H, leading in one case to as much as a 4:1 improvement in compression. Note that one may choose a different traversal order within each level. In fact, our experiments show that transmitting the missing edge samples first and then the face samples further improves compression, in part because the face samples may be predicted using the radial predictor with fully known (not simply estimated) neighborhoods.

5.3. Isocontouring

In many scientific, engineering, and medical applications, regularly sampled volumetric scalar fields are visualized in terms of isosurfaces. For instance, a remote viewer may wish to see the isosurface $S(t)$ formed by all points at temperature t or to explore the family $S(T)$ of isosurfaces with temperatures in a range $T = [t_{min}, t_{max}]$. Instead of transmitting the geometry of $S(t)$ or some compressed form of its animation, it is often more effective to transmit the minimal subset of scalar values needed to reconstruct the single isosurface $S(t)$ or family of isosurfaces $S(T)$ [19]. To satisfy this query, one needs to transmit not only the samples with values in T, but also some of their neighbors to obtain a complete "scaffold" around the surface. In a scenario where the the remote user later decides to extend T to a larger interval, compression and incremental transmission of the subset of additional samples would often be preferable over complete retransmission. For both initial and incremental transmission, it is not obvious how to predict the irregular subset of sample values using traditional means.

Because we are only interested here in illustrating the benefits of the spectral predictor, we will not discuss the transmission order nor how to encode the mask that identifies the missing samples.



(a) Density (b) Pressure (c) Diffusivity (d) Viscocity

Figure 7. Interval-volume renderings of some of the 3D scalar fields used in our experiments.

We focus on the prediction of the missing values and report experiments of extracting isolines in 2D from the Puget Sound 16-bit terrain surface (available at `http://www.cc.gatech.edu/projects/large_models/ps.html`). We first extracted an isocontour at 1000m elevation and predicted all necessary samples, then incrementally transmitted missing values for isocontours at 1001m and 1003m, resulting in an average number of known neighbors of 5.13, 5.42, and 5.41, respectively. Since samples are often not available for predictors like L^1 to be applied, we compare our spectral predictor with predictions based on the mean and median sample value in a 3×3 neighborhood centered on the predicted sample. We observed consistent reduction in corrector bit length (13–22%) using the spectral predictor, even for this lower-precision data set (Figure 5(c)).

6. Conclusions

We propose two new predictors, the bi-Lorenzian L^2 and the radial R, which predict the value of a sample f from eight values in a 3×3 neighborhood of which f is the corner (for L^2) or the center (for R). More importantly, we propose the spectral predictor S, which extends L^2 and R to all configurations of 0 to 8 known samples and locations of f in a 3×3 neighborhood. We argue that S is the best predictor from a 3×3 neighborhood, provide a strategy for selecting the most promising neighborhood that contains f, and demonstrate the benefits of S over competing predictors in three simple applications.

While applied only to 3×3 neighborhoods in 2D regular grids here, our framework, which is based on the eigenstructure of the combinatorial graph Laplacian, easily generalizes to higher dimensions and to irregular grids. One immediate application we envision is geometry prediction for polygonal and polyhedral meshes. In the more general setting, multiple and larger neighborhoods may arise, possibly leading to very large weight lookup tables. In order to reduce memory requirements, symmetry, non-uniqueness of weights and weight combinations, and unity constraints can be exploited, however having a more efficient procedure for on-demand computation of weights than symbolic or even numerical matrix inversion is clearly desirable.

Acknowledgements

This work was performed in part under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under contract W-7405-Eng-48.

References

[1] A. W. Cook, W. H. Cabot, P. L. Williams, B. J. Miller, B. R. de Supinski, R. K. Yates, and M. L. Welcome, “Tera-scalable algorithms for variable-density elliptic hydrodynamics with spectral accuracy,” in *ACM/IEEE Supercomputing*, 2005, p. 60.

[2] G. W. Larson, “LogLuv encoding for full-gamut, high-dynamic range images,” *Journal of Graphics Tools*, vol. 3, no. 1, pp. 15–31, 1998.

[3] D. F. Maune, *Digital Elevation Model Technologies and Applications: The DEM Users Manual*. American Society for Photogrammetry and Remote Sensing, 2001.

[4] M. J. Weinberger, G. Seroussi, and G. Sapiro, “The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS,” *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1309–1324, 2000.

[5] G. Roelofs, *PNG: The Definitive Guide*. O’Reilly, 2003, http://www.libpng.org/pub/png/book/.

[6] C. Touma and C. Gotsman, “Triangle mesh compression,” in *Graphics Interface*, 1998, pp. 26–34.

[7] V. Engelson, D. Fritzson, and P. Fritzson, “Lossless compression of high-volume numerical data from simulations,” in *Data Compression Conference*, 2000, pp. 574–586.

[8] P. Ratanaworabhan, J. Ke, and M. Burtscher, “Fast lossless compression of scientific floating-point data,” in *Data Compression Conference*, 2006, pp. 133–142.

[9] P. Lindstrom and M. Isenburg, “Fast and efficient compression of floating-point data,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 1245–1250, 2006.

[10] H. Kobayashi and L. R. Bahl, “Image data compression by predictive coding I: Prediction algorithms,” *IBM Journal of Research and Development*, vol. 18, no. 2, pp. 164–171, 1974.

[11] E. H. Feria, “Linear predictive transform of monochrome images,” *Image and Vision Computing*, vol. 5, no. 4, pp. 267–278, 1987.

[12] A. W. Paeth, “Image file compression made easy,” in *Graphics Gems II*, J. Arvo, Ed. San Diego: Academic Press, 1991.

[13] L. Ibarria, P. Lindstrom, J. Rossignac, and A. Szymczak, “Out-of-core compression and decompression of large *n*-dimensional scalar fields,” *Computer Graphics Forum*, vol. 22, no. 3, pp. 343–348, 2003.

[14] C. Chrysafis and A. Ortega, “Efficient context-based entropy coding for lossy wavelet image compression,” in *Data Compression Conference*, 1997, pp. 241–250.

[15] G. K. Wallace, “The JPEG still picture compression standard,” *Communications of the ACM*, vol. 34, no. 4, pp. 30–44, 1991.

[16] M. Isenburg, I. Ivrissimtzis, S. Gumhold, and H.-P. Seidel, “Geometry prediction for high degree polygons,” in *SCCG*, 2005, pp. 147–152.

[17] H. Zhang, “Discrete combinatorial Laplacian operators for digital geometry processing,” in *SIAM Conference on Geometric Design*, 2004, pp. 575–592.

[18] G. Taubin, “A signal processing approach to fair surface design,” in *ACM SIGGRAPH*, 1995, pp. 351–358.

[19] A. Mascarenhas, M. Isenburg, V. Pascucci, and J. Snoeyink, “Encoding volumetric grids for streaming isosurface extraction,” in *3DPVT*, 2003, pp. 665–672.