4.1 Introduction

Denoising techniques that are based on modifying the transform of an image are considered here. In these techniques, a reversible, linear transform (such as transforms discussed in Chapter 2) is used to map the noisy image into a set of transform coefficients, which are then filtered using a suitable thresholding technique.

Fig. 4.1 shows a typical denoising system that uses transform techniques. The system performs three relatively straightforward operations: transformation, thresholding, and inverse transform. The transformation process packs as much information as possible into the smallest number of transform coefficients. Thresholding [8] can be accomplished by hard thresholding, which means setting to zero the elements whose absolute values are lower than the threshold, or by soft thresholding, which involves first setting to zero the elements whose absolute values are lower than the threshold and then scaling the nonzero coefficients toward zero. The inverse transform reconstructs the original image.
In this chapter practical implementations of proposed denoising technique is focused. The next section will briefly survey existing methods, while section 4.4 explains thresholding techniques and section 4.5 demonstrates the algorithm of new denoising technique. Finally, the superiority of proposed technique over conventional methods is demonstrated.

### 4.2 Survey of Existing Denoising Methods

Over the last decade there has been abundant interest in wavelet methods for noise removal in signals and images. Many papers have been published in various fields of scientific and engineering disciplines, where a wide range of wavelet-based tools and ideas have been studied and proposed. Very simple ideas like thresholding [60-66] of wavelet coefficients of the noisy data, followed by reconstruction, was used initially. Later translation invariant methods, based on thresholding of an undecimated wavelet transform, were used to improve the perceptual quality. More recently, “tree-based” [67] wavelet denoising methods were developed in the context
of image denoising, which exploit the tree structure of wavelet coefficients.

4.3 New Multiscale Transform

To overcome the problems associated with wavelets, Candes and Donoho introduced new multiscale systems like curvelets, which is very different from Wavelet-like systems. Curvelets[1] take the form of basis elements which exhibit very high directional sensitivity and is highly anisotropic. In two-dimensions, for instance, curvelets are localized along curves, in three dimensions along sheets, etc. The details of curvelet transform are given in section 2.2 of chapter 2.

4.4 Thresholding

4.4.1 Background and Problem Formulation

In numerous applications of image acquisition or transmission, the images are often corrupted by additive noise. The goal of denoising is to remove the noise while retaining as much as possible the important image features. An obvious question which arises is how signal and noise are distinguished. If appropriate models exist for both the signal and noise, then this can be done effectively. (For example, if the power spectrum of the signal and noise are known, then Wiener filtering can be used.) however, it is not straightforward
to devise a general model for images. Most existing image models either describe self-similar or texture-like images or characterize the edge components, but they present a rather restrictive class of images.

The theoretical formalization of noise removal via thresholding wavelet coefficients was pioneered by Donoho and Johnstone[68]. The thresholding method compares the transform coefficients with the threshold and is set to zero if its magnitude is smaller than the threshold; otherwise, it is unchanged or modified (based on the threshold rule). The idea is that coefficients insignificant relative to the threshold are likely due to noise, whereas significant coefficients are important signal structures. Thus thresholding rules are especially effective for signals with sparse or near-sparse representations where only a small subset of the coefficient represents all or most of the signal energy. Thresholding essentially creates a region around zero where the coefficients are considered negligible. Outside of this region, the thresholded coefficients are kept to fully precision.

The problem formulation and proposed denoising method are shown in Fig. 4.2. Say the signal is \( f_{ij}, i, j = 1, \ldots, N \), where \( N \) is an integer power of 2. It is corrupted by additive noise and the result is

\[
g_{ij} = f_{ij} + \varepsilon_{ij} \quad i, j = 1, \ldots, N
\]

Where \( \varepsilon_{ij} \) are not dependent and identically distributed (i.i.d) as normal \( N(0, \sigma^2) \) and is not depends on \( f_{ij} \). The aim is to eliminate the noise
and to evaluate an estimate $\hat{f}_y$ of $f_y$, or to denoise $g_y$. The denoising operation is carried out in the curvelet transform domain of the observed corrupted signal.

Fig. 4.2 Problem formulation and proposed method for denoising. The noisy observation is the signal with additive noise. Noise removal is done in the curvelet transform domain by a soft-thresholding.

While there are many works especially in the statistics literature addressing “How to choose the threshold?”, very few are tailored for images, thus effort has been made to find a threshold more suitable for the proposed framework of image denoising. This approach to finding the threshold is Bayesian, where a priori each details subband of the signal is modeled with the Generalized Gaussians distribution with fixed unknown parameters, also used widely in literature [38,69,70]. Within each subband, the goal to find the threshold which minimizes the mean squared error among soft-threshold estimators. It has been proposed an estimation of the threshold which is near optimal and is easy to compute. It will also be shown that with the chosen prior, the optimal soft-threshold estimator performs better in the mean squared sense than the optimal hard-
threshold estimator, and hence soft-threshold has been used in the denoising algorithm.

After being thresholded, the non-zero coefficients are good estimates of the original signal and thus close to being distributed as Generalized Gaussian.

### 4.4.2 Curvelet Thresholding and Thresholding selection

Let $g = \{g_{ij}\}_{i,j}, f = \{f_{ij}\}_{i,j}, \varepsilon = \{\varepsilon_{ij}\}_{i,j}$, that is, the boldfaced letters will denote the matrix form of the signals which are under consideration. Say $Y = Wg$ denote the matrix of curvelet coefficients of $g$, where $W$ is the 2d curvelet transform operator, and similarly $X = Wf$ and $V = W\varepsilon$.

Define the soft-threshold function to be

$$\eta_T(x) = \text{sgn}(x) \cdot \max(|x| - T, 0),$$

The input to this expression is the argument and shrinks it toward zero by the value $T$, called the threshold. A well known thresholding alternative method is the hard-threshold function,

$$\psi_T(x) = x \cdot 1_{\{|x| > T\}},$$

This doesn’t change the input if it is greater than the threshold $T$; else, input is set to zero. The curvelet thresholding procedure eliminates noise by thresholding small curvelet coefficients while leaving the low resolution coefficients untouched.

Threshold denoising is especially effective for signals with sparse representations in the transform domain. Like the Fourier
transform and wavelet transform, curvelet transform has also good energy compaction properties, so in general, large coefficients correspond to dominant signal features, while small coefficients correspond to small details. When noise is added, the curvelet coefficients are perturbed. If the noise energy is low, then the perturbation is small, and only the very small coefficients should be killed. On the other hand, if the noise energy is high, only very large coefficients should be kept so at least the dominant features are discernible in the recovered signal. Thus the threshold must be chosen to reflect the proportion of the noise energy relative to the signal energy.

In practice, because the hard-thresholding rule tends to yield “blips” in the resultant image especially when the energy of noise is significant, soft thresholding is preferred here since it yields visually more pleasant images even if it tends to smooth out the image slightly more. Furthermore, for the Bayesian prior assumed here, optimal soft thresholding estimators results smaller mean squared errors than optimal hard thresholding estimators, which will be discussed later.

4.4.2.1 Survey of Existing Thresholding Methods

While the idea of thresholding is simple and effective, finding good threshold is not an easy task. There are several major approaches in addressing this issue. For 1d deterministic signal say length N, Donoho and Johnstone [68] developed a universal threshold
\[ T = \sigma \sqrt{2 \ln N} \] which gives in an estimate asymptotically optimal in the minimax manner. Other thresholds include that is derived from minimizing Stein’s unbiased risk [71], and its hybrid version [72]. Thresholds which are dependent on the scale of the wavelet transform were also addressed by [73] and [74-77], with the latter considering correlated noise. Unfortunately, these thresholds are rather counterintuitive for signal processing applications because of their explicit dependency on the sample size \( N \), and they are often found to yield less than satisfactory results for images.

Many works in literature have used parametric modeling for both the signal and noise to determine the best thresholding rule when there is some prior knowledge about the signal. A large class of signals and natural images has been observed to have many small valued wavelet coefficients and relatively few large coefficients due to edges and sharp transitions, resulting in a distribution with a sharp peak at zero and symmetric about zero. Thus, priors such as Laplacian, Generalized Gaussian, and mixture of Gaussians have been used in the Bayesian framework to find the best threshold or shrinkage factor [69,75-79]. The proposed threshold is also estimated from the Bayesian setting, using the Generalized Gaussian distribution under the soft–threshold rule, similar to the frame work in [78,79] for hard–thresholding. For non–parametric methods, a popular approach is to use cross-validation [80-83].
4.4.2.2 Proposed Threshold

With the estimates restricted to the class of soft threshold estimates, \( \hat{X}_y = \eta_f(Y_y) \), the goal is to find thresholds which minimize the mean squared error, \( \frac{1}{N^2} \sum_{i,j} (\hat{X}_{ij} - X_{ij})^2 \). In addition, in each subband the coefficients \( X_{ij} \) can be approximated by a Generalized Gaussian distribution (with unknown parameters), thus

\[
\frac{1}{N^2} \sum_{i,j} (\hat{X}_{ij} - X_{ij})^2 = E(\hat{X} - X)^2 ,
\]

where the summation is taken over the \( N^2 \) coefficients \( X_{ij} \) in one particular sub band, and \( E(.) \) is taken with respect to \( X \) having the Generalized Gaussian distribution. Therefore, it is required to minimize the expected squared error, \( E(\hat{X} - X)^2 \), to find the optimal threshold for each detail sub band.

A justification for the chosen prior is due to the fact that a wide variety of images have decaying spectrums, with most of the signal energy concentrated in the lower frequencies, or, visually, the slow-varying, smooth part of the image. The high frequency energy corresponds to additional details manifested in sharp transitions such as edges or busy textures. Since the detail curvelet coefficient capture the local- varying nature of the signal, this translates to many small coefficient and relatively few large coefficient, resulting in a distribution with a peak at zero and symmetric about zero. Moreover,
the curvelet coefficient in each detail sub band collectively form a histogram which can be well approximated by an analytical distribution such as the Generalized Gaussian distribution

\[ GG_{\alpha, \beta}(x) = C(a, \beta)e^{-\left(x^\alpha|\beta|\right)} \]  \hspace{1cm} (4.1)

Where \( C(a, \beta) = \frac{a\beta}{\Gamma\left(\frac{1}{\alpha}\right)} \) and \( \Gamma(t) = \int_0^\infty e^{-tu^{-1}}du \) is the gamma function.

As mentioned earlier, soft-thresholding is chosen primarily because it yields visually better images especially when noise is substantial. At the end of this section, it will also be shown to have better performance over hard–thresholding for chosen prior. The mean squared error (MSE) and/or PSNR are used for discriminating image qualities.

Consider now only coefficients \( \{X_{ij}\} \) from one particular detail sub band, which collectively have the \( GG_{\alpha, \beta}(x) \) distribution in eq.4.1. Let the parameters \( a \) and \( \beta \) be known for now. The distortion criterion to be minimized is the risk \( r(T) = E_{X}E_{Y|X}(\hat{X} - X)^2 \), where \( \hat{X} = \eta_T(Y) \), and \( Y | X : N(x, \sigma^2) \) and \( X : GG_{\alpha, \beta}(x) \), the optimal threshold \( T^* \) is the argument which minimizes \( r(T) \). Basically there are no closed form solutions for T which minimizes \( r(T) \) for this assumed prior. Hence, numerical calculations are used to find the optimal answer.
Prior to examine the general case, it is better to consider two special cases of the Generalized Gaussian distribution: the Gaussian and the Laplacian distributions.

**Case 4.1 (Gaussian)** For \( \beta = 2 \) and \( \alpha \) parameterized as \( \alpha = 1/(\sqrt{2}\sigma_x) \) (where SD(X)=\( \sigma_x \) is the standard deviation), the Gaussian distribution, \( X \sim N(0,\sigma_x^2) \). It is straightforward to verify that

\[
E_X E_{\gamma|X}(X - \hat{X})^2 = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (\eta_T(y) - x)^2 p(y|x)p(x)dydx
\]

\[
= \sigma^2 w(\frac{\sigma_x^2}{\sigma}, T)
\]

(4.2)

Where

\[
w(\frac{\sigma_x^2}{\sigma}, T) = \sigma_x^2 + 2(T^2 + 1 - \sigma_x^2)\Phi\left(\frac{T}{\sqrt{1 + \sigma_x^2}}\right) - 2T(1 + \sigma_x^2)\phi(T, 1 + \sigma_x^2)
\]

With \( \phi(x, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right) \) and \( \Phi(x) = \int_{-\infty}^{x} \phi(t,1)dt \).

A good approximation of the optimal threshold \( T^* \) is found to be

\[
T^* = \frac{\sigma^2}{\sigma_x}.
\]

(4.3)

Fig. 4.3a compares \( T^* \) and \( \tilde{T}^* \), parameterized by \( \sigma_x \) on the horizontal axis, and \( \sigma = 1 \). Their expected risks are shown in Fig. 4.3b, here the maximum deviation is smaller than 1% of the optimal risk. For a further comparison, the risk for hard-thresholding is also calculated. After some algebra, it can be shown that the risk for hard-thresholding is
\[ r_h(T) = \sigma^2 + (\sigma^2 - \sigma_u^2)(2T\phi(T, \sigma_u^2 + \sigma^2) + 2\Phi\left(\frac{T}{\sqrt{\sigma_u^2 + \sigma^2}}\right) - 1). \] (4.4)

By setting to zero the derivative of eq. 4.4 with corresponding to T, the optimal threshold is determined as

\[
T^*_h = \begin{cases} 
0 & \text{if } \sigma_u > \sigma \\
\infty & \text{if } \sigma_u < \sigma, \\
\text{anything} & \text{if } \sigma_u = \sigma
\end{cases}
\]

With the risk associated

\[
r_h(T^*_h) = \begin{cases} 
\sigma^2 & \text{if } \sigma_u > \sigma \\
\sigma^2 & \text{if } \sigma_u \leq \sigma
\end{cases}
\]

Fig. 4.3b shows that both the optimal and near-optimal soft threshold estimators, \( \eta_T() \) and \( \eta_f() \), and they get lower risks than the optimal hard threshold estimator.

The threshold \( \bar{T} = \sigma^2 / \sigma \) is nearly optimal. For such a choice, the normalized threshold \( \bar{T} / \sigma \) is inversely proportional to \( \sigma_u \), the standard deviation of X, and proportional to \( \sigma \), the noise standard deviation. When \( \sigma / \sigma_u \) is small relative to 1, the signal is much stronger than the noise, thus \( \bar{T} / \sigma \) is chosen to be small in order to preserve most of the signal and remove some of the noise; vice versa, when \( \sigma / \sigma_u \) is much larger than 1, the noise dominates and the normalized threshold is chosen to be large to remove the noise which has overwhelmed the signal. Thus, this threshold choice adapts to both the signal and noise characteristics reflected in the parameters \( \sigma \) and \( \sigma_u \).
Fig. 4.3 Gaussian prior and its Thresholding with $\sigma$. (a) Compares the optimal threshold $T^*(\sigma)$ (denoted as solid line — ) and the threshold $\tilde{T}(\sigma)$ (denoted as dotted line ...) versus the standard
deviation $\sigma_x$ on the horizontal axis. (b) Compares the mean square error (MSE) of optimal soft-thresholding (denoted as $\sigma$), $\tilde{T}$ for soft thresholding (denoted as $\sigma$), and optimal hard thresholding (denoted as $\sigma$).

**Case 4.2** (Laplacian) for $\beta = 1$ and $C(\alpha, \beta) = \alpha / 2$, the Laplacian distribution is $LAP(x) = \frac{\alpha}{\pi} e^{-\alpha|x|}$. Note that the variance of $X$ is $2/a^2$.

Without loss of generality, let $\sigma = 1$. The optimal threshold $T^*$ found by minimizing the risk is drawn against the standard deviation $SD(X) = \sqrt{2}/a$ on the horizontal axis in Fig. 4.4a. The curve corresponding to $T^*$ (denoted in solid line $\sigma$) is compared with the approximate threshold $\tilde{T} = 1 / SD(X) = \alpha / \sqrt{2}$ (denoted in dotted line $\sigma$) in Fig. 4.4a. Their corresponding expected risks are shown in Fig. 4.4b, and the deviation is smaller than 0.8%. This says that the risk at the minimum is not too sensitive to the threshold value.

For a general value of $\sigma$, the parameters $T$ and $\alpha$ are replaced by $T/\sigma$ and $\alpha$, respectively, and the proposed threshold is

$$\tilde{T}(\alpha) = \frac{\sigma^2}{SD(X)} = \frac{\alpha^2}{\sqrt{2}},$$

(4.5)

Which has the same form as the Gaussian case in eq.4.3, but with different parameters.

The threshold choice $2\sigma^2 \alpha$ was found independently in [78] for approximating the optimal hard-thresholding using the same prior. Fig. 4.4 compares the optimal soft-and hard-thresholds and their
approximations, and it shows that the soft-thresholding rule yields a smaller risk for this assumed prior. Generally, for SD(X) greater than approximately 1.3 $\sigma$, the MSE with the approximate hard threshold is worst compared to that of no thresholding was performed (which has a MSE of $\sigma^2$).
Fig. 4.4 Thresholding for the Laplacian prior, with $\sigma = 1$. (a) Compares the optimal soft threshold $T^*(a)$ (—), the approximation $\tilde{T}(a)$ (...), the optimal hard-threshold (---), and its approximation (–∙–∙) versus the SD(x) on the horizontal axis. (b) Their respective MSEs.

**Case 4.3** (Generalized Gaussian) similarly, proposed near optimal threshold is

$$\tilde{T}(a, \beta) = \frac{\sigma^2}{SD(X)} = \sigma^2 \frac{\alpha^2 \Gamma\left(\frac{1}{\beta}\right)}{\Gamma\left(\frac{1}{\beta}\right)}.$$

Let $\sigma = 1$. In Fig. 4.5a, each dotted line (...) is the optimal threshold $T^*(a, \beta)$ for a given fixed $\beta$, plotted against SD(X) on the horizontal axis, with $a$ varying. The proposed threshold $\tilde{T} = 1/SD(X)$ is plotted with the solid line (—). The plot of the optimal thresholds that lies closest to $\tilde{T}$ is the curve for $T^*(a, \beta = 1)$. Fig. 4.5b shows the corresponding risks. The difference in deviation between the optimal risk $r(T^*)$ and $r(\tilde{T})$ grows as $\beta$ moves away from 1, but the error is still within 5% for the curves shown Fig. 4.5b.
Fig. 4.5 Generalized Gaussian prior and its Thresholding with $\sigma = 1$.

(a) Compares the approximation $\tilde{T} = \sigma^2 / SD(X)$ (---) with the optimal threshold for $\beta = 0.6, 1, 2, 3, 4$ (…). The horizontal axis is the standard deviation, $SD(X)$. (b) The optimal MSEs are in (…), and the approximation in (---).

### 4.4.3 Parameter Estimation for Threshold

In the discussion thus far it has been assumed the parameters of the distribution to be known. Now discuss the estimation of these parameters, which in turn yield thresholds $\tilde{T}$ to different subband characteristics.

The first step is to estimate the noise variance, $\sigma^2$. In some practical cases, it is possible to measure $\sigma^2$ based on information other than the corrupted observation. If this is not the case, it is estimated by using the robust median estimator in the highest subband of the curvelet transform, $\hat{\sigma} = Median(|Y|)/.6745$. Also used in [68,72].

The parameters $\alpha$ and $\beta$ in $GG_{\alpha,\beta}$ can be found from the second and the fourth moments of the distribution [69]:

$$m_2 = \int_{-\infty}^{\infty} y^2 p(y)dy \quad \text{and} \quad m_4 = \int_{-\infty}^{\infty} y^4 p(y)dy.$$ 

Recall that model is $T = X + V$, with $X$ and $V$ independent of each other, thus it can be derived that

$$m_2 = \sigma^2 + \frac{\Gamma(\frac{3}{\beta})}{\alpha^{2\Gamma(\frac{1}{\beta})}} \quad \text{and} \quad m_4 = 3\sigma^4 + \frac{6\sigma^2 \Gamma(\frac{5}{\beta})}{\alpha^{2\Gamma(\frac{3}{\beta})}} + \frac{\Gamma(\frac{3}{\beta})}{\alpha^{4\Gamma(\frac{1}{\beta})}} \quad (4.6)$$
The moments are found empirically by

\[ \hat{m}_2 = \frac{1}{\tilde{N}^2} \sum_{i,j} Y_{ij}^2 \quad \text{and} \quad \hat{m}_4 = \frac{1}{\tilde{N}^2} \sum_{y} Y_{y}^4. \]

Where \( \tilde{N}^2 \) is the number of coefficients in the sub band under consideration. The parameters \( \alpha \) and \( \beta \) can be found by solving eq.4.6 with \( \hat{m}_2 \) and \( \hat{m}_4 \) in place of \( m_2 \) and \( m_4 \).

The Generalized Gaussian distribution offers more flexibility in the description of the subband coefficients. However, in practice the Laplacian prior performs well, and also leads to simple closed-form equations. Thus, the Laplacian distribution is assumed for the subband coefficients.

For the Laplacian case, \( \beta = 1 \), and \( m_2 = \sigma^2 + \frac{1}{\alpha^2} \), thus

\[ \hat{\sigma} = \sqrt{\frac{2}{\hat{m}_2 - \sigma^2}}. \]

In the rare case that \( \hat{\sigma} \geq \hat{m}_2 \), the threshold is effectively set to \( \infty \); that is, all coefficients are set to 0.

### 4.5 Proposed Denoising Algorithm

The following steps are involved in the denoising algorithm of curvelet Transform.

1. Corrupt the original image with the noise to obtain noisy image \( \tilde{f} \).
2. Apply the 2D FFT and obtain Fourier samples
\[ \hat{f}[n_1, n_2], -n/2 \leq n_1, n_2 < n/2. \]

3. For each scale \( j \) and angle \( l \), form the product \( \tilde{U}_{j,l}[n_1, n_2] \hat{f}[n_1, n_2] \).

4. Wrap this product around the origin and obtain
\[ \tilde{f}_{j,l}[n_1, n_2] = W(\tilde{U}_{j,l} \hat{f})[n_1, n_2], \] where the range for \( n_1 \) and \( n_2 \) is now
\[ 0 \leq n_1 < L_{1,j} \text{ and } 0 \leq n_2 < L_{2,j} \] (for \( \theta \) in the range \( (-\pi/4, \pi/4) \)).

5. Apply the inverse 2D FFT to each \( \tilde{f}_{j,l} \), hence collecting the discrete
coefficients \( c^D(j,l,k) \).

6. Compute threshold for curvelets.

7. Apply soft thresholding to the curvelet coefficients.

8. Apply inverse curvelet transform to the result of step 7.

Denoising of the images using curvelet transform[84] is carried out with the wrapping function, involving a decomposition level of 8. Since soft thresholding smooths the time series, it has been used. Soft thresholding [64,68,85] is applied to the coefficients after decomposition. Coefficients that are less than the chosen threshold are discarded. The image is reconstructed from the remaining coefficients. Denoising of the images using wavelet transform[64-68] is carried out with symlet4 wavelet, which is an integral part of the wavelet tool box. Three types of noise, viz. Gaussian, Speckle and Salt & Pepper noises, are chosen for mixing with the author and standard Lenna images. For each type of noise, the extent of mixing
corresponded to the standard deviations of 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45 and 0.50.

The quality of reconstructed image is usually specified in terms of peak signal to noise ratio (PSNR). The PSNR values are calculated using eq. 3.7.

4.6 Simulation Results

The Plain (standard Lenna), Building and Textured images are considered and corrupted with Gaussian, Speckle and Salt & Pepper noises.

- **Gaussian noise**: Additive noise with Gaussian distribution is known as Gaussian noise.
- **Speckle noise**: Which is a multiplicative noise. To corrupt the image with this noise, the image is multiplied with random noise and result is added to the original image.
- **Salt & Pepper noise**: Which is a data drop-out noise. The corrupted pixels are set to the maximum value or have single bits flipped over.

The noisy images are denoised using both wavelet and curvelet transforms. Figs. 4.6, 4.8 and 4.10 show original, noisy, wavelet denoised and curvelet denoised images corresponding to Lenna, Building and Textured images in which Gaussian, Speckle and Salt &
Pepper noises are used with standard deviation of 0.50. Figs. 4.7, 4.9 and 4.11 show plots between standard deviation Vs PSNR for Lenna, Building and Textured images corresponding to curvelet and wavelet for the Gaussian, Speckle and Salt & Pepper noises. The PSNR values for the reconstructed images corresponding to the Gaussian, Speckle and Salt & Pepper noise for the standard deviations of 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45 and 0.50 are summarized in Tables 4.1, 4.2 and 4.3 correspondingly for Lenna, Building and Textured images.

4.7 Discussion

From the Tables 4.1, 4.2 and 4.3 it is observed that as the standard deviation increases, the difference in PSNR values of curvelet denoised and wavelet denoised images decreases for Gaussian and Speckle noises irrespective of type of image. Where as in case of Salt & Pepper noise PSNRs of both curvelet and wavelet denoised images increases for low standard deviations and decreases for high standard deviations irrespective of type of images, which is also observed from the plots shown in Figs. 4.7(c), 4.9(c) and 4.11(c). The results presented in Figs. 4.6, 4.8 and 4.10 demonstrate that the curvelet transform out performs the wavelet transform, in reconstructing the images for standard deviation of 0.5.

From the analysis it is clearly observed that reconstruction of original image can be done with lesser coefficients using curvelet.
denoising technique compared with wavelet denoising technique irrespective of type of image in case of Gaussian and Speckle noises, Where as in case of Salt & Pepper noise irrespective of type of image wavelet denoising technique dominates the curvelet denoising technique for low standard deviations and for high standard deviations curvelet denoising technique outperforms the wavelet denoising technique. The Curvelet transform technique is found to be conceptually simpler, faster and far less redundant than the existing technique in case of Gaussian and Speckle noises irrespective of type of images. The Curvelet denoised images appear more closer to the original image than the Wavelet denoised images.
Fig. 4.6 (a) standard Lenna image, (b) noisy image obtained by adding Gaussian noise with standard deviation of 0.5, (c) wavelet denoised image, (d) curvelet denoised image. (e) noisy image obtained by adding speckle noise with standard deviation of 0.5, (f) wavelet denoised image, (g) curvelet denoised image. (h) noisy image obtained by adding
salt & pepper noise with standard deviation of 0.5, (i) wavelet denoised image and (j) curvelet denoised image.
Fig. 4.7 Standard deviation Vs PSNR corresponding to curvelet(DCvT) and wavelet(DWT) for Lenna image corrupted with (a) Gaussian noise, (b) Speckle noise and (c) Salt & Pepper noise.

Table 4.1 Comparison of PSNR w.r.t SD for Lenna image.

<table>
<thead>
<tr>
<th>S.N</th>
<th>Standard deviation (SD)</th>
<th>PSNR in dB (Lenna image)</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Gaussian noise</td>
<td>Speckle noise</td>
<td>Salt &amp; Pepper noise</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Curvelet</td>
<td>Wavelet</td>
<td>Curvelet</td>
<td>Wavelet</td>
</tr>
<tr>
<td>1</td>
<td>0.10</td>
<td>30.0275</td>
<td>27.2492</td>
<td>30.5790</td>
<td>27.5755</td>
</tr>
<tr>
<td>2</td>
<td>0.15</td>
<td>28.0531</td>
<td>25.4315</td>
<td>28.7477</td>
<td>25.7680</td>
</tr>
<tr>
<td>3</td>
<td>0.20</td>
<td>26.4716</td>
<td>24.0655</td>
<td>27.3993</td>
<td>24.6967</td>
</tr>
<tr>
<td>4</td>
<td>0.25</td>
<td>24.8790</td>
<td>22.9508</td>
<td>26.2611</td>
<td>23.7806</td>
</tr>
<tr>
<td>5</td>
<td>0.30</td>
<td>23.5234</td>
<td>21.8729</td>
<td>25.3183</td>
<td>22.9861</td>
</tr>
<tr>
<td>6</td>
<td>0.35</td>
<td>22.3560</td>
<td>20.9779</td>
<td>24.5403</td>
<td>22.4305</td>
</tr>
<tr>
<td>7</td>
<td>0.40</td>
<td>21.1874</td>
<td>20.0284</td>
<td>23.7981</td>
<td>21.7287</td>
</tr>
<tr>
<td>8</td>
<td>0.45</td>
<td>20.3181</td>
<td>19.2313</td>
<td>23.1596</td>
<td>21.1798</td>
</tr>
<tr>
<td>9</td>
<td>0.50</td>
<td>19.5261</td>
<td>18.5315</td>
<td>22.4118</td>
<td>20.7401</td>
</tr>
</tbody>
</table>
Fig. 4.8 (a) Original Building image, (b) noisy image obtained by adding Gaussian noise with standard deviation of 0.5, (c) wavelet denoised image, (d) curvelet denoised image. (e) noisy image obtained by adding speckle noise with standard deviation of 0.5,(f) wavelet denoised image, (g) curvelet denoised image. (h) noisy image obtained by adding salt & pepper noise with standard deviation of 0.5, (i) wavelet denoised image and (j) curvelet denoised image.
Fig. 4.9 Standard deviation Vs PSNR corresponding to curvelet(DCvT) & wavelet(DWT) for Building image corrupted with (a) Gaussian noise, (b) Speckle noise and (c) Salt & Pepper noise.
Table 4.2 Comparison of PSNR w.r.t SD for Building image.

<table>
<thead>
<tr>
<th>S.N</th>
<th>Standard deviation (SD)</th>
<th>PSNR in dB [Building image]</th>
<th>Gaussian noise</th>
<th>Speckle noise</th>
<th>Salt &amp; Pepper noise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Curvelet</td>
<td>Wavelet</td>
<td>Curvelet</td>
<td>Wavelet</td>
</tr>
<tr>
<td>1</td>
<td>0.10</td>
<td>31.5927</td>
<td>30.8180</td>
<td>31.1434</td>
<td>30.4367</td>
</tr>
<tr>
<td>2</td>
<td>0.15</td>
<td>27.8888</td>
<td>27.2494</td>
<td>27.5040</td>
<td>26.8838</td>
</tr>
<tr>
<td>3</td>
<td>0.20</td>
<td>25.2991</td>
<td>24.7170</td>
<td>24.9782</td>
<td>24.4764</td>
</tr>
<tr>
<td>4</td>
<td>0.25</td>
<td>23.1614</td>
<td>22.6598</td>
<td>22.9250</td>
<td>22.4432</td>
</tr>
<tr>
<td>5</td>
<td>0.30</td>
<td>21.3716</td>
<td>20.8594</td>
<td>21.2397</td>
<td>20.7712</td>
</tr>
<tr>
<td>7</td>
<td>0.40</td>
<td>18.7247</td>
<td>18.3175</td>
<td>18.6014</td>
<td>18.2126</td>
</tr>
<tr>
<td>8</td>
<td>0.45</td>
<td>17.6203</td>
<td>17.2780</td>
<td>17.5314</td>
<td>17.2509</td>
</tr>
<tr>
<td>9</td>
<td>0.50</td>
<td>16.7248</td>
<td>16.3974</td>
<td>16.6158</td>
<td>16.2558</td>
</tr>
</tbody>
</table>
Fig. 4.10 (a) Original Textured image, (b) noisy image obtained by adding Gaussian noise with standard deviation of 0.5, (c) wavelet denoised image, (d) curvelet denoised image. (e) noisy image obtained by adding speckle noise with standard deviation of 0.5, (f) wavelet denoised image, (g) curvelet denoised image. (h) noisy image obtained by adding salt & pepper noise with standard deviation of 0.5, (i) wavelet denoised image and (j) curvelet denoised image.
Fig. 4.11 Standard deviation Vs PSNR corresponding to curvelet(DCvT) & wavelet(DWT) for Textured image corrupted with (a) Gaussian noise, (b) Speckle noise and (c) Salt & Pepper noise.

Table 4.3 Comparison of PSNR w.r.t SD for Textured image.

<table>
<thead>
<tr>
<th>S.N</th>
<th>Standard deviation (SD)</th>
<th>PSNR in dB (Textured image)</th>
<th>Gaussian noise</th>
<th>Speckle noise</th>
<th>Salt &amp; Pepper noise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Curvelet</td>
<td>Wavelet</td>
<td>Curvelet</td>
</tr>
<tr>
<td>1</td>
<td>0.10</td>
<td></td>
<td>31.9951</td>
<td>29.1046</td>
<td>33.2211</td>
</tr>
<tr>
<td>2</td>
<td>0.15</td>
<td></td>
<td>29.2632</td>
<td>26.6192</td>
<td>30.7076</td>
</tr>
<tr>
<td>3</td>
<td>0.20</td>
<td></td>
<td>27.0007</td>
<td>24.7083</td>
<td>29.1033</td>
</tr>
<tr>
<td>4</td>
<td>0.25</td>
<td></td>
<td>25.0559</td>
<td>22.9259</td>
<td>27.9189</td>
</tr>
<tr>
<td>5</td>
<td>0.30</td>
<td></td>
<td>23.2726</td>
<td>21.6569</td>
<td>26.9987</td>
</tr>
<tr>
<td>6</td>
<td>0.35</td>
<td></td>
<td>21.7949</td>
<td>20.3206</td>
<td>26.0955</td>
</tr>
<tr>
<td>7</td>
<td>0.40</td>
<td></td>
<td>20.4145</td>
<td>19.1391</td>
<td>25.3873</td>
</tr>
<tr>
<td>8</td>
<td>0.45</td>
<td></td>
<td>19.3450</td>
<td>18.2383</td>
<td>24.6567</td>
</tr>
<tr>
<td>9</td>
<td>0.50</td>
<td></td>
<td>18.3209</td>
<td>17.4916</td>
<td>24.0076</td>
</tr>
</tbody>
</table>