# Compressed Sensing for Image Compression

David Fridovich-Keil and Grace Kuo

## I. Introduction

Over the last decade or so, there has been a lot of progress in the field of compressed sensing. Indeed, the entire field is only about ten years old. The term "compressed sensing" essentially refers to the idea that some signals – specifically those that are sparse in some basis – can be reconstructed from fewer samples than the amount dictated by the Nyquist sampling theorem [1]. This has been demonstrated with some success with application to audio, digital holography, and MRI [2], [3], [4].

This leads us to consider images, where there are many obvious applications. For example, compressed sensing could be used in a new CMOS sensor to improve the effective data rate and reduce power consumption by directly sensing the image in compressed form and transmitting only the compressed version off-chip. Alternatively, more samples could be obtained to compensate for a noisy sensor. For instance, infrared cameras are notoriously noisy, and compressed sensing could be used to alleviate this problem.

Most of the published work on compressed sensing in the image domain has been focused on designing cameras to implement compressive sensing schemes [5], [6] [7] [8] [9]. However, this makes it challenging to determine how well the reconstruction is working independently of the sensing architecture. To determine bounds on reconstruction quality, we examine image compression and reconstruction assuming an ideal sensor. First we develop and evaluate a compression scheme similar to JPEG. Then we use this compression scheme as the basis of compressed sensing, and we compare two different convex relaxations of the reconstruction optimization problem. Finally, we investigate and demonstrate an application in image denoising.

## II. Literature Review

Our work spans two separate fields – *lossy image compression* and *compressed sensing*. Here, we review what we have found to be the most relevant results in each. In the following sections, we synthesize these two distinct viewpoints into a single unified theoretical framework for image compression.

### A. JPEG

Perhaps the most popular lossy image compression standard is JPEG. While not directly related to the field of compressed sensing or convex optimization, the JPEG algorithm relies on a deep understanding of digital signal processing, which is essential to understanding compressed sensing. For this reason, here we provide a brief (and somewhat simplified) review the JPEG standard, as described in [10].

Essentially, JPEG assumes that the information in an image is most cleanly expressed in the Fourier domain, rather than in the standard basis. In other words, the Fourier transform of an image is more easily compressed than the image itself. As we will see, JPEG exploits this by quantizing the image in frequency space.

**Algorithm 1** Simplified JPEG Encoder

$image \leftarrow$ imread(FILE)
$blocks \leftarrow$ getBlocks($image$)
**for** $block$ in $blocks$ **do**
  $coefs \leftarrow$ dct2($block$)
  $quant \leftarrow$ quantize($coefs$)
  $encoded \leftarrow$ entropyEncode($quant$)
**end for**

Algorithm 1 is a sketch of the JPEG encoder. The general procedure (again, described in [10]) is to break the image up into smaller blocks, each 8x8 pixels, and compress the image blockwise. Each block is first converted to a matrix of discrete cosine transform (DCT) coefficients, and then quantized by dividing elementwise by a matrix $Q$, which is usually pre-specified for a given application. For example, the following quantization matrix is common [11]:

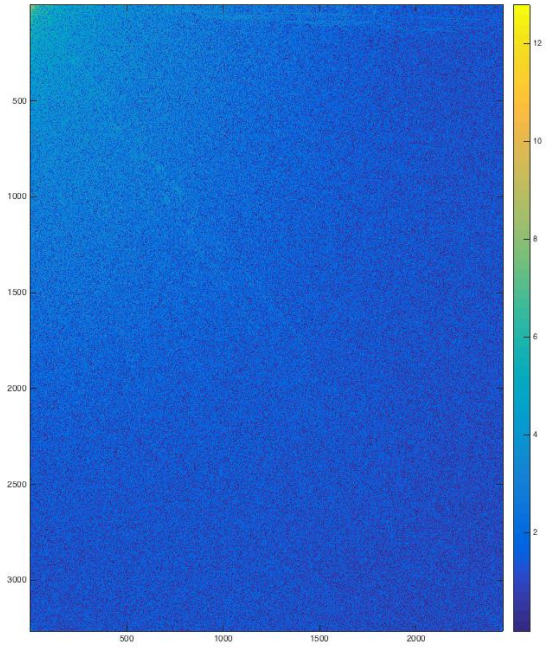
$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Since the elements of the 2D DCT corresponding to the lowest frequencies are in the top-left corner and those corresponding to the highest are in the bottom right, dividing elementwise by this matrix and quantizing (rounding) tends to zero out the high frequency content of each block. In effect, JPEG is doing a blockwise low-pass filter. The only remaining step is to do lossless entropy-based compression of these quantized DCT coefficients.

Now that we have explained *how* JPEG encoding works, we provide some intuition as to *why* it works.

First, we must explain why it makes sense to operate in the DCT/Fourier domain. What is it about frequency space that makes images more compressible there than in the standard basis? The short answer is that no one really knows for sure. There are no popular references we know of that offer any sort of formal explanation. However, [12] provides some

(a) Original image.




(b) 2D DCT, under the elementwise map $x \to \log(\text{abs}(x) + 1)$.

Fig. 1: Sparsity in the DCT domain. The vast majority of DCT coefficients are very nearly zero. In particular, the only non-small coefficients are highly localized to the top-left corner, corresponding to low frequencies.

reasonable intuition: each element in the Fourier basis has energy spanning each and every pixel in the image. This means that it is possible for a very small number of Fourier basis vectors to create a reasonable approximation to an entire image, provided that the image is *sparse* in the Fourier domain. That is, its Fourier transform must have most of its energy in a small number of components.

An image might be sparse in the Fourier domain if, for example, it does not have very much high frequency content. For example, consider Figure 1, which is a typical sight on the Berkeley campus. As shown in the log-scale transform image, the vast majority of DCT coefficients are approximately zero, and in particular, almost all of the energy in the DCT domain is localized in the low frequencies (in the top-left). Thus, it is reasonable to expect that we can achieve a high compression ratio by operating in the DCT domain.

Now that we have justified using frequency-domain representations like Fourier and DCT to do compression via low-pass filtering, we must only examine JPEG's usage of *blockwise* compression.

Clearly, there are computational benefits to blockwise (versus entire-image) compression: since blocks are non-overlapping, implementation can be parallelized very easily. Unfortunately, however, there are some downsides. First – compressing each block independently sets an arbitrary upper bound on the compression ratio since we require at the very least one DCT coefficient per block (otherwise it will appear completely black in the reconstruction). Using JPEG's 8x8 block standard, this gives a maximum compression ratio of $63/64 = 98.44\%$ (before lossless entropy encoding). Second – when operating at such high compression ratios, it is common to observe block-level artifacts, making the reconstruction appear highly pixelated.

### B. Compressed Sensing

Compressed sensing generalizes the ideas and intuition developed by compression schemes like JPEG.

JPEG relies on low-pass filtering in the DCT domain to compress an image block-by-block. In effect, then, JPEG assumes that the image is (blockwise) band-limited, i.e. that most of the energy in the DCT spectrum is confined to low frequencies. This is a very strong statement about a more general concept – *sparsity*.

For example, what would happen if an image had only a small number of non-zero DCT coefficients (such as the example image from Figure 1), but the coefficients corresponded to a uniform distribution over frequencies (instead of only low frequencies)? When JPEG low-pass filters each block, then, it will eliminate (or significantly quantize) important coefficients that represent high-frequency content, resulting in a drastic loss of information and a poor quality reconstruction.

*Sparsity* is the underlying concept that allows for compression in this case. We define a signal $X$ to be $k$-sparse

if no more than $k$ of its elements are non-zero. If a signal is $k$-sparse, it turns out that the signal can be compressed in a similar fashion to JPEG by encoding only the $k$ nonzero coefficients and their locations. Many real-world signals are sparse in some domain, such as images in Fourier space, where most of the coefficients are sufficiently small that they can be set to zero without adversely affecting the image quality.

Compressed sensing leverages a signal's inherent sparsity to allow for simultaneous data acquisition and compression. The standard process is to do data aquistion and compression sequentially. First the full data set is sampled at the Nyquist rate, and then it transformed into a sparse domain, such as DCT. Then, compression is performed by removing all but the most important coefficients, as is done in JPEG. However, this can feel wasteful, since a large amount of data was collected, but only a small amount was kept. We may ask, can we collect less data but still reconstruct the full signal? In other words, can we perform compression *during* the sensing process instead of doing it afterwards.

Compressed sensing and sparsity offer the theory that allows us to do precisely that. If $X$ is a signal in $\mathbb{R}^n$ and is sparse in some domain $\Psi$, then $X$ can be reconstructed from $m$ measurements where $m \ll n$ when $X$ is sampled in a domain $\Phi$ that is incoherent with $\Psi$ [13]. The most common sampling basis $\Phi$ is a random matrix because it has a high probability of being incoherent with $\Psi$ [1].

This framework defines an underdetermined linear system, so how does one solve it and get reliable reconstructions? The answer is sparsity - the reconstruction relies both the measurements and the prior that the signal is sparse. As a result, the number of measurements needed, $m$, depends more heavily on $k$, the signal sparsity, than on $n$, the signal length. According to [8], for full signal reconstruction we must have

$$ck \log(n/k) \leq m \ll n$$

where $c$ is a small constant. In practical cases, we can use the rule of thumb that $m$ must be at least three times larger that $k$.

Compressed sensing has been implemented with some success in many fields, such as audio [2], holography [3], and MRI [4]. MRI is one of most successful examples of compressed sensing because MRIs are naturally sparse in the Fourier domain, and compressed sensing dramatically accelerates the typically time-consuming data acquisition process.

There have also been many attempts to apply compressed sensing to images, and in fact, this was the first area where compressed sensing was attempted [5]. In imaging, compressed sensing offers several advantages over conventional techniques [9]:

- Through compressed sensing, we can take fewer measurements and thus reduce the power consumed by the sensor and speed up the data transfer between the sensor and memory. Concerns of power consumption and data transfer are increasingly common as the size of image files grows to unwieldy proportions.
- Compressed sensing can be done with a single detector, rather than an array, allowing for more sensitive (and expensive) detectors.
- Since the measurements are random linear combinations of the scene, the raw output is encrypted and cannot be reconstructed without the measurement model, which could be good for security.
- Since each pixel is measured multiple times, compressed sensing is more robust than conventional imaging. For instance, in compressed sensing, if a fraction of the measurements are missing, the image can still be reconstructed. However, in conventional imaging, loosing some measurements means there will holes in the image.

Most of the research on compressed sensing with images is focused on designing and building a setup to take appropriate measurements. For example, Duarte et. al. designed a single pixel camera for compressed sensing [7]. In this setup, light from a scene hits a digital micro-mirror device (DMD), which reflects parts of the scene to the detector. The DMD can be controlled electronically to get a different random projection at each measurement. Using the theory of compressed sensing, the researchers can reconstruct a scene with more pixels than the number of measurements. This setup was also used by Takhar et. al. in [9].

A different setup, called "random aperture encoding" was proposed by Stern et. al. in [8]. Here, a diffuser is placed in front of a pixel array sensor. The diffuser essentially scrambles the light such that each pixel on the sensor sees a different random combination of light. Since this design requires an array of sensors, one cannot upgrade to a more sensitive detector, as was done with the single pixel camera in [7], [9]. Also, in this scheme, the number of measurements is fixed. However, with this design all of the measurements can be taken simultaneously, which will improve its performance for moving objects.

Finally, Oike et. al. demonstrated a third setup in which the random linear projection step is done during analog to digital conversion [5]. Different pixel values are sequentially placed at the input of a $\Sigma\Delta$ ADC which results in averaging and quantization of the pixel values at the output. This method has higher SNR than several previous strategies.

Since most of the previous work has focused on designing measurement setups, there are few results on how compressed sensing with images works in theory. Instead, the reconstruction results from the above sources are influenced by the SNR of the physical setup, making it difficult to determine if artifacts in the reconstruction are due to poor reconstruction or bad measurements. In this paper, we take a more theoretical look into compressed sensing on images by examining the quality of reconstructions we could achieve using an ideal measurement system. This work will provide a theoretical baseline that systems, such as those described above, can be compared to.

The structure of the paper is as follows: In Section III,

we describe the problem statement and our formulation of the optimization problem. In Section IV, we perform lossy image compression in different domains. This work serves as a foundation for the rest of the paper. In Sections V, VI, and VII, we do compressed sensing and reconstruct the images using different convex relaxations, and we compare the results. In Section VIII, we demonstrate an application in image denoising, and in Section IX we describe future work in this field.

## III. Problem Formulation

We formulate compressed sensing problem as follows. Suppose that there exists some basis $\Psi$ (represented as a matrix) such that the image $y$ (as a column vector) is sparse in $\Psi$, i.e. $\exists x$ s.t. $x$ is $k$-sparse (x has at most $k$ non-zero elements) and the following equation holds:

$$y = \Psi x, ||x||_0 \leq k$$

In general, however, what we "measure" is not the true image $y$, but actually some transformed version which we denote $m$. We assume a linear measurement model, which we represent with the matrix $A$. We write:

$$m = Ay = A\Psi x$$

In general, this model places no restrictions on the matrix $A$. However, in the case of compressed sensing, we are interested in *random* matrices $A$, which we refer to as "mixing matrices." By using a random matrix, we represent $y$ with respect to a basis whose vectors are random linear combinations of the original basis, $\Psi$. This is important because, according to compressed sensing theory, the measurment matrix $A$ must be incoherent with the basis, $\Psi$, and this is true with high probability for a random matrix [1]. Moreover, we allow $A$ to be of less than full rank. This lets us model the *sensing* process in which we undersample (or oversample) the signal.

Now, we wish to find $x$ given $m$ and some guess of a sparse basis, $\hat{\Psi}$. In particular, we wish to minimize the $L_2$ distance between the reconstructed measurements and the actual measurements, while penalizing the number of non-zero entries in $x$ by some non-negative parameter $\alpha$. We cast this as the following optimization problem:

$$x^* = \arg\min_x \|A\hat{\Psi}x - m\|_2^2 + \alpha\|x\|_0 \tag{1}$$

Of course, this problem is not generally tractable for arbitrary bases $\hat{\Psi}$. In the following sections, we consider variations on this problem in which there exist tractable (or even explicit) solutions or approximations.

## IV. Lossy Image Compression

### A. Theory

Lossy image compression can be represented as a simplification of equation (1) in which A is set to the identity, and the $L_0$ penalization is replaced by a constraint. In this case, equation (1) can be rewritten as follows:

$$x^*(k) = \arg\min_x \|\hat{\Psi}x - y\|_2^2 : \|x\|_0 \leq k \tag{2}$$

From this formulation, it is clear that $x^*$ is a $k$-sparse representation of the original image $y$ in some basis $\hat{\Psi}$. $x^*$ is a "good" representation if $\hat{\Psi}x^*$ is similar to the original image. In order to measure how "good" a representation is, we define the error function with respect to the $L_2$ norm:

$$\text{error} \doteq \|\hat{\Psi}x - y\|_2 \tag{3}$$

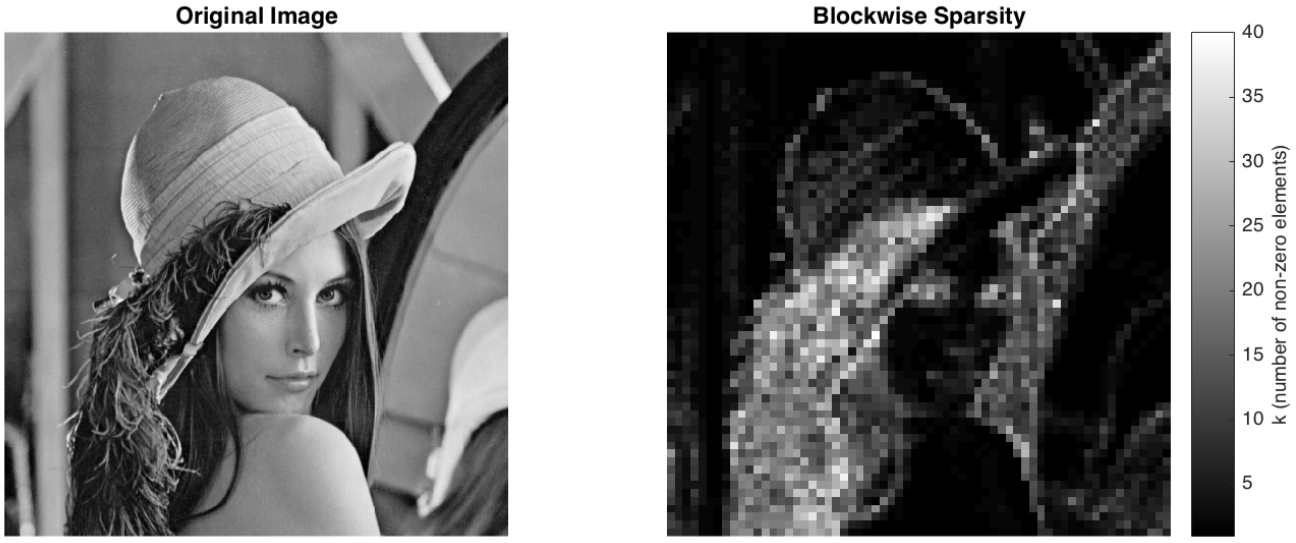



Fig. 2: Blockwise sparsity. Each 8x8 block in the original image is assigned a constant value equal to the number of nonzero DCT coefficients. Observe that regions with strong texture, such as the woman's hair, have many more coefficients than those without much texture. Also, note that the maximum number of coefficients across any particular block is 40, even though the blocks have $8 \times 8 = 64$ total coefficients.

In general, problem (2) is not tractable for arbitrary $\hat{\Psi}$. However, if $\hat{\Psi}$ is an orthonormal basis, such as Fourier or DCT, there is a closed form solution. Without loss of generality, we derive the closed form solution for the Fourier basis, $\hat{\Psi} = F$.

The intuition is simple: at optimum, we must have exactly $k$ non-zero elements in $x$ (otherwise the constraint is not active). Since the basis $F$ is orthogonal, we can simply project $y$ onto $F$ (i.e. take the Fourier transform, $F^T y$) and zero out all but the largest $k$ coefficients in the result. The result is precisely $x^*(k)$.

More formally, we can prove the result by recognizing that the problem is identical to:

$$x^* = \arg\min_x \|x - F^T y\|_2^2 : \|x\|_0 \leq k$$

where $F^T m$ is the Fourier transform of $m$. We are able to rewrite the objective function in the Fourier domain because Parseval's identity ensures the invariance of the squared $L_2$ norm under the Fourier transform. More generally, this holds for any unitary transformation $U : U^*U = I$ because $\|Uv\|_2^2 = v^*U^*Uv = \|v\|_2^2$.

In this new form, it is clear that the optimal $x$ must share the same largest $k$ entries as $F^T m$, but have the rest of its elements equal to zero.

### B. Blockwise Compression

Consider the optimization problems (1) and (2). If we begin with an image of size $n$ x $n$, the column vector $x$ will have $n^2$ elements. We multiply $x$ by a square matrix $\hat{\Psi}$, so $\hat{\Psi}$ will have $n^4$ elements. Therefore, we see that computation time and memory usage will increase rapidly for larger images. In order to apply compression and compressed sensing to larger images in a reasonable amount of time, we split the original image into "blocks" and treat each block separately. However, in an image, we expect that the information density (and therefore sparsity) will vary from block to block. Indeed, in a simple test image there is high variation in DCT-domain sparsity across the image – see Figure 2. Therefore, rather than fixing $k$ as in equation (2), we set $k$ adaptively as a function of the information density in the given block.

**Algorithm 2** Adaptive Blockwise Lossy Image Compression

```
image ← imread(FILE)
blocks ← getBlocks(image)
for block in blocks do
    coefs ← dct2(block)
    normalized_coefs ← coefs/ max(coefs)
    coefs[normalized_coefs < γ] = 0
    k ← nnz(coefs)
end for
```

We implement the algorithm as shown in Algorithm 2. First, a block is transformed into the orthonomal basis (for instance Fourier). Then, the coefficients are normalized by the largest magnitude coefficient. Finally, normalized coefficients that are less than some $\gamma$ are set to 0. The number of coefficients greater than $\gamma$ is equal to $k$. In this formulation, $\gamma$ is constant over all blocks, but $k$ changes from block to block. We use $k$ as a measure of sparsity. Figure 3 shows a visualization of the sparsity of two blocks, one with high information density and one with low information density.

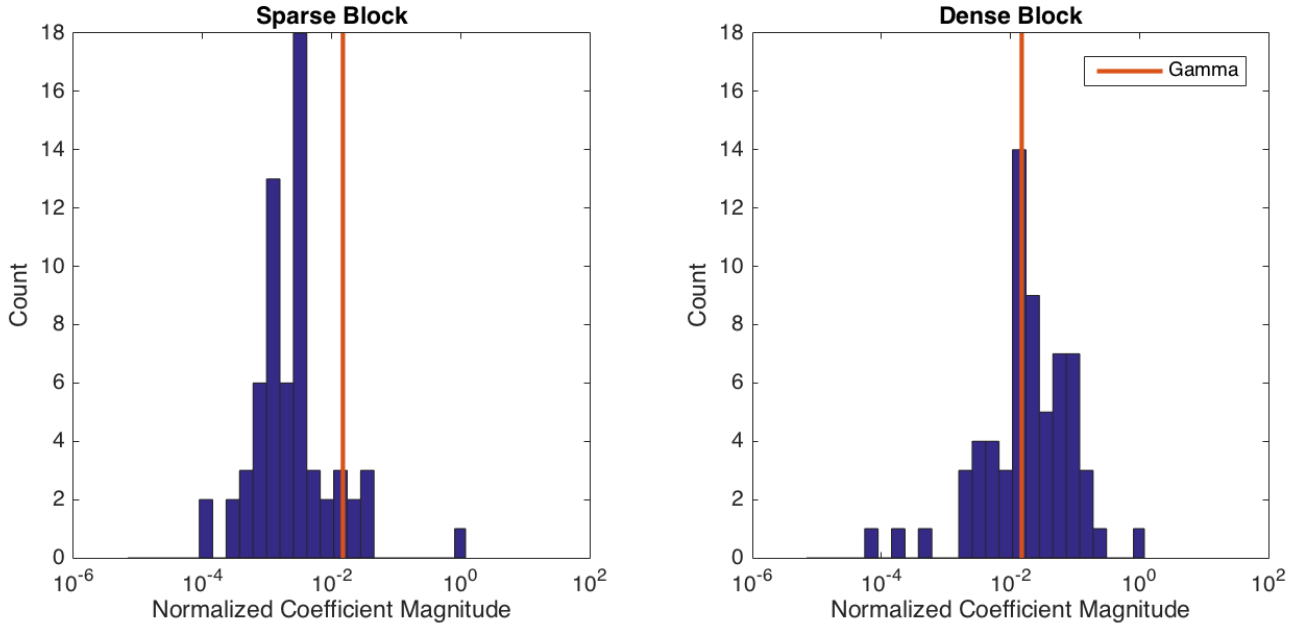



Fig. 3: Sparsity of two blocks. The block on the left is relatively sparse in the Fourier domain, and the one on the right is dense. As shown, the threshold $\gamma = 0.015$ is the measure of proximity to zero – i.e. the sparse block has most coefficients smaller than $\gamma$, and the dense block does not.

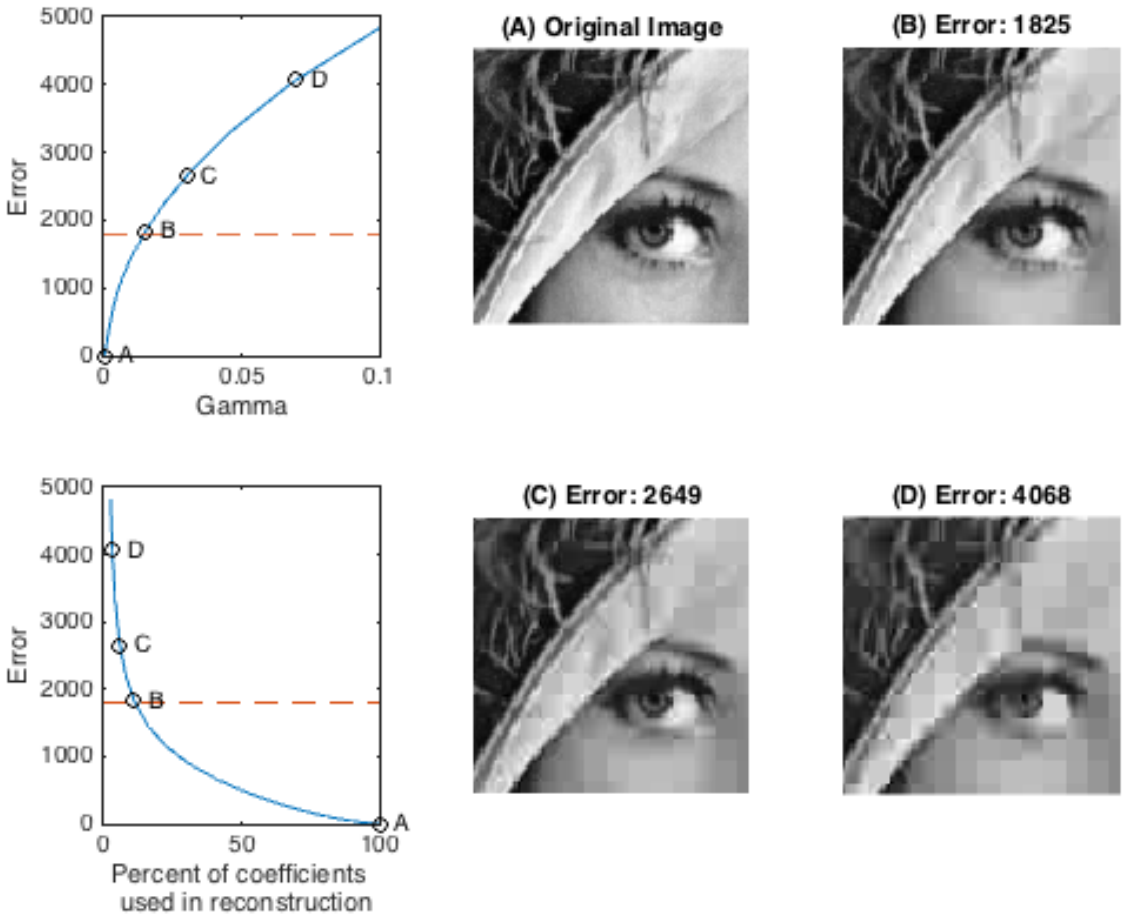



Fig. 4: Adaptive compression threshold versus reconstruction error. The top curve shows the relationship between increasing $\gamma$ (the fraction of the largest magnitude DCT coefficient below which all coefficients are replaced with zero), and the reconstruction error defined in equation (3). The bottom curve shows the corresponding average blockwise percentage of coefficients used in reconstruction. Predictably, as $\gamma$ increases, error increases because fewer coefficients are used in reconstruction. We show a zoomed-in section of the reconstruction at four different points along these curves. Empirically, point (B) is the highest error we can tolerate before the aberrations are obvious to the casual observer.

### C. Choosing the adaptive threshold

How do we choose $\gamma$, the fraction of the maximum coefficient below which coefficients are considered "small" and set to zero? As shown in Figure 4, there is a tradeoff between including fewer coefficients (farther right on the top plot and left on the bottom plot) and increasing error – that is, increasing compression ratio also increases reconstruction error. We observe that the highest tolerable error occurs near point (B) on these curves, resulting in the corresponding reconstruction in the top right. Thus, we determine an optimal (or at least, sufficiently near-optimal) value of $\gamma = 0.015$.

### D. Comparison of Domains

For the rest of our work in compressed sensing, we must choose a domain to work in. However, if we want to solve (1) with traditional tools such as CVX, we are forced to use only real numbers. Unfortunately, the Fourier basis is complex, and even for real-valued signals (like images). In order to circumvent this rather artificial dilemma, we replace the Fourier transform with the discrete cosine transform, or DCT (as is done in JPEG). The DCT basis is still orthogonal, so we can solve (2) in exactly the same way if we replace the Fourier basis with the DCT basis. Specifically, the same algorithm (Algorithm 2) is optimal at the block level for the problem formulated in equation (2): take the DCT of $m$, then zero out all but the largest $k$ components, where $k$ is chosen adaptively on each block.

Figure 5 shows this general algorithm applied to the same image in three different bases: the image domain, the Fourier domain, and the DCT domain. Here, we have chosen values of $\gamma$ such that the reconstruction error in each case is roughly equal to the value determined in Figure 4, at point (B). Observe that the Fourier and DCT representations are much more sparse than the standard basis. This agrees with our ear-

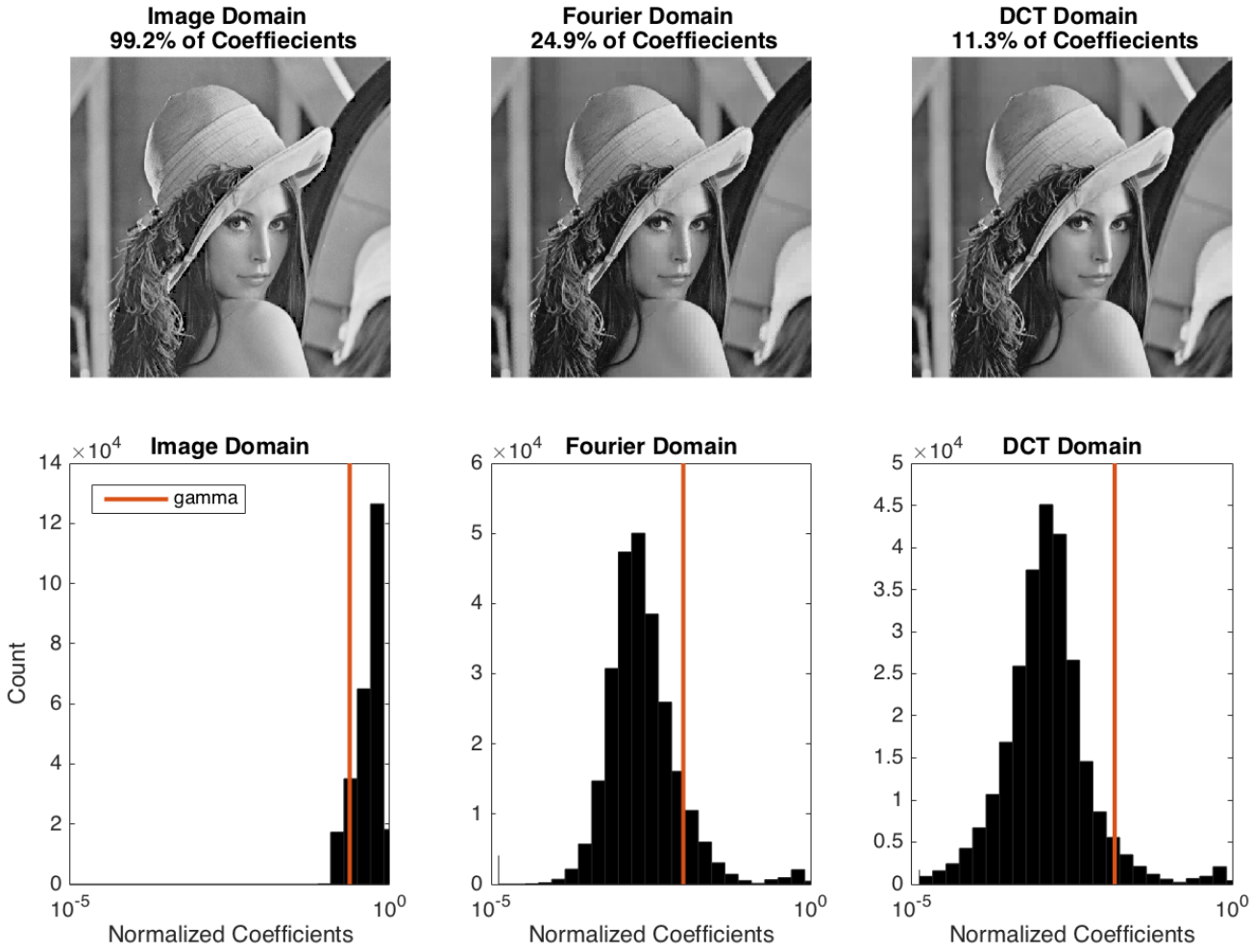



Fig. 5: Adaptive blockwise lossy image compression, in the standard basis, in the Fourier domain, and in the DCT domain. Here, we have set $\gamma$ in each image to achieve nearly the same reconstruction error – again, measured using equation (3). Observe that the image is drastically sparser in the Fourier and DCT domains than it is in the standard basis, and it is still sparser when expressed in the DCT basis than it is in the Fourier basis. In some sense, this justifies JPEG's (and our) usage of the DCT.

lier analysis regarding the JPEG encoding paradigm. Further, we note that the DCT-based compression outperforms the Fourier compression scheme. This may well be ascribed to chance – i.e. we cannot immediately conclude that *all* images are more easily compressed in the DCT basis. However, it is more likely that at some point the test image was converted to and back from the JPEG format, which means that at some point it was compressed using the DCT, which of course implies that it will be most easily re-compressed using the DCT.

## V. $L_1$ RELAXATION

### A. Compressed sensing in the DCT domain

At this point, we return to equation (1), replacing the $L_0$ penalization with $L_1$. As above, we set $\hat{\Psi} = C$, where $C$ is the DCT basis. The optimization problem becomes:

$$x^* = \arg\min_x \|ACx - m\|_2^2 + \alpha\|x\|_1 \tag{4}$$

The problem above is convex and real-valued, and moreover it is essentially in standard LASSO form, which means that it can be solved using an off-the-shelf LASSO solver. In practice, however, we use CVX for convenience and ease of extension to other relaxations. For a detailed explanation of CVX and it's style of Disciplined Convex Programming (DCP), see [17].

In addition to changing the $L_0$ norm to an $L_1$ norm, we have also re-introduced what we term the "mixing matrix," $A \in \mathbb{R}^{pxn}$. Recall that this mixing matrix effectively acquires $p$ randomized linear measurements of the image, such that $m = Ay \in \mathbb{R}^p$. We know from our review of the literature that, in general, it should be possible to set $p \ll n$ so long as the actual sparsity of the image is still sufficiently smaller

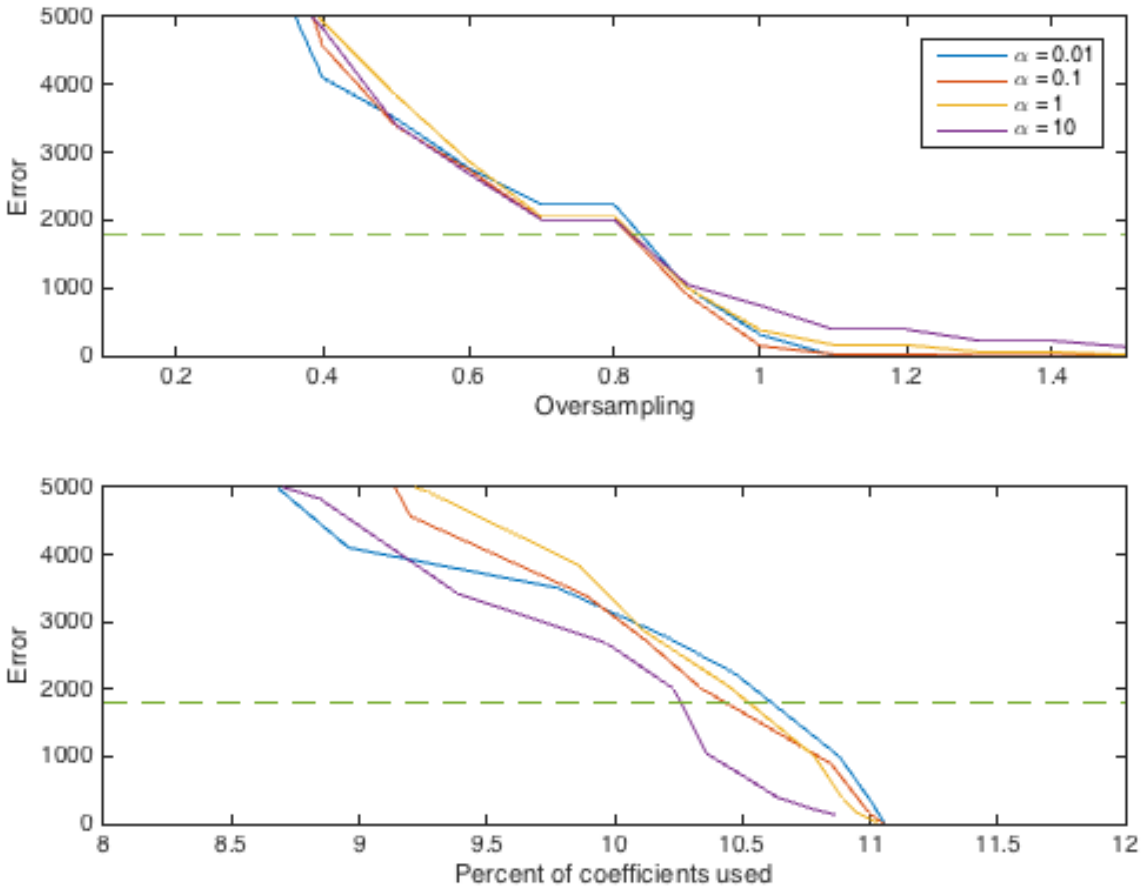



Fig. 6: Oversampling rate and sparsity versus reconstruction error, evaluated for $\alpha$ spanning four orders of magnitude. We observe that the curves are more or less invariant to $\alpha$, and that all reconstructions involve at most $11\%$ (on average) of the blockwise DCT coefficients. Note that the image cannot be compressed much beyond that threshold before the error becomes intolerable (above the dotted green line); however, the oversampling ratio can decrease a full $20\%$ and still lead to a high quality reconstruction.

than $p$. We term the ratio $p/n$ the "oversampling rate," or OSR.

Figure 6 shows the reconstruction results for a range of OSR, for four different values of $\alpha$ (spanning four orders of magnitude). In each case, $A$ is composed of independent, identically distributed (IID) standard Gaussian random variables, as in typical compressed sensing schemes (e.g. [16]). Observe that the dependence on $\alpha$ is only very slight – especially considering the exceptionally large range of $\alpha$ values plotted. We see that the result of (4) is never more than $11\%$ sparse, which is to say that $11\%$ is the average number of non-zero coefficients over all 8x8 blocks. The fact that we cannot decrease this number very much before the reconstruction error exceeds the threshold found in Figure 4 suggests that compressed sensing cannot *overcompress* an image – i.e. compressed sensing cannot make an image more sparse than it actually is without causing reconstruction aberrations. However, we can achieve nearly a $20\%$ undersampling rate (i.e. $OSR = 0.8$) while still achieving low error.

Hence, we see that even though compressed sensing cannot *overcompress* an image, it can *undersample* the image and still achieve an acceptable reconstruction error. In practice, this is very encouraging for physical implementations of compressed sensing in image acquisition systems.

### *B. What does undersampling look like?*

So far, we have only given a mathematical definition of oversampling, i.e. that $OSR = p/n$, the ratio of the number of measurements to the number of image pixels. Intuitively, we can see that setting $OSR > 1$ corresponds to taking more measurement samples than image pixels, and that, more interestingly, setting $OSR < 1$ corresponds to *undersampling* the image.

From elementary signal processing, we know that if the spectrum of the image is *bandlimited* to some bandwidth $B \ll W$ where $W$ is the total spectrum width, then according

(a) $\alpha = 0.1, OSR = 0.8$ (b) $\alpha = 0.1, OSR = 0.1$. (c) $\alpha = 10, OSR = 0.1$

Fig. 7: Effect of oversampling rate and $L_1$ sparsity penalization. On the left, we have only slight penalization for non-sparse solutions, and we set OSR to the $80\%$, which was roughly the lower bound we found in Figure 6 above which the error is sufficiently small. In the center, we decrease OSR dramatically, and we are not at all surprised that blockwise artifacts begin to appear. On the right, we increase the $L_1$ penalization dramatically, which introduces unexpected aberrations – it appears that the coefficients are all small (leading to a small $L_1$ penalty), but the error is still high because the dramatic undersampling causes interesting masking effects.

to the Shannon-Nyquist sampling theorem, we need only take samples at a rate $2B \ll W$ to achieve perfect reconstruction. In some sense this is also *undersampling*.

In our case, though, undersampling is slightly less intuitive, and we also have no absolute guarantee of being able to achieve perfect reconstruction. Figure 7 demonstrates the results from three $(\alpha, OSR)$ parameter pairs. As shown, for sufficiently high OSR (above the $80\%$ threshold from Figure 6) the error is imperceptible. However, when we undersample the image dramatically, the the reconstructions are necessarily worse. For small $\alpha$, the effect is visually similar to naive downsampling (i.e. setting all pixels in each 8x8 block to a single value). For large $\alpha$, the effect is more akin to masking each block (i.e. there appears to be single dominant coefficient and many smaller coefficients, which makes each block look like a masked version of the original).

### C. Theoretical predictions

From [8], we know that we can expect a reasonably good reconstruction if we take roughly three times as many samples as the actual sparsity of the image. In our case, the average blockwise sparsity is approximately $11\%$, so we set $OSR = 0.3$ and solve problem (4).

As shown in Figure 8, reconstructing using an OSR of 0.3 yields a completely recognizable result. Viewed from a distance, it is barely distinguishable from the original; up close, however, there are serious blockwise aberrations (though not quite as serious as in the center image of Figure 7). Since [8] makes no claim to work well for blockwise reconstructions such as ours, we ought only consider the reconstruction fidelity within each block. By visual inspection, we can readily verify that each block has very nearly the same texture as the corresponding block from the original image (recall Figure 2).

### D. Compressed sensing in the image domain

For comparison, we also consider reformulating equation (4) in the standard basis (a.k.a. the image domain). This is intended merely for pedagogical purposes, to reinforce what we learned in our discussion of JPEG regarding the importance of choosing a basis in which the image is actually sparse.

Consider setting $\hat{\Psi} = I$. That is, we assume that we do not know of any basis in which the image is likely to be sparser than it already is in the standard basis. This gives us:

$$x^* = \arg\min_x \|Ax - m\|_2^2 + \alpha\|x\|_1 \tag{5}$$

Figure 9 shows the results of running compressed sensing as above, for the same values of OSR and $\alpha$, again on the same Lenna image. Note that, in the top plot, reconstruction error is incredibly sensitive to OSR – the error increases much more quickly as OSR decreases below unity than in the DCT case (as shown). Also, we see that the percentage of coefficients used in reconstruction is always larger than $10\%$, yielding terribly inaccurate reconstructions, and the reconstructions only become tolerable when virtually all the coefficients are non-zero.

These results are completely intuitive. As we learned in our study of JPEG, what makes images sparse in bases like the DCT is the fact that DCT basis elements contain energy across all standard basis vectors. Since we know images are sparse in the DCT domain, they are almost certainly *not* sparse in the image domain because those few DCT principle basis vectors contain energy across the entire image. In short, unless the image has a lot of black pixels, it is by definition not sparse in the standard basis. Once again, we see that compressed sensing cannot *overcompress*, and in this case it cannot even *undersample* because the image is not sparse in this basis.

Fig. 8: Reconstruction, setting $\alpha = 0.1, OSR = 0.3$. Since $OSR < 0.8$, we expect from Figure 6 that reconstruction quality will be relatively poor. However, the image is completely recognizable, and in fact the most obvious aberrations are clearly the result of applying compressed sensing blockwise. Looking at each block individually, the similarity to the original image is striking.

## VI. Reversed Huber relaxation

### A. Theory

One other relaxation for the $L_0$ penalization is the so-called "reversed Huber penalty," which is derived in [14] and referenced in Exercise 13.6 of [15]. Here, the authors begin with the following problem, which we adjust slightly to be closer to the form of equation (1):

$$\phi(\rho, \alpha) = \min_x \|Q^T x - m\|_2^2 + \rho^2 \|x\|_2^2 + \alpha \|x\|_0 \quad (6)$$

This problem is not precisely equivalent to equation (1), because it includes an extra $L_2$ penalty for the size of $x$, and because it replaces the sensing matrix product $A\hat{\Psi}$ with an arbitrary $Q$ (this is simply an artifact of notation). The extra regularization term is actually quite common – for example, many LASSO implementations are precisely of this form, of course replacing the $L_0$ penalty with $L_1$. For this reason, we ignore the added regularization, and simply adjust parameters again to find empirically "good" values.

The first step is to solve the problem for $\alpha = 0$, i.e. when we disregard the cardinality penalty. This allows a closed form solution:

$$\phi(\rho, 0) = m^T \left( I + \frac{1}{\rho^2} \sum_{i=1}^{n} q_i q_i^T \right)^{-1} m$$

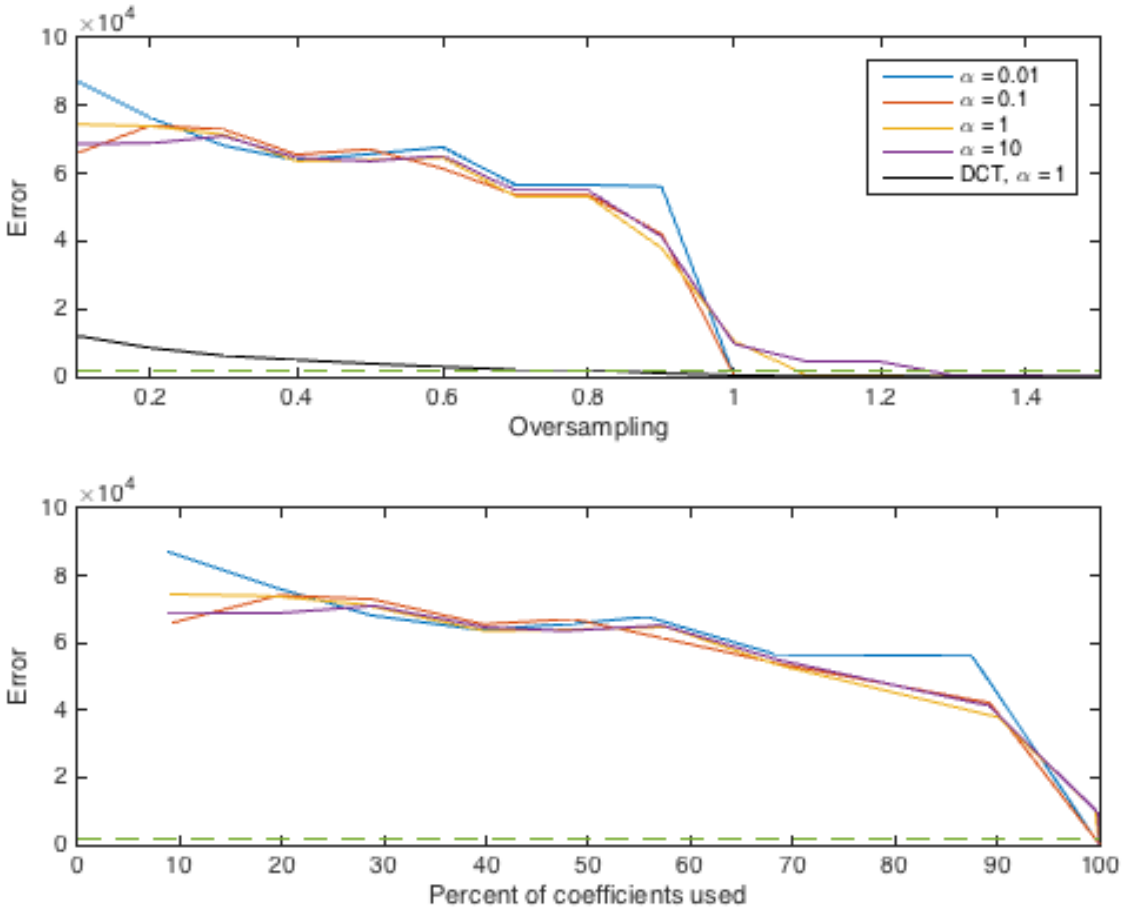



Fig. 9: Oversampling rate and sparsity versus reconstruction error for compressed sensing in the image domain, evaluated for $\alpha$ spanning four orders of magnitude. Note that the reconstruction error is extremely sensitive to OSR. Also, the reconstructions are no longer always sparse, and when they are sparse the error is absolutely intolerable (far above the dotted green line).

where $q_i^T$ is the $i^{th}$ row of the matrix $Q$.

Next, we rewrite the $L_0$ penalty using a Boolean vector, $u$, as follows, and adjust the previous expression by eliminating those outer products from the sum that correspond to zero elements of $x$ at optimum, and adding back the remaining penalization term. This gives us:

$$\phi(\rho,\alpha) = \min_{u\in\{0,1\}^n} m^T \left( I + \frac{1}{\rho^2}\sum_{i=1}^{n} u_i q_i q_i^T \right)^{-1} m + \alpha \sum_{i=1}^{n} u_i$$

So far, this is all a series of *exact* manipulations of equation (6). At this point, however, we relax the strict Boolean (non-convex) feasible set for the optimization variable $u$ to an interval constraint, $u \in [0,1]^n$. Strong duality gives the following result:

$$\psi(\rho,\alpha) = \max_{v} 2m^T v - v^T v - \sum_{i=1}^{n} \left( \frac{(q_i^T v)^2}{\rho^2} - \alpha \right)_+$$

Finally, we can use SOCP duality to show that the above is precisely equivalent to the following:

$$\psi(\rho,\alpha) = \min_{x} \|Q^T x - m\|_2^2 + 2\alpha \sum_{i=1}^{n} B\left(\frac{\rho x_i}{\sqrt{\alpha}}\right) \qquad (7)$$

where the function $B$, called the "reversed Huber function" (or in CVX, `berhu`) is defined as follows:

$$B(\xi) \doteq \frac{1}{2}\min_{z\in[0,1]} \left( z + \frac{\xi^2}{z} \right) = \begin{cases} |\xi| & \text{if } |\xi| \le 1 \\ \frac{\xi^2+1}{2} & \text{otherwise} \end{cases}$$

We now proceed to test the relaxation expressed in equation (7).

### B. Results

We follow exactly the same procedure as in the corresponding section using the $L_1$ relaxation. Here, we solve the following instance of equation (7):

$$x^* = \arg\min_{x} \|ACx - m\|_2^2 + 2\alpha \sum_{i=1}^{n} B\left(\frac{\rho x_i}{\sqrt{\alpha}}\right) \qquad (8)$$

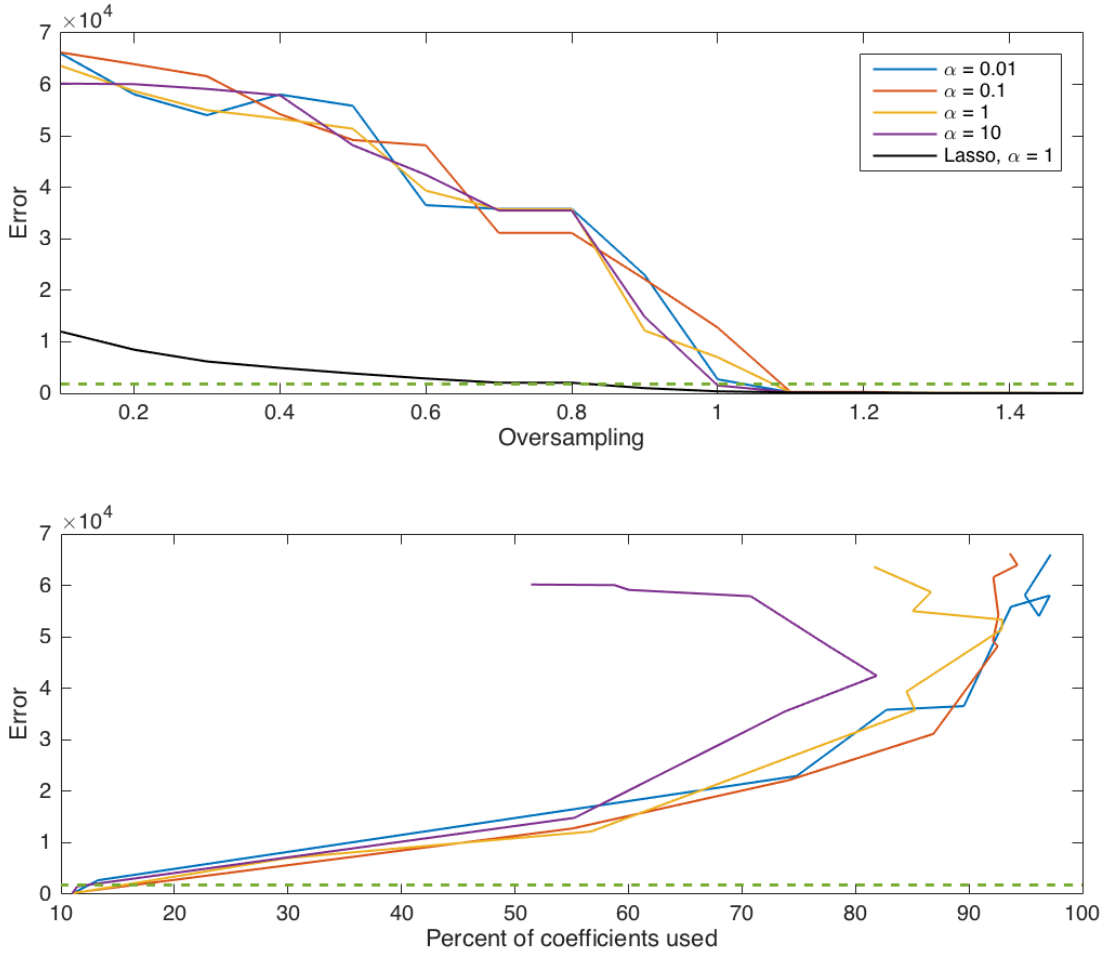



Fig. 10: Oversampling rate and sparsity versus reconstruction error for compressed sensing in the DCT domain using the reversed Huber relaxation, evaluated for $\alpha$ spanning four orders of magnitude. Note that the reconstruction error is still extremely sensitive to OSR (as we saw in the image domain). And again, as in the image domain, the reconstructions are no longer always sparse. However, in this case the best reconstructions are achieved with the fewest number of coefficients.

where as before $A$ is the random mixing matrix comprised of IID standard Gaussians and $C$ is the orthonormal DCT basis such that $Cx$ is the inverse DCT of $x$.

Figure 10 shows the results of compressing the Lenna image, using the same sets of parameters as above. Here, we observe that – as with compressed sensing in the image domain – the reconstruction error is very sensitive to OSR. Also, as before, the percentage of coefficients used at optimum is always greater than $10\%$. However, error actually increases as we use more coefficients, which is completely non-intuitive.

To reiterate, the solutions $x^*$ to (8) are *worse* when they are less sparse. Moreover, the correlation between increasing number of nonzero coefficients and increasing error seems to break down at high errors. We believe that this can be explained as one of the following.

1) The reversed Huber relaxation is not nearly as tight as the $L_1$ relaxation. This makes some intuitive sense because it looks like $L_2$ for large coefficients of $x^*$ – i.e. it penalizes large coefficients more than $L_1$, when all we really care about is the number of nonzero coefficients. This might explain why at high errors the percent of nonzero coefficients is less clearly correlated with error.
2) The reversed Huber norm is more computationally complex than $L_1$ or $L_2$, especially when expressed as a "graph implementation" as required by the disciplined convex programming approach of [17]. This may lead to computational inaccuracies during optimization, leading to non-optimal results.

## VII. Comparison of Relaxations

We have seen two different relaxations of equation (1): $L_1$ (4) and reversed Huber (8). To summarize, our results

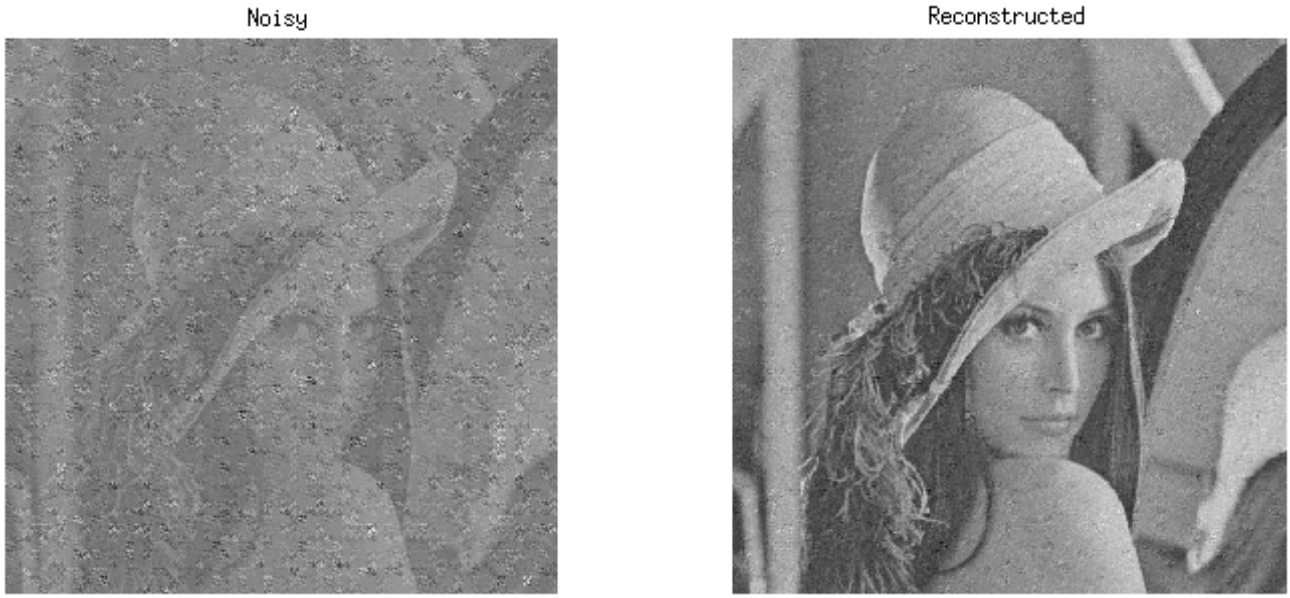



Fig. 11: Reconstruction after applying $L_1$ DCT-domain compressed sensing, with $\alpha = 1, OSR = 1$, where measurement noise $\eta_i \sim \mathcal{N}(0, 20)$ has standard deviation of 20 (raw pixel values are between 0 and 255).

indicate that the $L_1$ relaxation is far superior to the reversed Huber relaxation, both in terms of final compression ratio and in terms of potential for undersampling. That is, when performing compressed sensing in the DCT domain using the $L_1$ relaxation, we can undersample by $20\%$ and still reconstruct a high-fidelity image. The same is not true when operating in the image domain, or when using the reversed Huber relaxation in the DCT domain.

## VIII. Application: image denoising

### A. Theory

One interesting application of compressed sensing is image denoising. In all physical image acquisition systems – indeed in all data acquisition systems of any kind – there is always some sort of measurement inaccuracy, or noise. In all of our work thus far, we have simply assumed that our measurements were noiseless, i.e. that $m = Ay$ exactly. It would be disastrous for the field of applied compressed sensing if the results were highly sensitive to noise.

Formally, we model noise as an independent Gaussian vector $\eta \in \mathbb{R}^p$, yielding the following measurement model:

$$\begin{aligned} m &= Ay + \eta \\ \eta_i &\sim \mathcal{N}(0, \sigma) \end{aligned}$$

This is equivalent to a noiseless measurement of a noisy image. Assuming that $A$ is invertible (which it almost certainly is, for $OSR = 1$), we can calculate the equivalent noisy image $\tilde{y}$ as follows:

$$\begin{aligned} \tilde{y} &= A^{-1}m \\ &= y + A^{-1}\eta \end{aligned}$$

Since the the entries of $\eta$ are jointly Gaussian, $A^{-1}\eta$ is again a Gaussian vector, and all entries are jointly Gaussian.

Intuitively, we can expect that compressed sensing might be robust to this sort of additive white Gaussian noise for a very simple reason. That is, white noise is inherently non-sparse in the frequency (DCT) domain, which means that compression via compressed sensing ought to remove much of the noise (much as a low pass filter acting on $\tilde{y}$ would remove the noise). In other words, the noise will be poorly approximated by compressed sensing, as opposed to the image $y$ which is sparse and which will be well approximated by compressed sensing.

### B. Results

As shown in Figure 11, $L_1$ DCT-domain compressed sensing as in equation (4) is quite robust to noise. The reconstruction is completely recognizable, even though the corrupted version, $\tilde{y}$, is barely recognizable. Of course, compressed sensing does not remove *all* the noise – if it could do that, we would not need Wiener filters or Kalman filters. Still, we can be confident that the results above hold for noisy measurements, and that it is reasonable to apply the principles of compressed sensing in real-world image acquisition systems.

## IX. Future work

This project has been a wonderful opportunity for us to learn more about compressed sensing as it applies to imaging and more specifically to post-hoc image compression. We see many applications of compressed sensing, particularly with regard to acquiring image data in compressed form via measuring the image directly in a randomized sparse basis. In this paper we analysised two different convex relaxations and compared them using the standard CVX solver. In the future it would be worthwhile to also compare different solvers and examine the trade-offs between quality and speed.

One drawback of this work is that it is difficult to imagine a physical system for doing this, for example, in a CMOS camera sensor. However, if the compression ratios are sufficiently impressive, it will be worthwhile for someone to put in the effort to design a sensor. In general, we believe the field of imaging will need to go in this direction in the not-too-distant future, because as the number of pixels in typical sensors increases it becomes increasingly difficult to acquire an entire image, store it, and transmit it off-chip.

## REFERENCES


[1] Emmanuel Candes and Michael Wakin, "An Introduction to Compressive Sampling," in IEEE Sig. Proc. Mag., pp. 21-30, Mar. 2008.

[2] Anthony Griffin and Panagiotis Tsakalides, "Compressed Sensing of Audio Signals Using Multiple Sensors," in Proc. 16th European Signal Processing Conference, Lausanne, Switzerland, pp. 1-5, Aug. 2008.

[3] David J. Brady, Kerkil Choi, Daniel L. Marks, Ryoichi Horisaki and Sehoon Lim, "Compressive Holograhy," in Optical Express, vol. 17, no. 15, pp. 13040-13049, 2009.

[4] Michael Lustig, David Donoho, and John M. Pauly, "Sparse MRI: The Application of Compressed Sensing for Rapid MR Imaging," in Magnetic Resonance in Medicine, vol. 58, pp. 1182-1195, 2007.

[5] Yusuke Oike and Abbas El Gamal, "CMOS Image Sensor With Per-Column $\Sigma\Delta$ ADC and Programmable Compressed Sensing," in IEEE Journal of Solid-State Circuits, vol. 48, no. 1, pp. 318-328, Jan. 2013.

[6] S. Derin Babacan, Reto Ansorge, Martin Luessi, Rafael Molina, and Aggelos K. Katsaggelos, "Compressive Sensing of Light Fields," in Proc. 16th IEEE Int. Conf. on Image Processing (ICIP), pp. 2337-2340, Nov. 2009.

[7] Marco F. Duarte, Mark A. Davenport, Dharmpal Takhar, Jason N. Laska, Ting Sun, Kevin F. Kelly, and Richard G. Baraniuk, "Single Pixel Imaging via Compressive Sampling," in IEEE Signal Processing Magazine, pp. 83-91, March 2008.

[8] Stern, A., Rivenson, Y., and Javidi, B., "Optically compressed sensing using random aperture encoding," in Proc. of SPIE, 2008.

[9] Dharmpal Takhar, Jason N. Laska, Michael B. Wakin, Marco F. Duarte, Dror Baron Shriram Sarvotham, Kevin F. Kelly, and Richard G. Baraniuk, "A New Compressive Imaging Camera Architecture using Optical-Domain Compression," in Proc. Computational Imaging IV, 2006.

[10] Gregory Wallace, "The JPEG Still Image Compression Standard," in Comm. of the ACM, vol. 34, no. 4, pp. 31-44, Apr. 1991.

[11] Wikipedia. (21 Nov. 2015). *JPEG* [Online]. Available: https://en.wikipedia.org/wiki/JPEG

[12] Strang, Gilbert. (21 Nov. 2015). *Lecture 31: Change of basis; image compression* [Online]. Available: http://ocw.mit.edu/courses/mathematics/18-06-linear-algebra-spring-2010/video-lectures/lecture-31-change-of-basis-image-compression/

[13] David L. Donoho, "Compressed Sensing," in IEEE Transactions on Information Theory, vol. 52, no. 4, pp. 1289-1306, April 2006.

[14] Mert Pilanci, Martin J. Wainwright, and Laurent El Ghaoui, "Sparse Learning via Boolean Relaxations," Math. Prog., vol. 151, no. 1, pp. 63-87, June 2015.

[15] Giuseppe Calafiore and Laurent El Ghaoui, *Optimization Models*. Cambridge, UK: Cambrige Univ. Press, 2014.

[16] Aharon Ben-Tal and Arkadi Nemirovski, *Lectures on Modern Convex Optimization*. Philadelphia, PA: MPS-SIAM Series on Optimization, 2001.

[17] Michael Grant, Stephen Boyd, and Yinyu Ye, "Disciplined Convex Programming," in *Global Optimization: From Theory to Implementation*. Springer Online, 2006, pp 155-210.


## APPENDIX

All code for this project is open-source, and can be found at http://github.com/dfridovi/compressed_sensing. Note that this repository contains many images and saved matrices, which makes it quite large ($\sim$ 1 GB).