An Adaptive Lifting Algorithm and Applications

Marina Knight and Matt Nunes

Department of Mathematics
University of Bristol

ISI 2007, Lisbon
Outline

1. Background Material
   - Classical wavelets
   - Second generation wavelets

2. Adaptive Lifting
   - The lifting scheme removing one coefficient at a time
   - Two adaptive lifting algorithms

3. Applications
   - Adaptive lifting in nonparametric regression
   - Hydrophobic segment prediction for transmembrane proteins

4. Conclusions and further work
Function representation on wavelet bases

- Bases of wavelet functions allow the decomposition of a function in a location-scale environment.

- A function, $f$, can be represented on a wavelet basis as

$$f(x) = \sum_{k} c_{j_0,k} \phi_{j_0,k}(x) + \sum_{j \geq j_0} \sum_{k} d_{j,k} \psi_{j,k}(x),$$

where $\psi$ is the *mother wavelet* and $\psi_{j,k}(x) := 2^{j/2} \psi(2^j x - k)$ its translated and rescaled version. The index $j$ refers to the scale of the wavelet function, and the index $k$ to its location.

- The component $\sum_{k} c_{j_0,k} \phi_{j_0,k}(x)$ represents a coarse ("zoomed out") approximation of the function $f$, while the term $\sum_{j \geq j_0} \sum_{k} d_{j,k} \psi_{j,k}(x)$ contributes the detail lost by the approximation.
Function representation on wavelet bases

- Bases of wavelet functions allow the decomposition of a function in a location-scale environment.
- A function, $f$, can be represented on a wavelet basis as

$$f(x) = \sum_k c_{j_0,k} \varphi_{j_0,k}(x) + \sum_{j \geq j_0} \sum_k d_{j,k} \psi_{j,k}(x),$$

where $\psi$ is the *mother wavelet* and $\psi_{j,k}(x) := 2^{j/2} \psi(2^j x - k)$ its translated and rescaled version. The index $j$ refers to the scale of the wavelet function, and the index $k$ to its location.

- The component $\sum_k c_{j_0,k} \varphi_{j_0,k}(x)$ represents a coarse ("zoomed out") approximation of the function $f$, while the term $\sum_{j \geq j_0} \sum_k d_{j,k} \psi_{j,k}(x)$ contributes the detail lost by the approximation.
Function representation on wavelet bases

- Bases of wavelet functions allow the decomposition of a function in a location-scale environment.
- A function, \( f \), can be represented on a wavelet basis as

\[
f(x) = \sum_k c_{j_0,k} \varphi_{j_0,k}(x) + \sum_{j \geq j_0} \sum_k d_{j,k} \psi_{j,k}(x),
\]

where \( \psi \) is the *mother wavelet* and \( \psi_{j,k}(x) := 2^{j/2} \psi(2^j x - k) \) its translated and rescaled version. The index \( j \) refers to the scale of the wavelet function, and the index \( k \) to its location.

- The component \( \sum_k c_{j_0,k} \varphi_{j_0,k}(x) \) represents a coarse ("zoomed out") approximation of the function \( f \), while the term \( \sum_{j \geq j_0} \sum_k d_{j,k} \psi_{j,k}(x) \) contributes the detail lost by the approximation.
Classical wavelet limitations

The function $f$ can be regarded as being converted into a set of scaling $(c_{j_0,k})$ and wavelet $(d_{j,k})$ coefficients.

Features:
- Wavelet functions are localized.
- Wavelet smoothness can be tuned to match that of the decomposed function.

As such, an efficient representation (few wavelet coefficients) is achievable.

Limitations:
- In many practical applications the smoothness of a signal is unknown, which raises the question what the appropriate choice of decomposing wavelet is.
- Data is assumed to be equally spaced and of length $n = 2^J$ for some $J$. Heavy modifications are required in case one of these assumptions is broken.
The function $f$ can be regarded as being converted into a set of scaling ($c_{j_0,k}$) and wavelet ($d_{j,k}$) coefficients.

**Features:**
- Wavelet functions are localized.
- Wavelet smoothness can be tuned to match that of the decomposed function.

As such, an efficient representation (few wavelet coefficients) is achievable.

**Limitations:**
- In many practical applications the smoothness of a signal is unknown, which raises the question what the appropriate choice of decomposing wavelet is.
- Data is assumed to be equally spaced and of length $n = 2^J$ for some $J$. Heavy modifications are required in case one of these assumptions is broken.
Classical wavelet limitations

- The function $f$ can be regarded as being converted into a set of scaling ($c_{j_0,k}$) and wavelet ($d_{j,k}$) coefficients.

- **Features:**
  - Wavelet functions are localized.
  - Wavelet smoothness can be tuned to match that of the decomposed function.

As such, an efficient representation (few wavelet coefficients) is achievable.

- **Limitations:**
  - In many practical applications the smoothness of a signal is unknown, which raises the question what the appropriate choice of decomposing wavelet is.
  - Data is assumed to be equally spaced and of length $n = 2^J$ for some $J$. Heavy modifications are required in case one of these assumptions is broken.
Possible solutions

- Data observed in practice is likely to be irregularly spaced and not of a dyadic length. Many wavelet methods have been developed to remove the restriction of regular design, e.g. transformation (Cai and Brown, 1998) and interpolation (Kovac and Silverman, 2000).

- Second generation wavelets have also been constructed to enable wavelet decompositions to be applied to very general data situations. They can be designed through the lifting algorithm of Sweldens (1996, 1997).
Data observed in practice is likely to be irregularly spaced and not of a dyadic length. Many wavelet methods have been developed to remove the restriction of regular design, e.g. transformation (Cai and Brown, 1998) and interpolation (Kovac and Silverman, 2000).

Second generation wavelets have also been constructed to enable wavelet decompositions to be applied to very general data situations. They can be designed through the lifting algorithm of Sweldens (1996, 1997).
The Lifting Scheme

For a discrete data vector, $f$, Sweldens proposes:

- **Split** $f$ into two mutually exclusive subsets, say $f^I$ and $f^J$.
- **Predict** one subsample, $f^J$, using the other one, $f^I$, and encode the difference in a set of wavelet (or detail) coefficients $d^J$.
- **Update** $f^I$ by using the information contained in $d^J$ in order to preserve some scalar quantity that characterizes $f$.

Repeat the previous steps on the updated subsample until the end of the decomposition. The initial signal will be replaced by the coarsest updated subsample and the detail coefficients accumulated through the process.
The Lifting Scheme

For a discrete data vector, $f$, Sweldens proposes:

- **Split** $f$ into two mutually exclusive subsets, say $f^I$ and $f^J$.
- **Predict** one subsample, $f^J$, using the other one, $f^I$, and encode the difference in a set of wavelet (or detail) coefficients $d^J$.
- **Update** $f^I$ by using the information contained in $d^J$ in order to preserve some scalar quantity that characterizes $f$.

Repeat the previous steps on the updated subsample until the end of the decomposition. The initial signal will be replaced by the coarsest updated subsample and the detail coefficients accumulated through the process.
The Lifting Scheme

For a discrete data vector, $f$, Sweldens proposes:

- **Split** $f$ into two mutually exclusive subsets, say $f^I$ and $f^J$.
- **Predict** one subsample, $f^J$, using the other one, $f^I$, and encode the difference in a set of wavelet (or detail) coefficients $d^J$.
- **Update** $f^I$ by using the information contained in $d^J$ in order to preserve some scalar quantity that characterizes $f$.

Repeat the previous steps on the updated subsample until the end of the decomposition. The initial signal will be replaced by the coarsest updated subsample and the detail coefficients accumulated through the process.
The Lifting Scheme

For a discrete data vector, \( f \), Sweldens proposes:

- **Split** \( f \) into two mutually exclusive subsets, say \( f^I \) and \( f^J \).
- **Predict** one subsample, \( f^J \), using the other one, \( f^I \), and encode the difference in a set of wavelet (or detail) coefficients \( d^J \).
- **Update** \( f^I \) by using the information contained in \( d^J \) in order to preserve some scalar quantity that characterizes \( f \).

Repeat the previous steps on the updated subsample until the end of the decomposition. The initial signal will be replaced by the coarsest updated subsample and the detail coefficients accumulated through the process.
Variations on the split-predict-update stages exist in the literature.

- Sweldens (1996, 1997) proposes splitting the data into odd and even indices, while Jansen et al. (2001, 2004) propose generating just one wavelet coefficient at each step.
- The prediction step can also be done in various ways: predicting each odd by the following evens or neighbouring evens, or, when removing one coefficient at a time, by regression over the neighbours.
Variations on the split-predict-update stages exist in the literature.


- The prediction step can also be done in various ways: predicting each odd by the following evens or neighbouring evens, or, when removing one coefficient at a time, by regression over the neighbours.

Approaches to the lifting scheme

Variations on the split-predict-update stages exist in the literature.


- The prediction step can also be done in various ways: predicting each odd by the following evens or neighbouring evens, or, when removing one coefficient at a time, by regression over the neighbours.

The philosophy is to choose the “wavelet functions” locally to represent the signal in the most efficient way.


The above adaptive lifting techniques use the odd/even splitting, whereas we shall augment with adaptiveness the lifting “one coefficient at a time” (LOCAAT) methodology of Jansen et al (2001, 2004).

The LOCAAT algorithm is appealing as it can easily be extended from 1D to multiple dimensions, for example networks.
Adaptiveness in the lifting scheme

- The philosophy is to choose the “wavelet functions” locally to represent the signal in the most efficient way.


- The above adaptive lifting techniques use the odd/even splitting, whereas we shall augment with adaptiveness the lifting “one coefficient at a time” (LOCAAT) methodology of Jansen et al (2001, 2004).

- The LOCAAT algorithm is appealing as it can easily be extended from 1D to multiple dimensions, for example networks.
Adaptiveness in the lifting scheme

- The philosophy is to choose the “wavelet functions” locally to represent the signal in the most efficient way.


- The above adaptive lifting techniques use the odd/even splitting, whereas we shall augment with adaptiveness the lifting “one coefficient at a time” (LOCAAT) methodology of Jansen et al (2001, 2004).

- The LOCAAT algorithm is appealing as it can easily be extended from 1D to multiple dimensions, for example networks.
Adaptiveness in the lifting scheme

- The philosophy is to choose the “wavelet functions” locally to represent the signal in the most efficient way.
- The above adaptive lifting techniques use the odd/even splitting, whereas we shall augment with adaptiveness the lifting “one coefficient at a time” (LOCAAT) methodology of Jansen et al (2001, 2004).
- The LOCAAT algorithm is appealing as it can easily be extended from 1D to multiple dimensions, for example networks.
Suppose we have a function, $f$, sampled at $n$ irregularly-spaced points $x = \{x_i\}_{i=1}^n$.

Aim: Transform the sampled function values by means of lifting into a set of detail and scaling coefficients.

Associate an interval $I_{n,i}$ to each sampled point $x_i$ and take the initial scaling functions $\varphi_{n,i}$ to be their corresponding characteristic functions.

Therefore $f$ can be expressed as

$$f(x) = \sum_{i=1}^{n} c_{n,i} \varphi_{n,i}(x),$$

where $f(x_{n,i}) = c_{n,i}$. 
Suppose we have a function, $f$, sampled at $n$ irregularly-spaced points $\mathbf{x} = \{x_i\}_{i=1}^n$.

Aim: Transform the sampled function values by means of lifting into a set of detail and scaling coefficients.

Associate an interval $I_{n,i}$ to each sampled point $x_i$ and take the initial scaling functions $\varphi_{n,i}$ to be their corresponding characteristic functions.

Therefore $f$ can be expressed as

$$f(x) = \sum_{i=1}^{n} c_{n,i} \varphi_{n,i}(x),$$

where $f(x_{n,i}) = c_{n,i}$. 
Suppose we have a function, $f$, sampled at $n$ irregularly-spaced points $x = \{x_i\}_{i=1}^n$.

Aim: Transform the sampled function values by means of lifting into a set of detail and scaling coefficients.

Associate an interval $I_{n,i}$ to each sampled point $x_i$ and take the initial scaling functions $\varphi_{n,i}$ to be their corresponding characteristic functions.

Therefore $f$ can be expressed as

$$f(x) = \sum_{i=1}^{n} c_{n,i} \varphi_{n,i}(x),$$

where $f(x_{n,i}) = c_{n,i}$. 


Suppose we have a function, $f$, sampled at $n$ irregularly-spaced points $\mathbf{x} = \{x_i\}_{i=1}^n$.

Aim: Transform the sampled function values by means of lifting into a set of detail and scaling coefficients.

Associate an interval $I_{n,i}$ to each sampled point $x_i$ and take the initial scaling functions $\varphi_{n,i}$ to be their corresponding characteristic functions.

Therefore $f$ can be expressed as

$$f(x) = \sum_{i=1}^{n} c_{n,i} \varphi_{n,i}(x),$$

where $f(x_{n,i}) = c_{n,i}$. 
Split: choose the point to be lifted, $x_{j_n}$, such that
\[ \int \varphi_{n,j_n}(x) \, dx = \min_{k \in 1,n} \int \varphi_{n,k}(x) \, dx. \]

Predict $c_{n,j_n}$ by using regression over the cloud of points determined by a neighbourhood $I_n$ of $x_{j_n}$. This yields an estimate of the form $\hat{c}_{n,j_n} = \sum_{i \in I_n} a_i^n c_{n,i}$ and the detail coefficient is obtained as
\[ d_{j_n} := c_{n,j_n} - \hat{c}_{n,j_n}. \]

Update only the scaling coefficients associated to the neighbouring points
\[ c_{n-1,i} := c_{n,i} + b_i^n d_{j_n}, \quad \forall i \in I_n, i \neq j_n. \]

Remove $x_{j_n}$ and redistribute its corresponding interval by updating the lengths of the neighbouring intervals.
LOCAAT

- **Split**: choose the point to be lifted, \( x_{jn} \), *such that*
  \[
  \int \varphi_{n,jn}(x) \, dx = \min_{k \in 1,n} \int \varphi_{n,k}(x) \, dx.
  \]

- **Predict** \( c_{n,jn} \) by using regression over the cloud of points determined by a *neighbourhood* \( I_n \) of \( x_{jn} \). This yields an estimate of the form \( \hat{c}_{n,jn} = \sum_{i \in I_n} a^n_i c_{n,i} \) and the detail coefficient is obtained as
  \[
  d_{jn} := c_{n,jn} - \hat{c}_{n,jn}.
  \]

- **Update** only the scaling coefficients associated to the neighbouring points
  \[
  c_{n-1,i} := c_{n,i} + b^n_i d_{jn}, \quad \forall i \in I_n, \, i \neq j_n.
  \]

Remove \( x_{jn} \) and redistribute its corresponding interval by updating the lengths of the neighbouring intervals.
**LOCAAT**

- **Split**: choose the point to be lifted, \( x_{jn} \), *such that* 
  \[
  \int \varphi_{n,jn}(x) \, dx = \min_{k \in 1, n} \int \varphi_{n,k}(x) \, dx.
  \]
- **Predict** \( c_{n,jn} \) by using regression over the cloud of points determined by a *neighbourhood* \( I_n \) of \( x_{jn} \). This yields an estimate of the form \( \hat{c}_{n,jn} = \sum_{i \in I_n} a_i^n c_{n,i} \) and the detail coefficient is obtained as 
  \[
  d_{jn} := c_{n,jn} - \hat{c}_{n,jn}.
  \]
- **Update** only the scaling coefficients associated to the neighbouring points 
  \[
  c_{n-1,i} := c_{n,i} + b_i^n d_{jn}, \quad \forall i \in I_n, i \neq j_n.
  \]

Remove \( x_{jn} \) and redistribute its corresponding interval by updating the lengths of the neighbouring intervals.
The procedure is then repeated on the updated signal, and with each repetition a new detail coefficient is added.

After $n - L + 1$ removals, Jansen et al (2004) state that the signal can be represented as

$$f(x) = \sum_{i \in \{1, \ldots, n\} \setminus \{j_n, j_{n-1}, \ldots, j_L\}} c_{L-1,i} \varphi_{L-1,i}(x) + \sum_{k \in \{n, n-1, \ldots, L\}} d_{jk} \psi_{jk}(x).$$

At each stage, every location is associated to either a scaling or a wavelet coefficient, and the number of wavelet coefficients increases with each removal.

The transform is easily inverted by undoing the update step, then undoing the prediction and merging the data.
The procedure is then repeated on the updated signal, and with each repetition a new detail coefficient is added.

After $n - L + 1$ removals, Jansen et al. (2004) state that the signal can be represented as

$$f(x) = \sum_{i \in \{1, \ldots, n\} \setminus \{j_n, j_{n-1}, \ldots, j_L\}} c_{L-1,i} \varphi_{L-1,i} (x) + \sum_{k \in \{n, n-1, \ldots, L\}} d_k \psi_k(x).$$

At each stage, every location is associated to either a scaling or a wavelet coefficient, and the number of wavelet coefficients increases with each removal.

The transform is easily inverted by undoing the update step, then undoing the prediction and merging the data.
The procedure is then repeated on the updated signal, and with each repetition a new detail coefficient is added.

After \( n - L + 1 \) removals, Jansen et al (2004) state that the signal can be represented as

\[
f(x) = \sum_{i \in \{1, \ldots, n\} \setminus \{j_n, j_{n-1}, \ldots, j_L\}} c_{L-1,i} \varphi_{L-1,i}(x) + \sum_{k \in \{n, n-1, \ldots, L\}} d_k \psi_k(x).
\]

At each stage, every location is associated to either a scaling or a wavelet coefficient, and the number of wavelet coefficients increases with each removal.

The transform is easily inverted by undoing the update step, then undoing the prediction and merging the data.
The procedure is then repeated on the updated signal, and with each repetition a new detail coefficient is added.

After $n - L + 1$ removals, Jansen et al (2004) state that the signal can be represented as

$$f(x) = \sum_{i \in \{1, \ldots, n\} \setminus \{j_n, j_{n-1}, \ldots, j_L\}} c_{L-1,i} \phi_{L-1,i}(x) + \sum_{k \in \{n, n-1, \ldots, L\}} d_{j_k} \psi_{j_k}(x).$$

At each stage, every location is associated to either a scaling or a wavelet coefficient, and the number of wavelet coefficients increases with each removal.

The transform is easily inverted by undoing the update step, then undoing the prediction and merging the data.
Introducing adaptiveness in LOCAAT

Possible sources of adaptiveness are:

- At each step, different configurations of the neighbours can be chosen: the closest neighbours to the removed point might be a choice, or its symmetrical neighbours.

- The number of neighbours to use at each step is also subject to choice.

- The prediction method can be linear, quadratic or cubic regression over the chosen neighbourhood.

In what follows, we will exploit the flexibility of LOCAAT by adaptively choosing the type of prediction (according to a criterion) and the neighbourhood configuration.
Possible sources of adaptiveness are:

- At each step, different *configurations of the neighbours* can be chosen: the closest neighbours to the removed point might be a choice, or its symmetrical neighbours.

- The *number of neighbours* to use at each step is also subject to choice.

- The *prediction method* can be linear, quadratic or cubic regression over the chosen neighbourhood.

In what follows, we will exploit the flexibility of LOCAAT by adaptively choosing the type of prediction (according to a criterion) and the neighbourhood configuration.
Possible sources of adaptiveness are:

- At each step, different configurations of the neighbours can be chosen: the closest neighbours to the removed point might be a choice, or its symmetrical neighbours.

- The number of neighbours to use at each step is also subject to choice.

- The prediction method can be linear, quadratic or cubic regression over the chosen neighbourhood.

In what follows, we will exploit the flexibility of LOCAAT by adaptively choosing the type of prediction (according to a criterion) and the neighbourhood configuration.
Possible sources of adaptiveness are:

- At each step, different configurations of the neighbours can be chosen: the closest neighbours to the removed point might be a choice, or its symmetrical neighbours.
- The number of neighbours to use at each step is also subject to choice.
- The prediction method can be linear, quadratic or cubic regression over the chosen neighbourhood.

In what follows, we will exploit the flexibility of LOCAAT by adaptively choosing the type of prediction (according to a criterion) and the neighbourhood configuration.
Proposed adaptive algorithms

We propose two adaptive strategies:

- **AdaptPred**: At each step, the algorithm chooses the type of regression (linear, quadratic or cubic, with or without an intercept) which generates the smallest detail in absolute value.

  - This method surpasses the choice of regression order.

- **AdaptNeigh**: We introduce even more flexibility by allowing the neighbourhood size and configuration to change at each step.

  - Essentially, *AdaptNeigh* performs the *AdaptPred* procedure for different neighbourhood choices, and minimizes the detail coefficient from all implementations of *AdaptPred*.
  - This second construction completely frees the user from making any choice except for the neighbourhood size.
Proposed adaptive algorithms

We propose two adaptive strategies:

- **AdaptPred**: At each step, the algorithm chooses the type of regression (linear, quadratic or cubic, with or without an intercept) which generates the smallest detail in absolute value.
  
  This method surpasses the choice of regression order.

- **AdaptNeigh**: We introduce even more flexibility by allowing the neighbourhood size and configuration to change at each step.
  
  Essentially, *AdaptNeigh* performs the *AdaptPred* procedure for different neighbourhood choices, and minimizes the detail coefficient from all implementations of *AdaptPred*.
  
  This second construction completely frees the user from making any choice except for the neighbourhood size.
Proposed adaptive algorithms

We propose two adaptive strategies:

- **AdaptPred**: At each step, the algorithm chooses the type of regression (linear, quadratic or cubic, with or without an intercept) which generates the smallest detail in absolute value.
  - This method surpasses the choice of regression order.

- **AdaptNeigh**: We introduce even more flexibility by allowing the neighbourhood size and configuration to change at each step.
  - Essentially, *AdaptNeigh* performs the *AdaptPred* procedure for different neighbourhood choices, and minimizes the detail coefficient from all implementations of *AdaptPred*.
  - This second construction completely frees the user from making any choice except for the neighbourhood size.
Proposed adaptive algorithms

We propose two adaptive strategies:

- **AdaptPred**: At each step, the algorithm chooses the type of regression (linear, quadratic or cubic, with or without an intercept) which generates the smallest detail in absolute value.
  - This method surpasses the choice of regression order.

- **AdaptNeigh**: We introduce even more flexibility by allowing the neighbourhood size and configuration to change at each step.
  - Essentially, *AdaptNeigh* performs the *AdaptPred* procedure for different neighbourhood choices, and minimizes the detail coefficient from all implementations of *AdaptPred*.
  - This second construction completely frees the user from making any choice except for the neighbourhood size.
Proposed adaptive algorithms

We propose two adaptive strategies:

- **AdaptPred**: At each step, the algorithm chooses the type of regression (linear, quadratic or cubic, with or without an intercept) which generates the smallest detail in absolute value.
  - This method surpasses the choice of regression order.

- **AdaptNeigh**: We introduce even more flexibility by allowing the neighbourhood size and configuration to change at each step.
  - Essentially, *AdaptNeigh* performs the *AdaptPred* procedure for different neighbourhood choices, and minimizes the detail coefficient from all implementations of *AdaptPred*.
  - This second construction completely frees the user from making any choice except for the neighbourhood size.
Adaptively choosing the prediction order

**Figure:** Plot showing choice of prediction scheme for the *HeaviSine* test signal decomposed with *AdaptNeigh* with at most 4 neighbours on an irregular grid. Horizontal placement of symbol indicates location of following kinds of prediction: linear (□); quadratic (△); cubic (+); scaling functions (◇).
Comments on the proposed algorithms

- **Neighbourhood size:**
  - The neighbourhood size is user-specified.
  - We advise against large neighbourhoods as this decreases the stability of the transforms.

- **Nonlinearity:**
  - Adaptivity induces algorithm dependence on the signal, making the transform nonlinear.

- **Weights:**
  - Use prediction weights naturally obtained from regression over each neighbourhood.
  - Use update weights of minimum norm to help with the transform stability (Jansen et al, 2001).

- **Multiple observations at x-values:**
  - The prediction and update step can naturally cope with multiple \( f \)-values at the same \( x \)-location.
  - If the removed point is multiple, we produce one detail by taking the average of the individual details.
Comments on the proposed algorithms

- **Neighbourhood size:**
  - The neighbourhood size is user-specified.
  - We advise against large neighbourhoods as this decreases the stability of the transforms.

- **Nonlinearity:**
  - Adaptivity induces algorithm dependence on the signal, making the transform nonlinear.

- **Weights:**
  - Use prediction weights naturally obtained from regression over each neighbourhood.
  - Use update weights of minimum norm to help with the transform stability (Jansen et al., 2001).

- **Multiple observations at x-values:**
  - The prediction and update step can naturally cope with multiple $f$-values at the same $x$-location.
  - If the removed point is multiple, we produce one detail by taking the average of the individual details.
Comments on the proposed algorithms

- **Neighbourhood size:**
  - The neighbourhood size is user-specified.
  - We advise against large neighbourhoods as this decreases the stability of the transforms.

- **Nonlinearity:**
  - Adaptivity induces algorithm dependence on the signal, making the transform nonlinear.

- **Weights:**
  - Use prediction weights naturally obtained from regression over each neighbourhood.
  - Use update weights of minimum norm to help with the transform stability (Jansen et al, 2001).

- **Multiple observations at x-values:**
  - The prediction and update step can naturally cope with multiple $f$-values at the same $x$-location.
  - If the removed point is multiple, we produce one detail by taking the average of the individual details.
Comments on the proposed algorithms

- **Neighbourhood size:**
  - The neighbourhood size is user-specified.
  - We advise against large neighbourhoods as this decreases the stability of the transforms.

- **Nonlinearity:**
  - Adaptivity induces algorithm dependence on the signal, making the transform nonlinear.

- **Weights:**
  - Use prediction weights naturally obtained from regression over each neighbourhood.
  - Use update weights of minimum norm to help with the transform stability (Jansen et al, 2001).

- **Multiple observations at x-values:**
  - The prediction and update step can naturally cope with multiple f-values at the same x-location.
  - If the removed point is multiple, we produce one detail by taking the average of the individual details.
Comments on the proposed algorithms

- **Neighbourhood size:**
  - The neighbourhood size is user-specified.
  - We advise against large neighbourhoods as this decreases the stability of the transforms.

- **Nonlinearity:**
  - Adaptivity induces algorithm dependence on the signal, making the transform nonlinear.

- **Weights:**
  - Use prediction weights naturally obtained from regression over each neighbourhood.
  - Use update weights of minimum norm to help with the transform stability (Jansen et al, 2001).

- **Multiple observations at x-values:**
  - The prediction and update step can naturally cope with multiple f-values at the same x-location.
  - If the removed point is multiple, we produce one detail by taking the average of the individual details.
Comments on the proposed algorithms

- **Neighbourhood size:**
  - The neighbourhood size is user-specified.
  - We advise against large neighbourhoods as this decreases the stability of the transforms.

- **Nonlinearity:**
  - Adaptivity induces algorithm dependence on the signal, making the transform nonlinear.

- **Weights:**
  - Use prediction weights naturally obtained from regression over each neighbourhood.
  - Use update weights of minimum norm to help with the transform stability (Jansen et al, 2001).

- **Multiple observations at $x$-values:**
  - The prediction and update step can naturally cope with multiple $f$-values at the same $x$-location.
  - If the removed point is multiple, we produce one detail by taking the average of the individual details.
Comments on the proposed algorithms

- **Neighbourhood size:**
  - The neighbourhood size is user-specified.
  - We advise against large neighbourhoods as this decreases the stability of the transforms.

- **Nonlinearity:**
  - Adaptivity induces algorithm dependence on the signal, making the transform nonlinear.

- **Weights:**
  - Use prediction weights naturally obtained from regression over each neighbourhood.
  - Use update weights of minimum norm to help with the transform stability (Jansen et al, 2001).

- **Multiple observations at x-values:**
  - The prediction and update step can naturally cope with multiple f-values at the same x-location.
  - If the removed point is multiple, we produce one detail by taking the average of the individual details.
Why adaptiveness?

Pros:
- By using either adaptive strategy, we construct wavelet functions that are tuned to the local features of the signal.
- The usually difficult question of which wavelet to use is waived.
- Better data compression and denoising are achieved.
- Easy implementation for data with multiple $f$ observations at the same $x$ point.

Cons:
- As with all lifting techniques, stability is an issue.
- Computationally more expensive than classical wavelet methods.
Why adaptiveness?

Pros:

- By using either adaptive strategy, we construct wavelet functions that are tuned to the local features of the signal.
- The usually difficult question of which wavelet to use is waived.
- Better data compression and denoising are achieved.
- Easy implementation for data with multiple $f$ observations at the same $x$ point.

Cons:

- As with all lifting techniques, stability is an issue.
- Computationally more expensive than classical wavelet methods.
Wavelets in nonparametric regression

We model the observations as

\[ f_i = g(x_i) + \varepsilon_i, \]

where \( f \) denotes noise-contaminated observations at irregularly spaced points \( x \), \( g \) is the true unknown function, and \( \varepsilon \) is Gaussian noise.

Aim: remove the noise and extract the true signal from \( f \).

One possible solution is the wavelet shrinkage approach of Donoho and Johnstone (1994, 1995):

- Transform \( f \) into a sequence of detail and scaling coefficients.
- Shrink/threshold the wavelet coefficients.
- Invert the transform to recover \( g \).
Wavelets in nonparametric regression

We model the observations as

$$f_i = g(x_i) + \varepsilon_i,$$

where \(f\) denotes noise-contaminated observations at irregularly spaced points \(x\), \(g\) is the true unknown function, and \(\varepsilon\) is Gaussian noise.

**Aim:** remove the noise and extract the true signal from \(f\).

One possible solution is the wavelet shrinkage approach of Donoho and Johnstone (1994, 1995):

- Transform \(f\) into a sequence of detail and scaling coefficients.
- Shrink/threshold the wavelet coefficients.
- Invert the transform to recover \(g\).
Wavelets in nonparametric regression

We model the observations as

\[ f_i = g(x_i) + \varepsilon_i, \]

where \( f \) denotes noise-contaminated observations at irregularly spaced points \( x \), \( g \) is the true unknown function, and \( \varepsilon \) is Gaussian noise.

**Aim**: remove the noise and extract the true signal from \( f \).

One possible solution is the wavelet shrinkage approach of Donoho and Johnstone (1994, 1995):

- Transform \( f \) into a sequence of detail and scaling coefficients.
- Shrink/threshold the wavelet coefficients.
- Invert the transform to recover \( g \).
Wavelets in nonparametric regression

We model the observations as

\[ f_i = g(x_i) + \varepsilon_i, \]

where \( f \) denotes noise-contaminated observations at irregularly spaced points \( x \), \( g \) is the true unknown function, and \( \varepsilon \) is Gaussian noise.

**Aim**: remove the noise and extract the true signal from \( f \).

One possible solution is the wavelet shrinkage approach of Donoho and Johnstone (1994, 1995):

- Transform \( f \) into a sequence of detail and scaling coefficients.
- Shrink/threshold the wavelet coefficients.
- Invert the transform to recover \( g \).
Wavelets in nonparametric regression

We model the observations as

\[ f_i = g(x_i) + \varepsilon_i, \]

where \( f \) denotes noise-contaminated observations at irregularly spaced points \( x \), \( g \) is the true unknown function, and \( \varepsilon \) is Gaussian noise.

**Aim:** remove the noise and extract the true signal from \( f \).

One possible solution is the wavelet shrinkage approach of Donoho and Johnstone (1994, 1995):

- Transform \( f \) into a sequence of detail and scaling coefficients.
- Shrink/threshold the wavelet coefficients.
- Invert the transform to recover \( g \).
Wavelets in nonparametric regression

We model the observations as

\[ f_i = g(x_i) + \varepsilon_i, \]

where \( f \) denotes noise-contaminated observations at irregularly spaced points \( x \), \( g \) is the true unknown function, and \( \varepsilon \) is Gaussian noise.

**Aim:** remove the noise and extract the true signal from \( f \).

One possible solution is the wavelet shrinkage approach of Donoho and Johnstone (1994, 1995):

- Transform \( f \) into a sequence of detail and scaling coefficients.
- Shrink/threshold the wavelet coefficients.
- Invert the transform to recover \( g \).
Simulation study set-up

- We set up a simulation study on the test signals *Doppler*, *Bumps*, *Blocks*, *HeaviSine* and *Ppoly*, sampled on grids with varying degrees of irregularity and different additive noise levels.

- Take a wavelet shrinkage approach: decompose the noisy $f$ with our adaptive lifting transforms, threshold the wavelet coefficients with modified empirical Bayes shrinkage, then invert the transform to obtain an estimate of $g$.

Simulation study set-up

- We set up a simulation study on the test signals *Doppler*, *Bumps*, *Blocks*, *HeaviSine* and *Ppoly*, sampled on grids with varying degrees of irregularity and different additive noise levels.

- Take a wavelet shrinkage approach: decompose the noisy $f$ with our adaptive lifting transforms, threshold the wavelet coefficients with modified empirical Bayes shrinkage, then invert the transform to obtain an estimate of $g$.

Simulation study set-up

- We set up a simulation study on the test signals *Doppler*, *Bumps*, *Blocks*, *HeaviSine* and *Ppoly*, sampled on grids with varying degrees of irregularity and different additive noise levels.

- Take a wavelet shrinkage approach: decompose the noisy \( f \) with our adaptive lifting transforms, threshold the wavelet coefficients with modified empirical Bayes shrinkage, then invert the transform to obtain an estimate of \( g \).

Simulation study results

Extensive simulations show that:

- Best performance for compression and denoising is achieved by *AdaptPred* with neighbourhoods of size two for the smoother signals (*Doppler*, *HeaviSine* and *Ppoly*), and *AdaptNeigh* with neighbourhoods of size at most two for signals with sharper discontinuities (*Blocks* and *Bumps*).

- The irregularity of locations does not influence the denoising performance of our methods.

- On *Blocks* and *Bumps*, our adaptive method outperforms all competitors irrespective of noise level. On *Doppler*, our method outperforms the competitors with the closest being KS. On *HeaviSine*, *Locfit* and *SSCV* outperform our method, while on *Ppoly*, our method denoises best except for the higher noise level, where *CR* gives best results.
Simulation study results

Extensive simulations show that:

- Best performance for compression and denoising is achieved by *AdaptPred* with neighbourhoods of size two for the smoother signals (*Doppler*, *HeaviSine* and *Ppoly*), and *AdaptNeigh* with neighbourhoods of size at most two for signals with sharper discontinuities (*Blocks* and *Bumps*).

- The irregularity of locations does not influence the denoising performance of our methods.

- On *Blocks* and *Bumps*, our adaptive method outperforms all competitors irrespective of noise level. On *Doppler*, our method outperforms the competitors with the closest being KS. On *HeaviSine*, *Locfit* and SSCV outperform our method, while on *Ppoly*, our method denoises best except for the higher noise level, where CR gives best results.
Simulation study results

Extensive simulations show that:

- Best performance for compression and denoising is achieved by *AdaptPred* with neighbourhoods of size two for the smoother signals (*Doppler, HeaviSine* and *Ppoly*), and *AdaptNeigh* with neighbourhoods of size at most two for signals with sharper discontinuities (*Blocks* and *Bumps*).

- The irregularity of locations does not influence the denoising performance of our methods.

- On *Blocks* and *Bumps*, our adaptive method outperforms all competitors irrespective of noise level. On *Doppler*, our method outperforms the competitors with the closest being KS. On *HeaviSine*, *Locfit* and *SSCV* outperform our method, while on *Ppoly*, our method denoises best except for the higher noise level, where CR gives best results.
Simulation study results

Extensive simulations show that:

- Best performance for compression and denoising is achieved by *AdaptPred* with neighbourhoods of size two for the smoother signals (*Doppler, HeaviSine* and *Ppoly*), and *AdaptNeigh* with neighbourhoods of size at most two for signals with sharper discontinuities (*Blocks* and *Bumps*).

- The irregularity of locations does not influence the denoising performance of our methods.

- On *Blocks* and *Bumps*, our adaptive method outperforms all competitors irrespective of noise level. On *Doppler*, our method outperforms the competitors with the closest being *KS*. On *HeaviSine*, *Locfit* and *SSCV* outperform our method, while on *Ppoly*, our method denoises best except for the higher noise level, where *CR* gives best results.
Simulation study results

Extensive simulations show that:

- Best performance for compression and denoising is achieved by *AdaptPred* with neighbourhoods of size two for the smoother signals (*Doppler*, *HeaviSine* and *Ppoly*), and *AdaptNeigh* with neighbourhoods of size at most two for signals with sharper discontinuities (*Blocks* and *Bumps*).

- The irregularity of locations does not influence the denoising performance of our methods.

- On *Blocks* and *Bumps*, our adaptive method outperforms all competitors irrespective of noise level. On *Doppler*, our method outperforms the competitors with the closest being *KS*. On *HeaviSine*, *Locfit* and *SSCV* outperform our method, while on *Ppoly*, our method denoises best except for the higher noise level, where *CR* gives best results.
Simulation study results

Extensive simulations show that:

- Best performance for compression and denoising is achieved by \textit{AdaptPred} with neighbourhoods of size two for the smoother signals (\textit{Doppler}, \textit{HeaviSine} and \textit{Ppoly}), and \textit{AdaptNeigh} with neighbourhoods of size at most two for signals with sharper discontinuities (\textit{Blocks} and \textit{Bumps}).

- The irregularity of locations does not influence the denoising performance of our methods.

- On \textit{Blocks} and \textit{Bumps}, our adaptive method outperforms all competitors irrespective of noise level. On \textit{Doppler}, our method outperforms the competitors with the closest being KS. On \textit{HeaviSine}, \textit{Locfit} and SSCV outperform our method, while on \textit{Ppoly}, our method denoises best except for the higher noise level, where CR gives best results.
True and noisy signals

**Figure:** Plot showing on the left the normalized *Bumps* test function, sampled at \( n = 256 \) irregular locations and on the right its noisy version with signal-to-noise ratio of 7.
Denoised versions

**Figure:** Denoised versions: top left – *AdaptNeigh* with at most 2 neighbours; top right – KS with Daubechies’ Extremal Phase wavelet with 2 vanishing moments; bottom left – SSCV; bottom right – *Locfit*. 
Membrane proteins are an important class of protein structures, but experimental determination of their 3D structure can be very difficult, hence often the only available information is the protein’s primary structure (an ordered sequence of amino acids).

Proteins that span the plasma membrane feature a variable number of transmembranar segments whose locations are of interest.

**Aim:** predict the transmembranar segments starting only from the protein’s primary structure. As these segments consist of highly hydrophobic residues, we shall make use of this information.
Membrane proteins are an important class of protein structures, but experimental determination of their 3D structure can be very difficult, hence often the only available information is the protein’s primary structure (an ordered sequence of amino acids).

Proteins that span the plasma membrane feature a variable number of transmembranar segments whose locations are of interest.

**Aim:** predict the transmembranar segments starting only from the protein’s primary structure. As these segments consist of highly hydrophobic residues, we shall make use of this information.
Membrane proteins are an important class of protein structures, but experimental determination of their 3D structure can be very difficult, hence often the only available information is the protein's primary structure (an ordered sequence of amino acids).

Proteins that span the plasma membrane feature a variable number of transmembrane segments whose locations are of interest.

**Aim:** predict the transmembrane segments starting only from the protein's primary structure. As these segments consist of highly hydrophobic residues, we shall make use of this information.
Adaptive lifting for denoising the hydrophobicity profile

- Construct a hydrophobicity profile: on the horizontal axis take the residues in the order of their appearance, and on the vertical axis their corresponding hydrophobicity values (Kyte and Doolittle, 1982).
- Estimate the residue locations by using the 3D information contained in proteins similar to the protein of interest.
- Transmembrane helices are sequences of predominantly hydrophobic residues, hence our purpose is to detect the points at which sharp changes occur in the hydrophobicity signal.
- Model the hydrophobicity profile as noise-contaminated and estimate the underlying signal using wavelet shrinkage with our adaptive methodology.
Adaptive lifting for denoising the hydrophobicity profile

- Construct a hydrophobicity profile: on the horizontal axis take the residues in the order of their appearance, and on the vertical axis their corresponding hydrophobicity values (Kyte and Doolittle, 1982).
- Estimate the residue locations by using the 3D information contained in proteins similar to the protein of interest.
- Transmembrane helices are sequences of predominantly hydrophobic residues, hence our purpose is to detect the points at which sharp changes occur in the hydrophobicity signal.
- Model the hydrophobicity profile as noise-contaminated and estimate the underlying signal using wavelet shrinkage with our adaptive methodology.
Adaptive lifting for denoising the hydrophobicity profile

- Construct a hydrophobicity profile: on the horizontal axis take the residues in the order of their appearance, and on the vertical axis their corresponding hydrophobicity values (Kyte and Doolittle, 1982).

- Estimate the residue locations by using the 3D information contained in proteins similar to the protein of interest.

- Transmembrane helices are sequences of predominantly hydrophobic residues, hence our purpose is to detect the points at which sharp changes occur in the hydrophobicity signal.

- Model the hydrophobicity profile as noise-contaminated and estimate the underlying signal using wavelet shrinkage with our adaptive methodology.
Adaptive lifting for denoising the hydrophobicity profile

- Construct a hydrophobicity profile: on the horizontal axis take the residues in the order of their appearance, and on the vertical axis their corresponding hydrophobicity values (Kyte and Doolittle, 1982).

- Estimate the residue locations by using the 3D information contained in proteins similar to the protein of interest.

- Transmembrane helices are sequences of predominantly hydrophobic residues, hence our purpose is to detect the points at which sharp changes occur in the hydrophobicity signal.

- Model the hydrophobicity profile as noise-contaminated and estimate the underlying signal using wavelet shrinkage with our adaptive methodology.
Hydropathy profile

The hydropathy profile of undecaprenyl-phosphate galactosephosphotransferase (UniProt entry ‘rfbp-salty’) appears in the figure below. Note how due to its long sequence (476 residues), it is difficult to visually assess which could possibly be the hydrophobic segments.

Figure: Hydropathy profile of ‘rfbp-salty’.
**True and predicted segments**

**Figure:** True and predicted segments for ‘rfbp-salty’: horizontally filled rectangles = True, diagonally filled rectangles = adaptive method, vertically filled rectangles = classical wavelet method.
Conclusions

- We have introduced a new lifting transform which adapts to the features of the signal, and is capable of working on complex data, unlike classical wavelet transforms.
- The proposed algorithm displays good denoising performance on a wide range of signals.
- We used our proposed methodology on real data, where it outperformed classical wavelet techniques.
- Further work includes using adaptive lifting for spectral estimation of time series with missing observations, which induce an irregular design.
Conclusions

- We have introduced a new lifting transform which adapts to the features of the signal, and is capable of working on complex data, unlike classical wavelet transforms.
- The proposed algorithm displays good denoising performance on a wide range of signals.
- We used our proposed methodology on real data, where it outperformed classical wavelet techniques.
- Further work includes using adaptive lifting for spectral estimation of time series with missing observations, which induce an irregular design.
Conclusions

- We have introduced a new lifting transform which adapts to the features of the signal, and is capable of working on complex data, unlike classical wavelet transforms.

- The proposed algorithm displays good denoising performance on a wide range of signals.

- We used our proposed methodology on real data, where it outperformed classical wavelet techniques.

- Further work includes using adaptive lifting for spectral estimation of time series with missing observations, which induce an irregular design.
Conclusions

- We have introduced a new lifting transform which adapts to the features of the signal, and is capable of working on complex data, unlike classical wavelet transforms.
- The proposed algorithm displays good denoising performance on a wide range of signals.
- We used our proposed methodology on real data, where it outperformed classical wavelet techniques.
- Further work includes using adaptive lifting for spectral estimation of time series with missing observations, which induce an irregular design.