Signals, noise, and information

Fall 2008

Problem Set 5, Part Two

To do these problems you should download the data in `images.mat`. This contains ten 512×512 greyscale images. To pick one image, type

```
X = squeeze(imageArray(1,:,:));
```

and then `X` will be a single 512×512 image. You can look at it by typing `imagesc(X);` you'll need to check on the orientation, so maybe you should try also `imagesc(X');`. Also, this is a grey image, so you should use `colormap gray` to be sure that you don't generate false colors! Hopefully you'll see scenes from the woods.

Problem 1: Normalize the image so that the mean signal in each pixel is zero and the standard deviation is one. Experiment with adding Gaussian random noise to the image, e.g. as

```
Y = X + a*randn(512);
```

where `a` is the amplitude of the noise. As you change the amplitude of the noise and look at the images, what happens? Can you still understand what you are seeing even when the signal–to–noise ratio in each pixel is much smaller than one?

Problem 2: Explore what happens as you try to smooth the image. For example, generate an image in which each pixel intensity is the average of 3×3 pixels in the original image. You could try the same with 5×5 or larger ... Do you notice the loss of detail?

Problem 3: Compute the different between each pixel intensity and the average over its 3×3 neighborhood. Estimate and plot the probability distribution of these differences, being sure to get the right units on the axes; remember that you normalized the original image to have a standard deviation of one. Notice that the distribution is sharply peaked near zero (no difference from the local average), but there is a long tail (large differences). This provides us with a clear example of a strongly non–Gaussian distribution, coming from natural signals.

Problem 4: Take the images to which you added noise and try out your smoothing operations. Convince yourself that you can reduce the 'randomness' in the image, but only at the cost of blurring.

Problem 5: Compare the noisy and real images pixel by pixel. When you see a certain pixel value in the noisy image (say in some small range that you define), what is the average value of the pixel intensity in the real image? Plot this 'average real intensity' vs. the noisy intensity. This provides you with a strategy for removing noise—restoring the original pixel values. What does

this strategy look like if you plot your estimate vs. the input noisy data? How does the strategy change depending on the amplitude of the noise? Is the strategy the same for different images in your set of ten, so that the strategy is relevant to the environment rather than to individual scenes? If you use this strategy, do you improve the look of the noisy images?

Problem 6: This is open ended. Try to invent ways of combining the smoothing operation with the denoising strategy to see if you can do a better job of turning your noisy images back into something that looks like the real images.