ECE 901
Lecture 17: Denoising and Spatial Adaptivity

R. Nowak

5/17/2009

1 Introduction

In the previous lecture we saw how wavelets can approximate piecewise Holder smooth functions (recall the definition of $B^\alpha(C)$ in the previous lecture). Let $f \in B^\alpha(C)$ (assume f has m smooth pieces) and define

$$\boldsymbol{f} = \left[f(\frac{1}{n}), \ldots, f(\frac{n}{n}) \right] \ .$$

We have seen that there is a piecewise polynomial function f_p, with pieces of degree $\lfloor \alpha \rfloor$, such that

$$\forall t, \quad |f(t) - f_p(t)| \leq Ck^{-\alpha} \ ,$$

where k controls the quality of the approximation. Therefore

$$\frac{1}{n}\|\boldsymbol{f} - \boldsymbol{f_p}\|_{\ell^2}^2 = \frac{1}{n}\sum_{i=1}^{n}(f(i/n) - f_p(i/n))^2 \leq C^2 k^{-2\alpha} \ .$$

Furthermore $\boldsymbol{f_p}$ can be represented in a wavelet basis. If this basis is constructed using a wavelet transform with at least $\lfloor \alpha \rfloor + 1$ vanishing moments then the representation has at most $O((k+m)\log n) = O(k \log n))$ non-zero coefficients.

2 Denoising

Let $f^* \in B^\alpha(C)$ be the unknown target function. Assume also that $|f^*(t)| \leq M$ for all t. Define

$$\boldsymbol{f}^* = \left[f\left(\frac{1}{n}\right), \ldots, f\left(\frac{n}{n}\right) \right] \ ,$$

$$\boldsymbol{Y} = [Y_1, \ldots, Y_n] \ ,$$

where

$$Y_i = f^*(i/n) + W_i \ ,$$

and $W_i \overset{i.i.d.}{\sim} N(0, \sigma^2)$. Or in vector notation

$$\boldsymbol{Y} = \boldsymbol{f}^* + \boldsymbol{W} \ ,$$

where $\boldsymbol{W} = [W_1, \ldots, W_n]$.

Since we are assuming the Additive White Gaussian Noise (AWGN) model let's consider our familiar maximum penalized likelihood estimator

$$\begin{aligned}\hat{\boldsymbol{f}}_n &= \arg\min_{\boldsymbol{f}\in\mathcal{F}} \left\{ \frac{1}{n}\sum_{i=1}^{n} \frac{(Y_i - f_i)^2}{2\sigma^2} + \frac{2c(\boldsymbol{f})\log 2}{n} \right\} \\ &= \arg\min_{\boldsymbol{f}\in\mathcal{F}} \left\{ \|\boldsymbol{Y} - \boldsymbol{f}\|^2 + 4\sigma^2 c(\boldsymbol{f})\log 2 \right\} \ ,\end{aligned}$$

where we still need to define $\mathcal{F}$ and $c(\boldsymbol{f})$. This is where wavelets come into the picture. Since we know we can approximate $\boldsymbol{f}^*$ with a wavelet basis, it makes sense to "describe" these approximations in that basis (since we can have a simple description, corresponding to a low complexity model).

Let the matrix $A \in \mathbf{R}^{n\times n}$ correspond to an orthogonal discrete wavelet transform (DWT) with at least $\lfloor \alpha \rfloor + 1$ vanishing moments. Note that $A^{-1} = A^T$, that is, the inverse of A its just its transpose. Instead of working with the representation of $\boldsymbol{f}$ in the canonical domain it is preferable to work with the equivalent representation in the wavelet domain $\boldsymbol{\theta} = A\boldsymbol{f}$. We can write our estimator using the wavelet representation instead.

$$\hat{\boldsymbol{\theta}}_n = \arg\min_{\boldsymbol{\theta}\in\Theta} \left\{ \|\boldsymbol{Y} - A^T\boldsymbol{\theta}\|^2 + 4\sigma^2 c(\boldsymbol{\theta})\log 2 \right\} \ ,$$

and

$$\hat{\boldsymbol{f}}_n = A^T\hat{\boldsymbol{\theta}}_n \ .$$

2.1 Encoding the Wavelet Coefficients

Since our oracle bounds only allow us to deal with finite or countable classes of models we need to quantize the possible values for the coefficients, in a similar way we did in lecture 15. Let

$$\Theta = \left\{ \boldsymbol{\theta} \in \mathbf{R}^n : \theta_i \in \{-M, M(-1+2/\sqrt{n}), M(-1+4/\sqrt{n}), \ldots, 0, \ldots, M\} \right\} \ .$$

Therefore each coefficient can take one of possible $\sqrt{n}$ values.

We know that the good candidates for $\boldsymbol{f}^*$ have most of their wavelet coefficients equal to zero, so it makes sense to encode them in a way that reflects this. A simple way of doing it is to encode each non-zero coefficient of $\boldsymbol{\theta}$ separately, in a sequential fashion. Take one bit to indicate if there are any non-zero coefficients left to encode, then use $\log_2 n$ bits to encode the location of that coefficient, and finally $\frac{1}{2}\log_2 n$ bits to encode the magnitude (recall each coefficient can take $\sqrt{n}$ values). This yields a codeword with length

$$\begin{aligned}c(\boldsymbol{\theta}) &= 1 + (1 + \frac{3}{2}\log_2 n) \ \#\text{ non-zero coefficients} \\ &\approx \frac{3}{2}\log_2 n \ \#\{i : \theta_i \neq 0\} \ .\end{aligned}$$

This is a prefix code, and so $\sum_{\boldsymbol{\theta}\in\Theta} 2^{-c(\boldsymbol{\theta})} \leq 1$. Let's see an example for $n = 16$ and $M = 4$. Let

$$\boldsymbol{\theta} = (0,0,0,0,4,0,0,0,0,0,0,0,0,0,-2,0) \ .$$

Let the binary encoding of the possible non-zero amplitudes $\{-4,-2,2,4\}$ be $(00,\ 01,\ 10\,11)$ respectively. Then we encode $\boldsymbol{\theta}$ as

$$\underbrace{1}_{\substack{\text{next is the encoding}\\ \text{of one coef.}}} \quad \underbrace{0101}_{\text{location}} \ \underbrace{11}_{\text{magnitude}} \quad \underbrace{1}_{\substack{\text{next is the encoding}\\ \text{of one coef.}}} \quad \underbrace{1110}_{\text{location}} \ \underbrace{01}_{\text{magnitude}} \quad \underbrace{0}_{\text{we are done}}$$

Therefore

$$c(\boldsymbol{\theta}) = 1 + (1 + \frac{3}{2}\log_2 n) \times \underbrace{2}_{\text{number of coefs.}} = 15 \ .$$

We are now able to apply the Theorem 1 of lecture 14 (in particular the corollary we have shown in example 3 of the same lecture, and use to show the results in lecture 15). This yields the following oracle bound

$$\frac{1}{n}E\left[\|\hat{\boldsymbol{f}}_n - \boldsymbol{f}^*\|^2\right] \leq \min_{\boldsymbol{\theta}\in\Theta}\left\{\frac{2}{n}\|A^T\boldsymbol{\theta} - \boldsymbol{f}^*\|^2 + \frac{8\sigma^2\log 2c(\boldsymbol{\theta})}{n}\right\} . \tag{1}$$

We have seen before that there is a "good" approximation of $\boldsymbol{f}^*$ with wavelet representation $\tilde{\boldsymbol{\theta}} = A\tilde{\boldsymbol{f}}$ such that $\|f_i^* - \tilde{f}_i\| \leq Ck^{-\alpha}$, and $\tilde{\boldsymbol{\theta}}$ has $O(k\log n)$ non-zero entries. Note that in general $\tilde{\boldsymbol{\theta}}$ is not in Θ, but we can construct a quantized version $\bar{\boldsymbol{\theta}}$ such that

$$\|\tilde{\theta}_i - \bar{\theta}_i\| \leq \frac{M}{\sqrt{n}} .$$

Let's plug-in $\bar{\boldsymbol{\theta}}$ in the right-hand-side of the bound (1).

Begin by looking at $\|A^T\bar{\boldsymbol{\theta}} - \boldsymbol{f}^*\|^2$.

$$\begin{aligned}
\|A^T\bar{\boldsymbol{\theta}} - \boldsymbol{f}^*\|^2 &= \|A^T\bar{\boldsymbol{\theta}} - A^T\tilde{\boldsymbol{\theta}} + A^T\tilde{\boldsymbol{\theta}} - \boldsymbol{f}^*\|^2 \\
&= \|A^T\bar{\boldsymbol{\theta}} - A^T\tilde{\boldsymbol{\theta}}\|^2 + \|A^T\tilde{\boldsymbol{\theta}} - \boldsymbol{f}^*\|^2 + 2 < A^T\bar{\boldsymbol{\theta}} - A^T\tilde{\boldsymbol{\theta}}, A^T\tilde{\boldsymbol{\theta}} - \boldsymbol{f}^* > \\
&= \|A^T(\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}})\|^2 + \|\tilde{\boldsymbol{f}} - \boldsymbol{f}^*\|^2 + 2 < A^T(\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}}), \tilde{\boldsymbol{f}} - \boldsymbol{f}^* > \\
&= (\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}})^T AA^T(\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}}) + \|\tilde{\boldsymbol{f}} - \boldsymbol{f}^*\|^2 + 2 < A^T(\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}}), \tilde{\boldsymbol{f}} - \boldsymbol{f}^* > \\
&= \|\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}}\|^2 + \|\tilde{\boldsymbol{f}} - \boldsymbol{f}^*\|^2 + 2 < A^T(\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}}), \tilde{\boldsymbol{f}} - \boldsymbol{f}^* > \\
&\leq \|\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}}\|^2 + \|\tilde{\boldsymbol{f}} - \boldsymbol{f}^*\|^2 + 2\sqrt{\|\bar{\boldsymbol{\theta}} - \tilde{\boldsymbol{\theta}})\|^2\|\tilde{\boldsymbol{f}} - \boldsymbol{f}^*\|^2} \\
&\leq n\frac{M^2}{n} + nC^2k^{-2\alpha} + 2n\sqrt{\frac{M^2}{n}C^2k^{-2\alpha}} ,
\end{aligned}$$

where the first inequality follows from the CauchySchwarz inequality.

Now $c(\bar{\boldsymbol{\theta}}) = \frac{3}{2}\log_2 n)O(k\log n) = O(k\log^2 n)$, therefore plugging all this in the bound yields

$$\frac{1}{n}E\left[\|\hat{\boldsymbol{f}}_n - \boldsymbol{f}^*\|^2\right] = O\left(\max\left\{\frac{1}{n}, k^{-2\alpha}, \frac{k^{-\alpha}}{\sqrt{n}}, \frac{k\log^2 n}{n}\right\}\right) .$$

Choosing $k = \lfloor n^{\frac{1}{2\alpha+1}} \rfloor$ yields

$$\frac{1}{n}E\left[\|\hat{\boldsymbol{f}}_n - \boldsymbol{f}^*\|^2\right] = O\left(n^{-\frac{2\alpha}{2\alpha+1}}\log^2 n\right) ,$$

which is almost as good as if we knew the smoothness and the location of the various pieces (the $\log^2 n$ is the price we pay for this extra adaptivity).

3 Computational Remarks

Although having the theoretical guarantees for our estimator is quite interesting, it will only be useful if it can be computed easily. It turns out this is indeed the case, and this estimation strategy is a very intuitive thing to do. Let $\boldsymbol{Z} = A\boldsymbol{Y}$ (so that $\boldsymbol{Y} = A^T\boldsymbol{Z}$) and re-write our estimator.

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_n &= \arg\min_{\boldsymbol{\theta}\in\Theta}\left\{\|A^T\boldsymbol{Y} - A^T\boldsymbol{\theta}\|^2 + 4\sigma^2 c(\boldsymbol{\theta})\log 2\right\} \\
&= \arg\min_{\boldsymbol{\theta}\in\Theta}\left\{\|Y - \boldsymbol{\theta}\|^2 + 4\sigma^2 c(\boldsymbol{\theta})\log 2\right\} \\
&= \arg\min_{\boldsymbol{\theta}\in\Theta}\left\{\sum_{i=1}^n (Y_i - \theta_i)^2 + 4\sigma^2\log 2\frac{3}{2}\log_2 n\sum_{i=1}^n \mathbf{1}\{\theta_i \neq 0\}\right\} \\
&= \arg\min_{\theta_1,\ldots,\theta_n}\left\{\sum_{i=1}^n \underbrace{(Y_i - \theta_i)^2 + 6\sigma^2\log 2\log_2 n\mathbf{1}\{\theta_i \neq 0\}}_{U_i}\right\} .
\end{aligned}$$

So the computation of the estimator boils down to the solution of n one-dimensional optimization problems $\hat{\theta}_i = \arg\min_{\theta_i \in \mathbf{R}} U_i$. It is quite easy to see that the solution is simply

$$\hat{\theta}_i = \begin{cases} Z_i & \text{if } |Z_i| \geq \sqrt{6\sigma^2 \log_2 \log_2 n} \\ 0 & \text{otherwise} \end{cases} \equiv h(Z_i) \ .$$

This is what is called a *hard-thresholding*. So the roadmap for the computation of the estimator is: (i) compute $\boldsymbol{Z} = A\boldsymbol{Y}$; (ii) Threshold, that is compute $\hat{\boldsymbol{\theta}} = h(\boldsymbol{Z})$; (iii) Compute the final solution $\hat{\boldsymbol{f}}_n = A^T\hat{\boldsymbol{\theta}}$. Since the wavelet transform can be done efficiently in $O(n)$ operations the overall complexity of the above algorithm is $O(n)$, which is the best we can hope for!

Why does hard-thresholding work so well? Let's look at

$$\boldsymbol{Z} = A\boldsymbol{Y} = A\boldsymbol{f}^* + A\boldsymbol{W} = \boldsymbol{\theta}^* + A\boldsymbol{W} \ .$$

Now $A\boldsymbol{W}$ is a linear transformation of a multivariate Gaussian random vector, therefore it is also a multivariate Gaussian random vector, with mean $E[A\boldsymbol{W}]$ and covariance $\mathrm{Cov}(A\boldsymbol{W})$. Clearly

$$E[A\boldsymbol{W}] = AE[\boldsymbol{W}] = \boldsymbol{0} \ .$$

and

$$\mathrm{Cov}(A\boldsymbol{W}) = E[(A\boldsymbol{W})(A\boldsymbol{W})^T] = AE[\boldsymbol{W}\boldsymbol{W})^T]A^T = A\sigma^2 I A^T = \sigma^2 I = \mathrm{Cov}(\boldsymbol{W}) \ ,$$

so the statistics of the noise in the wavelet representation are exactly the same as in the canonical representation, which implies the noise is spread out over all the coefficients in a uniform like way. Unlike the noise, the signal is concentrated in a few basis vectors, so that the corresponding wavelet coefficients have large values. Therefore thresholding $\boldsymbol{Z}$ leaves behind a lot of the noise contribution, without affecting much the signal reconstruction.