EDGE GUIDED RECONSTRUCTION FOR COMPRESSIVE IMAGING

WEIHONG GUO* AND WOTAO YIN†

Abstract. We propose EdgeCS — an edge guided compressive sensing reconstruction approach — to recover images of higher qualities from fewer measurements than the current state-of-the-art methods. Edges are important images features that are used in various ways in image recovery, analysis, and understanding. In compressive sensing, the sparsity of image edges has been widely utilized to recover images. However, edge detectors have not been used on compressive sensing measurements to improve the edge recovery and thus the image recovery. This motivates us to propose EdgeCS, which alternatively performs edge detection and image reconstruction in a mutually beneficial way. The edge detector of EdgeCS is designed to faithfully return partial edges from intermediate image reconstructions even though these reconstructions may still have noise and artifacts. For complex–valued images, it incorporates joint sparsity between the real and imaginary components.

EdgeCS has been implemented with both isotropic and anisotropic discretizations of total variation and tested on incomplete k-space (Fourier) samples. It applies to other types of measurements as well. Experimental results on large-scale real/complex-valued phantom and magnetic resonance (MR) images show that EdgeCS is fast and returns high-quality images. For example, it exactly recovers the 256-by-256 Shepp-Logan phantom from merely 7 radial lines (3.03% k-space), which is impossible for most existing algorithms. It is able to accurately reconstruct a 512-by-512 MR image with 0.05 white noise from 20.87% radial samples. On complex-valued MR images, it obtains recoveries with faithful phases,, which are important in many medical applications. Each of these tests took around 30 seconds on a standard PC. Finally, the algorithm is GPU friendly.

Key words. Compressive sensing, edge detection, total variation, discrete Fourier transform, magnetic resonance imaging

AMS subject classifications. 65K10, 65F22, 65T50, 68U10, 90C25

1. Introduction.

1.1. Background.

Compressive sensing (CS) (cf. the pioneering work [4, 8]) acquires a signal of interest, not by taking many uniform samples, but rather by measuring a few incoherent linear projections followed by an optimization-based reconstruction that exploits the sparsity or compressibility of the signal. For sparse and compressible signals, it requires much fewer linear measurements than dictated by the Nyquist-Shannon sampling theory [27]. This fact makes it very useful in reduce sensing cost in a variety of applications.

One of the early applications of CS and sparsity–based reconstruction is image reconstruction. For example, the Rudin-Osher-Fatemi (ROF) model [25] recovers clean images from noisy ones by minimizing total variation, and the authors of [3] reconstruct 256×256 Shepp-Logan phantom from its incomplete Fourier measurements taken over 22 equally-spaced radial lines. Sparsity in wavelet or tight frame representations is utilized to reconstruct natural images [8] from incomplete measurements. Sparse and redundant representations over learned dictionary is deployed to remove white Gaussian noise [1]. To better sparsely represent signals, optimal sensing and sparsifying bases are learned simultaneously in [9]. Interested readers are referred to two recently published books [28, 10] for more details.

Edges are important features for object tracking and identification, image reconstruction, segmentation, and many other imaging tasks. Edges are usually sparser than images themselves, and the sparsity of edges has been widely exploited in image reconstruction. However, edges are not the only component in images. In the CS measurements, edge and non–edge components are encoded altogether. Recovering just the edges is not simple. On the other hand, there exist very reliable edge detectors, which effectively discriminate edges from other image components. However, these methods work on images, instead of CS measurements. We believe that one does not need to first reconstruct an image from CS measurements before using its edge information. Instead, the two can be integrated. Therefore, we propose to study novel CS image reconstruction approaches that take advantages of both edge sparsity and edge detection. We hope the results of this paper persuade the reader that CS image reconstruction and edge detection benefit each other. In the one–dimensional context, related existing work includes [35], which reconstructs one–dimensional signals through iterative support detection, and [33, 11], which assume in advance that partial support of the underlying signal is known.

This paper has the following two main contributions.

- EdgeCS, an edge guided CS image reconstruction scheme, is introduced. Given undersampled and/or noisy measurements, EdgeCS reconstructs images with fewer errors and artifacts than the state-

*Department of Mathematics, Case Western Reserve University, Cleveland, OH, 44106, USA. (weihong.guo@case.edu).

†Department of Computational and Applied Mathematics, Rice University, Houston, TX, 77005, USA. (wotao.yin@rice.edu)

of-the-art methods. Given measurements with a low sampling rate and high noise variance, a regular decoder usually fails to reconstruct meaningful images. Instead of discarding the failed reconstruction, partial edges are detected from it and then used to adjust sparsity constraints and iteratively improve results. The numerical implementation is based on a fast ℓ_1 like algorithm.
- To better recover complex-valued images, the joint sparsity between the real and imaginary components is utilized in the edge detector. Applied in Magnetic Resonance Imaging (MRI), it is able to improve phase information restoration from incomplete noisy spectral measurements.

Compared to our earlier conference paper [15], improved partial edge detectors are developed, and the second contributions is novel.

The rest of this paper is organized as follows. Section 2 presents the overall EdgeCS approaches for one–dimensional signals and two–dimensional real/complex–valued images, and Section 3 studies edge detection. Section 4 presents numerical implementation of EdgeCS and discusses parameter selections. Section 5 describes numerical simulations results. Finally, conclusions and discussions are given in Section 6.

2. EdgeCS for One–Dimensional Signals and Two–Dimensional images.

Our goal is to recover an image from very few measurements. When examining existing methods, we observed that although insufficient measurements lead to low-quality solutions, there exists useful information in these solutions such as obvious edges. We tried to “recycle” such information and found that after the regularization term (e.g., ℓ_1 or TV) is properly updated (e.g., to weighted ℓ_1 or TV), the new solution can become significantly better. Motivated by these observations, this section studies how to effectively extract jump/edge information from low-quality solutions and use it to improve weighted TV based reconstruction.

Starting from undersampled measurements with excessive noise, we alternate image reconstruction and edge detection to enhance CS reconstruction for one and two dimensional signals[1]. Subsection 2.1 gives a one-dimensional example, which connects the proposed work to a related previous result. Subsection 2.2 focuses on two-dimensional images. Joint sparsity enhanced EdgeCS is explained in subsection 2.3.

2.1. One-Dimensional Signals.

We begin our exposition with a simple demo of reconstructing a piece-wise constant signal from its random measurements. The signal $\bar{u}$ has $n = 200$ entries and 25 randomly located jumps with standard Gaussian sizes. Let Ψ be a 60×200 Gaussian random matrix and $b := \Psi\bar{u}$. Our goal is to recover $\bar{u}$ from b and Ψ.

Define $TV(u) = \sum_{i=1}^{n-1} |u_{i+1} - u_i|$. We compare the solutions of total variation (TV) minimization,

$$\min \{TV(u) : \Psi u = b\},$$

and weighted TV minimization,

$$\min \left\{ \sum_{i=1}^{n-1} g_i |u_{i+1} - u_i| : \Psi u = b \right\}, \tag{2.1}$$

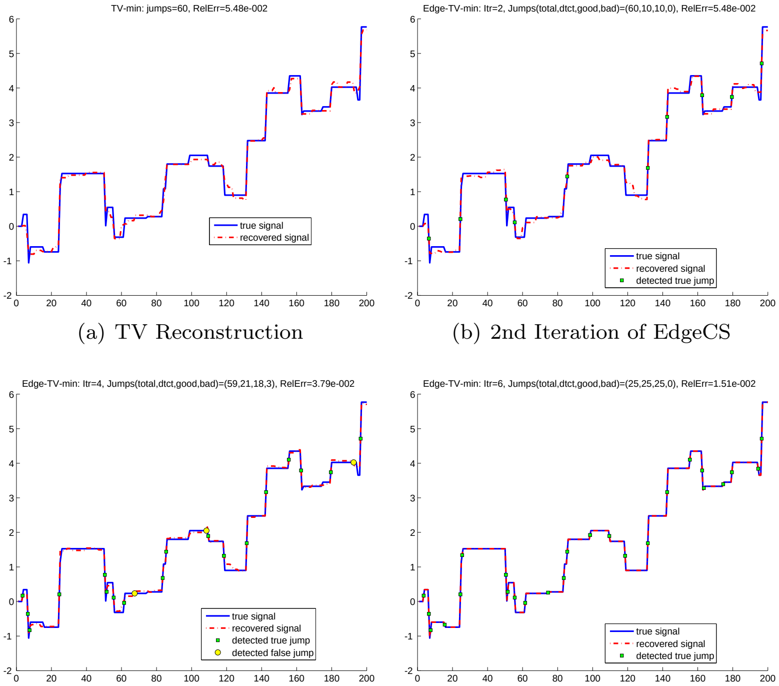
where weights g_i are iteratively learned through jump detection in the following algorithm:

Algorithm: 1D EdgeCS — Jump Guided TV Minimization

```
Input: $\Psi$, $b$, $n$.
1. Iteration number $k \leftarrow 1$; weights $g_i \leftarrow 1$, $\forall i$;
2. While the stopping condition is not met, iterate
   (a) Subproblem: $u^{(k)} \leftarrow$ solve (2.1);
   (b) Jump detection: $I^{(k)} \leftarrow \{i : |u_{i+1} - u_i| > 2^{-k} \max\{|u_{j+1} - u_j|, j = 1, \dots, n\}\}$;
   (c) Weight update in (2.1): $g_i \leftarrow 0,\ \forall i \in I^{(k)}$; $g_j \leftarrow 1,\ \forall j \notin I^{(k)}$;
   (d) $k \leftarrow k + 1$.
```

While direct TV minimization fails to recover $\bar{u}$, the algorithm 1D EdgeCS recovered $\bar{u}$ in merely six iterations. Since $g_i \equiv 1$ initially, the first iteration coincided with standard TV minimization. The solution of TV minimization and those at the end of 2nd, 4th, and 6th iterations are depicted in Figure 2.1. Subfigures (b), (c), and (d) highlight the detected jumps including both the true and false ones. Relative errors are

[1] Extensions to three and higher dimensional signals are straightforward.



(a) TV Reconstruction

(b) 2nd Iteration of EdgeCS

(c) 4th Iteration of EdgeCS

(d) 6th Iteration of EdgeCS

	Itr k	Jumps				Relative Error $\frac{\|u^{(k)}-\bar{u}\|_2}{\|\bar{u}\|_2}$
		Total	Detected	Good	Bad	
TV	1	60	6	6	0	5.48e-2
	2	60	10	10	0	4.71e-2
	3	59	14	14	0	3.79e-2
	4	59	21	18	3	2.86e-2
	5	59	23	21	2	1.51e-2
	6	25	25	25	0	1.82e-15

Fig. 2.1. *Comparison of TV reconstruction and EdgeCS (jump guided) reconstruction*

computed as $\|u^{(k)} - \bar{u}\|_2/\|\bar{u}\|_2$. The quality of these jumps are given in quadruplets "(total,dtct,good,bad)", which are defined as follows:

- total: the total number of jumps in current $u^{(k)}$.
- dtct: the number of detected jumps, equal to $|I^{(k)}|$.
- good: the number of true jumps.
- bad: the number of false jumps.

The TV solution (1st iteration) roughly matches the true signal but misses many small jumps. It contains false jumps and artifacts that are relatively small in size, but most of its large jumps are preserved at their exact locations. Hence, thresholding at $2^{-1}\max\{|u_{j+1} - u_j|\}$ identifies six of them (shown for $k = 1$ in the table) with no false detections. By setting the corresponding g_i to 0 for the second iteration, a smaller reconstruction error is obtained. The improved solution of iteration 2 is then used for detecting more jumps. As EdgeCS iterates, the solutions improve and the thresholds reduce, so more jumps are detected. Note that at iteration 4, the detection includes 3 false jumps; however, since it also introduces 4 true jumps, iteration 5 yields a reduced reconstruction error. Generally, as long as false detections are relative few, more detections lead to lower reconstruction errors. The detection at iteration 5, though including false jumps, has included enough true jumps to allow an exact reconstruction at iteration 6. The solution of iteration 6 exactly recovers all jumps and has a tiny error. All subproblems were solved with MATLAB's linear programming solver "linprog" with the default parameters.

This demo shows that iterative jump detection can be very effective and indeed help recover piece–wise constant signals. This result can be partially explained by analysis in [35], which uses, instead of truncated TV here, truncated ℓ_1 minimization to recover one–dimensional sparse signals. Applying the analysis in [35] to iterative jump detection, we can conclude

1. If the jumps are sparse enough, the recovery at the first iteration is exact; for fewer sparse jumps, more iterations and jump detections are needed.
2. For $u^{(k)}$ to improve over the iterations, the jump detection at each iteration must discover new jumps and they must include more correct ones than false ones. Detailed quantifications depend on the size and condition of Ψ.
3. The total number of iterations is small (no more than 9 empirically).

2.2. Two-Dimensional Images. Similar to reconstruction of one-dimensional signals, we use weighted two–dimensional TV to reconstruct two–dimensional images. This extension requires scrutinizing TV discretization and edge detection approaches. We analyze both the isotropic and anisotropic TV discretizations and describe corresponding edge detection approaches in this subsection.

For demonstration brevity, our presentation is based on incomplete spectral measurements, namely, sampling operator $\Psi = F_p$, which is a normalized partial Fourier ensemble defined as $F_p := PF$ where $P \in \mathbb{R}^{m\times n}$ is a down–sampling matrix. The results, however, apply to general images and sampling modalities. Beside weighted TV, one can also use an extra ℓ_1 norm of sparsifying transform Φ such as wavelets and curvelets to assist the reconstruction.

2.2.1. Isotropic TV. For a two-dimensional image u, the standard isotropic discretization of TV is defined as

$$TV^{iso}(u) := \sum_{i,j} \|D_{i,j}u\|_2 = \sum_{i,j} \sqrt{(u_{i+1,j} - u_{i,j})^2 + (u_{i,j+1} - u_{i,j})^2}.$$

With isotropic TV, we use the following iterative scheme to recover $\bar{u}$:

```
Algorithm: Isotropic EdgeCS
Input: $\Phi$, $b$.
1. Iteration number $k \leftarrow 1$; weights $g_{i,j} \leftarrow 1$, $\forall i, j$;
2. While the stopping condition is not met, iterate
   (a) Subproblem: $u^{(k)} \leftarrow$ solve weighted TV problem
       $$\min_u \ \mu \sum_{i,j} g_{ij}\|D_{ij}u\|_2 + \alpha\|\Phi u\|_1 + \frac{1}{2}\|F_p u - b\|_2^2 \qquad (2.2)$$
   (b) Apply edge detection on $u^{(k)}$ to obtain pixel set $E^k$ consisting of pixels on edges;
   (c) Weight update for (2.2): $g_{ij} \leftarrow 0$, $\forall (i,j) \in E^k$; $g_{ij} \leftarrow 1$, $\forall (i,j) \notin E^k$;
   (d) $k \leftarrow k + 1$.
```

In (2.2), μ and α are two weight parameters that balance total variation and wavelet sparsity. The motivation of assigning zero TV weight ($g_{ij} \leftarrow 0$) to pixels on edges is to free them from the TV minimization and thus encourage the edges to form in the recovery. Clearly, this assumes that those edges indeed exist and correctly detected; otherwise, the zero weight would encourage *false* edges in the recovery. Therefore, faithful edge detection is of critical importance.

2.2.2. Anisotropic TV. In the isotropic TV based model (2.2), the edge set E contains the pixels on edges, and these pixels are defined on grids. If $(i, j) \in E$, then $g_{i,j}$ is set to 0, which frees the whole term $\|D_{i,j}u\|_2$ from minimization. As a result, both $u_{i+1,j}$ and $u_{i,j+1}$ are now detached from $u_{i,j}$, and hence their differences in the recovery are likely to be larger. This is fine if both the pairs of pixels, $(i, j) \sim (i+1, j)$ and $(i, j) \sim (i, j + 1)$, are cut by edges. However, this is not common. More often seen is just one of them cut by an edge, not both. An example is given in Figure 2.3, where only between (i, j) and $(i, j + 1)$ there is an edge. In such a case, freeing just one pair from minimization is more desirable. Nevertheless, the two pairs are nonlinearly coupled by the 2–norm. A compromise is to assign, instead of weight 0, a small yet non-zero weight (e.g., 0.3) to g_{ij}. This works better. Columns 2 and 3 of Figure 3.4 demonstrate its advantage over the 0 weight.

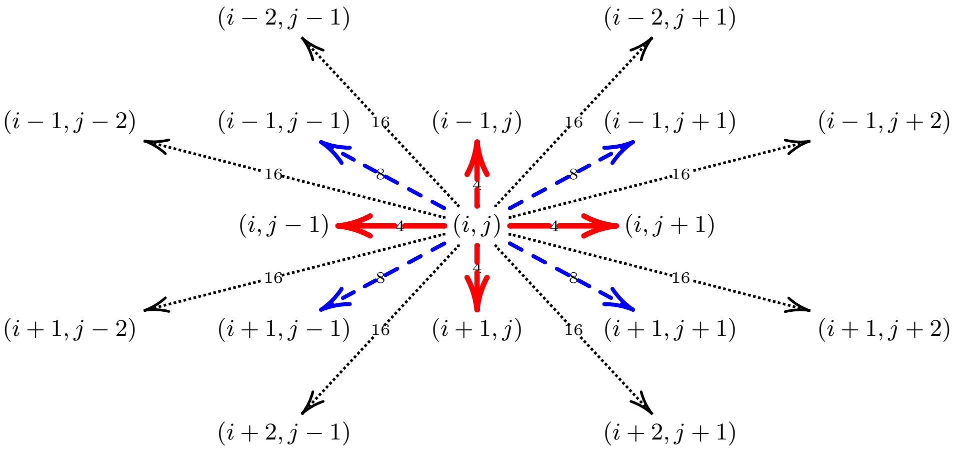


FIG. 2.2. *A graph representing three different anisotropic TV discretizations based on the 4, 8, and 16-neighborhoods of (i,j). These neighborhoods consist of those nodes connected to (i,j) by the solid, dashed, and dotted arcs, respectively. The original weights are as follows ([2]): for the 4-neighbor, all weights equal $\pi/4$; for the 8-neighbor type, nearest 4 neighbors have weights $\pi/8$ and next 4 neighbors have $\sqrt{2}\pi/16$; for the 16-neighbor type, nearest 4 neighbors have weights $\frac{1}{2}\tan^{-1}(\frac{1}{2})$, next 4 neighbors have $\frac{\sqrt{2}}{4}(\frac{\pi}{4}-\tan^{-1}(\frac{1}{2}))$, and last 8 neighbors have $\frac{\sqrt{5}}{80}\pi$.*

Yet another possible approach is to separately weigh the two pairs inside the 2–norm, i.e., applying a weighted 2–norm in the form of $\sqrt{g_a \cdot a^2 + g_b \cdot b^2}$ with nonnegative weights g_a and g_b. However, this leads to complications begging further investigation.

We found that it is rather easy to use anisotropic TV discretization and assign a weight to each $|u(i,j) - u(i',j')|$. More specifically, we follow the same scheme described in section 2.2.1 but substitute the weighted isotropic TV by the following weighted anisotropic TV:

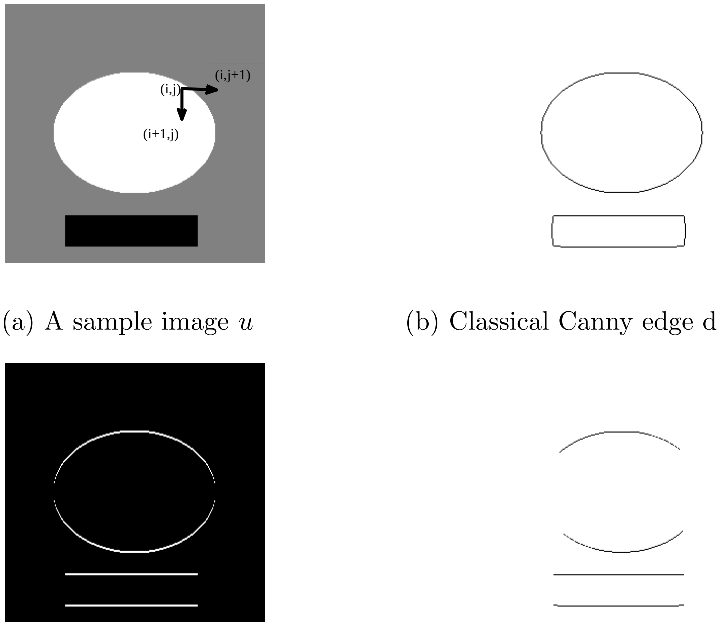
$$\sum_{\alpha} g_\alpha |D_\alpha u| := \sum g_{(i,j)\sim(l,m)} |u_{i,j} - u_{l,m}|$$

where the sum is taken over all pairs of neighbors (i,j) and (l,m), which are denoted by α. The initial values of $g_{(i,j)\sim(l,m)}$ is determined by the anisotropic TV discretization as shown in Figure 2.2. There are three major types for two–dimensional images. The more the neighbors used, the better the approximation to the original TV. If an edge between (i,j) and (l,m) is detected, $g_{(i,j)\sim(l,m)}$ is set to 0. This algorithm is described as follows:

```
Algorithm: Anisotropic EdgeCS
Input: $\Psi$, $b$.
1. Iteration number $k \leftarrow 1$; initialize $g_{(i,j)\sim(l,m)}$ according to Figure 2.2;
2. While the stopping condition is not met, do
   (a) Subproblem: $u^{(k)} \leftarrow$ solve weighted TV minimization
          $\min_u \ \mu \sum g_{(i,j)\sim(l,m)} |u_{i,j} - u_{l,m}| + \alpha \|\Phi u\|_1 + \frac{1}{2}\|F_p u - b\|_2^2;$   (2.3)
   (b) Apply sub-pixel edge detection on $u^{(k)}$;
   (c) Weight update: $g_{(i,j)\sim(l,m)} \leftarrow 0$ if there is an edge between $(i,j)$ and $(l,m)$;
   (d) $k \leftarrow k + 1$.
```

Anisotropic TV permits sub-pixel (i.e., between-pixel) edge detection, which we explain in the next section. With the original and sub-pixel (see edge detection section 3) versions of Canny edge detector, we found that a weighted anisotropic TV, even using the simplest 4-neighbor version, performs much better than weighted isotropic TV.

2.3. Complex–Valued Images. Complex–valued images are widely used in medical imaging.For example, both MR measurements and images are complex–valued, and they provide phase (argument) information that could be used to discriminate fat and water. Since potentially CS imaging has wide applications



(a) A sample image u

(b) Classical Canny edge detection;.

(c) $|\frac{\partial u}{\partial x}|$

(d) Horizontal “sub-pixel” edge detection.

Fig. 2.3. *Comparison of regular edge detection (b) and sub-pixel one (d). In (b), edges are detected based on normal of the gradients and $g_{i,j}$ is 0 (with black intensity) when (i, j) is on edge. In (d), sub-pixel edges are detected only on partial derivatives. For instance, using partial derivative along x-axis, horizontal edges are detected. $g_{(i,j)\sim(i+1,j)}$ is 0 (black) when there is an edge between (i, j) and $(i + 1, j)$.*

in medical imaging, we extend EdgeCS to complex–valued images and, furthermore, exploit the fact that the real and imaginary counterparts of an image share edge locations.

Existing work addressing complex–valued image recovery in MRI context includes [20, 19, 32, 42]. In [20, 19], ℓ_1 sparsity on both the real and imaginary components are assumed and homotopic ℓ_0 sparsity is used in [32]. More recently, separate regularity is applied on the magnitude (ℓ_1) and the phase (smooth) in [42]. We can apply both isotropic and anisotropic EdgeCS by minimizing the sum of the TVs of the real and imaginary components. As an alternative, below we apply joint sparsity between the real and imaginary counterparts under some reasonable assumptions in MRI.

Let u_r, u_i, u_m, θ be the real part, imaginary part, magnitude and phase of u, respectively. In MRI, phase variations occur at the tissue interfaces where magnetic susceptibility changes. Under the assumption that the susceptibility is homogeneous (constant) in one tissue, and it changes to another constant in another tissue, the phase field is piecewise constant. For simplicity, we examine a 1D signal u. From $u_r(x) = u_m \cos(\theta(x)), u_i = u_m \sin(\theta(x))$, it follows that

$$u_r'(x) = u_m'(x)\cos(\theta(x)) - u_m(x)\sin(\theta(x))\theta'(x), \quad u_i'(x) = u_m'(x)\sin(\theta(x)) + u_m(x)\cos(\theta(x))\theta'(x). \tag{2.4}$$

In a domain Ω where θ is constant, $\theta'(x) = 0$, which implies

$$u_r'(x) = u_m'(x)\cos(\theta(x)), \quad u_i'(x) = u_m'(x)\sin(\theta(x)), \quad \theta \in \Omega \tag{2.5}$$

i.e., u_r', u_i', u_m' are multiples of each other. Therefore, the real and imaginary parts share edges at $\partial\Omega$.

The following algorithm recovers a complex-valued image with the help of joint edge detection in $(u_r^{(k)}, u_i^{(k)})$:

```
**Algorithm: Complex EdgeCS**
Input: $\Psi$, $b$.
1. Iteration number $k \leftarrow 1$; initialize weights $g_{(i,j)\sim(l,m)}$ according to Figure 2.2;
2. While the stopping condition is not met, iterate
```

(a) Subproblem: $(u_r^{(k)}, u_i^{(k)}) \leftarrow$ solve the weighted TV problem

$$\min_{(u_r,u_i)} \sum \mu g_{(i,j)\sim(l,m)}(\|D_{ij}u_r\|_1 + \|D_{ij}u_i\|_1) + \alpha\|\Phi u\|_1 + \frac{1}{2}\|F_p u - b\|_2^2; \tag{2.6}$$

(b) Apply sub-pixel Canny edge detection on $(u_r^{(k)}, u_i^{(k)})$ jointly;
(c) Weight update: $g_{(i,j)\sim(l,m)} \leftarrow 0$ if there is an edge between (i,j) and (l,m);
(d) $k \leftarrow k+1$.

3. Edge Detection. This section focuses on developing customized edge detectors for EdgeCS, which are different from general edge detectors in various ways as the purpose here is to facilitate image reconstruction. First of all, while general edge detection recovers step edges (discontinuities in intensity), dirac edges (momentary changes of intensity), fractal edges (which are caused by noise present in the image), as well as other edges, only the step edges are needed by EdgeCS for setting the TV weights. Secondly, as reconstructed from undersampled measurements, the images subject to edge detection in EdgeCS usually contain more artifacts and errors than normal images. An example is the image at the bottom left in Figure 3.4, which is a reconstruction of the first EdgeCS iteration. Finally, while general edge detectors produce edges with compromise among accuracy, completeness, and in some cases, smoothness, EdgeCS needs detected edges to be accurate in location while allowing missing edges. This point is revealed in the analysis for one–dimensional jump detection. In all, we shall develop edge detectors for EdgeCS that reliably detect *partial* edges from the intermediate images of the iterative scheme EdgeCS.

In what follows, we discuss some existing edge detectors and explain how to adapt them for our needs. Their performances on EdgeCS are also compared.

3.1. Some Existing Intensity Based On-Pixel Edge Detectors.

1. Pre-filtering and differentiation based edge detectors.
   An important class of existing edge detectors is based on pre-filtering, followed by intensity differentiation and thresholding. Work in this streamline includes [5, 7, 16, 18, 22, 17]. This class of edge detectors typically take three steps:

Step 1: Removing noise by, for example, convoluting with a low–pass filter, e.g. Gaussian filter $G_\sigma = \frac{e^{-(x^2+y^2)/2\sigma^2}}{2\pi\sigma^2}$ or median filter.

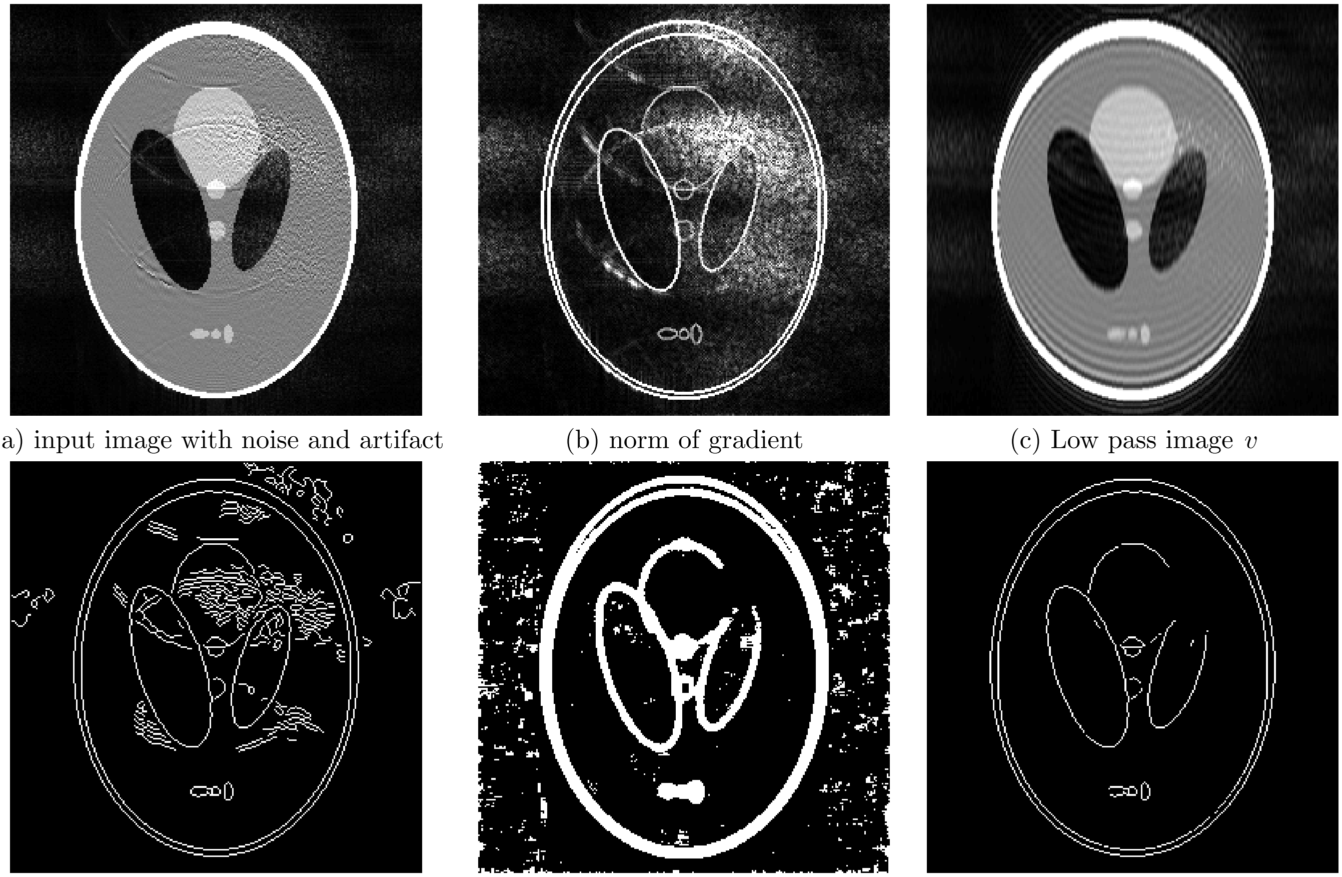
Step 2: Approximating the partial derivatives along $x-, y-$ directions through for instance convolving with two discrete differentiation kernels $k_h = \frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, k_v = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$ respectively. Various kernels have been developed for this purpose with different accuracies along different directions; see [43, 40] for details.

Step 3: Locating edges through thresholding the norm of the gradient values. The standard thresholding turns the pixels with gradients' norms higher than the threshold to edges. Hysteresis thresholding uses two threshold values. Any pixel above the upper threshold is characterized as edge, so are those pixels that are in the neighborhood of an edge pixel and with gradients higher than the lower threshold. The result is edges that tend to connected.

   Besides the above standard steps, post-processings such as non-maximum suppression and smoothing are sometimes conducted to further reduce false positives due to noise. Moreover, instead of maxima of the gradient, zeros of the Laplacian can also be used to locate edges.

   The Canny edge detector [6] (`edge(u,'canny')` in Matlab) is one of the popular detectors in this class that use hysteresis thresholding on gradients. It is robust to low-level noise, but when the noise/artifact is excessive, it picks up false edges for it relies on gradient values, which are sensitive to artifacts and noise. In general, it is difficult to tell edges from artifact-induced steps based solely on gradient values.

2. Local mutual information enhanced edge detector
   To enhance the robustness of the above class of edge detectors to noise and artifacts, one can combine them with local mutual information (LMI) in a method introduced in [14]. Starting with an image u, it yields an image v by applying a low-pass filter on u to remove high-frequency artifacts and noise. At each location x, it treats $u(x)$ and $v(x)$ as random variables and compute the local



(a) input image with noise and artifact (b) norm of gradient (c) Low pass image v

(d) edges detected by Canny (e) LMI detected edges (f) Canny∩LMI

Fig. 3.1. *Illustration of local mutual information (LMI) based edge detection.*

mutual information (LMI) based on their joint and marginal distributions estimated from intensity patches $\{u(y)\}_{y\in NB_r(x)}$ and $\{v(y)\}_{y\in NB_r(x)}$, where $NB_r(x)$ denotes an $r\times r$ patch centered at x. LMI provides a nonnegative measure of the local dependence between u and v at every location. Intuitively it measures how much common characteristic is shared between u and v pixel wisely. It is zero only when $u(x)$ and $v(x)$ appear to be independent, i.e., when their patches near x are quite different. The more similar the patches are, the higher the LMI will be. Therefore, since strong edges show up in both u and v, LMI values are high there. While in locations with high level noise and artifacts, u and v have different patches, and LMI values are lower. LMI is thus able to separate real edges from noise/artifact to some extent. However, it returns thick boundaries due to its dependence on the p.d.f. (see Figure 3.1(e)). Since Canny returns thin boundaries and is better on the image background (see Figure 3.1(d)), intersecting Canny and LMI, written as Canny∩LMI, leads to more reliable edges (set E in EdgeCS) as shown in (f).
The disadvantages of LMI are its slow computation of p.d.f. and its performance depending on the low pass filter used to create v. The Wavelet method below is both accurate and fast.

3. Wavelet based edge detectors.
A piece of image is regular if it can be approximated by a polynomial. As Wavelets can detect changes in regularity at different scales, it is able to detect edges [21] and even recognize their types. Below, we review the notion of Lipschitz regularity and the dyadic wavelet transform.
Definition: Let $0\le\alpha\le 1$. A function $f(x)$ is uniformly Lipschitz α over an interval (a,b) if there exists a constant K such that for any $x_0, x_1\in(a,b)$, $|f(x_0)-f(x_1)|\le K|x_0-x_1|^\alpha$.
Definition: Let $\theta(x)$ be a smoothing function that satisfies $\int_{-\infty}^{\infty}\theta(x)=1$, $\lim_{x\to-\infty}\theta(x)=0$, and

$\lim_{x\to\infty} \theta(x) = 0$. Let $\omega(x) := \theta'(x)$, which is a wavelet itself since $\int_{-\infty}^{\infty} \omega'(x)\mathrm{d}x = 0$. The wavelet transform of f at scale s is defined as $W_s f(x) := f * \omega_s(x)$, where $\omega_s(x) = \frac{\omega(x/s)}{s}$ is the scaled wavelet. For dyadic wavelet transforms, the scale s is chosen as $s = 2^j, j = 1, 2, ...$
Lipschitz regularity is difficult to verify directly and thus not practically useful, but the following theorem relates local Lipschitz regularity with the dyadic wavelet transform.
Theorem[21]: Let $0 < \alpha < 1$. A function $f(x)$ is uniformly Lipschitz α over (a, b) if and only if there exists a constant $K > 0$ such that the wavelet transform satisfies $|W_{2^j} f(x)| \leq K(2^j)^\alpha$, for all $x \in (a, b)$, and $j = 1, 2, \cdots$.
According to this theorem, if $\sup_{x\in(a,b)} |W_{2^j} f(x)|$ strictly decreases as j increases, then $-1 \leq \alpha < 0$, and there is an impulse at x; if the maxima increases with scale, then $0 < \alpha \leq 1$, and the signal/image intensity changes gradually and thus is smooth at x. When the maxima does not change much across scales, $\alpha = 0$, and there is a step edge at x. EdgeCS needs step edges, so the set E includes locations with $\alpha = 0$.

4. Other methods. Other edge detectors include morphological gradient [26, 23, 24], fractal geometry [41, 31], as well as high order and variable order total variation [29] based methods. We have tested a subset of them with EdgeCS, but the results are not competitive. Due to space limitation, we leave them out of discussions.

3.2. On-Pixel Edge Detector Comparisons. We conduct two sets of comparisons. One compares the three edge detectors Canny, Canny∩LMI, and the wavelet method on two intermediate reconstructions of RecPF [39]. The other compares them in the isotropic EdgeCS framework.

RecPF iteratively recovers an image from its incomplete Fourier samples based on solving

$$\min_u \mu \mathrm{TV}(u) + \alpha\|\Phi u\|_1 + \frac{1}{2}\|F_p u - b\|_2^2 \tag{3.1}$$

where $\mathrm{TV}(u)$ can be either isotropic or anisotropic.

For the 256×256 Shepp–Logan phantom, a set of incomplete Fourier measurements were collected on 8 spectral lines (3.98% samples). The three edge detectors were applied to images u at the end of RecPF iterations 20 and 100. The edge detection results are given in Figure 3.2. At iteration 20, u had severe artifacts, and some of them were falsely recognized as edges by Canny and Canny∩LMI. The wavelet detector was less affected. At iteration 100, u ad fewer artifacts. As a result, all three detectors produced better edges than previously. In particular, the wavelet edges were nearly exact and contained less false positives. Since EdgeCS is sensitive to not false edges, the wavelet detector appears to be the best choice. Furthermore, the wavelet detector took just one fifth of the time needed by Canny∩LMI.

We plugged in each of the three edge detectors in isotropic EdgeCS and compared their performance. From the same measurements above, intermediate reconstructions and detected edges at iterations 50, 300, and 2000 of EdgeCS are given in Figure 3.3. One can see that even after 2000 iterations EdgeCS with Canny still failed to reconstruct an accurate image while EdgeCS with either Canny∩LMI or wavelet succeeded after only 300 steps. EdgeCS with Canny∩LMI eventually returned slightly better results than with wavelet. However the former one was way slower. Considering both quality and speed, we found wavelet edge detector as the best fit for isotropic EdgeCS, and it was used in the simulation presented in section 5.

3.3. Sub-Pixel Edge Detection. The anisotropic EdgeCS in subsection 2.2.2 requires sub-pixel edges. We have modified Canny for sub-pixel edge detection, but it is difficult to do so on Canny∩LMI and the wavelet edge detector. The original Canny applies hysteresis thresholding on gradient magnitudes $\|D_{ij}u\|_2$. For sub-pixel edges, we let hysteresis thresholding be applied to directional gradients $D_\alpha u$, e.g., $|u(i, j) - u(i + 1, j)|$ and $|u(i, j) - u(i, j + 1)|$. The Canny sub-pixel edge detector was used with both anisotropic and complex EdgeCS in the simulation presented in section 5.

4. Algorithms. This section provides implementation detail of EdgeCS. Our implementation is based on iterative edge detection and the recent solvers RecPF [39] with split Bregman [13], which are applied to (2.3). RecPF and split Bregman can both be derived from the well-known alternating direction method of multipliers (ADMM) (cf. [12]), and they are very efficient because they break a non-smooth multi-term optimization problem (3.1) into subproblems with closed-form solutions. This advantage for TV regularization problems was first discovered in [34] for image denoising and deblurring, and has been generalized to

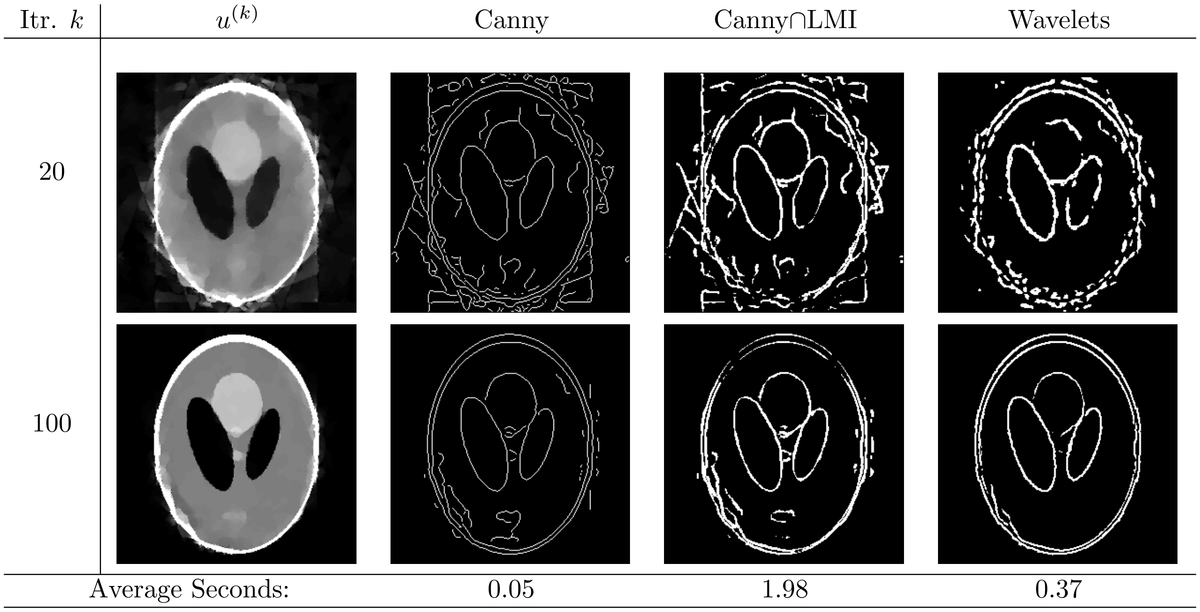


Fig. 3.2. *Comparison of three edge detectors on two intermediate RecPF reconstructions from spectral measurements taken on 8 radial lines (*3.98%*).*

multi-channel problems in [37], the TV-L^1 model in [38], and TV-based compressed sensing in [39] and [13]. Since we need weighted TV, which is not covered in the above work, we briefly describe the algorithm for (2.3) with weighted TV and discuss its performance. It extends to isotropic and complex versions easily.

Consider an $m \times n$ image. By introducing $H(u,b) = \frac{1}{2}\|F_p u - b\|_2^2$ and auxiliary variables $z \in \mathbb{C}^{m\times n}$ and $w = [w_1, \dots, w_{m\times n}]$, where each $w_i \in \mathbb{C}^2$, we can rewrite (2.3) as the equivalent constrained problem

$$\min_{u,w,z} \; \mu \sum_{\alpha} g_\alpha |w_\alpha| + \alpha \|z\|_1 + H(u,b), \quad \text{s.t. } z = \Phi u, \; w_\alpha = D_\alpha u, \forall i, \tag{4.1}$$

where α stands for a pair of neighbor pixels and $D_\alpha u$ is their difference. To simplify notation, we introduce

$$J_g(w) := \mu \sum_{\alpha} g_\alpha |w_\alpha|, \quad K(z) := \alpha \|z\|_1.$$

The augmented Lagrangian of (4.1) is

$$\mathcal{L}(u,w,z) := J_g(w) + \frac{\mu\beta}{2}\|w - Du - d_w\|_2^2 + K(z) + \frac{\alpha\beta}{2}\|z - \Phi u - d_z\|_2^2 + H(u,b), \tag{4.2}$$

where d_w and d_z are Lagrange multipliers of proper sizes. The problems of minimizing $\mathcal{L}(u,w,z)$ with respect to each of u, w, and z while fixing the rest two are called the u–, w–, and z–subproblems:

$$\begin{aligned}
&u\text{-subproblem:} && \min_u \quad \frac{\mu\beta}{2}\|w - Du - d_w\|_2^2 + \frac{\alpha\beta}{2}\|z - \Phi u - d_z\|_2^2 + H(u,b), && (4.3)\\
&w\text{-subproblem:} && \min_w \; J_g(w) + \frac{\mu\beta}{2}\|w - Du - d_w\|_2^2, && (4.4)\\
&z\text{-subproblem:} && \min_z \quad K(z) + \frac{\alpha\beta}{2}\|z - \Phi u - d_z\|_2^2. && (4.5)
\end{aligned}$$

The ADMM applied to ((4.1) or equivalently (2.3)) is

Algorithm 1. *Given b, Φ, F_p, α, μ*

```
1. Normalize input $b$ and parameters $\alpha$ and $\mu$
2. Set $\beta$ and $\gamma$
```

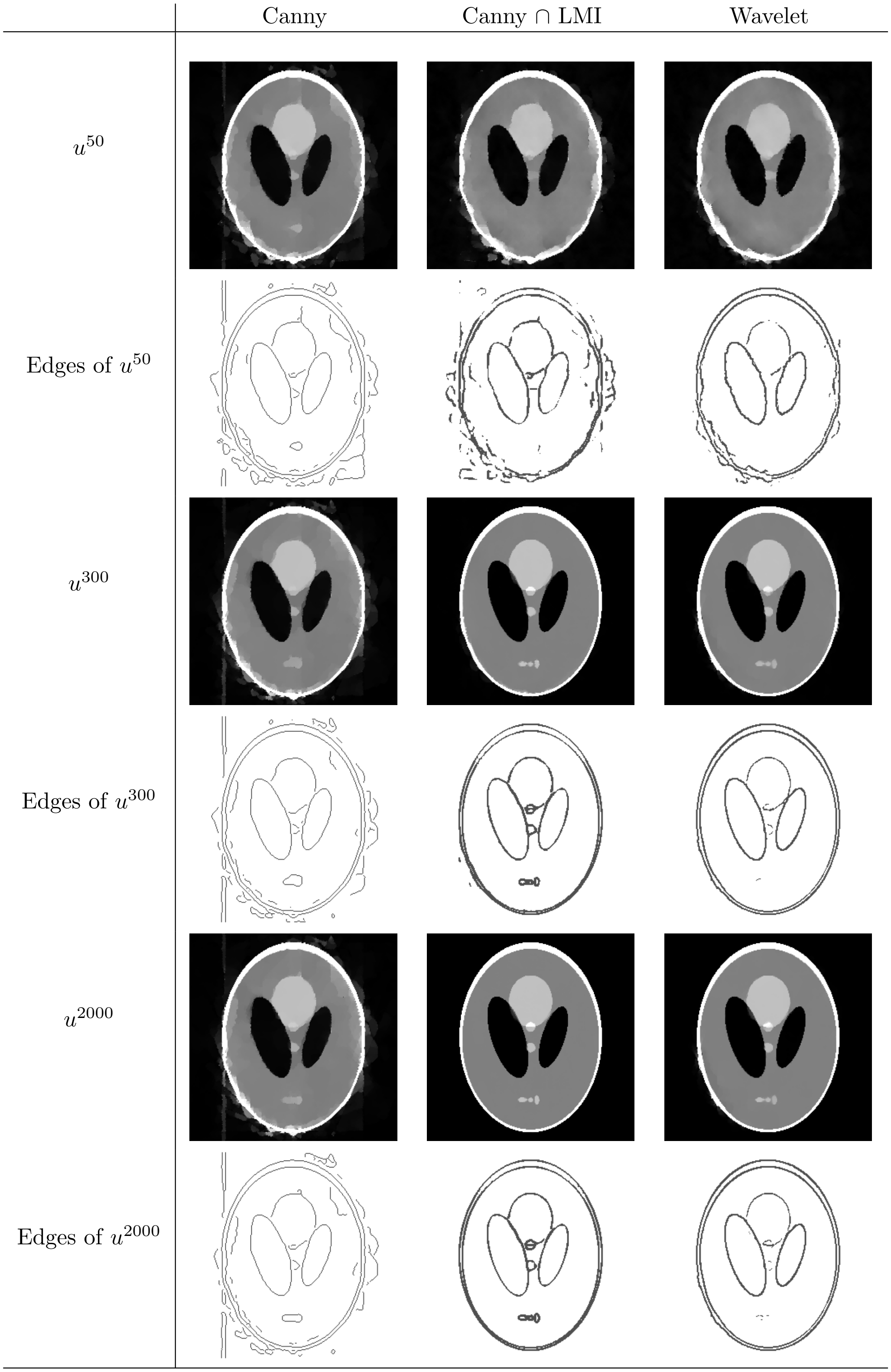
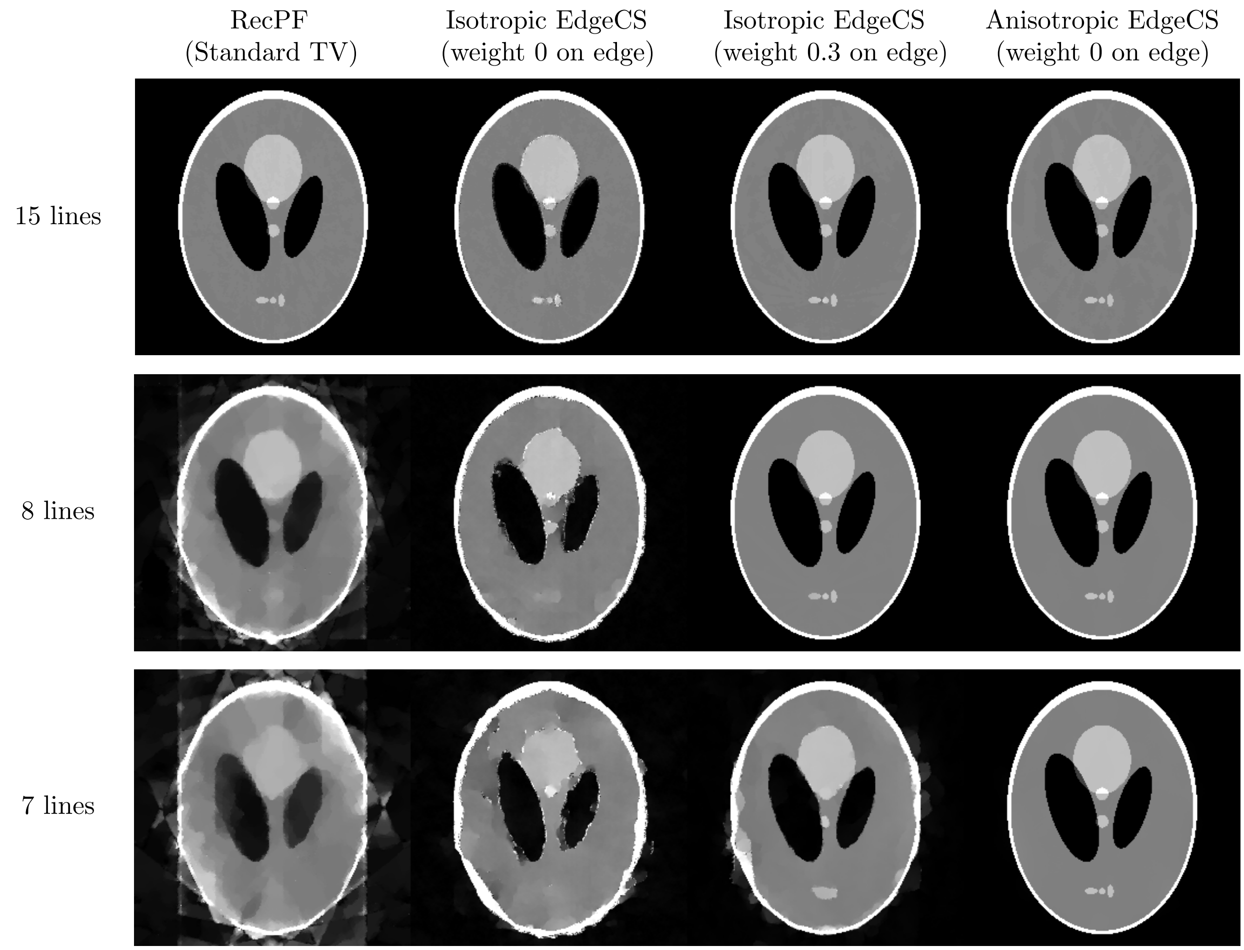


FIG. 3.3. *Comparison of three on-pixel edge detectors on three intermediate isotropic EdgeCS reconstructions from spectral measurements taken on 8 radial lines (*3.98%*).*



	Relative error			
15 lines	.11%	5.00%	.45%	.006%
8 lines	37.17%	30.55%	.74%	.086%
7 lines	51.65%	47.97%	27.90%	1.09%
	SNR (dB)			
15 lines	58.16	24.78	45.74	64.10
8 lines	7.4	9.06	41.40	60.10
7 lines	4.5	5.10	9.80	38.00(see caption)

FIG. 3.4. *Comparison of four reconstruction methods on measurements taken on 15, 8, 7 radial lines (sample rates* 6.44%, 3.98%, 3.03%, *respectively) with no noise added, and parameter* $\mu = 10^{-10}$. *Note: SNR 38dB of Anisotropic EdgeCS at 7 lines can be significantly improved if more iterations are allowed.*

```
3. $k \leftarrow 0$, $u^{(k)} \leftarrow \mathbf{0}$, $d_w^{(k)} \leftarrow \mathbf{0}$, $d_z^{(k)} \leftarrow \mathbf{0}$
4. while not converged do
5.     $k \leftarrow k+1$,
6.     $w^{(k)} \leftarrow$ solve (4.4) for $u = u^{(k-1)}$ and $d_w = d_w^{(k-1)}$,
7.     $z^{(k)} \leftarrow$ solve (4.5) for $u = u^{(k-1)}$ and $d_z = d_z^{(k-1)}$,
8.     $u^{(k)} \leftarrow$ solve (4.3) for $d_w = d_w^{(k-1)}$ and $d_z = d_z^{(k-1)}$,
9.     $d_w^{(k)} \leftarrow d_w^{(k-1)} - \gamma(w^{(k)} - Du^{(k)})$ and $d_z^{(k)} \leftarrow d_z^{(k-1)} - \gamma(\Phi u^{(k)} - z^{(k)})$,
10. End while.
11. Denormalize output $u^k$
```

The w-subproblem (4.4) is separable in each w_i, so it is straightforward to derive the minimizer (cf. [37]):

$$w_i^* = \text{shrink}\left(D_i u + (d_w)_i, \frac{g_i}{\beta}\right), \quad \text{where} \quad \text{shrink}(\mathbf{t}, a) := \max\{0, \|\mathbf{t}\| - a\} \cdot \frac{\mathbf{t}}{\|\mathbf{t}\|}$$

and $0 \cdot (0/0) = 0$ is assumed.

The z-subproblem (4.5) is also separable in each z_i, so the minimizer is given by

$$z_i^* = \text{shrink}\left((\Phi u)_i + (d_z)_i, \frac{1}{\alpha}\right).$$

The u-subproblem (4.3) is also surprisingly simple to solve. Since the objective function is convex quadratic, the minimizer u^* is the solution of the normal equations:

$$\left(\alpha\beta I + \mu\beta D^\top D + F_p^* F_p\right) u = \alpha\beta\Phi^*(z - d) + \mu\beta D^\top(w - \mathbf{b}) + F_p^* b, \tag{4.6}$$

where the first term $\alpha\beta I$ is obtained from the fact $\alpha\beta\Phi^*\Phi = \alpha\beta I$ because Φ is unitary. Under the periodic boundary conditions for u, the finite difference operator D is a block-circulant linear operator so $D^\top D$ and thus the entire left-hand side matrix in (4.6) can be diagonalized by the discrete Fourier transform F, namely,

$$F\left(\alpha\beta I + \mu\beta D^\top D + F_p^* F_p\right) F^* = \alpha\beta I + \mu\beta \hat{D}^* \hat{D} + P^\top P$$

is diagonal (noticing that $P^\top P$ is a diagonal 0/1 matrix). One does not need to form the matrix in the computation. Define

$$\hat{d} := \text{diag}(\alpha\beta I + \mu\beta \hat{D}^* \hat{D} + P^\top P).$$

Since $\hat{d}$ remain constant through out all iterations, it should be computed at the beginning of the algorithm and used repeatedly. The minimizer of (4.3) is given by

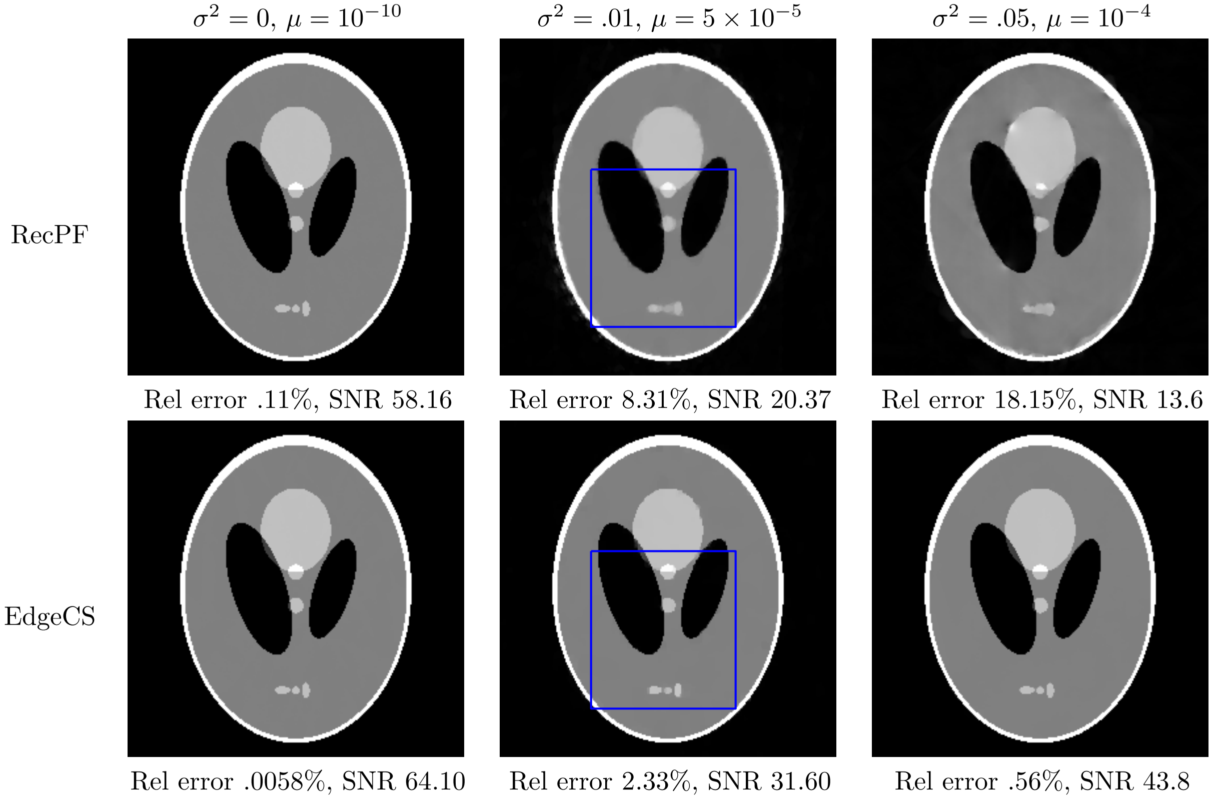
$$u^* = F^*\left(F(\alpha\beta\Phi^*(z - d) + \mu\beta D^\top(w - \mathbf{b}) + F_p^* b)./\hat{d}\right),$$

where ./ stands for component-wise division. As only two FFTs are needed, solving the u-subproblem (4.3) is simple and fast.

The algorithm is GPU-friendly since besides FFTs, which already have GPU implementations, the computations are local to pixels. A preliminary version has been implemented and tested on nVidia GPUs with a promising speedup.

4.1. Parameters and performance. The parameters β and γ in Algorithm 1 only moderately affect the convergence speed. It is proved in [12] that $\gamma < (\sqrt{5} + 1)/2$ guarantees global convergence. In our test, we found that $1 \leq \gamma < (\sqrt{5} + 1)/2$ consistently yields good performance. The other parameter β must be strictly positive but not upper bounded. To make β simple to choose, we normalize the input b and the regularization parameters α and μ in order to remove the variations due to the image size, pixel intensity range, and the number of measurements k (i.e., the size of b). Specifically, both α and μ are multiplied with $k/\sqrt{m \times n}$ for the normalization with respect to the image size and k, and b is rescaled by dividing the pixel intensity range (e.g., 255 for 8-bit or 65535 for 16-bit). With such normalization, we found that β between 5 and 20 consistently leads to good performance. Alternatively, one can increase β over the iterations; refer to [12] for details. Note that the above normalization also makes α and μ relatively independent of the three factors mentioned above, and they remain dependent on the gradient sparsity of the underlying image and the noise/error level in the measurements.

The algorithm is terminated after a fixed number of iterations. For most problems, 100 and 300 iterations were used for isotropic and anisotropic EdgeCS, respectively. Difficult problems may need that more iterations, and in general they lead to higher SNRs and lower errors, especially for problems with very low sampling rates.


$\sigma^2 = 0$, $\mu = 10^{-10}$
$\sigma^2 = .01$, $\mu = 5 \times 10^{-5}$
$\sigma^2 = .05$, $\mu = 10^{-4}$
RecPF
EdgeCS

Rel error .11%, SNR 58.16 — Rel error 8.31%, SNR 20.37 — Rel error 18.15%, SNR 13.6

Rel error .0058%, SNR 64.10 — Rel error 2.33%, SNR 31.60 — Rel error .56%, SNR 43.8

FIG. 5.1. *Comparisons of RecPF and anisotropic EdgeCS at different noise variances.*

5. Simulation Results.

As the first algorithm that iteratively uses edges to help recover images, there are no similar algorithms to compare with. We comment that a related paper [30] compares three methods for edge detection from incomplete Fourier measurements, but none of them produces images. Therefore, we compared three EdgeCS algorithms — isotropic, anisotropic, and complex — with RecPF [39], which is based on isotropic TV and does not exploit edge detection. In anisotropic and complex EdgeCS, 4 neighbors were used for each pixel. These algorithms were compared on image reconstructions from significantly under-sampled measurements, as well as such measurements with excessive noise.

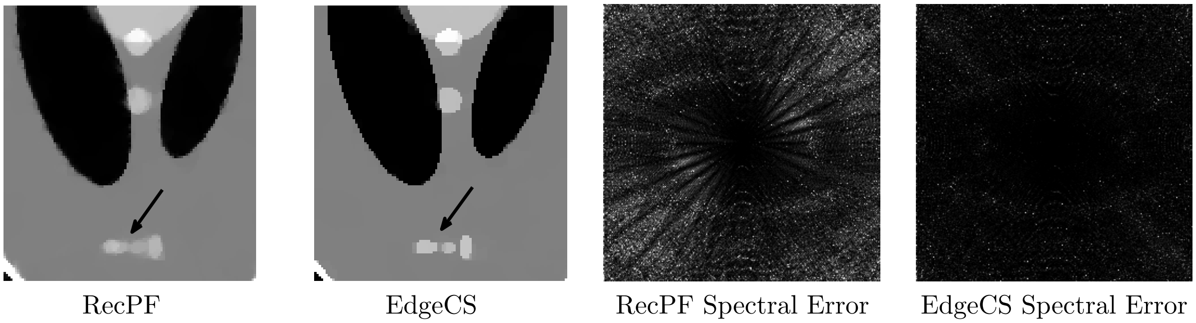
The parameters μ and α control the overall performance. We fixed α to 0 in all tests in order to focus on the contributions due to edge detection. Upon being called, all algorithms scaled the input data and parameter μ in the same way in order to normalize off the effects due to varying image sizes, pixel intensity ranges, and sample sizes. Specifically, each given μ was multiplied by the sample size and divided by the square root of pixel quantity, and b is divided by the intensity range. The returned images were denormalized by multiplying with the intensity range. All results were obtained with tight parameters to avoid loss of quality due to early stopping. All tests were run under Windows 7 and MATLAB v7.6 (R2008a) on a laptop with an Intel Core 2 Duo CPU at 2.0 GHz and 3 GB of memory.

We normalized the ground truth images to the intensity range $[0, 1]$. Their spectral measurements were collected on smooth radial sampling trajectories that are known to satisfy the restricted isometry property [36].

5.1. Shepp–Logan Phantom.

In Figure 3.4, we compare isotropic EdgeCS with either 0 or 0.3 weights on detected edges, anisotropic EdgeCS, and RecPF on recovering images from $6.44\%, 3.98\%, 3.03\%$ (or $15, 8, 7$ radial lines) of spectral measurements of the 256×256 Shepp-Logan phantom.

Given sufficient measurements (15 radial lines), the four results are visually comparable. In terms of error and SNR, however, anisotropic EdgeCS's image is slightly better. From the 8-line measurements, RecPF returned an image with a relative error of 37.15%, but isotropic EdgeCS with .3 weight on edges and anisotropic EdgeCS returned much better images. From the 7-line measurements, anisotropic EdgeCS still



RecPF EdgeCS RecPF Spectral Error EdgeCS Spectral Error

Fig. 5.2. *Zoom-in comparisons of RecPF and anisotropic EdgeCS results at* $\sigma^2 = .01$ *and* $\mu = 5 \times 10^{-5}$. *Black arrows point to small geometries that are preserved by anisotropic EdgeCS (middel–left) but not by RecPF (left).*

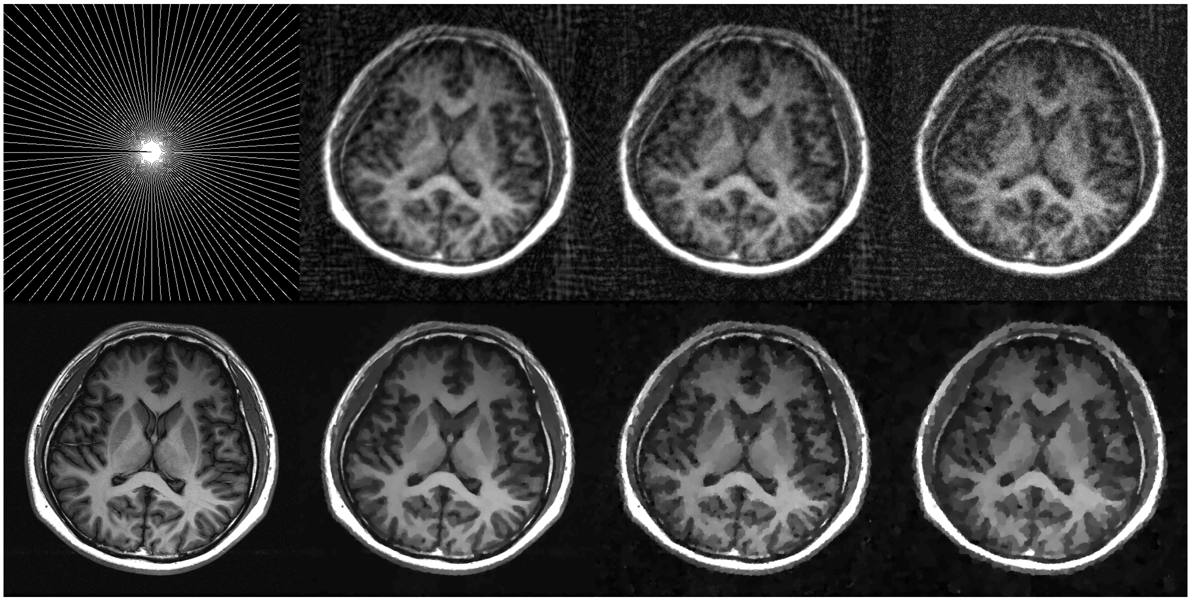


Fig. 5.3. *Comparison between back projection (top row, last three) and anisotropic EdgeCS (bottom row, last three) on reconstructing a real-world brain slide (bottom row, left) from spectral measurements taken on 50 radial lines (top row, left,* 10.64% *sampling rate) out of a* 512×512 *domain, added with noise of variances* 0, .05*, and* .1*.*

achieved an almost exact recovery while RecPF returned an image with huge errors and lots of artifacts, and isotropic EdgeCS was not effective either.

Anisotropic EdgeCS also returned better images than RecPF from noise-contaminated measurements. Figure 5.1 depicts the images recovered from measurements that are 6.44% of k-space (or 15 radial lines) added with Gaussian noise of varying variances $\sigma^2 = 0, .01, .05$ (recall that the phantom has intensity values between 0 and 1). Corresponding to $\sigma^2 = 0$ (or no noise), RecPF and anisotropic EdgeCS returned comparable images. Corresponding to $\sigma^2 = .01$ (or a low level of noise), anisotropic EdgeCS produced an image with sharper edges (observable in the boxed areas). Corresponding to $\sigma^2 = .05$, anisotropic EdgeCS returned a much better image than RecPF, which failed to recover the small shapes.

5.2. Real–Valued MR Images. Since anisotropic EdgeCS appears to perform better than the rest, we continue to test it on reconstructing anatomical MR images. We used a 512×512 real-valued fully sampled MR image as the ground truth and simulated different measurements. Figure 5.4 is based on 100-radial-line sampling (20.87% sampling rate) and noise variances $\sigma^2 = 0, .05$ and .1. Figure 5.3 is based on 50-radial-line sampling (10.64% sampling rate), and Figure 5.5 is based on a random sampling trajectory (26.22% sampling rate). It is observed that with a sampling rate as low as 10%, faithful results were still recovered even at the noise variance of .05. All μ's were chosen in between 10^{-4} and 10^{-3}. The entire computation took less than

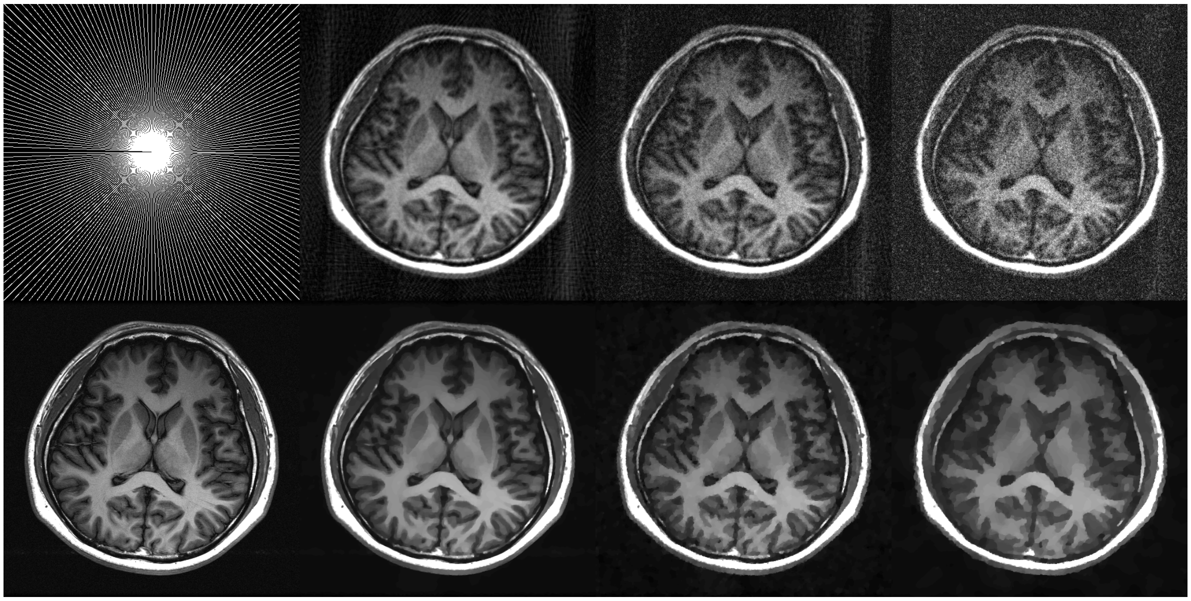


Fig. 5.4. *Comparison between back projection (top row, last three) and anisotropic EdgeCS (bottom row, last three) on reconstructing a real-world brain slide (bottom row, left) from spectral measurements taken on 100 radial lines (top row, left,* 20.87% *sampling rate) out of a* 512×512 *domain, added with noise of variances* 0, .05, *and* .1.

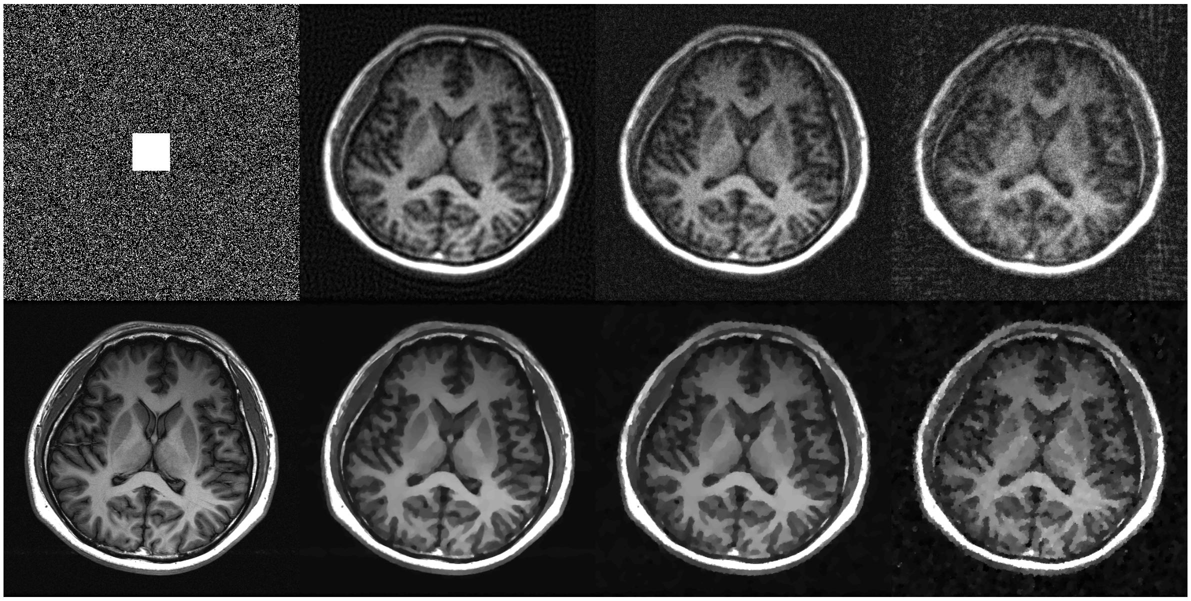


Fig. 5.5. *Comparison between back projection (top row, last three) and anisotropic EdgeCS (bottom row, last three) on reconstructing a real-world brain slide (bottom row, left) from spectral measurements taken on full-low-plus-random-high frequencies (top row, left,* 26.22% *sampling rate) out of a* 512×512 *domain, added with noise of variances* 0, .05, *and* .1.

30 seconds.

5.3. Complex–valued MR Images. Motivated by the fact that most of the MR images are complex–valued, we conducted a test on reconstructing complex–valued images. Here, we compare anisotropic EdgeCS to complex EdgeCS. The former treats the real and imaginary parts with one TV term for each with independent weights. The latter uses just on set of weights and utilizes the joint edge detection. The results depicted in Figures 5.6 5.7 and 5.8 indicate that the latter method returned significantly better results.

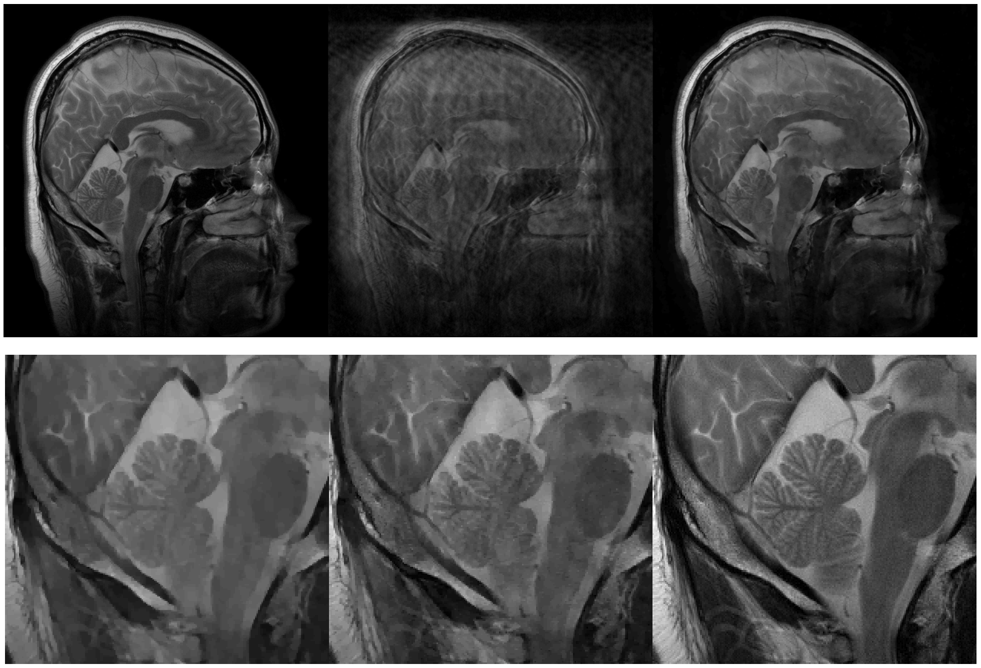


FIG. 5.6. *A complex-valued sagittal brain example (only magnitude images are shown), 200 radial lines out of* 512×500 *domain,* 39.06% *sampling rate. Top (left to right): ground truth, back projection, and anisotropic EdgeCS results. Bottom (left to right): zoom-in of anisotropic EdgeCS, complex EdgeCS, and ground truth. Relative errors of anisotropic EdgeCS and complex EdgeCS are* 21.95% *and* 16.36%*, and SNRs are* 12.5 *and* 14.5*, respectively.*

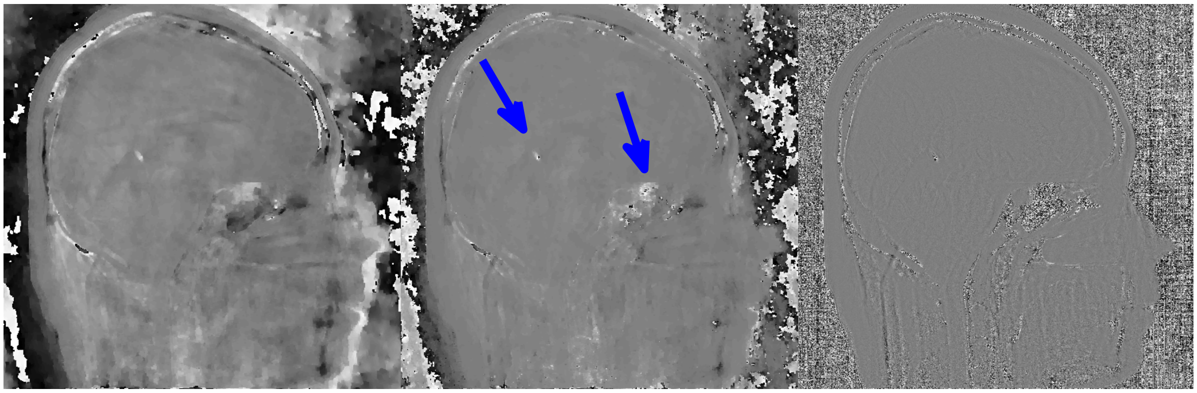


FIG. 5.7. *From left to right: phase map of anisotropic EdgeCS, complex EdgeCS, and ground truth. Blue arrows point to the areas where complex EdgeCS performs significantly better.*

The latter is better than the former in image quality (second row in Figure 5.6), relative error (21.95% vs. 16.36%), SNR (12.5 vs. 14.5), phase accuracy (Figure 5.7), and fine feature preservation (left and middle images of Figure 5.8). This slice of brain image has rich fine structures that are not so sparse under TV, yet complex EdgeCS largely reconstructed those fine structures.

6. Conclusions. We proposed an edge guided reconstruction method to recover images from incomplete measurements. It has a long history that edge sparsity and TV minimization are used for image recovery. But, edges are not only sparse but are also connected and have other properties, which have been well exploited by various edge detectors. It appears that coupling edge detection and TV minimization leads to more faithful reconstructions. Further investigation shows that the coupling of the two requires the edge detector to be tailored to intermediate image reconstructions and the TV term to be tailored to detected partial edges. In spite of getting state-of-the-art recoveries, our methods are rather simple, and we believe

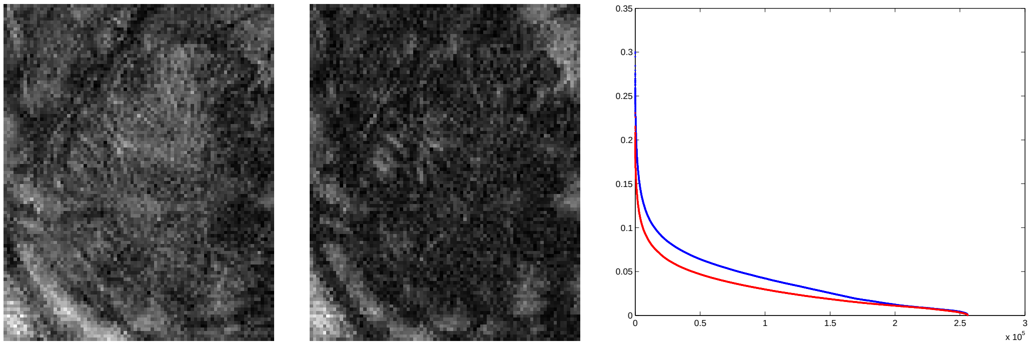


Fig. 5.8. *Zoom-in comparison of reconstruction errors between anisotropic EdgeCS (left) and complex EdgeCS (middle) on reconstructing a complex-valued brain image. Right plot: comparison of sorted reconstruction errors (blue: anisotropic EdgeCS, red: complex EdgeCS).*

that there is large room for further improvements by exploiting the structures of edges, as well as other image geometries, in a more effective way.

ACKNOWLEDGMENTS. The work of W. Yin was supported in part by NSF CAREER Award DMS-07-48839, ONR Grant N00014-08-1-1101, and an Alfred P. Sloan Research Fellowship.

REFERENCES

[1] M Aharon, M Elad, and A. Bruckstein. K-SVD: An algorithm for denoising overcomplete dictionaries for sparse representation. *IEEE Tran. Sig. Proc.*, 54(11):4311–4322, 2006.

[2] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. *IEEE International Conference on Computer Vision*, 1:26–33, 2003.

[3] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, 2006.

[4] E. Candès and T. Tao. Near optimal signal recovery from random projections: universal encoding strategies. *IEEE Transactions on Information Theory*, 52(1):5406–5425, 2006.

[5] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8:679–714, 1986.

[6] J. F. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6).

[7] R. Deriche. Fast algorithms for low-level vision. *IEEE Tran. Pattern Anal. Mach. Intell.*, 12(A):78–87, 1990.

[8] D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52:1289–1306, 2006.

[9] J. Duarte-Carvajalino and G. Sapiro. Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization. *IEEE Trans. Image Processing*, 18(7):1395–1408, 2009.

[10] M. Elad. Sparse and redundant representations: from theory to applications in signal and image processing. *Springer*, 2010.

[11] M. Friedlander, H. Mansour, R. Saab, and O. Yilmaz. Recovering compressively sampled signals using partial support information. *arXiv:1010.4612*, 2010.

[12] R. Glowinski, J. L. Lions, and R. Tremolieres. *Numerical Analysis of Variational Inequalities*. North-Holland, Amsterdam, New York, 1981.

[13] T. Goldstein and S. Osher. The split Bregman algorithm for L1 regularized problems. *UCLA CAM Report 08-29*, 2008.

[14] W. Guo and F. Huang. A local mutual information guided denoising technique and its application to self-calibrated partially parallel imaging. *MICCAI, Part II, LNCS 5242*, pages 939–947, 2008.

[15] W. Guo and W. Yin. Edgecs: Edge guided compressive sensing reconstruction. *Proceedings of SPIE Visual Communication and Image Processing*, 7744:77440L–1–77440L–10, 2010.

[16] B. Jahne, H. Scharr, and S. Korgel. Principles of filter design. *In B. Jahne, H. HauEecker, and P. GeiEJer, eds., Computer Vision and Applications, vol-volume 2, Signal Processing and Pattern Recognition, chapter 6, Academic Press, San Diego*, pages 125–151, 1999.

[17] J. J. Koenderink and A. J. van Doom. Generic neighborhood operators. *IEEE Trans. PAMI*, 14(F):597–605, 1992.

[18] S. Lanser and W. Eckstein. Eine modification des deriche-verfahrens zur kantendetektion. *B. Radig, ed., Mustererkennung 1991, vol. 290 of Informatik Fachberichte, DAGM Symposium, Munchen, Springer, Berlin*, pages 151–158, 1991.

[19] M. Lustig, D. Donoho, and J. Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, in press, 2007.

[20] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly. Compressed sensing MRI. *IEEE Signal Proc. Magazine*, 25(2):72–82, 2008.

[21] S. Mallat. Characterization of signals from multiscale edges. *IEEE. Trans. Patt. Anal. Machine Intell.*, 14:710–732, 1992.

[22] D. Marr and E. Hildreth. Theory of edge detection. *Proc. Royal Society, London*, pages 187–217, 1980.

[23] I. Pitas and A.N. Venetsanopoulos. Nonlinear digital filters: Principles and applications. *Kluwer Academic Publishers*, 1990.

[24] D. Qi, F. Guo, and L. Yu. Medical image edge detection based on omnidirectional multi-scale structure element of mathematical morphology. *IEEE International Conference on Automation and Logistics*, pages 2281–2286, 2007.

[25] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithm. *Physica D*, 60:259–268, 1992.

[26] J. Serra. Image analysis and mathematical morphology. *Academic Press*, 1982.

[27] C.E. Shannon. Communication in the presence of noise. *Proc. Institute of Radio Engineers*, 37(1):10–21, 1949.

[28] J. Starck, F. Murtagh, and J. Fadili. Sparse image and signal processing. wavelets, curvelets, morphological diversity. *Cambridge*, 2010.

[29] W. Stefan, R. A. Renaut, and A. Gelb. Improved total variation-type regularization using higher-order edge detectors. *SIAM Journal on Imaging Sciences*, 3(2):232–251, June 2010.

[30] E. Tadmor and J. Zou. Three novel edge detection methods for incomplete and noisy spectral data. *J. Fourier Anal Appl*, 14(5-6):744–763, 2008.

[31] B. Tian, H. Yuan, and X. Yue. Feature extraction algorithm for space targets based on fractal theory. *Second International on Space Information Technology, SPIE-INT SOC Optical Engineering*, 6795:79518–79518, 2007.

[32] J. Trzasko, A. Manduca, and E. Borisch. Highly undersampled magnetic resonance image reconstruction via homotopic ℓ_0-minimization. *SSP 2007*, 2007.

[33] N. Vaswani and W. Lu. Modified-cs: Modifying compressive sensing for problems with partially known support. *IEEE Trans. Signal Processing*, 58(9):4595 – 4607, 2010.

[34] Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

[35] Y. Wang and W. Yin. Sparse signal reconstruction via iterative support detection. *SIAM Journal on Imaging Sciences*, 3(3):462–491, 2010.

[36] R. M. Willett. Smooth sampling trajectories for sparse recovery in MRI. *Proc. Int. Sym. Biomedical Imaging*, pages 1044–1047, 2011.

[37] J. Yang, W. Yin, Y. Zhang, and Y. Wang. A fast algorithm for edge-preserving variational multichannel image restoration. *SIAM Journal on Imaging Sciences*, 2(2):569–592, 2008.

[38] J. Yang, Y. Zhang, and W. Yin. An efficient TVL1 algorithm for deblurring multichannel images corrupted by impulsive noise. *Rice University CAAM Technical Report TR08-12*, 2008.

[39] J. Yang, Y. Zhang, and W. Yin. A fast TVL1-L2 algorithm for image reconstruction from partial fourier data. *Submitted to IEEE Journal of Selected Topics in Signal Processing Special Issue on Compressed Sensing. Rice University CAAM Technical Report TR08-27*, 2008.

[40] L. Zhai, S. Dong, and H. Ma. Recent methods and applications on image edge detection. *International Workshop on Geoscience and Remote Sensing*, pages 332 – 335, 2008.

[41] L. Zhang, A. Butler, and C. Sun. Fractal dimension assessment of brain white matter structural complexity post stroke in relation to upper-extremity motor function. *Brain Research*, 1228(4):229–240, 2008.

[42] M. V. W. Zibetti and A. R. De Pierro. Separate magnitude and phase regularization in MRI with incomplete data: preliminary results. *Int. Sym. Bio. Med.*, pages 736–739, 2010.

[43] D. Ziou and S. Tabbone. Edge detection techniques - an overview. *International Journal of Pattern Recognition and Image Analysis*, 8:537–559, 1998.