

Wavelet deconvolution in a periodic setting using cross-validation

Leming Qu*, Partha S. Routh, and Kyungduk Ko

Abstract—Wavelet deconvolution method WaveD using band-limited wavelets offer both theoretical and computational advantage over traditional compactly-supported wavelets. The translation invariant WaveD with fast algorithm improves further. The twofold cross-validation method for choosing the threshold parameter and the finest resolution level in WaveD is introduced. The algorithm’s performance is compared with the fixed constant tuning and the default tuning in WaveD.

Index Terms—Wavelet deconvolution, cross-validation.

I. INTRODUCTION

Recent work by Johnstone et al. [4] has introduced WaveD: the method of wavelet deconvolution in a periodic setting. WaveD uses band-limited wavelets which offer both theoretical and computational advantage over traditional compactly-supported wavelets. The translation invariant version of WaveD [2] improves the performance of ordinary WaveD by cycle-spinning over all circulant shifts. The fast algorithm which implements the translation invariant version of WaveD takes full advantage of the Fast Fourier Transform (FFT) and runs in $O(n(\log(n))^2)$ steps only. The excellent asymptotic results and fast algorithm make WaveD a very attractive noniterative deconvolution technique. We refer WaveD as the translation invariant version of WaveD below if we do not explicitly distinguish them.

This letter introduces twofold cross-validation method for choosing the threshold parameter and the finest resolution level in WaveD. Section II reviews the WaveD method. The cross-validation algorithm is introduced in section III. Section IV illustrates the algorithm using simulation. All the discussion in this letter concerns signals in one dimension but can be extended to higher dimensions naturally. See [3] for WaveD image deblurring, a two dimensional case.

II. WAVED: WAVELET DECONVOLUTION IN WHITE NOISE

In the periodic setting and discrete data the deconvolution can be stated as follows: suppose we observe

$$y_i = f \star g(t_i) + \sigma n^{-1/2} z_i, \quad i = 1, 2, \dots, n, \quad (1)$$

where $t_i = i/n \in T = [0, 1]$, σ is a positive constant, z_i ’s are i.i.d. white Gaussian noise and

$$f \star g(t) = \int_T f(t-u)g(u)du. \quad (2)$$

L. Qu and K. Ko are with the Department of Mathematics, P.S. Routh is with the Department of Geosciences, Boise State University, Boise ID 83725, (email:qu@math.boisestate.edu; routh@cgiss.boisestate.edu; ko@math.boisestate.edu).

The goal is to recover the unknown function f from the noisy, blurred observations $y = (y_1, \dots, y_n)^T$ given the known function g . It is further assumed that the function $f \in L^2(T)$ is periodic on the unit interval T and g has a certain degree of smoothness quantified by the decay of its Fourier coefficients G_l . For ordinary smooth convolution, $|G_l| \sim |l|^{-\nu}$ and the decay parameter ν is referred as the Degree of Ill Posedness (DIP) of the deconvolution problem (1) according to [4]. (The notation $a_k \sim b_k$ means that $\lim_{k \rightarrow \infty} (a_k/b_k) = C$ for a constant C .)

The ordinary WaveD estimator of f , based on hard thresholding is

$$\tilde{f} = \sum_{\kappa \in I_0} \tilde{\alpha}_\kappa I\{|\tilde{\alpha}_\kappa| \geq \lambda_{j_0}\} \Phi_\kappa + \sum_{\kappa \in I_1} \tilde{\beta}_\kappa I\{|\tilde{\beta}_\kappa| \geq \lambda_j\} \Psi_\kappa.$$

The $I\{\cdot\}$ is the usual indicator function. The $\tilde{\alpha}_\kappa$ and $\tilde{\beta}_\kappa$ are estimated wavelet coefficients of the true wavelet coefficients α_κ and β_κ of f , respectively. $I_0 = \{(j_0, k) : k = 0, 1, \dots, 2^{j_0} - 1\}$ is the set of indices corresponding to a coarse resolution level j_0 and the set of indices $I_1 = \{(j, k) : k = 0, 1, \dots, 2^j - 1, j_0 \leq j \leq j_1\}$ details up to a fine resolution level j_1 . The λ_j is the threshold for the estimated wavelet coefficients at the j th resolution level. The chosen scaling function Φ and wavelet function Ψ are band-limited. In particular, the periodic Meyer wavelet basis is used. The algorithm implementing the discrete Meyer wavelet transform is different from the Pyramid algorithm of the usual discrete wavelet transform which uses the compactly supported wavelet basis [5]. The Kolaczyck’s algorithm for discrete Meyer wavelet transform operates on the Fourier coefficients of both the data and the Meyer wavelet and takes $O(n(\log(n))^2)$ steps.

With a slight abuse of notation, mainly in order to be consistent with the notation in [4], denote Y_l and Ψ_l^κ as the Fourier coefficients of y and Ψ_κ respectively. Denote G_l as the Fourier coefficients of g . Denote $C_j = \{l : \Psi_l^\kappa \neq 0\}$ and its cardinality as $|C_j|$. From the compact support of the Fourier transform of Meyer wavelet, we have

$$C_j \subset (2\pi/3)[-2^{j+2}, -2^j] \cup [2^j, 2^{j+2}].$$

With suitably chosen j_0, j_1 and $\eta > 0$, the main steps of the ordinary WaveD are:

- (a) Compute Fourier coefficients Y_l, G_l and compute $\hat{\beta}_\kappa = \sum_l \frac{Y_l}{G_l} \bar{\Psi}_l^\kappa$ by using Kolaczyck’s algorithm, where $\bar{\Psi}_l^\kappa$ is the complex conjugate of Ψ_l^κ .
- (b) Set the thresholds

$$\lambda_j = \eta \sigma \tau_j \sqrt{(\log(n)/n)}.$$

where

$$\tau_j = (|C_j|^{-1} \sum_{l \in C_j} |G_l|^{-2})^{1/2}$$

- (c) Apply level dependent thresholding on $\hat{\beta}_\kappa$. Finally, apply the inverse discrete Meyer wavelet transform on the thresholded wavelet coefficients estimate to obtain \tilde{f} , an estimate of f .

Essentially, the above thresholding algorithm is based on the following idealized independent normal model:

$$\tilde{\beta}_\kappa = \beta_\kappa + \sigma n^{-1/2} 2\sqrt{\pi} \tau_j z_\kappa, \quad (3)$$

where z_κ are white Gaussian noise. Then the level dependent universal threshold [1] is

$$\lambda_j^{UV} = \eta \sigma \tau_j \sqrt{(\log(n)/n)} \quad (4)$$

for $\eta = \sqrt{8\pi}$.

Note that $\sigma^2 4\pi \tau_j^2/n$ is an upper bound of the actual

$$\text{var}(\hat{\beta}_\kappa) = \sigma^2 n^{-1} 2^{-j} \sum_l \left| \frac{\hat{\psi}(2^{-j} \cdot 2\pi l)}{G_l} \right|_2^2$$

for the argument of the asymptotic theory in equation (48) of [4]. The

$$\sigma_j = (2^{-j} \sum_l \left| \frac{\hat{\psi}(2^{-j} \cdot 2\pi l)}{G_l} \right|_2^2)^{1/2} \quad (5)$$

is usually smaller than $2\sqrt{\pi} \tau_j$.

The exact model for unknown parameters β_κ is:

$$\hat{\beta}_\kappa \sim N(\beta_\kappa, \sigma^2 n^{-1} \sigma_j^2).$$

The $\hat{\beta}_\kappa$'s are correlated with

$$\text{cov}(\hat{\beta}_{\kappa_1}, \hat{\beta}_{\kappa_2}) = \sigma^2 n^{-1} \sum_l \frac{\bar{\Psi}_l^{\kappa_1} \Psi_l^{\kappa_2}}{G_l^2}.$$

The WaveD does not utilize any covariance information. Attempt to use these covariance information necessitates an iterative deconvolution procedure which is computationally intensive. The WaveD is a noniterative technique.

The algorithm for translation invariant WaveD fully exploits the periodicity of f and Ψ_κ . It computes the translation invariant estimate of f over all circulant shifts with complexity only $O(n(\log(n))^2)$ without actually going through n individual ‘‘cycle-spin’’ steps. The brute-force ‘‘cycle-spin’’ would apparently cost $O((n \log(n))^2)$. This fast algorithm of translation invariant WaveD is very attractive. The main steps of the translation invariant WaveD are:

- (a) Compute Fourier coefficients Y_l , G_l and perform deconvolution

$$\tilde{F}_l = Y_l/G_l;$$

where \tilde{F}_l is the estimated Fourier coefficients of f .

- (b) Set the thresholds λ_j ;
 (c) Loop resolution level j from j_0 to j_1 to compute \tilde{f}_j^H , the detail estimate of f at resolution level j . The \tilde{f}_j^H is computed by a sequence of operations including convolution, inverse Fourier transform, thresholding, Fourier transform. See section 6 of [2] for detail.

The performance of WaveD depends on the the tuning parameters j_0, j_1 and λ_j 's. The coarse scale j_0 has the default value 3 and is not as influential as the other tuning parameters. For j_1 , the asymptotic theory suggests that

$$2^{j_1} \sim (n/\log(n))^{1/(1+2\nu)}.$$

For example, when $\nu = 1, n = 2048$, we have $j_1 = 3$ by the above guidance. Apparently, this can not be used for finite sample. In the WaveD software, the j_1 is set to be the level proceeding $j(100\%)$ where $j(100\%)$ is the smallest level where 100% of thresholding occurs. This leaves the choice of λ_j 's more important.

In the direct data case ($y_i = f(t_i) + \sigma n^{-1/2} z_i$), there is no need to choose j_1 since it is always set to be $J - 1$, where $J = \log_2(n)$. That is, all the empirical wavelet coefficients are used in the thresholding process for denoising problem. This is because the variance of wavelet coefficients is considered to be constant across level j in the direct data case, so that a few coefficients which bear significant signal information is distinguishable from the rest which are pure noise. Note that the true wavelet coefficients β_κ of a wide class such as Besov space signals decay exponentially fast with increasing resolution level j . With indirect data as in model (1), it often occurs that the variance of empirical wavelet coefficients $\tilde{\beta}_\kappa$ increases with j as seen in (5), so that signal can hardly be separated from noise in those fine levels. Consequently, all the wavelet coefficients in those fine levels have to be discarded.

The λ_j 's depend on η . Asymptotic theory suggests that η should be large, but for finite sample size smaller η maybe desirable. Default value in WaveD is set to be $\sqrt{2}$. In the simulation study for Boxcar convolution in [2], η was set to 0.35.

The difficulty facing the choice of j_1 and η deems a data adaptive approach necessary. We propose a cross-validation (CV) based approach in the next section.

III. CROSS-VALIDATION FOR WAVED

The aim of deconvolution is the minimization of the mean integrated square error (MISE)

$$MISE = E\|\hat{f} - f\|_2^2.$$

The \hat{f} , hence the *MISE*, depends on the tuning parameters. In practice the f is unknown, so a tool which mimics *MISE* has to be devised. CV is such a tool widely used to choose a tuning parameter in many statistical settings. The classic leave-one-out CV is performed by systematically deleting one observation from the construction of an estimate, then comparing the observed value to the predicted value at the deleted point. This simple leave-one-out procedure cannot be directly applied to wavelet deconvolution because the WaveD algorithm [4] [2] only operates on data sets of size a power of 2.

In the direct data case, [6] introduced two-fold CV to choose a threshold for wavelet shrinkage estimate. The basic idea is to remove all the odd-indexed observations first, then use the remaining even-indexed observations to get the wavelet estimates of the even-indexed function values, then compare

the odd-indexed observations to the predicted values at odd-indexed points by linearly interpolating the even-indexed function estimates. The same procedure is done for all the even-indexed observations. The CV score function $CVS(\cdot)$ compares the interpolated wavelet estimates with the left-out observations to form an estimate of prediction error at a particular threshold. The $CVS(\cdot)$ is then numerically minimized over values of the threshold. This two-fold CV can be extended to the deconvolution setting.

The detailed steps forming the $CVS(\eta, j_1)$ is discussed below.

For the given $s = (\eta, j_1)$, from the given data $d_i = (t_i, g_i, y_i)$, $i = 1, \dots, n$ where $n = 2^J$ for an integer J , remove all the odd-indexed d_i 's from the set. This leaves $n/2$ evenly indexed d_i which are reindexed from $m = 1, \dots, n/2$. A function estimate \tilde{f}_s^E is then constructed from the re-indexed d_m by WaveD. The linear interpolation is used to predict the f_{2i-1} by

$$\bar{f}_{s,m}^E = \frac{1}{2}(\tilde{f}_{s,m}^E + \tilde{f}_{s,m+1}^E), \quad m = 1, \dots, n/2,$$

setting $\tilde{f}_{s,n/2+1}^E = \tilde{f}_{s,1}^E$ because f is assumed to be periodic. Then the left-out odd-indexed observation is predicted by

$$\hat{y}_{s,m}^E = \bar{f}_s^E \star g(t_{2m-1}), \quad m = 1, \dots, n/2.$$

The \tilde{f}_s^O is computed for the odd indexed points by WaveD and the interpolant \bar{f}_s^O , the predicted even indexed observation \hat{y}_s^O computed as above. The CV score function compares the predicted with the observed values:

$$CVS(\eta, j_1) = \sum_{m=1}^{n/2} \{(y_{2m-1} - \hat{y}_{s,m}^E)^2 + (y_{2m} - \hat{y}_{s,m}^O)^2\}$$

The computational complexity for getting $CVS(\eta, j_1)$ is still $O(n(\log(n))^2)$. The matlab code forming the $CVS(\eta, j_1)$ is given in the Appendix.

We seek the η, j_1 which minimize the $CVS(\eta, j_1)$, that is:

$$(\eta^{CV}, j_1^{CV}) = \underset{\eta > 0, j_0 < j_1 < J}{\operatorname{argmin}} CVS(\eta, j_1).$$

The simplest way to solve this optimization problem is to solve the minimization problem for each possible value of j_1 first: for $j_1 = j_0 + 1$ to $J - 1$, solve

$$\eta_{j_1}^{CV} = \underset{\eta > 0}{\operatorname{argmin}} CVS(\eta, j_1),$$

then select

$$j_1^{CV} = \underset{j_0 < j_1 < J}{\operatorname{argmin}} CVS(\eta_{j_1}^{CV}, j_1).$$

and set $\eta^{CV} = \eta_{j_1^{CV}}^{CV}$.

Empirical experiments show that this j_1^{CV} sometimes overestimate the j_1 . So we choose the final j_1 as the smaller value of this j_1^{CV} and $j_1^{Default}$. With $\eta = \eta^{CV}$, the default choice for j_1 , $j_1^{Default}$, is determined from the data as follows: it is set to be the level proceeding the smallest level where 100% of thresholding occurs in ordinary WaveD [4].

IV. SIMULATION RESULTS

We present some simulation results to compare the WaveD with fixed η and j_1 , with fixed η but default j_1 and the CV tuning based on four artificial signals borrowed from the statistical wavelet literature [2]. Each of the four test signals: (a)Lidar, (b)Bumps, (c)HeaviSine and (d)Doppler depicted on Figure 1 exhibits some inhomogeneous behavior.

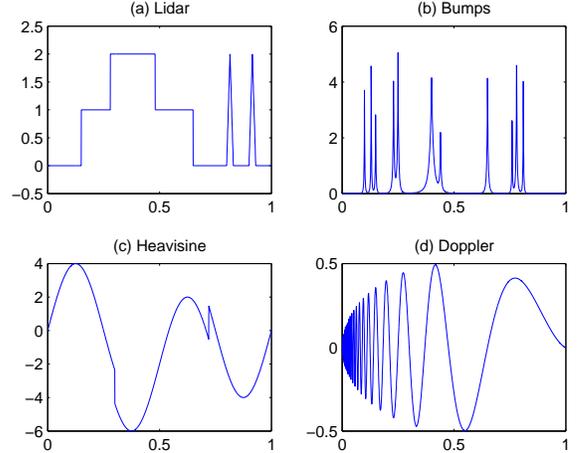


Fig. 1. Four (inhomogeneous) signals, $t_i = i/n$, $n = 2048$.

In Figure 2, a Gamma blur (with $g(t)$ being the probability density function of the $\Gamma(1, 0.0065)$ distribution) is applied to the signals in Figure 1. Such filter with DIP=1 is often referred as smooth convolution in the statistical literature since its Fourier coefficients G_l decay homogeneously [4].

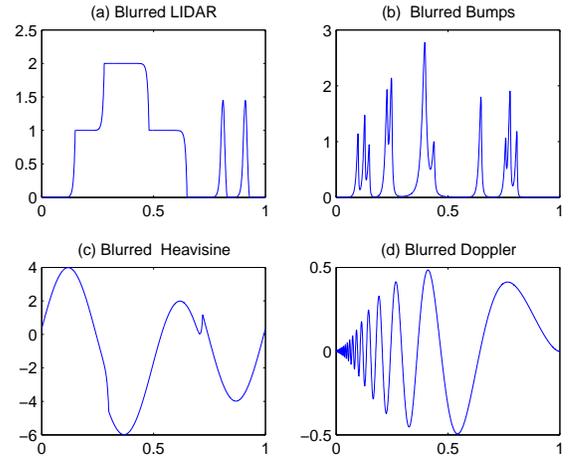


Fig. 2. Smooth blurred (DIP=1) signals of Figure 1

With fixed η , the default choice for j_1 is determined from the data by setting it as the level proceeding the smallest level where 100% of thresholding occurs in ordinary WaveD [4].

The results presented in Table I are based on 1000 independent simulations with $n = 2048$ for Gamma blur with medium noise level $\sigma = 0.5$. The corresponding blurred signal-to-noise ration ($BSNR = 20 \log_{10}(\|f \star g - \mu(f \star g)\|_2 / \sigma)$, where $\mu(f \star g)$ denotes the mean of the blurred signal $f \star g$ samples.) is 36.70, 33.18, 48.55 and 27.88 respectively for Lidar, Bumps, HeaviSine and Doppler signals.

TABLE I

MONTE CARLO APPROXIMATIONS TO $RMISE = \sqrt{E\|\hat{f} - f\|_2^2}$.
SMOOTH BLUR DIP=1

Tuning	Thresh	Lidar	Bumps	HeaviSine	Doppler
$j_1 = 6, \eta = \sqrt{2}$	Hard	0.1374	0.2983	0.1277	0.0920
$j_1 = 6, \eta = \sqrt{2}$	Soft	0.1844	0.3807	0.1603	0.1126
$j_1 = 5, \eta = \sqrt{6}$	Hard	0.1526	0.4255	0.1389	0.0982
j_1 default, $\eta = \sqrt{2}$	Hard	0.2158	1.0621	0.2052	0.1320
CV	Hard	0.1417	0.3039	0.1349	0.1035

The results presented in Table II are based on 1000 independent simulations with $n = 2048$ for Boxcar blur with medium noise level $\sigma = 0.5$. The corresponding BSNR is 36.27, 30.39, 48.41 and 26.92 respectively for Lidar, Bumps, HeaviSine and Doppler signals. For Boxcar blur, $g(t) = (1/2a)I_{[-a,a]}(t)$ with $a = 1/\sqrt{353}$, DIP=1.

TABLE II

MONTE CARLO APPROXIMATIONS TO $RMISE = \sqrt{E\|\hat{f} - f\|_2^2}$.
BOXCAR BLUR DIP=1.5

Tuning	Thresh	Lidar	Bumps	HeaviSine	Doppler
$j_1 = 4, \eta = 0.35$	Hard	0.2435	0.5020	0.1917	0.1313
$j_1 = 4, \eta = 0.35$	Soft	0.2705	0.5240	0.2149	0.1395
$j_1 = 4, \eta = \sqrt{6}$	Hard	0.3007	0.5417	0.2295	0.1636
j_1 default, $\eta = \sqrt{2}$	Hard	0.4215	0.9521	0.3834	0.3377
CV	Hard	0.2729	0.5308	0.2562	0.1823

In both tables, the best performance is obtained with fixed tuning parameters in the first row as used in the simulation study of [2]. Soft thresholding rule gave poorer results than the hard thresholding whether using fixed tuning parameter or CV based tuning parameter. The default value $\eta = \sqrt{2}$, $j_1 = j_1^{Default}$ perform worst. It tends to select the j_1 larger than the one in the first row. CV tuning is close to best performance. Similar patterns were observed when using small $\sigma = 0.01$ and large $\sigma = 1$, also when using $n = 1024$.

The best performance obtained in the Table I and II with fixed η and j_1 is not practically obtainable because it is not clear how the η and j_1 are chosen. The CV tuning provides an operational tool which mimics the best.

V. CONCLUSION

This letter has introduced twofold cross-validation to the wavelet deconvolution in a periodic setting. Simulation results show that this twofold cross validation is adept at selecting a threshold and the finest resolution level. The cross-validation method extends to image deconvolution in a straightforward manner.

In [6], a leave-one-out cross-validation algorithm has also been devised for denosing problem which works for data sets with any sample size. It is an interesting research topic to see if this more computationally intensive cross-validation can bring benefits in the wavelet deconvolution context.

APPENDIX

MATLAB SOURCE CODE FORMING THE TWO FOLD CROSS VALIDATION SCORE FUNCTION

```
function y=CVS(eta,yobs,g,L,deg,F);
```

```
%Two fold cross validation score function
%Inputs (required):
% eta: the smoothing parameter
% yobs =f*g+Noise
% g: Sample of the (known) function g
% Inputs (optional):
% L Lowest resolution level (default=3)
% deg deg of the Meyer Wavelet (default=3)
% F Finest resolution level (default=J-1)
% Outputs
% y=Two fold cross validation score function
n=length(yobs);
J=log2(n);

if nargin<6, F=J-1; end
if nargin<5, deg=3; end
if nargin<4, L=3; end

index_even=2*(1:1:n/2);
yobs_even=yobs(index_even);
g_even=g(index_even);

index_odd=index_even-1;
yobs_odd=yobs(index_odd);
g_odd=g(index_odd);

f_estimate_even=TIwaveD1(yobs_even,g_even,L,deg,F,eta);
f_predict_odd=interp1(index_even, f_estimate_even,...
    index_odd(2:end), 'linear');
f_predict_odd=[0.5*(f_estimate_even(end)+...
    f_estimate_even(1)), f_predict_odd];
ypredict_odd=real(ifft(fft(g_odd).*...
    fft(f_predict_odd)));

f_estimate_odd=TIwaveD1(yobs_odd,g_odd,L,deg,F,eta);
f_predict_even=interp1(index_odd, f_estimate_odd,...
    index_even(1:(end-1)), 'linear');
f_predict_even=[f_predict_even,0.5*...
    (f_estimate_odd(end)+f_estimate_odd(1))];
ypredict_even=real(ifft(fft(g_even).*...
    fft(f_predict_even)));

y=errorLp(yobs_odd,ypredict_odd,2)+...
    errorLp(yobs_even,ypredict_even,2);
```

ACKNOWLEDGMENTS

The authors would like to thank the associate editor and two anonymous reviewers for their constructive remarks.

REFERENCES

- [1] D.L. Donoho and I.M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol.81, pp.425-455, 1994.
- [2] D.L. Donoho and M. Raimondo, "Translation invariant deconvolution in a periodic setting," *The International Journal of Wavelets, Multiresolution and Information Processing*, vol.14, no.4, pp.415-432, 2004.
- [3] D.L. Donoho and M. Raimondo, M., "A Fast wavelet algorithm for image deblurring," *Australian and New Zealand Industrial and Applied Mathematics Journal*, vol.46, pp.C29-C46,2005.
- [4] I.M. Johnstone, G. Kerkycharian, D. Picard, and M. Raimondo, "Wavelet deconvolution in a periodic setting," *Journal of the Royal Statistical Society, Series B*, vol.66, no.3, pp.547-573, 2004. With discussion , pp.627-657.
- [5] E. Kolaczyk, "Wavelet methods for the inversion of certain homogeneous linear operators in the presence of noisy data," PhD dissertation. Department of Statistics, Stanford University, Stanford, 1994.
- [6] G.P. Nason, "Wavelet shrinkage using cross-validation," *Journal of the Royal Statistical Society, B.*, vol.58, pp.463-479, 1996.