

Directional step models for multiscale correlation thresholding.

Eric Chicken
Department of Statistics
Florida State University
Tallahassee, FL 32306

Abstract

Multiple simulations are performed to study the efficiency of forward, backward, and forward-backward wavelet thresholding techniques for signal separation. Various simulated functions are examined, as well as real data. The results are compared via number of coefficients selected, errors, and increase in correlation.

1 Introduction

The wavelet based thresholding techniques described in Chicken et al. (2006) are examined in detail in this report. Forward, backward, and forward-backward step algorithms for wavelet coefficient selection are applied to simulated and real data.

The real data is the tide and discharge data discussed extensively in Chicken et al. (2006). The simulated data varies several parameters of interest. In the model

$$Y_i = f(t_i) + \varepsilon_i = g(t_i) + h(t_i) + \varepsilon_i$$

for $i = 1, 2, \dots, n$, h is linearly related to h' , and $\int h = 0$. We are interested in different levels of the noise $\varepsilon \sim \text{normal}(0, \sigma^2)$. Noise was also added to the external function h' . The noise changes the relation between h and h' from linearly related to (highly) correlated. The level of the noise is set so that the signal to noise ratio (snr) of the function $f = g + h$ or h' is an integer value from 2 to 8:

$$snr = \frac{\text{var}(f)}{\sigma^2} = \frac{\text{var}(h')}{\sigma^2}.$$

The functions g and h were created to meet the requirements specified in Chicken et al. (2006). There are three cases considered. First is the case where g and h reside in different resolution levels of the multiresolution analysis (MRA) of the DWT. This was accomplished by taking the DWT of each function, setting the coefficients of some levels to 0 so that any resolution level that had non-zero coefficients for g was all zeros for h , and vice versa. Then, these modified coefficients were reconstructed with the IDWT to get g and h . The function used for g is the “corner” function, and the “wave” function

was used for h . Figure 1 shows “corner” and “wave”, Figure 2 shows g and h . For g , the highest 6 resolution levels were set to 0, for h , the remaining levels were zeroed. Note that noise was added to h' (linearly related to h) and $f = g + h$ after the functions g and h were determined through modification of their coefficients. The wavelet used on this method was $la8$ in the `waveslim` package of the software program R.

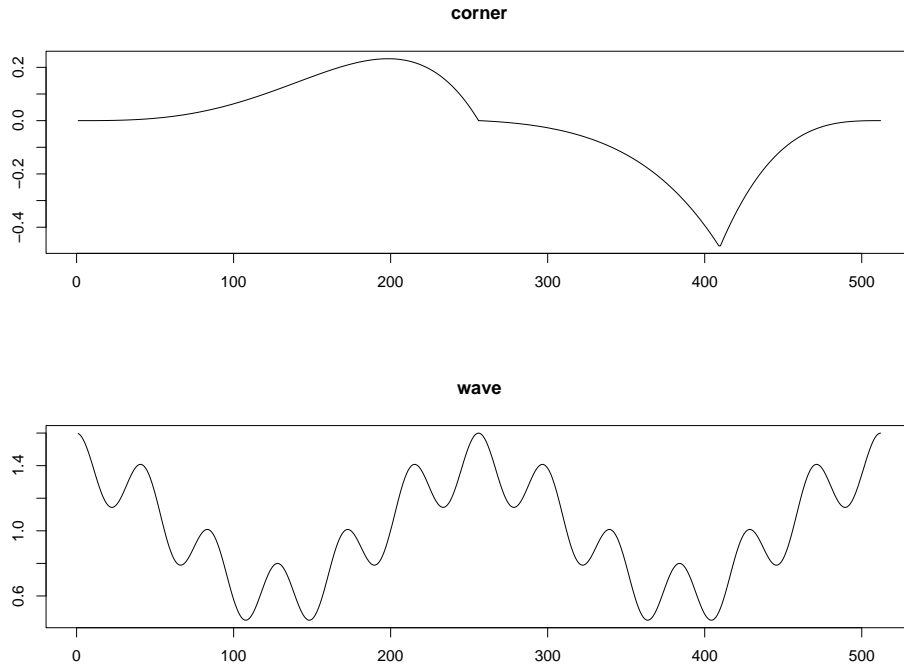


Figure 1: “corner” and “wave” functions.

The second case considered is when the functions g and h are still orthogonal as in the first case above, but the coefficients of the functions may reside on the same resolution levels. This was accomplished by again examining the DWT of the two functions. In this case, g is a multiple of the “corner” function and h is the “doppler” function. See Figure 3. Both functions had the highest 2 resolution levels set to 0, h had the next level set to 0, also. In the levels 4 through 8, coefficients were randomly selected to be 0 for h , and the corresponding coefficients for g were left alone. Finally, the lowest 2 resolution levels of h were set to 0. The functions h and g were then reconstructed using these modified coefficients. So, for any coefficient in the DWT of h that is not 0, the corresponding coefficient for g is 0, and vice versa. Thus, orthogonality of the functions in the wavelet domain with respect to the wavelet basis $la8$ is maintained. See Figure 4 for the g and h in this case.

The third case simulates when g and h are nearly orthogonal in the wavelet domain with respect to the wavelets derived from $la8$. The same functions and methods of

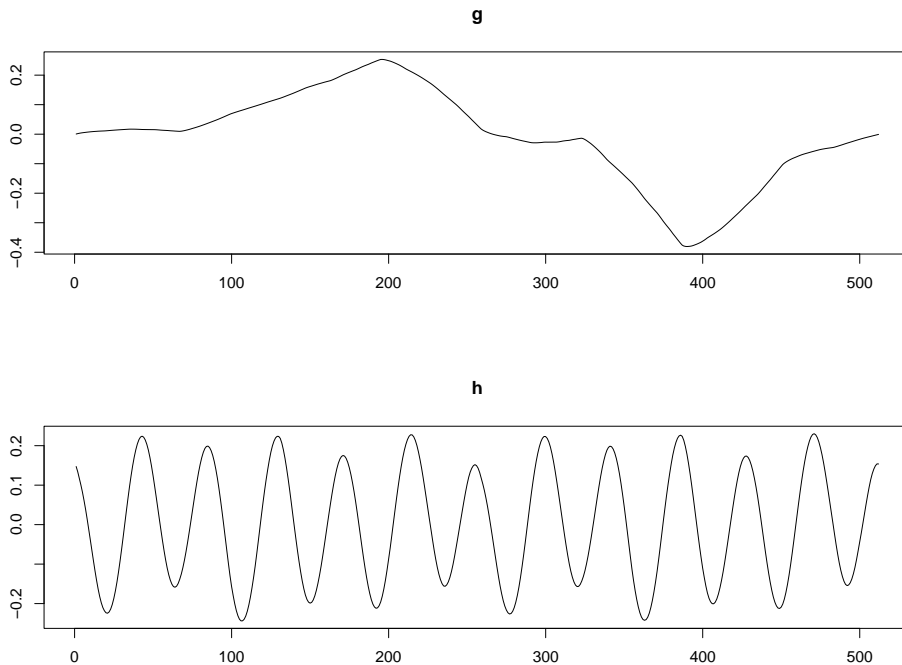


Figure 2: g and h for the case of residing in distinct resolution levels (case 1).

the previous case are used. However, in level 4, two coefficients in g and the two corresponding coefficients in h are both non-zero. So, h and g are nearly orthogonal. These h and g are shown in Figure 5. In all cases, h is centered so that $\int h = 0$.

In addition to varying the snr and the form of the orthogonality between g and h , we also varied the wavelet used for the algorithm. The functions h and g were orthogonalized using the wavelet *la8*. The algorithms were run using this wavelet, as well as the wavelet *bl20* from package *waveslim* in R. In such cases, the functions are “nearly” orthogonal. They are only exactly orthogonal in cases 1 and 2 when the wavelet *la8* is used.

Finally, Figure 6 provides an example of the functions f and h' from case 2 with noise added at snr 5. Recall that the goal of the algorithm is to extract h from f , where h is correlated with h' . In this figure, the ideal relation between h and h' is $h = 2h'$.

2 Simulation results

Using the functions described in the introduction section, simulations using the directional step algorithms were implemented. In each case, the functions were of length $n = 2^9 = 512$. There are a total of 9 detail levels and 1 coarse level of wavelet decomposition available from the DWT. In cases 2 and 3, the function h uses 39 coefficients, in

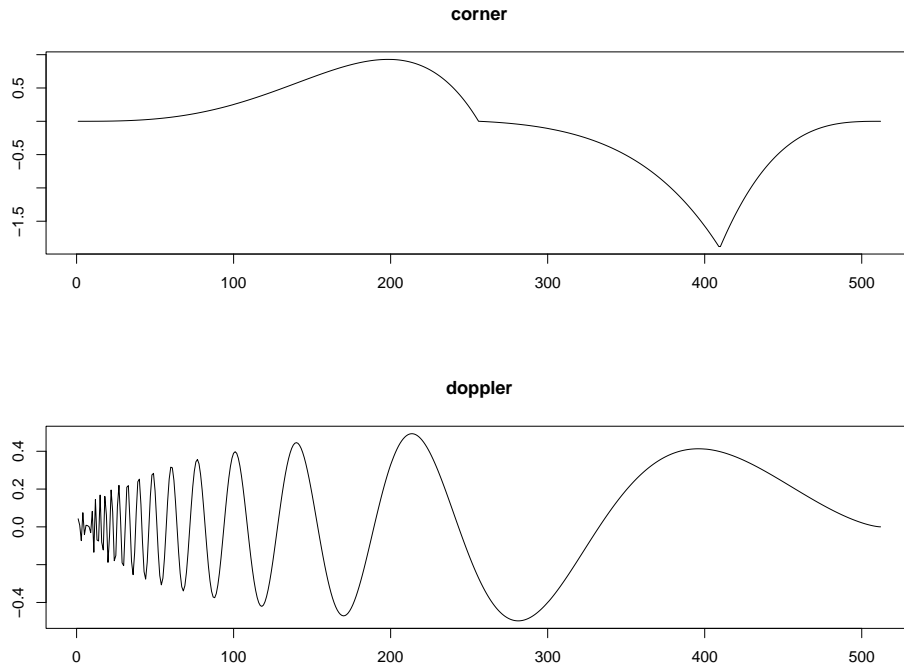


Figure 3: “corner” and “doppler” functions.

case 1, it uses 504 coefficients for its representation. In case 1, most of the coefficients used to describe h are near zero and would be thresholded out by most wavelet methods. In particular, all 256 coefficients at the highest level of wavelet details are near 0.

When implementing these directional step algorithms, the method used a value of 0.01 as the initial level-by-level correlation threshold as explained in Chicken et al. (2006). This initial step successively removes high levels of detail coefficients if their reconstruction do not cumulatively attain a correlation of at least 0.01 with the external function h' . The cutoff to add a step of this initial step is 0.

Table 1 shows the results of applying the forward searching algorithm to the simulated data. 100 runs were completed in order to estimate the mean squared error

$$MSE = \frac{1}{512} \sum_{i=1}^{512} (h(t_i) - \hat{h}(t_i))^2.$$

The wavelet *la8* was used.

Of interest in this table is the pattern with respect to increasing snr. For the lowest simulated snr, the MSE is largest because of the large amount of noise in the functions. Reducing this noise allows the estimator to more accurately model the function h . The table also displays the initial value of the correlation ρ_f between h' and f , the correlation ρ_h between h and h' , and the number of coefficients used to model h (n_c). As the

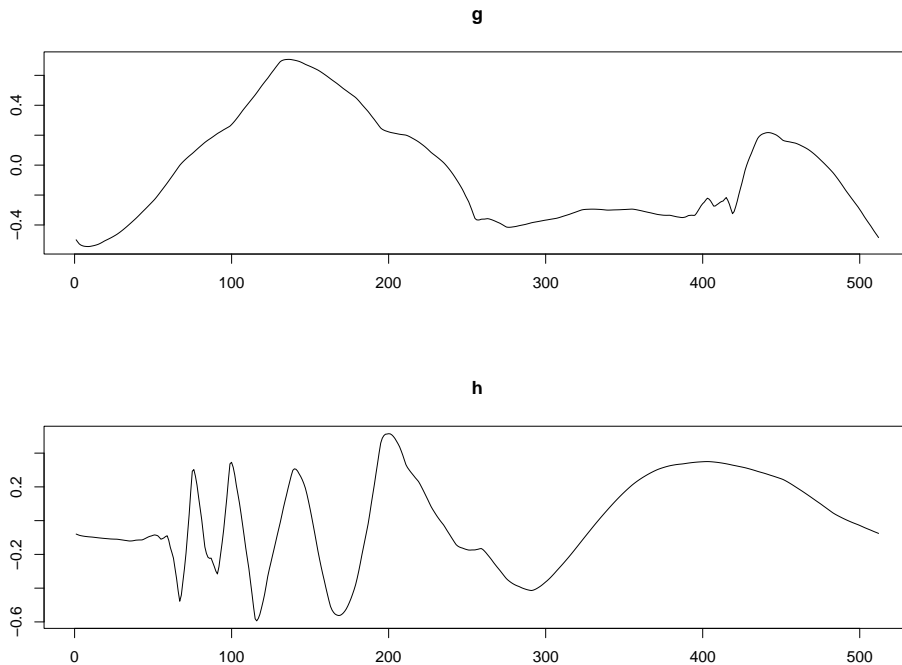


Figure 4: g and h for the case of not necessarily residing in distinct resolution levels (case 2).

snr increases, the MSE decreases, and the remaining quantities increase. The final correlation values ρ_h approach 1 as expected if we are accurately modeling h . The initial correlation approaches the true correlation of f and h' , although this value is not of particular interest. The number of coefficients is increasing as well. However, n_c does not approach the actual number of coefficients needed for h . This is because of the sparseness property of wavelets, especially for case 1.

Case 3 is the nearly orthogonal case. The only difference between the functions h and g in case 2 and case 3 is that in case 3, two of the coefficients are needed for both functions as described in the introduction. So, all the requirements are not met for the algorithm. However, the results are still reasonable. The MSEs are not as good as case 2, but are still very low. Since two of the wavelet coefficients are used for both g and h , then the corresponding coefficients for $f = g + h$ are not the correct size for h . The algorithm may or may not add this coefficient into the map for h , although some part of that coefficient is necessary to model h . This accounts for the larger errors and different number of coefficients used for h .

The best results are in case 1. Smallest errors, highest final correlations. This is not unexpected since in this case the functions are separated very well by resolution level of the MRA.

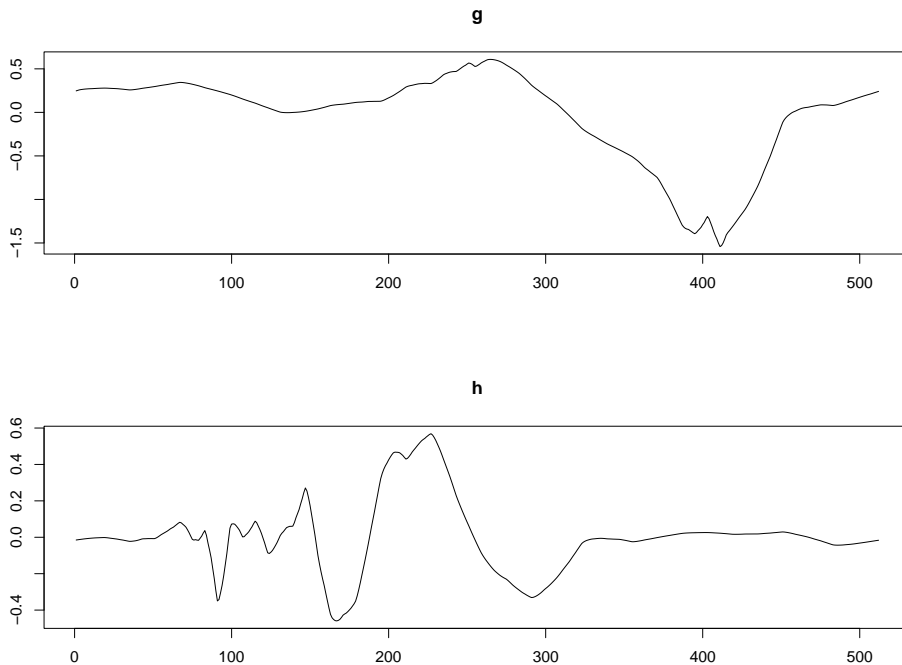


Figure 5: g and h for the case of nearly orthogonal (case 3).

For the situation as given in Figure 6, \hat{h} for the forward step algorithm is given in Figure 7. Not only does the estimate have excellent accuracy in terms of correlation and MSE as shown in table 1, but the estimate is visually pleasing as well. The noise has been removed by the initial level-by-level step described above and in Chicken et al. (2006). The error for this particular run of the algorithm was 0.00140, the initial correlation ρ_f was 0.4831, the final correlation ρ_h is 0.9164, and 35 coefficients were used to model h .

Example of the estimate \hat{h} for cases 1 and 3 are shown in Figures 8 and 9. The error for this particular run of the algorithm for case 1 was 0.00106, the initial correlation ρ_f was 0.5422, the final correlation ρ_h is 0.9007, and 43 coefficients were used to model h . The error for this particular run of the algorithm for case 3 was 0.00086, the initial correlation ρ_f was 0.3056, the final correlation ρ_h is 0.9111, and 17 coefficients were used to model h .

Other directional models were implemented, also. Again, the initial level-by-level step with cutoff 0.01 was used, and 0 was used as the cutoff for removing a coefficient.

Table 2 shows the results of the backward step algorithm. The same general patterns are found here as in Table 1. The main difference we note is that the MSEs are somewhat larger than the forward step method and that more coefficients are used to model h . Using more coefficients is expected since the algorithm starts with all

snr	MSE	ρ_f	ρ_h	n_c
Case 1				
2	0.00282	0.4487	0.7766	26.77
3	0.00162	0.5063	0.8481	38.20
4	0.00121	0.5412	0.8801	41.68
5	0.00092	0.5661	0.9007	43.37
6	0.00080	0.5776	0.9146	44.70
7	0.00071	0.5888	0.9256	45.10
8	0.00064	0.6007	0.9329	46.21
Case 2				
2	0.00686	0.4038	0.7873	13.26
3	0.00552	0.4519	0.8369	14.48
4	0.00508	0.4837	0.8655	15.24
5	0.00472	0.5034	0.8854	15.43
6	0.00451	0.5175	0.8990	15.48
7	0.00444	0.5295	0.9090	15.46
8	0.00371	0.5385	0.9210	17.72
Case 3				
2	0.00649	0.2326	0.7526	6.67
3	0.00570	0.2611	0.8010	7.27
4	0.00528	0.2820	0.8333	7.87
5	0.00449	0.2881	0.8609	8.69
6	0.00426	0.3020	0.8747	9.49
7	0.00358	0.3049	0.8935	10.12
8	0.00294	0.3103	0.9088	11.20

Table 1: Forward step algorithm.

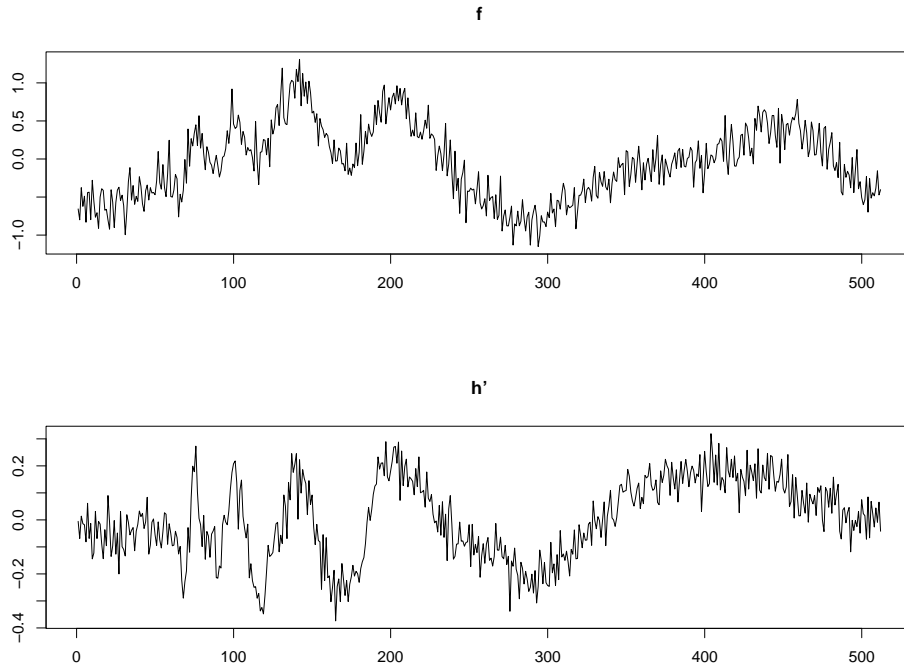


Figure 6: Noisy f and h' for case 2, snr 5.

coefficients. However, in terms of computation, the backward algorithm takes longer than the forward algorithm. Where the forward step method may add 10 coefficients, the backward step algorithm must removed many more than 10 to get to a final count of 10 or 11. So, we prefer the forward algorithm. Better errors, faster computation. Figures of \hat{h} are not provided for the backward step method since they provide little information beyond what can be discerned from the forward step figures.

Finally, a forward-backward and backward-forward algorithm was implemented. Here, the level-by-level step was performed as the first step as before. Then, starting with an empty (if forward-backward) or full (if backward-forward) map, at every step we consider removing or adding a wavelet coefficient and keep the change that increases the correlation the most. The results for the forward-backward step method are essentially the same as the forward method. Likewise, the results for the backward-forward method mimic the backward method. So, the results are not included in this report. We note that a multi-direction method is the most computationally intensive. However, it may be useful in a situation where the initial configuration map of the wavelet coefficients is not empty or full. In this case, the multi-direction map is necessary.

All of the above simulations were run using the wavelet *la8*. This is the same wavelet that was used to ensure that h and g were orthogonal. We then ran these same simulations using a different wavelet, *bl20*. We do this to see what impact the choice of

snr	MSE	ρ_f	ρ_h	n_c
Case 1				
2	0.00299	0.4511	0.7772	27.67
3	0.00187	0.5094	0.8418	38.01
4	0.00132	0.5416	0.8796	42.94
5	0.00107	0.5647	0.9015	44.68
6	0.00095	0.5793	0.9141	45.39
7	0.00084	0.5891	0.9251	46.13
8	0.00073	0.6005	0.9330	47.01
Case 2				
2	0.00925	0.4049	0.7805	14.12
3	0.00763	0.4563	0.8379	15.39
4	0.00707	0.4817	0.8652	15.86
5	0.00683	0.5061	0.8851	15.88
6	0.00643	0.5191	0.8990	16.61
7	0.00605	0.5272	0.9107	17.76
8	0.00565	0.5371	0.9196	18.85
Case 3				
2	0.00847	0.2339	0.7531	8.15
3	0.00763	0.2615	0.8035	8.35
4	0.00712	0.2762	0.8358	8.98
5	0.00662	0.2912	0.8602	9.97
6	0.00593	0.3018	0.8787	10.52
7	0.00562	0.3032	0.8933	11.55
8	0.00510	0.3108	0.9069	12.24

Table 2: Backward step algorithm.

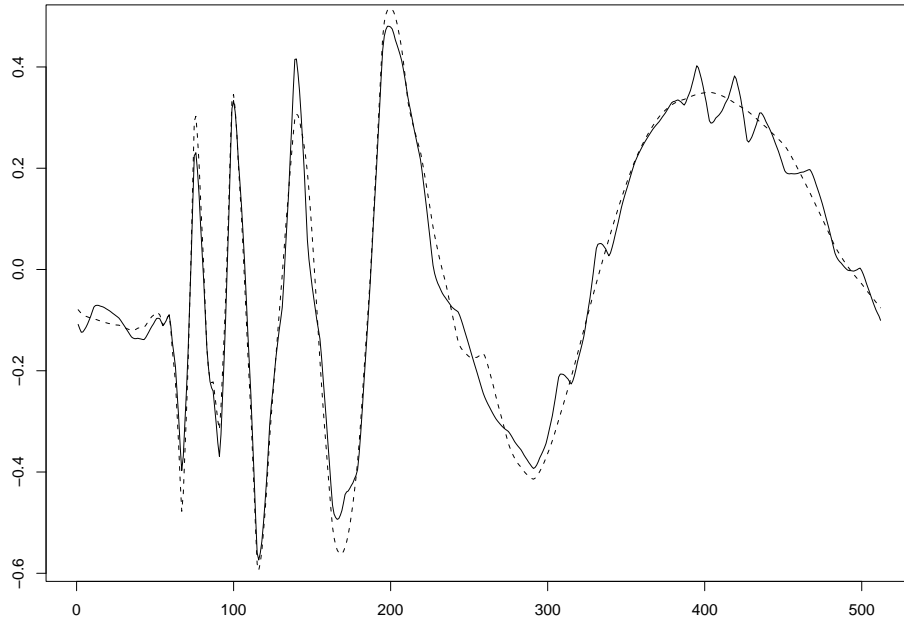


Figure 7: \hat{h} for case 2, snr 5, wavelet *la8*.

wavelet will have on the estimator's results. After all, in practice there is no reason to know ahead of time which wavelet basis comes closest to making the coefficients of the unknown functions g and h most nearly orthogonal.

Now, the wavelet coefficients will be orthogonal when g and h are separated by resolution level (case 1). In cases 2 and 3, the use of the new wavelet basis will not guarantee orthogonality of the coefficients of h and g which are orthogonal with respect to *la8*. Despite this, the results are similar to those obtained with the wavelet *la8*. This leads us to believe the choice of wavelet basis does not unduly affect the estimator's performance.

For the each step method, case 1 shows best results. This is unsurprising since here the functions h and g are separated in resolution spaces. See Tables 3 and 4. For visual reconstruction using the forward step method, see Figures 10, 11, and 12. The differences between forward and backward methods (and combination step methods) are similar to those observed for the wavelet *la8*. In Figure 10, the error is 0.00104, $\rho_f = 0.5727$, $\rho_h = 0.8909$, and $n_c = 36$. In Figure 11, the error is 0.00922, $\rho_f = 0.5251$, $\rho_h = 0.8688$, and $n_c = 27$. In Figure 12, the error is 0.00413, $\rho_f = 0.2909$, $\rho_h = 0.8780$, and $n_c = 19$.

We now turn to the variant of the estimator described in Chicken et al. (2006). Here, we add coefficients by level, not individually. First, the noise removal step is run

snr	MSE	ρ_f	ρ_h	n_c
Case 1				
2	0.00210	0.4458	0.7806	20.96
3	0.00183	0.5056	0.8315	21.12
4	0.00162	0.5391	0.8616	21.40
5	0.00154	0.5631	0.8814	21.64
6	0.00145	0.5761	0.8949	21.99
7	0.00139	0.5907	0.9052	21.85
8	0.00138	0.5973	0.9132	21.94
Case 2				
2	0.01579	0.3988	0.7335	13.20
3	0.01269	0.4526	0.7966	15.94
4	0.01196	0.4820	0.8250	17.50
5	0.01097	0.5031	0.8483	19.87
6	0.01122	0.5151	0.8572	19.49
7	0.00976	0.5317	0.8782	23.31
8	0.00940	0.5356	0.8862	23.62
Case 3				
2	0.00713	0.2311	0.7455	5.03
3	0.00625	0.2614	0.8046	5.14
4	0.00594	0.2794	0.8302	5.32
5	0.00582	0.2897	0.8487	5.17
6	0.00563	0.3012	0.8660	6.29
7	0.00551	0.3081	0.8724	6.32
8	0.00564	0.3120	0.8807	6.46

Table 3: Forward step algorithm, *bl20* wavelet.

snr	MSE	ρ_f	ρ_h	n_c
Case 1				
2	0.00232	0.4467	0.7787	21.92
3	0.00191	0.5068	0.8327	22.21
4	0.00177	0.5400	0.8621	22.47
5	0.00166	0.5663	0.8800	22.72
6	0.00156	0.5794	0.8945	22.61
7	0.00154	0.5926	0.9055	23.05
8	0.00150	0.6014	0.9134	23.65
Case 2				
2	0.01743	0.4007	0.7364	14.10
3	0.01497	0.4517	0.7956	17.19
4	0.01436	0.4828	0.8240	17.78
5	0.01290	0.5049	0.8482	20.18
6	0.01268	0.5138	0.8611	21.37
7	0.01193	0.5233	0.8742	23.35
8	0.01117	0.5371	0.8873	25.80
Case 3				
2	0.00911	0.2286	0.7539	5.95
3	0.00833	0.2655	0.8031	6.14
4	0.00827	0.2814	0.8299	6.40
5	0.00792	0.2944	0.8490	6.23
6	0.00784	0.3023	0.8630	6.65
7	0.00763	0.3076	0.8720	7.27
8	0.00735	0.3120	0.8833	7.72

Table 4: Backward step algorithm, *bl20* wavelet.

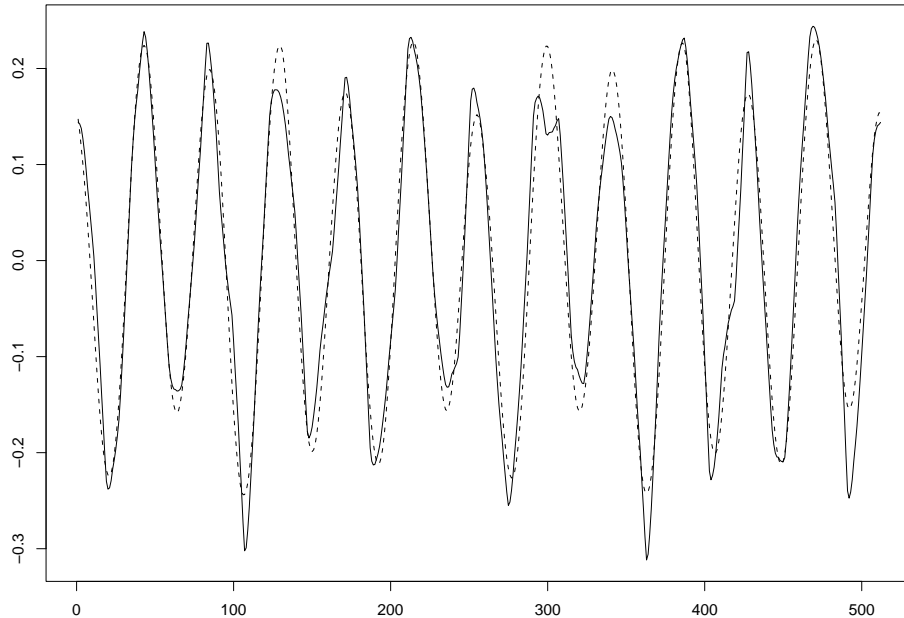


Figure 8: \hat{h} for case 1, snr 5, wavelet *la8*.

with $p' = 0.01$. Then, levels are added starting from the highest remaining coefficient level as long as the correlation of h' and the estimate of h using the selected resolution levels have an increase in correlation ($p = 0$). Levels are added from highest to lowest. Once a level has been added, it is not considered for removal.

Two wavelets were used in these simulations, *la8* and *bl20*. The snr range from 2 to 8. Only the first case of g and h is examined since this variant is specifically designed for this case. The results are shown in Table 5. The results for the two wavelets compare favorably. When compared with the results for the methods which add coefficients one-at-a-time, the only important difference is in n_c . The variant estimator always uses more coefficients. This is not surprising since it adds them by level rather than individually.

Some examples of reconstructions of h are given in Figures 13 and 14. In Figure 13, the error is 0.00090, $\rho_f = 0.5445$, $\rho_h = 0.8924$, and $n_c = 56$. In Figure 14, the error is 0.00129, $\rho_f = 0.5782$, $\rho_h = 0.9024$, and $n_c = 56$.

3 Real Data

For real data, we turn to the tide and discharge data described fully in Chicken et al. (2006). We present here the results from applying the methods to the two sets of data. The first set is similar to case 1. The tidal component of discharge h is separated by

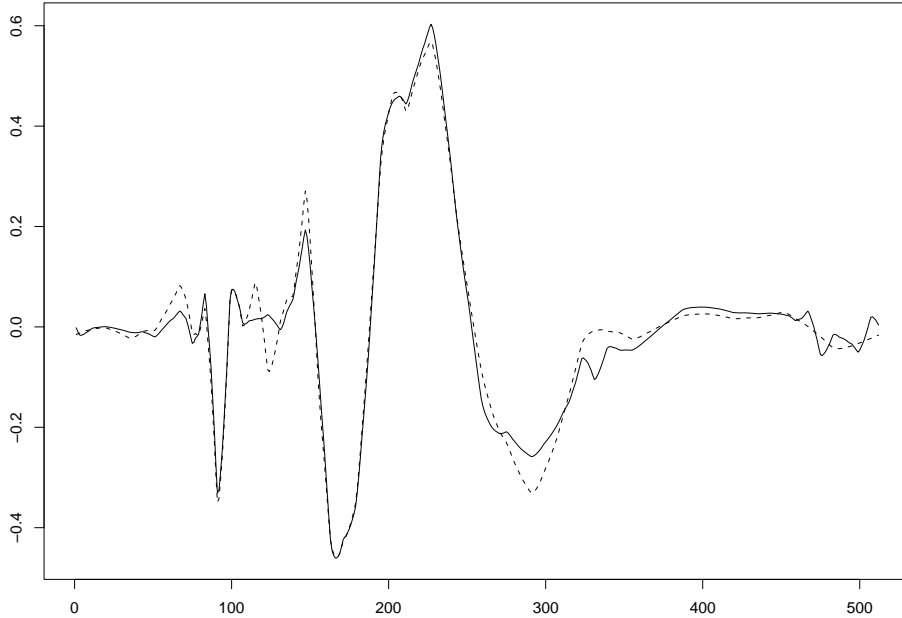


Figure 9: \hat{h} for case 3, snr 5, wavelet *la8*.

resolution spaces from the remainder of the discharge g . This data is shown in Figure 15. The data string is 1024 in length. In the second set, we have either case 2 or 3. The tidal component of discharge now resides in the same resolutions spaces as f . This data is shown in Figure 16. The length of this signal is 2048.

First, the forward step method is applied with wavelet *bl20*. The results are given in Figure 17 for the first set. The method selected $n_c = 111$ of a possible 1024 coefficients, $\rho_f = 0.2137$, $\rho_h = 0.7619$. Applying the backward, forward-backward, backward-forward give the same results but take longer to run. Figure 18 is the forward step method on the same data with wavelet *la8*. Again, all other step methods selected the same coefficients. Here, $n_c = 55$ of a possible 1024 coefficients and $\rho_h = 0.7597$. In Figure 18, note that the scale is m^3/s , which is the right-side scale units in Figure 17. As before, the other step methods give the same results.

Figures 19 and 20 are the analysis of the previous paragraph applied to the second set of data. Using *bl20* gives $n_c = 71$, $\rho_f = 0.6047$, $\rho_h = 0.7529$ (there are 2048 coefficients), while *la8* gives $n_c = 105$ and $\rho_h = 0.7499$. Changing the step method makes no difference in the results.

Next, the methods are applied to the second real data set.

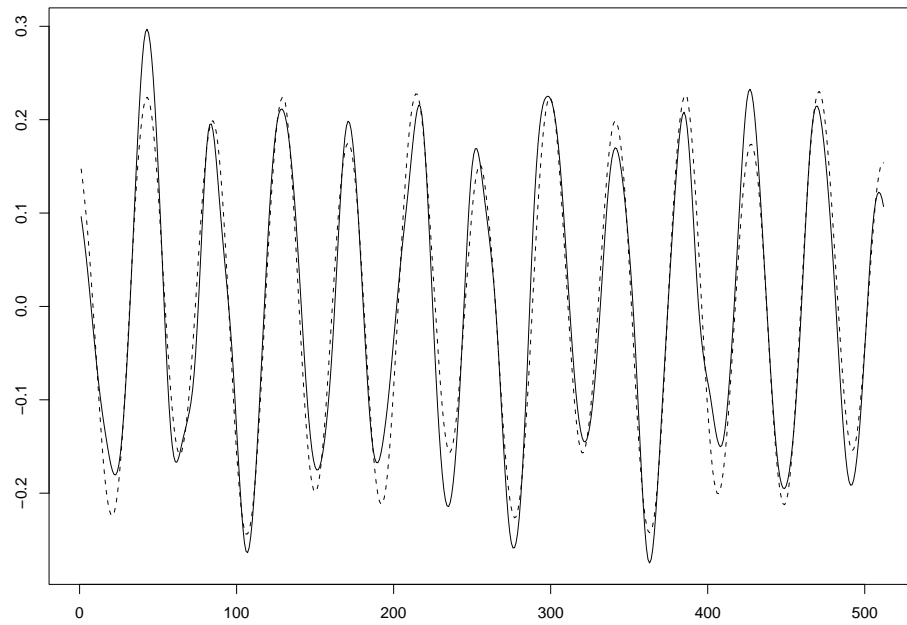


Figure 10: \hat{h} for case 1, snr 5, wavelet *bl20*.

References

CHICKEN, E., LOPER, D. and WERNER, C.(2006). Estimating tidal effects in spring discharge: a multiscale method using correlated phenomena. *Water Resources Research* **XX** XXX–XXX.

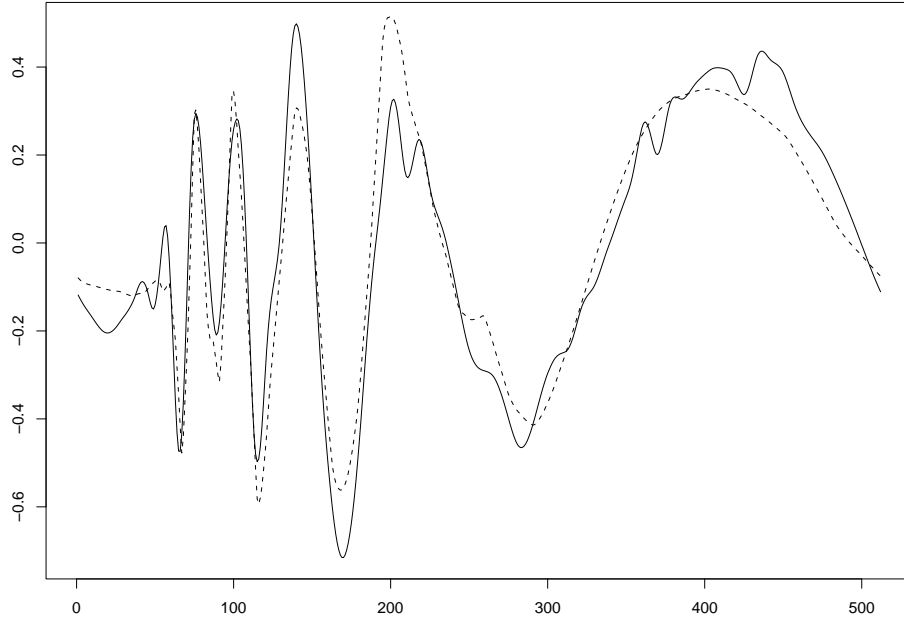


Figure 11: \hat{h} for case 2, snr 5, wavelet *bl20*.

snr	MSE	ρ_f	ρ_h	n_c
<i>la8</i>				
2	0.00316	0.4560	0.7576	40.48
3	0.00197	0.5071	0.8277	51.44
4	0.00130	0.5422	0.8689	55.68
5	0.00104	0.5642	0.8911	56.00
6	0.00088	0.5751	0.9072	56.00
7	0.00072	0.5921	0.9196	56.00
8	0.00066	0.5984	0.9290	56.00
<i>bl20</i>				
2	0.00234	0.4486	0.7731	21.28
3	0.00193	0.5091	0.8272	22.16
4	0.00172	0.5413	0.8577	22.96
5	0.00164	0.5648	0.8760	23.28
6	0.00160	0.5795	0.8903	23.28
7	0.00150	0.5920	0.9013	24.48
8	0.00148	0.6027	0.9093	24.24

Table 5: Variant algorithm, case 1.

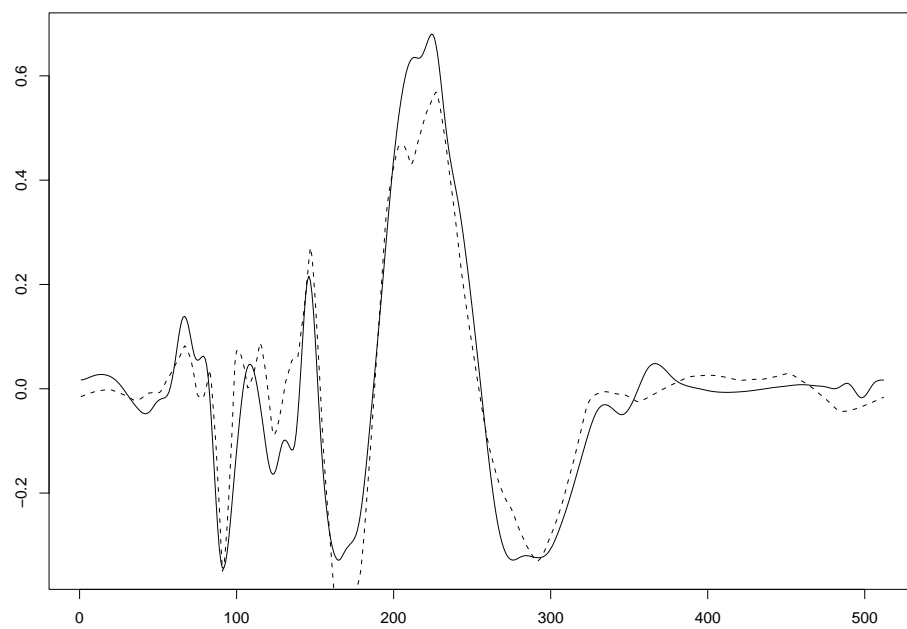


Figure 12: \hat{h} for case 3, snr 5, wavelet *bl20*.

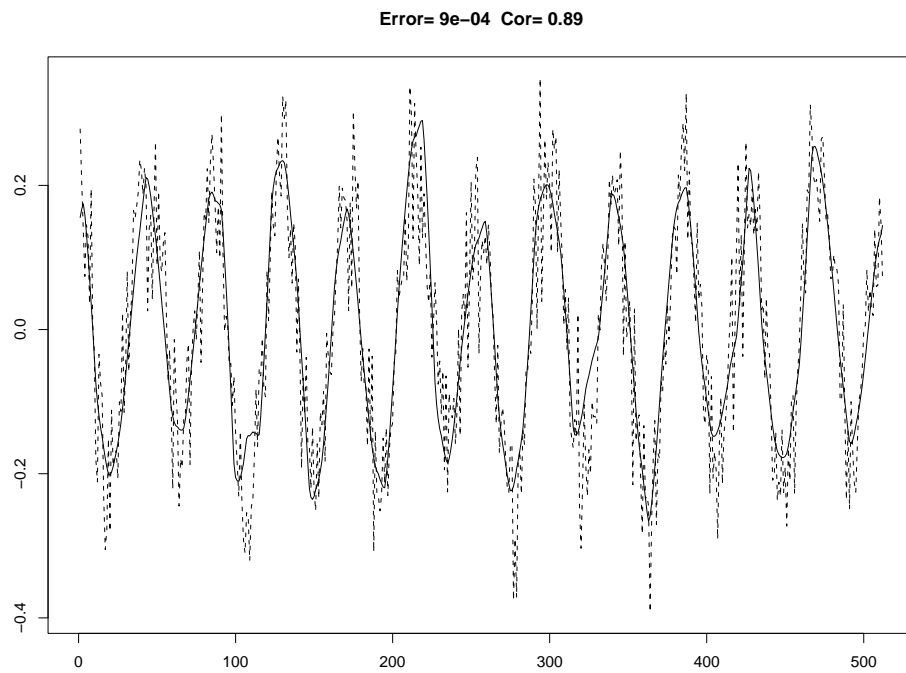


Figure 13: \hat{h} for case 1, snr 5, wavelet *la8*, variant method.

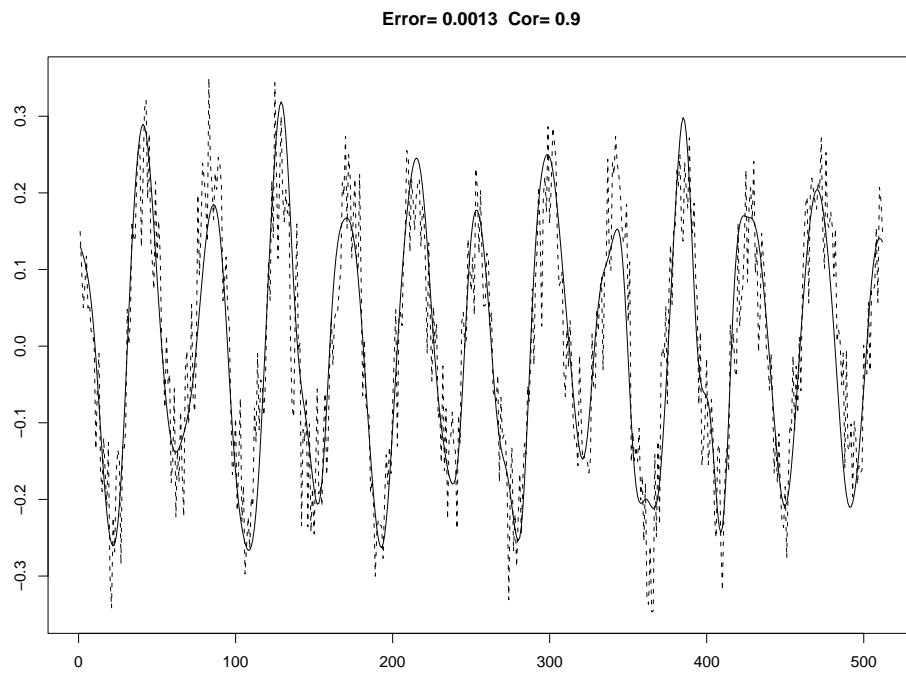
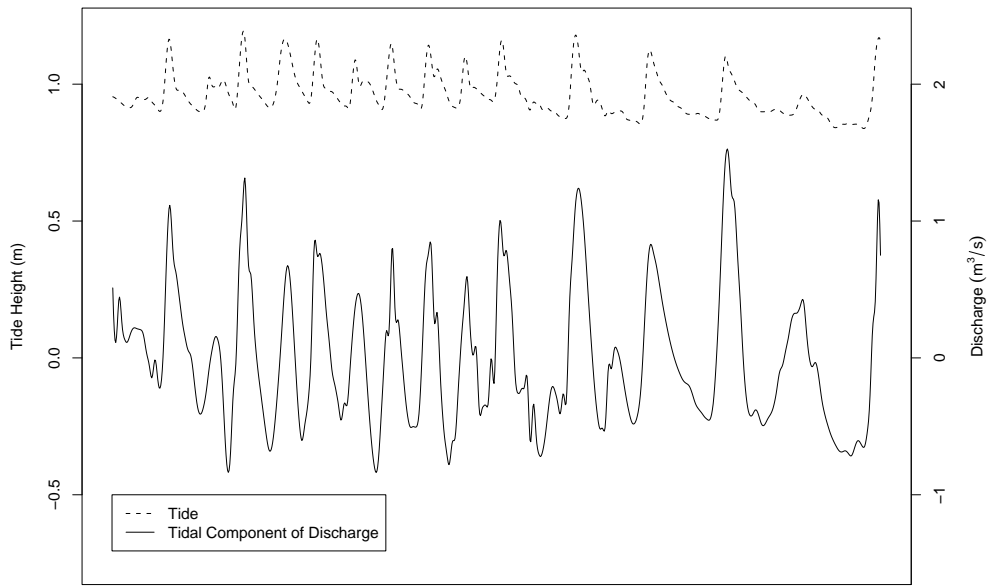
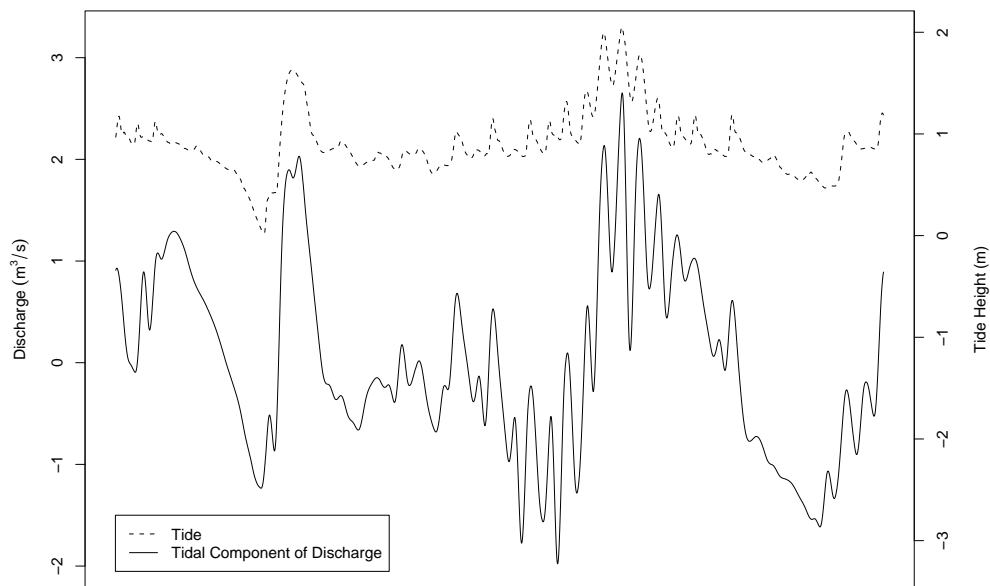


Figure 14: \hat{h} for case 1, snr 5, wavelet *bl20*, variant method.



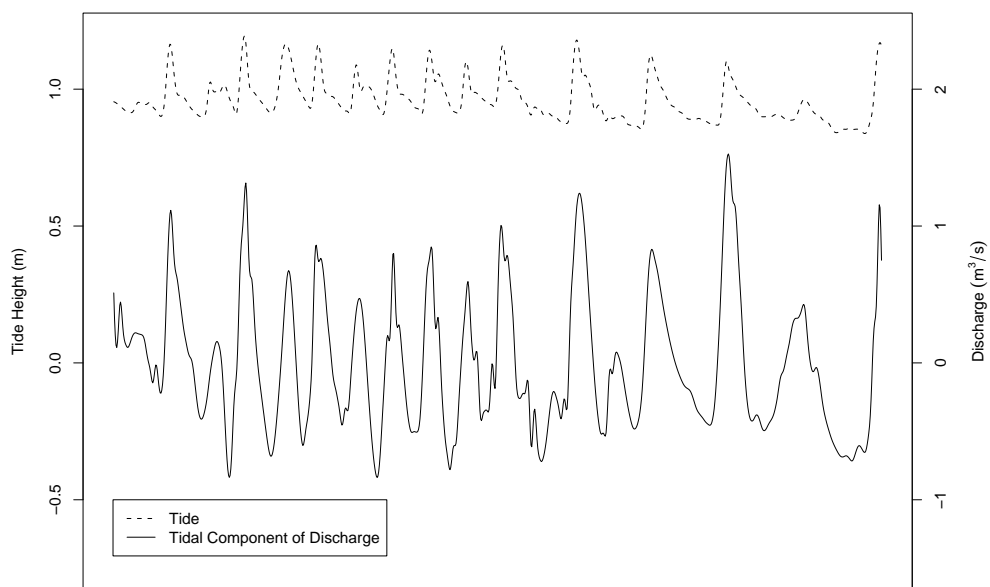
August 15, 2004 – August 28, 2004

Figure 15: Shell Point tide and Wakulla Spring discharge exhibiting regular behavior.



September 2, 2004 – September 23, 2004

Figure 16: Shell Point, Florida, tide and Wakulla Spring discharge exhibiting irregular behavior due to storms.



August 15, 2004 – August 28, 2004

Figure 17: The tidal component of the discharge signal, wavelet *bl20*.

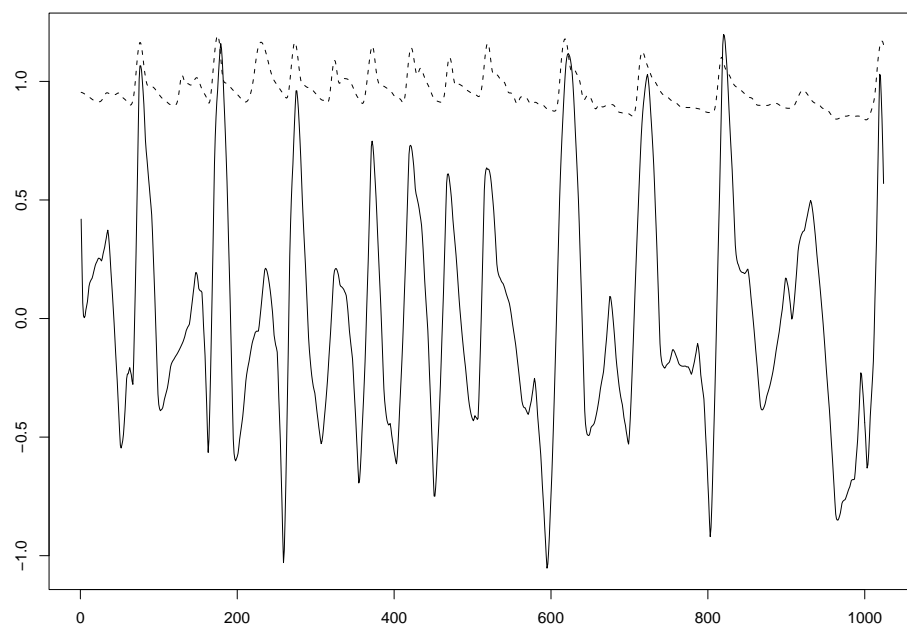


Figure 18: The tidal component of the discharge signal, wavelet $la8$.

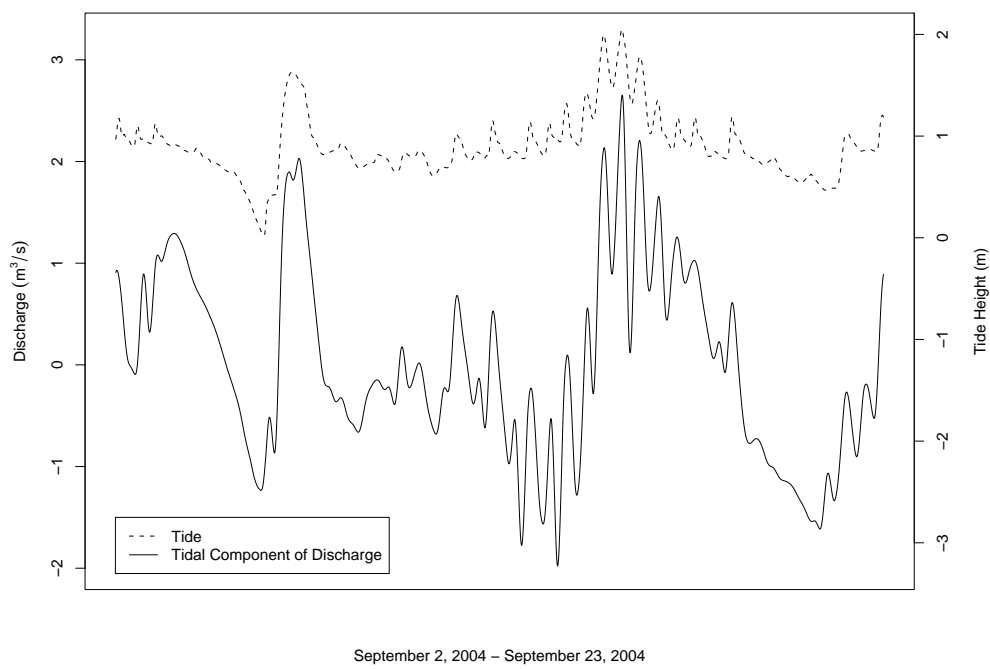


Figure 19: The tidal component of the discharge signal, wavelet $bl20$.

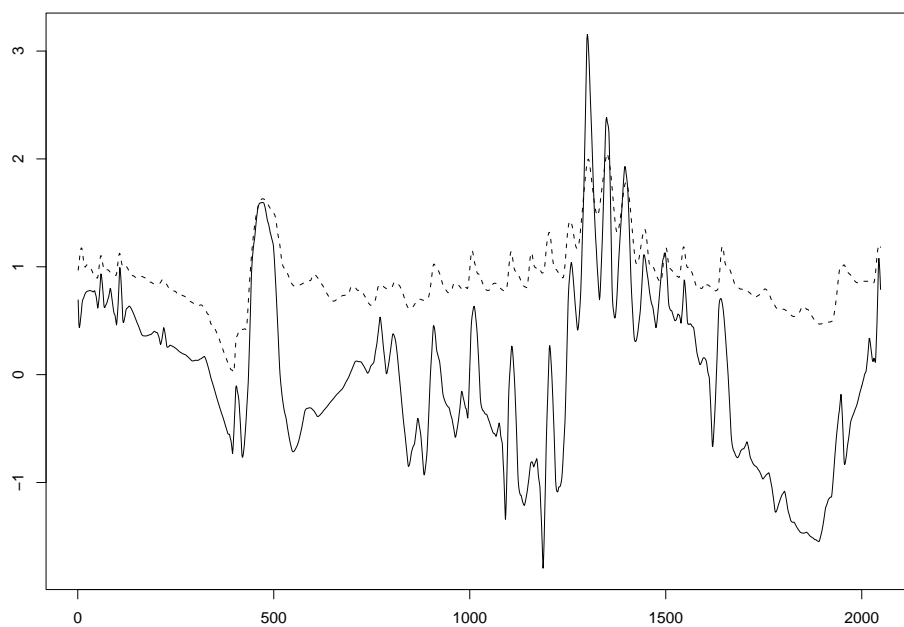


Figure 20: The tidal component of the discharge signal, wavelet $la8$.