ECE 8823a, Spring 2011

Homework #4

Due Wednesday April 13, in class

Suggested Reading:

- Mallat Chapter 11 and Chapter 13
- Candes, "Modern statistical estimation via oracle inequalities," 2006,
- Donoho, "Unconditional bases are optimal bases for data compression and for statistical estimation," 1993,
- Johnstone, "Function estimation and Gaussian sequence models," 2002.

Problems:

1. Using you class notes, prepare a 1-2 paragraph summary of what we talked about in class in the last week. I do not want just a bulleted list of topics, I want you to use complete sentences and establish context (Why is what we have learned relevant? How does it connect with other things you have learned here or in other classes?). The more insight you give, the better.

2. In this problem we will test out the theory of estimation we have developed in class. The file `signal.mat` contains two variables: a length 2048 piecewise polynomial called `f` and its length `n=2048`. The file `daub8_xform.mat` contains a 2048×2048 matrix called `Psi` which implements a Daubechies-8 wavelet transform ($\alpha = \Psi f$ takes a wavelet transform, $f = \Psi^* \alpha$ is the inverse). You will find it useful to load these into your MATLAB workspace (`signal.mat` should be loaded first). Throughout the problem, we will assume that we observe a noisy version of f:

   $$y = f + z,$$

   where the $z(t),\ t = 1, \ldots, n$ are iid Gaussian with standard deviation $\sigma = 0.0066$. In MATLAB, this is done with

   ```
   sigma = 0.0066;
   z = sigma*randn(n,1);
   y = f + z;
   ```

   (a) **Fourier-domain, oracle.** Given y, we wish to estimate f with a diagonal estimator in the discrete Fourier domain. That is, for the estimate by taking the Fourier transform of y, damping the Fourier coefficients by point-wise multiplying with a set of weights λ_k, then inverse Fourier transforming:

   $$\tilde{f} = \sum_k \lambda_k \beta_k \phi_k, \quad \{\beta_k\} = \Phi[y],$$

where Φ is the normalized DFT (so you calculate it using `beta = 1/sqrt(n)*fft(y)` in MATLAB, where `n` is the length of `y`). As we have shown on pages IV.4–5 of the notes, the best possible weights λ_k we could use are

$$\lambda_k = \frac{|\alpha_k|^2}{|\alpha_k|^2 + \sigma^2}, \quad \text{where } \{\alpha_k\} = \Phi[f].$$

Of course, you would need an "oracle" to tell you these weights, as they depend on f. But since we have access to f, we can form them to give us a feel for what the upper limits on performance for Fourier-domain denoising are for this particular signal.

Calculate the risk $r_{\text{inf}}^{\text{four}}(f)$ of this optimal oracle estimator using MATLAB. Verify your calculation by calculating the squared-error of your estimate over 1000 realizations of the noise. Along with your calculations, turn in your code and a plot of an example estimate.

(b) **Fourier-domain, linear.** In practice, we of course cannot calculate the oracle shrinkage weights for the optimal diagonal estimator. Instead we use a set of fixed weights, which are derived from some properties we expect f to have. For instance, we might know that the particular f in `signal.mat` obeys

$$\alpha_k \le c_k := \frac{9}{|k|+1}, \quad k = -n/2+1, \ldots, n/2.$$

(The f given actually only obeys this for $|k| \ge 2$, but this will not concern us too much.)

Calculate, using MATLAB, the risk $r_{\text{lin}}^{\text{four}}(f)$ of the corresponding diagonal estimator using weights

$$\lambda_k = \frac{c_k^2}{c_k^2 + \sigma^2}.$$

Verify your calculation by calculating the squared-error of your estimate over 1000 realizations of the noise. Along with your calculations, turn in your code and a plot of an example estimate.

(c) **Wavelet domain, oracle.** Calculate, using MATLAB, the oracle diagonal risk $r_{\text{inf}}^{\text{wave}}(f)$ in the wavelet domain. Verify your calculation by calculating the squared-error of your estimate over 1000 realizations of the noise. Along with your calculations, turn in your code and a plot of an example estimate.

(d) **Wavelet domain, oracle projection.** Calculate, using MATLAB, the risk $r_{\text{p}}^{\text{wave}}(f)$ of the oracle projection that uses a weight of 1 if $|\alpha_k| \ge \sigma$ and zero otherwise. Verify your calculation by calculating the squared-error of your estimate over 1000 realizations of the noise. Along with your calculations, turn in your code and a plot of an example estimate.

(e) **Wavelet domain, Donoho-Johnstone.** Estimate the wavelet-domain soft-thresholding risk $r_t^{DJ}(f)$ with the Donoho-Johnstone threshold $T = \sigma\sqrt{2\log n}$. Do this by averaging the squared-error of the soft-threshold estimate over 1000 realizations of the noise. Compare to the bound $r_t^{DJ}(f) \le (2\log n + 1) \cdot (\sigma^2 + r_{\text{p}}^{\text{wave}}(f))$. Turn in your code and a plot of an example estimate.

(f) **Wavelet domain, practical thresholding.** In practice, the Donoho-Johnstone thresholds are usually a bit to conservative. Estimate the risk (via Monte Carlo simulations, as in (e)) with threshold $T = 1.7\sigma$. Turn in your code and a plot of an example estimate.

(g) Very briefly summarize what you have learned. Compare the risks to the risk $n\sigma^2$ of the "do nothing" estimator (take the observations y as our estimate). Comment on the relative smoothness of the estimates in the plots you created.

3. In this problem, we will test out the theory of signal restoration (inverse problems) we have developed in class. We will use the same signal and wavelet transform as in Problem 1. The file `operator_1.mat` contains two variables: a length 2048 convolution kernel $k(t)$ and its discrete Fourier transform $\hat{k}(\omega)$. Note that

$$\hat{k}(\omega) = \frac{1}{|\omega| + 1}, \quad \omega = -n/2 + 1, \ldots, n/2.$$

As above, we will define the operator K by

$$Kf = f \circledast k.$$

The SVD $K = U\Lambda V^T$ can be quickly found from the discrete Fourier transform (DFT) coefficients of k. This is discussed in detail in the auxiliary notes titled "The SVD of a Circulant Matrix," and is implemented in the MATLAB file `conv_svd.m`.

With the SVD in hand, recall that the reproducing formula for f is

$$f = \sum_{m=1}^{n} \lambda_m^{-1} \langle u_m, Kf \rangle v_m,$$

where u_m and v_m are the mth columns of U and V, and $\lambda_m = \Lambda_{m,m}$. For the problems below, we observe

$$y = Kf + z,$$

where $z(t)$ is a rather mild noise perturbation: iid Gaussian with standard deviation $\sigma = 10^{-4}$.

Start by plotting an example of the observed signal y.

(a) **Pseudo Inverse.** The crudest way to attempt to recover f is by simply applying the pseudo inverse to the noisy observations:

$$\tilde{f} = \sum_{m} \lambda_m^{-1} \langle u_m, y \rangle v_m, \quad \text{or} \quad \tilde{f} = V\Lambda^{-1}U^T y.$$

Calculate, using MATLAB, the risk of this procedure[1] Plot an example of this estimator in action. (We will skip the verification of the risk calculation for this problem — feel free to verify on your own using Monte Carlo, though.)

[1] Recall that $K^\dagger y = (K^*K)^{-1}K^*y$ is also the solution to the optimization problem

$$\min_{f'} \ \|Kf' - y\|_2^2.$$

(b) **SVD, oracle.** Now we recover f by damping the singular values in the reproducing formula above:

$$\tilde{f} = \sum_m d_m \lambda_m^{-1} \langle u_m, y \rangle v_m, \tag{1}$$

where the d_m are the damping weights. Prove that the optimal choice of weights is

$$d_m = \frac{|\langle v_m, f \rangle|^2}{|\langle v_m, f \rangle|^2 + \sigma^2/\lambda_m^2}.$$

Of course, these are oracle weights, as they rely on knowledge of f. Calculate, using MATLAB, the risk of this estimator. Implement this estimator, and plot an example of it in action (you can just use the same noisy observation as in part (a), if you want to make a direct comparison).

(c) **SVD, linear.** Again, the oracle damping weights will not be available in general. An alternative are the Tikhonov weights

$$d_m = \frac{\lambda_m^2}{\lambda_m^2 + \delta}$$

where δ is given by the user and fixed beforehand[2]. Calculate the risk of this estimator when $\delta = \sigma/4$, implement it, and plot an example of it in action.

(d) **Wavelet-Vaguelette.** An appealing alternative to the SVD for signals such as our f here is the wavelet-vaguelette decomposition. If $\{\psi_{j,m}\}$ is an orthonormal wavelet basis, we define the complementary vaguelette basis $u_{j,m}$ by

$$K^* u_{j,m} = \kappa_j \cdot \psi_{j,m},$$

where the κ_j are chosen so that $\hat{k} \approx \kappa_j$ over the Fourier subband where the $\{\psi_{j,m}\}_m$ are localized. For this particular K, this is achieved with $\kappa_j = 2^j$ (here our coarsest scale is $j = 0$ and our finest scale is at $j = -10$). This gives us the reproducing formula

$$f = \sum_{j,m} \kappa_j^{-1} \langle u_{j,m}, Kf \rangle \psi_{j,m}.$$

The idea is that now the $\kappa_j^{-1}\langle u_{j,m}, y\rangle$ will be something like noisy wavelet coefficients of f, and we can threshold them.

i. For the given wavelet basis $\{\psi_{j,m}\}$, calculate the corresponding vaguelette basis U. Plot the wavelet in row 41 of the matrix `Psi` and the corresponding vaguelette.

ii. Plot the singular values of U, and explain why U might be considered "close" to orthogonal.

iii. We will recover f by thresholding the $\beta_{j,m} = \kappa_j^{-1}\langle u_{j,m}, y\rangle$. The threshold T_j will vary with scale j (since the noise is scaled by κ_j^{-1}, which is different at each scale). Implement a WVD estimator with $T_j = 1.5\kappa_j^{-1}\sigma$. Estimate its risk via Monte Carlo (at least 1000 realizations of the noise). Plot an example of it in action.

[2]The estimate (1) with these weights will be the solution to the optimization problem

$$\min_{f'} \ \|Kf' - y\|_2^2 + \delta \|f'\|_2^2.$$

(e) Very briefly summarize what you have learned.