Tree-Structured Wavelet Compressive Sensing

David A. Neal and Josh Hunsaker

***Abstract*—Compressive sensing making use of wavelet transforms are commonly used to help compress images as they are being captured. Structure in these transforms can be exploited and is commonly described by zero-trees. Convex optimization does not typically allow for inclusion of this information, but Bayesian approaches allow for priors to exploit structure. Tree-structured wavelet compressive sensing is presented as a new approach based on the introductory work by He and Carin [1]. This work presents a tutorial approach to their work. An attempt was made to impliment and independantly verify the results, but was unsuccessful. Performance as reported in the original work is reported and the code generated for the verification is included.**

I. Introduction

Many traditional sensor systems capture large amounts of data and then compress the data for transmission or storage purposes. Compressive sensing provides an approach for combining the sensing and compressing stages into a single step. This can provide both cost and efficiency benefits in sensor hardware. In the context of digital image processing, this can provide a way to capture a significantly reduced number of pixels while maintaining the ability to reconstruct the complete image later. Such reconstruction techniques require the use of a basis in which the image is sparse. One common approach is to use wavelet transforms as the basis for the image.

The wavelet transform of most natural images exhibits a "zero tree" structure in which the "children" of negligible coefficients tend to be negligible as well. In [1], the authors develop a statistical algorithm which exploits the zero-tree phenomenon to achieve increased reconstruction accuracy. By imposing a set of Bayesian priors on the wavelet coefficients, the expected structure is imposed statistically, which leads to a more flexible image model.

This paper reviews the theory behind the different elements of tree-structured wavelet compressive sensing (TSW-CS) and how they fit together. Starting with wavelet transforms and moving through compressive sensing and zero-trees, the paper concludes with an overview of Bayesian model and expresses the algorithm described in [1] as a Gibbs sampling model that itereratively estimates the original wavelet coefficients.

Results are reviewed from the original work. An attempt to reproduce the results was made, but was unsuccessful. The paper concludes with a discussion of what may have gone wrong. The implemented code for the algorithm is included as an appendix.

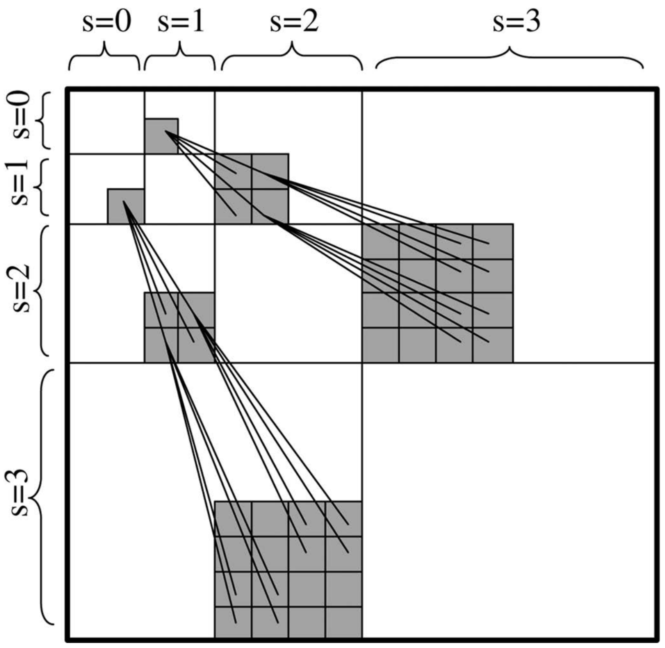


Fig. 1: Wavelet decomposition of an image, showing the tree structure. The top-left block ($s = 0$) represents scaling coefficients, and other regions are wavelet coefficients

II. Wavelet Transforms Of Natural Images

A. Discrete Wavelet Transform

A wavelet is a wave-like signal which has finite length. Wavelets are often used to model natural phenomena of finite duration. While many wavelets are continuous and differentiable, we can also construct a class of wavelets which are discretely sampled for analysis of discrete signals.

Consider an orthogonal basis Ψ, whose columns ψ_j are discrete wavelets, we can decompose a signal $\mathbf{x}$ into a sum of wavelets as follows

$$\theta = \Psi^T \mathbf{x} \tag{1}$$

Each value θ_j represents the amount of the wavelet ψ_j present in $\mathbf{x}$. This is analogous to the discrete fourier transform, which breaks a signal into its component frequencies. Unlike the fourier transform however, wavelet transforms simultaneously capture both frequency and spacial information. Also, wavelet decompositions of natural signals tend to be sparse. In other words, many natural signals can be constructed or approximated as the sum of a relatively small number of wavelets.

B. Wavelet Tree Structure

Discrete wavelet transforms of images have another property which may be exploited, which is the statistical relationship that exists between certain coefficients of the transform. The coefficients can be organized into a quadtree structure in which most coefficients serve as the parent for four other child coefficients. Within this structure, the coefficients which are the "children" of negligible coefficients will also be negligible with high probability.

Figure 1 shows an example of this tree structure in a transformed image. The coefficients at scale $s = 0$ serve as scaling coefficients for the coefficient trees, which each have their root node at scale $s = 1$. The leaf nodes at the finest scale ($s = 3$ in this example) represent the high-pass elements of the image.

III. Compressive Sensing

A. Sparse Signals

Consider a discrete signal $\mathbf{y}$ of length N in which all but S of the values are zero. We say that $\mathbf{y}$ is "S-sparse". Assume that we have prior knowledge about the sparsity of $\mathbf{y}$, but we do not have any specific information about *which* of the samples are non-zero. If we sample $\mathbf{y}$ directly, we must take N samples or we risk losing information.

Compressive sensing allows us to exploit the sparsity in $\mathbf{y}$ in order to collect less than N data points. In the absence of measurement noise we are able to reconstruct $\mathbf{y}$ perfectly, given sufficient data samples. Let Φ be a sensing matrix of size $M \times N$ with $M \ll N$. The following equation represents the compressive sampling operation:

$$\mathbf{v} = \Phi \mathbf{y} \tag{2}$$

Now $\mathbf{v}$ is significantly smaller than $\mathbf{y}$ but should still contain enough information to reconstruct $\mathbf{y}$, as long as Φ is known. Of course, this depends upon $\mathbf{y}$ being sufficiently sparse, and upon Φ being chosen appropriately (which we will discuss in more detail in III-C.)

We can also relax our definition of sparsity by considering any values that are sufficiently small to be insignificant. Under this new definition, we will consider a signal to be S-sparse if all but S of its values have a magnitude below some threshold ϵ. The signal is sparse in the sense that almost all of its energy is concentrated in a relatively small number of samples. While such an assumption may lead to some error in the reconstruction, the energy in the error will be proportional to the energy in the negligible coefficients.

B. Image Transform

When working with image data, we have no guarantee of sparsity. In fact, for most interesting images the data will tend not to be sparse. This would seem to suggest that image data is not a good candidate for compressive sensing. However, as mentioned in section II, wavelets provide a basis in which most images have a sparse representation. This means that if we had a way of sensing the wavelet coefficients instead of the pixel values, we would be sampling a sparse signal.

Consider a new sensing matrix

$$\Phi' = \Phi \Psi^T \tag{3}$$

If we apply Φ' to the image data $\mathbf{x}$, we get

$$\begin{aligned} \mathbf{v} &= \Phi' \mathbf{x} \\ &= \Phi \Psi^T \mathbf{x} \\ &= \Phi \theta \end{aligned} \tag{4}$$

This means that sensing the image data with Φ' is the same as sensing the (sparse) transform coefficients with Φ. Therefore, by using the measurements $\mathbf{v}$, we should be able to reconstruct the coefficient data and, by extension, the image.

C. Incoherent Sampling

It has been demonstrated [2], [3] that for compressive sensing of sparse signals, taking random measurements is in some sense an optimal strategy. For example, the rows of the sensing matrix Φ may be a collection random test vectors ϕ_k such as gaussian white noise or a series of bernoulli random variables.

Under such a sensing scheme, the number of samples required will be inversely proportional to the similarity between the sensing modality (Φ) and the signal model (Ψ). We can estimate this similarity by measuring the mutual coherence $\mu(\cdot)$ as follows [4], [5]

$$\mu(\Phi\Psi) = \underset{k,j}{max} |\langle \phi_k, \psi_j \rangle|. \tag{5}$$

In the implementation of this project, we have constructed the rows of Φ as a series of bernoulli random variables. We found the mutual coherence between our sensing matrix and our wavelet basis to be around 2.75. While it is possible to construct a sensing matrix whose coherence with the wavelet basis is ideal ($\mu(\Phi\Psi) = 1$), such matrices are almost always complex-valued which would significantly complicate the algorithm being used.

D. Reconstruction

The most straightforward way to reconstruct a sparse signal from compressed samples is via an ℓ_1 minimization known as *basis pursuit*. By solving the simple convex optimization problem

$$\theta = \arg\min_{\tilde{\theta}} \|\tilde{\theta}\|_{\ell_1} \text{ s.t. } \mathbf{v} = \Psi\tilde{\theta}, \tag{6}$$

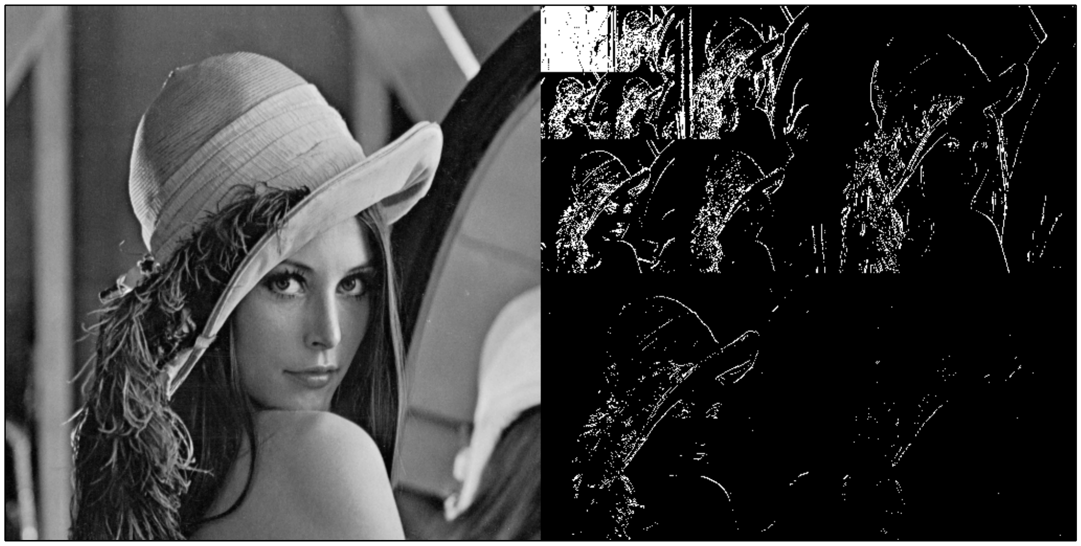


Fig. 2: Example of multi-level wavelet transform. The original image is on the left, with the transformed image on the right.

we acquire the coefficient vector θ with minimum ℓ_1 norm which satisfies the measurement data. Since the ℓ_1 norm emphasizes sparsity, we are essentially solving for the sparsest vector that can explain the data.

IV. Bayesian Inference

In a maximum likelihood sense, the goal is to use the samples $\mathbf{v}$ to estimate the most likely θ. If $p(\theta|\mathbf{v})$ is known, then the maximum likelihood value of $\theta|\mathbf{v}$ can be found. As in many cases, this density is not known. The most common approach is to apply Bayes' theorem, which yields $p(\mathbf{v}|\theta)p(\theta)/p(\mathbf{v})$. In this case, these densities are also not known, but some things are known about them.

If a density is selected based on what is known , the parameters are still unknown. However, in this case, the problem can be stated as $p(\theta,\xi|\mathbf{v})$, with ξ representing a vector of all the density parameters that are not known. Again applying Bayes' theorem, the problem becomes $p(\mathbf{v}|\xi,\theta)p(\theta,\xi)/p(\mathbf{v}|\xi)$. As in many problems, the denominator is simply a scaling constant and does not affect the choice of θ and ξ that maximize the likelihood.

Conditionally factoring the result and splitting up ξ into the different parameters yields

$$p(\mathbf{v}|\alpha_n,\theta)p(\theta|\mu,\pi,\alpha_s)p(\alpha_s)p(\alpha_n)p(\pi)p(\mu). \tag{7}$$

In [1], they assume that $p(\mathbf{v}|\alpha_n,\theta)$ is distributed Gaussin, so α_n is the precision and the mean $\phi\theta$. The mean is based on (2) and the precision is based on the noise in the measurement, which is Gaussian. They assume that θ is a spike and slab distribution, which means that some percentage of the time outcome is zero and the rest of the time it is a Gaussian draw. For this distribution, $\mathbf{pi}$ represents the probability that the Gaussian draw is chosen, μ is the mean of that Gaussian, and α_s is the precision. This is based on the fact that θ is sparse, so there are many zeros or small coefficients, and the rest of the coefficients are assumed to be Gaussian based around different means and variances common to their scale in the transform (hence the subscript s on α_s).

One issue remains to be able to solve this problem. The distributions of the parameters must be determined. A common distribution for precision parameters is the Gamma distribution. This is a common choice due to it being a conjugate distribution to a Gaussian distribution. This means that when it is used as a prior to the Gaussian, as with Bayes' theorem, the result is still a Gaussian distribution. This is desirable because it makes solving the problem somewhat easier.

The distribution on π is set to a Beta distribution. This works well because the two parameters of the Beta distribution can be set to pick between them with one providing numbers closer to one and the other providing numbers closer to zero. This can be used to weight the choice in the spike and slab distribution between the zero and the Gaussian.

There are means for estimating all the parameters and solving for the maximum likelihood solution. This typically involves some significantly difficult math. In many cases, it is desirable to solve the problem directly, but given that for a 128×128 image, there are 128^2 elements in many of the vectors, the problem complexity is high. An alternative approach that is typically applied is Gibbs sampling.

A. *Gibbs Sampling*

Gibbs sampling is an approach for determining a density by sampling from it. If a problem is set up similar to how (7) has been found, random draws for each parameter can be used flowed to estimate average paramaters and allow for draws from the primary distributions. In this case, the lower level parameter distributions are able to exploit information about structure or values and are considered prior distributions. Exploiting this prior information is one of the main benefits to selecting a Bayesian approach over convex optimization approaches.

The random draws can be run for a period of time with each iteration updating the estimation of the top level parameters. After a burn in period, it has been shown that this converges to the true distribution [6]. Since the true distribution of θ given its parameters has been determined, a series of random samples can be drawn and the maximum likelihood value can be estimated from the distribution.

B. Implemented Gibbs Sampling

In order to implement the Gibbs sampling approach for this model, the actual prior probabilities for the parameters had to be determined. Each parameter makes use of the prior information available to it and the structure that the model is imposing on it.

The first thing to consider is the structure of the model. The goal is to find θ, but the structure of the wavelet transform needs to be exploited. Since the structure of the wavelet transform changes with the scale, it is important to consider the scale of each element of θ. As such, the model bases all of the parameters on the scale. They also look at each element of each scale individually.

A graphical approach is often used to show the interrelationships between the different elements in a Bayesian model. Figure 3 shows a graph of the major elements that are either calculated or randomly drawn. All other coefficients are fixed and feed into the appropriate lower level elements. The equations for drawing or calculating these values given below express these interrelationships.

The data is used to help determine the noise precision $\mathbf{alpha}_n$. Since this noise is due to things like measurement noise, it does not change with scale or element, so there is only one value. The distribution is given as

$$p(\alpha_n^{(j)}|-) = \Gamma(a_0 + N/2, b_0 + \|(\mathbf{v} - \phi\theta^{(j)})\|/2. \qquad (8)$$

The Gamma distribution has a a_0 and b_0 as 10^{-6} to keep the arguments away from zero. The N value is the number of elements in θ, while the vectpr-matrix value in the second parameter is the mean-square error divided by two. When the error is small, the large number for the first parameter means that values of $\mathbf{alpha}_n$ will most likely be large. This makes sense, since the estimate is close and the noise is probably small. As the error increases in size, the estimate of the noise is more likely to be large, but is still evenly balanced with potentially small values. This both exploits information from the data $\mathbf{v}$ and changes its value based on the amount of error in the current estimation.

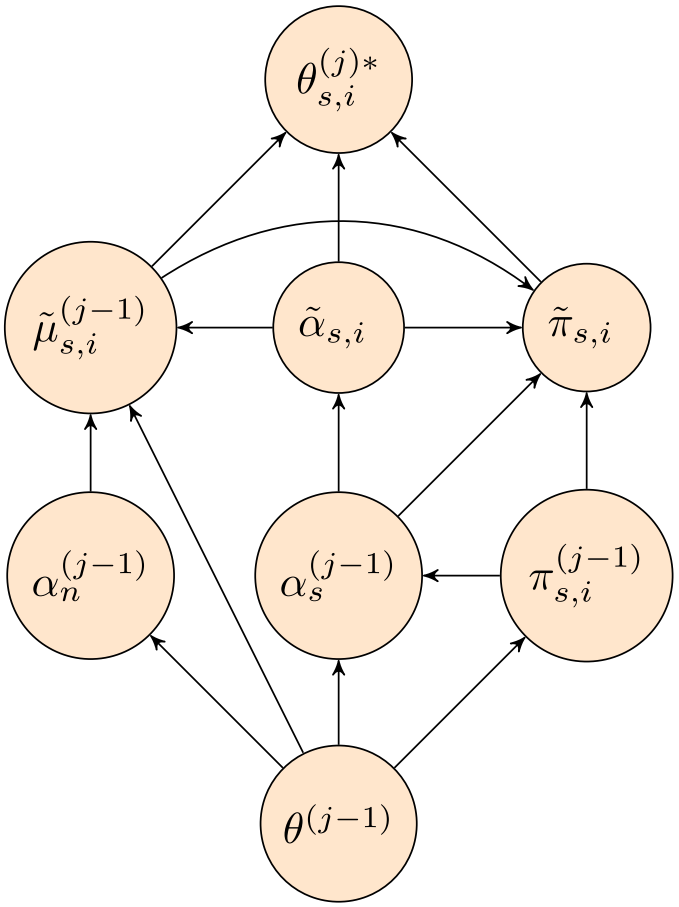


Fig. 3: Simplified Graph of the Bayesian Model

For the estimation of α_s, the s subscript indicates that the value is changing at each scale. The distribution is given as

$$p(\alpha_s^{(j)}|-) = \Gamma(c_0 + 0.5\sum_{i=1}^{M_s}\mathbf{1}(\theta_{s,i}^{(j)} \neq 0), d_0 + 0.5\sum_{i=1}^{M_s}\theta_{s,i}^{2,(j)})). \qquad (9)$$

Again, the values of c_0 and d_0 are set to 10^{-6}. The value of M_s is the number of elements at scale s. The $\mathbf{1}(x)$ is zero if x is false and one if x is true. So the first argument is counting the number of non-zero elements in θ at scale s and element number i and the second is counting the squared magnitude of those elements. This means that when the values are small, the precision will tend to be large, but when the elements are large, the precision can be smaller. This allows for variation across the elements relative to their magnitude.

The final parameter to draw for is π. This parameter determines the probabilility of an element being from the zero distribution or the Gaussian distribution when drawing for θ. The scale zero elements are typically non-zero, as these are scaling elements for the entire image, so those elements are set to 1. These probabilities are also related to the parent-child relationships in the transform, so they change with scale. Additionally, the scale 1 elements have no parents. This results in the distribution for $s = 1$ being given as

$$p(\pi_r^{(j)}|-) = \beta(e_0^r + \sum_{i=1}^{M_s} \mathbf{1}(\theta_{s,i}^{(j)} \neq 0), f_0^r + \sum_{i=1}^{M_s} \mathbf{1}(\theta_{s,i}^{(j)} = 0)). \tag{10}$$

Here, $e_0^r = .9M_s$ and $f_0^r = .1M_s$. The two indicator functions compare the number of elements that are non-zero for the first argument and that are zero for the second. When the first argument is larger, the Beta distribution selects a probability tends toward 1 and when the second is, it tends toward 0. This effectively selects a probability based on the current estimate of θ.

For scales larger than 1, there are two distributions. The first is

$$\begin{aligned} p(\pi_s^{0,(j)}|-) \quad = \quad & \beta(e_s^{s0} + \textstyle\sum_{i=1}^{M_s} \mathbf{1}(\theta_{s,i}^{(j)} \neq 0, \theta_{pa(s,i)}^{(j)} = 0), \\ & f_s^{s0} + \textstyle\sum_{i=1}^{M_s} \mathbf{1}(\theta_{s,i}^{(j)} = 0, \theta_{pa(s,i)}^{(j)} = 0)), \end{aligned} \tag{11}$$

which draws for the case where the parent of the s, ith element is zero and the other is

$$\begin{aligned} p(\pi_s^{(1,j)}|-) \quad = \quad & \beta(e_s^{s1} + \textstyle\sum_{i=1}^{M_s} \mathbf{1}(\theta_{s,i}^{(j)} \neq 0, \theta_{pa(s,i)}^{1,(j)} \neq 0), \\ & f_s^{s1} + \textstyle\sum_{i=1}^{M_s} \mathbf{1}(\theta_{s,i}^{(j)} = 0, \theta_{pa(s,i)}^{(j)} \neq 0)), \end{aligned} \tag{12}$$

which draws for the case where the parent is non-zero. In these equations, $e_0^{s0} = M_s/N$, $f_0^{s0} = (1 - 1/N)M_s$, $e_0^{s1} = .5M_s$, and $f_0^{s1} = .5M_s$. Again, the balance in the equation is set to draw a probability based on the what the number of zero versus non-zero elements suggests. Then, for each s, i, the value for $\pi_{s,i}$ is selected from these two by

$$\pi_{s,i}^{(j)} \quad = \quad \begin{cases} \pi_s^{0,(j)} & if \quad 2 \leq s \leq L, \theta_{pa(s,i)} = 0 \\ \pi_s^{1,(j)} & if \quad 2 \leq s \leq L, \theta_{pa(s,i)} \neq = 0 \end{cases} \tag{13}$$

Now, in order to draw for each $\theta_{s,i}$, the value for the mean, precision, and π must be estimated for that specific element. This can be determined by taking an expected value for the theta with respect to the specific elements. The result is

$$\tilde{\alpha}_{s,i} = \alpha_s + \alpha_n \phi_m^T \phi_m, \tag{14}$$

$$\tilde{\mu}_{s,i} = \tilde{\alpha}_{s,i}^{-1} \alpha_n \phi_m^T (\mathbf{v} - \sum_{k=1, k \neq m}^{N} \phi_k \theta_k) \tag{15}$$

$$\begin{aligned} & \tilde{\pi}_{s,i} = 1, s = 0 \\ & \tfrac{\tilde{\pi}_{s,i}}{1-\tilde{\pi}_{s,i}} = \tfrac{\pi_{s,i}}{1-\pi_{s,i}} \tfrac{\mathcal{N}(0,\alpha_s^{-1})}{\mathcal{N}(\tilde{\mu}_{s,i},\tilde{\alpha}_{s,i}^{-1})}, \quad 1 \leq s \leq L. \end{aligned} \tag{16}$$

Here, ϕ_m is the mth column of ϕ and θ_k is the kth element of θ.

These estimations of the parameters of $\theta_{s,i}$ can then be used to draw from

$$p(\theta_{s,i}|-) = (1 - \tilde{\pi}_{s,i})\delta_0 + \tilde{\pi}_{s,i} \mathcal{N}(\tilde{\mu}_{s,i}, \tilde{\alpha}_{s,i}^{-1}), \tag{17}$$

V. Algorithm

The actual algorithm for the model is not calculated in the order given above, as the updates and draws occur relative to a sample of θ. The algorithm is run by initializing a value for each $\theta_{s,i}$. This is then used to initialize the random draws in (8-13). These values are then used with the initial $\theta_{s,i}$ to initialize the estimates for (14-16).

Once this initialization takes place, the algorithm loops by drawing a new set of $\theta_{s,i}$ from (17), updating the estimates for the parameters in (14-16), and then drawing new values from (8-13).

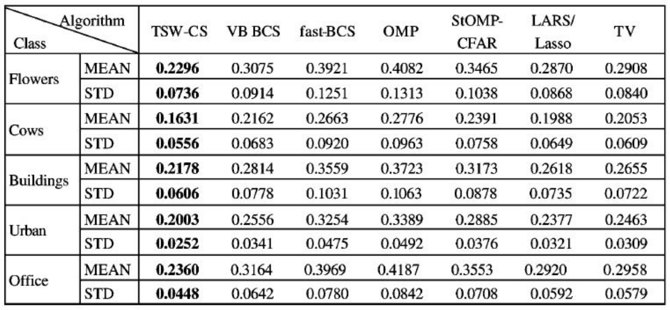
In the paper, they found that this converges to the true distribution in about 200 iterations. After this, they found that 100 samples was sufficient to estimate θ accurately.

In the paper, they note that this can be drawn sequentially by drawing for each (s, i) and then incorporating it into the draw for $(s, i + 1)$. It can also be drawin in a block method, drawing all the $\theta_{s,i}$ at once. Drawing sequentially converges faster since the newer data is better and is incorporated earlier into the updates.

VI. Results

In this paper, the algorithm proposed was compared with seven recently developed algorithms. The mean and variance for each algorithm was calculated and the algorithm was run in Matlab. All examples used 128×128 images and set scale zero as an 8×8. The calculated relative reconstruction error was found by $||\mathbf{x} - \hat{\mathbf{x}}$
$_2/||\mathbf{x}||_2$, where $\mathbf{x}$ is the original image and $\hat{\mathbf{x}}$ is the recovered image. The images used were natural images from Microsoft Research and are available at http://research.microsoft.com/en-us/projects/objectclassrecognition/. Several plots and tables taken from the original article are presented and discussed here as a summary of the algorithm performance.

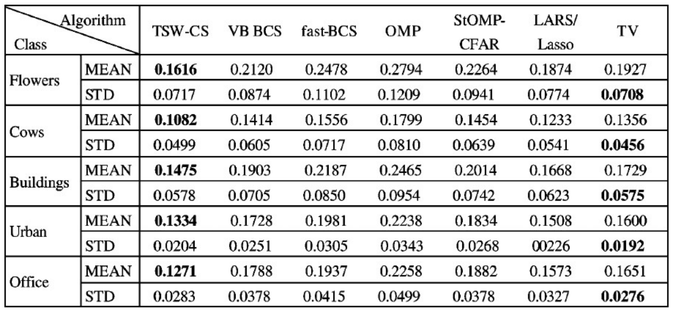
The results show that the algorithm outperforms all the algorithms with which it was compared in the sense that the mean and standard deviation are smaller for 2000 and 6000 samples, as seen in Figures 4 and 5.



Class \ Algorithm		TSW-CS	VB BCS	fast-BCS	OMP	StOMP-CFAR	LARS/Lasso	TV
Flowers	MEAN	**0.2296**	0.3075	0.3921	0.4082	0.3465	0.2870	0.2908
	STD	**0.0736**	0.0914	0.1251	0.1313	0.1038	0.0868	0.0840
Cows	MEAN	**0.1631**	0.2162	0.2663	0.2776	0.2391	0.1988	0.2053
	STD	**0.0556**	0.0683	0.0920	0.0963	0.0758	0.0649	0.0609
Buildings	MEAN	**0.2178**	0.2814	0.3559	0.3723	0.3173	0.2618	0.2655
	STD	**0.0606**	0.0778	0.1031	0.1063	0.0878	0.0735	0.0722
Urban	MEAN	**0.2003**	0.2556	0.3254	0.3389	0.2885	0.2377	0.2463
	STD	**0.0252**	0.0341	0.0475	0.0492	0.0376	0.0321	0.0309
Office	MEAN	**0.2360**	0.3164	0.3969	0.4187	0.3553	0.2920	0.2958
	STD	**0.0448**	0.0642	0.0780	0.0842	0.0708	0.0592	0.0579

Fig. 4: Results from He and Carin paper for 2000 samples

Figure 6 shows that the relative reconstruction error is smaller over all tested numbers of measurements.



Class \ Algorithm		TSW-CS	VB BCS	fast-BCS	OMP	StOMP-CFAR	LARS/Lasso	TV
Flowers	MEAN	**0.1616**	0.2120	0.2478	0.2794	0.2264	0.1874	0.1927
	STD	0.0717	0.0874	0.1102	0.1209	0.0941	0.0774	**0.0708**
Cows	MEAN	**0.1082**	0.1414	0.1556	0.1799	0.1454	0.1233	0.1356
	STD	0.0499	0.0605	0.0717	0.0810	0.0639	0.0541	**0.0456**
Buildings	MEAN	**0.1475**	0.1903	0.2187	0.2465	0.2014	0.1668	0.1729
	STD	0.0578	0.0705	0.0850	0.0954	0.0742	0.0623	**0.0575**
Urban	MEAN	**0.1334**	0.1728	0.1981	0.2238	0.1834	0.1508	0.1600
	STD	0.0204	0.0251	0.0305	0.0343	0.0268	00226	**0.0192**
Office	MEAN	**0.1271**	0.1788	0.1937	0.2258	0.1882	0.1573	0.1651
	STD	0.0283	0.0378	0.0415	0.0499	0.0378	0.0327	**0.0276**

Fig. 5: Results from He and Carin paper for 6000 samples

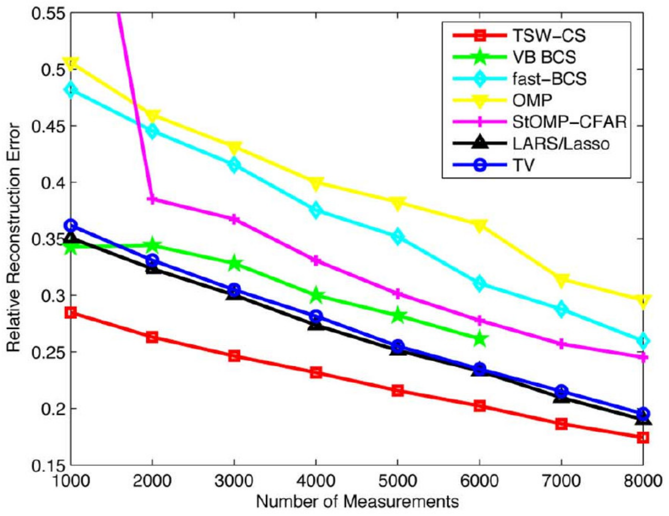


Fig. 6: Measurement Error versus Number of Measurements from He and Carin paper

Figure 7 shows the amount of time that each algorithm ran. It is most interesting that the speed did not appreciably decrease when more samples were drawn, while most of the other algorithms appear to slow down significantly as the number of samples increases.

Comparison of the estimated wavelet coefficients provides some insight. Figure 8 shows the coefficients with 2000 samples. The red is the true values. Most of the estimators have many non-zero large coefficients as the coefficient index increases, while only TSW-CS has set them to zero. While this introduces error by ignoring some significant coefficients, Figure 9 shows the superior matching for the earlier coefficients. Between this and the large magnitude of the estimates at higher indicies for other algorithms, the smaller error is easy to understand.

When the number of measurments is increased to 6000, the match is even better, as shown in Figure 10. The algorithm starts to pick up the significant values at the higher indicies as well. This is also more apparent in other algorithms.

In order to emphasize the improvement with more samples, Figure 11 shows the error bars on just the first 50 estimated coefficients. As can be seen, the error bars decrease significantly as the number of measurements goes up.

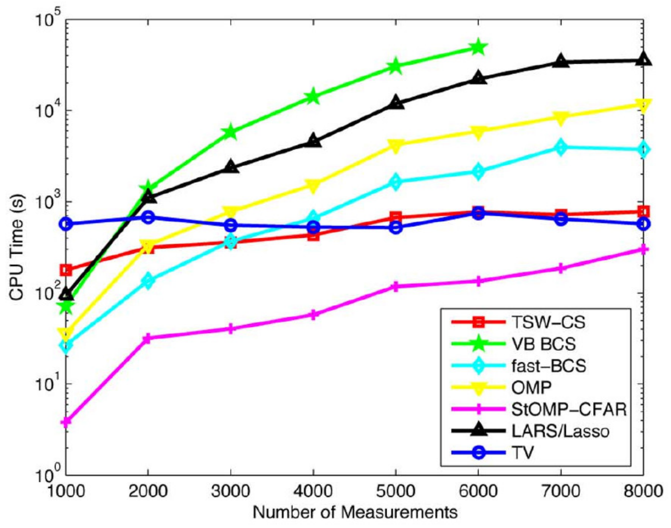


Fig. 7: Algorithm CPU Time versus Number of Measurements from He and Carin paper

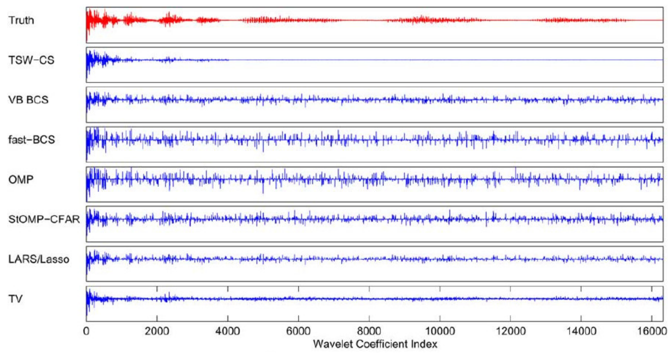


Fig. 8: Comparison of All Wavelet Coefficients for each Algorithm with 2000 Measurements

Figure 12 shows the result of six trials of the basis pursuit algorithm to construct an image from 2000 samples. Sampling at this rate represents a compression ratio of 8:1. While the images appear somewhat blurry, their relative error is fairly small.

VII. Conclusion

We were not able to achieve the desired results using Bayesian methods. As we began implementing the algorithm set forth in the paper, we realized how many parameters are involved. Where the details are not explicit, we were forced to make educated guesses about important implementation decisions. This often led to unexpected results. At times we were able to correct for such results, and other times it was unclear what was missing.

The nature of a probabilistic model is such that you set parameters to estimate a distribution. Since you do not know the exact values to give those parameters, you also estimate those, using more parameters. This creates a hierarchy of hyperparameters which helps to fine-tune the estimation

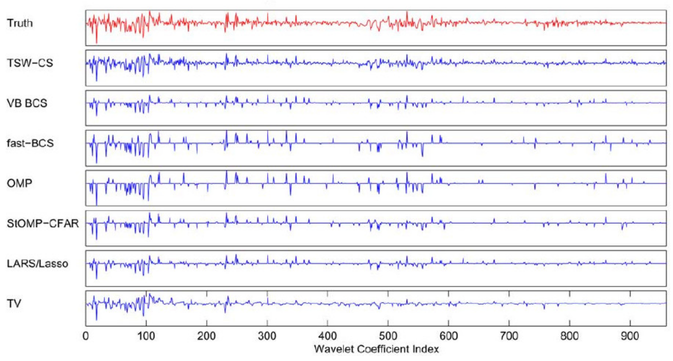


Fig. 9: Comparison of Scale 1 and 2 Wavelet Coefficients for each Algorithm with 2000 Measurements

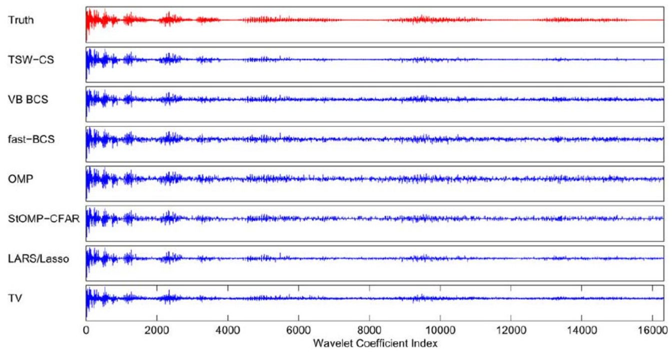


Fig. 10: Comparison of All Wavelet Coefficients for each Algorithm with 6000 Measurements

framework into a working state. Unfortunately, we were unable to fine-tune our bayesian inference framework to where it made good estimates of the wavelet transform coefficients. We believe that once the model is appropriately tuned, it should be fairly robust to incoming data.

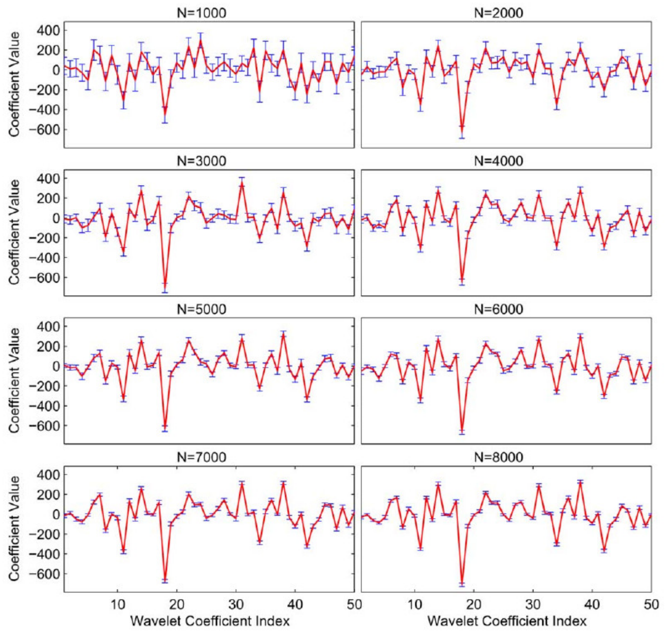


Fig. 11: Error Bars for the first 50 coefficients with various numbers of measurements

References

[1] L. He and L. Carin, "Exploiting structure in wavelet-based bayesian compressive sensing," *Signal Processing, IEEE Transactions on*, vol. 57, no. 9, pp. 3488–3497, 2009.

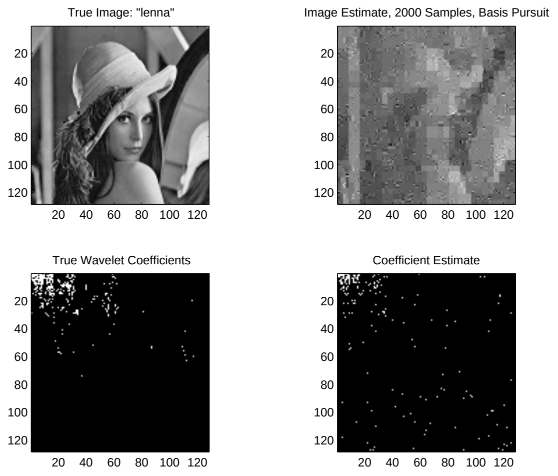
[2] E. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *Information Theory, IEEE Transactions on*, vol. 52, no. 12, pp. 5406–5425, Dec 2006.

[3] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, "The johnson-lindenstrauss lemma meets compressed sensing," *Submitted manuscript, June*, 2006.

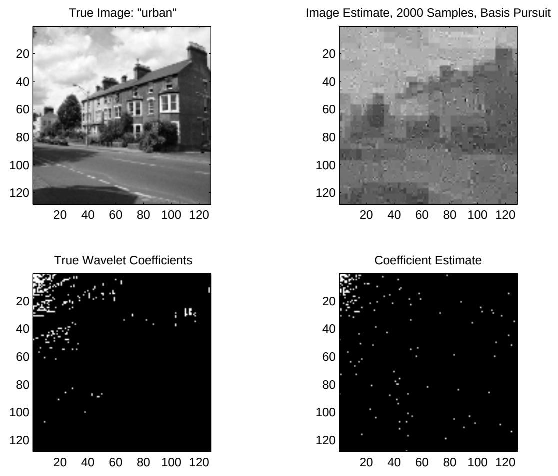
[4] D. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *Information Theory, IEEE Transactions on*, vol. 47, no. 7, pp. 2845–2862, Nov 2001.

[5] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 489–509, 2006.

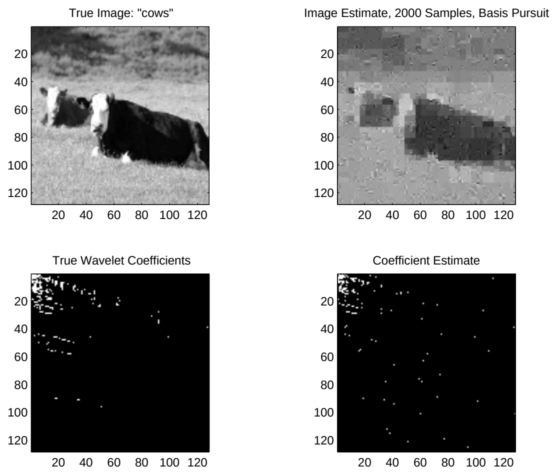
[6] D. MacKay, *Information theory, inference, and learning algorithms*. Cambridge, UK New York: Cambridge University Press, 2003.


True Image: "lenna"
Image Estimate, 2000 Samples, Basis Pursuit
True Wavelet Coefficients
Coefficient Estimate

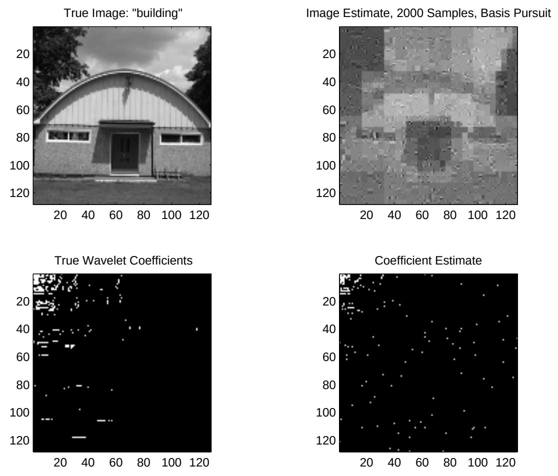
(a)


True Image: "urban"
Image Estimate, 2000 Samples, Basis Pursuit
True Wavelet Coefficients
Coefficient Estimate

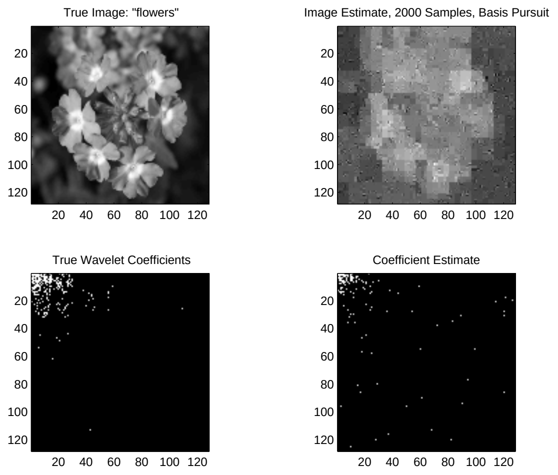
(b)


True Image: "cows"
Image Estimate, 2000 Samples, Basis Pursuit
True Wavelet Coefficients
Coefficient Estimate

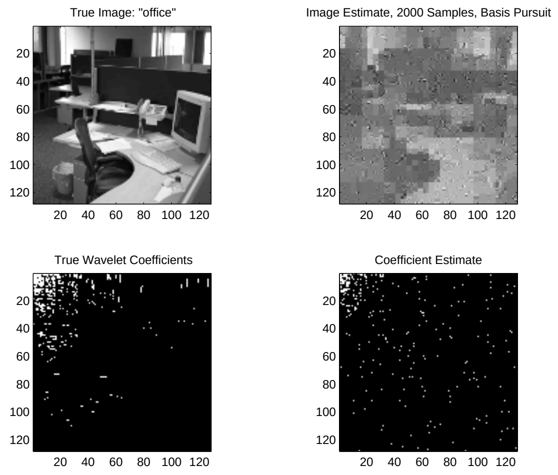
(c)


True Image: "building"
Image Estimate, 2000 Samples, Basis Pursuit
True Wavelet Coefficients
Coefficient Estimate

(d)


True Image: "flowers"
Image Estimate, 2000 Samples, Basis Pursuit
True Wavelet Coefficients
Coefficient Estimate

(e)


True Image: "office"
Image Estimate, 2000 Samples, Basis Pursuit
True Wavelet Coefficients
Coefficient Estimate

(f)

Fig. 12: Results of six trials of basis pursuit using 2000 data samples from a 128×128 image