COGL: Coefficient Graph Laplacians for Optimized JPEG Image Decoding

Sean I. Young, Aous T. Naman, and David Taubman, *Fellow, IEEE*

***Abstract*—We address the problem of decoding JPEG-encoded images with less visual artifacts. We view the decoding task as an ill-posed inverse problem, and find a regularized solution using a convex, graph Laplacian-regularized model. Since the resulting problem is non-smooth and entails non-local regularization, we use fast high-dimensional Gaussian filtering techniques with the proximal gradient descent method to solve our convex problem efficiently. Our patch-based "coefficient graph" is better-suited than the traditional pixel-based ones for regularizing smooth non-stationary signals such as natural images, and relates directly to classic non-local means de-noising of images. We also extend our graph along the temporal dimension to handle the decoding of M-JPEG-encoded video. Despite the minimalistic nature of our convex problem, it produces decoded images with similar quality to other more complex, state-of-the-art methods while being up to five times faster. We also expound on the relationship between our method and the classic ANCE method, reinterpreting ANCE from a graph-based regularization perspective.**

***Index Terms*—Image restoration, Image coding, Image quality, Optimization**

I. Introduction

THE ubiquitous JPEG (Joint Photographic Experts Group) image compression [1] uses the discrete cosine transform (DCT) to decorrelate sample intensity values and generate a set of transform coefficients. Most of the generated coefficients are quantized to zero so that only a small number of non-zero coefficients need to be communicated, effectively compressing the image. The decoder reverses this process by dequantizing the communicated transform coefficients and reconstructing an approximation of the original image with the inverse DCT of the dequantized coefficients.

The decoder has knowledge only of the quantized transform coefficients, or equivalently, the quantization bins to which the original, unquantized coefficients belong. The standard JPEG dequantizer takes the quantization bin centers as estimates of the original transform coefficients prior to quantization. This choice minimizes the mean squared error of the reconstruction if the image samples are not correlated across transform blocks and the coefficients of each bin are uniformly distributed, as typically assumed for high-rate quatization. Naturally, better decoded images can be obtained especially at lower rates by considering the correlation across transform blocks. This helps to reduces unpleasant artifacts such as ringing and blocking that can develop at the boundaries of transform blocks.

The authors are with the School of EE&T, UNSW Sydney, Sydney, NSW 2052, Australia (e-mail: sean.young@unsw.edu.au; aous.naman@unsw.edu.au; d.taubman@unsw.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

One classic technique used for improving image decoding is projection onto convex sets [2]–[4], an iterative approach in which the first step smooths the image from the previous iteration, and the other step projects the transform coefficients of the smoothed image onto their communicated quantization bins representing the problem constraints. Foi et al. [5] treat the reduction of block artifacts as a de-noising problem. They employ a shape-adaptive DCT for de-noising, estimating noise powers using the decoded quantization table. Zhang et al. [6] employ a nonlocal approach which estimates the original DCT coefficients using a weighted combination of the dequantized transform coefficients, and coefficients from other overlapped non-coded transform blocks—those straddling multiple coded blocks. Sparse reconstruction methods which employ a learned dictionary of image atoms [7], [8] can also exploit other types of similarities exhibited across different images. Approaches based on convolutional neural networks, e.g. [9], directly learn the mapping between the decompressed JPEG image and the uncompressed image using a training image set. A more recent method [10] combines the dictionary-based sparsity prior with a graph-Laplacian regularizer.

The assumption made, either implicitly or explicitly, in all previous work is that the DCT coefficients of the coded blocks are related to those of their neighboring (possibly) non-coded blocks. We incorporate such an assumption explicitly into the image decoding problem via a graph Laplacian [11], where graph vertices represent coded or non-coded DCT blocks, and graph edge weights express the correlation between a pair of DCT coefficient vectors. The graph regularizer is used to enforce smoothness of the solution anisotropically along the direction of the correlation in the coefficient vectors.

Construction of a sparse graph directly in the image domain is often accompanied by a Gauss-Markov assumption of the image, which deems each pixel to be conditionally independent of the rest of the image, given a neighborhood of a fixed size (often 3×3). This assumption thus introduces an implicit notion of some fixed scale, which is unreasonable for natural images since they contain patterns and texture across multiple scales. Our "coefficient graph" is a dense graph in the image domain, which models the image as a correlated Gaussian process instead. We also show that a direct connection exists between our dense graph regularizer and the non-local means algorithm proposed by Buades et al. [12]. Our graph can readily be extended in the temporal dimension for optimized decoding of M-JPEG video as well.

The key contribution of this work is our "coefficient graph" Laplacian regularizer itself. We show that the image decoding problem can be formulated, minimally, as the graph-regularized

optimization problem

$$\min f(\mathbf{u}) = \delta_C^{\alpha}(\mathbf{T}\mathbf{u}) + (1/2)\mathbf{u}^*\mathbb{L}\mathbf{u}, \tag{1}$$

in which δ_C^{α} is a lower semi-continuous convex function, $\mathbf{T}$ is the DCT, and $\mathbb{L}$ is our graph Laplacian. Since $\mathbb{L}$ is based upon a non-local graph topology with Gaussian weights, we employ fast Gaussian filtering techniques to yield the solution efficiently. We validate our proposed method using a number of common test images, and show that our method, despite its simplicity, performs similarly to other more complex, state-of-the-art JPEG image decoding methods while being up to five times faster. Note that image de-noising formulations, while similar to (1), often employ a 1- or 2-norm data fidelity term in place of δ_C^{α} [13]. They perform poorly if applied to image decoding, since the constraints of the image decoding problem are not captured by norm-based data terms.

II. The JPEG Pipeline

The JPEG file format can compress and store one to four image components[1]. In the three-component case, the format supports the sRGB and the more commonly used YCbCr color spaces, where the Cb and the Cr components of the latter are typically down-sampled vertically and horizontally by a factor of two. The transform-quantize-dequantize-inverse-transform pipeline of JPEG is now briefly reviewed; see, e.g., [14] for a more detailed treatment and for a review of the entropy-coding scheme used in JPEG compression.

First, each image component comprising N pixels is divided into M, 8×8 blocks of pixels, whose mth block is analyzed using the two-dimensional discrete cosine basis functions to produce a coefficient vector $\mathbf{x}_m \in \mathbb{R}^{64}$. This coefficient vector is then quantized using the table $\mathbf{q}_m \in \mathbb{R}^{64}$ to generate a vector of symbols. The mid-tread, uniform quantization rule is usually employed, which produces the output $\mathbf{y}_m \in \mathbb{R}^{64}$ as

$$\mathbf{y}_m = \mathbf{round}(\mathbf{diag}(\mathbf{q}_m)^{-1}\mathbf{x}_m), \tag{2}$$

rounding each element of the vector separately. Symbol vector $\mathbf{y}_m$ is entropy-coded and appended to the JPEG codestream.

In the JPEG standard, an image component uses only one quantization table, i.e., $\mathbf{q}_1 = \mathbf{q}_2 = \cdots = \mathbf{q}_M$. Also, whenever the YCbCr color space is employed, it is customary to use a total of two quantization tables, one for the Y component, and another for the Cb and Cr components. The quantization tables are included in the JPEG header.

The standard decoding of a JPEG image involves decoding each symbol vector $\mathbf{y}_m$ from the codestream and dequantizing them according to

$$\hat{\mathbf{x}}_m = \mathbf{diag}(\mathbf{q}_m)\mathbf{y}_m, \tag{3}$$

and subsequently, approximations of the original image blocks are synthesized from the dequantized coefficients. Using (2) and (3), we see that each original coefficient vector $\mathbf{x}_m$ is in the convex set

$$C_m = \{\hat{\mathbf{x}}_m + \mathbf{v} : -(1/2)\mathbf{q}_m \leq \mathbf{v} < (1/2)\mathbf{q}_m\}, \tag{4}$$

so choosing $\hat{\mathbf{x}}_m$ as the dequantization point of $\mathbf{y}_m$ is optimal in the sense of minimum mean square reconstruction error if the DCT coefficients are uncorrelated across the blocks, and the coefficients are uniformly distributed in each C_m.

One can regard JPEG's 8×8 block size as a compromise between the transform's ability to adapt to the non-stationary nature of the image signal's underlying stochastic process, and deriving a sufficient coding gain after decorrelating the image samples. The fixed block size approach requires no additional partition information to be communicated to the decoder since the partition implied is known by the decoder in advance.

However, as is usually the case with natural images, patterns and textures can appear at various scales and continue across multiple blocks. In this case, the JPEG encoder, with its fixed block transform, is not able to exploit the correlations across these blocks. This implies that the coded DCT coefficients may still be correlated, and one may devise a regularization model that exploits this remaining inter-block correlation to obtain better estimates of the original transform coefficients.

III. The Decoding Problem

From a slightly different perspective, one can view the task of decoding an image as an inverse problem, where the goal of the decoder is to recover the original un-quantized transform coefficients from the observed quantized ones. In general, the process of quantization involves a many-to-one mapping, and its inverse, de-quantization, a one-to-many mapping, rendering the inverse problem non-trivial.

First, consider a naïve form of the decoding problem, where one's objective is to de-quantize a set of quantized coefficients to some plausible values. One can express such a problem as

$$\min \quad f(\mathbf{x}) = \delta_C(\mathbf{x}), \tag{5}$$

in which $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M) \in \mathbb{R}^{64M}$ is the vector of all transform coefficients to decode, and $\delta_{\mathcal{C}} : \mathbb{R}^{64M} \to \{0, \infty\}$ is the characteristic function

$$\delta_{\mathcal{C}}(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \in C, \\ \infty & \text{otherwise,} \end{cases} \tag{6}$$

of the box (or $64M$-orthotope)

$$C = \mathrm{cl}(C_1 \oplus C_2 \oplus \cdots \oplus C_M), \tag{7}$$

the closure of the direct sum of the sets C_m in (4).

Problem (5) is ill-posed[2] in the sense of Hadamard, since it admits multiple solutions. The quantization process involves a many-to-one mapping, and the inverse problem (5) is devoid of a unique solution. Therefore, prior assumptions regarding the solution are required for decoding to form a well-posed problem. Using quadratic regularization, we obtain a problem of the form

$$\min \quad f(\mathbf{x}) = \delta_C(\mathbf{x}) + (1/2)\mathbf{x}^*\mathbf{\Lambda}\mathbf{x}. \tag{8}$$

In the context of Bayesian point estimation, the first term of the objective of (8) can be regarded as the log-likelihood

[1]The JPEG format can compress and store an alpha channel component in addition to three color components, although this is rarely done.

[2]A problem is well-posed if a unique solution exists, and the behavior of the unique solution changes continuously with the initial conditions.

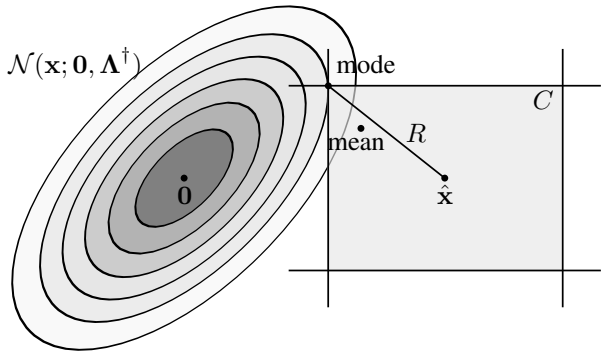

Fig. 1. The posterior mode, or the maximum a posteriori estimate is always attained at the boundary ∂C of C. The posterior mean, or the minimum mean square error estimate lies away from ∂C and closer to the center $\hat{\mathbf{x}}$ of C. We seek our solution on the line segment R connecting the mode and $\hat{\mathbf{x}}$.

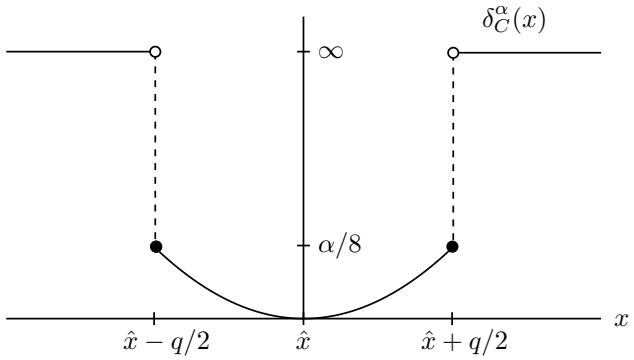

Fig. 2. Augmented characteristic function $\delta_C^\alpha : \mathbb{R} \to \{\mathbb{R}_+ \cup \infty\}$ of the set $C = [\hat{x} - q/2, \hat{x} + q/2]$ (in one dimension.) The penalty at the boundaries of the interval or set is $\alpha/8$. When $\alpha = 0$, this function is equivalent to the usual characteristic function δ_C of set C.

of the parameters $\mathbf{x}$, given the observations of some transform coefficients. The second term can be interpreted as the log of a prior distribution, where the particular prior being assumed in our case is evidently $\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{\Lambda}^\dagger)$, i.e., a zero-mean Gaussian distribution with covariance $\mathbf{\Lambda}^\dagger$. The solution of problem (8) can therefore be expressed equivalently as

$$\mathbf{x}^{\text{mode}} = \underset{\mathbf{x}\in C}{\operatorname{argmax}}\ \mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{\Lambda}^\dagger), \tag{9}$$

known as the maximum a posteriori solution or the posterior mode.

However, the posterior mode minimizes reconstruction error based on the Hamming distortion measure, not the squared error distortion measure. To minimize the reconstruction error using the latter, we require the computation of the conditional mean of $\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{\Lambda}^\dagger)$ over C, that is,

$$\mathbf{x}^{\text{mean}} = \int_C \mathbf{x} \exp(-(1/2)\mathbf{x}^*\mathbf{\Lambda}^\dagger\mathbf{x})\ \mathrm{d}\mathbf{x}, \tag{10}$$

also known as the minimum mean square error solution or the posterior mean. The posterior mean is more difficult to compute due to the integral in (10). The posterior mode, on the other hand, is easier to compute but always lies on the boundary of C unless all the coefficients belong to their respective zero bins. This is illustrated in Fig. 1.

Therefore, we appeal geometrically, and reason that the posterior mean must lie away from the boundary and closer to the center $\hat{\mathbf{x}}$ of C. This suggests that we construct a mean square error-like data term by augmenting (8) with an additional quadratic term as

$$\begin{aligned}\min \quad & f(\mathbf{x}) = \delta_C(\mathbf{x}) + (1/2)\mathbf{x}^*\mathbf{\Lambda}\mathbf{x} \\ & \qquad + (\alpha/2)\|\mathbf{Q}^{-1}(\mathbf{x}-\hat{\mathbf{x}})\|_2^2,\end{aligned} \tag{11}$$

denoting the vector of bin-centers and bin-widths by

$$\hat{\mathbf{x}} = (\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_M) \tag{12}$$

and

$$\mathbf{Q} = \mathbf{diag}(\mathbf{q}_1, \mathbf{q}_2, \ldots \mathbf{q}_M) \tag{13}$$

respectively. The solution of our formulation (11) lies on the line segment R shown in Fig. 1.

One is also at liberty to express problem (11) equivalently in terms of the image domain variables $\mathbf{u} = \mathbf{T}^*\mathbf{x} \in \mathbb{R}^M$:

$$\min \quad f(\mathbf{u}) = \delta_C^\alpha(\mathbf{T}\mathbf{u}) + \mathbf{u}^*\mathbb{L}\mathbf{u}, \tag{14}$$

using the similarity transform $\mathbf{\Lambda} = \mathbf{T}\mathbb{L}\mathbf{T}^*$, where $\mathbf{T}$ denotes the (orthogonal) block-DCT operator. At the same time, we introduce the lower semi-continuous, augmented characteristic function

$$\delta_C^\alpha(\mathbf{x}) = \begin{cases} (\alpha/2)\|\mathbf{Q}^{-1}(\mathbf{x}-\hat{\mathbf{x}})\|_2^2 & \text{if } \mathbf{x}\in C, \\ \infty & \text{otherwise,}\end{cases} \tag{15}$$

in which $\hat{\mathbf{x}}$ and $\mathbf{Q}$ can be interpreted as the center and the lengths of the sides of the box C respectively. Function (15) reduces to the usual characteristic function (6) of C when $\alpha = 0$. The augmented function is illustrated in Fig. 2. The data term δ_C^α, in addition to enforcing box constraints, also minimizes the distortion due to quantization, proportionate to the square of the quantization step size.

IV. Laplacians of Graphs

In classical discrete inverse problems, the Laplacian matrix usually corresponds to the discrete analog of the differential operator from the continuous function space. In this work, it can be more intuitive to interpret the Laplacian operator as representing a weighted graph, whose edge weights naturally correspond to a measure of diffusivity between the pairs of adjacent graph vertices. Our proposed solution may then be seen as a steady state or a fixed point of a non-local diffusion reaction process on this graph.

Transform coefficients vary rapidly near structural features of an image, where the assumption of stationarity no longer holds. To enforce that the Laplacian smooths only within image regions which are approximately stationary, we use the Laplacian of a weighted graph G, whose vertices $v_i \in V(G)$ are formed as

$$v_i = \begin{bmatrix} \dfrac{x_i}{\sigma_X} & \dfrac{y_i}{\sigma_Y} & \dfrac{\boldsymbol{z}_i}{\sigma_Z} \end{bmatrix}, \tag{16}$$

in which $\boldsymbol{z}_i \in \mathbb{R}^c$ is the first c transform coefficients[3] of the ith pixel block, and x_i and y_i are the top-left coordinates of

[3]The usual zig-zag scan order is assumed for the ordering of the transform coefficients.

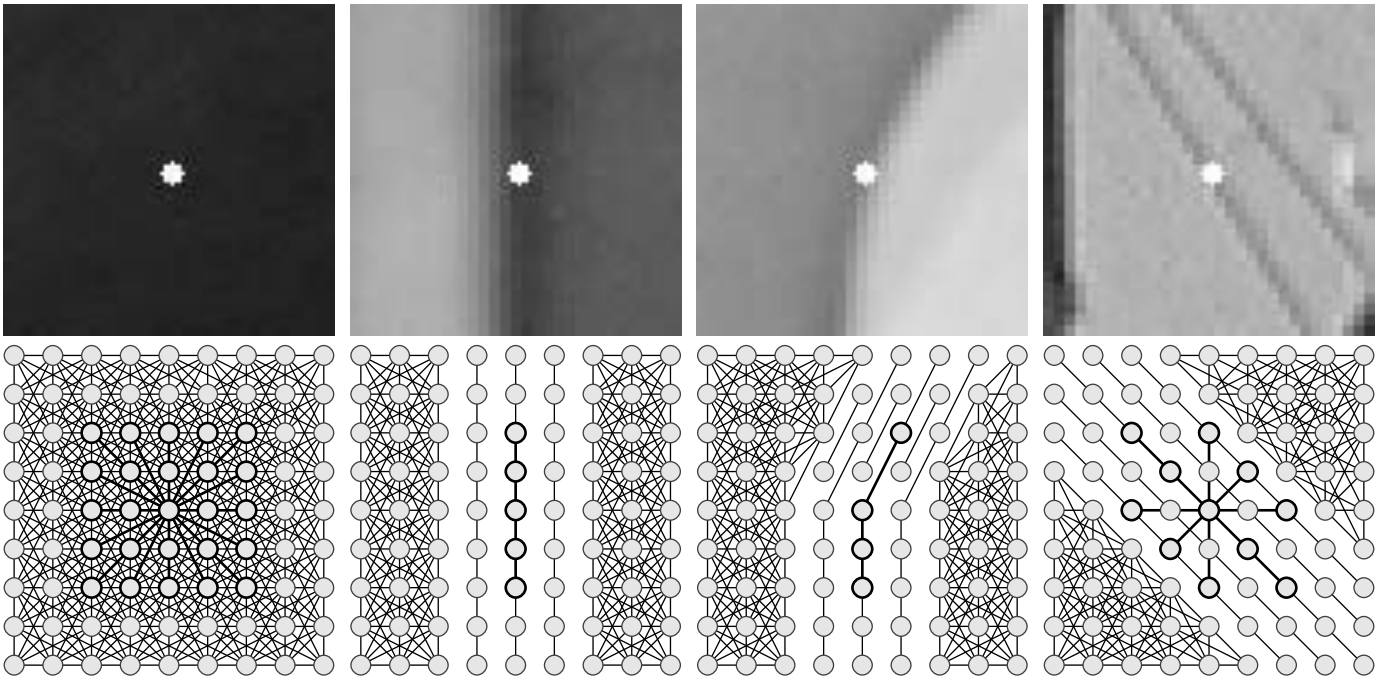

Fig. 3. Coefficient graph $G = (V, E, w)$ defined over coded and non-coded DCT blocks, shown as circles. The graph diffuses isotropically in smooth regions (a) and anisotropically in transition regions (b), (c). Graph edges need not be spatially adjacent (d). Image patches in the top row are taken from [12].

the block. A vertex v_i represents a coded DCT block when $(x_i, y_i) \in 8\,\mathbb{Z}_+^2$, and a non-coded DCT block otherwise. We define the edges of G as

$$E(G) = \{(v_i, v_j) \in V(G) : |v_i - v_j| \leq 2\}, \tag{17}$$

in which $|\cdot|$ is a norm on $\mathbb{R}^{n+2}$. The graph edges are weighted using

$$w(v_i, v_j) = \exp(-|v_i - v_j|^2/2), \tag{18}$$

such that, in particular, (17) represents pairs of vertices with distance equal to at most two standard deviations. This graph has Gaussian weighted edges with uniform variance, and the scaling of the vertices themselves via $\sigma_{X,Y}$ and σ_Z establishes the scale of the model. Fig. 3 illustrates G around typical image patches when $\sigma_X = \sigma_Y = 1$.

The Laplacian of the graph G is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, in terms of the degree $\mathbf{D}$ and the adjacency $\mathbf{A}$ matrices of G. The use of the Gaussian weight function (18) renders the adjacency matrix $\mathbf{A}$ a Gaussian filter matrix, so the mapping $\mathbf{x} \mapsto \mathbf{L}\mathbf{x}$ can be evaluated efficiently during gradient descent, using one element-wise multiplication for $\mathbf{D}$ and one application of fast Gaussian filtering for $\mathbf{A}$. We use the KD-tree implementation [15] for Gaussian filtering in our work.

One can also relate our graph Laplacian to the non-local means filter of Buades et al. [12], which can be expressed as the low-pass filter matrix $\mathbf{D}^\dagger\mathbf{A}$. For non-local means, a more common choice of vector $\boldsymbol{z} \in \mathbb{R}^c$ (16) is the intensities [12] or the KLT coefficients [15] of a c-pixel block. Due to Parseval's relation, all representations of $\boldsymbol{z}$ lead to an identical matrix $\mathbf{D}^\dagger\mathbf{A}$ when $c = 64$ (assuming 8×8 blocks.) The non-local means algorithm de-noises image pixel intensities directly, by associating each graph vertex with an image pixel.

In our image decoding problem, noise is introduced by destructive quantization of the high-frequency coefficients of the DCT, suggesting that these coefficients be discounted when measuring the similarity between a pair of blocks. Similarly [15] modifies the non-local means algorithm by first projecting the pixel intensities onto a lower-dimensional Karhunen-Loève subspace, thereby robustifying the similarity measure against high-frequency noise. The two approaches are ultimately related since the type-II DCT is asymptotically equivalent to the KLT for Markov signals of all orders as the support of the block transform tends to infinity [16, §3]. We observe that $c = 10$ produces good results at all quality factors.

Since coded and non-coded DCT coefficients are mostly correlated within each sub-band, we define our graph signal itself as a (redundant) DCT sub-band image. Consequently, we can construct 64 such signals for a given image if an 8×8 DCT is used. Compared with the graph signal of the non-local means algorithm, which is the actual image, our graph signals allow assignment of different regularization weights for each sub-band. In Section VII, we see that the regularization weight required for different sub-bands vary substantially, with low frequency sub-bands requiring larger weights than high frequency ones. When the regularization weights are identical for all sub-bands, regularizing the sub-band signals is equivalent to regularizing the image signal directly. While other redundant transforms can be used to generate the graph signals, the DCT allows the 64 sub-bands to be generated efficiently using the fast cosine transform implementation.

From a graph-theory perspective, the matrix $\mathbf{I} - \mathbf{D}^\dagger\mathbf{A}$ is known as the random-walk normalized Laplacian, which we can also express as $\mathbf{D}^\dagger\mathbf{L}$. Normalizing $\mathbf{L}$ as $\mathbf{D}^\dagger\mathbf{L}$ is useful

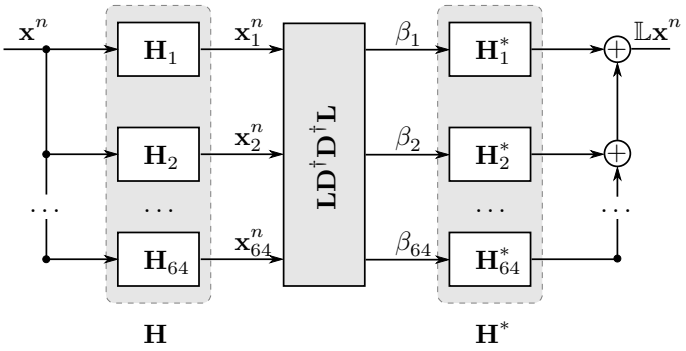

Fig. 4. Redundant filter-bank interpretation of the gradient-descent step. The input image $\mathbf{x}_m^n$ is analyzed with the filters $\{\mathbf{H}_m\}$ to produce the sub-band images $\{\mathbf{x}_m^n\}$. The images $\{\mathbf{x}_m^n\}$ are filtered by $\mathbf{LD}^\dagger\mathbf{D}^\dagger\mathbf{L}$, then scaled by $\{\beta_m\}$, and synthesized with the filters $\{\mathbf{H}_m^*\}$ to produce $\mathbb{L}\mathbf{x}^n$.

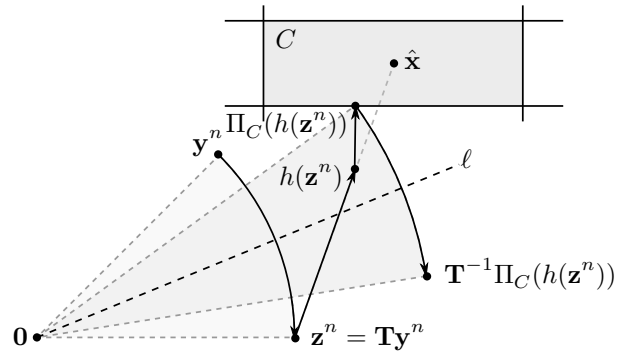

Fig. 5. Geometric interpretation of the proximal step in $\mathbb{R}^2$, where the DCT $\mathbf{T}$ reflects about line ℓ. The input image $\mathbf{y}^n$ is reflected to its mirror image $\mathbf{z}^n = \mathbf{T}\mathbf{y}^n$. Then, $\mathbf{z}^n$ gravitates towards $\hat{\mathbf{x}}$ to produce $h(\mathbf{z}^n)$, and projected onto C. The final output is the mirror image of this projection about ℓ.

for regularization, since $\mathbf{L}$ has the tendency to regularize a certain block more strongly when it has a lot of similarly textured adjacent blocks. Despite the non-symmetry, $\mathbf{D}^\dagger\mathbf{L}$ is positive semi-definite since it is related to the symmetric-normalized Laplacian by a similarity transformation. We can symmetrize $\mathbf{D}^\dagger\mathbf{L}$ by forming $\mathbf{LD}^\dagger\mathbf{D}^\dagger\mathbf{L}$, which we refer to as the symmetric random-walk-normalized Laplacian. Another choice of symmetrization is $\mathbf{LD}^\dagger\mathbf{L}$, but this matrix presents a difficulty when computing its largest eigenvalue, as is required for choosing the optimal descent step size in Section VI.

The symmetric normalized Laplacian $\sqrt{\mathbf{D}^\dagger}\mathbf{L}\sqrt{\mathbf{D}^\dagger}$ penalizes even the constant trends in the underlying solution, which can be undesirable for image reconstruction problems. We can see this from the fact that the eigenvector associated with its 0 eigenvalue is the vector $\sqrt{\mathbf{D}}\mathbf{1}$. By contrast, the eigenvector of the Laplacian $\mathbf{LD}^\dagger\mathbf{D}^\dagger\mathbf{L}$ associated with the eigenvalue 0 is the constant vector $\mathbf{1}$, such that DC offsets are neither penalized nor reshaped.

V. The Proposed Model

Using the operator $\mathbf{LD}^\dagger\mathbf{D}^\dagger\mathbf{L}$, we state our image decoding problem as

$$\min \quad f(\mathbf{u}) = \delta_C^\alpha(\mathbf{Tu}) + \textstyle\sum_{m=1}^{64} \beta_m \|\mathbf{D}^\dagger\mathbf{L}\mathbf{u}_m\|_2^2, \qquad (19)$$

in which $\mathbf{u}_m$ is the mth redundant DCT sub-band of $\mathbf{u}$, and β_m is a regularization weight for this sub-band. Let $\mathbf{H}_m^*$, $1 \le m \le 64$, be a filter whose impulse response is the mth basis function of the two-dimensional inverse 8×8 DCT. Since each sub-band image $\mathbf{u}_m = \mathbf{H}_m\mathbf{u}$, problem (19) is equivalent to

$$\min \quad f(\mathbf{u}) = \delta_C^\alpha(\mathbf{Tu}) + \mathbf{u}^*\mathbb{L}\mathbf{u}, \qquad (20)$$

denoting our "graph Laplacian" by

$$\mathbb{L} = \textstyle\sum_{m=1}^{64} \beta_m \mathbf{H}_m^*\mathbf{LD}^\dagger\mathbf{D}^\dagger\mathbf{L}\mathbf{H}_m; \qquad (21)$$

this shows that our overall problem can simply be understood as a graph Laplacian-regularized inverse problem in the image pixel domain. Since the DCT is orthonormal, (21) reduces to $\mathbb{L} = 64\mathbf{LD}^\dagger\mathbf{D}^\dagger\mathbf{L}$ owing to linearity when $\beta_m = 1$ for all $m = 1, 2, \ldots, 64$.

To prove that $\mathbb{L}$ is a graph Laplacian[4] matrix, it suffices to show that $\mathbb{L}$ has the eigenpair $(\mathbf{1}, 0)$ (from which the positive semi-definiteness of $\mathbb{L}$ follows). Indeed, the filter response of $\mathbf{H}_m$ to the constant vector $\mathbf{1}$ is

$$\mathbf{H}_1\mathbf{1} = 8 \cdot \mathbf{1}, \quad \mathbf{H}_m\mathbf{1} = \mathbf{0}, \quad 2 \le m \le 64, \qquad (22)$$

and this implies

$$\mathbf{H}_m^*\mathbf{LD}^\dagger\mathbf{D}^\dagger\mathbf{L}\mathbf{H}_m\mathbf{1} = \mathbf{0} \qquad (23)$$

for all m. It follows from (21) that $(\mathbf{1}, 0)$ is an eigenpair of $\mathbb{L}$.

We note that our Laplacian $\mathbb{L}$ regularizes the modulation in the individual sub-band components, rather than the variation in the image pixel intensity itself. Laplacian regularizers that operate directly on the image often encode one's assumption that the image is Gauss-Markov[5], i.e., a pixel is well-predicted given its immediate neighbors. This assumption is unreasonable for natural images, which often exhibit patterns and texture across multiple scales. Our regularization term in (19) assumes instead that the image is observed from a correlated Gaussian process (with higher order Markovity.) This is evident from the fact that the adjacencies in $\mathbb{L}$ extend well beyond immediate neighbors in the image pixel domain.

VI. Numerical Optimization

To recapitulate, our optimization problem is

$$\min \quad f(\mathbf{u}) = \delta_C^\alpha(\mathbf{Tu}) + (1/2)\mathbf{u}^*\mathbb{L}\mathbf{u}, \qquad (24)$$

and since $\delta_C^\alpha : \mathbb{R}^n \to \mathbb{R}$ is closed proper convex, we solve our problem using the accelerated version of the proximal gradient descent method [17], [18]. For our problem, the method amounts to iterating the update steps

$$\mathbf{x}^n = \mathbf{u}^{n-1} + \theta^{n-1}(\mathbf{u}^{n-1} - \mathbf{u}^{n-2}) \qquad (25)$$

$$\mathbf{y}^n = \mathbf{x}^n - \lambda\mathbb{L}\mathbf{x}^n \qquad (26)$$

$$\mathbf{u}^n = \operatorname{argmin} \ \delta_C^\alpha(\mathbf{Tu}) + (1/2\lambda)\|\mathbf{u} - \mathbf{y}^n\|_2^2 \qquad (27)$$

until a convergence criterion is met. We refer to the update steps (25), (26) and (27) as the extrapolation, gradient-descent

[4]Note that the graph represented by $\mathbb{L}$ can have many negatively-weighted edges since $\mathbb{L}$ involves both a fourth-difference operator $\mathbf{L}_{\text{sw}}$ and filters $\mathbf{H}_m$.

[5]Markovity of relatively low order is assumed by sparse regularizers.

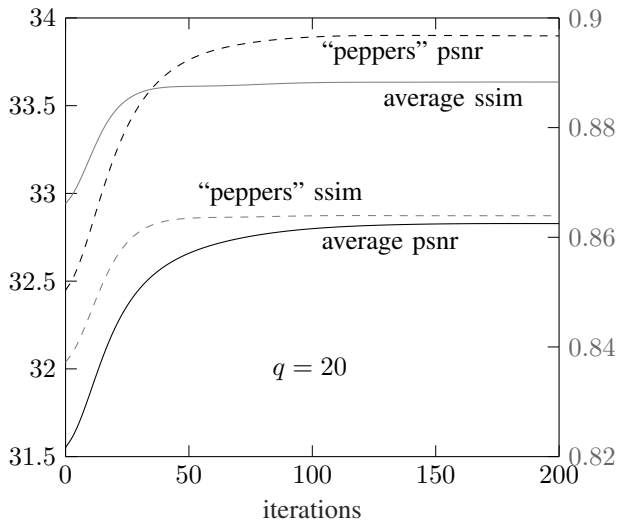

Fig. 6. Objective quality of the training images measured across iterations at quality factor $q = 20$. In general, the curves need not be monotonic since neither PSNR nor SSIM are the optimization objectives. The algorithm itself converges in the true objective value with an $O(1/K^2)$ rate of convergence.

and proximal steps respectively. Here, $\theta^n \in [0, 1)$ denotes an extrapolation parameter, and $\lambda > 0$, a descent step-size. This method converges in the objective value with an $O(1/N^2)$ rate if the parameters θ^n and λ are chosen carefully; see [19, §6.3].

The gradient-descent step (26) entails the evaluation of the mapping $\mathbf{x} \mapsto \mathbb{L}\mathbf{x}$. The linear map $\mathbb{L}$ can be expressed equivalently as

$$\mathbb{L} = \mathbf{H}^*(\mathbf{diag}(\boldsymbol{\beta}) \otimes \mathbf{L}\mathbf{D}^\dagger\mathbf{D}^\dagger\mathbf{L})\mathbf{H}, \tag{28}$$

in which $\mathbf{H} \in \mathbb{R}^{64N\times N}$ is the redundant DCT which outputs 64 transform coefficient vector at every image pixel, and $\boldsymbol{\beta} \in \mathbb{R}^{64}$ is the sub-band regularization weights. From (28), one sees that the mapping $\mathbf{x} \mapsto \mathbb{L}\mathbf{x}$ can be evaluated by (a) analyzing $\mathbf{x}$ with the fast cosine transform to generate a 64-band image $\mathbf{H}\mathbf{x}$, (b) Laplacian-filtering $\mathbf{H}\mathbf{x}$ via the fast Gauss transform and scaling the filtered bands using $\boldsymbol{\beta}$ to produce $\mathbf{z}$, and, (c) synthesizing the single-band signal $\mathbf{H}^*\mathbf{z}$ with the fast inverse cosine transform. This is summarized as a block diagram in Fig 4.

The proximal step (27) can be written as $\mathbf{u}^n = \mathbf{T}^*\Pi_C^\xi(\mathbf{z}^n)$ using $\xi = \lambda\alpha$ and $\mathbf{z}^n = \mathbf{T}\mathbf{y}^n$, where

$$\Pi_C^\xi(\mathbf{z}^n) = \operatorname{argmin}\ \delta_C^\xi(\mathbf{x}) + (1/2)\|\mathbf{x} - \mathbf{z}^n\|_2^2, \tag{29}$$

appealing to the invariance of the 2-norm under the orthogonal transform $\mathbf{T}$. To compute (29), we combine the quadratic term in (15) with the one in (29) and obtain

$$\begin{aligned}\Pi_C^\xi(\mathbf{z}^n) &= \operatorname{argmin}\ \delta_C(\mathbf{x}) + (1/2)\|\mathbf{x} - h(\mathbf{z}^n)\|_2^2 \\ &= \Pi_C(h(\mathbf{z}^n)),\end{aligned} \tag{30}$$

in which

$$h(\mathbf{z}^n) = (\mathbf{I} + \xi\mathbf{Q}^{-2})^{-1}(\mathbf{z}^n + \xi\mathbf{Q}^{-2}\hat{\mathbf{x}}), \tag{31}$$

and $\Pi_C(h(\mathbf{z}^n))$ is the projection of $h(\mathbf{z}^n)$ onto box C.

Fig. 5 provides a geometric interpretation of the proximal update step in $\mathbb{R}^2$, where the discrete cosine transform is the reflection across the line through the origin at angle $\pi/8$. Note that $\mathbf{u}^n$ is an image domain variable, so it does not necessarily reside inside the box C (but $\mathbf{T}\mathbf{u}^n$ must always do for all n.)

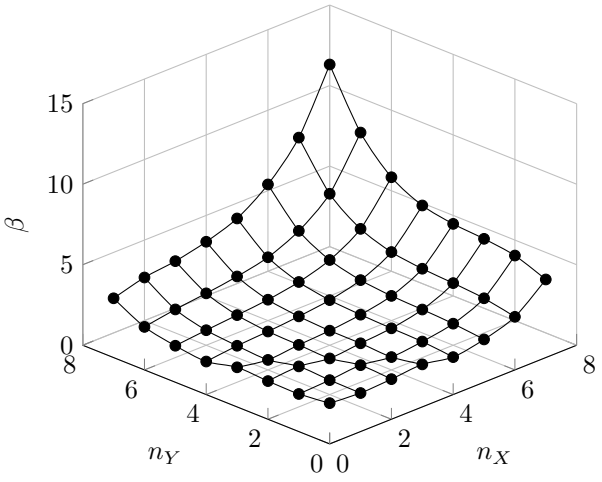

Fig. 7. Sub-band regularization weights β learned at $(\sigma_{X,Y} = 7, \sigma_Z = 38)$ across a training set of images. Each β_m is assigned inversely to the square of the average value of Laplacian norm of $\mathbf{u}_m$. The sub-band weights shown are ordered lexicographically (column-wise.)

TABLE I
MAXIMAL MAGNITUDE RESPONSES OF THE 1D DISCRETE COSINE BASIS VECTORS AND THE ASSOCIATED ANGULAR FREQUENCIES

m	$\max \lvert H_m(\omega)\rvert$	$\operatorname{argmax} \lvert H_m(\omega)\rvert$
1	2.828427124746	0.000000000000
2	2.197017447206	0.542681682359
3	2.080026042314	0.896683500259
4	2.051499497391	1.271883161185
5	2.045898905636	1.661283308095
6	2.055649543665	2.064472345659
7	2.101117117171	2.494590600955
8	2.562915447742	3.141592653590

The values of the extrapolation parameter θ^n and descent step-size λ remain to be chosen. One possible choice is to set

$$\lambda \in (0, 1/L], \quad \theta^n = n/(n + 3) \tag{32}$$

cf. [18, eq. (3.16)], in which L denotes the Lipschitz constant of $\mathbf{u} \mapsto \mathbb{L}\mathbf{u}$. The optimum descent step-size is $1/L$, so the algorithm can benefits from the knowledge of L or some low upper-bound thereof, which is now briefly derived.

The boundedness of $\mathbb{L}$ implies $L = \|\mathbb{L}\|_2$, and applying the triangle inequality on the right-hand side of (21) gives us

$$\|\mathbb{L}\|_2 \le \|\mathbf{L}\mathbf{D}^\dagger\mathbf{D}^\dagger\mathbf{L}\|_2 \left(\sum_{n=1}^{64} 2\beta_m\|\mathbf{H}_m\|_2\right) \tag{33}$$

$$= \lambda_{\max}^2(\mathbf{D}^\dagger\mathbf{L}) \left(\sum_{n=1}^{64} 2\beta_m\sigma_{\max}(\mathbf{H}_m)\right) \tag{34}$$

$$\le 8 \left(\sum_{n=1}^{64} \beta_m\sigma_{\max}(\mathbf{H}_m)\right), \tag{35}$$

the second inequality of which uses the fact that

$$1 < \lambda_{\max}(\mathbf{D}^\dagger\mathbf{L}) = \lambda_{\max}(\sqrt{\mathbf{D}^\dagger}\mathbf{L}\sqrt{\mathbf{D}^\dagger}) \le 2 \tag{36}$$

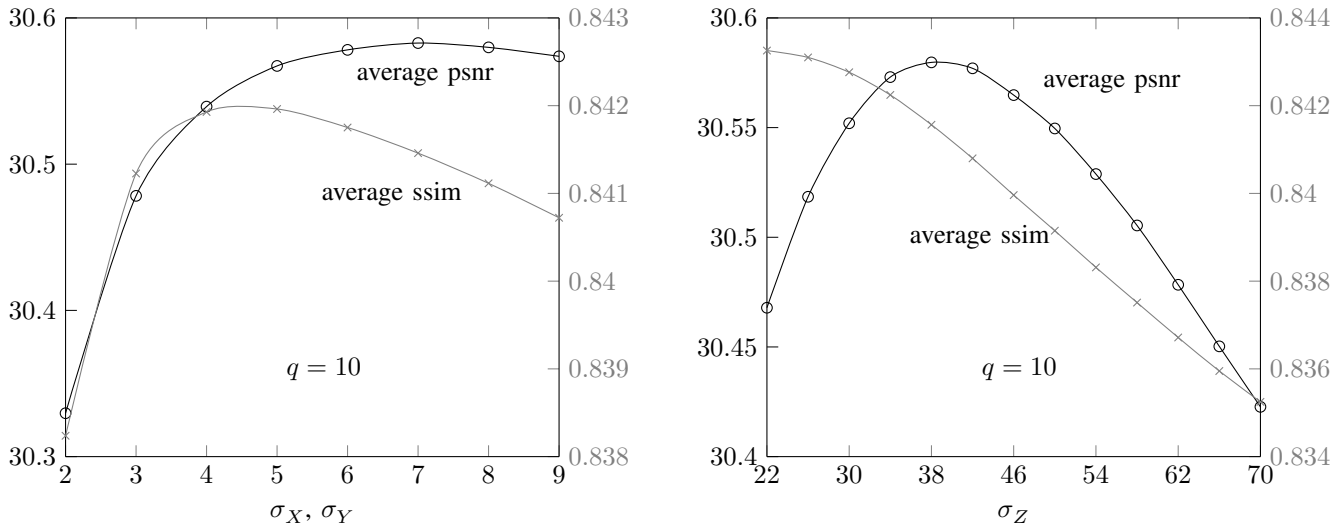

Fig. 8. The effect of varying $\sigma_{X,Y}$ (left) and σ_Z (right) on the average PSNR and SSIM at quality factor $q = 10$, keeping the other parameters fixed at their optimum values. The optimum parameter values are $(\sigma_{X,Y} = 7, \sigma_Z = 38)$, and $\alpha = 128$. PSNR and SSIM are optimized at different parameter values.

for the normalized Laplacians $\mathbf{D}^\dagger\mathbf{L}$ and $\sqrt{\mathbf{D}^\dagger}\mathbf{L}\sqrt{\mathbf{D}^\dagger}$ [20].

For a filter matrix $\mathbf{H}_m$, its maximal singular value is given by its peak magnitude response [21, §3], i.e.,

$$\sigma_{\max}(\mathbf{H}_m) = \max |H_m(\boldsymbol{\omega})|, \tag{37}$$

in which $H_m(\boldsymbol{\omega})$ is the two-dimensional discrete-time Fourier transform of the impulse response of $\mathbf{H}_m$. Since the transform is separable, maxima (37) can be computed as the products of the maximal magnitude responses of the one-dimensional cosine basis vectors. The maxima for the one-dimensional basis are computed efficiently using the Newton-Raphson or secant methods [22, §9]. The maxima are listed in Table I, alongside the angular frequencies at which they are attained. We see from (35) that our estimate of L is linear in β.

VII. Determining Parameters

To determine the parameters $\sigma = (\sigma_{X,Y}, \sigma_Z)$, the weights $\beta = (\beta_1, \beta_2, \ldots, \beta_{64})$ and α, we first learn β as a function $\beta(\sigma)$, and search for the values of α and σ that obtain the optimal image reconstruction. Since problem (24) is invariant to the scaling of α and β, it is sufficient to learn β up to an arbitrary scale factor, and later, specify α optimally for the learned β. Suppose that the optimal σ is found. From a Bayesian perspective, the regularizer in (19) should assign β_m inversely to the square of the Laplacian-norm (or energy) of the mth sub-band image, that is

$$\beta_m \propto 1/\|\mathbf{D}^\dagger\mathbf{L}\mathbf{H}_m\mathbf{u}\|_2^2, \text{ for all } m = 1, 2, \ldots, 64. \tag{38}$$

In practice, we have access only to one observation of the decoded image whose high-frequency sub-bands are severely corrupted by destructive quantization. Therefore, we learn the parameters β offline over training images. We filter the training images with the mth DCT filter to obtain their mth sub-band images, averaging their Laplacian-norms across the training set to obtain β_m. Since β is unique only up to a scale, we choose the normalization $\beta = (1 = \beta_1, \beta_2, \ldots, \beta_{64})$. We plot the learned values of β at $(\sigma_{X,Y} = 7, \sigma_Z = 38)$ and $\alpha = 128$ in Fig. 7.

Having learned the function $\beta(\sigma)$, the optimal values of σ and α are found using a three-dimensional grid search. The effect of varying σ on the average PSNR (peak signal-to-noise ratio) and SSIM (structural similarity) of the reconstructed test images is plotted in Fig. 8. It is interesting to observe that the parameter values which optimize SSIM values are significantly different from those for PSNR.

To optimize for perceived image quality, one can check the iterates visually, and stop the update steps once a satisfactory reconstruction quality is achieved. For reporting purposes, we require the objective quality measures e.g. PSNR and SSIM to converge adequately to a set precision. From Fig. 6, we see that we require about 200 iterations for the update steps for the PSNR and SSIM results to converge sufficiently.

VIII. Experimental Results

For qualitative evaluation, we compare the objective quality of reconstructed images obtained using the proposed method and standard JPEG [1], as well as a number of well-performing image de-noising algorithms—KSVD [23], BM3D [24]—and image decoding algorithms—ANCE[6] [6], AR-CNN [9] and LERaG [10]. (We use the implementations of these algorithms provided by their respective authors.) For validation, we use the Kodak lossless true color image suite [25] and images from the USC-SIPI database [26]. Fig. 9 graphs the average improvement in PSNR of the different methods, relative to the standard JPEG decoded images. Our method performs comparably to LERaG overall. The PSNR gap between LERaG and ours is 0.07 dB at $q = 5$.

Tables II and III report the PSNR and SSIM of the different methods based on the Y component of the images, at quality factors $q = 10$ and 20. It can be seen that our method performs comparably to LERaG [10] on an image-to-image basis at both

[6]ANCE code v3 is used for all experiments.

TABLE II
PSNR AND (SSIM) ACHIEVED BY DIFFERENT METHODS ON DECODED JPEG IMAGES COMPRESSED AT QF = 10.

	JPEG	KSVD [23]	BM3D [24]	ANCE [6]	AR-CNN [9]	LERaG [10]	COGL (Ours)
Airplane	29.63 (0.8438)	30.30 (0.8582)	30.48 (0.8602)	30.74 (0.8661)	29.74 (0.8668)	30.92 (0.8685)	**30.94** (**0.8693**)
Bikes	25.00 (0.7508)	25.56 (0.7727)	25.92 (0.7873)	26.10 (0.7865)	**26.39** (**0.7949**)	26.37 (0.7934)	26.22 (0.7929)
Cameraman	26.47 (0.7944)	27.06 (0.8205)	27.23 (0.8199)	27.56 (0.8296)	27.63 (0.8267)	27.76 (0.8344)	**27.86** (**0.8350**)
Fishing Boat	27.48 (0.7374)	28.06 (0.7504)	28.19 (0.7576)	28.31 (0.7619)	28.33 (0.7631)	28.39 (**0.7632**)	**28.40** (0.7613)
Girl	29.70 (0.7939)	30.41 (0.8171)	30.45 (0.8158)	30.71 (0.8269)	30.27 (0.8184)	30.89 (**0.8299**)	**30.97** (0.8293)
Hats	30.64 (0.8222)	31.72 (0.8591)	31.65 (0.8515)	31.94 (0.8633)	31.85 (0.8658)	**32.24** (**0.8682**)	32.09 (0.8657)
Lena	30.40 (0.8187)	31.70 (0.8571)	31.75 (0.8555)	31.97 (0.8613)	32.05 (0.8624)	32.04 (0.8628)	**32.10** (**0.8640**)
Lighthouse	27.79 (0.7616)	28.45 (0.7801)	28.67 (0.7820)	28.95 (0.7921)	29.11 (0.7933)	**29.43** (**0.7969**)	29.07 (0.7911)
Macaw	31.74 (0.8523)	32.91 (0.8931)	32.85 (0.8863)	33.44 (0.8996)	33.00 (0.9003)	**33.59** (0.9016)	33.47 (**0.9037**)
Peppers	30.14 (0.7846)	31.46 (0.8292)	31.40 (0.8253)	31.77 (0.8323)	31.50 (0.8307)	32.02 (0.8364)	**32.12** (**0.8395**)
Sailboat	30.08 (0.8372)	31.01 (0.8650)	31.04 (0.8614)	31.33 (0.8699)	**31.50** (**0.8740**)	31.50 (0.8715)	31.47 (0.8731)
Sailboat2	29.70 (0.8082)	30.61 (0.8393)	30.81 (0.8444)	31.04 (0.8481)	31.30 (0.8539)	**31.40** (**0.8555**)	31.28 (0.8527)
Statue	29.12 (0.7917)	29.92 (0.8284)	30.09 (0.8312)	30.20 (0.8367)	30.08 (0.8328)	30.29 (0.8396)	**30.32** (**0.8410**)
Window	29.73 (0.8532)	30.83 (0.8904)	31.05 (0.8926)	31.16 (0.8962)	31.19 (0.9012)	**31.47** (**0.9020**)	31.35 (0.9013)
Woman	29.84 (0.7649)	30.69 (0.7915)	30.78 (0.7927)	30.98 (0.8015)	30.99 (0.8023)	**31.19** (**0.8066**)	31.06 (0.8018)
Average	29.16 (0.8010)	30.05 (0.8302)	30.16 (0.8309)	30.41 (0.8381)	30.33 (0.8391)	**30.63** (**0.8420**)	30.58 (0.8415)

TABLE III
PSNR AND (SSIM) ACHIEVED BY DIFFERENT METHODS ON DECODED JPEG IMAGES COMPRESSED AT QF = 20

	JPEG	KSVD [23]	BM3D [24]	ANCE [6]	AR-CNN [9]	LERaG [10]	COGL (Ours)
Airplane	31.77 (0.8930)	32.27 (0.8921)	32.46 (0.8941)	32.83 (0.9065)	32.89 (0.9067)	32.96 (0.9064)	**33.01** (**0.9082**)
Bikes	27.30 (0.8448)	27.89 (0.8528)	28.24 (0.8626)	28.52 (0.8726)	**28.91** (**0.8785**)	28.88 (0.8768)	28.68 (0.8780)
Cameraman	28.59 (0.8563)	29.11 (0.8627)	29.38 (0.8691)	29.70 (0.8771)	29.39 (0.8654)	**30.08** (**0.8801**)	29.98 (0.8800)
Fishing Boat	29.60 (0.8236)	29.96 (0.8099)	30.16 (0.8242)	30.41 (0.8401)	**30.57** (**0.8418**)	30.45 (0.8394)	30.52 (0.8390)
Girl	31.92 (0.8570)	32.36 (0.8545)	32.48 (0.8625)	32.84 (0.8750)	32.26 (0.8620)	32.97 (**0.8771**)	**33.02** (0.8752)
Hats	33.10 (0.8826)	33.92 (0.8960)	33.96 (0.8977)	34.31 (0.9058)	34.41 (0.9063)	**34.53** (**0.9086**)	34.40 (0.9051)
Lena	32.95 (0.8736)	33.67 (0.8802)	34.04 (0.8908)	34.23 (0.8949)	34.16 (0.8930)	34.22 (0.8943)	**34.32** (**0.8959**)
Lighthouse	30.09 (0.8392)	30.59 (0.8317)	30.86 (0.8365)	31.16 (**0.8544**)	31.29 (0.8543)	**31.45** (0.8542)	31.27 (0.8531)
Macaw	34.47 (0.9046)	35.16 (0.9165)	35.49 (0.9231)	35.94 (0.9285)	35.79 (0.9266)	**36.05** (0.9291)	35.83 (**0.9294**)
Peppers	32.44 (0.8366)	33.22 (0.8505)	33.42 (0.8564)	33.67 (0.8604)	33.11 (0.8555)	33.75 (0.8612)	**33.88** (**0.8641**)
Sailboat	32.59 (0.8860)	33.30 (0.8976)	33.46 (0.9002)	33.78 (0.9071)	**33.92** (0.9078)	33.88 (0.9073)	33.90 (**0.9088**)
Sailboat2	32.24 (0.8702)	32.84 (0.8767)	33.33 (0.8922)	33.54 (0.8948)	**33.86** (0.8974)	33.81 (**0.8987**)	33.75 (0.8973)
Statue	31.48 (0.8662)	31.93 (0.8718)	32.22 (0.8815)	32.49 (0.8908)	32.11 (0.8830)	32.56 (0.8900)	**32.74** (**0.8951**)
Window	32.46 (0.9092)	33.29 (0.9249)	33.66 (0.9325)	33.84 (0.9344)	**34.13** (0.9366)	34.09 (**0.9377**)	34.04 (0.9375)
Woman	32.13 (0.8400)	32.55 (0.8363)	32.78 (0.8450)	33.08 (0.8595)	33.06 (0.8580)	**33.15** (**0.8601**)	33.09 (0.8582)
Average	31.54 (0.8655)	32.14 (0.8703)	32.40 (0.8779)	32.69 (0.8868)	32.66 (0.8848)	**32.86** (0.8881)	32.83 (**0.8883**)

quality factors. KSVD [23], BM3D [24] are image de-noising methods, and agnostic to the constraints of the image decoding problem. Their relatively low performance seem to emphasize the importance of incorporating the constraints introduced by quantization as done by the remaining methods. We optimize the parameters of the denoising algorithms KSVD and BM3D using grid search to maximize their respective average PSNRs.

For visual comparison, Fig. 10 and 11 show the reconstructed images and their differences with respect to the originals for "peppers", "Lena" and "cameraman", for AR-CNN, LERaG and our methods. Our method has a less apparent error pattern compared to AR-CNN and LERaG. In "peppers", our method exhibits less ringing artifact around the dark image border, and displays less staircase effect within the smooth surface of each pepper. For "cameraman", our error patterns are less apparent than the other two methods overall in the sky and within the cameraman. It can be seen from "Lena" and "cameraman" that LERaG enforces piece-wise smoothness even in the smooth regions to create staircasing artifacts.

The running times of the different methods are also listed in Table IV for 512×512 "Lena". All methods are timed on a single core[7] running at 2.30 GHz. Our method is about six times faster than LERaG. LERaG uses a large dictionary of image atoms for regularization [10, p. 513], so it is fundamentally slower than our method due to the dictionary matching that is involved. Both BM3D and AR-CNN are, in turn, an order of magnitude or two faster than our method. We note that BM3D is a non-iterative filter, and its running time is comparable to that of one iteration of our method. Since our formulation is convex, our proposed method may benefit from a warm-start using the BM3D solution as the initialization.

TABLE IV
RUNNING TIMES OF DIFFERENT METHODS ON A SINGLE CORE FOR 512×512 "LENA"

	KSVD	BM3D	ANCE	AR-CNN	LERaG	Proposed
$q = 10$	114.9s	1.8s	130.1s	10.6s	747.1s	119.7s
$q = 20$	95.7s	1.7s	132.4s	9.9s	721.3s	118.4s

Despite our minimalistic formulation, the results seem to indicate that our formulation is sound. ANCE, AR-CNN and

[7]Intel Xeon CPU E5-2699 v3 (45M Cache, 2.30GHz)

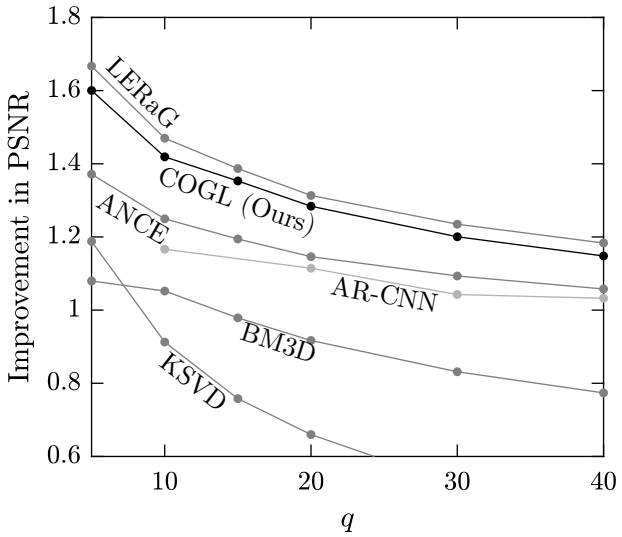

Fig. 9. Average improvement in PSNR (dB) achieved by different methods relative to standard JPEG decoding. The PSNR gap between LERaG and the proposed method is 0.07 dB at $q = 5$. The average PSNR of the JPEG decoded images are $26.47, 29.16, 30.58, 31.54, 32.89$, and 33.82 dB at $q = 5, 10, 15, 20, 30$, and 40, respectively.

LERaG are three best-performing image de-blocking methods at the time of submission. Our proposed algorithm compares favorably, with an average improvement in PSNR of 1.40 dB at $q = 10$ and 1.30 dB at $q = 20$, compared with the standard JPEG image decoder.

IX. Relation to Other Methods

Different image decoding algorithms are ultimately related since they all alternate between the low-pass filtering and the projection steps. This approach can be traced back to the early POCS method of Zakhor [2] in which the update steps comprise a projection onto a smooth Paley-Wiener subspace and another onto the quantization bins. Our method, ANCE [6] and LERaG [10] are examples of the so-called proximal methods, which replace the low-pass projection with a descent step. (Proximal methods can thus be regarded as a generalization of POCS [27].) Therefore, it is instructive to point out the similarities between our algorithm and others that perform different update steps.

While LERaG [10] and our method both use graph-based regularization, the two graphs are quite different. Our graph is constructed in the DCT domain, and it regularizes the DCT sub-band images. On the other hand, the graph used by [10] is constructed in the image domain, and it regularizes the image signal directly in the pixel domain. Our graph essentially enforces patch similarity similarly to non-local means denoising [12] while LERaG enforces pixel-similarity more similarly to the bilateral filter; see [10, eq. (20)]. However, the pixel graph of LERaG does not regularize the solution well in the absence of a second regularizer, which is based on a learned dictionary of image atoms [10, p. 522]. By contrast, our coefficient graph does not depend on a second regularizer.

Although ANCE is not originally expressed as a graph-based algorithm, its update steps can be re-written in a form which resemble ours. Denoting by $\mathbf{u}^n$, the image iterate from the nth iteration, the update steps of ANCE in the nth iteration can be re-cast as (see [6, Algorithm 1]):

for $m = 0, 1, \ldots, 63$,

$$\mathbf{u}_m^n = \mathbf{T}_m^* \mathbf{D}_m^{n-1\dagger} \mathbf{A}_m^{n-1} \mathbf{T}_m \mathbf{u}^{n-1} \tag{39}$$

$$\mathbf{x}_m^n = \underset{\mathbf{x}}{\operatorname{argmin}} \; (\mathbf{x} - \mathbf{T}_m \mathbf{u}_m^n)^* \mathbf{\Gamma}_m^{n\dagger} (\mathbf{x} - \mathbf{T}_m \mathbf{u}_m^n) + \mathbf{x}^* \mathbf{\Lambda}^\dagger \mathbf{x} + \delta_C(\mathbf{T}_0 \mathbf{T}_m^* \mathbf{x}) \tag{40}$$

end

$$\mathbf{u}^n = (1/64) \textstyle\sum_{m=0}^{63} \mathbf{T}_m^* \mathbf{x}_m^n \tag{41}$$

In (39)–(41), we identify via $0 \leq m < 64$, the set of 8×8 blocks whose top left pixel coordinates (x, y) belong to the set

$$Z_m = 8\,\mathbb{Z}_+^2 + (m\%8, \lfloor m/8 \rfloor) \tag{42}$$

(and note that Z_0 is associated with the set of coded blocks.)

The matrix $\mathbf{T}_m$ (39) is the shifted block-DCT for the mth shift. In (40), $\mathbf{\Gamma}_m^n$ is a matrix of the sub-band auto-variances of $\mathbf{T}_m \mathbf{u}_m^n$, and $\mathbf{\Lambda}$ is a matrix of the (auto-)variances of a prior coefficient distribution. In (39), $\mathbf{D}_m^{n-1}$ and $\mathbf{A}_m^{n-1}$ are respectively the degree and the adjacency matrices of the graph G_m^{n-1}, whose vertices $v_i \in V(G_m^{n-1})$ are given by:

$$v_i = \left[\frac{x_i}{\sigma_X} \quad \frac{y_i}{\sigma_Y} \quad \frac{\boldsymbol{z}_i^{n-1}}{\sigma_Z} \right], \tag{43}$$

in which $\boldsymbol{z}_i^{n-1}$ are the transform coefficients of the $n-1$th iterate $\mathbf{u}^{n-1}$, in the 8×8 block located at $(x_i, y_i) \in Z_m$. The edges of G^{n-1} and their weights are identical to ours; see (18) and (17). The set C in (40) is also defined identically to (7). A notable differences between ANCE and our method is that we project the averaged iterate (27), whereas ANCE averages the projected iterate (40), (41). Further, our Gaussian prior is fully specified by our graph Laplacian $\mathbb{L}$, whereas ANCE essentially involves both a graph Laplacian and a separate Gaussian prior parameterized by $\mathbf{\Lambda}$.

In each iteration, the update for the edge weights $\mathbf{A}^{n-1}$ is related to the method of iteratively re-weighted least-squares [28] or the lagged diffusivity method [29]. Since ANCE uses a Gaussian function to measure block similarity, the re-weighting procedure in (39) is identical to the one required for solving a non-convex Welsch loss-regularized model[8] [30], but without employing a global optimization scheme needed to overcome the non-convexity.

Since $\mathbf{A}_m^n = \mathbf{D}_m^n - \mathbf{L}_m^n$ in (39), our first observation is that update (39) is equivalent to the descent

$$\mathbf{u}_m^n = \mathbf{u}^{n-1} - \lambda \mathbf{T}_m^* \mathbf{D}_m^{n-1\dagger} \mathbf{L}_m^{n-1} \mathbf{T}_m \mathbf{u}^{n-1} \tag{44}$$

using the step size $\lambda = 1$. Since $L = \|\mathbf{D}_m^{n\dagger} \mathbf{L}_m^n\|_2 \leq 2$, this step size $\lambda \leq 2/L$ is the optimal one when the unaccelerated form of the proximal gradient method is used. Update step (40) is therefore equivalent to

$$\mathbf{x}_m^n = \underset{\mathbf{x}}{\operatorname{argmin}} \; (1/\lambda)(\mathbf{x} - \mathbf{T}_m^* \mathbf{u}_m^n)^* \mathbf{\Gamma}_m^{n\dagger} (\mathbf{x} - \mathbf{T}_m^* \mathbf{u}_m^n) + \mathbf{x}^* \mathbf{\Lambda}^\dagger \mathbf{x} + \delta_C(\mathbf{T}_0 \mathbf{T}_m^* \mathbf{x}), \tag{45}$$

[8] In the robust estimation literature, the loss function that corresponds to the Gaussian weighting function is known as the Welsch loss.

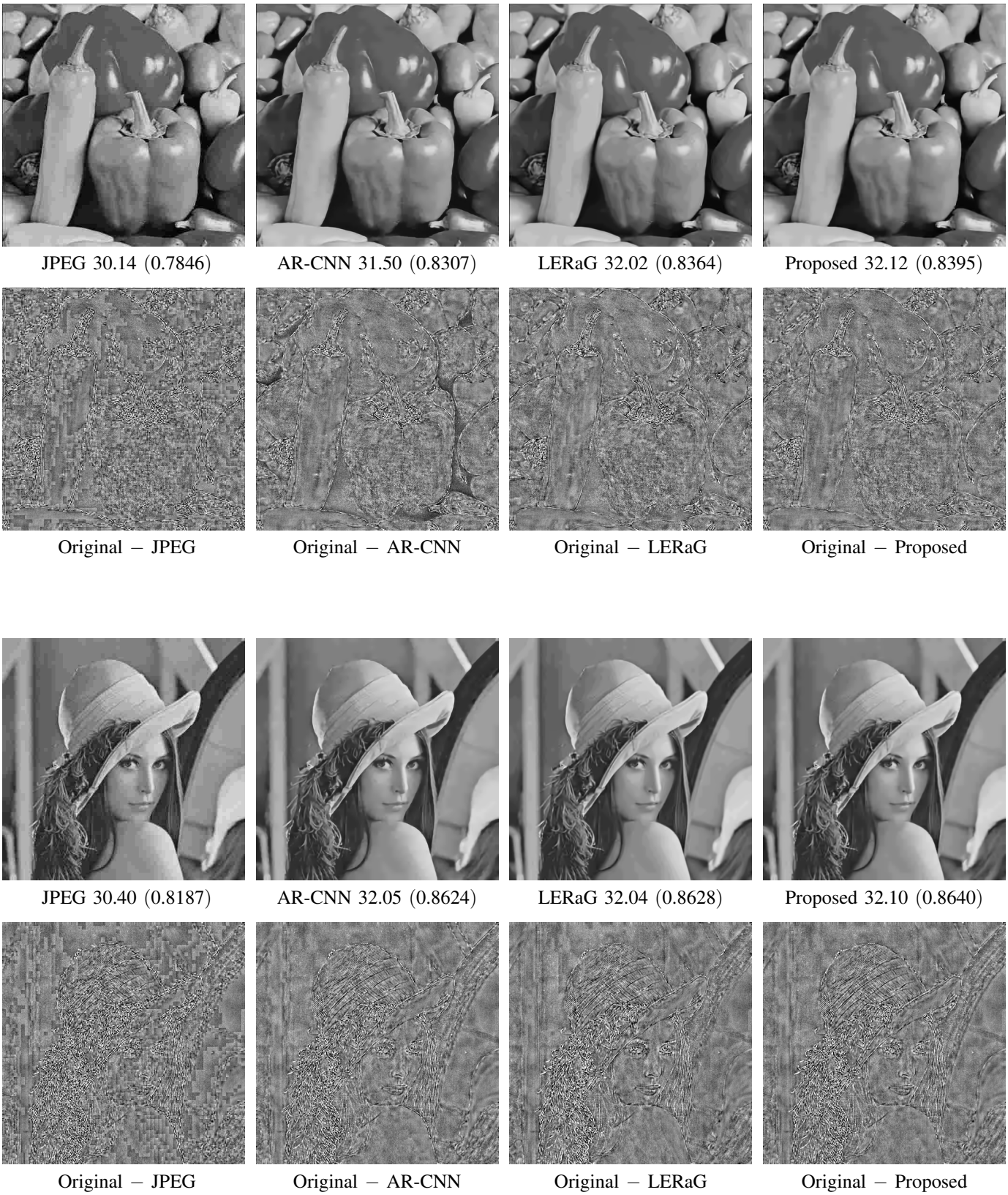

Fig. 10. Visual comparison of the reconstructed 512×512 images "peppers" and "Lena" at quality factor $q = 10$ using different methods. The pixel values of the difference images are scaled by a factor of 6 to aid presentation.

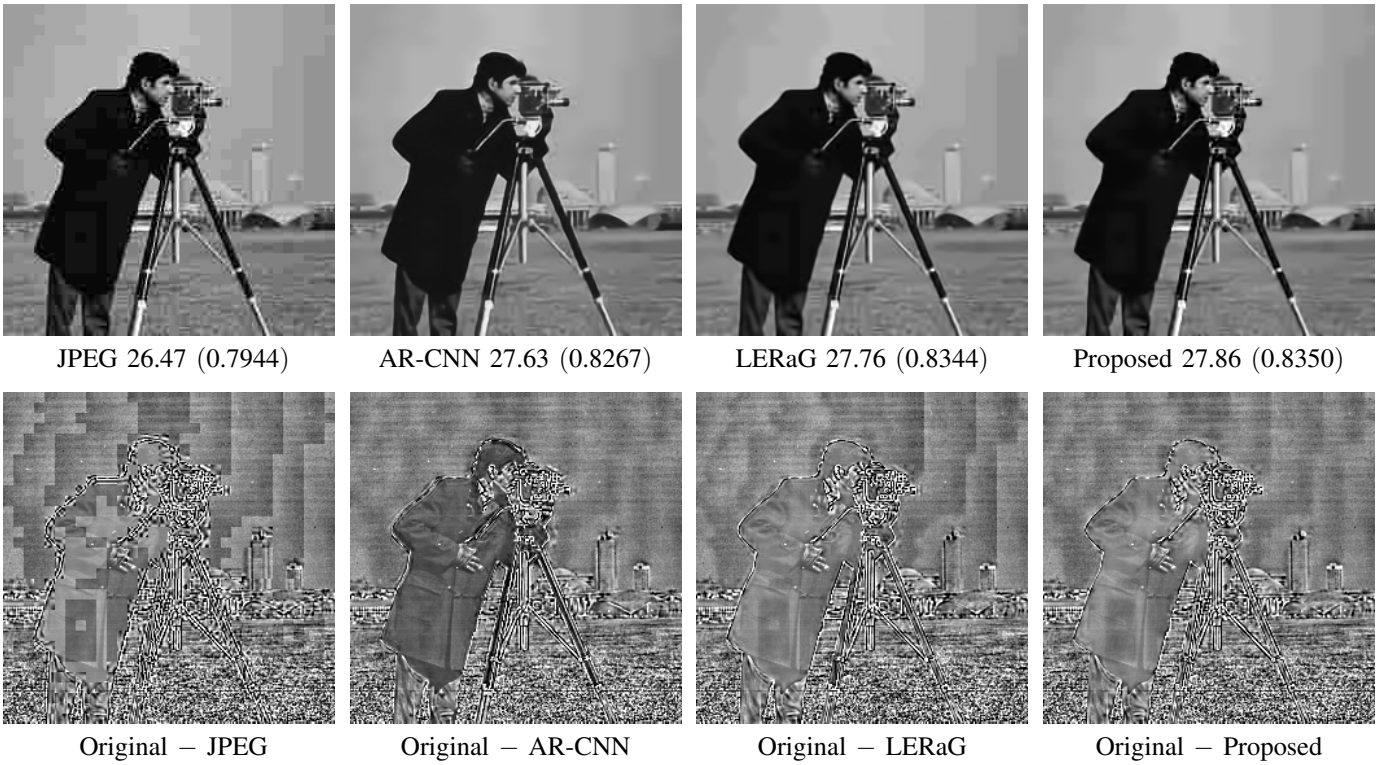

Fig. 11. Visual comparison of the reconstructed 256×256 image "cameraman" at quality factor $q = 10$ using different methods. The pixel values of the difference images are scaled by a factor of 6 to aid presentation.

resembling our proximal step (27). Here, (45) can be interpreted as a proximal mapping with the proximal point $\mathbf{T}_{\mathbf{m}}^{*}\mathbf{u}_{m}^{n}$. The involvement of $\mathbf{\Gamma}_{m}^{n\dagger}$, however, can be harder to justify from this proximal descent perspective[9]; see [27].

Based on this analysis, we recommend modifying ANCE to use $\mathbf{\Gamma}_{m}^{n} = \mathbf{I}$ and $\mathbf{D}_{m}^{n-1} = \mathbf{D}_{m}^{0}$, $\mathbf{A}_{m}^{n-1} = \mathbf{A}_{m}^{0}$ for all n, thus guaranteeing convergence to a global optimum. One subtler point is that no quadratic form of $\mathbf{u}$ has the gradient $\mathbf{T}_{m}^{*}\mathbf{D}_{m}^{\dagger}\mathbf{L}_{m}\mathbf{T}_{m}\mathbf{u}$, since $\mathbf{T}_{m}^{*}\mathbf{D}_{m}^{\dagger}\mathbf{L}_{m}\mathbf{T}_{m}$ is not symmetric. It is thus unclear which (quadratic) regularizing function is being descended in update step (44). Symmetrizing $\mathbf{D}_{m}^{\dagger}\mathbf{L}_{m}$ similarly to our method may be beneficial for convergence.

X. Decoding M-JPEG Video

Motion-JPEG is a video compression format in which each frame of a video sequence is compressed separately as a JPEG image. Independent compression of each frame imposes lower processing and memory requirements on the hardware, which renders Motion-JPEG particularly suitable for low-power and low-delay applications such as surveillance.

To optimize the decoding of M-JPEG video, one can exploit the temporal redundancy exhibited by the video, as well as the spatial redundancy exhibited by individual frames. Võ and Nguyen [32] propose to first estimate the motion between a target frame and multiple references, and obtain an improved estimate of the target frame by averaging it with the reference frames along valid motion trajectories. We can easily extend our work to the de-coding of M-JPEG frames by constructing a spatio-temporal graph Laplacian, without the need to estimate the motion if the motion is small. This spatio-temporal graph is essentially the graph equivalent of the spatio-temporal denoising filter of Buades et al. [33].

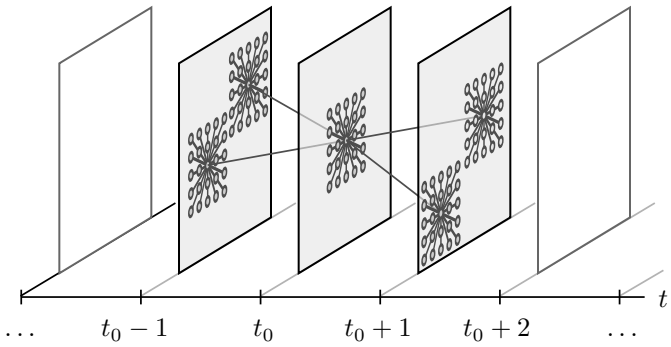

Fig. 12. Spatio-temporal graph kernel for the decoding of M-JPEG-encoded video, using a temporal window of 3 frames. Motion need not be estimated nor compensated for explicitly, since the edge weighting function (18) enforces diffusion in the temporal direction only between blocks that are similar.

Consider a graph G whose vertices $v_i \in V(G)$ are given by

$$v_i = \begin{bmatrix} \dfrac{x_i}{\sigma_X} & \dfrac{y_i}{\sigma_Y} & \dfrac{t_i}{\sigma_T} & \dfrac{z_i}{\sigma_Z} \end{bmatrix}, \tag{46}$$

in which the new component $t_i \in \{1, 2, \ldots, T\}$ is the frame number associated with the ith transform block. The set E of graph edges and the weights w are defined respectively using

[9]The authors of ANCE originally appeal to the central limit theorem to justify the use of $\mathbf{\Gamma}_{m}^{n\dagger}$ and fuse the weights $\mathbf{\Gamma}_{m}^{n\dagger}$ and $\mathbf{\Lambda}^{\dagger}$.

TABLE V
PSNR AND (SSIM) ACHIEVED BY DIFFERENT METHODS ON DECODED M-JPEG VIDEO SEQUENCES COMPRESSED AT QF = 10

		M-JPEG	ANCE[11] [6]	LERaG[11] [10]	Ours ($T = 1$)	VBM4D [31]	Ours ($T = 3$)	Ours ($T = 5$)
Stockholm	(011–030)	27.74 (0.7634)	28.61 (0.7924)	28.59 (0.7886)	28.64 (0.7922)	28.35 (0.7786)	28.97 (0.7990)	**29.04** (**0.8005**)
Mobile	(446–465)	28.58 (0.8031)	29.34 (0.8226)	29.51 (0.8278)	29.42 (0.8240)	29.17 (0.8146)	29.64 (0.8272)	**29.67** (**0.8281**)
Ducks	(101–120)	25.88 (0.7552)	26.73 (**0.7727**)	26.55 (0.7616)	26.61 (0.7630)	26.40 (0.7700)	26.83 (0.7687)	**26.84** (0.7691)
Parkrun	(011–030)	22.92 (0.6907)	23.41 (0.6962)	23.34 (0.6891)	23.33 (0.6952)	23.72 (**0.7131**)	23.74 (0.7068)	**23.83** (0.7094)
Shields	(011–030)	26.13 (0.7495)	26.95 (0.7802)	26.95 (0.7823)	27.00 (0.7858)	26.80 (0.7708)	27.39 (0.7949)	**27.46** (**0.7969**)
Average		26.25 (0.7524)	27.01 (0.7728)	26.99 (0.7699)	27.00 (0.7720)	26.89 (0.7694)	27.31 (0.7793)	**27.37** (**0.7808**)

TABLE VI
PSNR AND (SSIM) ACHIEVED BY DIFFERENT METHODS ON DECODED M-JPEG VIDEO SEQUENCES COMPRESSED AT QF = 20

		M-JPEG	ANCE[11] [6]	LERaG[11] [10]	Ours ($T = 1$)	VBM4D [31]	Ours ($T = 3$)	Ours ($T = 5$)
Stockholm	(011–030)	30.01 (0.8448)	30.81 (0.8634)	30.81 (0.8626)	30.86 (0.8636)	31.05 (0.8657)	31.33 (0.8709)	**31.47** (**0.8728**)
Mobile	(446–465)	30.74 (0.8703)	31.48 (0.8846)	31.64 (**0.8916**)	31.52 (0.8841)	31.66 (0.8856)	31.84 (0.8881)	**31.88** (0.8890)
Ducks	(101–120)	28.19 (0.8506)	28.92 (0.8629)	28.74 (0.8530)	28.87 (0.8589)	28.86 (**0.8642**)	29.12 (0.8636)	**29.13** (0.8636)
Parkrun	(011–030)	24.79 (0.7923)	25.26 (0.7970)	25.26 (0.7920)	25.22 (0.7977)	25.85 (**0.8186**)	25.95 (0.8128)	**26.08** (0.8160)
Shields	(011–030)	28.26 (0.8339)	29.05 (0.8571)	29.09 (0.8586)	29.13 (0.8616)	29.34 (0.8611)	29.67 (0.8710)	**29.80** (**0.8734**)
Average		28.40 (0.8384)	29.10 (0.8530)	29.11 (0.8516)	29.12 (0.8532)	29.35 (0.8590)	29.58 (0.8613)	**29.67** (**0.8630**)

(17) and (18) as before. Our video de-coding problem can be expressed as the optimization problem

$$\min \quad f(\mathbf{u}) = \delta_C^{\alpha}(\mathbf{T}\mathbf{u}) + \mathbf{u}^* \mathbb{L} \mathbf{u}, \tag{47}$$

in which $\mathbf{u} \in \mathbb{R}^{64MT}$ is the raster ordered vector of pixel intensities of MT blocks. Accordingly, we define

$$C = \mathrm{cl}(C_1 \oplus C_2 \oplus \cdots \oplus C_{NT}), \tag{48}$$

and $\mathbb{L}$ is the Laplacian of the spatio-temporal graph G.

If object or camera motion across the frames is small, then $\mathbb{L}$ can further reduce the block artifacts present in the frame at time t using its past and future frames; a pixel near some block boundary at time t moves away from a block boundary at $t - \Delta t$ or $t + \Delta t$, unless the motion is constant at 8 pixels per frame. While motion is not directly compensated for, weighting function (18) ensures that temporal smoothing is affected only in the direction of similar texture across the frames. Solving the problem jointly over all frames in a video sequence is prohibitive, so we take a sliding window approach to solve the problem over a window of 3 frames and generate the final result for the center frame of the window. Fig. 12 illustrates the resulting graph for this windowed approach.

In Tables V and VI, we show the average reconstructed PSNR and SSIM for five test video sequences[10] from [34] using different temporal window sizes. These sequences are chosen such that other methods perform similarly in the case of single frames. The temporal graph regularization produces an average PSNR improvement of 0.35 dB at quality factor $q = 10$ and 0.55 dB at $q = 20$. A comparison with VBM4D [31], which similarly uses inter-frame correlation, is also provided. Note that ANCE and LERaG denoise frame-by-frame, and they may not be optimal for de-noising video. Their results are included in the tables only as reference points.

[10] Subsequently down-sampled by a factor of two using ffmpeg (flag:spline).

XI. CONCLUSION

In this paper, we presented a novel yet minimalistic method for optimized decoding of JPEG images based on a coefficient graph Laplacian regularizer. Our method regularizes variations in the DCT coefficients, rather than in the pixel intensity values themselves. We also related our graph to the non-local means de-noising algorithm of Buades, showing that fast Gaussian filtering can be utilized within the proximal gradient-descent method to solve our problem efficiently. Our method achieves an average of 1.4 dB gain compared to the standard decoding by a JPEG decoder, and is on par with other well-performing JPEG decoding methods while being significantly faster.

Further, we applied our Laplacian regularizer to the problem of decoding M-JPEG video by extending our graph Laplacian in the temporal direction. The spatio-temporal graph Laplacian produces a further gain of 0.45 dB on average compared to the spatial regularizer. Another possible extension of our work could be to introduce a dictionary-based regularization term as in [10], which we defer to future work.

REFERENCES

[1] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[2] A. Zakhor, "Iterative procedures for reduction of blocking effects in transform image coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 1, pp. 91–95, 1992.

[3] W. Hu, G. Cheung, and M. Kazui, "Graph-based dequantization of block-compressed piecewise smooth images," *IEEE Signal Processing Letters*, vol. 23, no. 2, pp. 242–246, 2016.

[4] K. Bredies and M. Holler, "A TGV-based framework for variational image decompression, zooming, and reconstruction. part i: Analytics," *SIAM Journal on Imaging Sciences*, vol. 8, no. 4, pp. 2814–2850, 2015.

[5] A. Foi, V. Katkovnik, and K. Egiazarian, "Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images," *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1395–1411, 2007.

[11] ANCE and LERaG denoise frame-by-frame, and their results are included here only as reference points.

[6] X. Zhang, R. Xiong, X. Fan, S. Ma, and W. Gao, "Compression artifact reduction by overlapped-block transform coefficient estimation with block similarity," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4613–4626, 2013.
[7] H. Chang, M. K. Ng, and T. Zeng, "Reducing artifacts in JPEG decompression via a learned dictionary," *IEEE Transactions on Signal Processing*, vol. 62, no. 3, pp. 718–728, 2014.
[8] C. Zhao, J. Zhang, S. Ma, X. Fan, Y. Zhang, and W. Gao, "Reducing image compression artifacts by structural sparse representation and quantization constraint prior," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 10, pp. 2057–2071, 2017.
[9] C. Dong, Y. Deng, C. C. Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 576–584.
[10] X. Liu, G. Cheung, X. Wu, and D. Zhao, "Random walk graph Laplacian-based smoothness prior for soft decoding of JPEG images," *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 509–524, 2017.
[11] J. Pang and G. Cheung, "Graph Laplacian regularization for image denoising: Analysis in the continuous domain," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1770–1785, 2017.
[12] A. Buades, B. Coll, and J. Morel, "A non-local algorithm for image denoising," in *2005 IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 60–65.
[13] L. I. Rudin and S. Osher, "Total variation based image restoration with free local constraints," in *Proceedings of 1st International Conference on Image Processing*, vol. 1, 1994, pp. 31–35.
[14] W. B. Pennebaker and J. L. Mitchell, *JPEG: Still Image Data Compression Standard*. Springer, 1992.
[15] A. Adams, N. Gelfand, J. Dolson, and M. Levoy, "Gaussian KD-trees for fast high-dimensional filtering," *ACM Trans. Graph.*, vol. 28, no. 3, 21:1–21:12, Jul. 2009.
[16] V. Britanak, P. C. Yip, and K. Rao, *Discrete Cosine and Sine Transforms*. Academic Press, 2007.
[17] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," *Soviet Mathematics Doklady*, vol. 27, no. 2, pp. 372–376, 1983.
[18] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE Transactions on Image Processing*, vol. 18, no. 11, pp. 2419–2434, 2009.
[19] D. P. Bertsekas, *Convex Optimization Algorithms*. Athena Scientific, 2015.
[20] F. R. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997, vol. 92.
[21] M. Vetterli, J. Kovačević, and V. K. Goyal, *Foundations of Signal Processing*. Cambridge University Press, 2014.
[22] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in FORTRAN - The Art of Scientific Computing, 2 ed.* Cambridge University Press, 1992.
[23] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.
[24] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.
[25] Kodak. (2013). Kodak lossless true color image suite, [Online]. Available: http://r0k.us/graphics/kodak/.
[26] USC. (2017). The USC-SIPI image database, [Online]. Available: http://sipi.usc.edu/database/.
[27] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.
[28] I. Daubechies, R. DeVore, M. Fornasier, and C. Güntürk, "Iteratively reweighted least squares minimization for sparse recovery," *Communications on Pure and Applied Mathematics*, vol. 63, no. 1, pp. 1–38, 2010.
[29] T. Chan and J. Shen, *Image Processing and Analysis*. Society for Industrial and Applied Mathematics, 2005.
[30] L. J. Grady and J. R. Polimeni, *Discrete Calculus: Applied Analysis on Graphs for Computational Science*. Springer, 2010.
[31] M. Maggioni, G. Boracchi, A. Foi, and K. Egiazarian, "Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms," *IEEE Transactions on Image Processing*, vol. 21, no. 9, pp. 3952–3966, 2012.
[32] D. T. Võ and T. Q. Nguyen, "Quality enhancement for motion JPEG using temporal redundancies," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 5, pp. 609–619, 2008.
[33] A. Buades, B. Coll, and J.-M. Morel, "Nonlocal image and movie denoising," *International Journal of Computer Vision*, vol. 76, no. 2, pp. 123–139, Feb. 2008.
[34] Xiph.org. (2017). Xiph.org video test media, [Online]. Available: https://media.xiph.org/video/derf/.

Sean I. Young received B.Com and B.E. degrees in software engineering from The University of Auckland in 2008, and the M.EngSc and Ph.D. degrees from The University of New South Wales, in 2011 and 2018, respectively, where he is currently a post-doctoral researcher. In 2016, he was a visiting researcher at InterDigital Communications, San Diego, CA. His research interests are large-scale optimization and inverse problems in imaging.

Aous T. Naman received the B.Sc. degree in Electronics and Telecommunication Engineering from Al-Nahrain University, Baghdad, Iraq, in 1994, the M.Eng.Sc. degree in Engineering from University of Malaya, Kuala Lumpur, Malaysia, in 2000, and the Ph.D. degree in Electrical Engineering from the University of New South Wales (UNSW), Sydney, Australia, in 2011. He is currently pursuing postdoctoral research with the School of Electrical Engineering and Telecommunications, UNSW. His research interests are in image/video compression and delivery. He has served as a reviewer for many journals and conferences, such as the IEEE Transactions on Image Processing, IEEE Signal Processing Letters, and the International Conference on Image Processing.

David Taubman received B.S. and B.E. (Electrical) degrees in 1986 and 1988 from the University of Sydney, and M.S. and Ph.D. degrees in 1992 and 1994 from the University of California at Berkeley. From 1994 to 1998 he worked at Hewlett-Packard's Research Laboratories in Palo Alto, California, joining the University of New South Wales in 1998, where he is a Professor in the School of Electrical Engineering and Telecommunications. Dr. Taubman is author with M. Marcellin of the book, "JPEG2000: Image compression fundamentals, standards and practice". His research interests include highly scalable image and video compression, motion estimation and modeling, inverse problems in imaging, perceptual modeling and multimedia distribution systems. Dr. Taubman was awarded the University Medal from the University of Sydney. He has received two Best Paper awards: from the IEEE Circuits and Systems Society for the 1996 paper, "A Common Framework for Rate and Distortion Based Scaling of Highly Scalable Compressed Video"; and from the IEEE Signal Processing Society for the 2000 paper, "High Performance Scalable Image Compression with EBCOT".