COMPRESSIVE SINGLE-PIXEL IMAGING

Technical report

Andrew Thompson, University of Edinburgh

20 January 2011

COMPRESSIVE SINGLE-PIXEL IMAGING

Technical report

Andrew Thompson, University of Edinburgh

20 January 2011

1. INTRODUCTION

This report relates to an internship project carried out at *SELEX Galileo* from August 2010 to January 2011 entitled *Compressive Imaging*. The project was facilitated by the *Knowledge Transfer Network (KTN)* as part of the *Underpinning Defence Mathematics (UDM)* scheme, and was a collaboration between *SELEX Galileo* and the *University of Edinburgh*.

The project concerns a new approach to single-pixel imaging which exploits the theory of Compressed Sensing (CS). The camera design in question was first proposed by a team at Rice University [6], and a proof-of-concept model for visible light was also built. The aims of this report are

- to provide background on the camera design
- to report the results of numerical experiments conducted on a model of the camera design
- to highlight some of the key design issues that would be likely to arise.

The structure of the report is as follows: In Section 2, we provide background on Compressed Sensing, and then explain in Section 3 how the proposed single-pixel camera fits naturally into such a theory. Sections 4 and 5 describe the system model that we built, first focussing on the CS reconstruction algorithms options (Section 4), and then describing the model options more generally (Section 5). The most substantial part of the report is Section 6 which reports the results of numerical experimentation upon the model.

2. BACKGROUND ON COMPRESSED SENSING

Compressed Sensing (CS) is an emerging paradigm which challenges some of the established approaches to signal processing. The traditional wisdom is that, in order to completely capture the information contained in a signal, it is necessary to sample at a rate at least twice the bandwidth, often referred to as the Nyquist rate. In many technologies, sampling in this way leads to such a large volume of data that, in order to satisfy storage or transmission requirements, the data must be compressed. There seems to be something intuitively wasteful about such an approach in which a full set of samples are acquired, only for much of the data to be subsequently discarded. Compressed Sensing, by contrast, asserts that it is possible to accurately reconstruct signals from sub-Nyquist sampling, provided we make some additional assumptions about the signal in question, and provided we choose an appropriate sampling methodology. It is clear that the incentives for adopting such an approach in terms of sampling time and cost could be significant. CS has many potential applications, but our interest in this report is on the potential use of CS techniques in imaging.

Suppose then that we have an unknown discretized signal which we model as a vector $x \in R^N$, and suppose that we obtain linear measurements of x, which we can view as inner products of x with a series of 'test function' vectors $\{\phi_i\}$, so that

$$y_i = \langle x, \phi_i \rangle + e_i \text{ for } i = 1, \ldots, n,$$

where e_i is sampling noise. Writing Φ for the $n \times N$ sampling matrix whose rows are the test functions $\{\phi_i\}$, we may represent the sampling process as the matrix equation

$$y = \Phi x + e,$$

where $y \in R^n$ is the vector of samples.

Translated into the discrete setting, Nyquist sampling would correspond to taking a full set of measurements, *i.e.* $n = N$. If, in addition, these measurements were linearly independent, reconstruction would be possible by simply inverting the sampling matrix Φ. By contrast, Compressed Sensing in the discrete setting studies the case in which we take an incomplete set of measurements, *i.e.* $n < N$. In this case, the system is underdetermined and there is insufficient information as it stands to reconstruct the original signal x.

A key observation, however, is that many signals have information content much less than might be suggested by their dimension. In particular, signals are often compressible in the sense that there is some transform domain in which the sizes of signal coefficients decay rapidly, so that it can be well-approximated by a sparse vector in which many of the coefficients are zero. In imaging, for example, the JPEG and JPEG-2000 compression standards are based on the fact that natural images are often compressible in a Discrete Cosine Transform (DCT) or wavelet basis. We will therefore make the modelling assumption that the signal is sparse in some transform domain. In this regard, let Ψ be an $N \times N$ transform matrix, and suppose that $z = \Psi x$ has exactly k non-zero coefficients, where we assume $k \le n \le N$. We will refer to such a vector as being k-sparse.

The task of reconstructing the original signal from undersampled measurements now seems intuitively more realistic since the sparsity assumption is really saying that the true information content of the signal is much lower than its dimension. In fact, we can now frame the reconstruction task as an optimization algorithm. To do this, observe that we have two goals to achieve: we wish to obtain a close approximation to the linear system $y = \Phi x$, while we also wish to obtain a k-sparse solution. This can be achieved by solving the optimization problem

$$\min_{x \in R^N} \|y - \Phi x\|_2 \;\; subject\ to \;\; \|\Psi x\|_0 \le k,$$

where we minimize an l_2-norm objective function to achieve a good fit to the linear system, and where we use an l_0-norm constraint to control the sparsity (the l_0-norm simply counts the number of non-zero coefficients of a vector). The challenge of CS reconstruction is now to identify algorithms for solving this problem which, on the face of it, appears to be combinatorial in nature and therefore far from straightforward.

In fact, another key ingredient is needed for the approach just outlined to be effective: we must sample incoherently. Incoherent sampling can be described and motivated as followed: Suppose we take the sampling matrix Φ to be simply selected rows of the identity matrix. In this case, information from all the coefficients corresponding to the selected rows would be captured, but any coefficients corresponding to unselected rows would be completely ignored, meaning that accurate reconstruction would be impossible. What we need, then, is a sampling scheme which 'mixes' the information from all the coefficients, so that the samples appear as random noise: this is the principle of incoherence. Perhaps surprisingly, it turns out that completely random matrices have good incoherence properties with high probability, and are therefore often the favoured choice of sampling matrix.

In a nutshell, then, the CS paradigm asserts that it is possible to reconstruct compressible signals from undersampled incoherent measurements.

Many CS reconstruction algorithms have been proposed in recent years, supported by a strong theoretical foundation, see for example [5]. In fact, provided certain incoherence conditions are satisfied, it has been shown somewhat incredibly that the above l_0-constrained problem shares precisely the same solution as the l_1-constrained problem

$$\min_{x \in \mathrm{R}^N} \|y - \Phi x\|_2 \;\; subject\ to \;\; \|\Psi x\|_1 \le \tau.$$

Such an observation has massive implications since the problem can now easily be converted to a convex bound-constrained quadratic program, for which many tractable and efficient optimization methods exist. Theoretical guarantees for so-called l_1-minimization are especially well-developed, but other CS reconstruction algorithms have also been developed, some of which actually operate upon the original l_0-constrained problem.

We now move on to consider the CS single-pixel camera design and show how it fits naturally into the CS framework.

3. THE CS SINGLE PIXEL CAMERA

3.1 Hardware design

In contrast to a conventional camera which would have a vast array of photon detectors – one for each pixel – a single-pixel camera is so-called because it has only a single photon detector or 'pixel'. The incident light field is directed onto a specialized type of Spatial Light Modulator (SLM) known as a Digital Micromirror Device (DMD) which consists of an array of tiny mirrors, one corresponding to each pixel. Each mirror can be oriented in one of two directions: the 'on' position directs the light for that pixel towards the single detector, while the ‘off’ position directs the light away from the detector. The light from all the pixels set to the ‘on’ position is then summed and a measurement is recorded as an output voltage on the photon detector. The measurement is then subsequently digitized by an A/D convertor. A series of measurements can be obtained by flipping the mirrors and repeating the process a number of times. Clearly, the result of such a procedure is encoded measurements, and the other issue to address is how to decode these measurements and reconstruct the incident image.

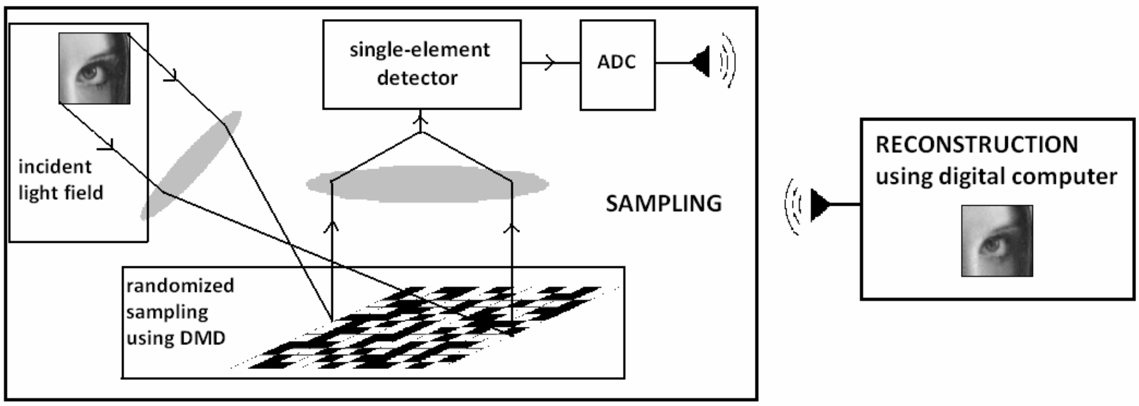


Figure 3.1: The Rice CS single-pixel camera

The concept of a single-pixel camera is not a new one, and it fits into the broad category of multiplexing imaging methods in which a series of consecutive measurements are made by a single detector. What sets the CS single-pixel camera model apart is a novel sampling approach which means that it is possible to take fewer measurements. It therefore offers an alternative approach to obtaining compressed images: rather than taking a full set of samples and subsequently compressing, compressed samples are acquired in the first place. To achieve this, CS theory motivates the use of a random sampling procedure in which each mirror is set randomly to either the ‘on’ or ‘off’ position with equal probability. Figure 3.1 shows a diagram of the proposed camera design.

Such an approach is likely to offer four main advantages.

- In common with other single-pixel architectures, only one photon detector is required as opposed to a full pixel array. Compared to visible light, photon detectors for other wavebands can be much more bulky and expensive to produce. A single-pixel approach could therefore make possible image acquisition at wavelengths for which it would otherwise have been impossible due to size or cost constraints.

- Compared to other single-pixel approaches, fewer samples means that sampling time could be shorter, a potentially crucial consideration for defence applications.

- Since fewer samples are taken, the volume of acquired data is reduced, which is another important consideration since there will often be constraints on the volume of data that can either be stored or transmitted, for example an airborne device.

- Compared to traditional sample-then-compress methods, computational processing is transferred from the front-end sampling phase to the back-end reconstruction phase. For example, in a traditional approach, an airborne sensing device may need to process and compress the data, in order to transmit the data to the ground. By contrast, in the CS paradigm, the compressed, encoded samples can be immediately transmitted to the ground for reconstruction. This is significant since it is likely that greater computational capacity is available on the ground as opposed to in the air.

We now turn to a mathematical model of the camera design outlined in this section.

3.2 A mathematical model

Let us model the incident light field as a discrete pixelated array consisting of N pixels, We may also represent this 'original image' as a vector $x \in R^N$. By means of the DMD, the light from all the pixels set to the 'on' position is summed and recorded as a voltage on the photon detector. We may model this summation as an inner product of the original image x with a test function $\phi_i \in R^N$ consisting of random ones and zeros, giving a single measurement $y_i \in R.$ Sampling error is likely to occur, particularly as a result of photon counting noise, and subsequently due to quantization error in the digitization. We choose to adopt a simplistic model for sampling noise in the form of additive Gaussian white noise. We may therefore write

$$y_i = \langle x, \phi_i \rangle + e_i,$$

where $e_i \sim N(0, \sigma^2)$ for some noise parameter $\sigma.$

In keeping with the CS framework introduced in Section 2, we choose to undersample the image and take n such measurements, where $n < N.$ Writing Φ for the $n \times N$ matrix whose rows are the test functions $\{\phi_i\}$, we may represent the entire sampling process by the matrix equation

$$y = \Phi x + e,$$

where $y \in R^n$ is the vector of samples. The sampling matrix Φ, as introduced here, is therefore a random Bernoulli matrix consisting of equiprobable ones and zeros. The DMD would in fact also permit the use of other random sampling matrices – for example random ±1 entries – and we refer the reader to Section 6.6 for a discussion of other options. Note that this model fits precisely into the general CS framework introduced in Section 2.

The samples are then sent to a digital computer for decoding. As was explained in Section 2, the reconstruction is based upon the assumption that the original image is compressible in some transform domain, so that it can be well-approximated by a sparse vector in that domain. Writing Ψ for the $N \times N$ matrix which applies some linear transform to the image (such as a DCT or Haar wavelet matrix), we then use an appropriate optimization algorithm to solve the problem

$$\min_{x \in \mathrm{R}^N} \|y - \Phi x\|_2 \ \ subject\ to\ \ \|\Psi x\|_0 \le k,$$

as outlined in Section 2. Since this is the key remaining challenge, we turn next to the question of reconstruction algorithms.

4. COMPRESSIVE IMAGING ALGORITHM OPTIONS

We showed in Section 2 how the Compressed Sensing problem of reconstructing a compressible image from undersampled linear measurements may be translated into the optimization problem

$$\min_{x \in \mathrm{R}^N} \|y - \Phi x\|_2 \ \ subject\ to\ \ \|\Psi x\|_0 \le k,$$

where Φ is the sampling matrix and Ψ is the sparsifying transform matrix. Equivalently, writing $z = \Psi x$ for the transform vector and $A = \Phi \Psi^{-1}$, we may instead solve in the transform domain the problem

$$\min_{z \in \mathrm{R}^N} \frac{1}{2}\|y - Az\|_2^2 \ \ subject\ to\ \ \|z\|_0 \le k,$$

where we now choose to square the l_2-norm and insert a factor of 1/2. A naïve approach to solving this problem would be an exhaustive search through all possible support sets for the vector z, which is combinatorial in complexity. The task therefore is to derive algorithms for solving the above problem which are computationally tractable. We now turn to the first of these which we chose to implement.

4.1 Normalized Iterative Hard Thresholding (NIHT)

All the algorithms implemented in the model fall into the broad category of gradient projection algorithms, which are established methods for solving constrained optimization problems of the form

$$\min_{x \in R^N} f(x) \ \ subject\ to\ \ x \in F,$$

where $f(x)$ is some objective function and F is some feasible set. Gradient projection methods are iterative procedures which alternate between two steps: a gradient step and a projection step. In the gradient step, we add on some multiple of the negative gradient of the objective function $f(x)$ evaluated at the current iterate, with the goal of reducing the objective. In the projection step, we project the current iterate onto to the feasible set F, thereby forcing the next iterate to satisfy the constraints.

For such a method to make sense, we require that the objective function is differentiable, and also that it is possible to project onto the feasible set. Both requirements are met in the case of our particular problem. Writing $f(z) = \frac{1}{2}\|y - Az\|_2^2$ for the objective function, the gradient is $\nabla f(z) = -A^T(y - Az)$.

Concerning the projection, we wish to project onto the set of k-sparse vectors, which may be done by applying the Hard Threshold operator H_k which simply sets all but the k largest in magnitude coefficients of a vector to zero. We therefore arrive at the Iterative Hard Thresholding (IHT) algorithm [2], in which a sequence of iterates $z_0, z_1, z_2, \ldots$ is generated by repeatedly applying the iteration

$$z_{m+1} = H_k[z_m + \alpha_m A^T(y - Az_m)],$$

where α_m is a step-size.

The most computationally burdensome steps in each iteration are the Hard Threshold operator (which involves sorting the coefficients) and matrix-vector multiplications by A and A^T. Since $A = \Phi\Psi^{-1}$, multiplication by A can be performed by applying the fast inverse transform Ψ^{-1} followed by multiplication by the sampling matrix Φ. If the transform matrix Ψ is orthogonal (as is the case for the DCT and Haar wavelets), we have $A^T = (\Phi\Psi^T)^T = \Psi\Phi^T$ which can then be computed in a similar fashion. In the case of Daubechies wavelets, Ψ is not orthogonal, but we nonetheless use the approximation $A^T \approx \Psi\Phi^T$ so as to be able to use the fast transform for Ψ. Also built into the code is the option to switch both the 5-3 and 9-7 Daubechies wavelet transforms to an equivalent orthogonal version.

It is important to choose the step-size scheme wisely, and the particular scheme we employed was the Normalized IHT step-size scheme proposed by Blumensath and Davies [4]. This step-size scheme in fact gives the greatest possible decrease in the objective whenever the support of consecutive iterates is the same. In the case where the support changes a backtracking procedure is used to guarantee a decrease in the objective. NIHT is guaranteed to converge to a local solution of the optimization problem. However, since the problem is nonconvex, there are potentially many such local solutions, and somewhat strong conditions on the conditioning of the matrix A are in fact required to guarantee accurate reconstruction of the original image x.

Since this algorithm employs the Hard Threshold operator H_k, the desired sparsity of the solution k is the tuning parameter in this case.

Though the algorithm generates a sequence of iterates in the transform domain, the iterates $x_0, x_1, x_2, \ldots$ in the image domain may be obtained by applying the inverse transform Ψ^{-1}.

A suitable termination criterion must also be chosen. In our case, we terminated whenever the difference between consecutive iterates $\|x_{m+1} - x_m\|_2$ was less than some fixed tolerance level.

4.2 l_1-Projection (SPGL1)

Another approach is to reformulate the original optimization problem by replacing the l_0-norm by the l_1-norm, which is defined as $\|z\|_1 = \sum_{i=1}^{N} |z_i|$. This is motivated by the well-established observation that solutions with small l_1-norm also tend to have small l_0-norm, a property that has been explored extensively in the CS literature. We therefore choose to solve the problem

$$\min_{z \in \mathrm{R}^N} \frac{1}{2}\|y - Az\|_2^2 \;\; subject\ to \;\; \|z\|_1 \le \tau,$$

for some choice of the tuning parameter τ. This problem may also be solved using a gradient projection approach, since it is also possible to efficiently project onto the l_1-ball $\{\|z\|_1 \le \tau\}$. Such an approach was proposed by Van den Berg and Friedlander as part of the SPGL1 (Spectral Projected Gradient for l_1) algorithm, which is freely available to download on the web [7]. The code we implemented is very similar to SPGL1.

Analogous to IHT, the algorithm consists of repeated application of the iteration

$$z_{m+1} = P_\tau[z_m + \alpha_m A^T(y - Az_m)],$$

where P_τ denotes projection onto the the l_1-ball $\{\|z\|_1 \le \tau\}$. The choice of step-size scheme is again important, and we followed SPGL1 in selecting the so-called Barzilai-Borwein step-size scheme. Since the projection in this case is a convex projection, the problem above has a unique solution, and (provided backtracking is used – see below) our algorithm is guaranteed to converge to this unique solution. In general, somewhat strong conditions on the conditioning of the matrix A are required to guarantee that this solution does in fact give an accurate reconstruction of the original image x, though in the specific case of Gaussian matrices, much weaker conditions have been derived which mirror how the algorithm behaves in practice.

Two additionally options have also been built into the l_1-projection algorithm: debiasing and backtracking. Debiasing refers to a post-processing phase after the l_1-projection algorithm has terminated. Using the solution from the l_1-projection algorithm as a starting point, a simplified version of the NIHT algorithm is run in which we do not allow any further changes to the support set. In this case, the NIHT algorithm simply reduces to the classical steepest descent algorithm for unconstrained optimization. The motivation for the use of a debiasing phase is that projecting onto the l_1-ball has the effect of shrinking the non-zero coefficients, leading to a bias towards solutions with smaller coefficient magnitudes. The debiasing phase is an attempt to correct for this. Essentially, l_1-projection is used to identify the support of the solution, and the debiasing phase is used to determine the precise values of the selected coefficients.

Backtracking is a common technique in optimization algorithms for appropriately controlling step-sizes. As was described earlier, the NIHT algorithm includes a backtracking stage whenever the support set of the iterates changes. In fact, the

SPGL1 algorithm (which our algorithm resembles closely) also includes a backtracking step. However, we found that for ±1 and Gaussian sampling schemes the backtracking step was not in practice needed and, moreover, slowed the algorithm down considerably, which is why the algorithm was implemented with the default option of no backtracking. However, we found that a backtracking step was required when a Bernoulli zero-one sampling scheme was used – see the discussion on sampling schemes in Section 6.5.

4.3 Iterative Tree Thresholding

The previous two algorithms are based upon the assumption that the original image can be well-approximated by a sparse vector in the transform domain. However, in the case of wavelet transforms, it is possible to make further assumptions about the transform coefficients. In particular, 2D wavelet transforms break up the image into progressively finer and finer scales, which leads to a quad-tree structure in which each coefficient has exactly one 'parent' and four 'children'. Large coefficients of piecewise-smooth images tend to be 'inherited' by their 'children', so that the large coefficients of such images can be well-approximated by a connected subtree. This motivates a more refined image model, where we assume not only that our image is k-sparse in a given wavelet domain, but also that its coefficients form a connected subtree.

We may now conceive of an adaptation to NIHT in which we replace the Hard Threshold with a projection onto the set of vectors whose coefficients form a connected subtree. What makes it possible to implement this idea is that such a projection is well-defined and, furthermore, there exists an algorithm which accomplishes it, namely the Condense Sort and Select Algorithm (CSSA) [3]. Unfortunately, however, projection by means of the CSSA becomes the most computationally burdensome part of the algorithm. To mitigate this, we implemented an approximation to the projection in the form of a greedy quad-tree search algorithm, which is significantly faster.

Writing G_k for the operator which applies the greedy quad-tree search algorithm, we proceed by repeated application of the iteration

$$z_{m+1} = G_k[z_m + \alpha_m A^T(y - Az_m)],$$

where we use the same step-size scheme as for NIHT.

5. CAMERA MODEL DESCRIPTION

The CS single-pixel camera may be modelled as a two-stage process of sampling followed by reconstruction. The sampling model takes an input image, converts it to a vector x, and computes the undersampled measurements $y = \Phi x + e$. The reconstruction process seeks to reconstruct the original image from the measurements by employing a CS reconstruction algorithm. A MATLAB implementation of this model has been built, which enabled us to carry out a range of numerical tests, the results of which are the subject of the next section.

5.1 Model options

The model is implemented with a number of options.

(a) Input image. The model allows essentially any square input image to be entered.

(b) Sampling scheme. The simplest design of the DMD, in which each mirror is randomly assigned an 'on' or 'off' position, corresponds to a sampling matrix consisting of random zeros and ones. We may also consider adaptations to this design, which give rise to different sampling matrices. For example, if for one frame all the mirrors are set to the 'on' position, the mean light intensity can be measured and subtracted from each measurement, and the sampling process can be modelled by a Bernoulli matrix consisting of random plus or minus ones. Alternatively, by dithering the mirrors on and off during the integration time, it may be possible to program the DMD with other sampling matrices, such as Gaussian matrices, where each entry is drawn independently from a standard Normal distribution. The model allows the choice of any of the three sampling scheme options described above.

(c) Reconstruction algorithm. There is a choice of three reconstruction algorithms:

(i) Normalized Iterative Hard Thresholding (NIHT)
(ii) l_1-projection (based on SPGL1)
(iii) Iterative Tree Thresholding.

The l_1-projection algorithm also has two additional options: debiasing and backtracking. The l_1-based method tends to shrink the image coefficients, and a debiasing phase can be used to correct for this and achieve a sharper image. See the discussion of signal-to-noise ratio in the next section for an illustration of the value of debiasing. Backtracking refers to a more conservative step-size scheme in the l_1-projection method.

(d) Sparsifying transform. Underpinning the CS approach is the assumption that the image in question is naturally compressible (or approximately sparse) in some transform domain. Four 2D transforms are available to choose from:

(i) Discrete Cosine Transform (DCT)
(ii) Haar wavelets
(iii) Daubechies 5-3 biorthogonal wavelets
(iv) Daubechies 9-7 biorthogonal wavelets.

These transforms are established industry standards for image compression. The DCT is the basis of the JPEG compression standard, and the two Daubechies wavelets are the two transforms used by JPEG-2000. Daubechies wavelets are generally a good option for piecewise smooth images, while Haar wavelets represent an alternative for naturally angular images. The various options allow an appropriate choice of sparsifying transform to be made for the image in question. Due to their quad-tree structure, all 2D wavelet transforms require the image size to be a power of two. This quad-tree structure is directly exploited by the Iterative Tree Thresholding algorithm, and it only makes sense to use this algorithm option in conjunction with wavelet transforms, and not the DCT.

5.2 Parameter choice

As well as the above options, there are also three parameters that the model user is able to control.

(a) Undersampling ratio. Given an input image of dimension N, we take $n \leq N$ samples, and so the ratio $\delta = n / N \in (0,1]$ gives the level of undersampling.

(b) Tuning parameter. Each algorithm requires the input of a tuning parameter which either directly or indirectly controls the sparsity of the reconstructed solution in the transform domain. For NIHT and Iterative Tree Thresholding, the tuning parameter is explicitly the sparsity k. For l_1-projection, the tuning parameter is τ, the l_1-norm of the reconstructed solution.

(c) Noise level. Noise in the sampling process is modelled as additive Gaussian white noise, the volume of which may be controlled by the parameter σ, so that noise drawn i.i.d. from a $N(0,\sigma^2)$ distribution is added to each measurement.

A key aim of the experimentation carried out was to understand the interplay between these parameters, and how they affect performance metrics such as accuracy of reconstruction. In particular, the crucial practical question is how the accuracy of reconstruction is impacted by reducing the number of measurements, *i.e.* decreasing the undersampling ratio δ.

6. EXPERIMENTATION

6.1 A standard test image

Our interest is especially in quantifying image reconstruction accuracy, and investigating how it is affected by the key model parameters. A natural choice of metric for reconstruction accuracy is the RMSE of the output $\hat{x}$ compared to the input x, *i.e.*

$$RMSE = \sqrt{\frac{1}{N}\|\hat{x} - x\|_2^2}.$$

There are other possible choices of metric for reconstruction accuracy, for example peak signal-to-noise ratio (PSNR) which is considered in Section 6.10. As well as reconstruction accuracy, we are also interested in computational efficiency, and so the running time of the reconstruction is also a metric of interest. As a first example, we applied the model to a 256x256 cut from the standard lena test image, which is a grey-scale image with integer pixel values between 0 and 255. The sampling scheme was ± 1 sampling and reconstruction was using the l_1-projection algorithm, and with 9-7 wavelets as the sparsifying transform. The noise level was set to zero, and we selected an undersampling ratio of 0.2 (number of samples a fifth of the number of pixels). We sought to optimize the tuning parameter τ, and found a value of 0.5 to be roughly optimal. The original and reconstructed images are shown in Figure 6.1. The RMSE was 15.36 and the running time on a University of Edinburgh workstation consisting of two x5650 6-core processors was 48.4 seconds. The result illustrates that, while undersampling leads to a degradation in image quality, a clearly recognizable image has been produced using significantly fewer samples than is traditionally thought possible. The next stage was a more systematic investigation into the impact of undersampling upon performance metrics.

Figure 6.1: 256x256 lena test image original and reconstruction using l_1-projection (δ **=0.2)**

6.2 Systematic testing of the parameter space

If we make the simplifying assumption that there is zero sampling noise, there are two parameters that we are free to vary: the undersampling ratio δ and the tuning parameter (k or τ). We may therefore test the model on a mesh of equally-spaced grid-points throughout the parameter space, and use this to identify the optimal parametric configuration for the test data. For each point on the mesh, 100 trials were performed and the results averaged. In each trial, the model is applied to the same 64x64 cut from lena (showing the recognizable feature of an eye). Performance metrics of the solution RMSE and running time, along with other data of interest such as number of iterations and sparsity of the solution, are then recorded. This test framework can then be applied to any combination of the sampling scheme, reconstruction algorithm and sparsifying transform options.

Figure 6.2 shows a plot of average RMSE throughout the parameter space for ± 1 sampling, the l_1-projection algorithm and 9-7 wavelets. The tuning parameter τ is here normalized as a factor of the l_1-norm of the original image in the transform domain, which we refer to as the τ -factor.

A key feature, which was in fact observed across the whole range of model options, is the existence of an 'optimal' tuning parameter for a given level of undersampling, *i.e.* the value of τ -factor which minimizes the RMSE. The superimposed black curve traces out the optimal tuning curve. The interpretation of this dynamic is as follows: Below the curve, the l_1-projection algorithm is able to obtain a good approximation to the optimally-compressed image for a given compression ratio determined by the tuning parameter. As the tuning parameter is increased, the compression ratio increases and a better and better approximation is achieved to the image as more and more wavelet coefficients are included. Above the curve, while the target compression ratio becomes more and more favourable, the algorithm becomes less and less able to find a good approximation to the optimal compression – essentially the algorithm no longer has enough information to identify the required solution.

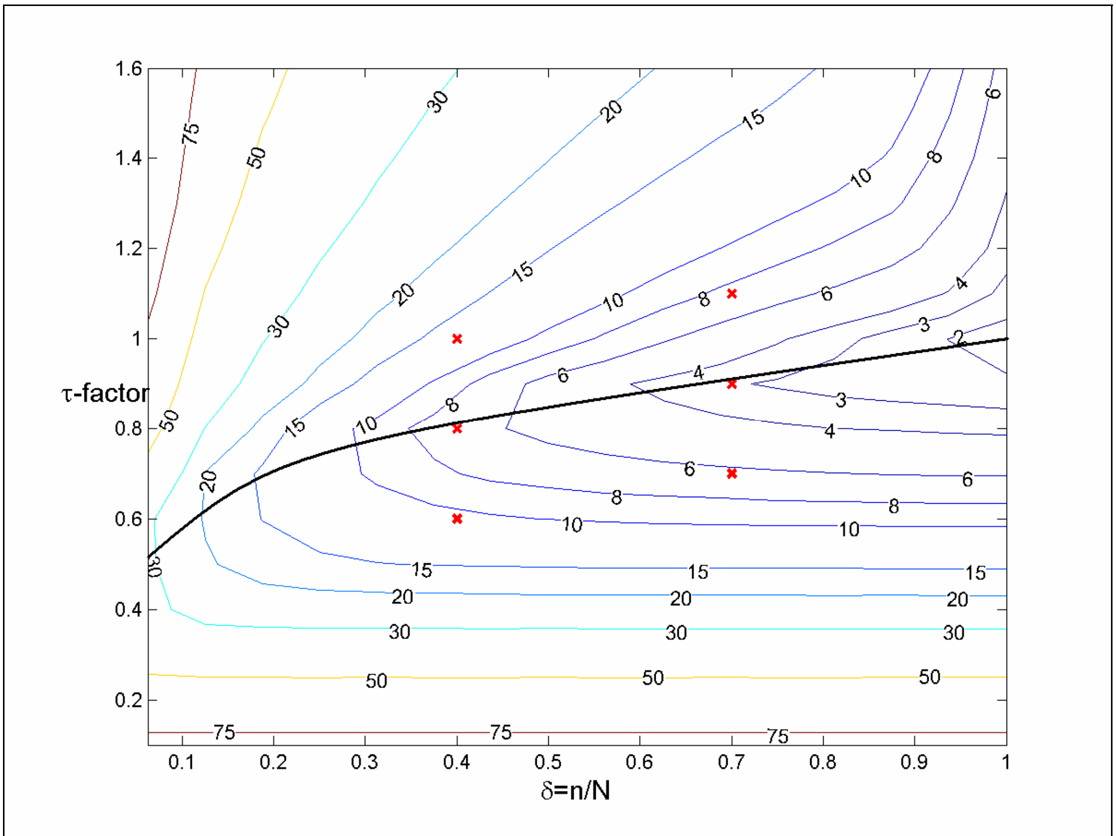


Figure 6.2: Average RMSE for l_1-projection

Following the optimal tuning curve through the parameter space from right to left, we see that the solution RMSE gradually degrades as the undersampling ratio is decreased. In other words, we would expect a controlled degradation in reconstruction accuracy as the number of samples is reduced.

To illustrate these observations, Figure 6.3 gives six reconstructed images corresponding to the points marked with red crosses on the RMSE plot. Two different undersampling ratios (δ=0.4 and δ=0.7) have been selected, and for each we provide an example at optimal tuning, below-optimal tuning and above-optimal tuning.

The plot of average running time (see Figure 6.4) is particular promising in that running times are everywhere under 5 seconds. The l_1-projection algorithm used here is a gradient projection algorithm in which the most expensive operations are matrix-vector products. Such an approach is likely to compare favourably in terms of computational efficiency with many other CS reconstruction algorithms, especially interior point methods or those which employ computationally expensive projection steps. The same comment would also apply to NIHT.

A striking feature of Figure 6.4 is how the running time increases as one approaches the optimal tuning curve. This points to another possible trade-off: reconstruction accuracy versus running time.

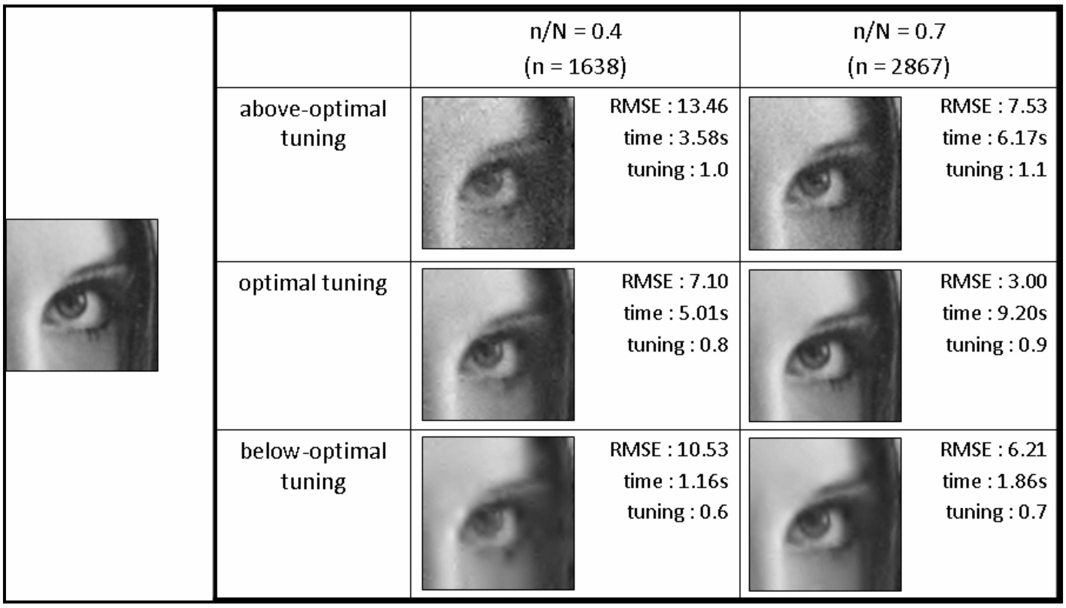


Figure 6.3: Example reconstructed images using l_1-projection

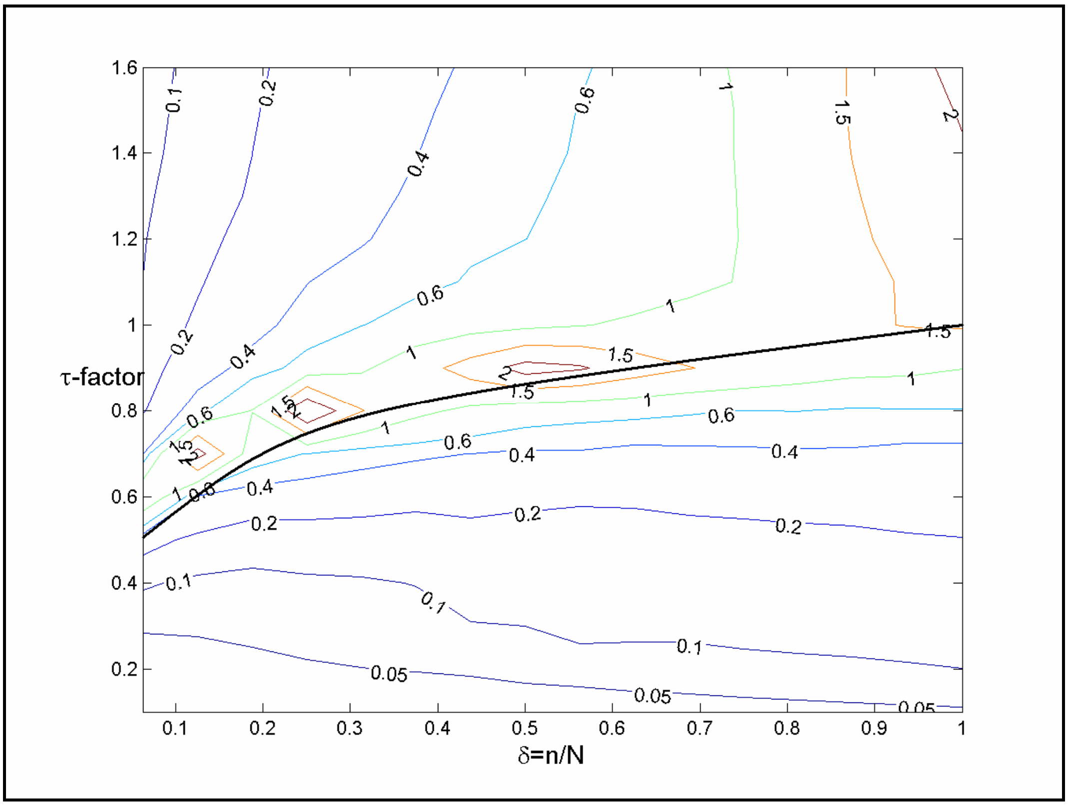


Figure 6.4: Average running time (s) for l_1-projection

6.3 Results for different algorithms

It is instructive to compare the performance of each of our three algorithms on the same test image, sampling scheme and sparsifying transform that were used in Section 6.2. In this regard, we ran multiple-trial tests equivalent to the one described in Section 6.2 for both the other two algorithms: NIHT and iterative tree thresholding. Plots of average RMSE and average running time were obtained and are displayed in Figures 6.5 and 6.6 (NIHT), and in Figures 6.7 and 6.8 (iterative tree thresholding) .

(a) NIHT

For NIHT, the tuning parameter is the sparsity k, which we here normalize to $\rho = k/n$ which gives the sparsity as a fraction of the number of samples. The optimal tuning curve is superimposed as the black curve, demonstrating that the same 'optimal tuning' behaviour persists for NIHT as well. Comparing Figure 6.5 with Figure 6.2, we observe that the optimal RMSE for NIHT for a given value of δ is generally slightly higher than the corresponding value for l_1-projection, suggesting that l_1-projection has slightly better reconstruction accuracy than NIHT in terms of the RMSE metric.

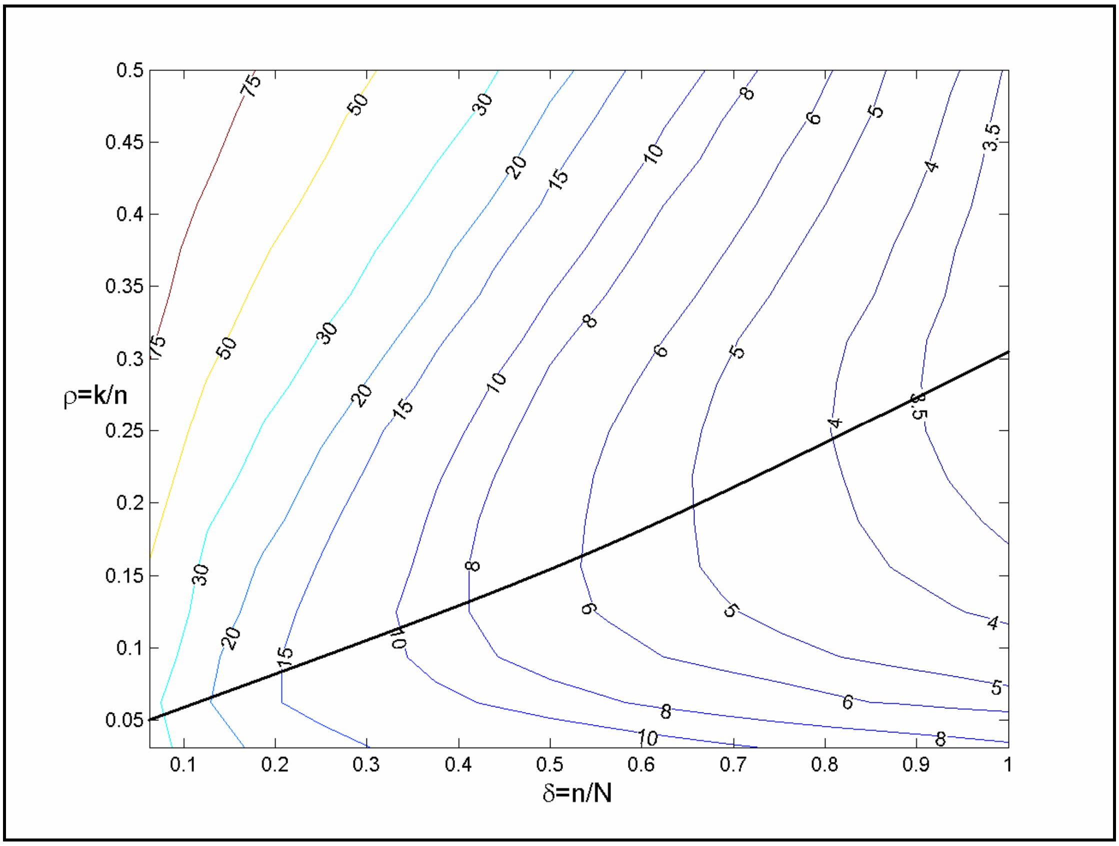


Figure 6.5: Average RMSE for NIHT

Figure 6.6 shows that the running times for NIHT behave somewhat differently from l_1-projection in that running time continues to increase as the tuning parameter (sparsity) is increased, even above the optimal tuning curve. Running times for NIHT are generally higher than for l_1-projection, with the difference being especially marked for large values of δ. However, the goal of Compressed Sensing is to achieve small values of δ, and in this regime the difference in running time is small.

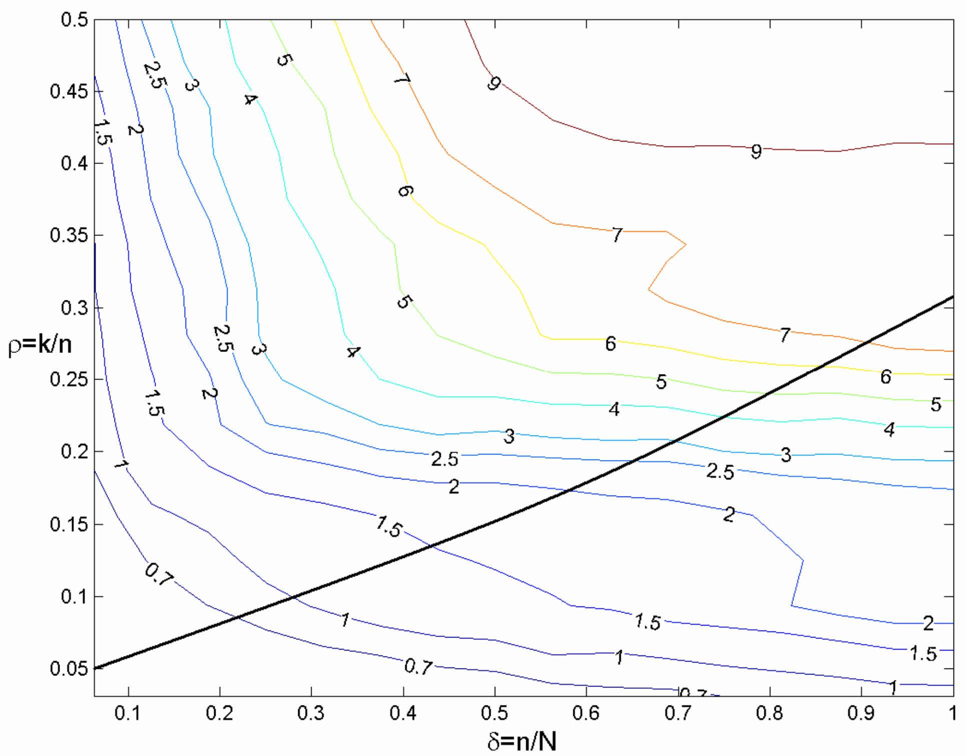


Figure 6.6: Average running time (s) for NIHT

(b) Iterative tree thresholding

Due to longer running times, the experiment for iterative tree thresholding was limited to 10 trials. We see in Figure 6.7 that the tree-based approach gives an improvement when compared to NIHT (the algorithm on which it is based), but it does not in fact give an improvement over l_1-projection in terms of the RMSE metric. However, it may be that the lena image does not possess sufficient tree structure in its wavelet coefficients for a benefit to be observed, or equally that the image size is too small for the tree-based approach to offer an advantage.

The reconstruction times (Figure 6.8) are significantly increased for the tree-based approach except for very small values of δ, reflecting the fact that the projection step is more computationally demanding.

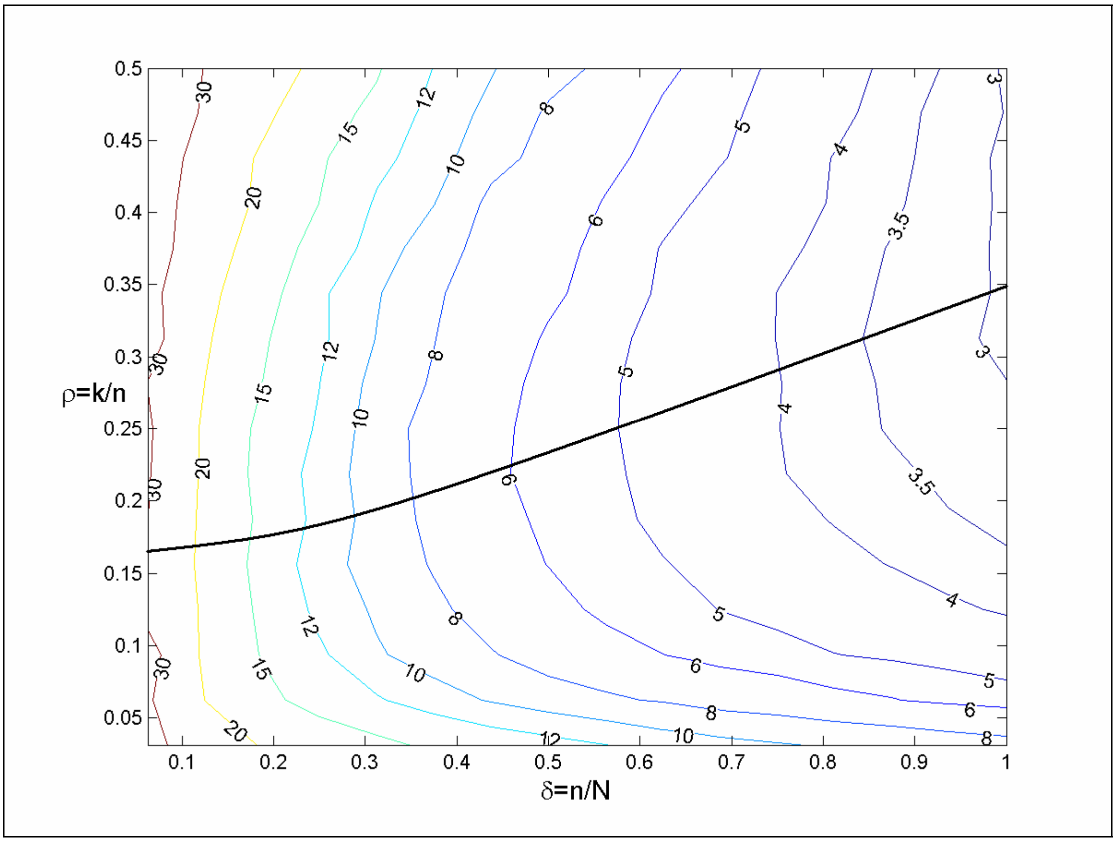


Figure 6.7: Average RMSE for iterative tree thresholding

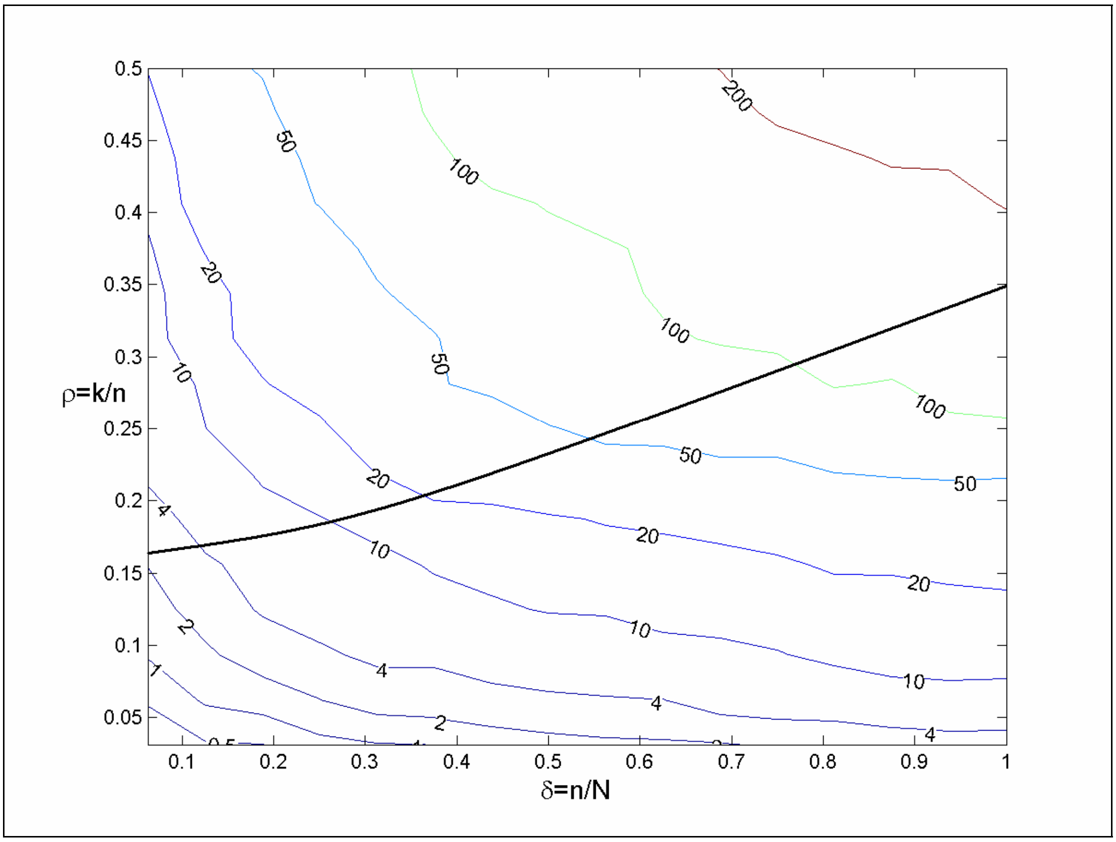


Figure 6.8: Average running time (s) for iterative tree thresholding

Though the tree-based approach does not offer an advantage for the lena test image, we can however give an example of an image where it does. Figure 6.9 shows an image (original and reconstruction by both l_1-projection and iterative tree thresholding) generated by the CAMEOSIM package which has simulated a short-wave infra-red image of a vehicle against an uncluttered background, a scene which is the subject of extensive experimentation in later sections. As well as having a slightly higher dimension (128x128), the image also has large regions that are approximately smooth, and so one would think it a good candidate for a wavelet connected tree model. Reconstruction in both cases is with ±1 sampling and Haar wavelets, with undersampling ratio δ =0.15. Figure 6.9 indicates that a result with a lower RMSE can be achieved using iterative tree thresholding. In particular, the reconstructed image appears to be sharper for the tree-based approach – see Section 6.10 for a comparison of the reconstruction algorithms in terms of signal-to-noise preservation.

original	l_1-projection	iterative tree thresholding
	 RMSE : 9.13 t : 72.57s	 RMSE : 7.40 t : 103.92s

Figure 6.9: A favourable example for iterative tree thresholding

6.4 Comparison with classical compression

Compressed Sensing may be viewed as an alternative method of compression: instead of acquiring a full set of samples and subsequently compressing, the CS approach is to acquire compressed samples in the first place. It is natural therefore to compare the results of the CS camera model with other classical compression strategies. In particular, let us consider the 'optimal' compression strategy of keeping only the k largest (in magnitude) transform coefficients while setting the rest to zero. Figure 6.10 shows a plot of compression error RMSE against compression ratio (k/N) when this optimal strategy is applied in the 9-7 wavelet domain to the 64x64 lena test image used in Sections 6.2 and 6.3.

Using this data, we may compare the RMSE of a reconstructed solution with the optimal RMSE achievable for the same sparsity level. We choose to use NIHT for this comparison since the tuning parameter k is explicitly the sparsity, and therefore directly related to the compression ratio. Figure 6.11 shows a plot of the RMSE amplification factor, *i.e.*

$$AMP = \frac{\sqrt{\frac{1}{N}\|\hat{x}-x\|_2^2}}{\sqrt{\frac{1}{N}\|x_k - x\|_2^2}},$$

where x_k is the optimal k-sparse compression in the 9-7 wavelet domain, for the 64x64 lena image, and where $\hat{x}$ is the k-sparse solution obtained using NIHT.

Superimposed as a black curve is the same optimal tuning curve from the previous section. In the region of interest, *i.e.* close to the optimal tuning curve, we see that the amplification factor is well-behaved, typically taking a value around 1.5. In fact, below the optimal tuning curve, the amplification factor is always less than 2. The interpretation is that, even though we are taking potentially far fewer measurements, it is possible to recover compressed solutions whose compression error is only a small factor worse than the best conceivable compression.

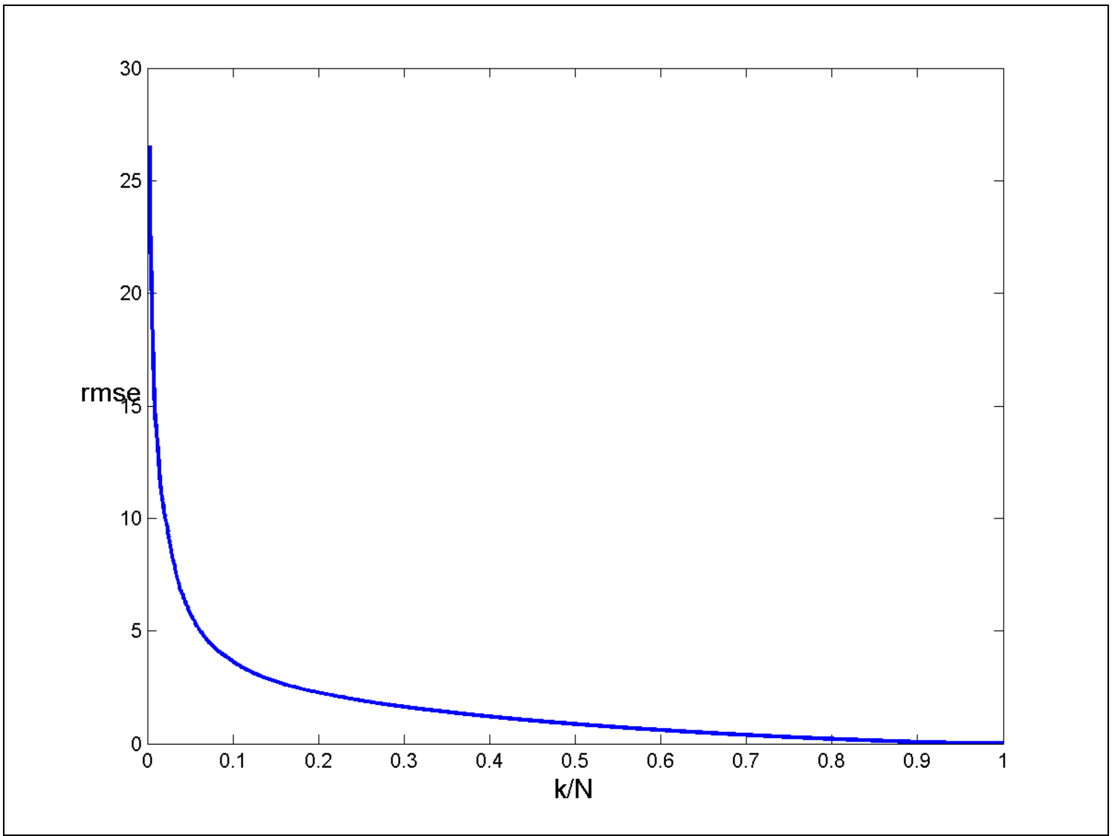


Figure 6.10: ‘Optimal’ RMSE against compression ratio for lena with 9-7 wavelets

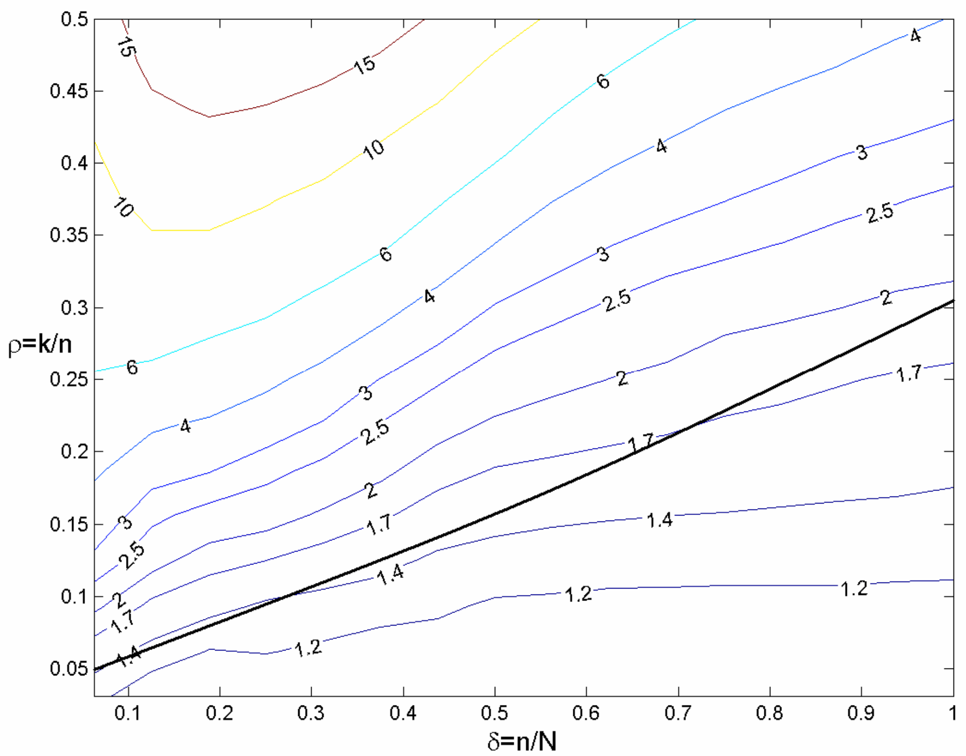


Figure 6.11: Average RMSE amplification factor for NIHT

6.5 Exactly sparse images: further insight

Further insight into the behaviour of our algorithms can be gained by considering what happens when our original image is exactly sparse in the given transform domain (so that the classical compression error is zero). As is well documented in CS theory [5], in the exactly sparse case, there exists a region of the parameter space within which the original image can be exactly reconstructed with overwhelming probability for large problem sizes. We may identify this region for NIHT by first compressing the 64x64 lena test image so as to be k-sparse in the 9-7 wavelet domain, where k is the sparsity tuning parameter for NIHT. We then run 100 trials on a mesh as in previous sections and record the proportion of successful exact reconstructions, which gives an estimate of exact reconstruction probability. Figure 6.12 shows the result of this experiment in the form of a plot of success probability. A clear phase transition phenomenon can be observed, with a quite sharp transition between guaranteed failure (probability 0) and guaranteed success (probability 1). This plot makes plain the region of the parameter space where the NIHT algorithm is able to function, and so provides the underlying explanation for the 'optimal tuning' behaviour reported in Sections 6.2 and 6.3. Such a phase transition phenomenon would also be expected for the other two algorithms, l_1-projection and tree-based thresholding.

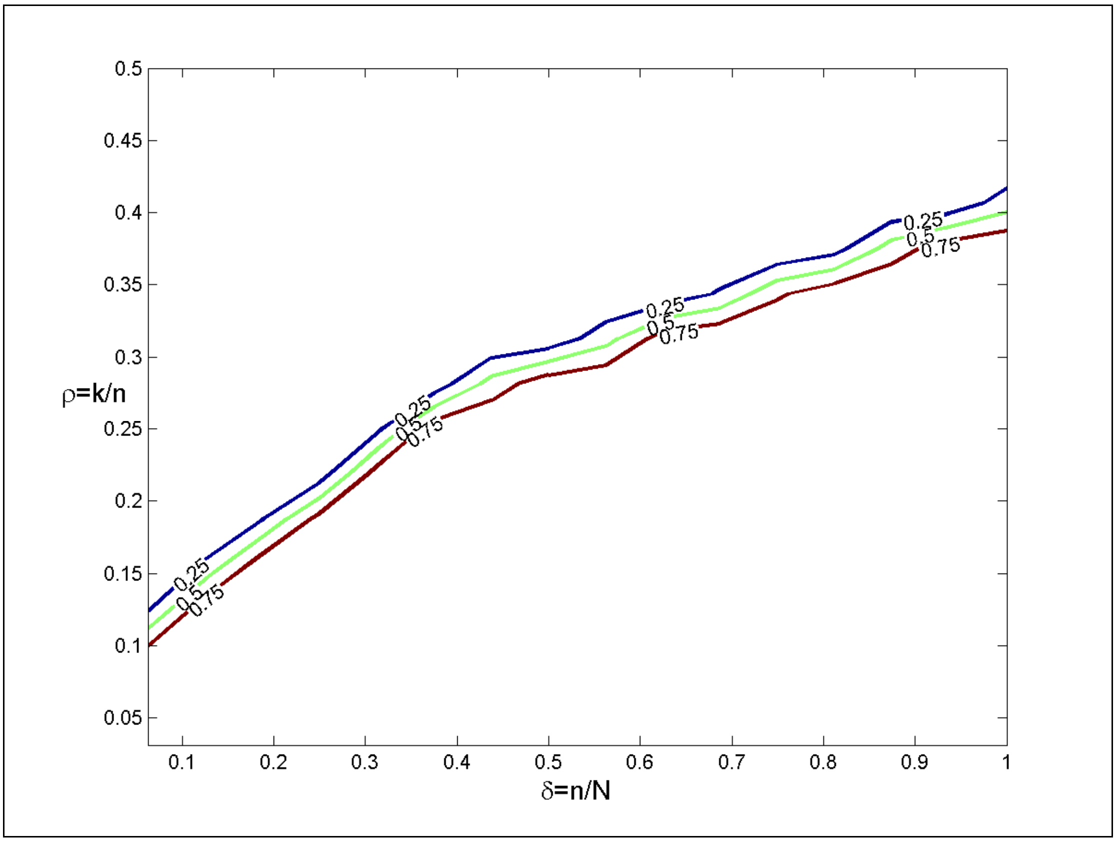


Figure 6.12: Phase transition for reconstruction of exactly sparse images using NIHT

6.6 Robustness to noise

In sections 6.1 to 6.5, we worked under the simplifying assumption that there is no noise in the sampling process. In order to now capture the inherent imprecision of any imaging device, we now examine the impact of the injection of additive Gaussian white noise e such that $y = \Phi x + e$ and each entry of e is distributed as $N(0, \sigma^2)$. Figure 6.13 shows a plot of average RMSE when Gaussian sampling noise with standard deviation $\sigma = 2.5$ (approximately 1% of the dynamic range of the image) is built into the model. All other model options are the same as for the equivalent noiseless plot in Figure 6.2 – to avoid repetition we refer the reader to section 6.2 for the details. Comparing Figure 6.13 with Figure 6.2, we see in general a small increase in average RMSE across the parameter space: the contours move slightly to the right when noise is added. However the change is small, suggesting that the model is robust to noise. Figure 6.14 gives the equivalent plot for a higher level of noise: $\sigma = 25$. As would be expected, we same a more considerable increase in the RMSE throughout the parameter space, though accuracy of reconstruction is not lost entirely.

Another way to examine the effect of the presence of noise on the system is to ask by what factor the RMSE of the noise is amplified in the reconstructed solution. Identifying this factor, however, is complicated by the fact that we have reconstruction error due not only to the noise, but also due to natural compression loss. In Section 6.4, we investigated the reconstruction error as an amplification of the compression error. In order to isolate the contribution from the noise alone, we designed a slightly different experiment for NIHT in which we first compress the 64x64 lena test image so as to be k-sparse in the 9-7 wavelet domain, where k is the sparsity tuning parameter for NIHT (as in Section 6.5). CS theory tells us that, in the absence of noise, such an image can be exactly reconstructed with high probability within a certain region of the phase space. Any reconstruction inaccuracy can therefore be fairly attributed to the presence of noise in the sampling process. Given a reconstructed solution $\hat{x}$, we therefore define the noise amplification factor to be the solution RMSE divided by the RMSE of the noise vector e, *i.e.*

$$AMP2 = \frac{\sqrt{\frac{1}{N}\|\hat{x} - x\|_2^2}}{\sqrt{\frac{1}{n}\|e\|_2^2}}.$$

Figure 6.15 gives the noise amplification factor plot for $\sigma = 2.5$. Superimposed over the plot is the optimal tuning curve from Section 6.3. We see that in the region of the optimal tuning curve, the noise amplification factor is well-behaved and in fact never exceeds 2. In fact, the amplification factor can be significantly lower even than this, especially for small δ. In other words, the CS reconstruction can even have a denoising effect, actually reducing the RMSE. An explanation for this behaviour is that, since the CS reconstruction obtains a good approximation to the best classical compression in which small coefficients are thresholded out, the result is that small noise contributions can be thresholded out as well.

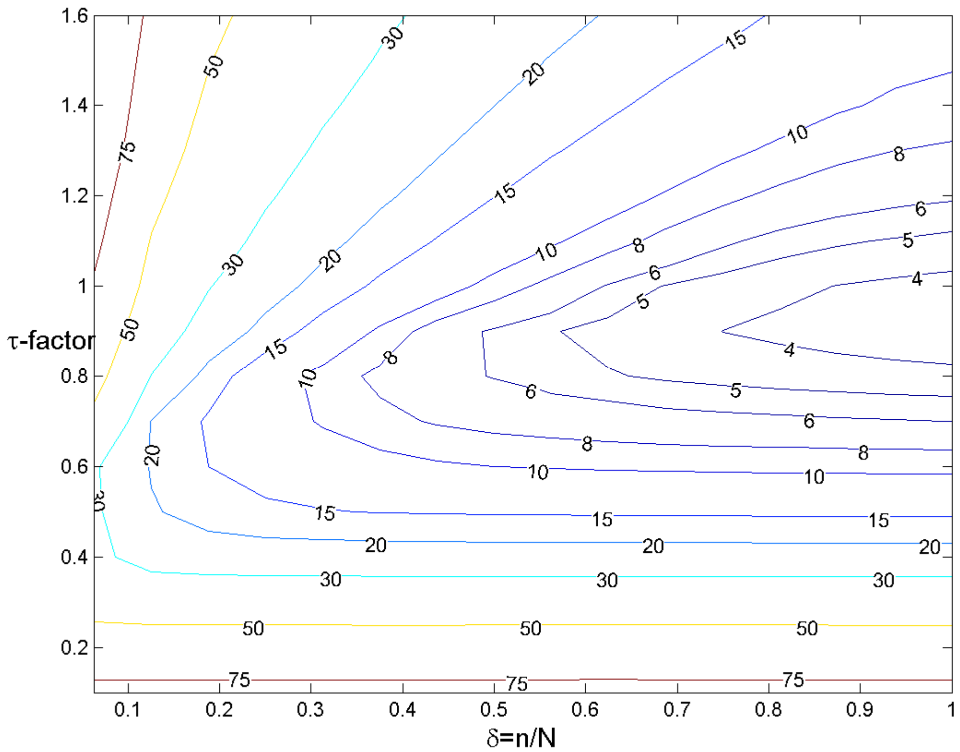


Figure 6.13: Average RMSE for Gaussian sampling noise with $\sigma = 2.5$

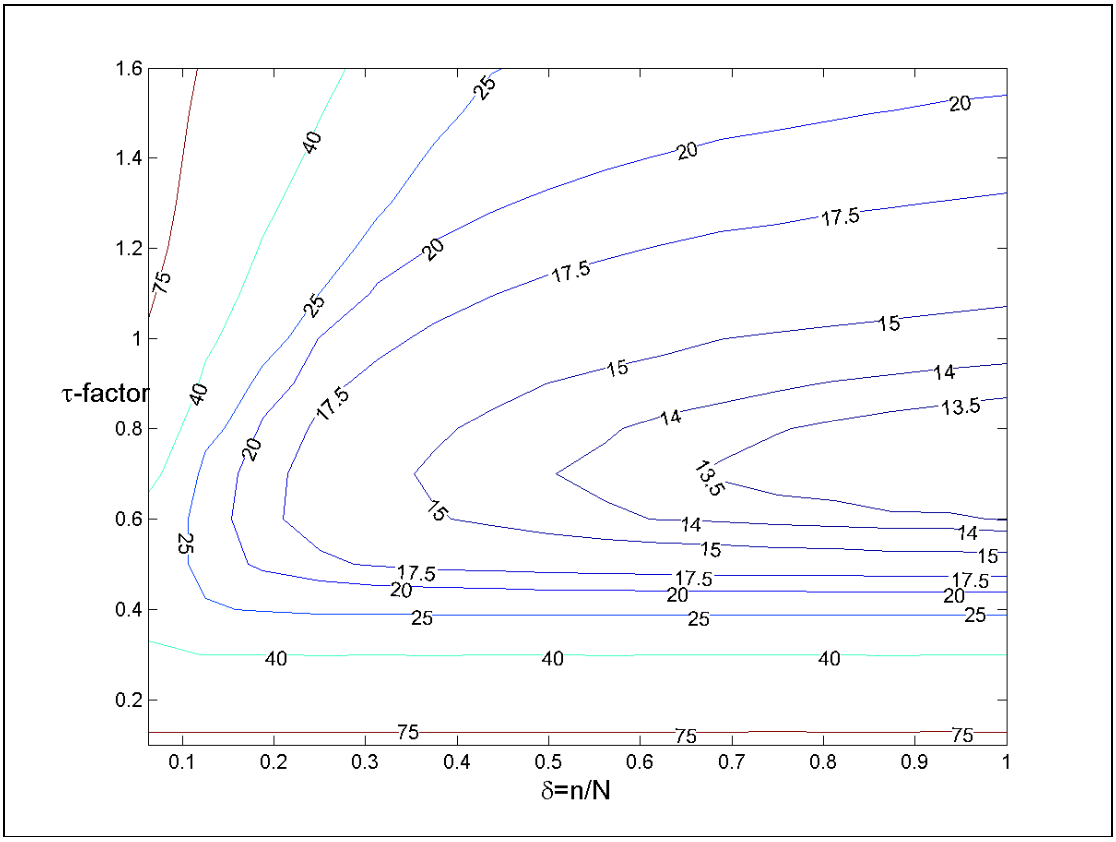


Figure 6.14: Average RMSE for Gaussian sampling noise with $\sigma = 25$

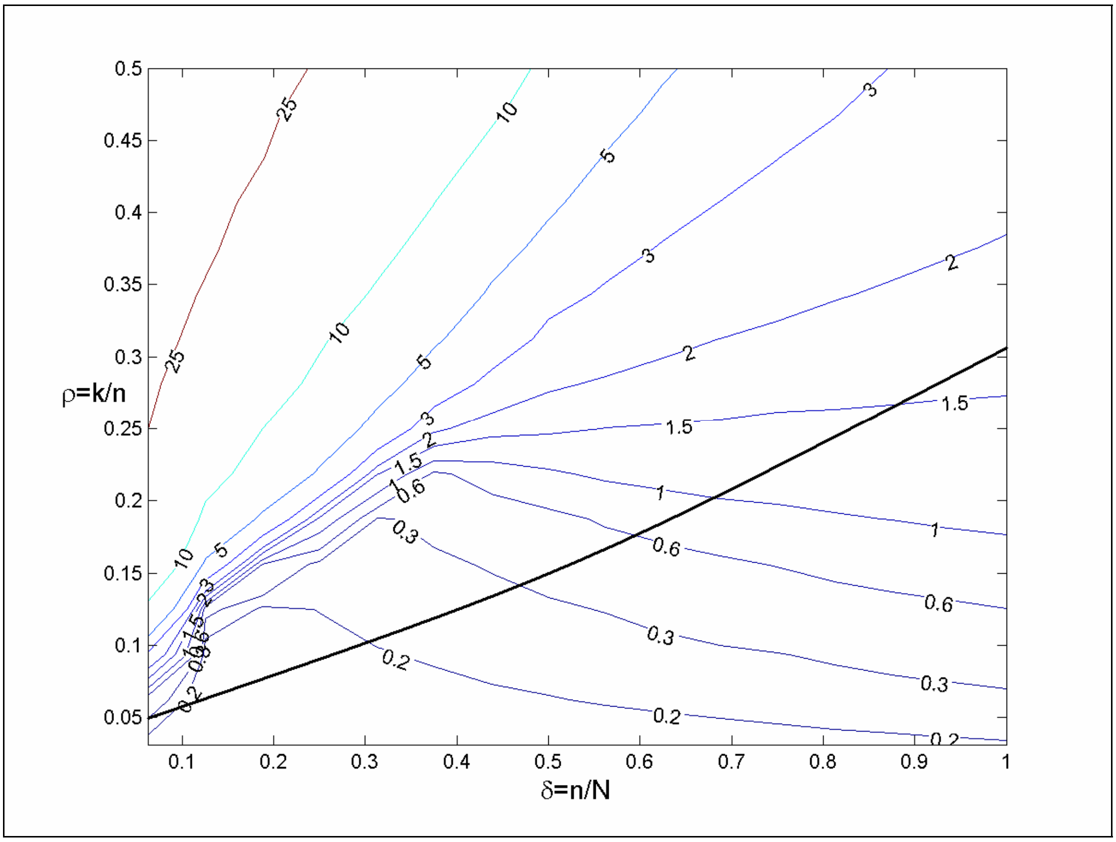


Figure 6.15: Noise amplification factor for NIHT

6.7 Discussion of sampling schemes

This section addresses a hitherto unresolved issue: the results presented in Sections 6.1 to 6.6 are for the ±1 sampling scheme, whereas the camera design outlined in a previous section is modelled naturally by a Bernoulli matrix consisting of random zeros and ones. Why the discrepancy?

The Bernoulli sampling option was implemented in the model and experimentation along the lines already outlined for ±1 sampling was carried out. What these experiments revealed is that all three algorithms performed significantly poorer for Bernoulli sampling. RMSEs of reconstructed solutions were higher, the behaviour of the algorithms more erratic, and sometimes the algorithms cycled and didn't even converge. The underlying reason for this appears to be that Bernoulli matrices are badly conditioned – in particular, they have one eigenvalue that is much larger than the rest. Such a problem is generally a challenge for gradient projection methods, which are sensitive to the conditioning of the matrices involved.

The solution to this problem was the re-introduction of a backtracking step into the l_1-projection algorithm. Provided this step is included, it was found that l_1-projection gave similar reconstruction properties for Bernoulli matrices as for ±1 matrices. Running times were increased however, as the backtracking step significantly slows down the progress of the algorithm. While we were able to demonstrate that it is possible to reconstruct from Bernoulli sampling, it still remains preferable from the point of view of reconstruction to work with ±1 matrices.

The other sampling option available in the model is Gaussian sampling. In this case, the reconstruction algorithms were observed to behave in a similar manner to when ±1 sampling is used. This is also to be expected from a theoretical point of view: both ±1 and Gaussian matrices have entries that are centred about zero, which leads to them being well conditioned.

6.8 Comparison of wavebands using CAMEOSIM

While the single-pixel camera was first studied in the context of visible light, it is for other wavebands, such as infra-red, that it is more likely to find an application. Sampling light outside the visible spectrum often requires the use of more exotic detectors which may be either expensive or bulky. In such cases, there may be considerable incentives to explore the option of moving from many photon detectors to a single detector or 'pixel'. One possible application of the CS single-pixel approach is in multi-spectral imaging in which the light field would be simultaneously directed (by means of an appropriate DMD configuration) onto a number of detectors corresponding to different wavebands. It is interesting therefore to address the issue of how reconstruction of the same scene may vary across different bands.

The CAMEOSIM package was used to simulate a 128x128 image of a vehicle against an uncluttered background in different wavebands. As well as visible light, images corresponding to short-wave (SW), medium-wave (MW) and long-wave (LW) infra-red bands were obtained, along with a range image. Using ± 1 sampling, the l_1-projection algorithm and Haar wavelets (which are especially well-suited to angular objects such as the vehicle in question), and choosing an undersampling ratio of δ=0.15, the tuning parameter was optimized for each band, and the ‘optimal’ solutions found in terms of RMSE. The tests referred in Sections 6.8 to 6.10 were conducted on a standard HP xw6400 desktop, which is why the running times are generally longer than those reported in other sections. The recovered images, along with the originals, are shown in Figure 6.16.

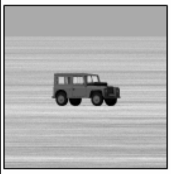
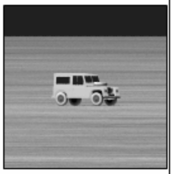
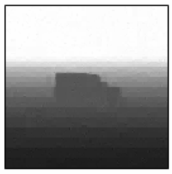
	visible	SW	MW	LW	range
original					
reconstruction					
data	RMSE : 12.94 t : 29.71s tuning : 0.55	RMSE : 9.13 t : 72.57s tuning : 0.8	RMSE : 11.37 t : 54.24s tuning : 0.6	RMSE : 8.60 t : 52.60s tuning : 0.7	RMSE : 3.87 t : 22.65s tuning : 0.85

Figure 6.16: Example reconstructions in different wavebands

Figure 6.16 shows some variation in reconstruction accuracy between the bands. In particular, the range image has a markedly lower RMSE than the light intensity bands. Also, the reconstruction for SW and LW is better than for visible and MW. There are two contributory factors to these differences. Firstly, the images in the different bands vary in terms of their natural compressibility. For example, the background for visible light, though relatively uncluttered, is more complex than for SW. Since CS algorithms do not make use of *a priori* information about the location of objects of interest, the reconstruction algorithm will ‘use up’ some its allocation of wavelet coefficients in capturing the background, leaving less coefficients available for the vehicle. On the other hand, the range image is less complex and therefore more naturally compressible, which is why a more accurate recovery is possible. Secondly, the images vary in terms of the contrast between object of interest and background. The best reconstruction (discounting the range image) is obtained for LW, due in part to the marked contrast between the vehicle and its background.

More generally, these two issues capture the image-dependent nature of CS reconstruction performance. In particular, the CS approach is predicated on the assumption that the image is naturally compressible, and therefore its effectiveness is limited by how well a given image satisfies this assumption.

6.9 Effect of foreground and background clutter

In reality, objects of interest are likely to be surrounded by clutter, either in the foreground or background, and a crucial issue for any imaging approach is the dependency of reconstruction accuracy upon the level of such clutter. To investigate this, we obtained from CAMEOSIM simulated images of the same scene, except with background clutter added in the form of trees. Figure 6.17 shows the reconstructed images, with and without trees, for the three infra-red wavebands. The same model options were again chosen: ± 1 sampling, the l_1-projection algorithm, Haar wavelets, and undersampling ratio 0.15.

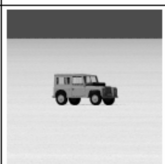
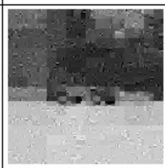
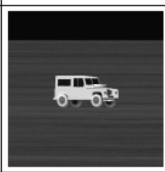
	without trees			with trees		
	data	original	recovered	recovered	original	data
SW	RMSE : 9.13 t : 72.57s tuning : 0.8					RMSE : 26.87 t : 26.86s tuning : 0.4
MW	RMSE : 11.37 t : 54.24s tuning : 0.6					RMSE : 18.78 t : 55.14s tuning : 0.5
LW	RMSE : 8.60 t : 52.60s tuning : 0.7					RMSE : 15.97 t : 34.33s tuning : 0.45

Figure 6.17: Example reconstructions with and without background clutter

It may be observed that the impact of additional clutter is much greater for SW than for MW. For MW the vehicle itself is reconstructed to a similar degree of accuracy as before, whereas for SW the reconstruction is so poor that the top of the vehicle cannot be distinguished from the background. Part of the reason for this is again contrast: for SW the light-coloured background is replaced with a background of intensity much closer to the vehicle. However, the most complete explanation of the difference is in terms of how the natural compressibility of the images changes when the trees are added. Figure 6.18 plots the RMSE obtained from the 'optimal' compression of an image (thresholding out the smallest wavelet coefficients) against compression ratio. The plots entirely mirror the image results: without trees the SW image is more naturally compressible, whereas when trees are added the SW image actually becomes the least naturally compressible.

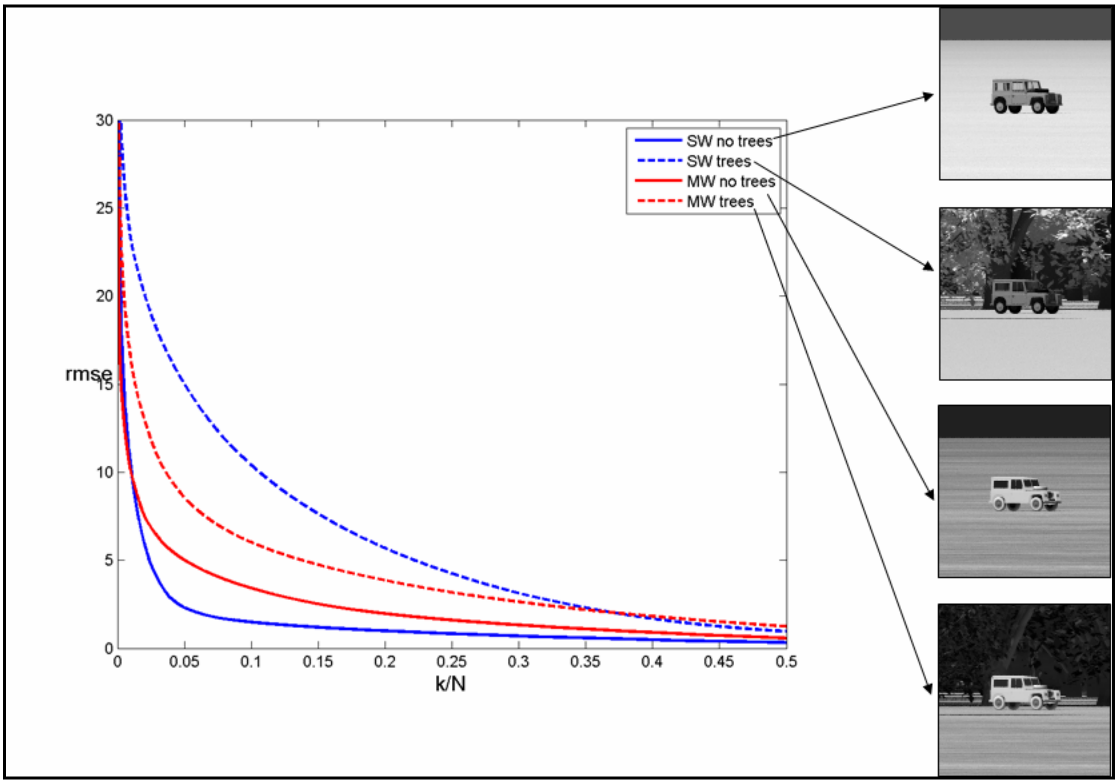


Figure 6.18: Compressibility of images in different bands

6.10 Peak signal-to-noise ratio and debiasing

Section 6.9 demonstrates how the ability of a CS algorithm to reconstruct objects of interest in the presence of background clutter depends upon the compressibility of the background and also upon the contrast between the object and the background. Attached to this, another natural question to ask for a potential surveillance application is whether the CS approach can be used to carry out remote sensing.

We therefore designed an experiment to explore this issue. CAMEOSIM was used to simulate images of the scene featuring a vehicle against a background of trees used in Section 6.9, viewed from different horizontal distances of 1.35km (as in the previous section), 5km and 10km. A particularly appealing choice from the point of view of SNR was to use the LW infra-red images, since the vehicle appears as an object of high intensity against a low intensity background. We may define a peak signal-to-noise ratio of the original image to be the average intensity of the vehicle divided by the average intensity of the background. By recording a mask corresponding to the location of the vehicle in the original image, we may then obtain a measure of the PSNR of the reconstructed image as the average intensity of the masked region divided by the average intensity of the unmasked region. We are interested in whether the PSNR is preserved by a particular recovery algorithm, and so in this regard we define the 'PSNR factor' metric as the PSNR of the reconstructed image given as a percentage of the PSNR of the original image. A PSNR factor of 100% indicates that the contrast between object and background is entirely preserved; a PSNR factor significantly below 100% indicates that significant contrast is lost in the sampling and reconstruction process.

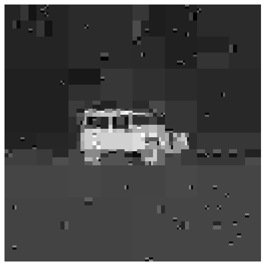
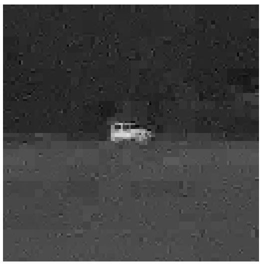
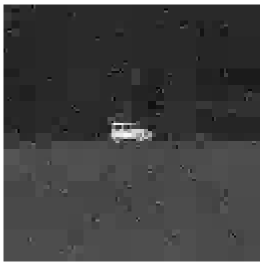
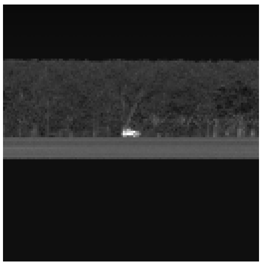
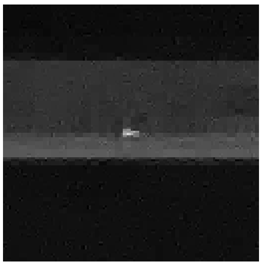
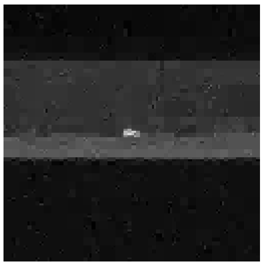
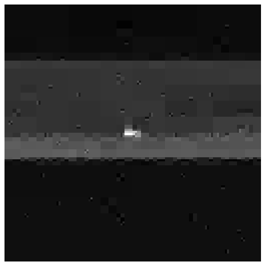
	Original	l_1	l_1-debias	NIHT
1.35km		 PSNR factor : 89% RMSE : 16.0630 t : 37.2s	 PSNR factor : 90% RMSE : 16.7401 t : 15.0s	 PSNR factor : 95% RMSE : 17.4469 t : 33.3s
5km		 PSNR factor : 79% RMSE : 10.7793 t : 20.5s	 PSNR factor : 92% RMSE : 10.4922 t : 9.8s	 PSNR factor : 94% RMSE : 10.9946 t : 32.0s
10km		 PSNR factor : 79% RMSE : 6.8040 t : 20.9s	 PSNR factor : 88% RMSE : 6.8157 t : 17.4s	 PSNR factor : 97% RMSE : 6.6104 t : 62.0s

Figure 6.19: Example LW reconstructions at varying distances

Figure 6.19 shows the reconstructed image for the three distances and for three different algorithms: l_1-projection without debiasing, l_1-projection with debiasing, and NIHT. As in Sections 6.8 and 6.9, the model options are ±1 sampling with δ =0.15 and Haar wavelets. It can be observed that, even though l_1-projection tends to give better overall reconstruction accuracy in terms of RMSE, NIHT gives better results when the metric of PSNR factor is used. The difference between the reconstructions between the different algorithms is observable: the vehicle appears brighter and stands out more noticeably from the background in the images reconstructed using

NIHT than those reconstructed using l_1-projection. The reason for this is the tendency of l_1-projection to shrink the transform coefficients.

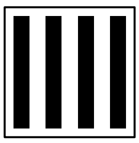
This effect can be mitigated by including a debiasing step in the l_1-projection algorithm. There is clear evidence here that the addition of debiasing tends to increase the PSNR factor. It is interesting to note, however, that the optimal tuning parameter is lower when debiasing is used, and the image as a whole often looks less convincing, even though the PSNR has been enhanced. This suggests that the decision as to whether to include a debiasing step would depend upon which performance metric is most desirable.

Overall, however, the conclusion from these tests is promising. Especially when NIHT is used as the reconstruction algorithm, the PSNR is almost entirely preserved, even at remote distances of 10km. The underlying explanation for this success is that CS reconstruction algorithms are able to give a good approximation to the 'optimally compressed' image in the particular transform basis. In our particular case, the sparsifying transform is the Haar wavelet basis, and it is well-known that wavelets in general are particularly effective at localizing objects in a scale-independent way. It is the use of a wavelet basis therefore that is the crucial reason why the CS camera model is able to preserve PSNR at distance. If, for example, the DCT had been used as the sparsifying transform, one would expect the PSNR results to be much poorer.

It is worth also mentioning here an experiment that was conducted as an aside. Typically we need to assume that our image is compressible in some transform domain. However, the images considered in this section feature a small object of high intensity against a background of low intensity. By subtracting off the mean intensity of the background, it is possible to model the image as being compressible in the image domain, *i.e.* the identity as the sparsifying transform. Because such a strategy effectively sets the background to zero, compression in the image domain actually magnifies the PSNR. For example, using NIHT with 100 pixels in the image domain, a PSNR factor of 29,000% can be achieved! It is worth bearing in mind, then, that very simple images consisting of small objects of high intensity against a low intensity background can be most effectively compressed in the image domain. For this reason, the option to choose the identity as the sparsifying transform is also built into our MATLAB model.

6.11 Frequency response

One feature of interest for any compression strategy is the frequency response: Does the method keep the low frequencies and discard the higher ones? Is the compression error evenly spread across the frequency domain, or is there some observable bias? The answers to these questions give important information about the ability of a given technique to retain detail. To investigate this, we used for our original image a 64x64 structured pattern consisting of four vertical bars. In order to capture the frequency



behaviour, we used the DCT transform and compared the result of classical compression with the result of reconstruction using NIHT, matching the sparsity tuning parameter of NIHT to the compression ratio as was outlined in Section 6.4. The sampling scheme was again taken to be ±1, with the undersampling ratio set to δ =0.5.

Figure 6.20 shows a plot of the DCT of the original uncompressed image. Figures 6.21 to 6.23 are plots of compression error for different sparsity levels (Figure 6.21: k =50; Figure 6.22: k =100; Figure 6.23: k =150). The blue lines show the classical compression error, whereas the red lines give the additional error that occurs from reconstruction by NIHT on top of the classical compression error.

At k =50, both the compression and reconstruction error appear to be well spread-out across the frequency domain. At k =100, we observe that classical compression picks up all the low frequency components, leaving error only in the high frequency components. On the other hand, the additional error resulting from NIHT reconstruction for k =100 appears evenly spread out over the frequency domain, and clusters around the true non-zero DCT coefficients of the original image. By k =150, classical compression completely captures the original signal, indicating that our signal has so much structure that it is in fact exactly sparse in the DCT domain with sparsity less than or equal to 150. The NIHT compression error is small, indicating that NIHT is converging to the exact solution – this additional error could be made arbitrarily small by strengthening the termination criterion. We see again that additional compression error is spread out across the frequency domain, clustered around the true non-zero coefficients, with no observable bias towards low or high frequencies.

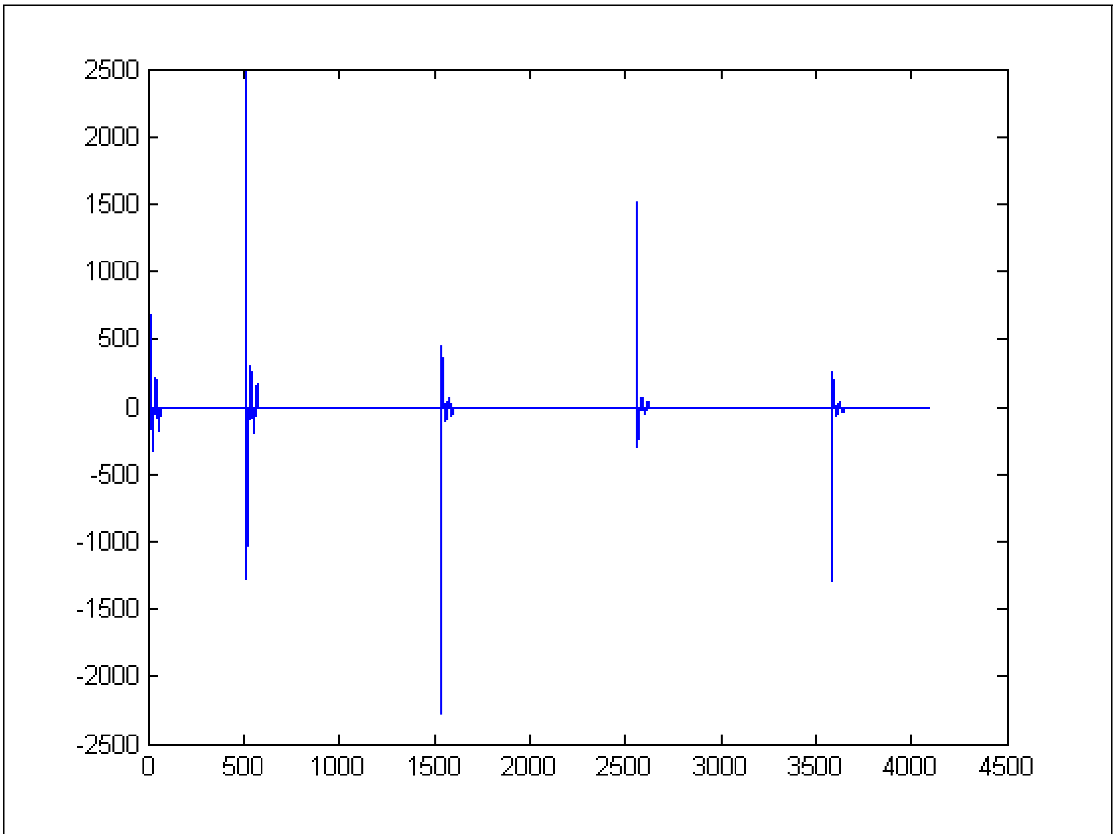


Figure 6.20: DCT of original uncompressed bar pattern

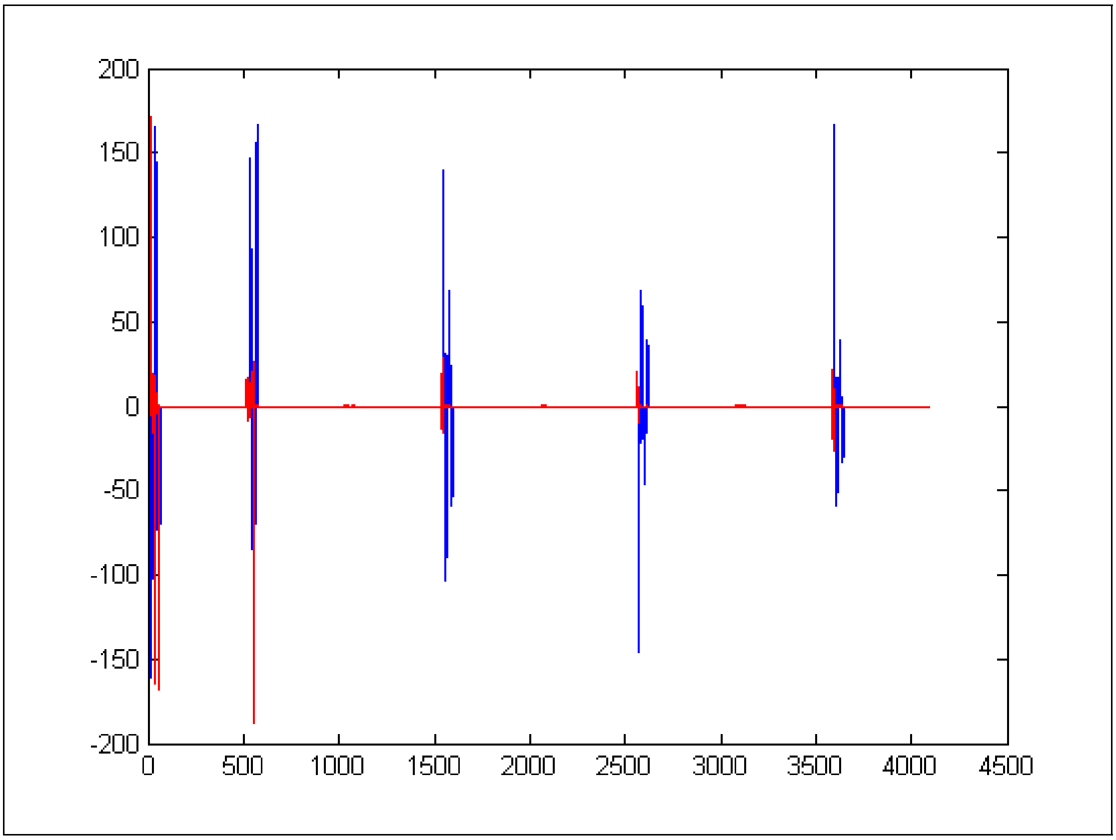


Figure 6.21: Compression/reconstruction error (k=50)

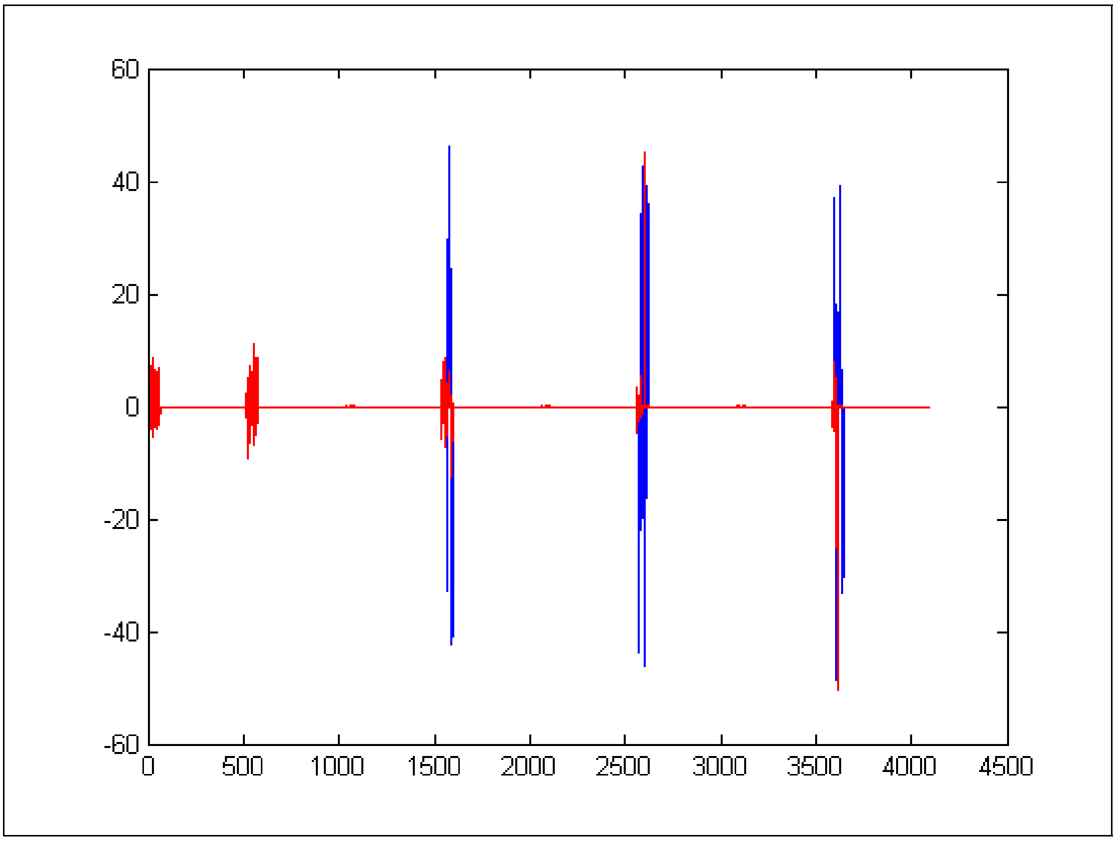


Figure 6.22: Compression/reconstruction error (k=100)

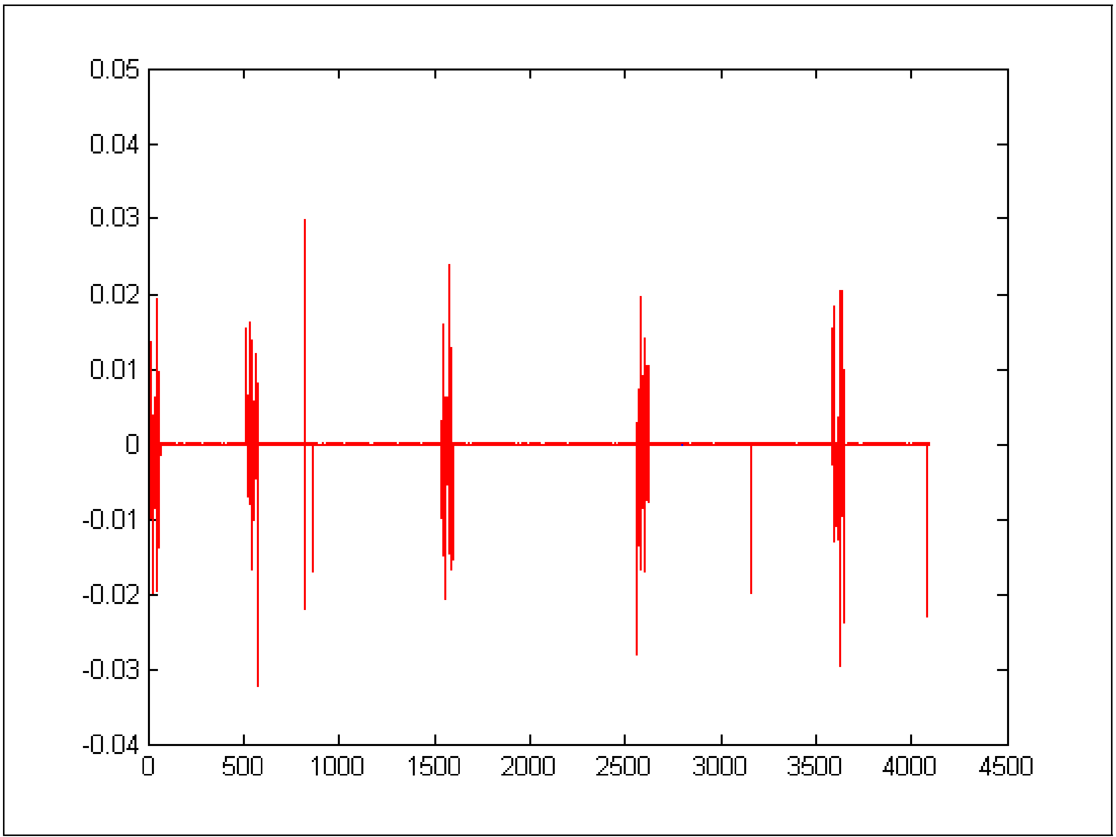


Figure 6.23: Compression/reconstruction error (k=150)

6.12 Extension to 3D moving images

So far in this report we have assumed that the scene of interest is static over the period of the image acquisition. A natural next question to ask is whether these CS techniques can be extended to 3D moving images. This question has already been addressed in the CS literature, see for example [6].

Traditional CS imaging theory considers a series of measurements taken from a single image. However, for a moving scene, each measurement will act on essentially a different image. To be able to leverage traditional CS imaging theory, therefore, we must make the modelling assumption that the image changes slowly over a group of snapshots, which we can then equate to a single video frame. Let us, then, represent an acquired video as a sequence of F frames, each consisting of N pixels, where we take n measurements per frame. Omitting sampling noise from our consideration for simplicity, we may therefore model the acquired measurements as $y_i = \Phi_i x_i$ for $i = 1, \ldots, F$, where each Φ_i is an $n \times N$ sampling matrix.

A basic approach is now to view the problem as F separate CS problems. Selecting Ψ to be some appropriate 2D sparsifying transform, we may then apply any of the reconstruction algorithms introduced in Section 4 to each frame. The option to do frame-by-frame reconstruction of a 3D moving image is built into the model, with the same built-in options for the sampling scheme, reconstruction algorithm and sparsifying transform as were outlined for the 2D case in Section 5. There is the additional option to write the reconstructed video sequence to an AVI file.

However, an alternative approach is possible in which we attempt to exploit the temporal dependence between frames. To do this, we consider the entire video sequence, and represent it as a single vector

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_F \end{bmatrix}$$

of length NF. The sampling process may now be written in the form of a single acquisition as

$$y = \Phi x = \begin{bmatrix} \Phi_1 & 0 & \dots & 0 \\ 0 & \Phi_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \Phi_F \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_F \end{bmatrix}.$$

We may now choose Ψ to be a 3D sparsifying transform and apply any of the reconstruction algorithms introduced in Section 4 to reconstruct the entire video sequence.

The option to do joint reconstruction of a 3D moving image is also built into the model with the same options for the sampling scheme and reconstruction algorithm. Concerning the transform option, a number of orthogonal 3D tensor wavelet transforms are available: Haar wavelets, Daubechies-D4 wavelets and Daubechies-D8 wavelets. The D4 and D8 wavelets are related to the 5-3 and 9-7 biorthogonal wavelets available for 2D reconstruction. Each of these transforms in fact requires the inputted data to be an exact data cube consisting of d frames of a $d \times d$ image, *i.e.* $N = d^2$ and $F = d$. The reconstructed video sequence can also be written to an AVI file.

CAMEOSIM was again used to simulate a moving image, based upon the image of the vehicle against a clutter-free background introduced in Section 6.8. The temporal dependence between the slides is well-structured: the vehicle moves from the left to right of the shot at constant speed. The video sequence consists of 64 frames where each frame is a 64x64 pixel image, giving a 64x64x64 datacube. Both frame-by-frame and joint reconstructions were performed, using the ±1 sampling scheme, the l_1-projection algorithm and Haar wavelets (either 2D or 3D). The undersampling ratio was set to $\delta = 0.4$, and the tuning parameter was optimized in terms of the solution RMSE of the entire datacube.

Results for three selected frames are shown in Figure 6.24. There is clear evidence from the RMSEs that the joint reconstruction gives a more accurate reconstruction than the frame-by-frame reconstruction. Indeed, the difference may be observed visually: more detail, both of the vehicle and the background, is reconstructed using the joint approach. There are two reasons for this improvement. Firstly, the joint approach takes into account the temporal dependence between the frames. 3D wavelets therefore represent a more efficient means of capturing the information than a collection of 2D transforms. This means that the datacube is more compressible in the 3D wavelet basis and, as we have seen, compressibility is the key image-dependent factor which impacts on accuracy of reconstruction. Secondly, there is a scaling up of the dimension of the wavelet transforms from 64x64 to 64x64x64. This increase in dimension in itself will cause the datacube to be more compressible, further compounding the difference.

The video sequence used here exhibits a high degree of structured temporal dependence. If the temporal dependence is less straightforward, it would be expected that the joint reconstruction would offer less of an advantage. Also, there is the issue of running time. Overall running time for frame-by-frame reconstruction was 142.5s (≈ 2.5 minutes), compared to 1567s (≈ 26 minutes) for joint reconstruction. Scaling the dimension in such a radical way has considerable implications for running time.

Despite these caveats, a clear message emerges from this experiment: it is possible to extend the CS approach to moving scenes, provided we can model the scene as a series of static frames. More than this, we can actually do even better by exploiting temporal dependency by means of 3D transforms.

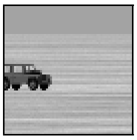
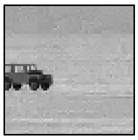
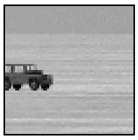
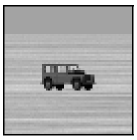
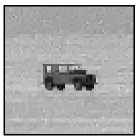
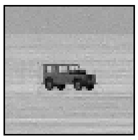
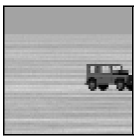
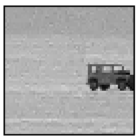
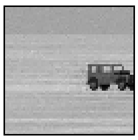
frame	original	frame-by-frame reconstruction	joint reconstruction
24		 RMSE: 7.92	 RMSE: 4.99
32		 RMSE: 9.06	 RMSE: 6.27
40		 RMSE: 8.33	 RMSE: 5.92

Figure 6.24: Comparison of selected frames for frame-by-frame and joint reconstruction

CONCLUSION

The results of testing on the proposed camera model are promising in a variety of areas, including single-band and multi-spectral imaging, and 3D dynamic imaging. Furthermore, we observe robustness in the sense of a controlled degradation in image quality in the presence of sampling noise and foreground/background clutter .

ACKNOWLEDGMENT

This project was facilitated by the *Knowledge Transfer Network (KTN)* as part of the *Underpinning Defence Mathematics (UDM)* programme. The project was a collaboration with the *University of Edinburgh* which made available additional computing facilities. The project was carried out at *SELEX Galileo* largely under the direction of David Humphreys, and I wish to thank him for his significant input. Thanks also go to Robert Lamb for generally overseeing the project. At the university, I would like to thank my supervisors, Jared Tanner and Coralia Cartis, for helpful discussions. Finally, I would like to acknowledge the assistance of David Allwright in his role as *KTN Technology Translator*.

REFERENCES

[1] *Compressive Imaging for Video Representation and Coding* (M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly and R. Baraniuk); Proceedings of the Picture Coding Symposium, Beijing, China (April 2006).

[2] *Iterative Hard Thresholding for Compressed Sensing* (T. Blumensath & M. Davies); Applied and Computational Harmonic Analysis, Vol. 27(3), pp 265-274 (2009); http://ecos.maths.ed.ac.uk/publications.shtml.

[3] *Model-based Compressive Sensing* (R. Baraniuk, V. Cevher, M. Duarte & C. Hegde); IEEE Transactions In Information Theory, Vol. 56, pp 1982-2001 (2010).

[4] *Normalized Iterative Hard Thresholding: guaranteed stability and performance* (T. Blumensath & M. Davies*);* IEEE Journal of Selected Topics in Signal Processing, Vol. 4(2), pp 298-309 (2010); http://ecos.maths.ed.ac.uk/publications.shtml.

[5] *Precise Undersampling Theorems* (D. Donoho & J. Tanner); Proceedings of the IEEE, Vol. 98(6), pp 913-924 (2010);
http://ecos.maths.ed.ac.uk/publications.shtml.

[6] *Single-Pixel Imaging via Compressive Sampling* (M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly & R. Baraniuk); IEEE Signal Processing Magazine, Vol. 25(2), pp 83-91 (2008).

[7] SPGL1*: A solver for large-scale sparse reconstruction* (E. van den Berg & M. Friedlander); http://www.cs.ubc.ca/labs/scl/spgl1/.