Compressed Sensing Overview

Jasper van de Gronde and Erald Vuçini

December 19th, 2008

1 Introduction

Compressed sensing (or Compressive sampling) tries to exploit the sparsity of signals to either improve the quality of reconstructions, or reduce the required number of samples.

The samples are usually taken by multiplication with some matrix with independent — normally distributed — random values, or by selecting values from the Fourier transform of a signal. Important is that the signal should *not* be sparse in the transformed domain. How many samples need to be taken depend on the characteristics of the matrix and the signal.

Several introductory texts, as well as a lot of reference material, can be found on `http://www.dsp.ece.rice.edu/cs/`.

2 Motivation

When taking a photograph or making an MRI scan of some object a lot of samples are taken, say l. However, using the right basis only a small number of coefficients m ($m << l$) suffices for a good representation of the signal, as is demonstrated by the success of compression formats like MP3 and JPEG. Compressed sensing is about bridging this gap, letting you take a number of samples that is more closely related to m than l.

Given some measuring device that would yield, for example, wavelet or Fourier coefficients of a compressible signal, and complete knowledge of the distribution of the coefficients it would be easy to take exactly the right m samples and walk away with an excellent representation of the signal, see figure 1.

Unfortunately this presents a paradox. To take the samples you need to know which ones to take, but to select the right coefficients you have to know the distribution of the coefficients, which you usually only know *after* having sampled the signal.

Compressed sensing provides a way out by using a transform (represented by multiplication with a matrix) which is definitely not able to sparsely represent the signal, as in figure 2. This way each coefficient is "worth" more or less the same and there is little need to worry about which coefficients to

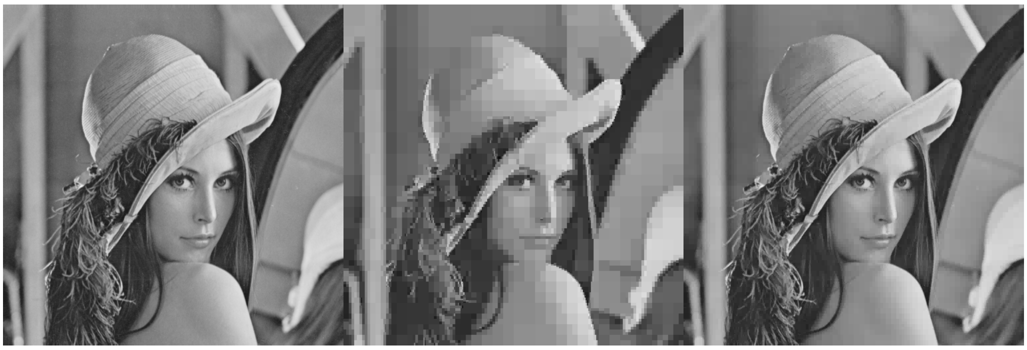

(a) Original (b) 1%, PSNR=26.26 (c) 10%, PSNR=35.26

Figure 1: Lena, the middle and right images are reconstructed from the largest 1% and 10%, respectively, of the Haar wavelet coefficients (see equation 6 for a definition of PSNR).

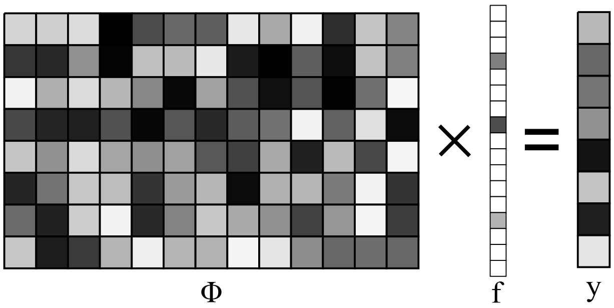


Figure 2: Graphical representation of taking compressed sensing measurements. Instead of measuring the sparse vector (here with three non-zero components) a limited number of "noisy" measurements are taken. Conceptually a matrix multiplication is used, but in an actual acquisition device it is usually not implemented that way, see section 3.

take. Then, using the knowledge that the original signal is sparse (in some basis) a good approximation of the original signal can be reconstructed from the measured data.

Theory and empirical data suggest that with this scheme the number of samples that have to be taken is larger than the number of coefficients needed to represent the signal in a "good" basis, but only by (roughly) a small constant factor (usually about 3-6) [48].

However, compressed sensing is not the only way to solve this problem, and it might not even (always) be the best way. For example, in [21] a technique is described that speculatively samples a subset of wavelet coefficients, starting at the coarsest level and using the results at each level to guess where the most important wavelet coefficients at the next level are. If samples can indeed be taken adaptively based on previous samples, and the signal has a predictable coefficient pattern, such an adaptive method can be significantly more effective than traditional compressed sensing.

3 Acquisition devices

To apply compressed sensing to real world scenarios it is not sufficient to have some matrix to "measure" the signal, you have to have a device that takes suitable measurements of the signal. These devices can be grouped into two categories: devices which inherently yield measurements suitable for compressed sensing and devices which are explicitly constructed to yield suitable measurements. With respect to imaging a popular example of the former is a CT scanner (including X-ray CT and MRI) and an example of the latter is the single-pixel camera [27].

The single pixel camera uses a micromirror array and two lenses to optically take arbitrary sums of the image pixels. Without actually having to take a photograph first. This enables the use of more expensive sensors for example. So this is a relatively direct application of compressed sensing, essentially performing the matrix multiplication in the optical domain. As can be imagined this makes the acquisition process relatively slow (it can only take one sample at a time), but in this case the idea is that it would be possible to use more advanced sensors than would normally be feasible, as you only need one.

For CT reconstructions it might be less clear that compressed sensing applies, but using the common interpretation of projections as slices of the Fourier transform it is clear that each of the projections at least capture distinct information. For compressed sensing to apply it is then only necessary that the Fourier transform of the image is *not* sparse. A promising area of research in this direction is the application of compressed sensing to dynamic MRI, see for example [29].

Another example of a device explicitly constructed for compressed sensing is a coded aperture camera. In its crudest form it can be made by simply putting a piece of cardboard in front of a lens, as in [32]. This can also be applied to video [35], leveraging the inter-frame redundancy to further reduce the required number of samples (or improve the reconstruction quality). Such schemes can yield improved resolution and/or depth information (from a single photograph).

4 Reconstruction problem

Before proceeding with the actual reconstruction algorithms this section gives a mathematical formulation of the reconstruction problem, which is generally formulated as a minimization problem. Appendix B discusses theoretical bounds on the quality of the reconstruction.

4.1 Definitions

- $f \in \mathbb{R}^l$ is the real data
- $\tilde{f}$ is an estimate/approximation of the data
- Φ is the measurement matrix
- $y \in \mathbb{R}^k$ is whatever has been sensed ($y = \Phi f$)
- Ψ is a representation basis
- $x \in \mathbb{R}^m$ is a representation of f in basis Ψ ($f = \Psi x$), which is assumed to be sparse
- $\|v\|^p = \|v\|_p^p$

4.2 A posteriori formulation

The most common formulation used in compressed sensing assumes that the measurement matrix is applied directly to the original signal and you can choose any suitable sparsity basis afterwards (hence the "a posteriori"). Obviously there are some limitations, but as long as the choice of measurement matrix is somewhat reasonable this works just fine in practice, see section 6) for results of using different sparsity inducing transforms for reconstructions based on the same measurements.

Since x ($\Psi x = f$) is assumed to be sparse, most reconstruction methods try to solve the following minimization problem (assuming no noise):

$$\begin{array}{rl} \text{minimize}(\tilde{x}) & \|\tilde{x}\|_1 \\ \text{subject to} & \Phi\Psi\tilde{x} = y \end{array}, \tilde{f} = \Psi\tilde{x} \tag{1}$$

Usually it is assumed Ψ is square and orthogonal. See [10, p. 4] for the common case where Φ is a selection of rows from an orthogonal matrix. In addition, appendix A covers a few alternative formulations.

It is possible to use the l_p (with $0 < p < 1$) norm (see [38]) in the minimization, which can improve the quality of reconstructions. See section A.3 for a link between l_p norms and probability density functions. It is also possible to consider signals with are not truely sparse, but are merely compressible. That is, the coefficients decay approximately exponentially. See for example [48] and [13].

In addition, compressed sensing can be applied in a multiscale setting, as in [48], by decomposing the signal into a multiscale representation and then applying compressed sensing separately on each level (except the finest scale). So for each level separate measurements are taken and the reconstruction consists of a sequence of minimization problems (one for each scale).

5 Reconstruction algorithms

Of course, defining a minimization problem that "solves" the reconstruction problem does not immediately give you a feasible method of actually performing a reconstruction. An algorithm for performing the minimization is still needed.

When comparing different reconstruction algorithms it is important to keep in mind that there are three (more or less equivalent) minimization problems that can be solved (note that the equivalence in equation 1 is replaced by a tolerance here), see [12, §4], [19], [6], [5] and [25]:

$$\text{minimize}(\tilde{f})\ \|\Psi^T\tilde{x}\|_1, \text{ subject to } \|y - \Phi\tilde{f}\|_2 \leq \epsilon \tag{2}$$

$$\text{minimize}(\tilde{f})\ \|y - \Phi\tilde{f}\|_2, \text{ subject to } \|\Psi^T\tilde{x}\|_1 \leq R \tag{3}$$

$$\text{minimize}(\tilde{f})\ \|y - \Phi\tilde{f}\|_2 + \lambda\|\Psi^T\tilde{x}\|_1 \tag{4}$$

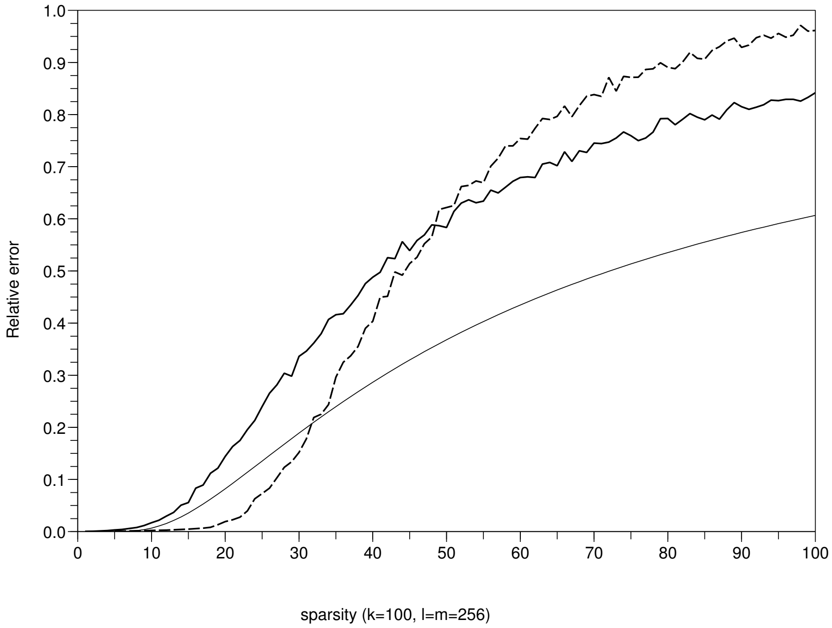


Figure 3: Relative l_2 error ($\frac{\|x-\tilde{x}\|}{\|x\|}$) versus sparsity, using l_1 (solid line) and $l_{\frac{1}{2}}$ (dashed line) minimization with IRLS. The thin solid line depicts the relative l_2 error of the best $\frac{k}{4}$-term approximation of x, note that it behaves similar (upto a scaling factor) to the error of the solution of the l_1 minimization. Also note that the results are roughly in line with the theory in [23], which also suggests a sharp transition boundary (in the limit as n and k go to infinity) between being able to reconstruct a signal perfectly and not being able to reconstruct it at all.

There are a few other possibilities, like the Dantzig selector (see for example [12, §5.2]), but the above covers most cases and will be the only ones considered in this report.

Usually problem 2 is most interesting to solve, as it gives a solution that is guaranteed to be in agreement with the measurements (upto a certain tolerance). However, so far it also appears to be the most difficult problem to solve, requiring either a very slow method or multiple iterations of an algorithm that solves one of the other two very fast.

5.1 Newton-type methods

One of the traditional workhorses of optimization has been the Newton iteration, and it can be applied directly to problem 2 by recasting it as a second order cone program and then solving it using a log-barrier method (other possibilities exist, but this method is relatively easy to use). This is done in the popular l_1-magic package. The author recommends [8] for a practical introduction to these topics.

This method works quite well but is very slow, hundreds to thousands of times slower than a simple least squares reconstruction for images of 256×256 pixels. This is mostly due to the Newton iterations which require the solution of a large matrix equation, which has a time complexity of $O(l^3)$ in the most general case. It is possible to mitigate this by using less expensive approximations of the Newton step, however, it is not clear whether such an algorithm can compete with the methods detailed below.

5.2 Iteratively reweighted least squares

This method is relatively easy to implement and reasonably fast if a weighted least squares solution can be solved cheaply, as it simply solves a sequence of weighted least squares problems. In [15] the algorithm is described in more detail and its convergence properties are discussed at great length in

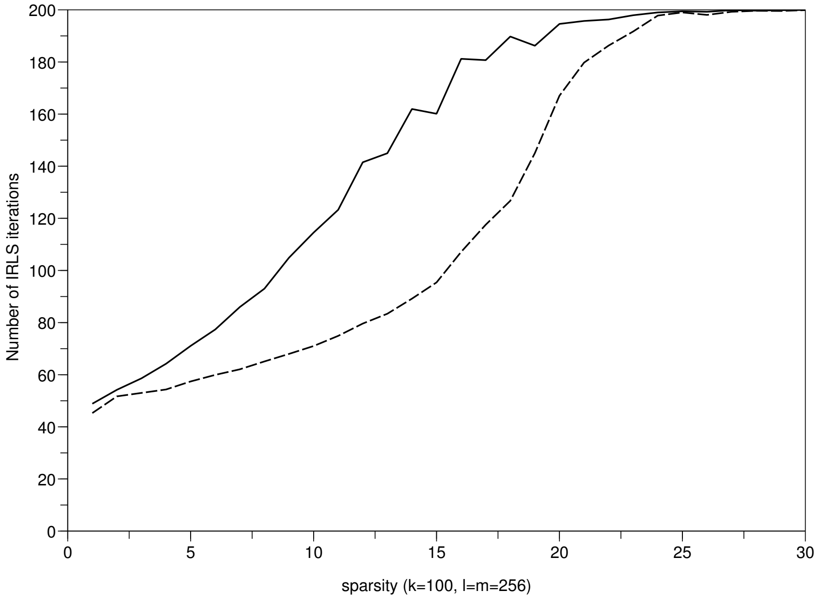


Figure 4: Number of iterations before convergence versus sparsity, using l_1 (solid line) and $l_{\frac{1}{2}}$ (dashed line) minimization with IRLS. Note that the maximum number of iterations was set to 200.

[20]. The following is the weighted least squares problem solved at each iteration:

$$\text{minimize}(\tilde{x}) \sum_{i=1}^{l} w_i \tilde{x}_i^2, \text{ subject to } \Phi\Psi\tilde{x} = y$$

After each iteration the weights are adjusted in such a way that the sequence of solutions converges to a solution of equation 1.

An implementation based on the work [15] of Chartrand and Yin was used to conduct tests similar to those in the paper. However, instead of looking at the reconstruction rate, relative error was used as an error metric and the data was constructed so that the coefficient magnitudes decay exponentially instead of simply having a few non-zero entries (in section A.3 it is shown why this particular distribution is relevant). Specifically, the data was constructed in such a way that almost 90% of the data is in the largest s coefficients (the i-th largest coefficient $x_{(i)}$ obeys $x_{(i)} = \lambda e^{-\lambda(i-1)}$, $\lambda = 2/s$).

Figure 3 shows the relative error of $l_{\frac{1}{2}}$ minimization compared to l_1 minimization, using 10 randomly selected matrices with 10 randomly selected vectors each, for each sparsity level. And figure 4 shows the number of iterations required. Clearly it can be advantageous to employ a method that can do l_p minimization (for $0 < p < 1$).

However, for large amounts of data this method can become quite slow because of having to repeatedly solve a weighted least squares problem, which can be $O(k^2 l)$ in the worst case. Furthermore the described method only solves problem 2 with an exact equality constraint ($\epsilon = 0$), while in practice it is useful to allow a larger value of ϵ, to be able to deal with noise.

5.3 Greedy methods

There is also a (huge) class of reconstruction methods that are not a minimization method per se, but that do find sparse representations of data. The best known examples of which are Matching Pursuit and its close relative Orthogonal Matching Pursuit (OMP) [45]. The latter works by repeatedly increasing the support of the estimate with the index corresponding to the highest element of $(\Phi\Psi)^T r$, where r is the "current" residual, and then solving a least squares problem within the current support of the estimate.

In general a lot of these methods only work well for very sparse signals. For instance, the least squares problem in OMP becomes larger and larger as the algorithm adds more and more non-zero coefficients, which can make it prohibitively slow if a lot of coefficients have to be added. But there are a number of variations with different performance characteristics, for example Stagewise OMP (StOMP) [26], which selects multiple coefficients at a time.

5.4 Iterative thresholding/shrinkage

With this method the following iteration is done many times (typically in the hundreds or thousands of iterations are used):

$$\tilde{x}^{i+1} = H(\tilde{x}^i + A^*(y - A\tilde{x}^i)) \tag{5}$$

Where $A = \Phi\Psi$ and H is some kind of (non-linear) thresholding or shrinkage operator. Proximity operators [17] (like a shrinkage or soft thresholding operator) solve a minimization problem of the form $\min_x \varphi(x) + \|x - y\|^2$, which for soft thresholding becomes $\min_x \lambda\|x\|^1 + \|x - y\|^2$.

The main advantages of this method are that it is easy to implement, very flexible and has a good time/quality ratio. That is, it can give reconstructions of a quality similar to the more traditional optimization methods in less time.

Using hard thresholding this can be used to (at least approximately) minimize the l_0-norm. This is detailed in [6], which uses formulations similar to problems 3 and 4 and makes a comparison between (Orthogonal) Matching Pursuit and iterative hard thresholding. In essence it is shown that iterative hard thresholding behaves very similar to (O)MP, with a few twists. The main differences are that OMP performs a least squares minimization in each iteration and that (O)MP always keeps an element in its non-zero set once it has been selected, while iterative thresholding may select a completely new non-zero set at each iteration.

Appendix C discusses a modification of iterative thresholding that gives results comparable to (or even better than) using a Newton-type method with Total Variation (TV) instead of a simple l_1-norm (in practice using TV often results in a high quality reconstruction), as is common when reconstructing images [37], [5], [33].

5.5 Recursive Filtering

This is a method introduced in [28] which iteratively filters the result image and projects it onto the feasible set $\{f \in \mathbb{R}^l \mid \Phi f = y\}$. Originally formulated only for using Fourier measurements, but in appendix D it is shown how it can be adapted to arbitrary measurement matrices. This adaptation results in an iterative scheme that is almost equivalent to a generalization of iterative thresholding.

The advantage of this method is that filters based on much more complicated signal models can be used. The disadvantage is that it would obviously be very hard to provide any theoretical bounds on the performance of this method in general, without considering a specific filter. Also, it can be quite slow in practice because of the complicated filters that may be used (see section 6, and specifically figure 14).

5.6 Hierarchical Least Squares

This is a heuristic method, developed while trying to improve the initialization of the other methods in this report. Some of these methods perform poorly if the initialization is too noisy (e.g. figure 6c). To get a less noisy, smoother, estimate of the signal very fast I tried to satisfy as much of the constraints as possible in a coarse-scale representation. This made quite a large difference (producing a clearly recognizable picture instead of the collection of points that can be seen in figure 6c). And I found that a simple thresholding scheme would improve the quality of the reconstruction even more (upto 6 PSNR points), leading to a very fast (see figure 14) reconstruction algorithm.

The method is described in algorithm 1. Note that it is assumed that the magnitude of the largest singular value of matrix Φ is ≤ 1. For any Φ with the property $\Phi\Phi^* = I$ this is obviously true and this applies to both the DCT and the noiselet ensembles used for the tests in section 6.

In line 2 the variable τ is chosen to minimize the total number of iterations needed for the for loop starting at line 8. This choice was found to give the fastest results (which should be obvious), without sacrificing quality. Note that this minimization can be performed quite easily using a simple loop that computes the iteration counts for a reasonable range of (integer) values of τ (e.g. $[1, n]$). The value of t was determined empirically to yield a good trade-off between speed and quality.

Algorithm 1 $f = \text{solve}(y \in \mathbb{R}^k,\ \Phi \in \mathbb{R}^{k \times n^2},\ \epsilon)$

```
$t := 0.85$
choose $\tau$ such that it minimizes the needed number of iterations
$f \leftarrow \mathbf{0}$
$r \leftarrow y$
$ns \leftarrow 1$
while $ns < n$ do
    $Ns \leftarrow ns^2$
    for $i = 1$ to $\max(1, \left[\frac{1}{\tau} \log_t(\frac{1}{2ns})\right])$ do
        $\Delta f \leftarrow \text{downscale}(\Phi^* r,\text{ns},\text{ns})$
        if $(1-t)Ns > 1$ then
            $\theta \leftarrow t \cdot 100$-th percentile of $|\Delta f|$
            $\Delta f \leftarrow H_\theta(\Delta f)$
        end if
        $f \leftarrow f +$ upscale($\Delta f$,n,n)
        $f \leftarrow \text{constrainToImage}(f)$
        $r \leftarrow y - \Phi f$
        if $(1-t)Ns \leq 1$ then
            break
        end if
    end for
    set $l$ to the smallest value for which $\left[2^{l/\tau}\right] > ns$
    $ns \leftarrow \left[2^{l/\tau}\right]$
end while
$i_{max} := n$
$i \leftarrow 0$
while $i < i_{max} \wedge \|r\|_2 < \epsilon$ do
    $f \leftarrow f + \Phi^* r$
    $f \leftarrow \text{constrainToImage}(f)$
    $r \leftarrow b - \Phi f$
    $i \leftarrow i + 1$
end while
```

The idea behind the bounds in line 8 is that the number of iterations should be related to how much of the coefficients would be selected in total. Assuming that during each iteration a different set of coefficients is selected by the thresholding operation, the fraction of coefficients that was "corrected" after j iterations is $1 - t^j$. So if you want to leave a fraction of α then you would need approximately $\log_t(\alpha)$ iterations. The value for α used in the algorithm was determined empirically to give good results. Finally, if τ is larger less iterations at each resolution should be needed as the changes between each resolution are smaller. This is compensated for by the $\frac{1}{\tau}$ factor.

The downscale and upscale functions are resizing/resampling operations (using a bilinear kernel), H_θ is a hard thresholding operator with threshold θ and constrainToImage clamps all pixel values to the range $[0, 255]$.

Last but not least, the second while loop starting at line 26 attempts to reduce the norm of any remaining residual to below the bound given by ϵ. This is mostly done in this separate loop (without thresholding) for efficiency reasons.

6 Results

Using the images shown in figure 17 the following reconstruction methods are compared:

lsqr Least squares reconstruction using a conjugate gradient method.

lsqr cs The same but using noiselet measurements.

tvqc The l_1-magic "Total Variation with Quadratic Constraints" minimizer (available from `http://www.acm.caltech.edu/l1magic/`).

grthMP Gradient thresholding as described in appendix C, with "model" M and minimizing the l_P norm. The model can be 1, 2 or 3, with 1 using only (a discrete approximation to) the first derivative, 2 using only the second and 3 using a combination of both.

filter Recursive filtering as in [28], but without adding noise and operated in a manner similar to gradient thresholding (gradually using lower denoising strengths). Adding noise is in the original paper [28] was tried but did not seem effective in this setting.

l1qc The l_1-magic "l_1-norm with Quadratic Constraints" minimizer. Using a Haar wavelet transform as sparsifying transform.

twist TwIST (see [5]), an iterative thresholding scheme (using soft thresholding), using a Haar wavelet transform as sparsifying transform and operated in manner similar to the gradient thresholding scheme by starting with $\tau = 1$ and decreasing it by a factor of 1.1 until the error bound is satisfied. Here τ is the regularization weight, (mostly) called λ in the original paper and τ in the implementation.

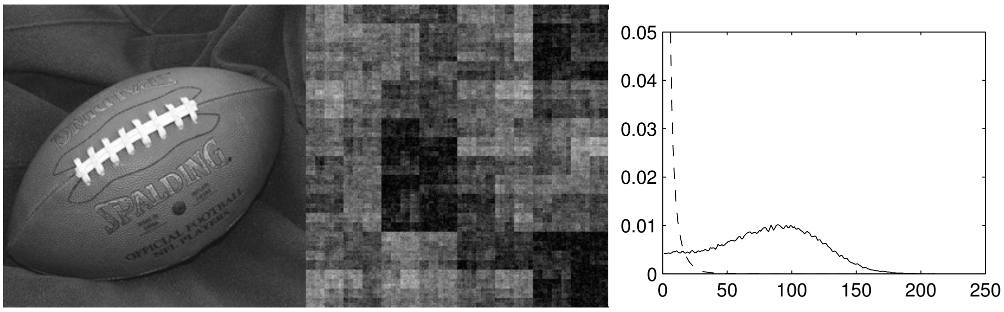
thresholdP Exactly the same scheme as gradient thresholding, except that it uses the Haar wavelet transform as with TwIST, with $p = P$.

hierlsqr Experimental least-squares based method that tries to build up the solution by solving least squares problems on increasing levels of detail. Interesting because it is extremely fast (taking seconds for reconstructions for which gradient thresholding takes minutes).

The methods were tested with the following measurement ensembles:

- Low-pass Discrete Cosine Transform (DCT), with the coefficients selected in a zig-zag order similar to that used in JPEG.
- Noiselets [16] (using the implementation from [37], available at `http://users.ece.gatech.edu/~justin/spmag/`).

The DCT is very closely related to the discrete Fourier transform and tends to concentrate most of the energy of a photograph into coefficients corresponding to low frequencies [44]. This means that it does not really act as a compressed sensing measurement matrix for photographs. Figure 6 demonstrates this, the DCT-based reconstruction clearly favours low-frequency components, yielding a somewhat blurred image.



(a) Original (b) Noiselet coefficients (c) Distributions of magnitude of gradient (dashed line) and noiselet coefficients (solid line)

Figure 5: As can be seen the noiselet transform gives a very noisy result. The non-sparse nature of the noiselet coefficients is also illustrated in the graph on the right. As can be seen the gradient is very sparse (clearly concentrated around zero), in contrast to the noiselet coefficients.

The noiselet ensemble is much better suited for compressed sensing than the DCT for photographs, as it is incoherent with Haar wavelets (see appendix B for a definition of incoherent), so for any signal that has a sparse representation in a Haar wavelet basis (and this is true for most photographs, or in general any piecewise-smooth image) it will act as a compressed sensing measurement matrix. Figure 5 shows the noisy nature of the noiselet coefficients.

A combination of low-pass DCT (5% of the measurements) and noiselets (95%), as in [37], is not used (anymore), as I feel this obfuscates the behaviour of compressed sensing reconstructions and the quality of the reconstructions is virtually identical for most methods and values of k (e.g. for tvqc 92% of the PSNR values are less than 0.5 apart). The only exception is the filter method, as it currently gives extraordinarily bad results (PSNRs below 10 for $k \leq 0.2$) using just noiselets. This is probably due to the initialization as the filter tries to match different blocks in the (extremely noisy) initialization image, which most likely fails to give sensible results. But since it mainly serves to demonstrate the possible advantages of a different model it is simply used with the combined ensemble.

The error metric used throughout all the tests shown below is PSNR, this is defined as:

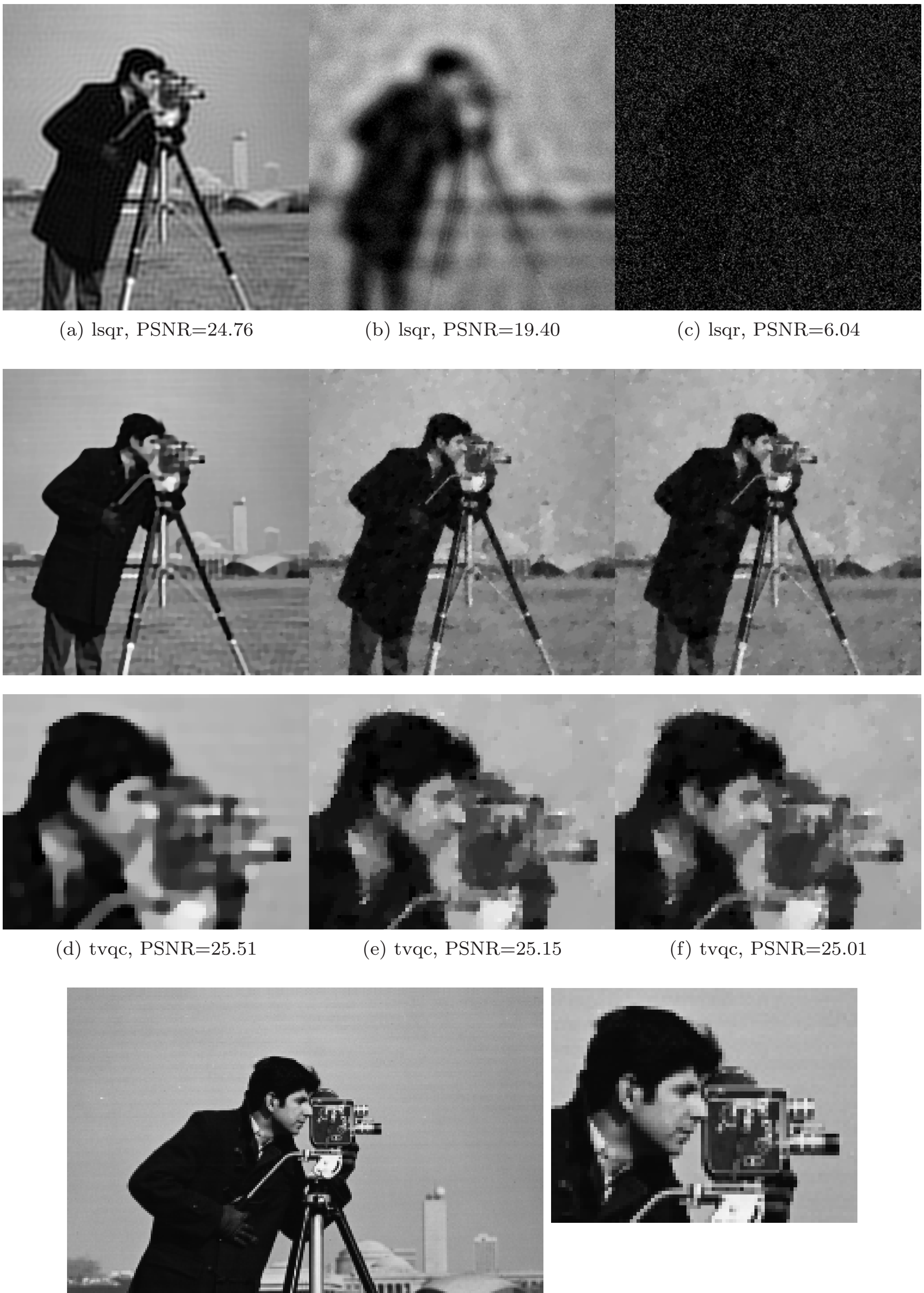
$$\mathrm{PSNR}(f^*) = -20\log_{10}\left(\frac{\|f^* - f\|}{255\sqrt{l}}\right) \tag{6}$$

6.1 Visual comparison

This section shows some of the reconstructions that were made with the above-mentioned methods and compares their visual appearances. In the next section a more numerical comparison is made. To be able to clearly see some of the differences between the different methods the focus is on low quality reconstructions (for higher quality reconstructions the differences between methods can be hard to see, especially on paper).

First consider figure 6, this shows how a simple least squares reconstruction fails with noiselets (fig. 6c), as well as the effect different measurement ensembles can have on the compressed sensing reconstruction. Noiselets are used as a typical example of the measurement matrices used with compressed sensing. As can be seen in figure 6a the DCT data gives a (good) low-pass approximation of the image, while the use of noiselet data increases the error for the least squares reconstruction quite noticeably. For the tvqc reconstruction, on the other hand, the noiselet-based reconstructions (fig. 6e–(f)) are only slightly worse. In fact, while giving a noisier impression, the tvqc reconstructions based on noiselet data are actually able to show some of the smaller details that are lost using only the low-pass DCT data (see the face and camera in figure 6f for instance).

Figure 7 shows the effect of using a (normal, decimated) Haar wavelet transform as a sparsifying transform. For compression purposes it is a reasonable choice (although other wavelet transforms generally perform better) as it can represent an image quite well with a small number of coefficients (as demonstrated in figure 1). For compressed sensing, however, the decimated wavelet transform is not satisfactory due to the "blocky" results as in figure 7c. An undecimated wavelet transform [40]



(a) lsqr, PSNR=24.76

(b) lsqr, PSNR=19.40

(c) lsqr, PSNR=6.04

(d) tvqc, PSNR=25.51

(e) tvqc, PSNR=25.15

(f) tvqc, PSNR=25.01

(g) Original

Figure 6: Reconstructions of cameraman. From left to right: pure (low-pass) DCT, 5% DCT + 95% noiselets and pure noiselets, every reconstruction uses the same number of samples, 10% of the number of pixels in the reconstructed image ($k = 0.1l$). Note that the tvqc reconstruction (d) based on the DCT data has a lot less ringing than the least squares reconstruction (a).

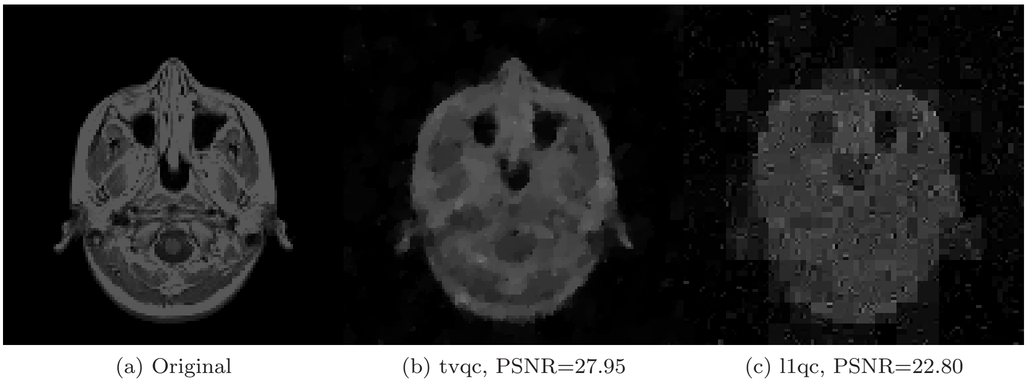

(a) Original (b) tvqc, PSNR=27.95 (c) l1qc, PSNR=22.80

Figure 7: Reconstructions of mri using a pure noiselet ensemble with $k = 0.1l$.

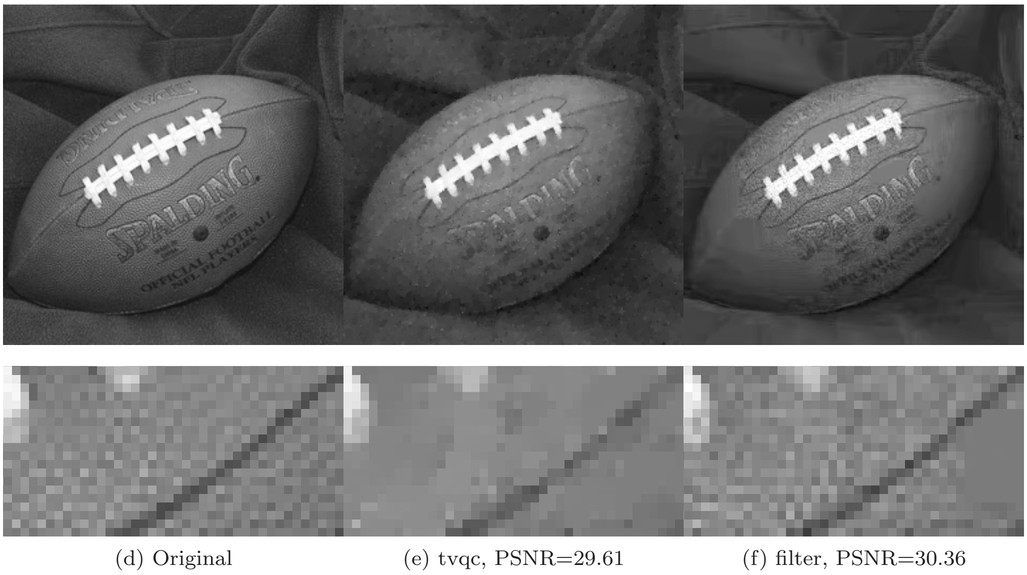
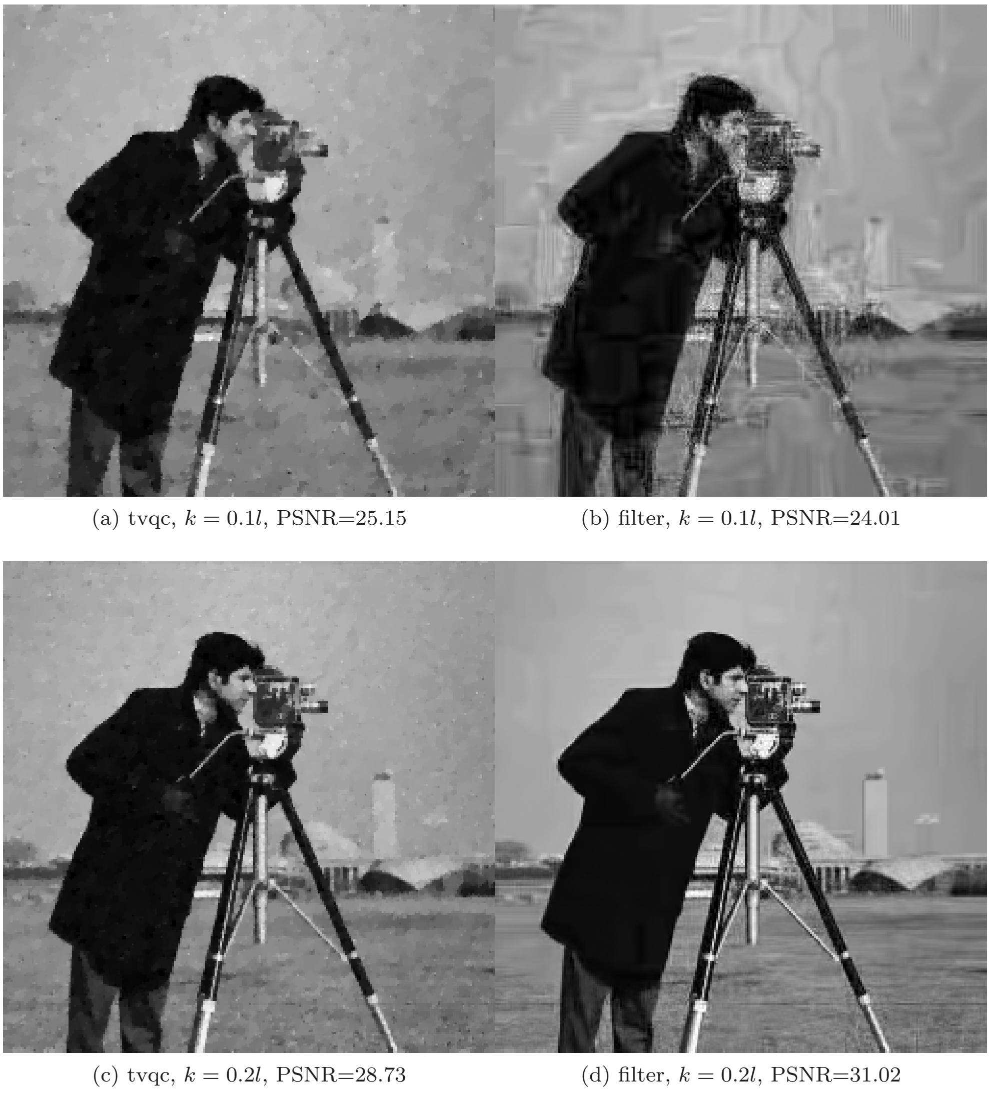

(d) Original (e) tvqc, PSNR=29.61 (f) filter, PSNR=30.36

Figure 8: Reconstructions of football, both using an ensemble with 5% DCT and 95% noiselets, $k = 0.2l$. Note the texture on the football.



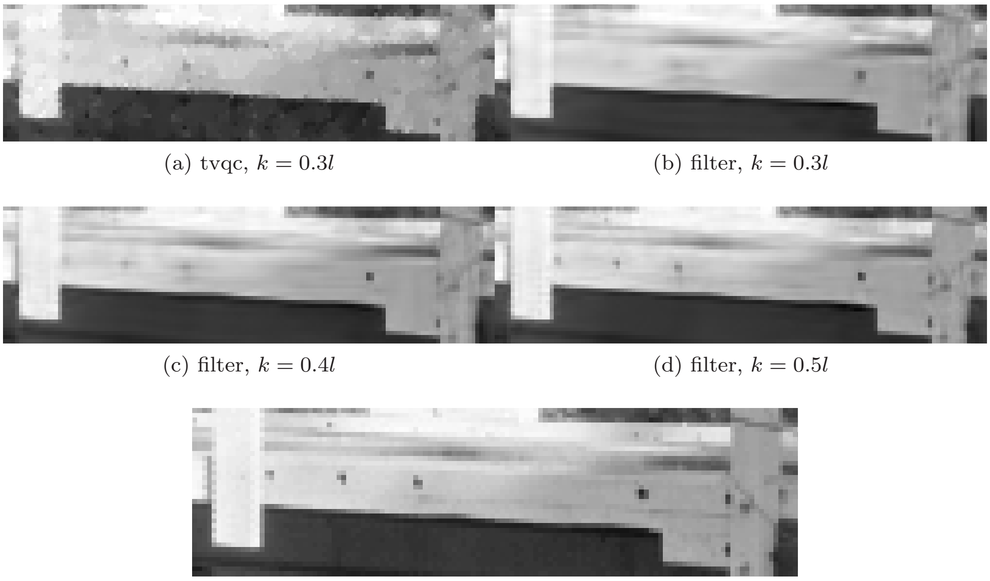
(a) tvqc, $k = 0.1l$, PSNR=25.15

(b) filter, $k = 0.1l$, PSNR=24.01

(c) tvqc, $k = 0.2l$, PSNR=28.73

(d) filter, $k = 0.2l$, PSNR=31.02

Figure 9: Reconstructions of cameraman, all using an ensemble with 5% DCT and 95% noiselets. Note that on the first row tvqc is clearly best and on the second row filter is best by a wide margin.



(a) tvqc, $k = 0.3l$

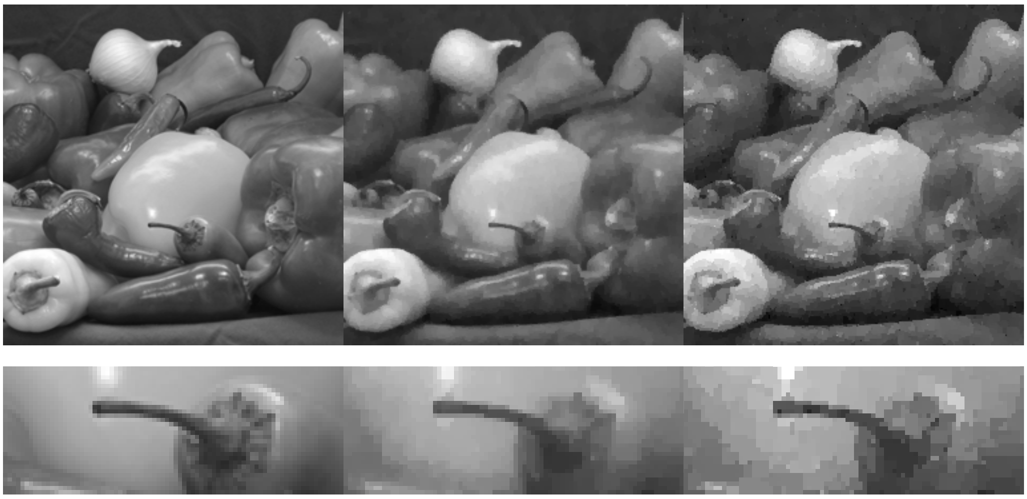
(b) filter, $k = 0.3l$

(c) filter, $k = 0.4l$

(d) filter, $k = 0.5l$

(e) Original

Figure 10: Reconstruction of bridge (just a fragment shown), using an ensemble with 5% DCT and 95% noiselets. Note the three bolts(?) missing in (b).

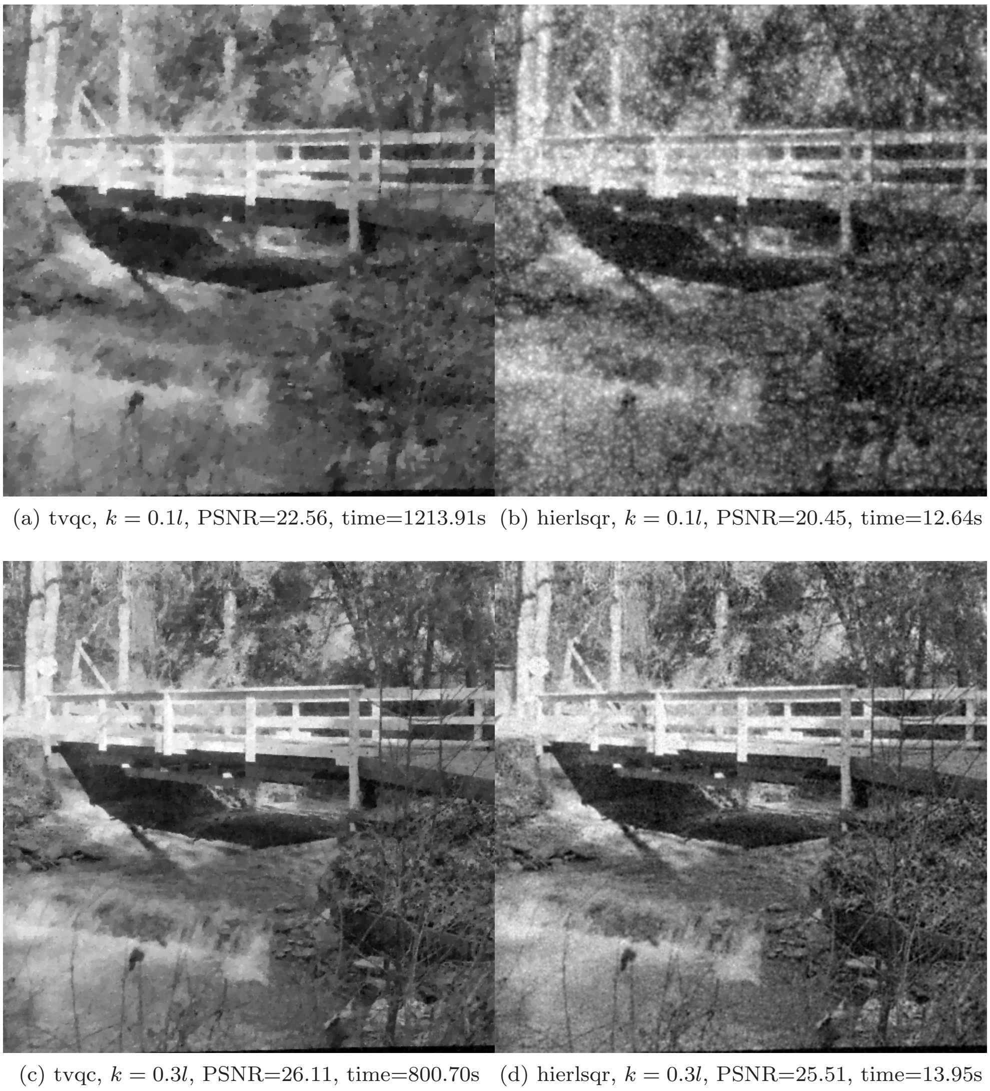


(d) Original

(e) grth11, PSNR=33.07

(f) grth10, PSNR=29.89

Figure 11: Reconstructions of peppers using a pure noiselets ensemble, $k = 0.2l$. The assumption implicit in grth10 that the gradient of the image is (truely) sparse clearly does not hold for this image and results in degraded reconstruction in (f) compared to the grth11 reconstruction (the same is true for grth10 vs. tvqc).



(a) tvqc, $k = 0.1l$, PSNR=22.56, time=1213.91s (b) hierlsqr, $k = 0.1l$, PSNR=20.45, time=12.64s

(c) tvqc, $k = 0.3l$, PSNR=26.11, time=800.70s (d) hierlsqr, $k = 0.3l$, PSNR=25.51, time=13.95s

Figure 12: Reconstructions of bridge using a pure noiselet ensemble.

may give much better results due to translation invariance. It would be interesting to see if this is indeed the case and if so, how much. Especially since total variation is similar to using only one level of an undecimated wavelet transform [41].

The effects of using the more advanced model implicit in the filter method are shown in figures 8 and 9. As is apparent from the images the model can lead to much improved reconstruction accuracy, especially of textures, but can also fail spectacularly, as in figure 9b where a lot of high-frequency noise is introduced. This may be due to the (noisy) initialization, but may also be due to "phase transition" effects similar to those observed in figure 3.

A potentially more serious effect of using a more advanced filter is shown in figure 10. Here three bolts(?) are not clearly visible (fig. 10b) in the filter reconstruction upto approximately $k = 0.4l$, whereas in the tvqc reconstruction they are merely made a little less pronounced by noise (fig. 10a). *The advanced filter can obscure relatively large image features compared to the simple gradient thresholding filter.* Such behaviour might be extremely problematic in medical and industrial applications.

Figure 11 demonstrates what can happen when the sparsity assumption does not hold. Here the mismatch is only minor, the image gradient is compressible but definitely not sparse, so the effect is reasonably benign. However, it does demonstrate a problem inherent in all methods that try to reconstruct truely sparse solutions (such as OMP), they clearly fail on signals that are not sparse. On the other hand, they do (in principle) allow better reconstruction of signals that *are* sparse, as demonstrated by the results for the (relatively sparse) mri image shown in figure 15.

As an indication of what can be achieved with hierlsqr figure 12 shows the results of reconstructing bridge using different values for k. As can be seen hierlsqr produces a completely different kind of error than tvqc. Apart from hierlsqr giving a larger error (on most images) for the same value of k it is not immediately clear which kind of error is to be preferred.

At `http://www.cg.tuwien.ac.at/~jasper/imtest.html` these images (and many more) can be compared on-line. The site allows specific sets of images to be selected based on several attributes, to make comparisons easier. Note that the site should work with any browser (except Internet Explorer).

6.2 Numerical comparison

As the least squares reconstruction consistently works much better with a low-pass DCT ensemble, I show both results for the least squares reconstruction and just use noiselets for the rest in figures 13, 14 and 15. In addition I show the PSNR values for a "best-k" reconstruction using a DCT basis and a Haar wavelet basis. These reconstructions were made by performing the respective transform and setting all coefficients except the k largest to zero before doing the inverse transform.

Some observations related to the quality of the reconstructions:

- Gradient thresholding can be configured to give similar reconstruction error rates to the l_1 magic total variation minimizer, while it can be (in some cases considerably) faster, see figure 14. It should be noted, though, that for relatively high k the difference in performance diminishes. Intuitively this can be understood by considering that tvqc tries to refine the solution (starting with an initial estimate which is increasingly good for larger values of k), while gradient thresholding tries to build up the solution (starting with a very smooth image that is a poor fit for the constraints).

- By varying different parameters of gradient thresholding it can yield better quality reconstructions than tvqc for certain images (like mri). See figure 15.

- Although many texts stress that l_0-norm minimization is the "ideal", and that l_1-norm minimization is merely a way to make the problem more tractable, this is only true for truely sparse signals. Minimizing the l_0-norm for signals that are not sparse can actually result in a lower quality reconstruction (see figures 11 and 15).

- TwIST is faster than ordinary soft thresholding in the Haar wavelet domain.

- Total variation reconstruction and gradient thresholding (and to some extent the l1qc-related reconstructions) allow a reconstruction of the original data that is comparable to or better than a linear reconstruction based on low-pass DCT data.

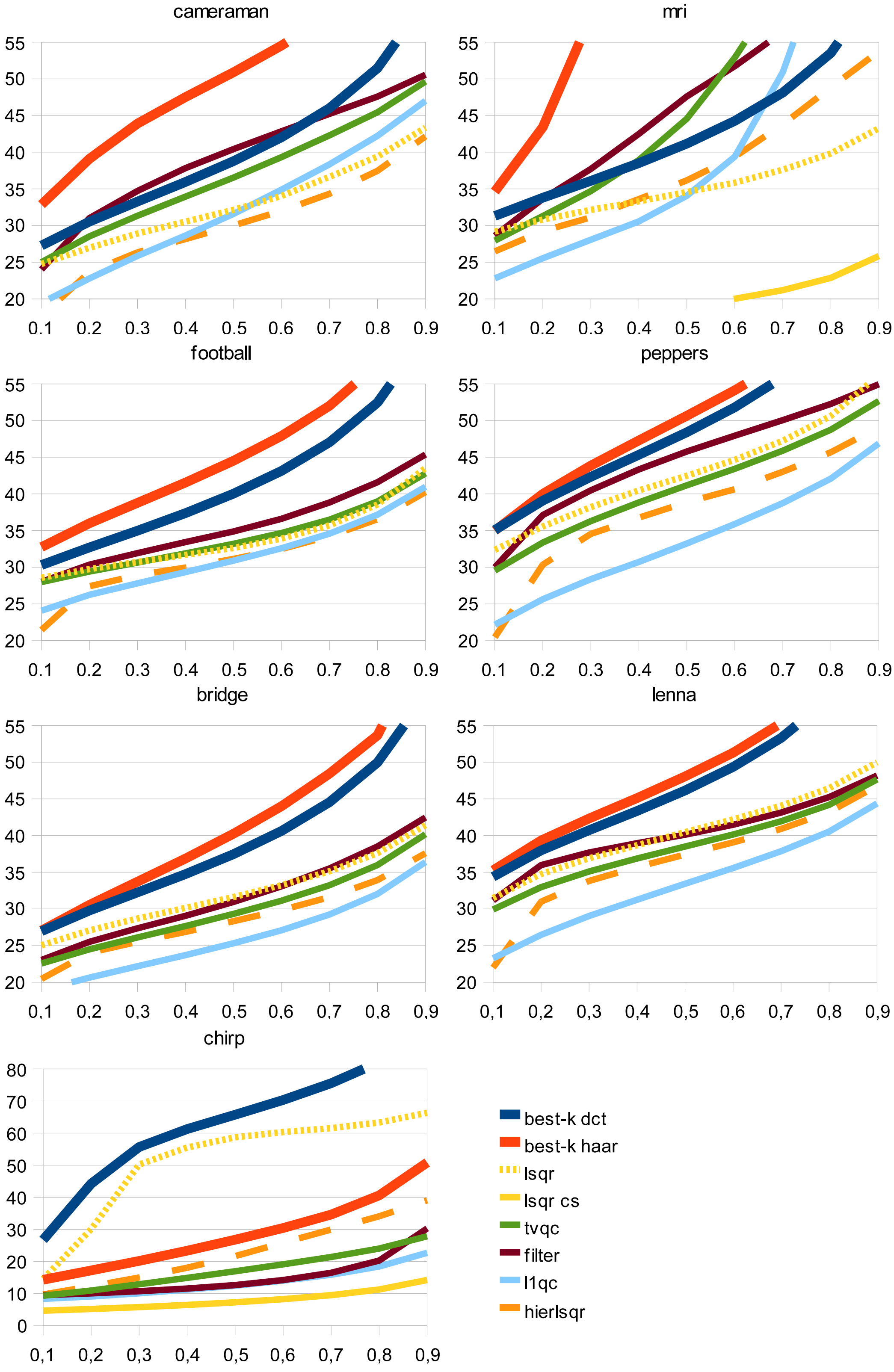


Figure 13: Graphs of the PSNR values for different values of k as a fraction of l. Note that the PSNR values for the compressed reconstructions are bounded from above, due to the use of a non-zero ϵ for the reconstruction, and that "lsqr cs" is not missing, its PSNR values are just very low. See figure 18 for RMS versions of these graphs.

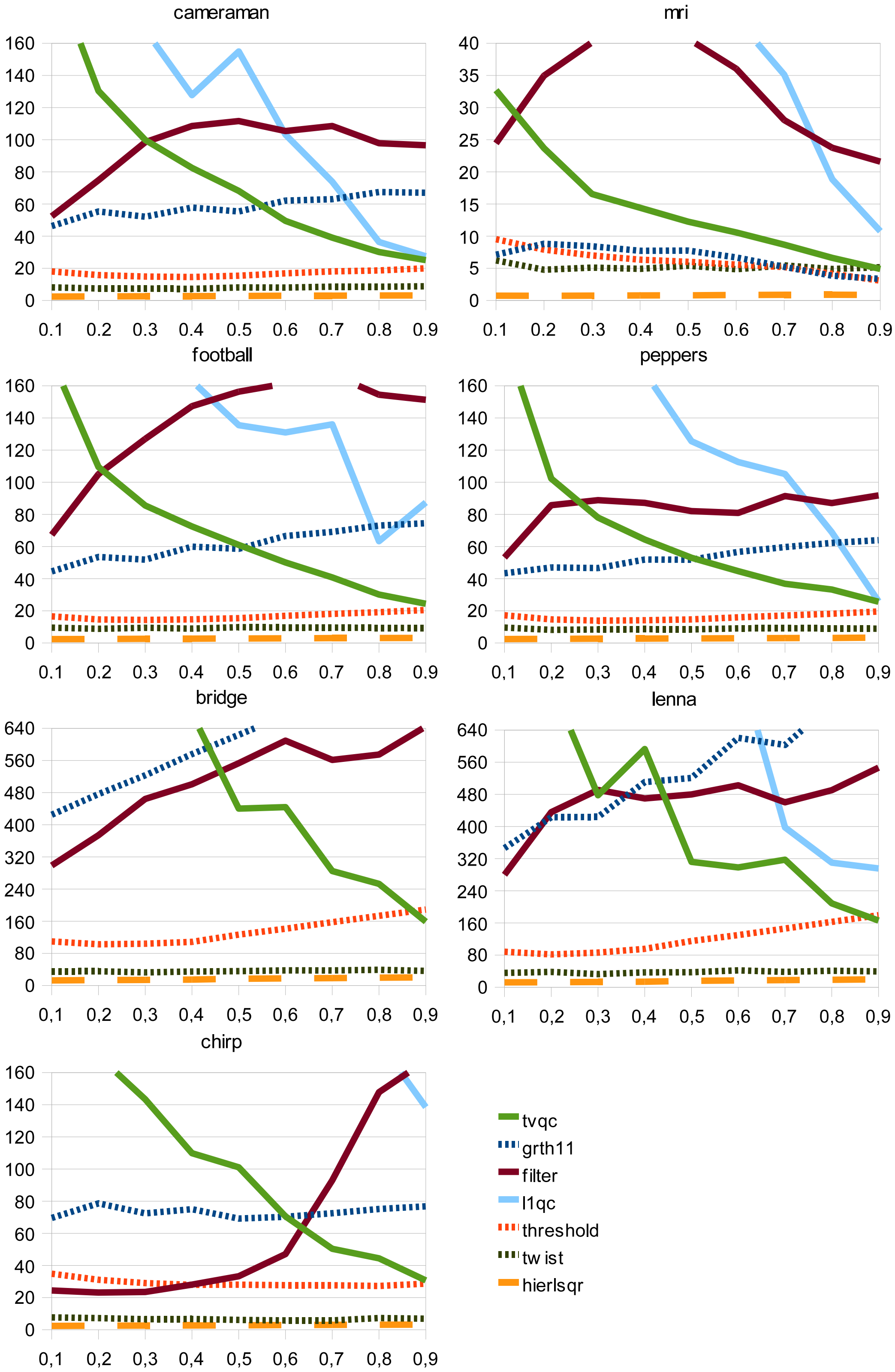


Figure 14: Graphs of the time (in seconds) needed for one reconstruction for different values of k as a fraction of l. Note that although the transforms are implemented using “fast” operators, in all cases the full transform is performed, only selecting a subset of the coefficients afterwards.

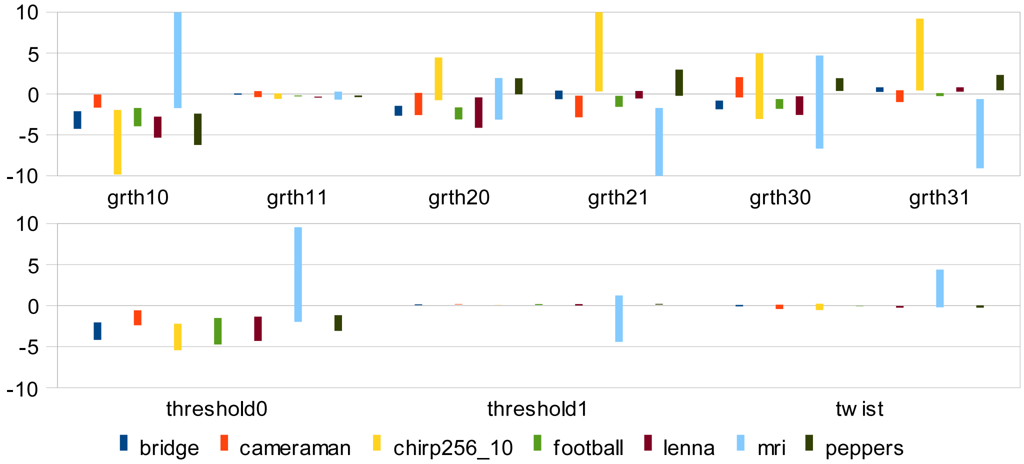


Figure 15: Difference in PSNR compared to tvqc for grth and compared to l1qc for threshold and twist, using only the (discrete) first derivative, only the second derivative and both. And also grth with soft thresholding (using only the first derivative). The bars show the range of the difference in PSNR (so the top corresponds to the maximum difference and the bottom to the minimum difference). Higher is better. grth11 (soft thresholding and using only the first derivative) is clearly quite similar in performance to tvqc.

- The best-k reconstructions are of higher quality, as would be expected. But mri and peppers show that the compressed sensing methods can take advantage of choosing a good representation aposteriori, which means existing signals (taken using "noisy" measurements) might be reconstructed better in the future as better models become known. Although the choice of measurement matrix (in this case noiselets) is of course also dependent on the signal model.

- Comparing the quality of the filter method and tvqc also shows that **a good model** [§**A.4**] **of the image is crucial** for the quality of a compressed sensing reconstruction.

Considering that the DCT should give a reasonably sparse (compressible) approximation of the image (that's why it's used in JPEG compression after all), with coefficients concentrated towards the DC component, it is quite nice that these methods are able to match (and even exceed) the quality of a least squares low-pass DCT reconstruction using noiselet measurements. Especially if you consider that a least squares reconstruction does very poorly on noiselet measurements (the PSNR is less than 10 for the noiselets ensemble and for the combination ensemble the PSNR is less than for the $k/l = 0.1$ pure DCT reconstruction).

Note that the results for TwIST are actually better (3-4 PSNR points) for the low-pass DCT ensemble than for the noiselet ensemble (for $k/l = 0.1, 0.3$ at least, for large values of k the difference is reversed), but they are still worse than a simple least squares reconstruction based on the DCT data. The results shown in figure 13 are better than those for the least squares reconstruction using the noiselet ensemble though. So applying TwIST with a Haar wavelet transform is not a very good method, unless speed is an absolute necessity.

Finally, during testing the l_1-magic tvqc minimizer demonstrated a *very* high sensitivity to the measurement ensemble used. Minimization using a pure noiselets ensemble is upto 10 times faster than minimization using a pure DCT ensemble or even upto 20-30 times faster than minimization using a combination of 5% DCT and 95% noiselets. The quality of the reconstruction is almost identical (in the latter case, for which 55 out of 63 cases have a PSNR difference of less than 0.3). Also note that this speedup is not seen with grth11, with the noiselet ensemble it is on average just as fast as with the pure DCT ensemble and only upto about a factor of 2 faster than with the combination ensemble (which is quite natural, as twice as many transforms are performed).

7 Conclusion

Compressed sensing is about being able to reconstruct a signal from a relatively small number of measurements, given the right kind of measurements. From the results above it is clear that this is indeed possible in practice, despite the lack of very concrete theoretical underpinnings. Already quite a few measurement processes and application areas have been defined, and low error reconstructions, using significant “undersampling” factors, are often obtained (shown above for greyscale, natural, images).

For the quality of the reconstructions it is clear that, apart from the measurement matrix, the model (implicitly) used for the reconstruction is of the utmost importance. This gives some room for trade-offs between speed and quality, as well as improved reconstructions as models get better. The latter also immediately suggests that it may often be a good idea to not just store the reconstructed signal, but keep the original measurements around.

Currently, the main drawback of compressed sensing reconstruction methods is their performance, as testified by the great number of methods avaiable (practically all emphasizing speed!). As shown in the results there is quite a lot of variance in their performance, not always directly correlated with the quality of the reconstructions.

Of the reconstruction methods considered here, the family of methods based on iterative thresholding (including gradient thresholding) currently seems to have the best credentials. These methods are among the fastest methods currently available, quite flexible and easy to use, while giving results comparable to, or even better than, more traditional methods. For extremely large (3D) data sets a multiscale approach like hierarchical least squares might be better suited though (depending on the time available for reconstruction and the desired quality of course).

8 Future work

There are still many open problems in compressed sensing, the main one obviously being the lack of any straightforward theoretical bounds on the quality of reconstructions, as can so easily be given for a best-k reconstruction. And this is of course followed directly by the continuing quest for fast reconstruction algorithms.

Intertwined with these two important endeavours is the choice of model. It is shown in this report (appendix A), as well as in e.g. [3], that the choice of model has a definite effect on the reconstruction method that must be used. And at the same time the choice of model can have a profound effect on the quality of the reconstruction. Therefore, finding a (class of) model(s) that fits real-world signals while allowing a fast reconstruction method could be very rewarding.

When considering to evaluate models for real-world signals another gap in the current theoretical framework is immediately apparent. How can one quantify how well a model fits to a (class of) signals? Apart from appealing to one’s intuition and the link between the penalty function used and probability density functions described in section A.3, very little seems to have been done in this area. Solving this problem would make it possible to define what the *best* model would be in a certain context (instead of using an educated guess), and perhaps give a deeper understanding of the performance of compressed sensing.

Also, any current theory on compressed sensing assumes that the measurements are fully incoherent with the data. However, for the widely used Fourier transform this assumption already often does not hold completely. This means that it does become interesting which samples are taken. Usually this is ignored or an ad-hoc solution is presented. It would be very interesting to be able to give optimal sampling positions, given an approximate distribution of the measurement transform coefficients. Or if we assume “good” sampling positions (for example adaptive), it would be interesting to evaluate the performance of compressed sensing reconstruction methods, both in theory and practice. This would have immediate applications to (de)compression (and possibly resampling).

Similarly, it may be interesting to consider the performance of compressed sensing reconstruction methods when appliead to a mixture of adaptive measurements using a good basis and (non-adaptive) measurements using an incoherent basis. For example, traditionally an image might be compressed by taking the (Haar) wavelet transform and storing only the k largest coefficients, would it be advantageous to also use some noiselet measurements? It could be expected to be advantageous as noiselets are in a sense complementary to Haar wavelets and might provide additional information that can otherwise only be given using many more wavelet coefficients.

In a more concrete setting, based on the results shown in this report it would be interesting to use TwIST with a gradient thresholding scheme (or the TV minimizer that is used in [5]) and see if it is also faster in such a set up. And of course any theoretical validation of hierarchical least squares would be great, both to inspire confidence in the method and to identify ways of improving the current algorithm.

One way of improving hierarchical least squares might lie in different ways of representing the different scales. For instance, I have experimented with using anisotropic diffusion instead of thresholding and resizing. This seemed to give a better quality result but was much slower, at least using my implementation. However, as anisotropic diffusion is similar to applying a bilateral filter [4], this could link hierarchical least squares to the multiscale method presented in [47], which uses a bilateral filter on each scale.

8.1 Computed Tomography

Already there is great interest in using compressed sensing methods for CT reconstructions. Below is a list of papers that deal with reconstruction of CT scan (not limited to X-ray CT) data using compressed sensing methods:

1. Sparseness prior based iterative image reconstruction for retrospectively gated cardiac micro-CT [39]: an application of compressed sensing to (3D) cone beam CT, using total variation regularization with a non-linear conjugate gradient solver.

2. Sparse MRI: The Application of compressed sensing for Rapid MR Imaging [33]: suitable sparsifying transforms and sampling trajectories are discussed, and three different undersampling schemes are considered (two in 2D and one in 3D).

3. Highly Undersampled Magnetic Resonance Image Reconstruction via Homotopic L0-Minimization [46]: introduces a homotopic method that tries to minimize the l_0-norm instead of the l_1-norm. In [47] it is shown how such a method can be implemented using a (fast) multiscale method.

4. Improved k-t BLAST and k-t SENSE using FOCUSS [30]: shows how a newly developed method (which can be linked to compressed sensing) can be seen as a generalization of two "celebrated" methods, and presents results obtained with this new method.

5. k-t FOCUSS: A General Compressed Sensing Framework for High Resolution Dynamic MRI [29]: extends k-t FOCUSS to explicitly incorporate prediction and residual coding.

6. Improving Non-Contrast-Enhanced Steady-State Free Precession Angiography with Compressed Sensing [18]: minimization of an l_1-norm is used to improve image quality and/or reduce acquisition time.

The existing methods either try to take advantage of the sparsity in the scanned data directly, or try to use a sparsity inducing transform to increase its sparsity, yielding to improved reconstruction quality. Usually the aim is to minimize the number of required projections, which can lead to a reduced radiation dose and/or shorter acquisition time.

8.1.1 Difference reconstruction

One way of increasing the sparsity of the image to be reconstructed is to use a reference image. If you have a reasonably good model of what you want to scan, then the difference between the model and the actual data will in most cases be much sparser than the actual data itself. This technique is exploited in [14] for the purpose of background subtraction in videos, but may also be interesting in industrial CT when scanning for defects for example.

By exploiting the sparsity of the difference image it is expected that less projections would be necessary for a good reconstruction. This is because the quality of the reconstruction using compressed sensing is directly related to the sparsity of the signal to reconstruct. In addition it might be possible to reconstruct to a higher resolution than the resolution of the individual slices if a high resolution model is available.

To be able to use this concept it is only necessary to measure the scanned data as usual and to have the measurements corresponding to the model. Assuming a linear measurement process the two sets of measurements can simply be subtracted to yield measurements corresponding to the difference image.

An important issue will probably be coping with differences in position and orientation between the model and the scanned object. A given 3D registration algorithm could simply be applied once, at the beginning, or after each iteration of a reconstruction process, depending on how accurate the estimate turns out to be. Applying a registration algorithm after each iteration will probably be costly, but perhaps the registration algorithm used would be able to benefit from a warm start.

8.1.2 D^{2}VR

In [36] a variation on backprojection is presented which directly yields a new projection based on the measured projections. By eliminating an intermediate representation and the related interpolation this method can achieve a better reconstruction quality.

Compressed sensing could potentially be applied in two ways. You may want to start with an undersampled set of projections and reconstruct a full set of projections, or you may want to try to improve the quality of the reconstruction directly. For the latter it is not clear how this would be done without an intermediate representation though.

When going from a small number of projections to a full set of projections the main problem that could show up is that there will probably be way too much correlation between the measurements (the small number of projections) and the data to reconstruct (the full set of projections). In addition a suitable sparsity inducing transform should be found, just applying a wavelet transform or something similar per projection will most likely not promote as much sparsity as a 3D wavelet transform on the equivalent volume data.

To see how D^{2}VR can be reduced to computing a sum of transformed (filtered) projections, amenable to a fast implementation (on a GPU for example), assume we have (filtered) projection data representing a volume. A projection i is taken along lines parallel to D_i and its centerline is described by $tD_i + C_i$ ($t \in \mathbb{R}$). For simplicity we define both the (unit-length) up and right vectors for a projection, U_i and R_i (U_i, R_i and D_i should be orthogonal). So any point $p \in \mathbb{R}^3$ projects onto $T_i(p - C_i)$, where $T_i = \begin{bmatrix} R_i & U_i \end{bmatrix}^T$. For the moment the projections have infinite width and height. A line $td + c$ projects onto projection i as:

$$\begin{aligned} td_i + c_i &= T_i(td + c - C_i) \\ &= tT_i d + T_i(c - C_i) \end{aligned}$$

Thus ($a_i(x)$ and $b_i(x)$ give bounds on the volume considered):

$$\begin{aligned} P_i(x) &= \int_{a_i(x)}^{b_i(x)} f(tD_i + C_i + T_i^T x)\,dt \\ &= \int_{a_i(x)}^{b_i(x)} \sum_j Q_j(T_j(tD_i + C_i - C_j + T_i^T x))\,dt \\ &= \sum_j \int_{a_i(x)}^{b_i(x)} Q_j(tT_j D_i + T_j(C_i - C_j) + T_j T_i^T x)\,dt \\ &= \sum_j \int_{a_i(x)}^{b_i(x)} Q_j(tT_j T_i^T (T_j T_i^T)^{-1} T_j D_i + T_j(C_i - C_j) + T_j T_i^T x)\,dt \\ &= \sum_j \int_{a_i(x)}^{b_i(x)} Q_j(T_j(C_i - C_j) + T_j T_i^T (x + t(T_j T_i^T)^{-1} T_j D_i))\,dt \end{aligned}$$

Note that for $i = j$ the integral reduces to $(b_i(x) - a_i(x))Q_i(x)$.

Looking just at the integral it can be simplified as follows, showing that each projection is a sum of the affinely transformed, smeared, filtered projections:

$$\begin{aligned} Q_{ij}^S(x, \chi) &= \int_{a_i(x)}^{b_i(x)} Q_j(tT_j D_i + \chi)\,dt \\ Q_{ij}^A(x) &= Q_{ij}^S(x, T_j(C_i - C_j) + T_j T_i^T x) \\ P_i(x) &= \sum_i Q_{ij}^A(x) \end{aligned}$$

Or alternatively:

$$\begin{aligned} Q^A_{ij}(x) &= Q_j(T_j(C_i - C_j) + T_j T_i^T x) \\ Q^S_{ij}(x) &= \int_{a_i(x)}^{b_i(x)} Q^A_j(x + t(T_j T_i^T)^{-1} T_j D_i)\, dt \\ P_i(x) &= \sum_i Q^S_{ij}(x) \end{aligned}$$

When implementing this the question of what bounding volume to take should be addressed, too large a bounding volume will probably not give satisfactory results, but the bounding volume must obviously be at least as large as the object to reconstruct.

8.2 GPU-related

Due to the very computationally demanding nature of compressed sensing it can be interesting to implement (parts of) a reconstruction method on the GPU. Some of the papers that deal with either CT reconstructions or compressed sensing reconstruction methods on the GPU (or similar architectures):

1. Fast GPU Implementation of Sparse Signal Recovery from Random Projections [2]: an implementation of Matching Pursuit on the GPU.

2. A Simple Compressive Sensing Algorithm for Parallel Many-Core Architectures [7]: an algorithm that is superficially similar to iterative thresholding (embedded in a continuation scheme, not unlike gradient thresholding) suitable for parallel implementation.

3. How GPUs Can Improve the Quality of Magnetic Resonance Imaging [43]: describes how a reconstruction method that can operate directly on non-Cartesian data can be implemented on the GPU.

4. Accelerating advanced MRI reconstructions on GPUs [42]: describes an implementation of an MRI reconstruction method on the GPU that allows arbitrary sampling trajectories and can incorporate prior anatomical knowledge.

5. Hyperfast parallel-beam and cone-beam backprojection using the cell general purpose hardware [31]: shows how backprojection can be sped up using the Cell Broadband Engine (not a GPU, but similar in architecture).

In the interest of speeding up compressed sensing reconstructions it would no doubt be interesting to implement a(nother) thresholding algorithm on the GPU and compare it to other methods on the GPU and CPU. As the thresholding operation can be done per-pixel, this kind of algorithm naturally lends itself to a parallel implementation, provided that multiplication by the measurement matrix can also be performed efficiently.

9 Acknowledgements

The work presented in this publication is partly supported by the Austrian Science Fund (FWF) grant no. P18547. The first author would like to thank Jos Roerdink and Eduard Gröller (of the University of Groningen and the Vienna University of Technology, respectively) for making possible the internship during which this report was written.

References

[1] *Proceedings of the International Conference on Image Processing, ICIP 2007, September 16-19, 2007, San Antonio, Texas, USA*. IEEE, 2007.

[2] M. Andrecut. Fast gpu implementation of sparse signal recovery from random projections, 2008.

[3] Richard G. Baraniuk, Volkan Cevher, Marco F. Duarte, and Chinmay Hegde. Model-based compressive sensing. *CoRR*, abs/0808.3572, 2008.

[4] Danny Barash. A fundamental relationship between bilateral filtering, adaptive smoothing, and the nonlinear diffusion equation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(6):844–847, 2002.

[5] José M. Bioucas-Dias and Mário A. T. Figueiredo. A new twist: Two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image Processing*, 16(12):2992–3004, 2007.

[6] Thomas Blumensath and Mike E. Davies. Iterative thresholding for sparse approximations. To be published in: The Journal of Fourier Analysis and Applications.

[7] Alexandre Borghi, Jérôme Darbon, Sylvain Peyronnet, Tony F. Chan, and Stanley Osher. A simple compressive sensing algorithm for parallel many-core architectures. Technical report, UCLA Computational and Applied Mathematics, September 2008.

[8] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[9] E. J. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, 2006.

[10] Emmanuel Candès and Justin Romberg. Sparsity and incoherence in compressive sampling. *Inverse Problems*, 23(3):969–985, 2007.

[11] Emmanuel Candes and Terence Tao. Decoding by linear programming, 2005.

[12] Emmanuel J. Candès. Compressive sampling. In *International Congress of Mathematicians. Vol. III*, pages 1433–1452. Eur. Math. Soc., Zürich, 2006.

[13] Emmanuel J. Candès. The restricted isometry property and its implications for compressed sensing. *C. R. Math. Acad. Sci. Paris*, 346(9-10):589–592, 2008.

[14] Volkan Cevher, Aswin C. Sankaranarayanan, Marco F. Duarte, Dikpal Reddy, Richard G. Baraniuk, and Rama Chellappa. Compressive sensing for background subtraction. In David A. Forsyth, Philip H. S. Torr, and Andrew Zisserman, editors, *ECCV (2)*, volume 5303 of *Lecture Notes in Computer Science*, pages 155–168. Springer, 2008.

[15] Rick Chartrand and Wotao Yin. Iteratively reweighted algorithms for compressive sensing. *The 2008 IEEE International Conference on Acoustics, Speech, and Signal Processing,* Las Vegas, Nevada, USA, April 2008.

[16] R. Coifman, F. Geshwind, and Y. Meyer. Noiselets. *Applied and Computational Harmonic Analysis*, 10(1):27 − 44, 2001.

[17] Patrick L. Combettes and Valérie R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[18] T. Cukur, M. Lustig, and D.G. Nishimura. Improving non-contrast-enhanced steady-state free precession angiography with compressed sensing. To be published in: Magnetic Resonance in Medicine, 2008.

[19] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004.

[20] Ingrid Daubechies, Ronald DeVore, Massimo Fornasier, and C. Sinan Gunturk. Iteratively reweighted least squares minimization for sparse recovery, 2008.

[21] S. Dekel. Adaptive compressed image sensing based on wavelet-trees. Preprint, 2008.

[22] David L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52(4):1289–1306, 2006.

[23] David L. Donoho and Jared Tanner. Thresholds for the recovery of sparse solutions via l1 minimization. In *Proceedings of the IEEE Conference on Information Sciences and Systems*, March 2006.

[24] David L. Donoho and Jared Tanner. Phase transition phenomena in sparse approximation, 2008.

[25] David L. Donoho and Yaakov Tsaig. Fast solution of l1-norm minimization problems when the solution may be sparse, October 2006.

[26] David L. Donoho and Yaakov Tsaig. Sparse solution of underdetermined linear equations. by stagewise orthogonal matching pursuit, March 2006.

[27] Marco Duarte, Mark Davenport, Dharmpal Takhar, Jason Laska, Ting Sun, Kevin Kelly, and Richard Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):83–91, March 2008.

[28] Karen O. Egiazarian, Alessandro Foi, and Vladimir Katkovnik. Compressed sensing image reconstruction via recursive spatially adaptive filtering. In *ICIP* [1], pages 549–552.

[29] H. Jung, K. H. Sung, K. S. Nayak, E. Y. Kim, and J. C. Ye. k-t focuss: A general compressed sensing framework for high resolution dynamic mri. To be published in: Magnetic Resonance in Medicine, 2008.

[30] Hong Jung, Jong Chul Ye, and Eung Yeop Kim. Improved k-t blast and k-t sense using focuss. *Physics in Medicine and Biology*, pages 3201–3226, 2007.

[31] Marc Kachelrieß, Michael Knaup, and Olivier Bockenbach. Hyperfast parallel-beam and cone-beam backprojection using the cell general purpose hardware. *Medical Physics*, 34(4):1474–1486, 2007.

[32] Anat Levin, Rob Fergus, Frédo Durand, and William T. Freeman. Image and depth from a conventional camera with a coded aperture. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 70, New York, NY, USA, 2007. ACM.

[33] Michael Lustig, David Donoho, and John M. Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine*, 9999(9999):NA+, 2007.

[34] Ray Maleh, Anna C. Gilbert, and Martin J. Strauss. Sparse gradient image reconstruction done faster. In *ICIP* [1], pages 77–80.

[35] Roummel F. Marcia and Rebecca M. Willett. Compressive coded aperture video reconstruction. In *Proceedings of the 16th European Signal Processing Conference, EUSIPCO 2008*, Lausanne, Switzerland, August 2008.

[36] Peter Rautek. D^{2}VR high-quality volume rendering of projection-based volumetric data. Master's thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, 2005.

[37] Justin Romberg. Imaging via compressive sampling [introduction to compressive sampling and recovery via convex programming]. *IEEE Signal Processing Magazine*, 25(2):14–20, March 2008.

[38] Rayan Saab, Rick Chartrand, and Özgür Yilmaz. Stable sparse approximations via nonconvex optimization. In *33rd International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2008.

[39] J. Song, Q. H. Liu, G. A. Johnson, and C. T. Badea. Sparseness prior based iterative image reconstruction for retrospectively gated cardiac micro-ct, November 2007.

[40] Jean-Luc Starck, Jalal Fadili, and Fionn Murtagh. The undecimated wavelet decomposition and its reconstruction. *IEEE Transactions on Image Processing*, 16(2):297–309, 2007.

[41] G. Steidl, J. Weickert, T. Brox, P. Mrázek, and M. Welk. On the equivalence of soft wavelet shrinkage, total variation diffusion, total variation regularization, and SIDEs. *SIAM Journal on Numerical Analysis*, 42(2):686–713, May 2004.

[42] Sam Stone, Justin Haldar, Stephanie C. Tsao, Wen mei Hwu, Zhi-Pei Liang, and Bradley P. Sutton. Accelerating advanced mri reconstructions on gpus. In *Proceedings of 5th International Conference on Computing Frontiers*. ACM, April 2008.

[43] Sam Stone, Haoran Yi, Wen-mei Hwu, Justin Haldar, Bradley Sutton, and Zhi-Pei Liang. How gpus can improve the quality of magnetic resonance imaging. The First Workshop on General Purpose Processing on Graphics Processing Units (GPGPU), October 2007.

[44] Antonio Torralba and Aude Oliva. Statistics of natural image categories. In *Network: Computation in Neural Systems*, pages 391–412, 2003.

[45] Joel A. Tropp, Anna, and C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Inform. Theory*, December 2007.

[46] Joshua Trzasko and Armando Manduca. Highly undersampled magnetic resonance image reconstruction via homotopic l0-minimization, 2007.

[47] Joshua Trzasko, Armando Manduca, and Eric Borisch. Sparse mri reconstruction via multiscale l0-continuation. *Statistical Signal Processing, 2007. SSP '07. IEEE/SP 14th Workshop on*, pages 176–180, Aug. 2007.

[48] Yaakov Tsaig and David L. Donoho. Extensions of compressed sensing. *Signal Process.*, 86(3):549–571, 2006.

[49] M. Yaghoobi, T. Blumensath, and M. Davies. Quantized sparse approximation with iterative thresholding for audio coding. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1:I–257–I–260, April 2007.

[50] Yin Zhang. On theory of compressive sensing via l_1-minimization: Simple derivations and extensions. Technical Report 08-11, CAAM, Rice University, Houston, July 2008.

A Alternative formulations

A.1 A priori formulation

This formulation requires knowledge of the sparsity basis beforehand:

$$\begin{aligned} \text{minimize}(\tilde{f}) \quad & \|\Psi^T \tilde{f}\|_1 \\ \text{subject to} \quad & \Phi\Psi^T \tilde{f} = y \end{aligned} \tag{7}$$

It is assumed that Φ has properties CS1-3 from [22] and that Ψ is orthogonal (or a tight frame), see [22] and [48]. This could also be expressed as:

$$\begin{aligned} \text{minimize}(\tilde{x}) \quad & \|\tilde{x}\|_1 \\ \text{subject to} \quad & \Phi\tilde{x} = y \end{aligned}, \tilde{f} = \Psi\tilde{x}$$

A.2 Filtering formulation

A more general formulation similar to the a posteriori formulation (adapted from [28]) which may be given as the following fixed point equation:

$$\tilde{z} = \bar{\Phi}\Xi(\begin{bmatrix}\Phi \\ \bar{\Phi}\end{bmatrix}^{-1}\begin{bmatrix}y \\ \tilde{z}\end{bmatrix}) \tag{8}$$

The original paper uses a (probably not essentially) more specific formulation (Φ is a subset of the rows of a Fourier transform and $\bar{\Phi}$ is the rest). The following additional definitions are used in this formulation:

- Ξ is a (non-linear) filter that "smooths"/regularizes the data (for example: $\Xi(a) = \Psi S \Psi^{-1} a$ where S is a shrinkage operator)
- $\bar{\Phi}$ is a basis for the null-space of Φ (such that $\text{rank}(\begin{bmatrix}\Phi \\ \bar{\Phi}\end{bmatrix}) = l$)
- $z = \bar{\Phi} f$

It is likely that this formulation is a generalization of the aposteriori formulation, as the algorithms are nearly equivalent (given the right filter). See appendix D for a description of the algorithm corresponding to the filtering formulation.

A.3 Probability density functions

A compressed sensing reconstruction tries to minimize the following penalty function (usually for $p = 1$):

$$\varphi(x) = \|x\|^p$$

This is done because signals are often sparse and this penalty function tends to favour sparse signals. However, which p is best? In the literature people mostly tend to focus on as small a p as possible, but this is not a very good fit for real-world images. So what does make a good fit?

To analyze how well a certain norm/penalty function "fits" a signal it is possible to rewrite the penalty function to a form which uses a function that is similar to a probability density function:

$$\begin{aligned}\varphi(x) &= \|x\|^p \\ &= \sum_i |x_i|^p \\ &= \sum_i \log(e^{|x_i|^p}) \\ &= -\sum_i \log(g(x_i)) \\ g(x) &= e^{-|x|^p}\end{aligned}$$

As can be seen g is similar to a probability density function (pdf), specifically that of an exponential distribution. It is not properly scaled for it to be a true pdf, but this is not important for the minimization as any scaling would simply result in an additional constant term (due to the logarithm). *Since minimizing the negated sum of logarithms of $g(x_i)$ is equivalent to maximizing the product of $g(x_i)$, compressed sensing reconstruction can be interpreted as maximizing the probability of x under a distribution implied by $\varphi(x)$.*

Now it is both clear how the penalty function used in compressed sensing can be related to a probability density function, and how an (arbitrary) probability density function can be related to the penalty function. This suggests that a good penalty function would correspond to a pdf that comes as close to the pdf of the true signal as possible. Preliminary testing seems to support this.

A.4 Model-based recovery

Compressed sensing implicitly (or explicitly) uses a model of the signal to improve the quality of the reconstruction by finding a signal that both matches the measurements and is likely to occur, according to the model. Normally this model just assumes sparsity of the signal (possibly in some basis), which may fail to fully take advantage of the structure in real-world signals. In recursive filtering described in section 5.5 pretty much any model is allowed, but practically no guarantees can be given. In [3] an attempt is made to bridge this gap by introducing a restricted form of model that falls somewhere in-between the two extremes discussed here.

B Theoretical bounds

Naturally there is wide interest in establishing firm theoretical bounds on the quality obtainable through compressed sensing methods. This issue is far from settled, but this appendix tries to give an overview of some of the more common attempts at specifying bounds.

B.1 Important concepts

(In)coherence: Coherence $\mu(\Phi, \Psi)$ measures the the largest correlation between any two elements of Φ and Ψ [10, p. 4]:

$$\mu(\Phi, \Psi) = \sqrt{n} \cdot \max_{1 \leq k,j \leq n} |<\varphi_k, \psi_j>|$$

Restricted Isometry / Uniform Uncertainty: The isometry constant δ_s ($s \in \mathbb{P}$) of a matrix A is the smallest number such that:

$$(1 - \delta_s)\|x\|^2 \leq \|Ax\|^2 \leq (1 + \delta_s)\|x\|^2$$

holds for all s-sparse vectors x. See [13].

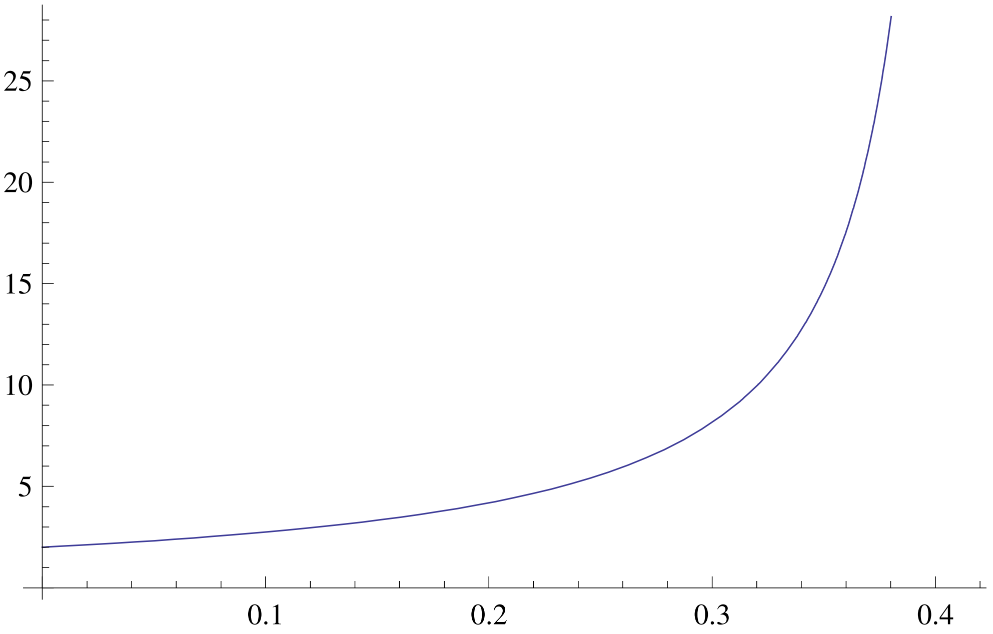


Figure 16: The value of C plotted against δ_{2k} as in bound 3.

B.2 Bound 1

According to [10] formulation 1 can recover any sparse signal with R components exactly with "overwhelming probability" $(1-\delta)$ as long as the following inequalities hold:

$$k \geq C\,R\,\mu^2(\Phi,\Psi)\log(l/\delta)$$
$$k \geq C'\ \log^2(l/\delta)$$

Or similarly (from [9]):

$$k \geq C\ \log^2(l)$$

Where C is roughly $23(M+1)$ and the probability of exact reconstruction is $1-O(l^{-M})$, as M approaches zero l^M approaches 1 (and then C is still 23), so the quality of the reconstruction is determined by the constants hidden in the O notation. Since the constant appears to be 1.25 (see page 17 of [9]) this does not give much real-world information, as in practice much lower constants are needed to give satisfactory results.

B.3 Bound 2

If Φ has properties CS1-3 from [22] and (k,l) is a sequence of problem sizes such that $k < l = Ak^\gamma$ $(A > 0, \gamma > 1)$, than the solution x^* of formulation 7 has the following guarantee (with R an upper bound for the p-norm of x, with $0 < p \leq 1$ and C depending on $p, A, \gamma, (\eta_i), \rho$):

$$\|x - x^*\|_2 \leq C\,R\,(k/\log(l))^{1/2-1/p}, k \to \infty$$

B.4 Bound 3

Assume that $\delta_{2k} < \sqrt{2}-1$ (with δ_{2k} an isometry constant of $\Phi\Psi$) then the solution x^* of formulation 1 obeys (x_k being x with all entries, except the k largest, set to zero):

$$\|x - x^*\|_1 \leq C\,\|x - x_k\|_1$$
$$\|x - x^*\|_2 \leq C\,k^{-1/2}\|x - x_k\|_1$$

See [13]. The constant C takes on the following value (in both cases), also see figure 16:

$$C = 2(1+\rho)(1-\rho)^{-1}$$
$$\rho = \sqrt{2}\,\delta_{2k}(1-\delta_{2k})^{-1}$$

B.5 Discussion

Obviously there have been quite a few attempts at establishing theoretical bounds on the performance of compressed sensing. But, they all have their problems that make it quite hard to really apply any of them to real-world scenarios. I'll point out some of the problems I encountered with all of the above mentioned methods.

Bound 1 only applies to truely sparse signals. And, perhaps more importantly, it contains a constant for which it isn't exactly clear what it is (it might be documented somewhere, but I haven't found a clear definition).

Bound 2 includes a hard to determine constant and only seems to apply to the otherwise least used and least flexible approach (in that it requires knowledge of the sparsity basis before taking measurements), the a priori formulation described in appendix A. In [48] some numerical estimates are computed of the constant used, and they are indeed quite reasonable in practice (between 0.0253 and 0.216).

Bound 3 (as well as any other bound that might use isometry constants) suffers from two major flaws. First of all, it's not that easy to get an actual isometry constant. There a number of proofs that indirectly give some probabilistic bounds on these constants for certain types of (random) matrices, but it is not immediately clear that the probabilities involved are indeed "overwhelming", nor is it obvious what the isometry constants actually are (see for example [11], section 3).

In an attempt to verify bound 3 numerically I had a look at estimating the isometry constants numerically by computing singular values of random subsets of colums of the often used Gaussian ensemble (a k-by-l matrix with $\mathcal{N}(0, \frac{1}{k})$ distributed random entries). This did indicate that the isometry constants for this matrix can indeed lie within the required range ($\delta_{2k} < \sqrt{2} - 1$). However, this is of course by no means a proof and definitely says nothing about other matrices.

Finally, the RIP apparently suffers from a number of flaws that could make it a bad indicator of the suitability of a matrix for compressed sensing. See for example [50] and [23] (and the presentation [24] comparing their data to the RIP).

Note that [50] contains an error bound that is superficially similar to bound 3 (theorem 4), also dealing with data that is not exactly sparse. However, this bound assumes that the attained solution is indeed a minimizer of 1 (and specifically that $\|x\|_1 \geq \|x^*\|_1$), which is not necessarily true in practice (and in fact was not true in my experiments). As the constant in this bound depends on the error in the solution I have not been able to get some values for this constant yet.

Finally, the theory on thresholds explained in [23] is also used in [26] to evaluate the performance of Stagewise OMP and it shows quite an excellent correspondence between theory and practice. As in practice the performance of compressed sensing does seem to more or less behave in the all-or-nothing way modelled by this theory (see figure 3 for example) it would be interesting to compare it to the real-world performance of other method.

C Gradient thresholding

This appendix desribes the gradient thresholding method as used in the results section. First the actual thresholding operation is introduced, then the algorithm is presented, and in the last subsection connections to other methods are discussed.

C.1 Thresholding

Thresholding applied directly to TV minimization can be quite tricky, however, a method similar in spirit to [34] is quite possible. Here the horizontal and vertical gradients are thresholded separately. In [19] the following iteration is used (where the columns of Ψ form an orthonormal basis):

$$\tilde{f}^{i+1} = \Psi H(\Psi^T(\tilde{f}^i + \Phi^*(y - \Phi\tilde{f}^i))) \tag{9}$$

Note that the original paper uses a slightly different formulation, but it should be equivalent with the substitution $\tilde{f} = \Psi\tilde{x}$ and some trivial manipulation.

Equation 9 can be rewritten and then moulded a bit to rely less on Ψ being a square orthogonal matrix:

$$\begin{aligned}
\tilde{f}^{i+\frac{1}{2}} &= \tilde{f}^{i} + \Phi^{*}(y - \Phi\tilde{f}^{i})) \\
\tilde{x}^{i+\frac{1}{2}} &= \Psi^{T}\tilde{f}^{i+\frac{1}{2}} \\
\tilde{x}^{i+1} &= H(\tilde{x}^{i+\frac{1}{2}}) \\
\tilde{f}^{i+1} &= \Psi\tilde{x}^{i+1}
\end{aligned}$$

$$\begin{aligned}
\tilde{f}^{i+1} &= \Psi(\tilde{x}^{i+1} + \tilde{x}^{i+\frac{1}{2}} - \tilde{x}^{i+\frac{1}{2}}) \\
\tilde{f}^{i+1} &= \Psi\tilde{x}^{i+\frac{1}{2}} + \Psi(\tilde{x}^{i+1} - \tilde{x}^{i+\frac{1}{2}}) \\
\tilde{f}^{i+1} &= \tilde{f}^{i+\frac{1}{2}} + \Psi(\tilde{x}^{i+1} - \tilde{x}^{i+\frac{1}{2}})
\end{aligned}$$

Using a directional derivative $D = \Psi^{*}$ this results in the following definitions which seem to work quite well in practice:

$$\begin{aligned}
\tilde{x}^{i+\frac{1}{2}} &= D\tilde{f}^{i+\frac{1}{2}} \\
\tilde{f}^{i+1} &= \tilde{f}^{i+\frac{1}{2}} + D^{T}(\tilde{x}^{i+1} - \tilde{x}^{i+\frac{1}{2}})
\end{aligned} \tag{10}$$

C.2 Algorithm

The thresholding iteration discussed above is used in an iterative manner as in [6, §3]. With a maximum of 50 inner loops and not increasing the sparsity bound by one at a time, but rather by the width/height of the image. The maximum number of inner loops is intentionally made higher than is usually needed; for most signals a few tens of inner iterations suffices for the first outer iteration and after that only a few (two or three) inner iterations are required.

In addition the initialization is done recursively (and/or based on a resized image) to make the method more robust to a noisy initialization. In the final algorithm (alg. 2) n is the width/height of the image, S is an upscale operator and $H_{V,s}/H_{H,s}$ are thresholding operations like the one defined above, for the vertical and horizontal direction (each keeping only the largest s coefficients).

Algorithm 2 f = solve($y \in \mathbb{R}^{k}$, $\Phi \in \mathbb{R}^{k\times n^{2}}$, ϵ, n, recursivecall)

```
if $(\frac{n}{2})^2 > k$ then
    $f \leftarrow S^{*}$solve($y$, $\Phi S$, $\epsilon$, $\frac{n}{2}$, true)
else
    $f \leftarrow S(\Phi S)^{+}y$
end if
$i_{max} := 50$
for $s = n$ to $n^2$ with step size $n$ do
    $pf \leftarrow f$
    for $i = 1$ to $i_{m}ax$ do
        $f \leftarrow H_{V,s}(f + \Phi^{T}(y - \Phi f))$
        $f \leftarrow H_{H,s}(f + \Phi^{T}(y - \Phi f))$
        if converged then
            break
        end if
    end for
    if $\|y - \Phi f\|_2 < \epsilon$ then
        break
    else if $recursivecall \wedge TV(f) > TV(pf)$ then
        break
    end if
end for
```

As the recursive calls mainly serve as a denoising step they are aborted as soon as the total variation rises. Also, in the implementation it is ensured that Φ is scaled so as to ensure that the magnitude of its largest singular value is 1 (by dividing it by the square root of the largest eigenvalue of $\Phi\Phi^{*}$).

C.3 Relationship with other methods

To relate the above defined iterations to other methods it is possible to work out the thresholding iterations for a more general sparsity inducing matrix A. The objective has the form:

$$F_{w,p}(f) = \|\Phi f - y\|^2 + \lambda \|Af\|^p$$

Now define the surrogate functional and its gradient with respect to f as in [19] ($p > 1$):

$$\begin{aligned}
F^S_{w,p}(f,a) &= \|\Phi f - y\|^2 + \lambda\|Af\|^p - \|\Phi f - \Phi a\|^2 + \|f - a\|^2 \\
&= f^*\Phi^*\Phi f - 2f^*\Phi^* y + y^* y + \lambda\|Af\|^p \\
&\quad - f^*\Phi^*\Phi f + 2f^*\Phi^*\Phi a - a^*\Phi^*\Phi a + f^* f - 2f^* a + a^* a \\
&= f^* f - 2f^*(a + \Phi^*(y - \Phi a)) + \lambda\|Af\|^p + a^* a + y^* y - a^*\Phi^*\Phi a \\
\nabla_f F^S_{w,p}(f,a) &= 2f - 2(a + \Phi^*(y - \Phi a)) + \lambda p A^T(|Af|^{p-1} * \operatorname{sign}(Af))
\end{aligned}$$

Note that either the largest eigenvalue of $\Phi^*\Phi$ should be less than one, or $\|f - a\|^2$ should be multiplied by a constant larger than the largest eigenvalue of $\Phi^*\Phi$.

Using $e = a + \Phi^*(y - \Phi a)$ (the result of a Landweber iteration) and solving for a zero of the gradient results in the following equation:

$$\begin{aligned}
\nabla_f F^S_{w,p}(f,a) &= 0 \\
2f - 2e + \lambda p A^T(|Af|^{p-1} * \operatorname{sign}(Af)) &= 0 \\
f + \frac{\lambda p}{2} A^T(|Af|^{p-1} * \operatorname{sign}(Af)) &= e \qquad (11)
\end{aligned}$$

Using the definitions:

$$\begin{bmatrix} A \\ \bar{A} \end{bmatrix} f = \begin{bmatrix} x_r \\ x_n \end{bmatrix} \qquad\qquad \begin{bmatrix} A \\ \bar{A} \end{bmatrix} e = \begin{bmatrix} w_r \\ w_n \end{bmatrix}$$

Equation 11 can be rewritten as:

$$\begin{aligned}
f + \frac{\lambda p}{2} A^T(|x_r|^{p-1} * \operatorname{sign}(x_r)) &= e \\
\begin{bmatrix} A \\ \bar{A} \end{bmatrix} (f + \frac{\lambda p}{2} A^T(|x_r|^{p-1} * \operatorname{sign}(x_r))) &= \begin{bmatrix} A \\ \bar{A} \end{bmatrix} e \\
\begin{bmatrix} x_r + \frac{\lambda p}{2} AA^T(|x_r|^{p-1} * \operatorname{sign}(x_r))) \\ x_n \end{bmatrix} &= \begin{bmatrix} w_r \\ w_n \end{bmatrix}
\end{aligned}$$

Assuming the above is solved for some x_r, then:

$$\begin{aligned}
f &= \begin{bmatrix} A \\ \bar{A} \end{bmatrix}^{-1} \begin{bmatrix} x_r \\ x_n \end{bmatrix} \\
&= \begin{bmatrix} A \\ \bar{A} \end{bmatrix}^{-1} \left(\begin{bmatrix} x_r \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ w_n \end{bmatrix} \right) \\
&= A^T(AA^T)^{-1} x_r + (I - A^T(AA^T)^{-1}A)e \\
&= e + A^T(AA^T)^{-1}(x_r - Ae)
\end{aligned}$$

Note that the above reduces to the usual thresholding scheme if A is orthonormal. Also note that if A is rectangular, but the rows are orthonormal ($AA^T = I$), then an iteration like 10 would be exact (in the sense that it would minimize the surrogate functional).

In the case of gradients the matrix $A = D$ is rectangular and the rows are not orthonormal. However, the matrix AA^T is diagonally dominant, with very few nonzeros off the diagonal. Even so, it seems to be advantageous to not use the exact form above, but rather the simpler form of equation 10 shown above (it has much better convergence behaviour).

Another way to view iteration 10 is as an undecimated (Haar) wavelet thresholding (see [40]), where the low-frequency components are left untouched. This can be seen by considering the following (not unique!) reconstruction formula for the undecimated Haar wavelet transform (where A gives the low-frequency components):

$$\begin{aligned}\tilde{f}^{n+1} &= A^T A \tilde{f}^{n+\frac{1}{2}} + D^T \tilde{x}^{n+1} \\ &= A^T A \tilde{f}^{n+\frac{1}{2}} + D^T (\tilde{x}^{n+1} - \tilde{x}^{n+\frac{1}{2}} + \tilde{x}^{n+\frac{1}{2}}) \\ &= A^T A \tilde{f}^{n+\frac{1}{2}} + D^T D \tilde{f}^{n+\frac{1}{2}} + D^T (\tilde{x}^{n+1} - \tilde{x}^{n+\frac{1}{2}}) \\ &= \tilde{f}^{n+\frac{1}{2}} + D^T (\tilde{x}^{n+1} - \tilde{x}^{n+\frac{1}{2}})\end{aligned}$$

The above ignores what happens at the boundary. To get an exact reconstruction the first and last row of A^T and D^T should be scaled by 2, but this usually hardly makes a difference.

In a more general setting matrices A, B and A^R, B^R with the property $A^R A + B^R B = I$ can be considered. First rewrite the surrogate functional as is done in [49] (above $\varphi(Af) = \lambda \|Af\|^p$ was used):

$$\begin{aligned}F^S_{w,p}(f,a) &= \|\Phi f - y\|^2 + \varphi(Af) - \|\Phi f - \Phi a\|^2 + \|f - a\|^2 \\ &= f^* \Phi^* \Phi f - 2 f^* \Phi^* y + y^* y + \varphi(Af) \\ &\quad - f^* \Phi^* \Phi f + 2 f^* \Phi^* \Phi a - a^* \Phi^* \Phi a + f^* f - 2 f^* a + a^* a \\ &= f^* f - 2 f^* (a + \Phi^* (y - \Phi a)) + \varphi(Af) + a^* a + y^* y - a^* \Phi^* \Phi a \\ &= \|f - (a + \Phi^*(y - \Phi a))\|^2 + \varphi(Af) + a^* a + y^* y - a^* \Phi^* \Phi a \\ &\quad - \|(a + \Phi^*(y - \Phi a))\|^2\end{aligned}$$

Using the definitions:

$$\begin{aligned}& A^R A + B^R B = I && x_A = Af && x_B = Bf \\ & e = a + \Phi^*(y - \Phi a) && w_A = Ae && w_B = Be \\ & \mathcal{A}^R = \begin{bmatrix} A^R & B^R \end{bmatrix} && x = \begin{bmatrix} x_A \\ x_B \end{bmatrix} &&\end{aligned}$$

Minimization of the surrogate functional with respect to f can be rewritten as follows (dropping all terms of $F^S_{w,p}(f,a)$ that do not depend on f):

$$\begin{aligned}\arg\min_f \ F^S_{w,p}(f,a) &= \arg\min_f \quad \|f - e\|^2 + \varphi(Af) \\ &= \mathcal{A}^R \arg\min_x \ \|A^R(x_A - w_A) + B^R(x_B - w_B)\|^2 + \varphi(x_A) \\ &= \mathcal{A}^R \arg\min_x \ \|A^R(x_A - w_A)\|^2 + 2 < A^R(x_A - w_A), B^R(x_B - w_B) > \\ &\qquad\qquad + \|B^R(x_B - w_B)\|^2 + \varphi(x_A)\end{aligned}$$

Note that the objective in general has no unique minimum. As x_A is supposed to be the "interesting" part to minimize, and we want to leave unrelated features alone as much as possible, we can choose to fix $x_B = w_B$. It is easily seen that this leads to the equation below and that this minimization also leads to a minimum of the above objective (as the gradient of the two middle terms is then zero):

$$\begin{aligned}\arg\min_f \ F^S_{w,p}(f,a) &= \mathcal{A}^R \begin{bmatrix} \arg\min_{x_A} \ \|A^R(x_A - w_A)\|^2 + \varphi(x_A) \\ w_B \end{bmatrix} \\ &= A^R x_A^* + B^R w_B \\ &= e + A^R \left(x_A^* - w_A\right)\end{aligned}$$

Where x_A^* is the solution of an ordinary regularized minimization problem, with the same regularisation term as in the original minimization problem, except that it has one less linear transformation (and can be solved by any ordinary thresholding method):

$$x_A^* = \arg\min_{x_A} \ \|A^R(x_r - w_r)\|^2 + \varphi(x_A)$$

This results in the following thresholding iteration:

$$\tilde{x}_A^{i+1} = H_\varphi(\tilde{x}_A^i + \left(A^R\right)^T A^R(w_r - \tilde{x}_A^i))$$

Note that in practice it does not seem to be advantageous to use an extra level of iteration though... This is somewhat in contrast to the results in [40], where instead of implementing the thresholding operation iteratively, the (pseudo-)inverse of the undecimated wavelet transform is computed iteratively, with very good results.

D Recursive filtering

In [28] the filtering formulation (see section A.2) is (indirectly) introduced. Here the original signal is iteratively approximated by finding a fixed point of a filtering and projection operation. Specifically, they define the following iterative process (where η^n is a noise term):

$$\begin{aligned}
\tilde{f}^n &= \begin{bmatrix}\Phi\\ \bar{\Phi}\end{bmatrix}^{-1}\begin{bmatrix}y\\ \tilde{z}^n\end{bmatrix}\\
\tilde{f}^{n+\frac{1}{3}} &= \Xi(\tilde{f}^n)\\
\tilde{z}^{n+\frac{1}{3}} &= \bar{\Phi}\tilde{f}^{n+\frac{1}{3}}\\
\tilde{z}^{n+\frac{2}{3}} &= \tilde{z}^{n+\frac{1}{3}} + \eta^n\\
\tilde{z}^{n+1} &= \gamma^n\tilde{z}^{n+\frac{2}{3}} + (1-\gamma)\tilde{z}^n
\end{aligned}$$

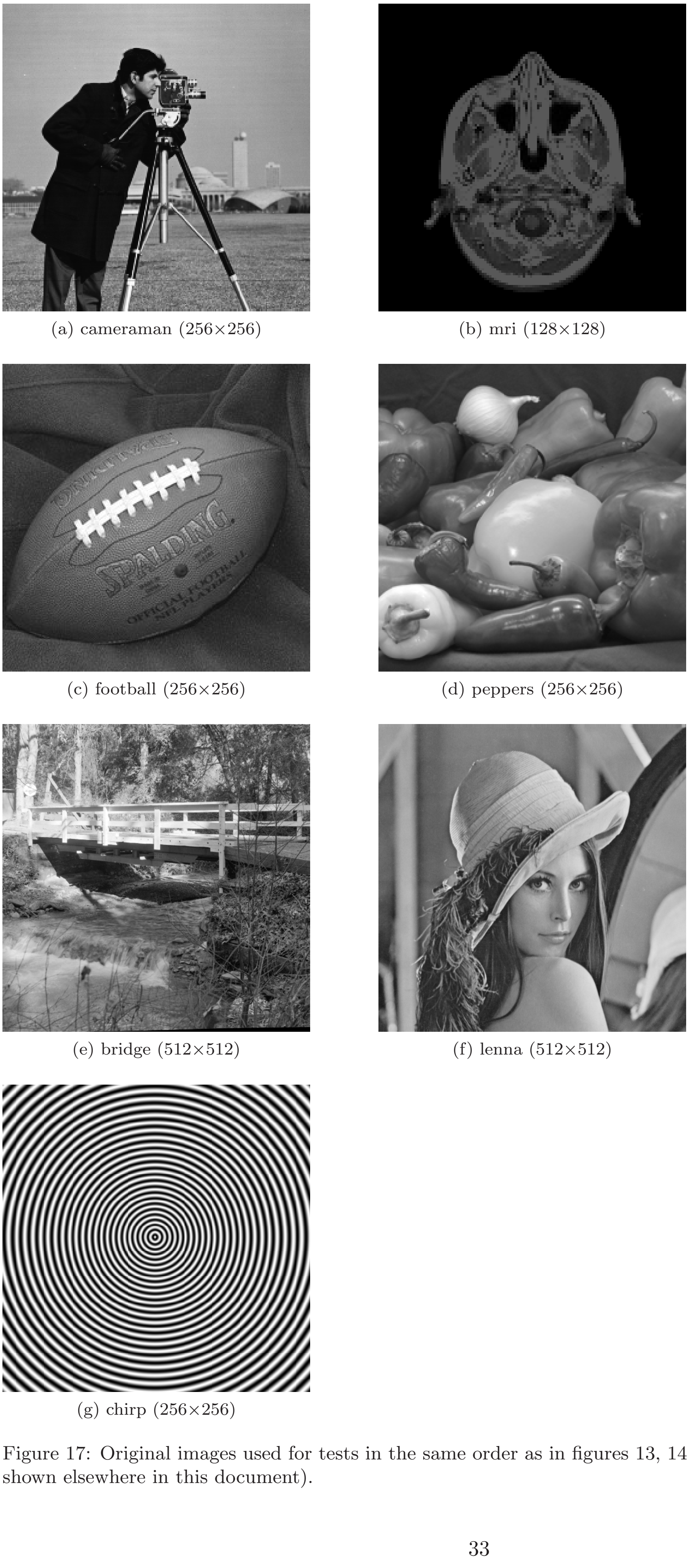
Equivalently, using $u = \begin{bmatrix}y\\ z\end{bmatrix}$ and $A = \begin{bmatrix}\Phi\\ \bar{\Phi}\end{bmatrix}$, we can define the following iteration:

$$\begin{aligned}
\tilde{f}^{n+\frac{1}{4}} &= \Xi(\tilde{f}^n)\\
\tilde{u}^{n+\frac{1}{4}} &= A\tilde{f}^{n+\frac{1}{4}}\\
\tilde{u}^{n+\frac{2}{4}} &= \tilde{u}^{n+\frac{1}{4}} + \eta^n\\
\tilde{u}^{n+\frac{3}{4}} &= \begin{bmatrix}y\\ 0\end{bmatrix} + \begin{bmatrix}0\\ \tilde{z}^{n+\frac{2}{4}}\end{bmatrix}\\
\tilde{u}^{n+1} &= \gamma^n\tilde{u}^{n+\frac{3}{4}} + (1-\gamma)\tilde{u}^n\\
\tilde{f}^{n+1} &= A^{-1}\tilde{u}^{n+1}
\end{aligned}$$

Note that $\Phi^*(\Phi\Phi^*)^{-1}y = \Phi^*(\Phi\Phi^*)^{-1}\Phi f$ is the projection of f onto the row space of Φ and the null space of $\bar{\Phi}$. Similarly, note that $(I - \Phi^*(\Phi\Phi^*)^{-1}\Phi)f$ is the projection of f onto the null space of Φ and the row space of $\bar{\Phi}$. Using this the above iteration can be kept completely in the domain of the original signal:

$$\begin{aligned}
\tilde{f}^{n+\frac{1}{4}} &= \Xi(\tilde{f}^n)\\
\tilde{f}^{n+\frac{2}{4}} &= \tilde{f}^{n+\frac{1}{4}} + A^{-1}\eta^n\\
\tilde{f}^{n+\frac{3}{4}} &= A^{-1}(\begin{bmatrix}y\\ 0\end{bmatrix} + \begin{bmatrix}0\\ \tilde{z}^{n+\frac{2}{4}}\end{bmatrix})\\
&= \Phi^*(\Phi\Phi^*)^{-1}y + (I - \Phi^*(\Phi\Phi^*)^{-1}\Phi)\tilde{f}^{n+\frac{2}{4}}\\
&= \tilde{f}^{n+\frac{2}{4}} + \Phi^*(\Phi\Phi^*)^{-1}(y - \Phi\tilde{f}^{n+\frac{2}{4}})\\
\tilde{f}^{n+1} &= \gamma^n\tilde{f}^{n+\frac{3}{4}} + (1-\gamma)\tilde{f}^n
\end{aligned}$$

Note that this iteration bears great similarity to the iterative thresholding iteration. In fact, if a thresholding operation is substituted for Ξ, and no noise is added, the only essential difference is in using $\tilde{f} + \Phi^*(\Phi\Phi^*)^{-1}(y - \Phi\tilde{f})$ instead of the Landweber iteration $\tilde{f} + \Phi^*(y - \Phi\tilde{f})$. Assuming the eigenvalues of $\Phi^*\Phi$ are not too large the Landweber iteration converges to the least squares solution though, and in practice this difference seems to have little impact (and the Landweber iteration is considerably faster!).



(a) cameraman (256×256)

(b) mri (128×128)

(c) football (256×256)

(d) peppers (256×256)

(e) bridge (512×512)

(f) lenna (512×512)

(g) chirp (256×256)

Figure 17: Original images used for tests in the same order as in figures 13, 14 and 18 (most of these are also shown elsewhere in this document).

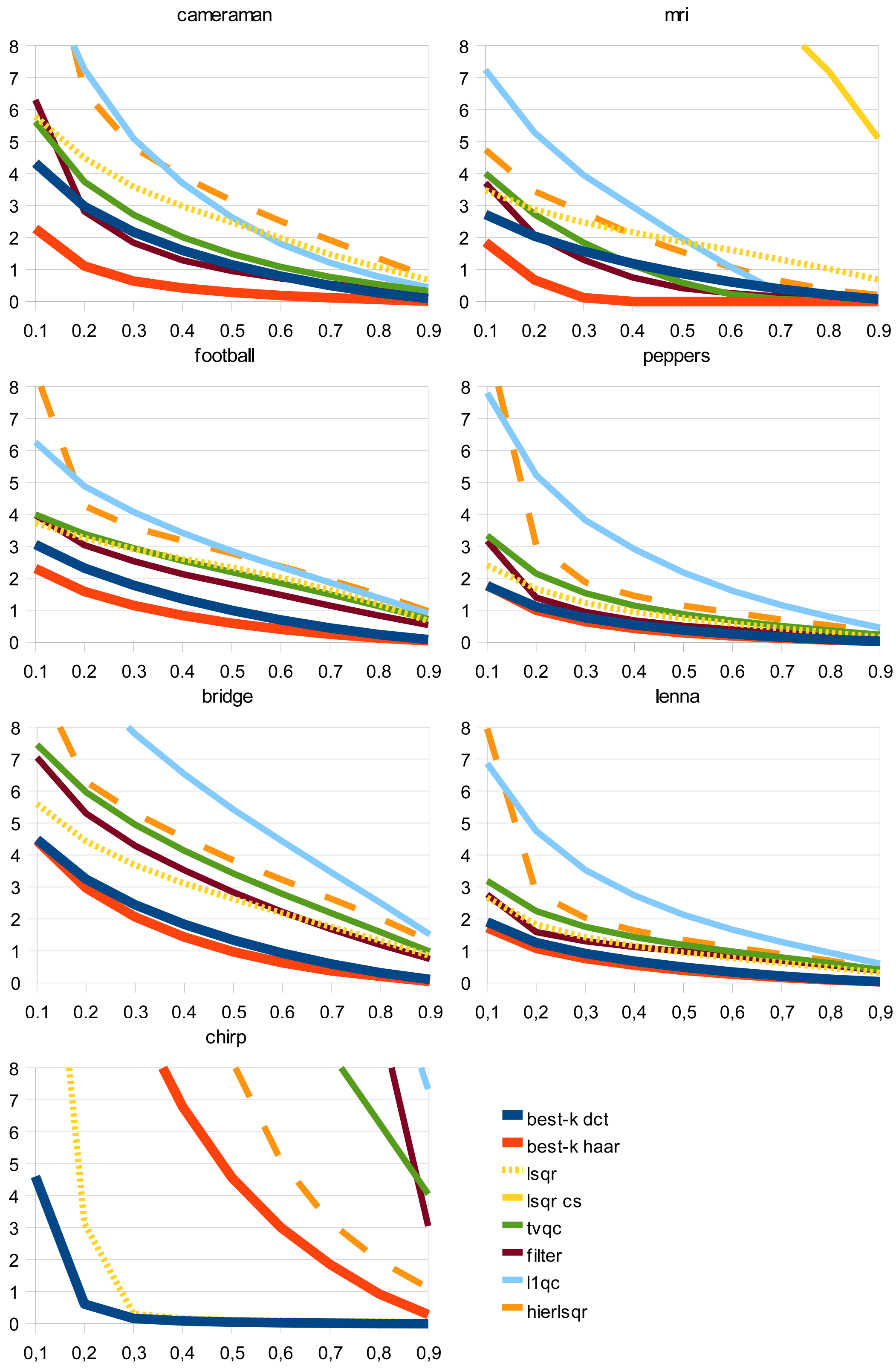


Figure 18: Graphs of the RMS values (scaled to percentages of full-scale) for different values of k as a fraction of l. The formula used is $100 \cdot \frac{\|x^* - x\|}{255\sqrt{l}}$.