Scale recognition, regularization parameter selection, and Meyer's G norm in total variation regularization *

David M. Strong † Jean-François Aujol ‡ Tony F. Chan §

Abstract

We investigate how TV regularization naturally recognizes scale of individual features of an image, and we show how this perception of scale depends on the amount of regularization applied to the image. We give an automatic method driven by the geometry of the image for finding the minimum value of the regularization parameter needed to remove all features below a user-chosen threshold. We explain the relation of Meyer's G norm to the perception of scale, which provides a more intuitive understanding of this norm. We consider other applications of this ability to recognize scale, including the multiscale effects of TV regularization and the rate of loss of image features of various scales as a function of increasing amounts of regularization. Several numerical results are given.

1 Introduction

Consider the problem of denosing or filtering a noise-contaminated or otherwise degraded (but not blurred) image in $\mathbf{R}^n$: given a measured image $u_0(\vec{x})$, find an approximation $u(\vec{x})$ to the true image $u_{true}(\vec{x})$, where $u_0 = u_{true} + \eta$, and where $\eta(\vec{x})$ is the noise or other degradation in the image. Typically our goal is to recover the true image u_{true} as exactly as possible and/or to find a new image u in which the information of interest is more obvious and/or more easily extracted.

1.1 Total variation regularization in image processing

Rudin, Osher and Fatemi (ROF) proposed [26] to modify a given image by decreasing the total variation

$$TV(u) \equiv \int |\nabla u(\vec{x})| \, d\vec{x} \tag{1}$$

in the image while preserving some fit to the original data u_0. Equation (1) is typically referred to as the *total variation* or *bounded variation* seminorm of u. There are two common formulations of this problem as solved by the ROF model: the unconstrained or Tikhonov formulation [31],

$$\min_u \ \tfrac{1}{2}\|u - u_0\|^2 + \alpha \, TV(u) \ , \tag{2}$$

and the noise-constrained problem,

$$\min_u TV(u) \quad \text{subject to} \quad \|u - u_0\|^2 = \sigma^2, \tag{3}$$

where the error or noise variance σ^2 is assumed to be known (e.g. Gaussian noise). As shown in [13], solving (2) is equivalent to solving (3). In this paper we consider primarily the unconstrained formulation (2). We note that throughout this paper $\|\cdot\| \equiv \|\cdot\|_{L^2}$, unless otherwise noted.

*This work was supported by grants from the NSF under contracts DMS-9973341, ACI-0072112, INT-0072863, the ONR under contract N00014-03-1-0888, the NIH under contract P20 MH65166, and the NIH Roadmap Initiative for Bioinformatics and Computational Biology U54 RR021813 funded by the NCRR, NCBC, and NIGMS.

†Department of Mathematics, Pepperdine University, David.Strong@pepperdine.edu

‡Centre de Mathématiques et de Leurs Applications, ENS Cachan, aujol@cmla.ens-cachan.fr; work done while visiting UCLA Department of Mathematics

§Department of Mathematics, UCLA, chan@math.ucla.edu

In (2), larger values of α result in more regularization (and thus less total variation) and less goodness of fit of u to the original data u_0, as illustrated in Figure 1. Although TV regularization was originally introduced for deblurring and denoising grayscale images, it has subsequently been employed in a variety of other image processing tasks such as denoising color or other vector-valued images [10], blind deconvolution [16], segmentation [32], inpainting [15], image decomposition [8, 23], and upsampling [2].

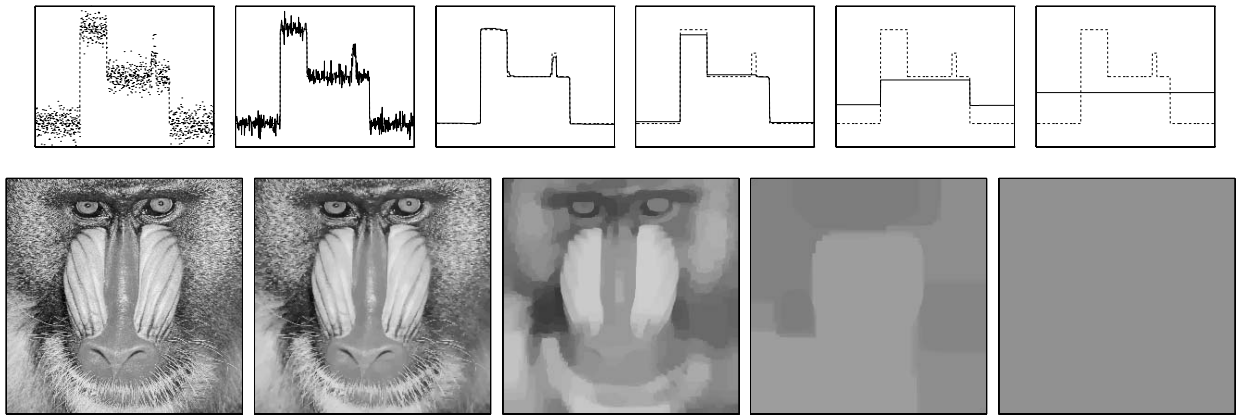


Figure 1: Results of TV regularization of a simple noisy $\mathbf{R}^1$ function (top row) and the standard Mandrill image (bottom row) when using different values of α in solving (2). For the $\mathbf{R}^1$ function, the first plot shows the true and noisy images. For the Mandrill image, the first image is the original (noise-free) image. For both, the subsequent figures are the results of solving (2) using $\alpha = 0.0001, 0.001, 0.01$ and 0.1 (and $\alpha = 1.0$ for the $\mathbf{R}^1$ function), respectively.

1.2 Scale recognition and choice of regularization parameter

Scale is important in both understanding and manipulating an image. At present the effects of TV regularization—in particular, how these effects relate to the scale of the various image features—are only partially understood. Additionally, how to choose the regularization parameter α when solving (2) is often done haphazardly or experimentally. In contrast, a variety of researchers have more thoroughly investigated other aspects of TV regularization, such as existence and uniqueness of solutions, development and convergence analysis of numerical schemes, and the basic effects of TV regularization on an image. A representative sampling of the literature includes [1], [4], [9], [13], [14], [17], [18], [19], [21], [26], [29] and [34].

If there is some regularity to the noise and if the noise level is known, then we might use (3) to solve the TV regularization problem (and the choice of α in (2) would be inherent). Otherwise, we must in some intelligent way choose a value for α. In the past, this has been done more by trial and error rather than by any well understood theory. While this might result in (indeed, the choice of α is still often driven by) an image which "looks nice," it is generally unclear precisely how the image itself has been affected. Even in the case of known noise level and type, we may want to choose α based on criteria other than trying to match a noise constraint. Additionally, we may want to apply regularization to a noise-free image in order to more easily extract the desired information from the image, e.g. in segmentation.

There are existing methods for estimating a "good" value of α, such as generalized cross validation and the L-curve. See Vogel's survey of regularization parameter selection methods in [33], and further details in the papers he references. These methods for finding α are typically based on minimizing a certain functional or estimating the "corner" point of the L-curve. That is, they are based on numerics. In contrast, the approach we propose in this paper is driven by the geometry of—in particular, the scales present in—the image to regularize. It is clear that it would be helpful to have a more automatic, reliable and geometry-based approach for choosing α. Also, applying TV regularization would be an even more mathematically sound and predictable approach to image processing if we better understood how the original image has been changed, particularly with respect to scale, in producing the regularized image.

1.3 Assumptions

In this paper, we choose the image domain to be the unit square $[0,1]$ x $[0,1]$ as we prefer to have the scale of image features be consistent, regardless of the discretization (resolution) of the image. In the Appendix we give a brief discussion on how the value of α is affected by the choice of domains. Also, in our numerical examples all images in $\mathbf{R}^2$ are grayscale and, again for consistency, have been normalized so that the minimum and maximum image intensity values (prior to addition of noise and/or regularization) are 0 and 1, respectively.

1.4 Outline

In Section 2 we discuss how TV regularization naturally perceives scale in an image, including how this perception changes with increasing amounts of regularization (larger values of α in solving (2)) applied to the image. The main contributions of this paper are given in Sections 3 - 5. In Section 3, we motivate and give an algorithm for determining the minimum value of α in (2) that will result in the removal of all features of scale equal to or smaller than any given threshold. Section 4 is devoted to relating Meyer's G norm to scale, to some degree a consequence of the algorithm given in Section 3, which gives us new insight and a more intuitive understanding of this norm. In Section 5 we give several numerical results of this algorithm. Finally, in Section 6, we begin to explore additional ways to employ TV regularization's ability to recognize scale, including to better understand both the multiscale effects of TV regularization and the rate at which features of any given scale disappear from an image as a function of increasing α. Conclusions and other final remarks are given in Section 7.

2 Scale, as perceived by TV regularization

In this section we further develop and discuss the notion of scale introduced in [29]. We show how TV regularization naturally recognizes scale and how perception of scale varies with α. We denote by Ω the domain of the image. In general, we assume that Ω is a bounded connected open set. In subsequent numerical examples, Ω is the unit square.

2.1 A geometric definition of scale

This paper relies on results from [29] in which Strong and Chan analyzed the effects of TV regularization on a discrete (e.g. digital) image. If u_0 is the original image, and u the image resulting from TV regularization, then the intensity change δ in the image due to regularization at position $\vec{x}$ is defined as

$$\delta(\vec{x}) = |u(\vec{x}) - u_0(\vec{x})| \,. \tag{4}$$

The intensity change will always be in the direction that reduces contrast between adjacent image features, as seen in Figure 1, for example. As shown in [29], there are two fundamental properties of TV regularization:

1. Edge locations of image features tend to be preserved, and under certain conditions, as described in [29], are preserved exactly.

2. The intensity change δ experienced by an individual constant-valued image feature $E \subset \Omega$ (i.e. the feature is a characteristic function on E) is inversely proportional to the *scale* of that feature,

$$\delta(\vec{x}) = \frac{\alpha}{scale(\vec{x})}, \tag{5}$$

where we define

$$scale(\vec{x}) = \frac{|E|}{|\partial E|}. \tag{6}$$

for $\vec{x} \in E$.

This results are exact for radially symmetric, piecewise constant image features (e.g. constant-valued circles). The validity of these results in the more general case are discussed in a bit more detail in [29]. Also, in the latter part of Section 2.3, we further study (experimentally) a few more general cases in which this theory no longer precisely describes the behavior of TV regularization.

The notion of scale described above arises naturally in TV regularization, as described in [29], in which the authors consider primarily piecewise constant images. This, of course, limits the generality of the approach. But it appears, in fact, that it is an approximation of a seemingly unrelated result by Strang [27] from over two decades ago in which the scale of an object is defined as basically the ratio of an area divided by a perimeter. We will briefly discuss Strang's result later in Section 4 with formula (16) after we have introduced the necessary mathematical tools for our analysis. It is both interesting and important to note that the authors of [29] derived their empirical laws (5) and (6) prior of learning of the results of [27]. One important advantage of using (5) and (6) for defining scale is that they are easily applicable and implementable in practice, as we will show in this paper. The reader should be aware, however, that the theory developed in this paper implicitly deals with the case of piecewise constant images, and that for general images it is an approximation of more general and abstract definitions of scale, which nevertheless provides very interesting and useful results.

The notion of scale defined in (6) may at first be unclear if new to the reader. To make it easier to understand, we explain it for two simple examples. A circle of radius r would have $scale = \pi r^2 \,/\, 2\pi r = r/2$, and similarly the scale of a sphere is $scale = \frac{4}{3}\pi r^3 \,/\, 4\pi r^2 = r/3$. Second, a rectangle of k_1 x k_2 pixels on an n x n discretized grid of the unit square would have $scale = k_1 k_2 \;/\; 2n(k_1 + k_2)$. Consequently, a k x k square has the same scale as a rectangle of width $k/2$ and infinite length (and therefore the scale of a k x k square will always be larger than the scale of a rectangle with one side of $k/2$ or less). In general, large, blocky features have relatively large scale, while thin features—even those that are very long—have relatively small scale. This fact is related to the main results of [18], in which Dobson and Santosa use Fourier analysis to show that TV regularization is particularly suited to denoising images comprised of large, blocky features. Interestingly, a circle and square of equal diameter/width have the same scale. This is also true in higher dimensions.

There is currently an on-going discussion about other, more general—and also more mathematically abstract—ways of defining scale as perceived by TV regularization. For instance, one may define the scale of an object as being the radius of the largest ball which can be contained in the object. This is directly related to (6) if the object is simply a ball. In general, definition (6) is intuitively simpler and is practically (as opposed to theoretically) more useful than other existing notions of scale, and ultimately it makes possible the results that we give in this paper. We will further discuss this point in Section 4 with formula (16).

Property 1 above is quite significant and is a primary reason TV regularization is used in a variety of image processing applications, such as those listed at the end of Section 1.1, not to mention its potential use in applications other than image processing. Property 2 explains in a very basic way how TV regularization works: smaller-scaled features (including noise) experience large reduction in intensity relative to background and surrounding features, thus removing or nearly removing them them by "flattening" them, i.e. reducing contrast with adjacent image features, while larger-scaled features experience relatively little intensity change and are consequently left more intact. This notion of "flattening" is especially reasonable for digital images which are comprised of numerous piecewise constant features on the order of a single pixel or larger. This was seen in Figure 1. As Figure 1 also illustrates, a less than precise understanding of Property 2 can lead to undesirable results when using TV regularization.

Figure 2 is a juxtoposition of four simple but illustrative plots which show the results of TV regularization on a very basic R^1 function using four different values of α in solving (2). The plots illustrate the well known fact that change in intensity of an image feature due to TV regularization depends only on its scale, not on its original amplitude/intensity. Of course the change in intensity relative to adjacent features cannot be larger than the original intensity relative to those features, i.e. the original contrast. The plots also illustrate that greater original relative intensity requires a larger value of α in order to completely remove that feature by regularization. Further basic behavior of TV regularization, including how the scales present in an image evolve for increasing values of α, are subsequently discussed and illustrated, including in Figures 5 and 7.

For completeness, we note that as described in [28] TV regularization can be viewed as an unbiased (where biased means prefering high-contrast edges to low-constrast edges, or vice versa, as discussed in [24]) case of anisotropic diffusion, and consequently Property 2 is also one way of explaining how anisotropic diffusion

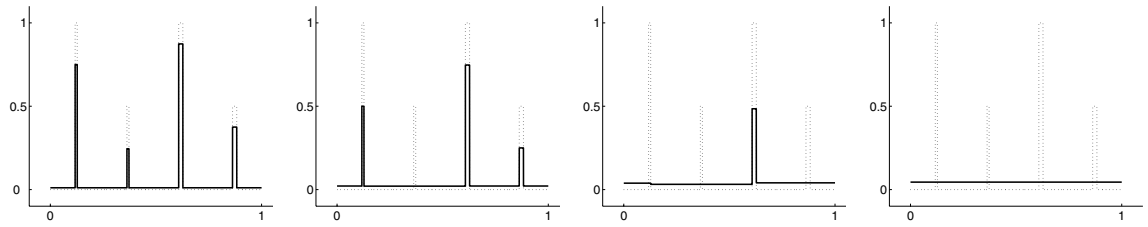


Figure 2: Change in intensity depends only on scale. In each plot, the dotted line is the original function and the solid line is the regularized function using $\alpha = 0.00125, 0.00250, 0.00500$ and 0.01000 in solving (2). The width of each of the first two features is 0.01 and of each of the last two features is 0.02, so that the scale of the last two features is double the scale of the first two. Consequently, the change in intensity for the first two features is double that of the second two, since change in intensity is inversely proportional to scale. The "background" to the sides of the four "features" also changes in intensity proportionally to its width, but in generally the change in the background is relatively negligible.

works. We also note that Bellettini, Caselles and Novaga did a related analysis [9] of TV regularization as it relates to anisotropic diffusion by considering the eigenvalue problem of $-\nabla \cdot (\frac{\nabla u}{|\nabla u|}) = u$. Finally, in [11], Brox and Weickert recently proposed to use the TV flow to compute a local measure of scale in an image.

2.2 Scale as a function of change in intensity

We further consider how (5) relates change in intensity to scale. When rewritten as

$$scale(\vec{x}) = \frac{\alpha}{\delta(\vec{x})}, \tag{7}$$

we see that scale can be viewed as a function of change in image intensity. Although simple—in fact, in part because it is so simple—this relationship is potentially very useful. Essentially what it means is that we can determine what the scales of the various image features are throughout the image by looking at how much intensity change occurs as a result of applying TV regularization to the image.

Rewriting (5) as (7) induces a slight change in the definition of scale. While the more geometric definition of scale (6) gives the scale at each location in terms of what image feature it is part of, the intensity change definition (7) defines scale at each location, e.g. at each pixel, without knowledge of surrounding features. (This simple notion can be complicated by the fact that a specific location in the image might be part of different image features of varying scales.) This is an advantage of the new formulation (7) of scale here. The definition (6) of [29] was derived for piecewise constant images, in which case each pixel can be associated with a feature of the image so that both defintions (6) and (7) are equivalent. This is illustrated with the first example given in Section 2.3. In the case of general images, they are no longer equivalent. But since TV regularization creates piecewise constant images, particularly when the image is discrete (e.g. digital), then even a small amount of regularization results in an image that is piecewise contant, and (6) and (7) are once again quite compatible. We will return to the issue of how (6) and (7) are approximations of more abstract definitions of scale in Section 4 with formula (16).

Understanding how to measure scale as perceived by TV regularization has many potential uses, including four that we investigate in this paper:

1. For any given image, we can find the smallest α needed to remove all features whose scale is less than any user-chosen scale threshold.

2. We give an intuitive explanation of Meyer's G norm by relating it to the above notion of scale.

3. We develop a better understanding of how TV regularization can be used to produce multiscale representations of images.

4. We better understand how rapidly various scales present image disappear with increasing values of α.

In this paper we will investigate the first application in detail, and to a lesser extent, the second application. We will also consider the final two applications, but we expect that our results will be just the beginning

of more investigation of these ideas. A fifth promising application is that once TV regularization has been applied, we can determine the scales of the remaining features and using (5) and (7) we can determine how much intensity was lost due to TV regularization, and add back this lost intensity to the regularized image to get a more accurate approximation u of the true image u_{true}. This fifth application turns out to be a bit more complicated than it might first seem, and consequently it is being considered in a separate paper.

2.3 Determining scale using the scale recognition probe

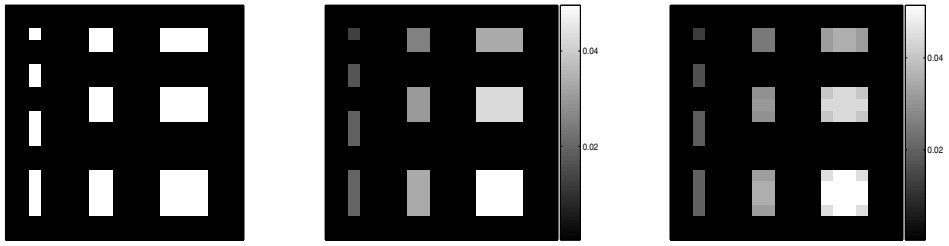
It turns out that TV regularization can be used for recognizing the scales present in an image, either the original image or a regularized image. This scale recognition is accomplished by using (7) in conjunction with performing what we refer to as a *scale recognition probe* for determining $scale(\vec{x})$ in a given image v:

Scale Recognition Probe Algorithm

1. Choose α_{probe}.
2. Find $u_{probe} = arg \min_u \frac{1}{2}||u - v||^2 + \alpha_{probe} TV(u)$.
3. Compute $\delta(\vec{x}) = |u_{probe}(\vec{x}) - v(\vec{x})|$.
4. Compute $scale(\vec{x}) = \dfrac{\alpha_{probe}}{\delta(\vec{x})}$.

To be clear, the purpose of the scale recognition probe is not to regularize an image; it is regularization done in order to determine the scale, as perceived by TV regularization, present in the image. The image to probe, denoted by v, in order to determine scale might be the measured image u_0 itself, a regularized version u of u_0, or a noise-free image.

To illustrate the scale recognition probe, we apply the above algorithm to the simple image shown in Figure 3. Image (a) is the noise-free image in which to determine scale. Image (b) is the image showing scales as predicted by (6). For example, since the domain is the unit square and the image is 20 x 20, the scale of the first object, a single pixel, is $scale = (1/20)^2 \,/\, 4(1/20) = 0.0125$. Image (c) is the image showing scales computed using the scale recognition probe. Since the scale of the background is quite large relative to the scales of the rectangles, in order to more easily see the scales of the shapes relative to each other, in both (b) and (c) we assign the background a value of 0. Since the image in (b) results from the geometric definition of scale (6) and the image in (c) results from the intensity change definition of scale (7), then the agreement between these two images illustrates the agreement between these two defintions of scales.



(a) Original image. (b) Theoretical scales. (c) Computed scales.

Figure 3: The scales of individual features as perceived by TV regularization. Image (a) is the noise-free image in which to recognize the scales of various rectangular image features. Image (b) is the image showing the scale as theoretically predicted by (6). The scales of all ten objects, from smallest to largest (top to bottom, left to right in the image) are 0.0125, 0.0167, 0.0188, 0.0200, 0.0250, 0.0300, 0.0333, 0.0333, 0.0429 and 0.0500. Image (c) is the image showing computed scales found by the scale recognition probe (using $\alpha_{probe} = 0.0001$—the value of α_{probe} should be relatively small, as discussed in Section 2.4), which is based on finding scale using (7). The computed scales in (c) found using (7) match nearly exactly the theoretically predicted scales in (b) found using (6). In both (b) and (c), we set the scale of the background to be 0 in order to better see the scales of the rectangular objects.

As seen in the image in (c), the corners of the squares and the corners and ends of the rectangles experience a slightly greater change in intensity, and thus are interpreted as having smaller scale. The definition of scale and the change in intensity (5) is exact for radially symmetric features, for example, constant-valued circles. From this point of view, the corners of each square, for example, are like smaller-scaled features attached to a larger one: like four small circles connected to each "corner" of a larger circle. Of course for discrete images, any notion of scale will be influenced by the resolution of the image, as well as the discretization strategies incorporated into the numerical schemes used to solve the TV regularzation problem. The interpretation of the results of the Scale Recognition Probe supposes clustering of the residual (equivalently, of its inverse–see Steps 3 and 4 in the algorithm above). Essentially it relies on the trend of TV regularization to yield piecewise constant images and signals, which has been repeatedly observed experimentally and shown analytically by a number of researchers.

Even in the simple example seen above, it is clear that the Scale Recognition Probe will encounter certain difficulties when used on more general images. The Scale Recognition Probe is precise only in the context of the conditions under which (5) and (6) are exactly true, which, as described earlier, is when the image features are radially symmetric and constant-valued. (In the above example, the rectangular features are constant-valued, but not radially symmetric.) We now briefly consider other examples which illustrate other cases where the results of the Scale Recognition Probe are less than precise or may even seem to begin to break down altogether.

In Figure 4 are six pairs of images: six test images and the corresponding images showing scales as found using the scale recognition probe, which in short says that $scale(\vec{x}) = \alpha \,/\, \delta(\vec{x})$. (Later in Section 5 we employ the Scale Recognition Probe is working with several real images.) The first three test images deal with other piecewise constant, but non-radially symmetric (and non-rectagular) features. The next three images compare and contrast the results of the Scale Recognition Probe on constant-valued features with its results on smooth (gradual intensity change) features.

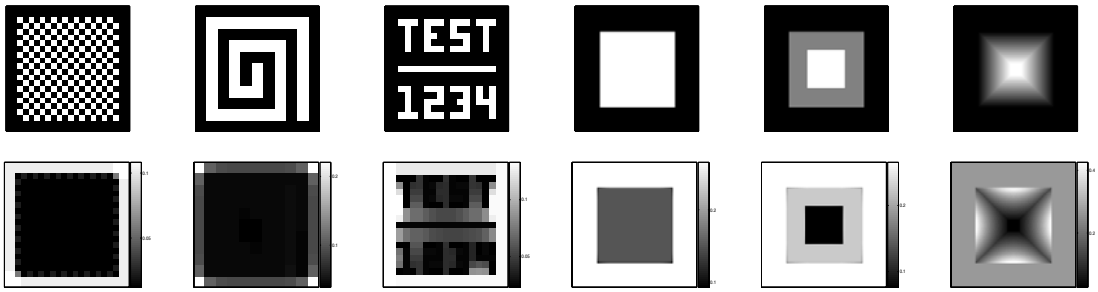


Figure 4: Test images which illustrate some of the difficulties that arise when trying to determine scale using the Scale Recognition Probe. The two main difficulties result from features comprised of various scales and of features having different (in particular, gradually changing) intensities. Both of these difficulties are ultimately inherent in the nebulous notion of scale in general and as perceived by TV regularization in particular, rather than being a weakness particular to the approach used in the Scale Recognition Probe in computing the scale.

Consider the first image, an 18 x 18 (on the unit square) checkerboard image. As seem in the bottom image of the image's computed scales, the scales of each square in the checkered pattern are correctly determined to be $scale = (1/18)^2 \,/\, 4(1/18) = 0.014$. However, the scales image is not helpful in distinguishing one square from another–we simply know that the image is full mostly of features of scale 0.014. All of the black squares surrounding the checkerboard have larger scales because they are also connected to the surrounding black background, and in particular the scale of the two black squares at the top right and bottom left are even larger because they are connected on two sides to the background. This same basic analysis is also true of the second image of black and white spirals or snakes. For the third image, "TEST 1234," as seen in the bottom scales image, the background (surrounding the writing) has uniform scale, as do the letters and numbers, numbers and the separating bar, but the same ambiguity about feature boundaries is present as was the case in the first two images. Also, there are several distinct scales present in the patches of background within individual letters or numbers as well as between different letters and/or numbers. While this might be perceived as an inaccuracy or even a breakdown of the Scale Recognition Probe, perhaps a more appropriate description would be that these different patches of background simply have different

scales, although because of the ambiguity of the notion of scale, for these patches of background we cannot really measure the accuracy of the results produced by the Scale Recognition Probe.

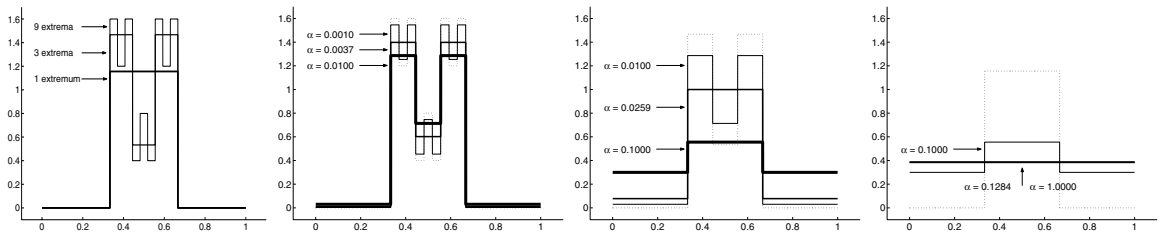
The final three test images are to compare and contrast the scale of piecewise constant and smooth (gradually changing intensity) image features. In the first image it is easy to compute the scale of this square of width 0.6 (or equivalently the largest circle that would fit inside of this rectangle): $scale = 0.6^2 \,/\, 4(0.6) = 0.150$. Similarly, since the image is on the unit square and we are using Neumann boundary conditions (so that the boundary of the image domain is not included when computing the boundary size of an image feature adjacent to the boundary) the scale of the surrounding background is $scale = (1 - 0.6^2) \,/\, 4(0.6) \approx 0.267$. These two scales are clearly seen in the bottom scales image. In the fifth image, with one square half the width of the first on top of the first square, the scale of the larger square is identical to that found in the previous image, while the scale of the smaller square is also correctly found to be $scale = 0.3^2 \,/\, 4(0.3) = 0.075$, and the scale of the background is again correctly computed to be 0.267. Even in this image, we observe the tendency of TV regularization to yield images with features that are both more piecewise constant and more radially symmetric that the original image. In the final image, we have a series of stacked squares, decreasing in width, to nearly approximate a frustum with square cross-section. The scale of the background is again found to be 0.267. However, it is now not even clear how to best define the scale of the frustum, as it is essentially comprised of several squares of varying scales stacked on top of each other. We still see that the corners of each square experience more change in intensity than at the middle of each side of each square. Not surprisingly, even though each square corner is "attached" to a successively smaller square, the corners themselves are all identical in scale. Notice now, however, the computed scale near the middle of each side of each square, as seen in the interesting pattern of scale in the final bottom image, which corresponds to the view of the frustum as a series of squares decreasing in size stacked on top of each other. (The level-set view of TV regularization seems especially relevant in this case.) As shown in [29], the change in intensity of a radially symmetric with gradually increasing intensity is $\delta(r) = \alpha \,/\, r$ where r is the distance from the center of the object. Thus in this case, the scale will be computed as $scale(r) = \alpha \,/\, \delta(r)$, then the scale as measured by the Scale Recognition Probe will find that $scale(r) = \alpha \,/\, \alpha/r = r$. While this theory does not exactly match the given results, as seen in the bottom image, the basic relationship that scale is directly proportional to distance from the center of the square is evident (except, of course, around the corners).

The above examples demonstrate some of the difficulties that arise when trying to determine scale of various features in an image by using the Scale Recognition Probe. However, these difficulties are not necessarily a problem inherent in the Scale Recognition Probe itself; at the root of these problems is the fact that most images are comprised of features that are really the composition of multiple smaller- and larger-scaled features of various intensities. Fortunately, what is most important in how we will use the Scale Recognition Probe in the applications discussed later in this paper is that it does accurately measure the smallest scales present in the image, which can be observed in all of the test images considered above.

2.4 Perception of scale dependent on amount of regularization done

We next comment on how the different scales present in a regularized image depend in large part on how much regularization has been done to the image. When using TV regularization, there are two natural ways to recognize the various levels of scale in an image: first is, of course, by simple inspection, which can be nebulous; second, and more interestingly and usefully, is how TV regularization will perceive scale. It is well known that for increasing values of α there is increasing loss of smaller scale (finer detail) in the image. We consider how both perception and loss of scale are affected by the value of α.

Consider the function labeled as "9 extrema" in Figure 5(a). Depending on how much regularization has been done to this function, there are three levels of scale at which this function could be viewed: at the finest level is the actual function, at the next level is the "3 extrema" function, and at the coursest level is the "1 extremum" function. At successively courser levels, the value of the function in each region is simply the mean of the values over the subregions in the finer levels. Let $\alpha_{9\to 3}$ be the value of α in applying TV regularization by solving (2) at which the 9-extrema function transitions into the 3-extrema function, as seen in (b). It turns out that for $\alpha \geq \alpha_{9\to 3}$, TV regularization perceives the function as the 3-extrema function. This is illustrated by the fact that the function produced by solving (2) using $\alpha = 0.0100$ in (b), in which u_0 was the (true) 9-extrema function, is identical to the function produced by solving (2) using $\alpha = 0.0100$ in (c), when using the 3-extrema version of u_0 in (a) to solve (2). Similarly, the function produced by solving



(a) The original function, when perceived as having 9, 3 or 1 extrema.

(b) The evolution from a 9-extrema function to a 3-extrema function. The transition occurs at $\alpha_{9\to 3} \approx 0.0037$.

(c) The evolution from a 3-extrema function to a 1-extremum function. The transition occurs at $\alpha_{3\to 1} \approx 0.0259$.

(d) The evolution from a 1-extremum function to a constant function. The transition occurs at $\alpha_{1\to 0} \approx 0.1284$.

Figure 5: The smaller scales present throughout a function, as perceived by TV regularization, disappear as α increases. In (a) is the original R^1 function, when perceived as having 9, 3 or 1 extrema. For $0 \leq \alpha < 0.0037$, the function is perceived as the original 9-extrema function, as seen in (b). For $0.0037 \leq \alpha < 0.0259$, the function is perceived as the 3-extrema function, as seen in (b) and (c). For $0.0259 \leq \alpha < 0.1284$, the function is perceived as the 1-extrema function, as seen in (c) and (d). For $0.1284 \leq \alpha$, the function is perceived as the constant-valued function in (d). Note that the background from 0 to 1/3 and 2/3 to 1) also changes in intensity proportionally to its scale and that its change is relatively small.

(2) using $\alpha = 0.100$ in (c) in which u_0 was the 3-extrema function in (a) (or equivalently, if the u_0 used were the original 9-extrema function itself in (a)), is identical to the function produced using $\alpha = 0.100$ in (d), when using the 1-extremum version of u_0 to solve (2). This illustrates that for $\alpha \geq \alpha_{3\to 1}$, TV regularization perceives this function as the 1-extremum function seen in (c). For $\alpha \geq \alpha_{1\to 0}$, the resulting regularized image will simply be the constant image shown in (d).

For this simple function, it is easy to analytically predict what these transitional values of α are, as well as the exact results of TV regularization on this function for other values of α. We found analytically the values for $\alpha_{9\to 3}$, $\alpha_{3\to 1}$ and $\alpha_{1\to 0}$ given in Figure 5. For example, in Figure 5(b), the "9 extrema" function evolves into the "3 extrema" function when the decreasing maxima meet the increasing minima as α increases. For each of these 9 extrema, $scale = width/2 = 1/27\,/\,2 = 1/54$. Consequently, the value $\alpha_{9\to 3}$ at which the 9-extrema function transitions into the 3-extrema function is the value of α for which $\delta = \alpha\,/\,1/54 = 54\alpha$, where δ is the change in intensity needed for each extrema to complete this transition. For the first and third groups of three extrema, the maxima and minima are at 1.6 and 1.2, respectively, so each needs to change by 0.2 for the decreasing maxima to meet the increasing minima, and similarly for the middle three extrema. Thus we need $\delta = 0.2$ so that $\alpha_{9\to 3} = 0.2/54 \approx 0.0037$, as seen in (b). We found $\alpha_{3\to 1}$ and $\alpha_{1\to 0}$ similarly.

The above example illustrates the well known fact that, in general, any image in $\mathbf{R}^n$ will gradually evolve into an image with larger scales—that is, with less detail, as smaller-scaled features are lost—as α increases. Of course, in general these transitions do not occur at a few distinct values of α. Indeed, in general these transitions are more continuous: for most images, at various image locations and for various values of α this transition is almost continually occuring as α increases. Also, images are not comprised only of piecewise constant features, although in the discrete case an n x n image has n^2 pixels, each with a particular value, so in this sense the image could be thought of as being piecewise constant, albeit on a very fine scale. Consequently, the notion of scale is more complicated, as discussed earlier in this section. The analysis in this paper helps (but does not exhaustively) develop a precise understanding of how TV regularization perceives scale in an image and how TV regularization resolves an image into its various scales. Also, the relatively simple notion of scale we use in this paper is sufficient (and, indeed, necessary) to obtain the results we give later.

3 Selection of regularization parameter

We now consider applications of our understanding of TV regularization's recognization of scale in an image. The first application is the specific task of removing from an image all features whose scales are less than or equal to a specific threshold, while leaving all other larger-scaled features as intact as possible. That is, we would like to find the value of α in solving (2) that is just large enough to result in removing all features below a given scale threshold, $scale_{thresh}$, but no larger. We denote this particular α value as α_{thresh}. Where $scale(\vec{x})$ is the scale of $u = arg \min_u \frac{1}{2}\|u - u_0\|^2 + \alpha TV(u)$, we define

$$\alpha_{thresh} = \min\{\alpha : \min_{\vec{x}} scale(\vec{x}) > scale_{thresh}\}. \tag{8}$$

3.1 An example of what we want to accomplish

As a simple example of what we want to accomplish, we apply TV regularization to the image shown in Figure 6. This image contains checkboard "texture" of two distinct scales, 0.00125 and 0.00250 (where scale is as in (6)) and constant-valued circles and rectangles of four distinct scales, 0.005, 0.010, 0.020 and 0.040. The first image is the original image, while the following six are the images in which we have removed all features at or below six scale thresholds, corresponding to the six different scales present in the image. The values of α_{thresh} corresponding to each of the six scale thresholds are given in the caption of Figure 6.

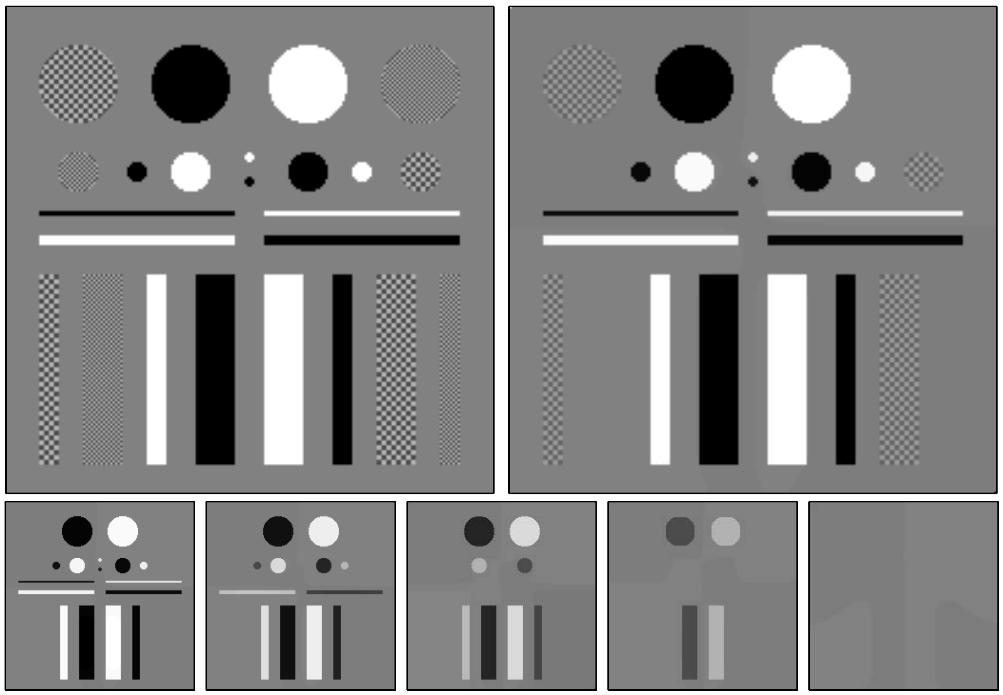


Figure 6: Results of solving (2) using values of α that are just large enough to remove all features at or below specific scale thresholds. The first image is the original image, comprised of textures and objects of six distinct scales: 0.00125, 0.00250, 0.00500, 0.01000, 0.02000 and 0.04000. The second image is the result of solving (2) using the minimum value of α that results in removal of the smallest scale, the smaller of the two checkboard textures: $\alpha_{0.00125} = \min\{\alpha : \min_{\vec{x}} scale(\vec{x}) > 0.00125\} = 0.00025$. The subsequent images are regularized images found by solving (2) using α_{thresh} values of 0.00050, 0.00239, 0.00495, 0.00990 and 0.01550, which correspond to the *scale thresholds* of the other texture and the circles and circles, 0.00250, 0.00500, 0.01000, 0.02000 and 0.04000. These α_{thresh} values were found automatically, using the α_{thresh} Algorithm given later in Section 3.3. The intensities of the objects are the actual intensities; no rescaling has been done to enhance contrast. In this example, there is essentially no change in the intensity of the background. The initial background intensity was 0.5 and the intensities of the features were 0 and 1, thus the mean of the entire image was approximately 0.5, which is essentially the result seen in the final image found using the relatively large value of $\alpha = 0.01550$.

The values of α_{thresh} and the corresponding images given in Figure 6 were *not* found experimentally, i.e. by choosing a sequence of α values and looking at the resulting images in order to see where the different

features of varying scales are completely removed. The algorithm to find the α_{thresh} values is automatic and is based on the ability of TV regularization to recognize scale. We give this algorithm later in Section 3.3.

3.2 Basic strategy for finding α_{thresh}

The strategy we use to find α_{thresh} is an iterative process based on the standard bisection method, where the desired "root" is α_{thresh} in (8). With this approach, there are two questions. First, how do we choose the initial lower and upper bounds, α_{min} and α_{max}, on our estimate for α_{thresh} to ensure that $\alpha_{min} \leq \alpha_{thresh} \leq \alpha_{max}$. The simplest and safest choice for α_{min} is 0, since by definition $\alpha \geq 0$. Since $\alpha_{thresh} \leq \alpha_{max}$, the choice of α_{max} will depend on $scale_{thresh}$ and on the image itself. We revisit how to choose α_{max} when we consider our first numerical example in Section 5.1.

As with the standard bisection method, given the current interval $[\alpha_{min}, \alpha_{max}]$ in which our "root" lies, we will move to either the lower half $[\alpha_{min}, (\alpha_{min} + \alpha_{max})/2]$ or the upper half $[(\alpha_{min} + \alpha_{max})/2, \alpha_{max}]$ of the interval, and thus we update either $\alpha_{min} = (\alpha_{min} + \alpha_{max})/2$ or $\alpha_{max} = (\alpha_{min} + \alpha_{max})/2$ with each iteration. The second question then is how to decide which subinterval to move to with each iteration. If α_{min} and α_{max} are the current lower and upper bounds on α_{thresh} prior to iteration i, then where $\alpha_i = (\alpha_{min} + \alpha_{max})/2$, we find $u_i = arg \min_u \frac{1}{2}\|u - u_0\|^2 + \alpha_i TV(u)$. We need to determine if there are any features or portions of features in u_i with scale at or below $scale_{thresh}$. To do this we perform a scale recognition probe, as described in Section 2.3, to find $scale(\vec{x})$ for u_i in order to determine whether $scale(\vec{x}) \leq scale_{thresh}$ anywhere $\vec{x}$ in the image u_i. If so, then our choice α_i is too small and we should move to the upper half of the interval $[\alpha_i, \alpha_{max}]$. If not, our choice was sufficiently large, and since α_{thresh} is the smallest of all such values of α, we know that $\alpha_{thresh} \leq \alpha_i$, in which case we move to the lower half of the interval $[\alpha_{min}, \alpha_i]$.

Conceptually, we want to compare $scale(\vec{x})$ to $scale_{thresh}$. Unfortunately, if $\delta(\vec{x}) = 0$ anywhere in the image, we end up dividing by 0. We avoid this by instead simply comparing $\delta(\vec{x})$ to δ_{thresh} where $\delta_{thresh} = \alpha_{probe}/scale_{thresh}$. Since $scale(\vec{x}) \leq scale_{thresh} \iff \delta(\vec{x}) \geq \delta_{thresh}$, if $\delta(\vec{x}) \geq \delta_{thresh}$ anywhere $\vec{x}$ in the image, then there are still features at or below $scale_{thresh}$, in which case we need to increase the value of α by moving to $[\alpha_i, \alpha_{max}]$; otherwise we move to $[\alpha_{min}, \alpha_i]$.

In the end, as with the standard bisection method, our approach will give us an interval $[\alpha_{min}, \alpha_{max}]$ which contains the "root," the true value of α_{thresh}. Convergence is guaranteed, since the estimate interval is halved with each iteration. As we are trying to finding the smallest value of α that will remove all features of scale at or below $scale_{thresh}$, if we choose α_{thresh} too small we don't quite remove all unwanted features. In otherwords, underestimating α_{thresh} is not acceptable. Therefore, given that we know only that $\alpha_{thresh} \in [\alpha_{min}, \alpha_{max}]$, we necessarily choose for our estimate $\alpha_{thresh} = \alpha_{max}$.

It turns out that we actually need to take $\alpha_{thresh} = \alpha_{max} + \alpha_{probe}$. Related to this is the fact that we need to choose α_{probe} small relative to α_{max}. To avoid an excessively lengthy explanation of these two facts, we give Figure 7 to demonstrate that if a particular feature is nearly gone (i.e. contrast with its neighbors is nearly 0), then that feature's scale may not be accurately recognized, as seen in the first pair of plots. As demonstrated by the second pair of plots, for a given image, we could choose a smaller value of α_{probe}

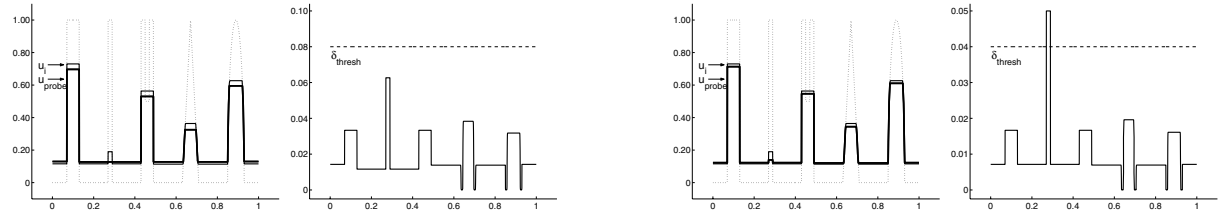


Figure 7: For each pair of plots, the first plot shows u_i, the result of applying TV regularization (2) to the original function (the dotted function) using α_i, and u_{probe}, the result of applying TV regularization (2) to u_i using α_{probe}. For each pair, the second plot shows the change in intensity $\delta = |u_i - u_{scale}|$, due to the scale recognition probe, which is then used to determine scale of remaining features. δ_{thresh} is the change in intensity threshold corresponding to a scale threshold just larger than the feature at 0.3. In the first pair of plots, the true scale of the feature at 0.3 would not be recognized when using the current value of α_{probe}, as there is not sufficient contrast between that feature and neighboring features. In the second pair of plots we used a value of α_{probe} half that used in the first pair of images, and the the true scale of the feauture at 0.3 would now recognized.

that would lead to accurate measurement of scales. Ideally, then, we would like α_{probe} to be as small as possible. However, the smaller it is, the more precise our solution in solving (2) must be, which requires more iterations and thus potentially much higher computational expense. There can be a wide range of values of α that arise in trying to find α_{thresh}. This range of α values will depend on the image domain (as mentioned in Section 1.3 and discussed in the Appendix), the range of intensities in the image, and the desired scale threhold. Consequently, it is not obvious a priori how small is sufficiently small for α_{probe}. Therefore, we use α_{max}, our current upper bound on the estimate for α_{thresh}, as a point of reference. That is, we want α_{probe} to be small relative to α_{max}. (Choosing α_{probe} small relative to α_{min} could prove disastrous, as α_{min} could be very small itself, even 0, as it generally is initially.) Still, since we must choose some finitely small value of α_{probe} when performing the scale recognition probe, for any given value of α_{probe} we could easily find a simple function that would result in failure to recognize that all unwanted features (scale less than the chosen scale threshold) are actually not removed, as seen in the first pair of plots. For that pair of plots, using the given value of α_{probe}, it would appear that we have found a value of α_{thresh} that has resulted in removal of all features of scale below the given threshold, when in fact α_{thresh} would actually need to be slightly larger. In the end we must choose as our estimate $\alpha_{thresh} = \alpha_{max} + \alpha_{probe}$. Of course, since we are trying to find the smallest value of α that results in the removal of all unwanted features, we want to choose α_{probe} to be relatively small, which we already established anyway. Lemma 2 in Section 4.3 further explains why we need $\alpha_{probe} \ll \alpha_{max}$. In our subsequent numerical examples, we have arbitarily chosen $\alpha_{probe} = \alpha_{max}/100$.

3.3 The α_{thresh} Algorithm

We now give the complete algorithm for estimating α_{thresh}, as defined in (8).

α_{thresh} Algorithm

1. Set $\alpha_{min} = 0$; choose α_{max}, $scale_{thresh}$.
2. Initialize $i = 1$, $\alpha_1 = (\alpha_{min} + \alpha_{max})/2$, $\alpha_{probe} = \alpha_{max}/100$, $\delta_{thresh} = \alpha_{probe}/scale_{thresh}$.
3. Repeat steps a - d until $error \leq$ error tolerance:
   - 3a. $u_i = arg \min_u \frac{1}{2}\|u - u_0\|^2 + \alpha_i TV(u)$;
     $u_{probe} = arg \min_u \frac{1}{2}\|u - u_i\|^2 + \alpha_{probe} TV(u)$.
   - 3b. $\delta_{max} = \max_{\vec{x}} |u_{probe}(\vec{x}) - u_i(\vec{x})|$ $(= \|u_{probe}(\vec{x}) - u_i(\vec{x})\|_{L^\infty})$.
   - 3c. If $\delta_{max} \geq \delta_{thresh}$ then $\alpha_{min} = \alpha_i$
     else $\alpha_{max} = \alpha_i$.
   - 3d. Update: $i = i + 1$, $\alpha_i = (\alpha_{min} + \alpha_{max})/2$, $\alpha_{probe} = \alpha_{max}/100$, $\delta_{thresh} = \alpha_{probe}/scale_{thresh}$.
4. $\alpha_{thresh} = \alpha_{max} + \alpha_{probe}$ $(= 1.01\alpha_{max})$.

The stopping point is described at the top of Step 3, where *error* can be either the maximum tolerable *absolute* or *relative* error. The absolute error after i iterations would be $\leq (\alpha_{max} - \alpha_{min}) / 2^i$, given the initial values of α_{min} and α_{max}, since we are halving this interval with each iteration, and similarly for the maximum possible relative error. We allow α_{probe} to vary with α_{max}, to ensure that $\alpha_{probe} \ll \alpha_{max}$; in the above algorithm we update α_{probe} at each step to be $\alpha_{probe} = \alpha_{max}/100$. Also, to account for numerical imprecisions, to be more conversative one might choose δ_{thresh} to be slightly smaller than the theoretical δ_{thresh}. In subsequent numerical results we compare δ_{max} with $0.95\,\delta_{thresh}$.

Prior to giving numerical results for the α_{thresh} Algorithm, in the following section we carry out a mathematical study of the α_{thresh} Algorithm.

4 Mathematical analysis of the α_{thresh} Algorithm

In this section we analyze the α_{thresh} Algorithm from the perspective of the *G norm* introduced by Meyer in [22]. Essentially, we show how our notion of scale helps give an intuitive interpretation of the G norm and conversely how this norm gives some enlightening insight into the α_{thresh} Algorithm.

4.1 Meyer's G norm

Recently, Meyer did an interesting mathematical analysis of the ROF model in [22]. He introduced a new space, the G space, to model oscillating patterns:

Definition 1 *G is the Banach space composed of the distributions f which can be written*

$$f = \partial_1 g_1 + \partial_2 g_2 = \operatorname{div}(g) \tag{9}$$

with g_1 and g_2 in L^∞. On G, the following norm is defined:

$$\|f\|_G = \inf \left\{ \|g\|_{L^\infty} \, : \, f = \operatorname{div}(g),\ g = (g_1, g_2),\ g_1 \in L^\infty,\ g_2 \in L^\infty,\ \|g(\vec{x})\| = \sqrt{|g_1|^2 + |g_2|^2}(\vec{x}) \right\} \tag{10}$$

See [6] for an analysis of G in a discrete setting and [5] for a generalization of Meyer's definition to bounded domains in a continuous setting. We will use the following ball in G ($\alpha > 0$):

$$G_\alpha = \{ f \in G : \|f\|_G \leq \alpha \}. \tag{11}$$

We consider the discrete setting. It is shown in [6] that G is then the set of functions with zero mean. Therefore what is important is not the space G itself, but the G norm: indeed, oscillating patterns such as textures have a small G norm and are thus well captured when minimizing the G norm. The following Lemma will prove to be useful. We note that $1/n$ is the discretization step if our image is n x n on the unit square.

Lemma 1 *If f belongs to G (i.e. if f is of zero mean), then*

$$\frac{1}{4n}\|f\|_{L^\infty} \leq \|f\|_G \leq 4n\|f\|_{L^\infty} \tag{12}$$

Proof In [6], it is shown that there exists g such that $f = \operatorname{div}(g)$ and $\|f\|_G = \|g\|_{L^\infty}$. It is easy to check that $\|\operatorname{div}\|_{L^\infty} \leq 4n$, which gives the right-hand side inequality in (12). Since the identity $I = \operatorname{div}^{-1} \operatorname{div}$, we have $1 \leq \|\operatorname{div}^{-1}\|_{L^\infty} \|\operatorname{div}\|_{L^\infty}$, from which we get the left-hand side of (12). ■

It is a standard result that in a finite dimensional normed space, all of the norms are equivalent. Lemma 1 gives the equivalence constants explicitly. It is clear that as $n \to \infty$ the G norm and the L^∞ norm are no longer equivalent norms.

4.2 Relating the G norm and ROF model

The following proposition is shown in [12] (the proof is based on convex analysis):

Proposition 1 *The solution to (2) is given by*

$$u = u_0 - P_{G_\alpha}(u_0) \tag{13}$$

where $P_{G_\alpha}(u_0)$ denotes the orthogonal projection (with respect to the L^2 scalar product) of u_0 on G_α, defined by (11).

One of the main results of [22] happens to be a straightforward corollary of Proposition 1. Where $\bar{u_0}$ is the mean of u_0 over the image domain, we first define

$$\alpha^G = \|u_0 - \bar{u_0}\|_G, \tag{14}$$

and then give the corollary.

Corollary 1 *Where u is the solution of (2), we have:*

- *If $\alpha \leq \alpha^G$, then $\|u - u_0\|_G = \alpha$.*
- *If $\alpha \geq \alpha^G$, then $u = \bar{u_0}$.*

It is well known that for any image there is a finite value of α above which the solution to (2) is simply the mean of the original image u_0. Thus we see from Corollary 1 that $\alpha^G = \|u_0 - \bar{u_0}\|_G$ is precisely this value of α. As we can see, the behavior of the ROF model is closely related to the G norm of the initial data u_0. Before now, there has been no easy or intuitive interpretation of the G norm.

Let us again consider (6) and (7), which link scale to α. When rewritten as

$$\alpha = \delta \, scale \tag{15}$$

we see that the G norm is proportional to some "average" scale present in the image. The G norm takes into account all the features contained in the image: it therefore can be seen as a generalization of the notion of scale (in the case when several scales are available in the image). Notice however that formula (15) is an approximation: one of its main advantage is that it holds for each pixel, as opposed to the exact formula (16) presented in the next paragraph. We give a rough explanation of Corollary 1:

- If u_0 has features with scale larger than $\delta \, scale$, then $u - u_0$ contains all the features with scale smaller than $\delta \, scale$.
- If all the features in u_0 are of scale smaller than $\delta \, scale$, then $u = \bar{u_0}$.

This is another confirmation that the analysis of the ROF model in [29] based on scale.

Another way to see the link between the G norm and the notion of scale is a result by Strang [27]. If $f \in G$, then

$$\|f\|_G = \sup_{E \subset \Omega} \frac{\int_E f}{P(E, \Omega)} \tag{16}$$

where Ω is the domain of the image, and $P(E, \Omega)$ stands for the perimeter of E in Ω (see [3] for the precise definition of the perimeter of a set). The G norm is therefore equal to an area divided by a perimeter, i.e. a scale, and thus it is a further justification of our definition of scale (6), since it appears that it is a simplified version of Strang's result. As noted earlier, the advantage of our definition of scale (6) is that we can easily use it numerically. As we have explained at the beginning of the paper, the definition of scale (6) is valid for piecewise constant images, and is just an approximation for general images. The exact definition of scale would be with Meyer'e G norm and Strang's formula (16). Putting this result into practice should be the subject of future studies. In this paper, we do not claim to use the most theoretically general definition of scale; the definition we use (6) is a good and quite implementable approximation of more general and abstract definitions of scale.

In this paper, we have presented the α_{thresh} Algorithm to compute the parameter α in (2) to remove all the features with scale equal to or smaller than a given threshold. Thanks to (15), we see that this essentially amounts to constraining the residual $u - u_0$ to be such that $\|u - u_0\|_G = \delta \, scale$. However, as far as we know, the only algorithm that has been proposed to compute the G norm of an image is the one introduced in [7]. Interesting, this algorithm is also based on the bisection method: one compares u with $P_{G_\alpha}(u)$, that is, one determines whether all the features in u are smaller than $\delta \, scale$.

4.3 Mathematical study of the α_{thresh} Algorithm

In this final subsection of our mathematical analysis, we give an interesting theoretical insight into the α_{thresh} Algorithm. We take the same notations as in the description of the α_{thresh} Algorithm.

Proposition 2 *If $scale_{thresh} < 1/4n$, then the α_{thresh} Algorithm will return $\alpha_{thresh} = 0$.*

The aim of this proposition is simply to confirm that the α_{thresh} Algorithm does what we would expect it to do in one extreme case. Indeed, $1/4n$ is the smallest available scale in the image: by (6), it is the scale of a single pixel when the image is n x n and the domain is the unit square. If we choose $scale_{thresh} < 1/4n$, than we expect to keep all features of the original image, that is, we expect there to be no regularization done, which is the case if $\alpha = 0$.

Proof of Proposition 2 ¿From Step 3a of the α_{thresh} Algorithm and Proposition 1, we have

$$u_{probe} = u_i - P_{G_{\alpha_{probe}}}(u_i). \tag{17}$$

We recall that $\delta_{max} = \|u_{probe} - u_i\|_{L^\infty}$. ¿From (17) and Lemma 1, we deduce that

$$\frac{1}{4n}\delta_{max} \leq \|P_{G_{\alpha_{probe}}}(u_i)\|_G \leq 4n\,\delta_{max}. \tag{18}$$

But by definition we know that $\|P_{G_{\alpha_{probe}}}(u_i)\|_G \leq \alpha_{probe}$. We thus get

$$\delta_{max} \leq 4n\|P_{G_{\alpha_{probe}}}(u_i)\|_G \leq 4n\,\alpha_{probe}. \tag{19}$$

Using the fact that $\alpha_{probe} = \delta_{thresh}\, scale_{thresh}$, we deduce that

$$\delta_{max} \leq scale_{thresh}\,\delta_{thresh}\,4n. \tag{20}$$

Since we assume that $scale_{thresh} < 1/4n$, we then get $\delta_{max} < \delta_{thresh}$. With each iteration of the α_{thresh} Algorithm, in Step 3c we will always choose $\alpha_{max} = \alpha_i = (\alpha_{min} + \alpha_{max})/2 = \alpha_{max}/2$, since initially we take $\alpha_{min} = 0$ in the α_{thresh} Algorithm So given an initial α_{max}, where α^k_{max} is the value of α_{max} at iteration k, we will have $\alpha^k_{max} = \alpha_{max}/2^k$, which of course $\to 0$ as $k \to \infty$. Finally, since $\alpha_{probe} = 0.01\alpha_{max}$ in our implementation of the α_{thresh} Algorithm, so that $\alpha_{thresh} = 1.01\alpha_{max}$, then $\alpha_{thresh} \to 0$ as $k \to \infty$. ■

The following result helps to further explain why α_{probe} needs to be small. Let us first denote $\delta_\alpha = |u_\alpha - u_0|$, where $u_\alpha = arg\min_u \ \frac{1}{2}\|u - u_0\|^2 + \alpha\, TV(u)$, and for $i \geq 1$, $\alpha_i^G = \|u_i - \bar{u_0}\|_G$, with notations of the α_{thresh} Algorithm.

Lemma 2 *If $\alpha_{probe} \geq \alpha_i^G$, then we have $\delta_{\alpha_{probe}} = \delta_{\alpha_i^G}$.*

Proof ¿From Step 2a of the α_{thresh} Algorithm and Proposition 1, we have

$$u_i = u_0 - P_{G_{\alpha_i}}(u_0). \tag{21}$$

¿From Corollary 1, we know that exactly one of the following statements (i) or (ii) holds:

(i) If $\alpha_{probe} \leq \alpha_i^G$, then $\|u_{probe} - u_i\|_G = \alpha_{probe}$.

(ii) If $\alpha_{probe} \geq \alpha_i^G$, then $u_{probe} = \bar{u_0}$.

If $\alpha_{probe} \geq \alpha_i^G$, then we will have $\delta_{\alpha_{probe}} = \delta_{\alpha_i^G}$. ■

As a direct consequence of Lemma 2, we see that if $\alpha_{probe} \geq \alpha_i^G$, then (7) cannot be used to compute the scale anymore. Therefore, if we want to check if features with a given scale are still present in u_i, then we need to have $\alpha_{probe} \leq \alpha_i^G$. Ideally we have $\alpha_{probe} \ll \alpha_i^G$.

The result of Lemma 2 is illustrated in Figure 7, which illustrates why α_{probe} needs to be small. On the other hand, the smaller α_{probe} is, the more accurate we need to be when computing u_{probe}, which is more expensive. As previously mentioned, we arbitrarily choose $\alpha_{probe} = \alpha_{max}/100$ in our implementation of the α_{thresh} Algorithm.

This ends our mathematical analysis of the α_{thresh} Algorithm. In the following two sections we turn our attention to numerical results of the α_{thresh} Algorithm and to other ways in which to exploit our understanding of how TV regularization recognizes scale in an image.

Figure 8: Results of applying the α_{thresh} Algorithm to the Mandrill image, where $scale_{thresh} = 1/1024$ is the scale of a single pixel. First is a plot of the values of α_{min}, α_i and α_{max} produced by the α_{thresh} Algorithm. Next is the original image, followed by the images corresponding to the first three values of α_i found by the α_{thresh} Algorithm. The final estimate is $\alpha_{1024} \approx 0.00052$.

5 Numerical results of the α_{thresh} Algorithm

We now give some examples of applying the α_{thresh} Algorithm to both noise-free and noisy images.

5.1 A detailed look at the α_{thresh} Algorithm

We first apply the α_{thresh} Algorithm to the Mandrill image shown in Figure 8. In this example we wish to find the value of α_{thresh} that will result in the removal of all features of scale less than or equal to the scale of a single pixel. Of course, larger features will also be affected by the regularization, and some may even be removed, depending on their initial intensity levels and contrast with surrounding features. This image is 256 x 256 on the unit square, thus $scale_{thresh} = (1/n)^2 \,/\, 4/n = 1 \,/\, 4n = 1/1024$. So we are trying to find $\alpha_{1/1024}$. The intensity of the image is normalized to be between 0 and 1, thus we choose α_{max} to be large enough to completely change the intensity of a single pixel by $\delta_{max} = 1$. Using (5), $\alpha_{max} = \delta_{max}\, scale_{thresh} = 1 \cdot 1/1024 \approx 0.000977$.

As this is the first time we have seen this algorithm in action, it is enlightening to see what each iteration of the algorithm produces: both the value of α_i, the midpoint of the estimate interval $[\alpha_{min}, \alpha_{max}]$ for α_{thresh} for each iteration, and the corresponding regularized image. The first plot in Figure 8 is α_{min}, $\alpha_i = (\alpha_{min} + \alpha_{max})/2$ and α_{max} values for each iteration. Next is the orginal (noise-free) image and the images corresponding to the first few α_i values found by the algorithm. Subsequent images appear virtually identical and are omitted. As seen in the plot of α values and as observed in the images themselves, most of the change occurs within the first few iterations, particularly if good initial values of α_{min} and α_{max} are chosen.

We can find as precise an estimate for α_{thresh} as wanted. In general, where α_i is our estimate at iteration i for α_{thresh}, and where α_{min} and α_{max} are the initial lower and upper bounds for the α_{thresh} Algorithm, then after each iteration we have a bound on the absolute error of $|\alpha_i - \alpha_{thresh}| \leq (\frac{1}{2})^i(\alpha_{max} - \alpha_{min})$, and similarly for the relative error. This additional precision comes at a numerical price and is normally unnecessary, given the resulting insignificant amount of change in the image. The final estimate where $scale_{thresh} = 1/1024$ is $\alpha_{1/1024} = 0.00052$.

5.2 Results of the α_{thresh} Algorithm for noise-free images

We next give results, both the values found for α_{thresh} and the corresponding images, of applying the α_{thresh} Algorithm to three standard (noise-free) images using scale thresholds of 2^{k-1} x 2^{k-1} pixels for $k = 1$ to 7 (e.g. for $k = 1$, the scale threshold is a single pixel). For an n x n image on the unit square, these correspond to scales of $2^{k-1} \,/\, 4n$ for $k = 1$ to 7. Of course, these scales given for square features correspond to a variety of non-square features. For example, the scale of an 8 x 8 pixels square is also the scale of both a circle of diameter 8 pixels and a rectangle of 4 pixels width and infinite length. Results are given in Figure 9 and Table 1. The Mandrill and Toys images are both 256 x 256, while the Canaletto image is 512 x 512. Consequently, the scale thresholds for the Mandrill and Toys images range from 1/1024 to 1/16 while the scales thresholds for the Canaletto image range from 1/2048 to 1/32, as seen in Table 1.

In all three cases, it seems that a significant amount of regularization was necessary even for this smallest possible scale threshold of a single pixel. This is seen in comparing the first and second images, the original and the result of using $scale_{thresh}$ of one pixel, in each of the three sets of images. Later, in Section 6.1, in which we briefly consider the multiscale effects of TV regularization, we look at the results of TV regularization applied to the Mandrill image when using $\alpha = 0.1\,\alpha_{1/1024}, 0.2\,\alpha_{1/1024}, ..., 1.0\,\alpha_{1/1024}$.

The results seen in Figure 9 are not quite as obvious and perhaps not as dramatic as those seen in Figure 6. This is expected, as for these images we have not attempted to choose scale thresholds corresponding to specific scales present in the images, as we had done in obtaining the results of Figure 6. Still, for each of the three images in Figure 9, there are a number of specific features which are obviously present in a few of the images in the sequence, but then disappear once a certain scale threshold is reached. The conclusion is that each feature (or portion of a feature) was larger than the scale threshold used to obtain the images in which it was still present, but smaller than the scale threshold used in obtaining the image in which it first was absent.

It is not completely obvious from inspection that each of the images displayed has been regularized just enough to remove all features at or below the given threshold. What is more apparent is the similar levels of scale present in the three different images after applying the α_{thresh} Algorithm using the same scale threshold, as we now describe. Recall that the Mandrill and Toys images are 256 x 256, while the Canaletto image is 512 x 512. Then the first (the original) Mandrill and Toys images both have features with scale up to and including 1/1024, the scale of a single pixel in a 256 x 256 image. The first (original) Canaletto image has scale up to and including 1/2048, the scale of a single pixel in a 512 x 512 image. It is the *second* Canaletto image, the result of the α_{thresh} Algorithm using a $scale_{thresh} = 1/1024$, that should be compared to the first Mandrill and Toys images. Similarly, the second Mandrill and Toys images and the third Canaletto image all contain features of scale up to and including 1/512, as a resulting of applying the α_{thresh} Algorithm using $scale_{thresh} = 1/512$, while the third Mandrill and Toys images and the fourth Canletto image all contain features of scale $\leq 1/256$, and so on. Thus we compare Mandrill and Toys images 1, 2, ..., 7 along with Canaletto images 2, 3, ..., 8, respectively. By comparing the appropriate images, we do indeed see similar levels of scale, i.e. degrees of detail, in all three of the images for any given scale threshold.

5.3 Results of the α_{thresh} Algorithm for noisy images

We next apply the α_{thresh} Algorithm to three noisy images, using four different noise levels, as shown in Figure 10. We consider the 256 x 256 Peppers image, the 256 x 256 Elaine image, and the 140 x 140 Blood Vessels image. Before adding noise, as usual the images are normalized to minimum and maximum intensities of 0 and 1, and the domain is the unit square. The same Gaussian noise is added to all three images. We do this for four different levels of noise, which found are by scaling the noise to have maximum magnitude (both positive and negative) of 0.25, 0.50, 0.75 and 1.00. Because each image has a different signal level, although exactly the same noise is added to each image, the resulting noisy images have different signal-to-noise ratios, as is seen in the second table of Table 2. The α_{thresh} Algorithm is applied to each of the twelve noisy images where in all cases the scale threshold is $1\,/\,4n$ (where the image is n x n on the unit square), which corresponds to a single pixel. The resulting images are given in Figure 10. The α_{thresh} values found and the old and new SNRs are given in Table 2.

The numerical results given for noisy images (before and after regularization) are not meant to demonstrate the basic effects of TV regularization on a noisy image, which are of course well known by now. What is novel about these results is that they were obtained without any knowledge of noise level being explicity incorporated into the process for finding the optimal value of α and the corresponding regularized image. The only information used by the α_{thresh} Algorithm was the scale threshold to use: we chose a scale of one pixel in all twelve cases (three images, four noise levels for each). Of course, the amount of noise in the image inherently influences the value of α_{thresh} found by the α_{thresh} Algorithm. This relates to our earlier discussion centered around Figure 2. As expected, applying the α_{thresh} Algorithm to noisier images results in larger α_{thresh} values, as seen in Table 2.

Obviously it is quite useful to have an approach to denoising that does not depend an accurate measure of noise present in the image, particularly since noise level often is unknown or is, at best, an estimate. As the α_{thresh} Algorithm is not necessarily a denoising algorithm, and we leave to a separate paper further investigation of it as a denoising technique.

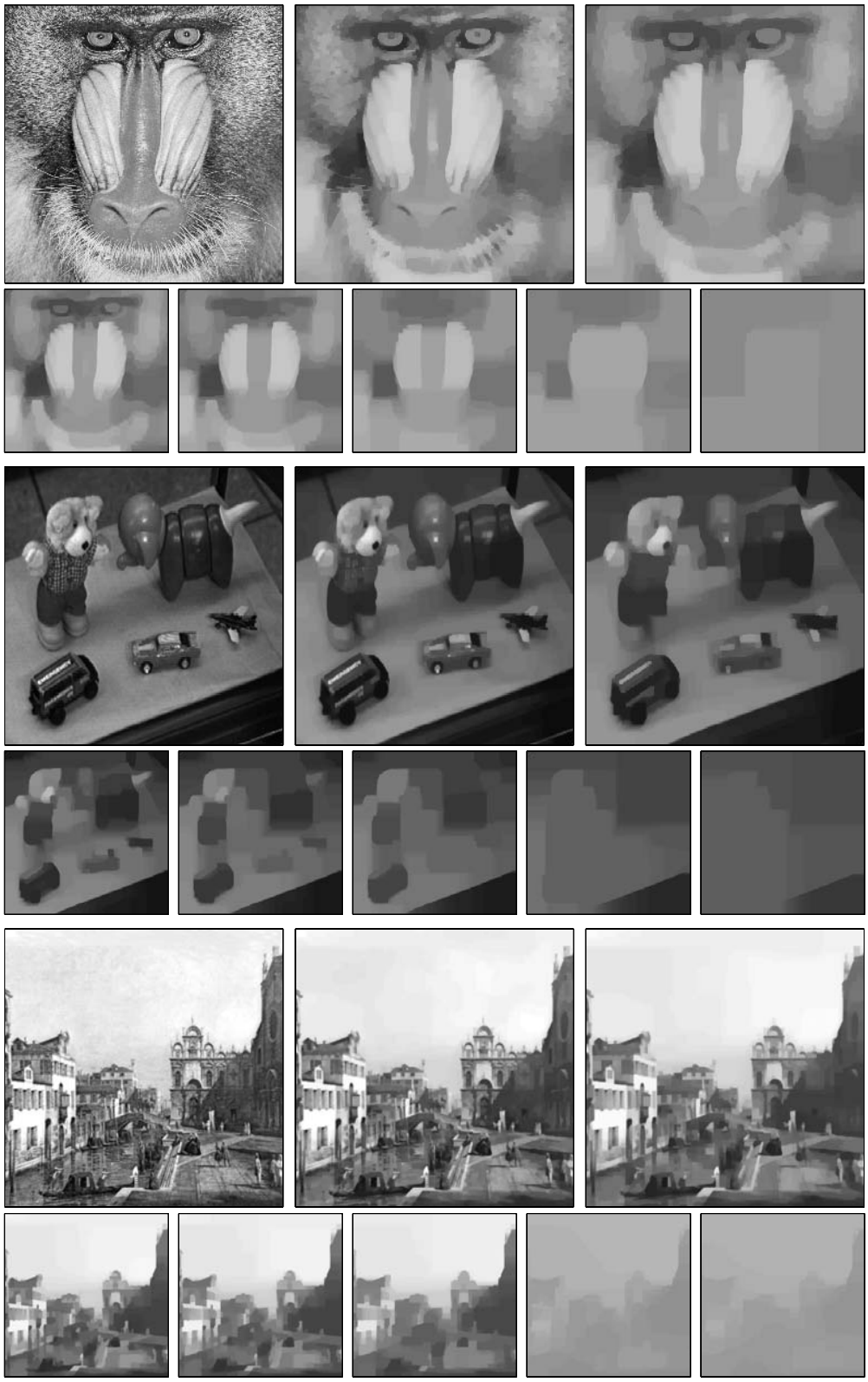


Figure 9: Results of the α_{thresh} Algorithm for scale thresholds of 2^{k-1} x 2^{k-1} pixels for $k = 1$ to 7. The actual scale threshold depends on the size (resolution) of the image. Values of α_{thresh} found for each scale threshold for each image are given in Table 1 and are plotted in Figure 11. In each set is the original image followed by the seven regularized images corresponding to the seven α_{thresh} values found by the α_{thresh} Algorithm for the seven different scale thresholds.

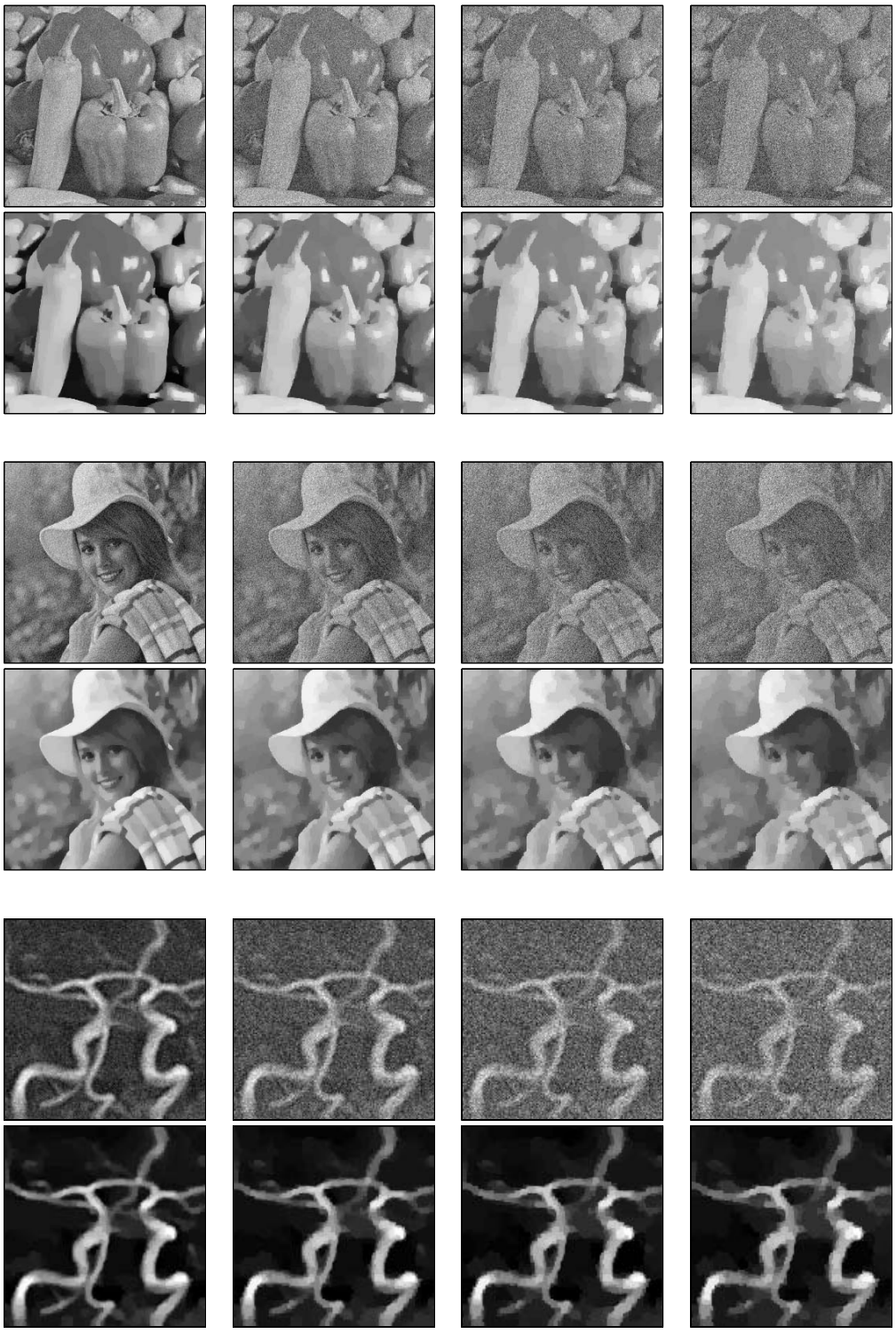


Figure 10: The α_{thresh} Algorithm applied to three noisy images, with four levels (magnitudes) of noise. The α_{thresh} values found using the α_{thresh} Algorithm, where the scale threshold was a single pixel, are given in the first table in Table 2. Noise levels, before and after regularization, are given in the second table in Table 2. For each pair of images, the top image is the noisy image and the bottom image is the regularized image from solving (2) using the α_{thresh} value found using the α_{thresh} Algorithm.

$scale_{thresh}$	Mandrill	Toys	Canaletto
1/2048	-	-	0.00012
1/1024	0.00052	0.00018	0.00031
1/512	0.00100	0.00066	0.00068
1/256	0.00132	0.00151	0.00099
1/128	0.00224	0.00308	0.00175
1/64	0.00418	0.00457	0.00256
1/32	0.00750	0.01007	0.00346
1/16	0.01793	0.01670	-

Table 1: The α_{thresh} values found for the scale thresholds used when applying the α_{thresh} Algorithm to the (noise-free) Mandrill, Toys and Canaletto Images in Figure 9. These values are plotted in Figure 11(a).

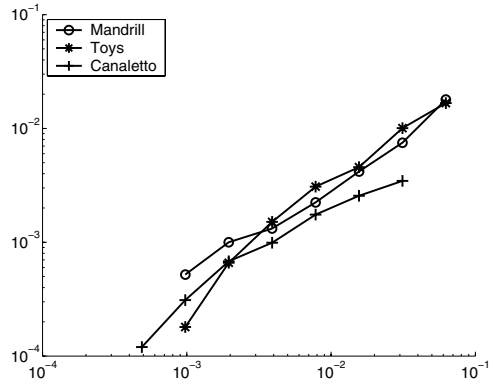
α_{thresh} values

Noise level	Peppers	Elaine	Blood Vessels
0.25	0.00044	0.00031	0.00045
0.50	0.00052	0.00051	0.00092
0.75	0.00080	0.00074	0.00134
1.00	0.00106	0.00098	0.00176

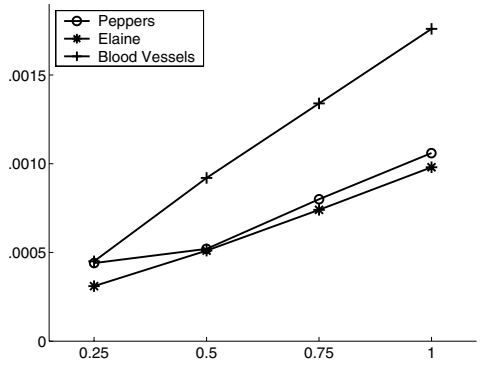
Signal-to-noise ratios, where SNR $= \sigma^2_{signal}/\sigma^2_{noise}$

Noise	Peppers		Elaine		Blood Vessels	
level	Old	New	Old	New	Old	New
0.25	11.11	22.23	11.56	24.54	10.18	31.29
0.50	2.78	17.18	2.89	15.78	2.55	14.36
0.75	1.23	11.29	1.28	11.60	1.13	9.81
1.00	0.69	8.55	0.72	9.20	0.64	7.53

Table 2: Data for the images in Figure 10. The first table gives the α_{thresh} values found for the four noise levels that were added to each image. The second table gives the corresponding signal-to-noise ratios for each image and noise level, both before and after regularization using the α_{thresh} value given in the first table. A scale threshold of one pixel was used in all cases. The values in the first table are plotted in Figure 11(b).



(a) α_{thresh} as function of $scale_{thresh}$.



(b) α_{thresh} as function of noise level.

Figure 11: Plots of α_{thresh} values as a function of $scale_{thresh}$ in (a) and noise level in (b). The data in the first plot are the α_{thresh} values from Table 1, which were used to obtain the results for the noise-free images in Figure 9. The data in the second plot are the α_{thresh} values from the first table of Table 2, which were used to obtain the results for the noisy images in Figure 10.

5.4 α_{thresh} as a function of $scale_{thresh}$ and SNR

We conclude this section by examining how α_{thresh} increases with $scale_{thresh}$ for the three noise-free images considered and how α_{thresh} increases with *noise* for the three noisy images just considered. These α_{thresh} values were given in Tables 1 and Table 2. The plots of these data are given in Figure 11.

The first plot in Figure 11 shows the values of α_{thresh} as a function of $scale_{thresh}$ for the Mandrill, Toy and Canaletto images. Although each of the noise-free images is quite different from the other two, the values of α_{thresh} found for each $scale_{thresh}$ are quite similar. Also, the resolution of the image does not seem to greatly affect the relationship between α_{thresh} and the chosen scale threshold, as illustrated by the similar results of both of the 256 x 256 Mandrill and Toy images as compared to the 512 x 512 Canaletto image.

The other plot in Figure 11 shows the values of α_{thresh} found as a function of noise level. For all three images and for all four noise levels, we found the α_{thresh} corresponding to a scale threshold of $1/4n$, i.e. a single pixel. For all three images, α_{thresh} appears to increase as noise level increases at approximately the same rate. Quite interestingly, the relationship between α_{thresh} and noise level is nearly exactly linear for the given range of noise levels.

The α_{thresh} values for the Blood Vessels image are larger than those for the Peppers and Elaine images because it is 140 x 140 as opposed to the Peppers and Elaine images being 256 x 256. Since the domain for all three images is the unit square, the scale a single pixel in the Peppers image is 140/256 the scale of a single pixel in the Elaine and Blood Vessel images. Not surprisingly, the ratios of the Peppers and Elaine α_{thresh} values to the Blood Vessels α_{thresh} values are close to 140/256.

6 Other applications of scale recognition

In this final section, prior to our summary and conclusions, we briefly consider other ways in which to exploit our understanding of TV regularization's natural ability to perceive scale in an image. As already seen above, we can measure the scale throughout the image in order to find the minimum value of α required to remove all features at or below any given scale threshold. We briefly consider two other potential uses for this ability to measure scale. First, we can determine at exactly which locations there is a feature or a portion of a feature of or below any given scale. This leads to some insight on the multiscale effects of TV regularization, which we briefly examine in Section 6.1. Second, in Section 6.2 we use the ability to determine scale at each discrete location throughout the image to examine the rate at which scale is lost as α increases.

6.1 Multiscale and scalespace effects of TV regularization

The multiscale and scalespace-generating effects of TV regularization are well known and are the subject of ongoing investigation. See, for example, [20], [25] and [30]. Of course, a more accurate and complete understanding of the multiscale and scalespace-generating nature of TV regularization is really only possible if there exists a more precise and complete notion of scale as perceived by TV regularization. Therefore, we expect that the theory and discussion presented in the previous sections will lead to a better understanding of the multiscale and scalespace-generating effects of TV regularization. As mentioned earlier, as this is a fairly complex issue, we do not attempt to treat it in detail in this paper. We do give two examples that lend some insight into the inherent ability of TV regularization to recognize scale, insight that we expect to lead to further discussion and development of theory.

We consider in more detail the Mandrill image shown earlier in Figures 8 and 9. In Section 5.1 we found that for the Mandrill image, $\alpha_{1/1024} = 0.00052$ is the minimum value of α necessary to remove all features at or below a scale threshold of 1/1024, the scale of a single pixel. We now examine the results when solving (2) using a range of values between 0 and $\alpha_{1/1024}$, in order to see in more detail the effects of the regularization. The resulting images are given in Figure 12. There are eleven sets of images, the first corresponding to the original image, and the other ten corresponding to the results of solving (2) using the ten values of $\alpha = 0.1\,\alpha_{1/1024}, 0.2\,\alpha_{1/1024}, ..., 1.0\,\alpha_{1/1024}$.

For each set (organized by columns), the top image is the image itself. The second image (second row of the set) shows the locations throughout the image at which there are features at or below the scale threshold of $1/4n$ (where $n = 256$), the scale of a single pixel. Similarly, the third and fourth rows of images show the locations in the image at which there are features at or below the scale thresholds of $1/3n$ and $1/2n$,

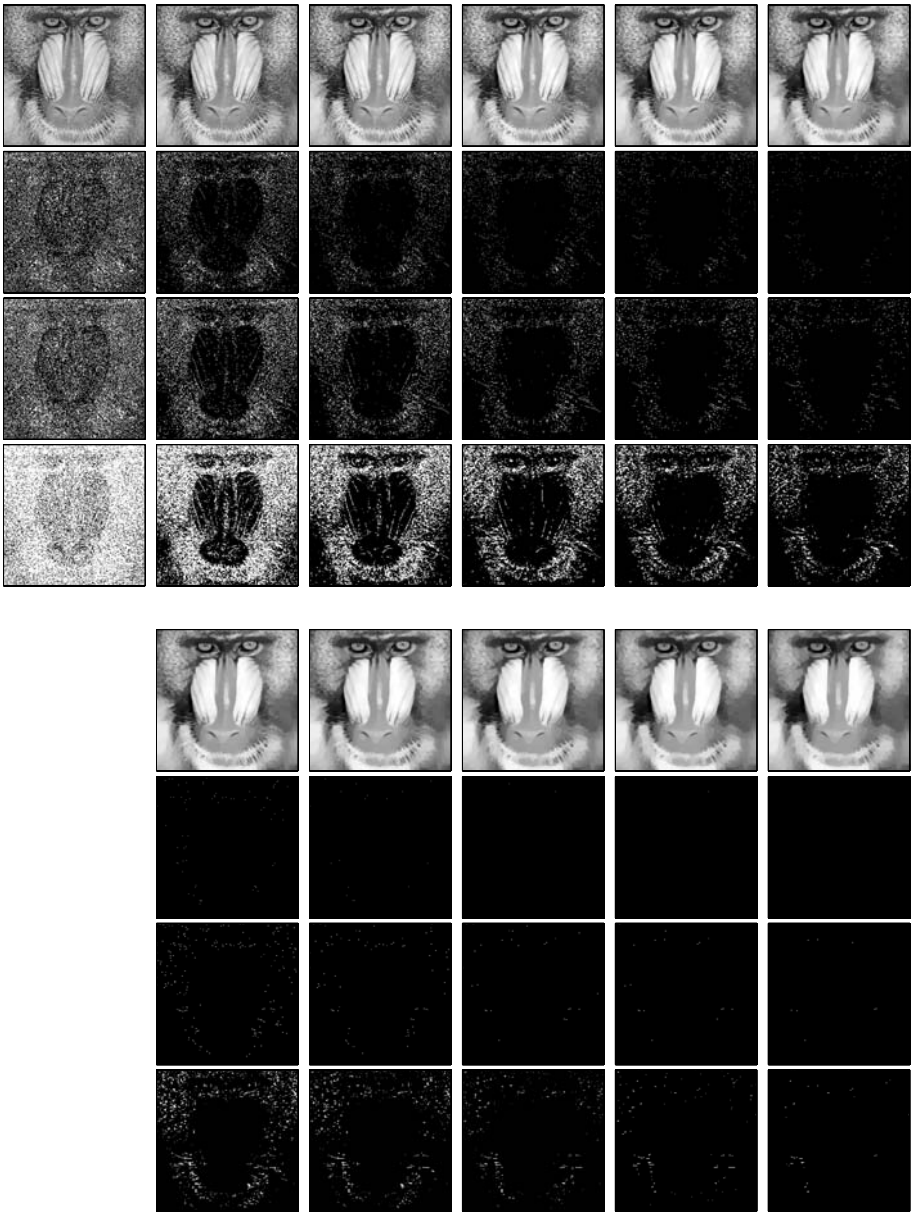


Figure 12: Results of applying TV regularization (2) to the Mandrill image. The eleven sets of images correspond to the original image plus the ten images resulting from solving (2) using $\alpha = 0.1\,\alpha_{thresh}$, $0.2\,\alpha_{thresh}$, ... , $1.0\,\alpha_{thresh}$. For each set (column) of images, the top image is the image itself, while the second through fourth images show the locations of all (portions of) features with scale at or below $1\,/\,4n$ (1 x 1 pixel), $1\,/\,3n$ (1 x 2) and $1\,/\,2n$ (2 x 2), respectively ($n = 256$).

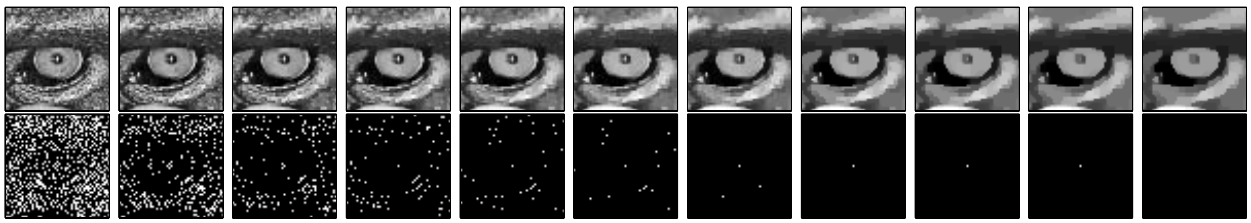


Figure 13: A portion of the two top images from the eleven sets of images in Figure 12. The eleven sets of images correspond to the original image plus the ten images resulting from solving (2) using $\alpha = 0.1\,\alpha_{1/1024}$, $0.2\,\alpha_{1/1024}$, ... , $1.0\,\alpha_{1/1024}$. For each pair of images, the top image is taken from the regularized image itself (the first row of images in Figure 12), while bottom image is taken from the image showing remaining features at or below scale 1/1024 (the second row of images in Figure 12).

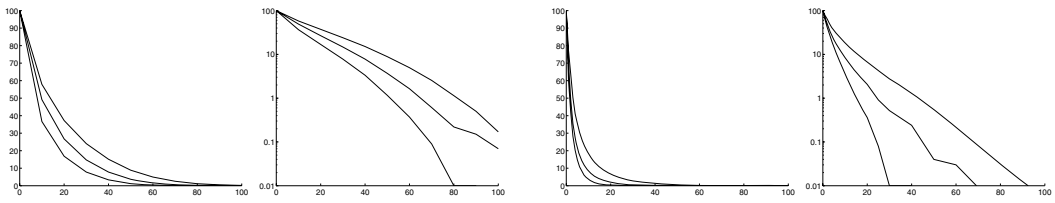
Mandrill Image

α, as % of $\alpha_{1/4n}$	% of scale remaining $1/4n$	$1/3n$	$1/2n$
0	100.00	100.00	100.00
10	36.77	49.15	57.88
20	16.85	26.73	37.25
30	7.79	14.74	24.03
40	3.34	7.79	15.11
50	1.16	3.68	8.87
60	0.37	1.64	4.96
70	0.09	0.61	2.54
80	0.01	0.22	1.14
90	0.01	0.15	0.50
100	0.00	0.07	0.17

Canaletto Image

α, as % of $\alpha_{1/2n}$	% of scale remaining $1/4n$	$1/3n$	$1/2n$
0	100.00	100.00	100.00
1	60.39	70.30	77.39
2	41.95	49.31	63.76
3	28.28	35.50	50.32
4	20.22	27.42	41.07
5	14.77	21.67	35.75
6	10.79	17.38	30.41
8	6.08	11.97	23.32
10	3.74	8.44	18.46
12	2.11	6.14	14.54
14	1.29	4.50	11.80
17	0.68	3.00	8.78
20	0.36	2.06	6.71
25	0.08	0.91	4.25
30	0.01	0.52	2.76
40	0.01	0.24	1.30
50	0.00	0.04	0.54
60	0.00	0.03	0.22
75	0.00	0.00	0.05
100	0.00	0.00	0.00

Table 3: The percentage of features at or below three specified scales remaining after applying TV regularization (2) to the Mandrill image (shown in Figure 12) and the Canaletto image (not shown) for various values of α. The three scales considered are $1/4n$, $1/3n$ and $1/2n$, which correspond to scales of 1 x 1, 2 x 1 and 2 x 2 pixel features, respectively. Each column shows the percentage of the orginal pixel locations recognized as being at or below the specific scale for the given value of α. α_{thresh} was found using using the α_{thresh} Algorithm. For the Mandrill image, we used $\alpha_{1/4n}$ (notice the 0.00 in the final entry of the 1 / 4n column) while for the Canaletto image we used $\alpha_{1/2n}$ (notice the 0.00 in the final entry of the 1 / 2n column).



(a) Decay of features in Mandrill image.

(b) Decay of features in Canaletto image.

Figure 14: A plot of the remaining percentages of scales listed in Table 3. For each pair of images, the first plot is the linear plot of the data, and the second plot is the log (in the y-axis) plot of the data. The nearly linear behavior seen in the log plots illustrates the nearly exponential decay of the image features of the three scales considered. For each plot, the three curves, from top to bottom, show the percentage of features at or below scale thresholds of $1/4n$, $1/3n$ and $1/2n$, respectively.

the scales corresponding to 1 x 2 pixel and 2 x 2 pixel features, respectively. The remaining percentage of features at or below each of the given scale thresholds for each value of α is given in the first table in Table 3.

In examining the images in Figure 12, it is apparent that most of the feature removal is relatively immediate, i.e. for the smaller values of α. For example, the second row of images shows the location of features whose scale corresponds to that of a single pixel. Although a value of $\alpha_{1/1024} = 0.00052$ is needed to completely remove *all* features of scale $\leq 1/1024$ from the image, even for $\alpha = 0.3\,\alpha_{1/1024}$ or $\alpha = 0.4\,\alpha_{1/1024}$, the image is almost entirely devoid of these one-pixel features. We demonstrate this in more detail for a portion of this image in Figure 13. Notice, in particular, that the one feature that is still present until the final image is the center of pupil of the Mandrill's left eye (the right eye, from our perspective). So if the goal is to remove all single-pixel features, perhaps a smaller value of α should be used, even if there are a few single-pixel features still remaining, in order to better preserve the (wanted) larger features. This decision will depend on the image and the reason for applying regularization.

In the images shown in Figure 12 and especially in the images shown in Figure 13, it is clear that once scale at any given location is recognized as being at or above a certain thresold, it will never drop below that threshold, and in fact, the scale at every location throughout the image will increase asymptotically to a maximum scale as α increases. The white "dots" in Figures 12 and 13 are the locations at which there are features at or below a given scale. Notice that you see only the disappearance of the dots, but no reappearance of dots or appearance of new dots anywhere.

6.2 Rate of loss of features

We last briefly examine the "decay" (rate of loss) of features of any given scale in an image. In the previous section we saw that we can recognize scale throughout the image. It is illuminating to look at the rate of decay of the remaining scale for increasing values of α. For the images seen in Figure 12, Table 3 gives us the percentage of all features at or below a given scale remaining for each value of α_{thresh}. These data are plotted in Figure 14(a).

As a second example we find the same information about remaining percentages at the same three scale levels for the Canaletto image. we first find $\alpha_{1/2n}$, the value of α_{thresh} where $scale_{thresh} = 1/2n$, e.g. the size of a 2 x 2 pixel square. In this second case, since most of the features for each of the three scales in the Mandrill image seemed to be removed rather quickly, we now use more values of α, particularly smaller values, in order to observe more gradually the decrease in percentages. These data are listed in the second table in Table 3 and are plotted in Figure 14(b).

For both images, in Figure 14, we first plot the standard (linear) plot of each scale percentage, and then we give the log (in the y-axis, which corresponds to the remaining percentage of features at or below a certain scale) plot of the same data. ¿From these plots, we see that the rate of loss or decay of the features at or below the three given scales seems nearly exponential for both images. Of course we could easily contrive an image for which scale decay is not exponential. Still, it may be that for a variety of natural images, scale decay would be exponential. That is, most of the features at or below a given scale disappear rapidly, while there are a few features that still remain for a while until α is too large. This was especially evident in Figures 12 and 13. This decay of scale and our ability to measure it using TV regularization certainly merit further investigation.

7 Summary and Conclusions

TV regularization naturally recognizes scale in an image. This allows us great insight into how TV regularizations works, and it leads to a number of ways in which this ability to recognize scale can be exploited. As shown, we can automatically and precisely determine how much regularization is needed (i.e. what value of α to choose) to remove all features at or below a given scale threshold from an image. We accomplish this using the α_{thresh} Algorithm, introduced in this paper, in which we find an optimal regularization parameter value based on the geometry of the image. There is a nice connection between Meyer's G Norm and both our notion of scale and our α_{thresh} Algorithm. This connection leads to a more intuitve explanation of the G norm and how it relates to scale in an image. The ability to recognize scale leads to a better understanding of already known TV-based ideas and schemes, including scalespace, and it leads to a number of new and potentially very useful tasks for manipulating and understanding images, including measuring the decay of

features of various scales in an image. Using this ability to measure scale, for the examples we considered, we have seen that most features at a given scale tend to disappear quickly, while a relatively small fraction persists longer. Some of the ideas investigated in this paper are essentially complete, and some of the work was intended to show how more possible avenues of investigation have been opened due to this ability to recognize scale. In particular, as we have explained in the paper, the definition of scale that we use here is valid for piecewise constant images, but is just an approximation in general. Future studies should be conducted on how to integrate Strang's exact definition of scale in numerical algorithms. Finally, although this work is done for images in $\mathbf{R}^2$, the theory developed can be extended to any function in any dimension. Other work that naturally stems from the work done in this paper includes a spatially adaptive α_{thresh} Algorithm and more efficient approaches to finding α_{thresh}, such as multigrid and domain decomposition approaches to the α_{thresh} Algorithm, which we are currently investigating.

A Appendix

Because some readers may be more familiar with values of α in solving (2) for a discrete n x n image when the domain is taken to be $[0,n]^2$ rather than $[0,1]^2$, we give the following lemma.

Lemma 3 *Let $\alpha_{[0,1]}$ and u be the regularization parameter and resulting image, respectively, when solving (2) on $[0,1]^k$. Similarly, let $\alpha_{[0,n]}$ and U be the regularization parameter and resulting image, respectively, when solving (2) on $[0,n]^k$. Then we will have $u = U$ (defined on their respective domains) when $\alpha_{[0,n]} = n\,\alpha_{[0,1]}$.*

Proof Let $[0,1]^k$ be the unit hypercube in R^k, and similarly for $[0,n]^k$. Let $\vec{t} = n\vec{x}$; that is, $(t_1, t_2, ..., t_k) = n(x_1, x_2, ..., x_k)$. Then $d\vec{t} = n^k\, d\vec{x}$. Let u_0 and U_0 be the original image, if defined on $[0,1]^k$ and $[0,n]^k$, respectively. Similarly, let u and U be the regularized image (the solution to (2)) if defined on $[0,1]^k$ and $[0,n]^k$, respectively. That is, for $\vec{t} \in [0,n]^k$, $U_0(\vec{t}) = u_0(\frac{\vec{t}}{n}) = u_0(\vec{x})$ and $U(\vec{t}) = u(\frac{\vec{t}}{n}) = u(\vec{x})$. Since $\frac{\partial u(\vec{x})}{\partial x_i} = \frac{\partial U(\vec{t})}{\partial x_i} = \frac{\partial U(\vec{t})}{\partial t_i}\frac{\partial t_i}{\partial x_i} = n\frac{\partial U(\vec{t})}{\partial t_i}$, then

$$\nabla u(\vec{x}) \;=\; (\frac{\partial u(\vec{x})}{\partial x_1}, \frac{\partial u(\vec{x})}{\partial x_2}, ..., \frac{\partial u(\vec{x})}{\partial x_k}) \;=\; (n\frac{\partial U(\vec{t})}{\partial t_1}, n\frac{\partial U(\vec{t})}{\partial t_2}, ..., n\frac{\partial U(\vec{t})}{\partial t_k}) \;=\; n\,\nabla U(\vec{t}).$$

Then (2) solved on domain $[0,1]^k$ can be related to (2) solved on domain $[0,n]^k$ as follows:

$$\begin{aligned}
\tfrac{1}{2}\|u-u_0\|^2 + \alpha_{[0,1]}\,TV(u) &= \tfrac{1}{2}\int_{\vec{x}\in[0,1]^k} [u(\vec{x}) - u_0(\vec{x})]^2\, d\vec{x} + \alpha_{[0,1]} \int_{\vec{x}\in[0,1]^k} |\nabla u(\vec{x})|\, d\vec{x} \\
&= \tfrac{1}{2}\int_{\vec{t}\in[0,n]^k} [U(\vec{t}) - U_0(\vec{t})]^2\, \tfrac{1}{n^k}\, d\vec{t} + \alpha_{[0,1]} \int_{\vec{t}\in[0,n]^k} n\,|\nabla U(\vec{t})|\, \tfrac{1}{n^k}\, d\vec{t} \\
&= \tfrac{1}{n^k}\left\{\tfrac{1}{2}\|U-U_0\|^2 + n\,\alpha_{[0,1]}\,TV(U)\right\}
\end{aligned}$$

And, of course,

$$arg\min_U\; \tfrac{1}{n^k}\left\{\tfrac{1}{2}\|U-U_0\|^2 + n\,\alpha_{[0,1]}\,TV(U)\right\} = arg\min_U\; \tfrac{1}{2}\|U-U_0\|^2 + n\,\alpha_{[0,1]}\,TV(U).$$

■

The above shows that for any integer value of k, changing the domain from $[0,1]^k$ to $[0,n]^k$ requires us to change α to $n\,\alpha$ in order to produce the same results. For example, for a 256 x 256 image, if $\alpha_{[0,1]} = 0.001$ on the domain $[0,1]^k$, then we would need $\alpha_{[0,256]} = 0.256$ if our domain were instead $[0,256]^k$, in order to produce the same regularized image in their respective domains.

References

[1] R. Acar and C. Vogel. Analysis of total variation penalty methods. *Inverse Problems*, 10:1217–1229, 1994.

[2] H. Aly and E. Dubois. Image up-sampling using total-variation regularization with a new observation model. *IEEE Trans. Image Processing*, 2005. to appear.

[3] L. Ambrosio, N. Fusco, and D. Pallara. *Functions of bounded variations and free discontinuity problems*. Oxford mathematical monographs. Oxford University Press, 2000.

[4] F. Andreu, V. Caselles, J. Diaz, and J. Mazon. Some qualitative properties for the total variation flow. *J. Functional Analysis*, 188:516–547, 2002.

[5] G. Aubert and J-F. Aujol. Modeling very oscillating signals. Application to image processing. *Applied Mathematics and Optimization*, 51(2), March/April 2005.

[6] J-F. Aujol, G. Aubert, L. Blanc-Féraud, and A. Chambolle. Image decomposition into a bounded variation component and an oscillating component. *J. Mathematical Imaging and Vision*, 22(1), January 2005.

[7] J-F. Aujol and A. Chambolle. Dual norms and image decomposition models. *International J. Computer Vision*, 63(1), June 2005.

[8] J-F. Aujol, G. Gilboa, T. Chan, and S. Osher. Structure-texture image decomposition – modeling, algorithms, and parameter selection. *International Journal on Computer Vision*, 2005. in press.

[9] G. Bellettini, V. Caselles, and M. Novaga. The total variation flow in $\mathbf{R}^n$. *J. Differential Equations*, 184:475–525, 2002.

[10] P. Blomgren and T. Chan. Color TV: total variation methods for restoration of vector-valued images. *IEEE Trans. Image Processing*, 7:304–309, 1998.

[11] T. Brox and J. Weickert. A TV flow based local scale measure for texture discrimination. In *ECCV 04*, volume 2, pages 578–590, May 2004.

[12] A. Chambolle. An algorithm for total variation minimization and applications. *J. Mathematical Imaging and Vision*, 20:89–97, January 2004.

[13] A. Chambolle and P-L. Lions. Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76:167–188, 1997.

[14] T. Chan, G. Golub, and P. Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM J. Scientic Computing*, 20:1964–1977, 1999.

[15] T. Chan and J. Shen. Mathematical models for local non-texture inpaintings. *SIAM J. Applied Mathematics*, 62:1019–1043, 2001.

[16] T. Chan and C.-K. Wong. Total variation blind deconvolution. *IEEE Trans. Image Processing*, 7:370–375, 1998.

[17] Q. Chang and I-L. Chern. Acceleration methods for total variation-based image denoising. *SIAM J. Applied Mathematics*, 25:982–994, 2003.

[18] D. Dobson and F. Santosa. Recovery of blocky images from noisy and blurred data. *SIAM Journal on Applied Mathematics*, 56:1181–1198, 1996.

[19] D. Dobson and C.R. Vogel. Convergence of an iterative method for total variation denoising. *SIAM Journal on Numerical Analysis*, 34:1779–1971, 1997.

[20] G. Gilboa, N. Sochen, and Y. Y. Zeevi. PDE-based denoising of complex scenes using a spatially-varying fidelity term. In *Proc. ICIP 2003, Barcelona, Spain*, volume 1, pages 865–868, 2003.

[21] Y. Gousseau and J-M. Morel. Are natural images of bounded variation? *SIAM J. Mathematical Analysis*, 33:634–648, 2001.

[22] Yves Meyer. Oscillating patterns in image processing and in some nonlinear evolution equations. Providence, RI, 2001. The Fifteenth Dean Jacquelines B. Lewis Memorial Lectures.

[23] S.J. Osher, A. Sole, and L.A. Vese. Image decomposition and restoration using total variation minimization and the H^{-1} norm. *SIAM Journal on Multiscale Modeling and Simulation*, 1(3):349–370, 2003.

[24] P. Perona and J. Malik. Scale space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 12:629–639, 1990.

[25] E. Radmoser, O. Scherzer, and J. Weickert. Scale-space properties of regularization methods. In *Scale-space theories in computer vision, Lecture Notes in Computer Science, Vol. 1682, Springer, Berlin, M. Nielsen, P. Johansen, O.F. Olsen, J. Weickert (Eds.)*, pages 211–222, 1999.

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

[27] G. Strang. L^1 and L^∞ approximation of vector fields in the plane, 1982. Lecture Notes in Num. Appl. Anal, Volume 5.

[28] D. Strong. *Adaptive Total Variation Minimizing Image Restoration.* PhD thesis, UCLA Math Department, August 1997. CAM Report 97-38.

[29] D. Strong and T. Chan. Edge-preserving and scale-dependent properties of total variation regularization. *Inverse Problems*, 19(6):165–187, 2003.

[30] E. Tadmor, S. Nezzar, and L. Vese. A multiscale image representation using hierarchical (BV, L^2) decompositions. *SIAM journal on Multiscale Modeling and Simulation*, 2(4):554–579, 2003.

[31] A. Tikhonov and V. Arsenin. *Solutions of ill-posed problems.* Winston and Sons, Washington D.C, 1977.

[32] L. Vese and S. Osher. The level set method links active contours, mumford-shah segmentation, and total variation restoration, February 2002. UCLA CAM Report 02-05, available at http://www.math.ucla.edu/applied/cam/index.html.

[33] C.R. Vogel. *Computational methods for inverse problems.* Philadelphia, PA. SIAM, 2002.

[34] C.R. Vogel and M.E. Oman. Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing*, 17:227–238, 1996.