Algorithms for Reducing Noise in Synthetic Aperture Radar Images

Troy Peterson Kling & Jeffrey Kidwell

Abstract—Images of Earth’s surface gathered by Uninhabited Aerial Vehicle Synthetic Aperture Radar (UAVSAR) contain a plethora of information about the nature of the scattering media covering the ground. Depending on the type of instrument used to scan the landscape, a combination of radiometric, polarimetric, and interferometric data can be gathered. This data can be used for a wide range of purposes, from conducting biomass studies to building digital elevation maps.

Despite its usefulness, synthetic aperture radar confers a speckled appearance to the images it takes of Earth’s surface. This speckle noise hinders efforts to analyze image contents, and therefore it is necessary to develop new, efficient techniques for removing this noise. Non-local approaches are particularly well-suited for this task, since they are able to reduce speckle noise while preserving important structures in the image.

In this paper several different speckle filters are described and analyzed. A brief overview of local filters is conducted to show why more advanced de-noising techniques are necessary. Non-local filters are then introduced and applied to optical images. Finally, each of the filters is applied to a synthetic aperture radar image of Earth’s surface to see how it performs on a problem that has significant real-world applications.

I. INTRODUCTION

Remote sensing images of Earth’s surface gathered by NASA JPL’s Uninhabited Aerial Vehicle Synthetic Aperture Radar (UAVSAR) provide valuable information about the nature of the scanned landscape. The type of scattering media on the surface (e.g. forests, vegetation, buildings, etc.) affects the polarization of the radio waves received by the SAR instrument. In general, three different channels are used for polarization information - HH, VV, and HV - and the intensity and phase of each channel varies based on the type of scattering media [1]. This polarimetric SAR (PolSAR) data allows researchers to characterize images of the Earth’s surface by different types of land cover. In this paper, polarimetric is mostly ignored; only the magnitude of the HH channel is used.

Interferometric SAR (InSAR), on the other hand, performs multiple parallel passes over a single area [2]. These passes are separated by a spatial baseline and are acquired either simultaneously or in a repeat-pass configuration. By detecting small changes in the phase of the received signals it is possible to determine the height of the scattering media. This makes it possible to construct digital elevation models (DEMs) of the Earth’s surface [3]. InSAR can also be used to detect temporal changes in a landscape by comparing data from two passes separated by time. In this paper, interferometry is mostly ignored; only the image taken on the first pass is used.

Despite its usefulness, SAR imagery is inherently affected by speckle noise. This random fluctuation in the received signal degrades the quality of the image and prevents accurate analysis of its contents. As a result, many filters have been developed to reduce speckle noise in SAR imagery. A speckle filter is a mathematical technique for reducing noise in images and potentially recovering information that was lost due to noise. By reducing the granularity of the image, one hopes to get a more accurate representation of the actual landscape. Once this is accomplished, existing algorithms can be employed to gather qualitative and quantitative information about the contents of the image.

A. Problem Description

1) Natural Language Description: Given a noisy synthetic aperture radar image and a selection of image filters (Boxcar, Gaussian convolution, Lee’s filter, non-local means, Fourier transform filter, and modified ART-2 neural networks), the goal is to select the filter that best reduces noise in the image while also retaining image clarity by preserving edge details.

2) Formal Description: Given an n-by-m synthetic aperture radar image, X, with speckle noise modeled as a standard complex Gaussian distribution [4]

\[ p(z) = \frac{1}{\pi} \exp(-|z|^2) \]  

where \( z = Ae^{i\theta} \) is a vector with amplitude \( A \) and phase \( \theta \), and given a set of noise reduction filters \( F = \{f_1, f_2, ..., f_k\} \), find \( f_p \in F \) such that

\[ RMSE(f_p(X)) \leq RMSE(f_q(X)) \forall f_q \in F \]

where

\[ RMSE(f_q(X)) = \frac{1}{nm} \sum_{i=0}^{n} \sum_{j=0}^{m} (f_q(X_{ij}) - T_{ij})^2 \]

and where \( T \) is the ideal, non-noisy image.

Equivalently, one could also attempt to find \( f_p \in F \) such that the objective function defined by the peak signal-to-noise ratio (PSNR) is maximized.

\[ PSNR(f_p(X)) = 20 \cdot \log_{10} \left( \frac{MAX_f}{RMSE(f_p(X))} \right) \]

where \( MAX_f = 2^B - 1 \) and \( B \) is the number of bytes used to represent pixel values in the image.
II. IMAGE FILTERING ALGORITHMS

Image filtering algorithms first came onto the scene in the late 1970s and early 1980s, and they are still being developed and improved upon to this day. Statistical image filters can be broken down into two major categories: local and non-local. Local filters attempt to estimate each pixel’s true intensity by looking at the pixels immediately surrounding it (hereafter known as that pixel’s neighborhood). This assumes that pixels nearby to one another are relatively similar. Non-local filters reject this idea and are instead built upon the principle that two pixels can belong to the same statistical distribution even if they are located quite far from one another in the image. In general, this allows non-local filters to perform better noise reduction than local filters.

Filters from both of these paradigms - and some from other paradigms entirely - will be introduced and analyzed in this section. The standard image of Lena is used throughout this paper to show how well different filters perform. Figure 1 shows what the image of Lena looks like before and after noise was programmatically added to it. Each of the image filters used in this section will be taking the noisy image as input and outputting a denoised image that hopefully looks similar to the original, non-noisy image of Lena. The RMSE and PSNR statistics will be used throughout this paper to gauge how well the filters perform relative to one another. As a benchmark, the RMSE between the original image of Lena and the noisy version is 14.628, and the PSNR is 24.827.

A. Boxcar Filter

The Boxcar filter is one of the simplest local filters in existence. It reduces the overall variation present in an image by setting each pixel’s intensity equal to the average of its neighborhood. In this way, it acts as a moving average that uniformly blurs the image.

Let $X_i$ be a pixel from the unfiltered image, and let $Y_i$ be the corresponding pixel in the filtered image. Then the Boxcar filter can be written as follows:

$$Y_i = Box(X_i) = \frac{1}{|W_i|} \sum_{j \in W_i} X_j$$  \hspace{1cm} (5)

where $W_i$ denotes the set of all pixels in pixel $i$’s neighborhood.

Figure 2 shows how well the 5x5 Boxcar filter works. As stated previously, the noisy image is uniformly blurred. This does reduce the granularity of the image, but it also blurs important edges in the image, causing crucial information to be lost. In this example, the RMSE between the original image and the denoised image is 10.444, and the PSNR is 27.753.

This RMSE value is slightly lower than the RMSE for the noisy image, which should indicate to the reader that the Boxcar filter produced a denoised image that more closely approximated the true, non-noisy image. Similarly, the higher PSNR value indicates that the level of noise has been reduced, allowing the original signal to more accurately cut through the remaining noise.

B. Gaussian Convolution

Before discussing how the Gaussian convolution filter works, it is essential to first define what convolution is. For one-dimensional signals $f$ and $g$, convolution is defined as

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$ \hspace{1cm} (6)

This procedure can be visualized by sliding the signal $f$ over the length of $g$ and recording the area under the curve at each infinitesimal step. This is similar to calculating the cross-correlation between two signals.

Gaussian convolution is simply the process of convolving the original signal with a Gaussian kernel. This terminology lends itself to a one-dimensional description, but Gaussian convolution can easily be extended to work with images in two dimensions.

The multivariate normal distribution is defined as

$$f(x) = \frac{1}{\sqrt{(2\pi)^k|\Sigma|}} \exp\left(-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$ \hspace{1cm} (7)

where $x = [x_1, x_2, ..., x_k]^T$, $\mu$ is a vector of means, and $\Sigma$ is the covariance matrix.

For a symmetric bivariate normal distribution, one can create a discrete version of a Gaussian kernel by simply sampling the density at certain points and assuming all values below some threshold equal zero. For example, the following 5x5 truncated Gaussian kernel could be used.

$$K_G = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$ \hspace{1cm} (8)

Note that $K_G$ has the nice property that its values sum to 1, as is true of all probability density functions.

Gaussian convolution can then be achieved by convolving an image with this kernel. Similar to the Boxcar filter, this amounts to a moving average achieved using a sliding window that has the same dimensions as $K_G$.

$$Y_i = \sum_{j \in W_i} K_G(j) \cdot W_i(j)$$ \hspace{1cm} (9)

The key difference between this method and the Boxcar filter is that Gaussian convolution weights pixels closer to the center of the window more heavily than those near the border, whereas the Boxcar filter weights all pixels equally.

Figure 3 shows the result of applying a Gaussian convolution filter to the noisy image of Lena. Unsurprisingly, the denoised image looks very similar to the one obtained by using the Boxcar filter. However, the Gaussian convolution filter is slightly more powerful. The RMSE between the original image and the denoised image is 9.506, and the PSNR is 28.571. This indicates that Gaussian convolution is capable of reducing noise better than the Boxcar filter, assuming a good value is chosen for $\sigma$. 

\[
\begin{bmatrix}
2 & 4 & 5 & 4 & 2 \\
4 & 9 & 12 & 9 & 4 \\
5 & 12 & 15 & 12 & 5 \\
4 & 9 & 12 & 9 & 4 \\
2 & 4 & 5 & 4 & 2
\end{bmatrix}
\]
C. J. S. Lee’s Filter

The Lee filter [5], [6], developed by J. S. Lee, is an example of an adaptive local filter. Compared to the Boxcar and Gaussian convolution filters, the Lee filter performs better at preserving sharp edges and point scatterers. It does this by using an adaptive filtering coefficient and attempting to minimize the mean squared error between the actual denoised image and our estimate of it.

The Lee filter can be written as follows.

\[ Y_i = \text{Lee}(X_i) = \text{Box}(X_i) + k_i(X_i - \text{Box}(X_i)) \]  

(10)

where \( \text{Box}(X_i) \) is defined in Equation (5) and \( k_i \) is an adaptive filtering coefficient defined by

\[ k_i = \frac{\text{Var}(Y)}{\text{Var}(W_i)} = \frac{\text{Var}(W_i) - E^2(W_i)\sigma^2}{\text{Var}(W_i)[1 + \sigma^2]} \]  

(11)

where \( \text{Var}(\cdot) \) is the variance operator, \( E(\cdot) \) is the expectation, and \( \sigma^2 \) is the \textit{a priori} variance of the speckle noise.

The adaptive filtering coefficient, \( k_i \), is affected by the level of heterogeneity in a given region. If the neighborhood of a given pixel is mostly homogeneous, then \( k_i \approx 0 \), and the Lee filter acts as a simple Boxcar filter. However, over regions of heterogeneity, \( k_i \approx 1 \) makes it so that \( Y_i = X_i \), which sets the filtered pixel intensity equal to the unfiltered pixel intensity, effectively doing no noise reduction whatsoever. In regions of moderate heterogeneity, \( k_i \) taken on values between 0 and 1, which allows the Lee filter to find a middle-ground between the two extremes of perfect homogeneity and perfect heterogeneity. This malleability allows Lee’s filter to effectively reduce noise over homogeneous regions while leaving fine structures and details intact.

Lee’s filter can be improved by implementing directional masks. While this addition does help the filter better preserve sharp edges and point scatterers, it can also introduce unwanted artifacts into the filtered image.

Figure 4 shows the result of applying a Lee filter to the noisy image of Lena. The denoised image possesses a level of clarity that could not be achieved using the previously discussed filters. Regions of the image that contain high variability (e.g. Lena’s eyes and the brim of her hat) have been preserved, whereas regions of low variability have been smoothed over. This behavior is made possible by the adaptive filtering coefficient. The RMSE between the original image and the denoised image is 8.740, and the PSNR is 29.300. This indicates that the Lee filter is a better candidate for noise reduction than the Boxcar and Gaussian convolution filters.
D. Non-local Means

Local filters assume that, for any given neighborhood in an image, all nearby pixels come from the same statistical distribution as the center pixel. In other words, pixels that are close to one another are assumed to be relatively similar. Non-local filters reject this idea. From a non-local perspective, there is no reason to suspect that proximity between pixels is a good measure of similarity [7]. Instead, these filters work based on the principle that two pixels can belong to the same statistical distribution even if they are located quite far from one another.

One approach in particular, the non-local means algorithm [3], [8], [7], attempts to filter an image by defining each pixel’s neighborhood in a new way. Instead of restricting a pixel’s neighborhood to the area immediately surrounding it, the non-local means algorithm considers all pixels in the entire image. By comparing a small window around one pixel to a small window around all others, the filter is able to gauge their similarity. Using this knowledge about similarity within an image, the filter is capable of more accurately estimating each pixel’s true intensity.

In the non-local paradigm, each pixel is assigned a new value equal to a weighted sum of all pixels in the image (or all pixels in a large search window around the center pixel). Therefore, the non-local means filter can be written as follows

\[ Y_s = NL(X_s) = \sum_{t \in W_s} w(s,t)X_t \]  

where \( w(s,t) \) is a weight associated with each pair of pixels, calculated as follows.

\[ w(s,t) = \frac{1}{Z(s)} e^{-\frac{d^2(s,t)}{h^2}} \]  

where \( h \) is a tuning parameter that controls the rate of exponential decay, \( d^2(s,t) \) is the Euclidean distance between the two similarity windows, and \( Z(s) \) is the normalizing constant

\[ Z(s) = \sum_j e^{-\frac{d^2(s,t)}{h^2}} \]  

An illustration of the non-local means algorithm is shown in Figure 5. In this diagram, the search window \( W_s \) has size 7x7 and the similarity windows \( \Delta_s \) and \( \Delta_t \) have size 3x3, but any size can be chosen for these windows. In general, the larger the search window the better the filter performs. However, very large window sizes may cause the algorithm to run quite slowly.
Pseudocode for the non-local means algorithm can be written as follows:

**Input** a noisy image, $X$.

For each pixel $s \in X$:
- Generate a large, square search window, $W_s$, around $s$.
- Generate a smaller similarity window, $\Delta_s$, around $s$.
  For each pixel $t \in W_s$:
    - Generate a similarity window, $\Delta_t$, around $t$.
    - Calculate $w(s, t)$ using Equation (13).
  End for loop.

Estimate the filtered pixel intensity using Equation (12).
End for loop.

**Output** the denoised image, $Y$.

This pseudocode hides some of the complexity of the algorithm, but it gives a good overview of how the filter works. Combined with the diagram in Figure 5, this should be enough information to assist readers in implementation.

Figure 6 shows the result of applying a non-local means filter to the noisy image of Lena. 5x5 similarity windows and 35x35 search windows were used to produce this result. Larger windows can be used for slightly better results, but the time trade-off is likely not worth it. The value of the filtering parameter, $h$, was chosen by hand.

By visually inspecting the images in Figure 6, it is plain to see that the non-local means filter does an extraordinary job of reducing noise. It does everything the Lee filter does (smoothing over homogeneous regions, preserving edges and other fine structures), but it also accurately reduces noise in regions of high variance.

The RMSE between the original image and the denoised image is 6.158, the lowest value we have achieved so far. Similarly, the PSNR is 32.342, which is the highest value we have achieved so far. These statistics reflect the fact that the non-local means filter performs much better than any of the filters previously discussed.

### E. Fourier Transform

A Fourier transform takes a signal in the time domain and transforms it into the complex-valued frequency domain. The Fourier transform of a function $f$ is defined as

$$f(x) = \int_{-\infty}^{\infty} F(k)e^{2\pi ikx} dk \quad (15)$$

This transformation can be reversed with the inverse Fourier transform

$$F(x) = \int_{-\infty}^{\infty} f(k)e^{-2\pi ikx} dk \quad (16)$$

The transformations defined above are in terms of one-dimensional signals, but a similar process can be used to compute the Fourier transform of an image.

Noise reduction can be achieved by taking the Fast Fourier Transform (FFT) of an image, filtering certain frequencies, and then performing the inverse FFT. The filtering process can take any number of forms (e.g. low-pass filter, high-pass filter, band-pass filter). We chose to keep only those frequencies that were in the range $(-12.75, 12.75)$, setting the rest to zero.

Figure 7 shows the result of applying a Fourier transform that utilizes a bandpass filter to the noisy image of Lena. The results are not ideal. There is still a fair amount of noise in the image, and many of the sharp edges have been blurred. In this way, the simplistic Fourier transform filter is similar in performance to the Boxcar and Gaussian convolution filters. The RMSE between the original image and the denoised image is 10.197, and the PSNR is 27.961. While these values are slightly better than the results for the Boxcar filter, they are not anywhere close to the results produced by non-local means.

It should be noted that this is not the only way in which Fourier transforms can be applied to the problem of noise reduction. More advanced algorithms exist that filter the image in a more adaptive manner, producing better results. One can also investigate wavelet transforms for even greater noise reduction.
F. Modified Adaptive Resonance Theory

Developed by Stephen Grossberg and Gail Carpenter, the adaptive resonance theory model (ART-2) is a neural network that was originally designed for classifying input patterns into categories based on their similarity [9]. This neural network can be trained to recognize patterns in signals and images so that it can correctly classify new input patterns that it receives. ART-2 can be applied to many different classification problems, but it can also be used for noise reduction.

Figure 8 shows a diagram of the ART-2 neural network. There are six nodes in the $F_1$ layer (which can be thought of as one big node itself), labeled $w_i, x_i, v_i, u_i, p_i, q_i$. This layer functions as the neural network’s short-term memory. There is also an $F_2$ layer, which functions as long-term memory. This layer contains a number of nodes equal to the desired number of categories. The bottom-up and top-down weights $z_{ji}$ and $z_{ij}$ are used to communicate between the $F_1$ and $F_2$ layers.

The short-term memory is governed by the following equations. Note that $e$ can be set to zero, which effectively turns $x_i$ into the norm of $w_i$, $u_i$ into the norm of $v_i$, and $q_i$ into the norm of $p_i$. This normalization step is indicated by filled black circles in Figure 8.

\[
\begin{align*}
  p_i &= u_i + \sum_j g(y_j)z_{ji} \\
  q_i &= \frac{p_i}{e + ||p_i||} \\
  u_i &= \frac{v_i}{e + ||v_i||} \\
  v_i &= f(x_i) + b f(q_i) \\
  w_i &= I_i + au_i \\
  x_i &= \frac{w_i}{e + ||w_i||}
\end{align*}
\]

One should also note that there exists a filter function, $f$, that is used for noise suppression. This function can take different forms, but for our purposes we define it as a piecewise linear function.

\[
f(x) = \begin{cases} 
  0 & \text{if } 0 \leq x < \theta \\
  x & \text{if } x \geq \theta
\end{cases}
\]

where $\theta$ is a threshold variable.

This filter function is where the built-in noise reduction step takes place in ART-2. However, we would like to extend ART-2’s noise reduction capability by implementing some ideas from the non-local means filter.

Given a noisy image, $X$, we can divide it up into a series of overlapping windows. If we were to take these windows and
treat them as signal patterns, then we could feed them into the ART-2 neural network. ART-2 would then place them into a pre-determined number of categories based on their similarity to one another. After this process completes, we would know which windows in the noisy image are most similar to one another.

Given this knowledge about window similarity, we could attempt to filter the noisy image in several different ways. One option is to take a page out of the non-local means book and compute a weighted average within each category. Each intra-category average could then be substituted for the original windows belonging to that category, thereby reducing noise in the image. However, it is unclear whether or not this would work properly.

We could also take a more simplistic approach. Instead of performing a weighted average within each category based on Euclidean distances, we could compute the unweighted average within each category and average that with the original window in order to get our new, denoised window. This would allow for less uniformity in the resulting image, which is likely a good thing.

At this point in time, all of these ideas are theoretical. It is even possible that the standard ART-2 neural network performs enough noise reduction on its own, and no additional steps are necessary. We will not know which scheme is the best at noise reduction until we implement them all and test them out. Since we have not yet implemented ART-2, we have no results for this section.

### III. Conclusion

#### A. Results for Optical Images

From the figures shown previously, it is clear to see that the non-local means filter performs the best out of all the filters we researched. Although it remains to be seen whether or not the modified ART-2 neural network will outperform the standard non-local means algorithm, we can safely conclude that non-local means is at least a front-runner for the best image filter.

<table>
<thead>
<tr>
<th>Filter</th>
<th>RMSE</th>
<th>PSNR</th>
<th>Big-O</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-local means</td>
<td>6.16</td>
<td>32.34</td>
<td>$O(nm^2k^2)$</td>
</tr>
<tr>
<td>J. S. Lee’s filter</td>
<td>8.74</td>
<td>29.3</td>
<td>$O(nmk^2)$</td>
</tr>
<tr>
<td>Gaussian convolution</td>
<td>9.51</td>
<td>28.57</td>
<td>$O(nmk^2)$</td>
</tr>
<tr>
<td>Fourier transform filter</td>
<td>10.20</td>
<td>27.96</td>
<td>$O(nm \log(nm))$</td>
</tr>
<tr>
<td>Boxcar filter</td>
<td>10.44</td>
<td>27.75</td>
<td>$O(nmk^2)$</td>
</tr>
<tr>
<td>Modified ART-2</td>
<td>N/A</td>
<td>N/A</td>
<td>O(?)</td>
</tr>
</tbody>
</table>

Figure 9 summarizes the results obtained in the previous section, and it also includes the algorithmic complexity (using Big-O notation) for each of the filters. The filters are ordered by ascending RMSE; those higher in the list produced lower RMSE values, which indicates that they were better at reducing noise. There are no entries for the modified ART-2 filter, because we have not yet finished implementing it. Considering SAR images are many thousands of pixels wide, non-local means would take a very long time filtering the entire image.

Figure 10 shows how long each filtering algorithm took to execute for images of varying sizes, from 16x32 pixels to 512x1024 pixels. The Boxcar, Gaussian convolution, and FFT filters all run extremely quickly. The Lee filter takes somewhat longer to execute, because it spends a lot of time computing local statistics. The non-local means filter takes longest of all to execute, with average runtime approaching one minute when the image is quite large.

#### B. Results for SAR Images

So far, we have only seen image filters applied to standard photographs. Images taken by a synthetic aperture radar instrument are an entirely different beast, so how can we be sure our algorithms will work equally well on them? Since SAR images are inherently noisy, there is no surefire way to determine what the true non-noisy image should look like.

However, since we understand how noise is modeled in SAR imagery, we should be fairly confident that the image filters we discussed previously will work equally well on them. And since we’re only working with single-channel magnitudes, we don’t have to worry about coherence between polarimetric or interferometric channels. Denoising the single-channel magnitude of a SAR image is very similar to denoising an optical image, so our algorithms should work just fine.

Indeed, Figures 11 and 12 show just how well the different filters perform on a SAR image. The landscape being imaged in these figures is a track of land in Harvard Forest, Massachusetts. For the most part, the forested areas should appear to be homogeneous. However, these regions are heavily
affected by speckle noise. Applying the different filters to this noisy SAR images produces varying results.

The Boxcar filter uniformly blurs the image, just as it did for the test image of Lena. Gaussian convolution has a similar effect, although the result appears to be slightly better. Much the same can be said about the Fourier transform filter. The Lee filter manages to significantly reduce noise, while preserving edges and point scatterers. However, these areas of high variance are not smoothed over at all. The non-local means filter improves upon this by performing additional smoothing over the forested regions and also reducing noise around edges and point scatterers, without excessively blurring them. This agrees with results obtained for optical images.

In conclusion, we have shown that the non-local means filter is by far the best image filter we have investigated. However, it is also the least tractable of the filters if large images are being used. SAR images are usually much larger than the examples we have included here, so running them through the non-local means filter could take an appreciable amount of time. We therefore leave the choice of image filter up to the researcher. The Lee filter is great if you need an image filtered quickly, and the non-local means filter is excellent if you need an image filtered optimally.

C. Future Work

This research was conducted during the Spring semester of 2015 for the course CSC 380 - Design and Analysis of Algorithms. Over the course of the semester, we only had enough time to research and implement a handful of noise reduction algorithms. We also restricted our tests to optical images and single-channel HH magnitude SAR images.

In the future, we plan to:
- Finish implementing ART-2.
- Test out Daubechies wavelet transforms as a replacement for the Fourier transform filter.
- Extend the existing filters to work with polarimetric and/or interferometric SAR images.

Currently, the most promising area of research seems to be combining ART-2 with the non-local means filter. We expect good results from this hybrid model.

D. Acknowledgements

Thanks to Dr. Maxim Neumann of NASA's Jet Propulsion Laboratory for supplying the synthetic aperture radar images of Harvard Forest, Massachusetts, and to Dr. Gene Tagliarini for introducing us to ART-2.


Fig. 11. *Top row:* Original SAR image; Boxcar; Gaussian convolution. *Bottom row:* Lee’s filter; non-local means; FFT filter.

### E. Questions

This sections contains questions that readers can be expected to answer after familiarizing themselves with the content of this paper.

**Q1.** What is the difference between local and non-local filters, and why do non-local filters generally perform better at noise reduction than local filters?

**A1.** Local filters estimate each pixel’s intensity by looking at the pixels immediately surrounding it. Non-local filters estimate each pixel’s intensity by looking at every pixel in the *entire* image (or at least a very large search window). Assuming the image has some degree of self-similarity, this will result in the non-local means filter performing better than a local filter.

**Q2.** The non-local means filter gave the lowest RMSE out of all the filters we discussed, but it’s not the best in every respect. What is one drawback of non-local means?

**A2.** The non-local means algorithm has complexity $O(n \cdot m \cdot w^2 \cdot k^2)$, where $n \cdot m$ is the size of the image, $w$ is the width of the search window, and $k$ is the width of the similarity window. For even modestly large values of $n$, $w$, and $k$, non-local means takes a long time to finish running.

**Q3.** For the Lee filter, how does the adaptive filtering coefficient, $k_i$, affect the filtering procedure?

**A3.** Homogeneous regions of the image produce $k_i$ values close to zero, which causes the Lee filter to act more like a Boxcar filter. Heterogeneous regions of the image that contain a lot of variation produce $k_i$ values close to one, which causes the filtered pixel value to be set equal to the unfiltered pixel value (thereby preserving fine structures in the image). For regions of the image in between the extremes of perfect homogeneity and perfect heterogeneity, $k_i$ adapts accordingly.

**Q4.** What was the ART-2 system originally designed for, and how can it be applied to noise reduction?

**A4.** The ART-2 neural network was originally designed for categorization of signal patterns. ART-2 can be applied to noise reduction in several ways - one of which is by categorizing overlapping windows in the image and then applying a filter function for each category, in a similar manner to how the non-local means filter works. However, ART-2 can also perform noise reduction/suppression on its own - it is built into the system.
Fig. 12. Top row: Original SAR image; Boxcar; Gaussian convolution. Bottom row: Lee’s filter; non-local means; FFT filter.

REFERENCES


