Image Deblurring and Denoising with Various Fidelity Terms and Regularizers

Kelsey Maass, Samuel Rudy, Kevin Mueller, and Riley Molloy
University of Washington

Abstract

Image deblurring and denoising is often formulated as a regularized least-squares problem. In this project, we compare the performance of two different regularizers, the l_1 wavelet regularization and total variation regularization used by Beck and Teboulle in [3] and [2]. We also experiment with different robust penalty functions, such as the Huber loss, in order to restore images with noise drawn from heavy-tailed distributions. Results are obtained using variants of the FISTA algorithm.

1 Introduction

Image denoising and deblurring problems can be modeled with the linear system

$$Ax + w = b, \tag{1}$$

where b is the observed image, w is some unknown noise, and x is the true image we would like to recover. For a blurred image we let A represent the blurring process, and for noisy images we let A be the identity matrix. For many inverse problem applications the least squares approximation,

$$\hat{x}_{LS} = \arg\min_x \|Ax - b\|_F^2, \tag{2}$$

gives a reasonable solution. Unfortunately, this approach doesn't provide any new information for the denoising problem, and for image deblurring problems A is often ill-conditioned, so the solution is contaminated by round-off error and amplified noise [1]. To illustrate this, we consider the pepper image in Figure 1. If we blur the image and compute the naive solution $x = A^{-1}b$, the result is obscured by round-off error in the form of ringing artifacts. The result is even worse when we add noise.

In order to stabilize the solution, a variety of regularizers can be used which exploit features of the true image such as smoothness or sparsity in a wavelet domain. This results in the problem formulation

$$\hat{x} = \arg\min_x \|Ax - b\|_F^2 + \lambda R(x), \tag{3}$$

where the parameter $\lambda > 0$ is chosen to balance the tradeoff between fidelity to the model and the assumed feature. In this paper we compare two choices for the regularization term: l_1 wavelet regularization with $R(x) = \|Wx\|_1$, and total variation regularization with $R(x) = TV(x)$, which are explored by Beck and Teboulle in [2] and [3] respectively.

It is also known that the quadratic penalty is extremely sensitive to outliers. While this does not pose problems for images with Gaussian noise, it becomes an issues for images with noise from heavy-tailed distributions, such as the Student's t-distribution, which could result from a dirty camera lens. Therefore we also consider different choices for the fidelity term which are more robust to outliers, including the Huber norm

$$h_\gamma(x) = \min_y \frac{1}{2}\|x - y\|_F^2 + \gamma\|y\|_1, \tag{4}$$

and the function

$$g_\gamma(x) = \frac{1}{\gamma}\sum_i \log\left(\cosh\left(\gamma x_i\right)\right). \tag{5}$$

For the sake of completeness we also wanted to experiment with the 1-norm penalty but were not able to implement it due to the lack of smoothness. Instead, we tried using a smooth approximation to the 1-norm. The function we came up with was $\gamma^{-1}\log(\cosh(\gamma x))$ seen in figure 2 with $\gamma = 5$, which uniformly converges to the absolute value function as γ tends towards positive infinity. Further, the Lipschitz constant of the gradient only increases linearly with γ. The 1-norm could also have been approximated by the Huber loss via taking $\gamma \to 0$ while simultaneously adjusting λ to effectively increase the slope of the fidelity function. This presents other challenges however. For instance it is highly nontrivial to tune the parameters of the algorithm and doing so while also worrying about taking two limits would further complicate an already challenging problem. The behavior of $\gamma^{-1}\log(\cosh(\gamma x))$ away from the origin does not significantly change with γ and hence parameter tuning for the near L^1 case is simpler than with Huber.

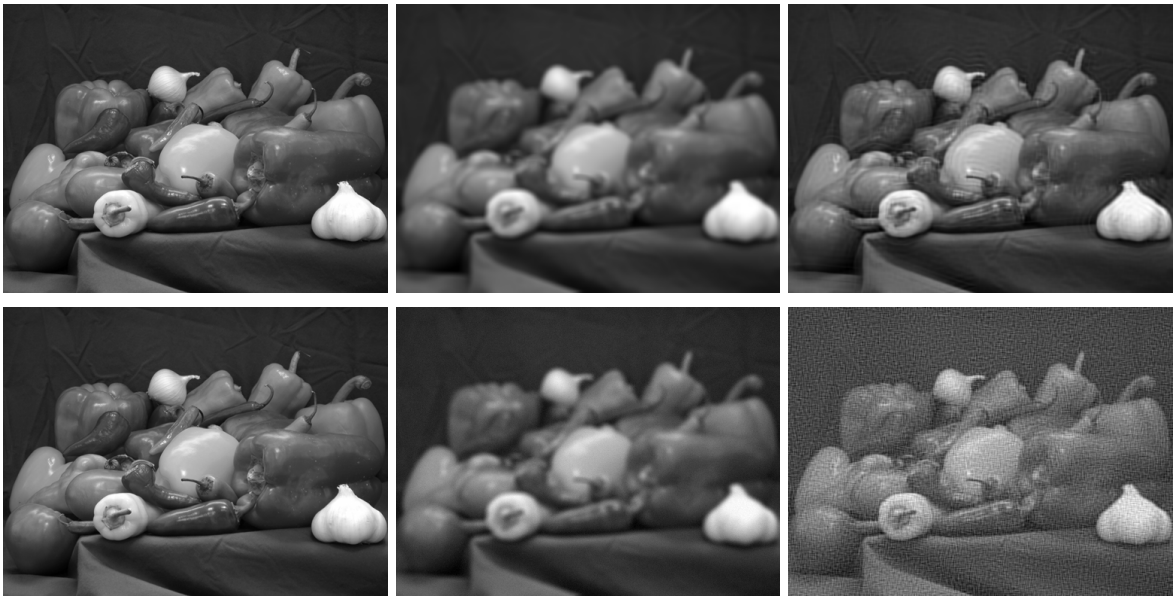

Figure 1: In the top row we observe the naive solution applied to a blurred image, including the true image x (left), the blurred image b (middle), and the recovered image $A^{-1}b$. On the bottom row, we see the effects of adding noise illustrated by the true image x (left), the blurred and noisy image $b-w$ (middle), and the recovered image $A^{-1}(b-w)$.

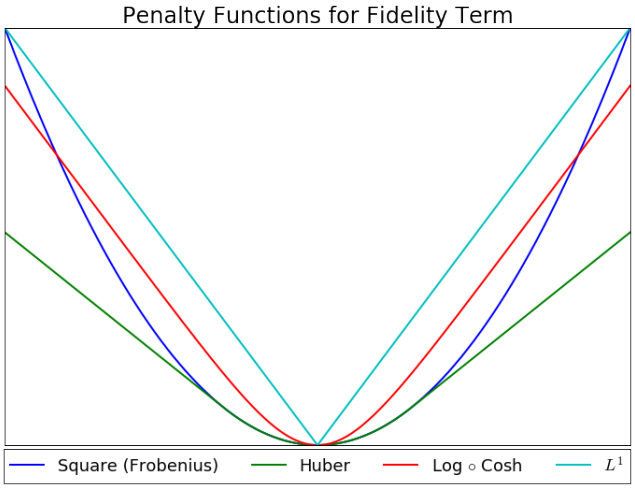

Figure 2: The various penality functions we utilize in our fidelity term, and their comparision to the l_1 norm.

2 Problem Formulation

The general approach to the image deblurring and denoising problem can now be stated as a minimization problem of the form

$$\min_x f(Ax - b) + \lambda R(x), \tag{6}$$

where Ax is a discrete convolution of the true image with a blur kernel, the *fidelity term* $f(Ax - b)$ measures how well the recovered image complies with the linear model, and $R(x)$ is our chosen *regularization*. Furthermore, if we assume that the pixels of our image satisfy $x_{i,j} \in [0, 1]$, we can add an additional term to the objective function

$$L_b(x) = f(Ax - b) + \lambda R(x) + \delta(x|[0, 1]), \tag{7}$$

where $\delta(x|[0, 1])$ is the indicator function for the set $[0, 1]$.

2.1 Blur Operators

The blurring matrix A is built from two ingredients: the blur kernel, which specifies how each pixel spreads out in the blurred image, and the boundary conditions, which specify our assumptions about the scene just outside of our observed image. For our examples, we use a 9×9 Gaussian blur kernel with a standard deviation of one, depicted in Figure 3. The Gaussian kernel is popular in blurring applications, and can be used to simulate atmospheric turbulence [1].

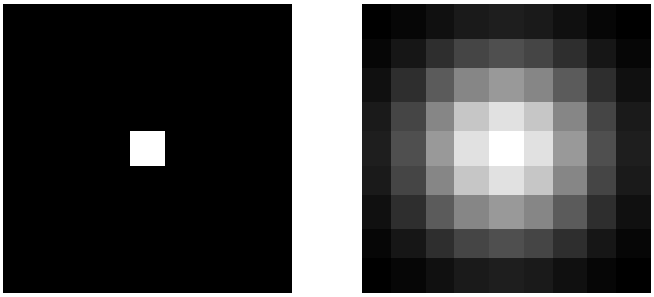

Figure 3: A single pixel before blurring (left) and after blurring (right). The image on the right is called the *point spread function*, or the blur kernel. In this example the blur kernel is computed by generating a 9×9 Gaussian with standard deviation one centered about the pixel and normalized so that all elements sum to one.

In order to blur or deblur pixels near the border of our image, we need to decide how to represent the pixels just outside of our image boundaries, which we do not have. If we assume that these pixels mirror those inside the image, then we impose reflexive boundary conditions. For spatially invariant and doubly symmetric blur kernels, these boundary conditions result in a blurring matrix that can be diagonalized by the orthogonal two-dimensional discrete cosine transform

$$A = C^T \Lambda C, \tag{8}$$

where the eigenvalues of A are determined by the blur kernel. There are two advantages of representing the blur operator in this way. First, we do not have to construct the matrix explicitly. Instead, we can take advantage of MATLAB's efficient discrete cosine transform function to compute our blurred image:

$$Ax = C^T \Lambda C x = \texttt{idct2}(\Lambda \texttt{dct2}(x)) = b. \tag{9}$$

Second, when our fidelity function is given by $f(Ax - b) = \|Ax - b\|_F^2$, we can exploit the fact that the Frobenius norm is orthogonally invariant and C is isometric to rewrite the fidelity term as

$$\begin{aligned} f(Ax - b) &= \|Ax - b\|_F^2 \\ &= \|C^T \Lambda C x - b\|_F^2 \\ &= \|\Lambda C x - C b\|_F^2 \\ &= \|\Lambda \hat{x} - \hat{b}\|_F^2. \end{aligned}$$

In this case we take the Frobenius form of our transformed images $\hat{x}$ and $\hat{b}$, multiplying $\hat{x}$ by the diagonal eigenvalue matrix rather than the full blur matrix. We can also compute and store $\hat{b}$ for additional efficiency.

2.2 l_1 Wavelet Regularization

The wavelet regularization deblurring model, as seen in [3], can be written in the form of (6) as

$$\min_x f(Ax - b) + \lambda \|Wx\|_1, \tag{10}$$

where W corresponds to a given wavelet transform and $\|Wx\|_1$ is the sum of absolute values of all entries of the matrix Wx. This regularization is motivated by the fact that natural images often have a sparse representation in wavelet domains, which we can encourage in our recovered image using the l_1 norm.

To illustrate this fact, we look at the orthogonal two-dimensional Haar wavelet transform of the peppers image. First we transform the image using the spot operator `opHaar2` with one level. This results in an image where most of the elements are zero or near zero, except for those in the upper left quadrant. If we repeat this process for more times, taking the transform of the resulting corner image, we get the Haar transform of the image for five levels, which is even more sparse.

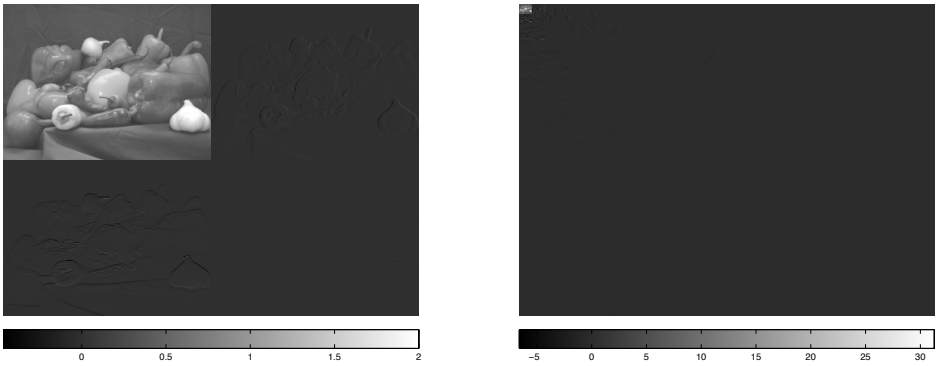

Figure 4: The 2D Haar wavelet transform of the peppers image using one level (left) and five levels (right). Note that the images are sparse.

The Haar wavelet is just the first of many wavelets developed for multi-resolution analysis. In addition to the orthogonal two-dimensional Haar wavelet transform, we therefore also consider the orthogonal two-dimensional Daubechies wavelet transform implemented with the spot operator `opWavelet2`.

2.3 Total Variation Regularization

The total variation deblurring model, as seen in [2], can be written in the form of (6) as

$$\min_x f(Ax - b) + \lambda \mathrm{TV}(x) + \delta(x|[0,1]), \tag{11}$$

where $\mathrm{TV}(x)$ is the total variation semi-norm. We include the indicator function to ensure that the resulting image is within the correct range of pixel values. Two choices exist for the TV-norm: the isotropic version and the l_1-based, anisotropic version. Since both methods yield very similar results, in this project we work exclusively with the l_1-based TV-norm, defined as

$$TV_{l_1}(x) = \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}(|x_{i,j} - x_{i+1,j}| + |x_{i,j} - x_{i,j+1}|) + \sum_{i=1}^{m-1}|x_{i,n} - x_{i+1,n}| + \sum_{j=1}^{n-1}|x_{m,j} - x_{m,j+1}|,$$

for $x \in \mathbb{R}^{m\times n}$, where reflexive boundary conditions

$$\begin{aligned} x_{m+1,j} - x_{m,j} &= 0, \text{ for all } j \\ x_{i,n+1} - x_{i,n} &= 0, \text{ for all } i \end{aligned}$$

are assumed.

The motivation behind the TV regularization is that images are often "smooth", or that most pixels will have values similar to their neighbors. Alternatively, it assumes that high frequency content such as noise increases the total variation of an image, defined as the absolute value of the gradient. To illustrate this, we

look at the absolute value of the row and column differences of the peppers image. Here we see that most of the values are indeed small, except for those which represent edges.

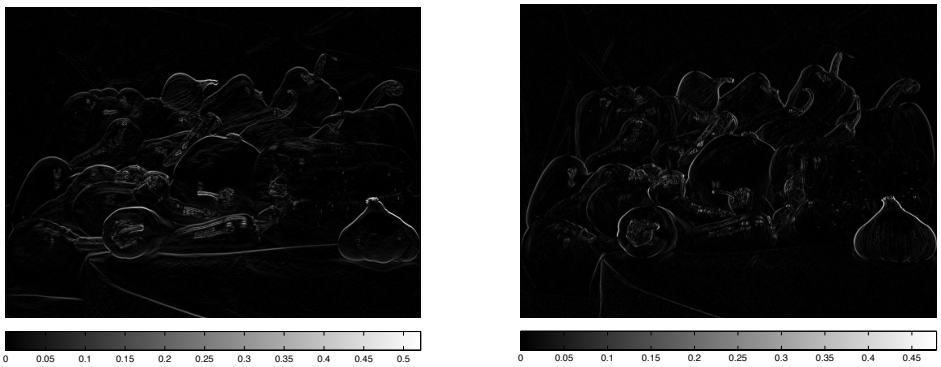

Figure 5: The absolute value of the column (left) and row (left) differences of the peppers image. Note that most of the values are zero or near zero, with most of the variation present in the image edges.

3 Algorithms

In order to solve our deblurring and denoising problem, where our objective function is the sum of two convex functions (one smooth and one with a convenient proximal representation), we can employ the proximal gradient method. For each iteration we take a gradient step in f and apply the proximal operator of R,

$$\begin{aligned} x_{k+1} &= \text{prox}_{\alpha^{-1}\lambda R}\big(x_k - \alpha^{-1}A^T\nabla f(Ax_k - b)\big) \\ &= \min_y \frac{1}{2}\Big\|y - \big(x_k - \alpha^{-1}A^T\nabla f(Ax_k - b)\big)\Big\|_2^2 + \alpha^{-1}\lambda R(y) \end{aligned} \tag{12}$$

where $\alpha \geq Lip(\nabla f)$. For faster convergence, we use the modified version of the proximal gradient algorithm, FISTA, which utilizes information from previous iterations:

$$\begin{aligned} x_k &= \text{prox}_{\alpha^{-1}\lambda R}\big(y_k - \alpha^{-1}A^T\nabla f(Ay_k - b)\big) \\ t_{k+1} &= \frac{1+\sqrt{1+4t_k^2}}{2} \\ y_{k+1} &= x_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(x_k - x_{k-1}) \end{aligned} \tag{13}$$

We detail the implementation of FISTA for each choice of regularization term in the two sections that follow.

3.1 l_1 Wavelet Regularization

Because the wavelet transform W is an orthogonal operator, we can evaluate the proximal operator of $R(x) = \|Wx\|_1$ via the soft-thresholding operation:

$$\begin{aligned} \text{prox}_{\alpha^{-1}\lambda\|W\cdot\|_1}(x) &= \arg\min_y \frac{1}{2}\|y - x\|_2^2 + \alpha^{-1}\lambda\|Wy\|_1 \\ &= \arg\min_y \frac{1}{2}\|Wy - Wx\|_2^2 + \alpha^{-1}\lambda\|Wy\|_1 \\ &= W^T \arg\min_{Wy} \frac{1}{2}\|Wy - Wx\|_2^2 + \alpha^{-1}\lambda\|Wy\|_1 \\ &= W^T \text{prox}_{\alpha^{-1}\lambda\|\cdot\|_1}(Wx) \\ &= W^T \text{sgn}(Wx)\max(0, |Wx| - \alpha^{-1}\lambda) \end{aligned} \tag{14}$$

To deblur and denoise our images using the l_1 wavelet regularization and either the Frobenius or Huber norm fidelity term, we therefore employ the following algorithm:

FISTA(b, f, λ)
$y_1 = x_0 = b$; $t_1 = 1$
$\alpha \geq Lip(\nabla f)$
for $k = 1 : N$ **do**
$u_k = y_k - \alpha^{-1} A^T \nabla f(Ay_k - b)$
$x_k = W^T \text{sgn}(Wu_k) \max(0, |Wu_k| - \alpha^{-1}\lambda)$
$t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
$y_{k+1} = x_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(x_k - x_{k-1})$
end for
return x_N

3.2 Total Variation Regularization

Optimization of the objective function (11) is based on the proximal gradient algorithm and exploits the idea of momentum used in FISTA as well as an extra function evaluation at each iteration in order to ensure a non-increasing objective function. The proximal gradient step is given as follows,

$$\begin{aligned} x_{k+1} &= \text{prox}_{\alpha^{-1}(\lambda \text{TV}(\cdot) + \delta_{[0,1]})}(\underbrace{x_k - \alpha^{-1} A^T \nabla f(Ax_k - b)}_{u_k}) \\ &= \arg\min_z \left(\|u_k - z\|_F^2 + \alpha^{-1}\lambda TV(z) + \delta(z|[0,1]) \right) \\ &= P_{[0,1]} \left(\arg\min_z \left(\|u_k - z\|_F^2 + \alpha^{-1}\lambda TV(z) \right) \right) \end{aligned}$$

Here the learning rate α is taken to be no larger than than the multiplicative inverse of the Lipschitz constant of ∇f. Note that in order to evaluate this expression we must solve a denoising problem with input $u_k = x_k - \alpha^{-1} A^T \nabla f(Ax_k - b)$. Of particular note is that regardless of the original fidelity term we have a Frobenius norm fidelity term on the new denoising problem. This allows for the easy manipulation of the original fidelity function, so long as it is convex and has Lipschitz gradient.

The resulting denoising problem with Frobenius norm fidelity function may be solved rapidly using the method developed in [2]. The TV function is shown to have a dual representation as a trace and the relationship between trace and Frobenius norm is heavily exploited to develop a dual formulation of the denoising problem. We present their method here, starting with a few new definitions which will be necessary.

$$\begin{aligned} &\mathcal{P} = \{(p, q) \in \mathbb{R}^{(m-1)\times n} \times \mathbb{R}^{m\times(n-1)} : |p_{i,j}| \leq 1, |p_{i,j}| \leq 1\} \\ &\mathcal{L} : \mathbb{R}^{(m-1)\times n} \times \mathbb{R}^{m\times(n-1)} \to \mathbb{R}^{m\times n} \text{ such that } \mathcal{L}(p,q)_{i,j} = p_{i,j} + q_{i,j} - p_{i-1,j} - q_{i,j-1} \\ &\quad \text{for } i = 1, \ldots, m,\ j = 1, \ldots, n \text{ and } p_{0,j} = p_{m,j} = q_{i,0} = q_{i,n} = 0 \end{aligned}$$

Using these new definitions we claim without proof that the total variation functional may be written as

$$TV(x) = \max_{(p,q)\in\mathcal{P}} \text{Tr}\big(\mathcal{L}(p,q)^T x\big), \tag{15}$$

we may now exploit the relationship between trace and Frobenius norm to obtain the dual problem for Frobenius denoising using total variation regularization.

$$
\begin{aligned}
\min_{x\in[0,1]} \|x-b\|_F^2 + 2\lambda \mathrm{TV}(x) &= \min_{x\in[0,1]} \max_{(p,q)\in\mathcal{P}} \|x-b\|_F^2 + 2\lambda \operatorname{Tr}\big(\mathcal{L}(p,q)^T x\big) \\
&= \max_{(p,q)\in\mathcal{P}} \min_{x\in[0,1]} \|x-b\|_F^2 + 2\lambda \operatorname{Tr}\big(\mathcal{L}(p,q)^T x\big) \\
&= \max_{(p,q)\in\mathcal{P}} \min_{x\in[0,1]} \|x\|_F^2 + \|b\|_F^2 - 2\operatorname{Tr}\big(x^T b\big) + 2\lambda \operatorname{Tr}\big(\mathcal{L}(p,q)^T x\big) \\
&= \max_{(p,q)\in\mathcal{P}} \min_{x\in[0,1]} \|x\|_F^2 + \|b\|_F^2 - 2\operatorname{Tr}\big(x^T (b-\lambda\mathcal{L}(p,q))\big) \\
&= \max_{(p,q)\in\mathcal{P}} \min_{x\in[0,1]} \underbrace{\|x\|_F^2 + \|b-\lambda\mathcal{L}(p,q)\|_F^2 - 2\operatorname{Tr}\big(x^T (b-\lambda\mathcal{L}(p,q))\big)}_{=\|x-(b-\lambda\mathcal{L}(p,q))\|_F^2} - \|b-\lambda\mathcal{L}(p,q)\|_F^2 + \|b\|_F^2 \\
&= \max_{(p,q)\in\mathcal{P}} \min_{x\in[0,1]} \|x-(b-\lambda\mathcal{L}(p,q))\|_F^2 - \|b-\lambda\mathcal{L}(p,q)\|_F^2 + \|b\|_F^2
\end{aligned}
$$

Here the switching of min and max is permitted due to the convexity of the objective in x and concavity in p and q [2]. Note that the problem of maximizing over x is simply the projection of each index of $b-\lambda\mathcal{L}(p,q)$ onto the set $[0,1]$. This gives the optimality condition for x in terms of p and q which we plug back in to obtain the dual problem.

$$
\begin{aligned}
(p^*,q^*) &= \arg\max_{(p,q)\in\mathcal{P}} \big\|P_{[0,1]}(b-\lambda\mathcal{L}(p,q)) - (b-\lambda\mathcal{L}(p,q))\big\|_F^2 - \|b-\lambda\mathcal{L}(p,q)\|_F^2 + \|b\|_F^2 \\
x &= P_{[0,1]}(b-\lambda\mathcal{L}(p^*,q^*))
\end{aligned}
$$

Note that in the dual problem the objective is Lipschitz differentiable. We claim without proof that the gradient has Lipschitz constant $L \le 16\lambda^2$ and refer the reader to [2] for proof. This smoothness allows us to implement a projected gradient algorithm.

We implement the above optimization scheme by using a slightly optimized method referred to as Monotone FISTA or MFISTA. In FISTA, the objective function values are not guaranteed to be nonincreasing, so MFISTA requires the evaluation of the objective function to check for monotonicity. Convergence plots shown in [2] indicate that MFISTA offers substantial improvements for images with substantial amounts of noise. We also implement a sped-up projected gradient algorithm using the same momentum technique as in FISTA. This algorithm is labeled as FGP, for fast gradient projection.

MFISTA(b, f, λ)
- $y_1 = x_0 = b$; $t_1 = 1$
- $\alpha \ge Lip(\nabla f)$
- **for** $k = 1 : N$ **do**
  - $u_k = y^k - \frac{A^T\nabla f(Ay_k - b)}{\alpha}$
  - $z_k = FGP(u_k, \frac{\lambda}{2\alpha})$
  - $x_k = \underset{x\in\{x_{k-1}, z_k\}}{\operatorname{argmin}} L_b(x)$
  - $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
  - $y_{k+1} = x^k + \frac{t_k}{t_{k+1}}(z_k - x_k) + \frac{t_{k-1}}{t_{k+1}}(z_k - x_k)$
- **end for**
- **return** x_N

FGP(b, λ)
- $(r_{ij}^1, s_{ij}^1) = (p_{ij}^0, q_{ij}^0) = 0$; $t_1 = 1$
- **for** $k = 1 : N$ **do**
  - $(p_k, q_k) = P_{\mathcal{P}}\left((r_k, s_k) - \frac{\mathcal{L}^T P_{[0,1]}(b-\lambda\mathcal{L}(r_k,s_k))}{8\lambda}\right)$
  - $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
  - $(r_k, s_k) = (p_k, q_k) + \frac{t_k - 1}{t_{k+1}}(p_k - p_{k-1}, q_k - q_{k-1})$
- **end for**
- **return** $P_{[0,1]}(b - \lambda\mathcal{L}(p_N, q_N))$

3.3 Choosing parameters

Before we can restore a corrupted image, we must first set our objective function parameters λ and γ. For our results in the in next section we tried various parameter values and chose those that resulted in the smallest error from the true image, $\|x - \hat{x}\|_F$, which seemed to correspond to the best looking image. Unfortunately, in most applications the true image is not known, so parameters must be chosen by visual inspection. This is demonstrated in Figure 6 below, where we vary λ while we deblur and denoise the image with Student's t noise with one degree of freedom and a standard deviation of 0.0001 and the Frobenius norm penalty function.

We see that smaller values of λ result in underregularization: the image is successfully deblurred, but the sensitivity of the Frobenius norm to outliers results in star-like artifacts. As we increase λ these artifacts diminish, but we pay the price in terms of overregularization. For the Haar wavelet basis this results in pixelation, while for the Daubechies wavelet basis the image appears blurred or smudged. It is clear from looking at the images that $\lambda = 0.01$ does the best job at balancing fidelity and regularization, in fact returning the lowest error.

Unfortunately, the final objective function value does not seem to be a good indicator for image quality, as it increases with the amount of regularization applied regardless of how well the image is actually restored. Therefore hand-tuning the parameters seems to be the only choice when the true image is not known.

Error: $\|x - \hat{x}\|_F$

	λ = 1e-4	λ = 1e-3	λ = 1e-2	λ = 1e-1
H	50.4436	22.2812	10.8628	15.8965
D	50.0977	20.8619	7.6839	12.9139

Objective Function: $\frac{1}{2}\|Ax - b\|_F^2 + \lambda\|Wx\|_1$

	λ = 1e-4	λ = 1e-3	λ = 1e-2	λ = 1e-1
H	3.0954	9.3127	52.2841	379.5130
D	3.0598	8.8598	48.0596	350.2697

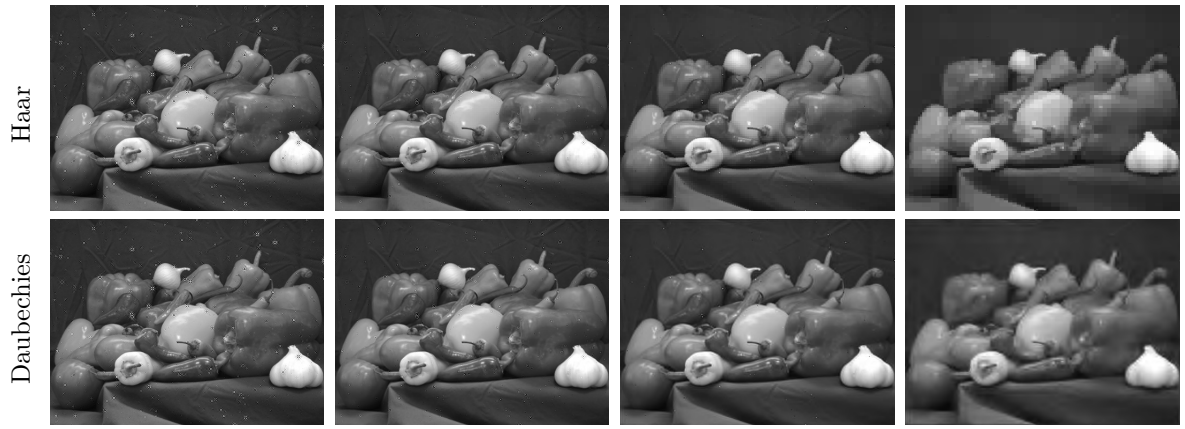

Figure 6: Results obtained from debluring and denoising the peppers image with Student's t noise, the Frobenius norm fidelity function, and l_1 wavelet regularization. As we vary λ, we note how under- and overregularization affects image quality for our two wavelet bases.

4 Examples

In this section we compare the performance of our different fidelity term penalty functions and regularizers. In each example the peppers image is first blurred with a 9×9 Gaussian kernel with standard deviation 1, and then either Gaussian noise or Student's t noise is added, both with standard deviation 0.01. Additionally, we determined how many iterations to run each algorithm based on convergence criterion rather than using a fixed number of iterations. For our examples, we stopped iterating once the objective function failed to decrease by more than 0.1% of its previous value. When testing the total variation algorithm the objective function often remained constant due to the structure of the MFISTA algorithm so the break condition was restricted to the case in which decay of the objective function was both sufficiently small and non-zero.

4.1 l_1 Wavelet Regularization: Gaussian Noise

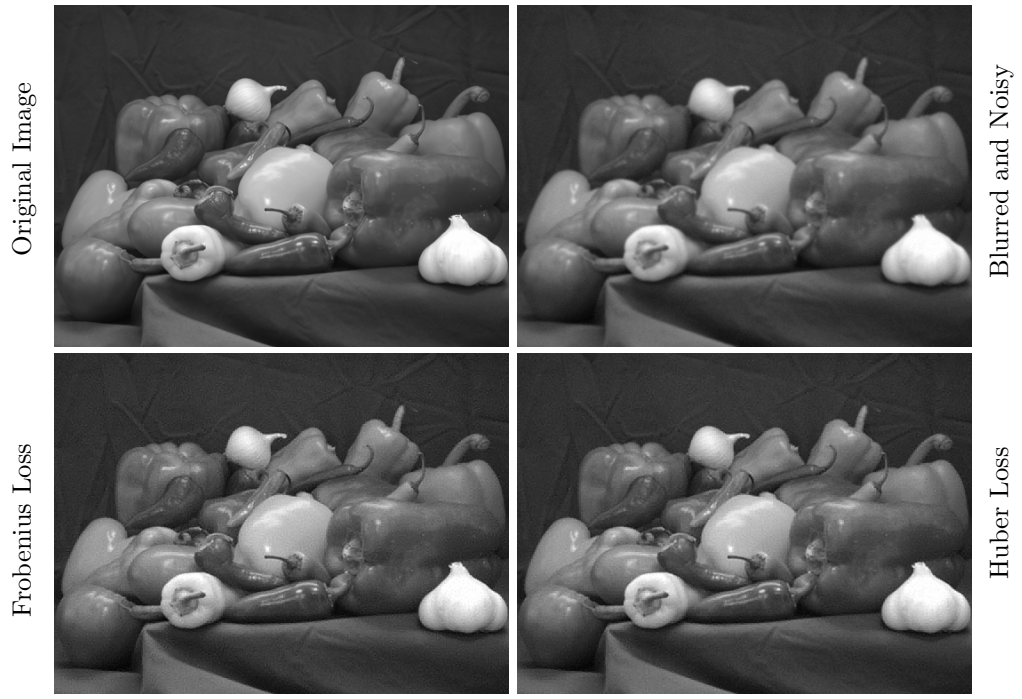

Figure 7: Images with Gaussian noise restored with the l_1 Haar wavelet regularizer. The Frobenius loss formulation uses $\lambda = 0.001$, while the Huber loss formulation uses $\lambda = 0.001$ and $\gamma = 0.01$.

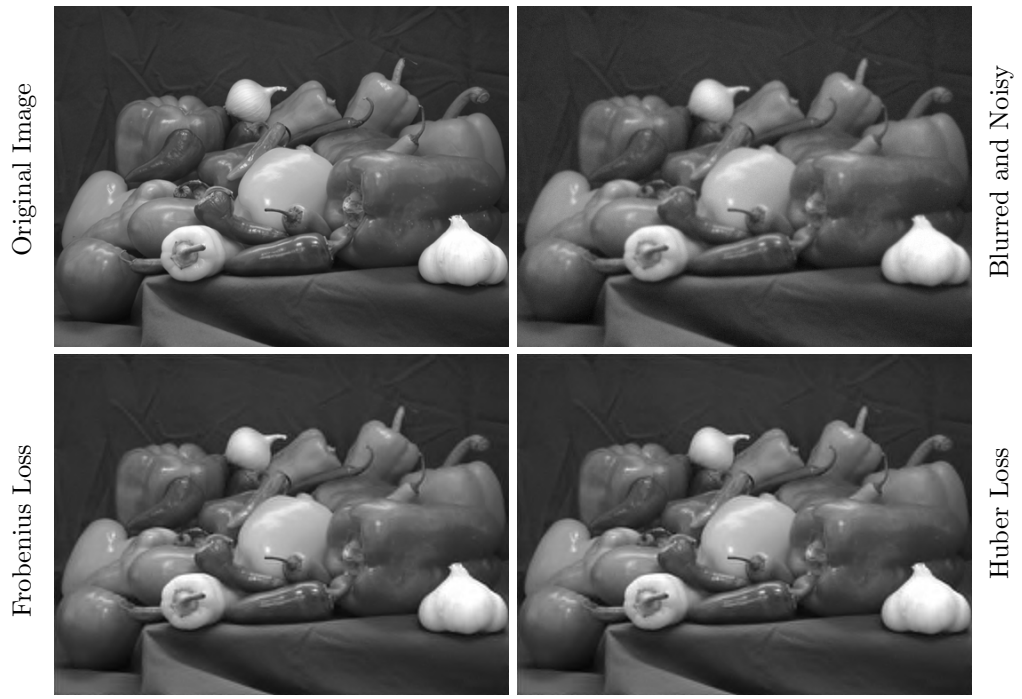

Figure 8: Images with Gaussian noise restored with the l_1 Daubechies wavelet regularizer. The Frobenius loss formulation uses $\lambda = 0.1$, while the Huber loss formulation uses $\lambda = 0.01$ and $\gamma = 0.01$.

4.2 l_1 Wavelet Regularization: Student's t Noise

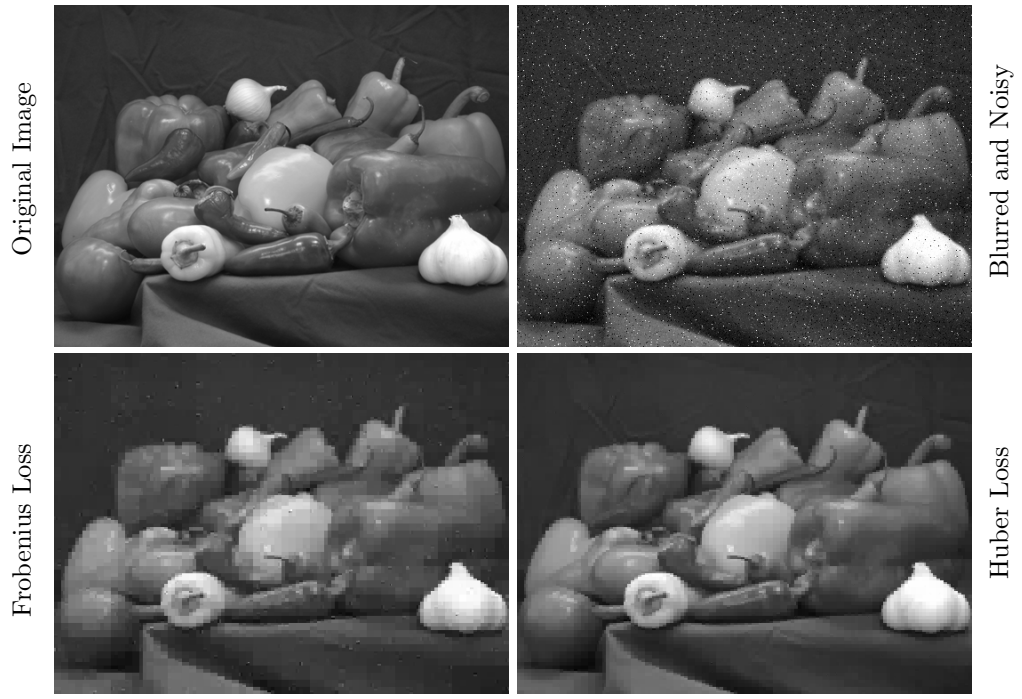

Figure 9: Images with Student's t noise restored with the l_1 Haar wavelet regularizer. The Frobenius loss formulation uses $\lambda = 0.01$, while the Huber loss formulation uses $\lambda = 0.001$ and $\gamma = 0.01$.

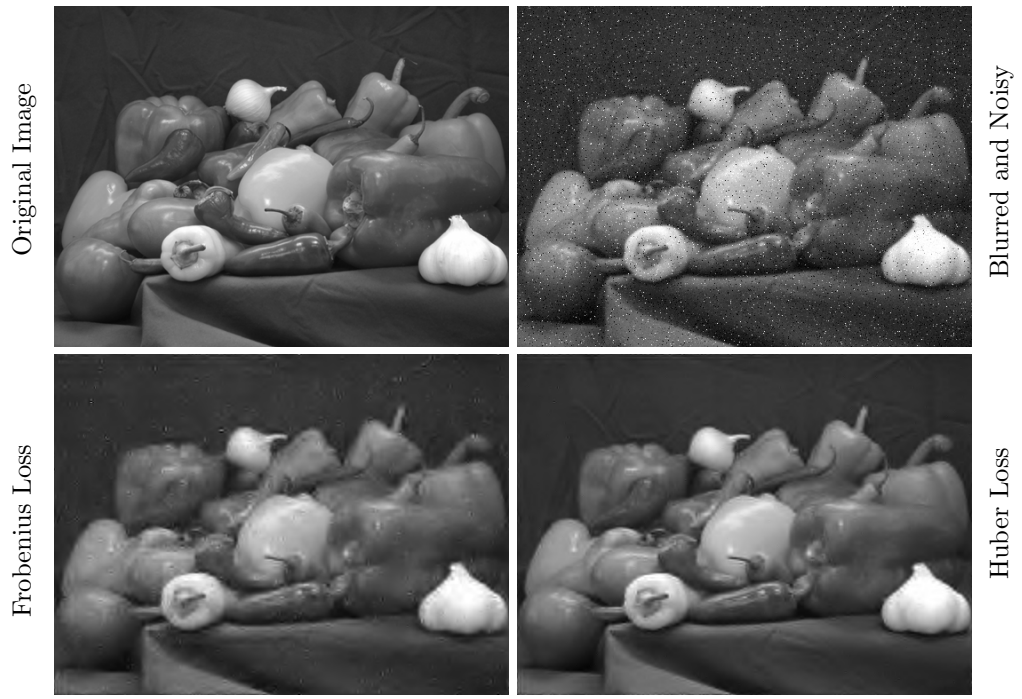

Figure 10: Images with Student's t noise restored with the l_1 Daubechies wavelet regularizer. The Frobenius loss formulation uses $\lambda = 0.1$, while the Huber loss formulation uses $\lambda = 0.01$ and $\gamma = 0.01$.

4.3 Total Variation Regularization

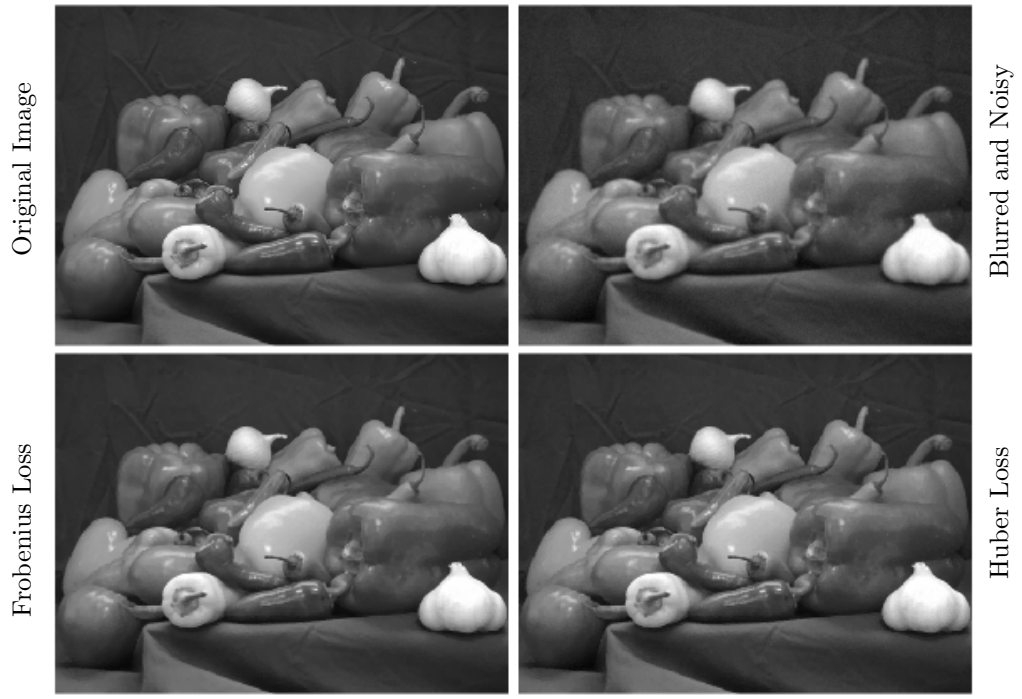

Figure 11: Images with Gaussian noise restored with the total variation regularization. The Frobenius loss formulation uses $\lambda = 0.001$, while the Huber loss formulation uses $\lambda = 0.001$ and $\gamma = 0.02$.

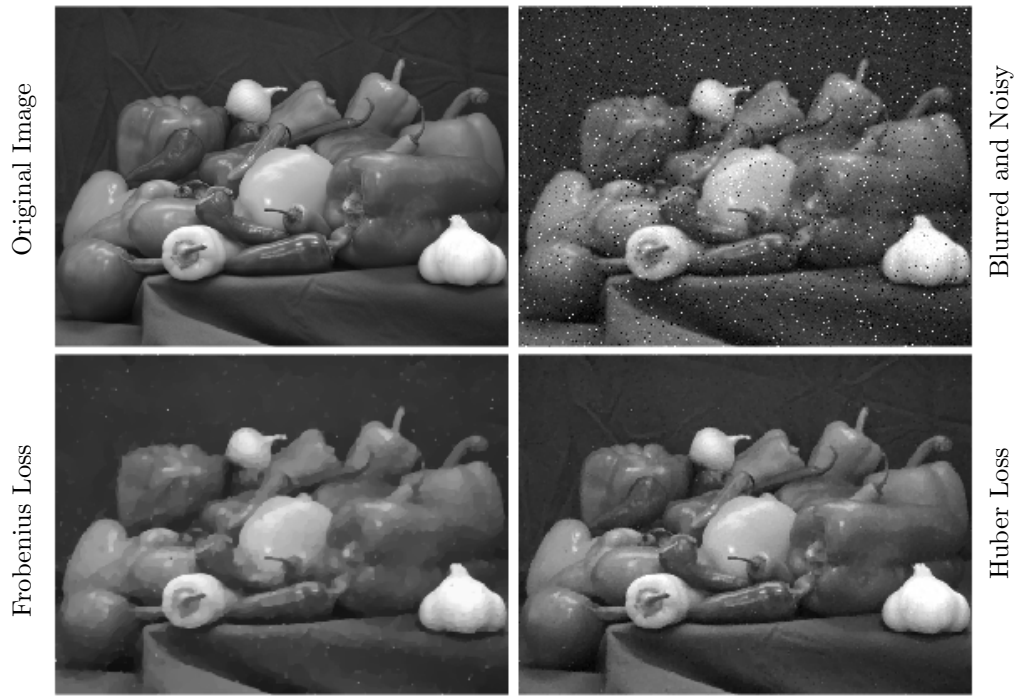

Figure 12: Images with Student's t noise restored with the total variation regularization. The Frobenius loss formulation uses $\lambda = 0.002$, while the Huber loss formulation uses $\lambda = 0.002$ and $\gamma = 0.02$.

4.4 The $\log \circ \cosh$ Penalty Function

Recall that for large values of γ the function $\gamma^{-1} \log(\cosh(\gamma \cdot))$ uniformly converges to the absolute value function. Since it is also Lipschitz differentiable we are able to use it in our algorithm for TV regularized denoising and deblurring with high values of γ to approximate how the standard L^1 norm would perform without needing to worry about how to optimize it. Figure 13 compares the effectiveness of $\log \circ \cosh$ with Huber for the denoising and deblurring problem with Student's t noise.

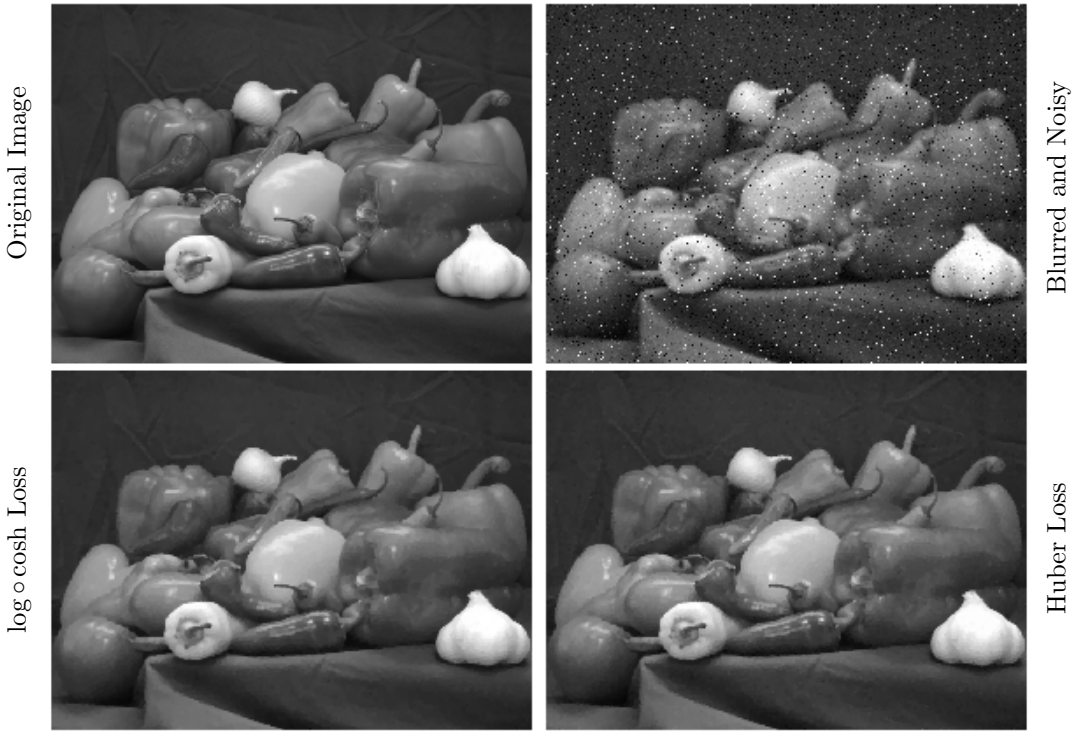

Figure 13: A comparison between the $\log \circ \cosh$ and Huber fidelity terms

5 Discussion and Conclusions

From our results, it appears that for Gaussian noise the Frobenius norm and Huber norm produce similar results, but the Huber norm outperforms the Frobenius norm when Student's t noise is present. In Figures 9, 10, and 12, we see that the images produced using the Frobenius norm fidelity term demonstrate over-regularization, resulting in either pixelation, smudges, or a painted effect depending upon the regularizer. Furthermore, all of the regularizers perform fairly well for Gaussian noise, with the total variation regularizer producing the clearest image overall, and the Daubechies wavelet regularizer producing a better image than the Haar wavelet regularizer.

In Figure 13 we compare the $\log \circ \cosh$ penalty to the Huber with Student's t noise, where it appears to perform slightly better in terms of edge resolution and noise reduction. However, their performance may be reversed for other types of noise. For example, we may expect $\log \circ \cosh$ to over-penalize low magnitude differences in the case of more tightly distributed (e.g. Gaussian) noise. Another caveat is that since the Lipschitz constant of $\nabla \gamma^{-1} \log(\cosh(\gamma x)) = \tanh(\gamma x)$ is equal to γ, the step size in the MFISTA algorithm decreases as we increase γ, thereby increasing the computational expense for this penalty function. Timing information was not collected for this method, but there did not appear to be a significant increase in the number of iterations needed for convergence for the examples we implemented.

Different pictures, different amounts of noise...

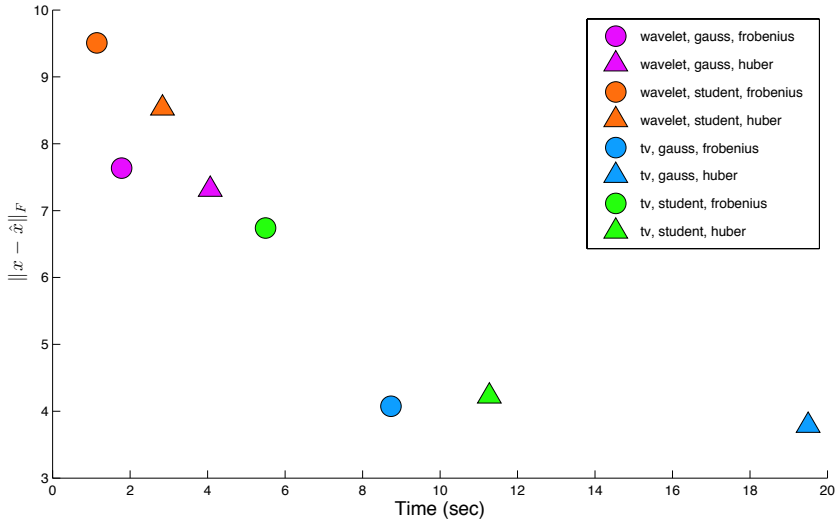

Figure 14: Frobenius norm error between the true and observed image plotted against time for relative change in the objective function less than .001.

Figure 14 shows a rough comparison of our methods by displaying the time complexity and Frobenius error for different fidelity functions, regularization techniques and noise-types. It is clear that TV regularization outperformed wavelet regularization at the price of additional computational complexity. This is not surprising due to TV's extra requirement of solving an inner optimization problem for denoising the image. It can also be seen that student noise generally required less time than Gaussian noise but at the expense of higher error. Finally, applying the Huber for student noise improved results drastically for a slight increase in computation time.

In conclusion, we present a survey of regularization and fidelity functions for the image denoising/deblurring problem. In particular, we compare regularization of the 1-norm in the wavelet domain to TV regularization. Furthermore we explore the affect of using the Huber and $\log \circ \cosh$ as fidelity functions for potential improvements when applied to heavy tailed noise. Our results show that TV regularization performs better for high amounts of noise, but the wavelet domain is well suited for smaller amounts of noise due to computation speed improvements.

In our research we came across many challenges and avenues for future work. It is clear from figure 6, that the correct choice of the regularization parameter is important for maximizing results but varies for different classes of images. Therefore, there is a high need for finding a correct method for attaining the regularization parameter across images. Additionally, finding an appropriate error metric can be a difficult challenge when quantitatively analyzing performance for the methods shown. The Frobenius norm of the error is the most common metric, but can do a poor job of correctly evaluating the improvements from denoising and deblurring algorithms due to the large structural changes of the images.

References

[1] J.G.Nagy P.C. Hansen and D.P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering*. SIAM, Philadelphia, 2006.

[2] A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE Transactions on Image Processing*, 18(11):2419–2434, November 2009.

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal of Imaging Sciences*, 2(1):183–202, 2009.

All of our code can be found on github.
The spot operators used to implement blur operators and wavelet transforms can be found here.