Lecture 17: Wavelets

Pratheepa Jeganathan

05/10/2019

Recall

- One sample sign test, Wilcoxon signed rank test, large-sample approximation, median, Hodges-Lehman estimator, distribution-free confidence interval.
- Jackknife for bias and standard error of an estimator.
- Bootstrap samples, bootstrap replicates.
- Bootstrap standard error of an estimator.
- Bootstrap percentile confidence interval.
- Hypothesis testing with the bootstrap (one-sample problem.)
- Assessing the error in bootstrap estimates.
- Example: inference on ratio of heart attack rates in the aspirin-intake group to the placebo group.
- The exhaustive bootstrap distribution.

- Discrete data problems (one-sample, two-sample proportion tests, test of homogeneity, test of independence).
- Two-sample problems (location problem - equal variance, unequal variance, exact test or Monte Carlo, large-sample approximation, H-L estimator, dispersion problem, general distribution).
- Permutation tests (permutation test for continuous data, different test statistic, accuracy of permutation tests).
- Permutation tests (discrete data problems, exchangeability.)
- Rank-based correlation analysis (Kendall and Spearman correlation coefficients.)
- Rank-based regression (straight line, multiple linear regression, statistical inference about the unknown parameters, nonparametric procedures - does not depend on the distribution of error term.)
- Smoothing (density estimation, bias-variance trade-off, curse of dimensionality)
- Nonparametric regression (Local averaging, local regression, kernel smoothing, local polynomial, penalized regression)

- Cross-validation, Variance Estimation, Confidence Bands, Bootstrap Confidence Bands.

Wavelets

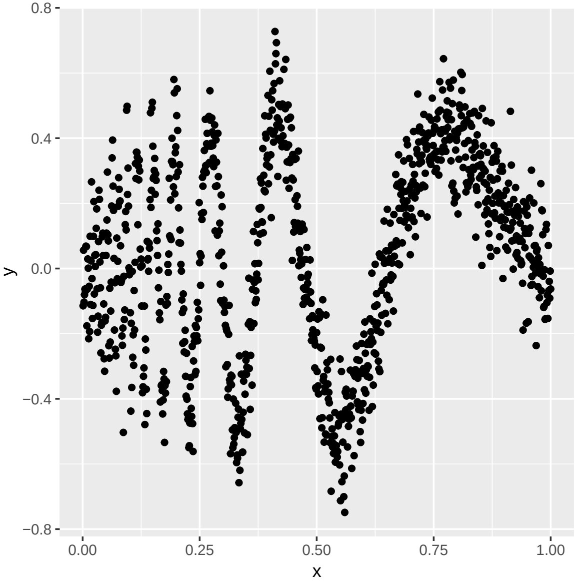
Spatially inhomogeneous functions

Example

- Doppler function

```
library(ggplot2)
r = function(x){
  sqrt(x*(1-x))*sin(2.1*pi/(x+.05))
}
ep = rnorm(1000)
y = r(seq(1, 1000, by = 1)/1000) + .1 * ep
df = data.frame(x = seq(1, 1000, by = 1)/1000, y = y)
ggplot(df) +
  geom_point(aes(x = x, y = y))
```

Example



Example

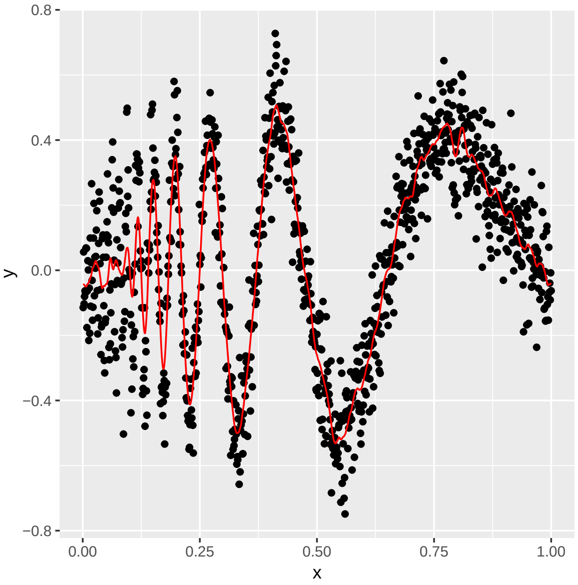
- Doppler function is spatially inhomogeneous (smoothness varies over x).
- Estimate by local linear regression

```
library(np)
doppler.npreg <- npreg(bws=.005,
  txdat=df$x,
      tydat=df$y,
  ckertype="epanechnikov")

doppler.npreg.fit = data.frame(x = df$x,
  y = df$y,
  kernel.fit = fitted(doppler.npreg))

p = ggplot(doppler.npreg.fit) +
  geom_point(aes(x = x, y = y)) +
  geom_line(aes(x = x, y= kernel.fit), color = "red")
```

Example



Example

- Doppler function fit using local linear regression.
  - Effective degrees of freedom 166.
  - Fitted function is very wiggly.
  - If we smooth more, right-hand side of the fit would look better at the cost of missing structure near $x = 0$.

Introduction

- Construct basis functions that are
  - multiscale.
  - spatially/ locally adaptive.
- Find sparse set of coefficients for a given basis.

- Function f belongs to a class of functions $\mathcal{F}$ possessing more general characteristics, such as a certain level of smoothness.
- Estimate f by representing the function in another domain.
- Use an orthogonal series representation of the function f.
- Estimating a set of scalar coefficients that represent f in the orthogonal series domain.
- Tool: Wavelets
  - ability to estimate both global and local features in the underlying function

Sparseness

- **W 2006** Chapter 9
- A function $f = \sum_j \beta_j \phi_j$ is sparse in a basis $\phi_1, \phi_2, \cdots$ if most of the β_j's are zero.
- Sparseness generalizes smoothness: smooth functions are sparse but there are also non smooth functions that are sparse.
- Sparseness is not captured by L_2 norm.
    - Example $\boldsymbol{a} = (1, 0, \cdots, 0)$ and $\boldsymbol{b} = \left(1/\sqrt{n}, 1/\sqrt{n}, \cdots, 1/\sqrt{n}\right)$.
    - $\boldsymbol{a}$ is sparse.
    - L_2 norms are $||\boldsymbol{a}||_2 = ||\boldsymbol{b}||_2 = 1$.
    - L_1 norms are $|\boldsymbol{a}|_1 = 1$ and $|\boldsymbol{b}|_1 = \sqrt{n}$.

Wavelets

- Data: There are n pairs of observations $(x_1, Y_1), (x_2, Y_2), \cdots, (x_n, Y_n)$.
- Assumptions
  - $Y_i = f(x_i) + \epsilon_i$.
  - ϵ_i are IID.
  - $\int f^2 < \infty$ and f is defined on a close interval $[a, b]$. For simplicity, we will consider $[a, b] = [0, 1]$.

Wavelet representation of a function

Basis functions

- $\boldsymbol{\psi} = \{\psi_1, \psi_2, \cdots\}$ is called a basis for a class of functions $\mathcal{F}$. Then, for $f \in \mathcal{F}$,

$$f(x) = \sum_{i=1}^{\infty} \theta_i \psi_i(x).$$

  - θ_i- scalar constants/coefficients
  - Basis functions are orthogonal if $< \psi_i, \psi_j >= 0$ for $i \neq j$.
  - If basis functions are orthonormal, they are orthogonal and $< \psi_i, \psi_i >= 1$.
- How do we construct basis functions ψ_i's ?

Basis functions

- If ψ is a wavelet function, then the collection of functions

$$\mathbf{\Psi} = \{\psi_{ij}; j.k \ \text{ integers}\},$$

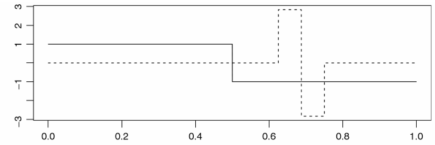
  where

$$\psi_{ij} = 2^{j/2}\psi\left(2^j x - k\right),$$

  forms a basis for square integrable functions.
- $\mathbf{\Psi}$ is a collection of translation (shift) and dilation (scaling) of ψ.
  - ψ can be defined in any range of real line.
  - $\int \psi = 1$
    - value of ψ is near 0 except over a small range.

Some examples for wavelets

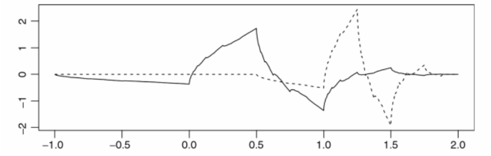
- Haar wavelets (1910)



Source: Hollander, Wolfe, and Chicken (2013)

Some examples for wavelets

- Daubechies wavelets (1992)



Source: Hollander, Wolfe, and Chicken (2013)

Multiresolution analysis (MRA)

- Carefully construct wavelet function ψ.
- MRA: interpretation of the wavelet representation of f in terms of location and scale.
- Translation and dilation of ψ gives

$$f(x) = \sum_{j \in \mathcal{Z}} \sum_{k \in \mathcal{Z}} \theta_{jk} \psi(x),$$

where $\mathcal{Z}$ is a set of integers.

  - scale - frequency.
  - For fixed j, k represents the behavior of f at resolution scale j and a particular location.
  - function f at differing resolution (scale, frequency) levels j and locations k - MRA.

Multiresolution analysis (MRA)

- Cumulative approximation of f using $j < J$,

$$f_J(x) = \sum_{j<J} \sum_{k \in \mathcal{Z}} \theta_{jk} \psi(x).$$

  - J increases - f_J models smaller scales (higher frequency) of f - changes occur in the small interval of x.
  - J decreases - f_J models larger scale (lower frequency) behavior of f.
- A complete representation of f is the limit of f_J.

Multiresolution analysis (MRA)

- Write $f_J(x)$ as follows:

$$f_J(x) = \sum_{k \in \mathcal{Z}} \xi_{j0} \phi_{j0k}(x) + \sum_{j0 \leq j < J} \sum_{k \in \mathcal{Z}} \theta_{jk} \psi_{jk}(x),$$

  where $f_{j0} = \sum_{k \in \mathcal{Z}} \xi_{j0} \phi_{j0k}(x)$.
  - Add second term to f_{j0} allows for modeling higher scale-frequency behavior of f.
  - f_{j0} approximation at the smooth resolution level.
  - Each of the remaining resolution level series is a "detail" level.
  - ϕ - scaling function (Father wavelet).
  - ψ - wavelet function (Mother wavelet).

MRA Using the Haar Wavelet (Example)

- Approximate $f(x) = x, x \in (0,1)$.
- Define Haar wavelet function

$$\psi(x) = \begin{cases} 1 & x \in [0, 1/2), \\ -1 & x \in [1/2, 1), \end{cases} \tag{1}$$

  and

$$\phi(x) = 1, x \in [0, 1]. \tag{2}$$

- Haar wavelet allows exact determination of the wavelet coefficients θ_{jk}.

- Source **HWC**

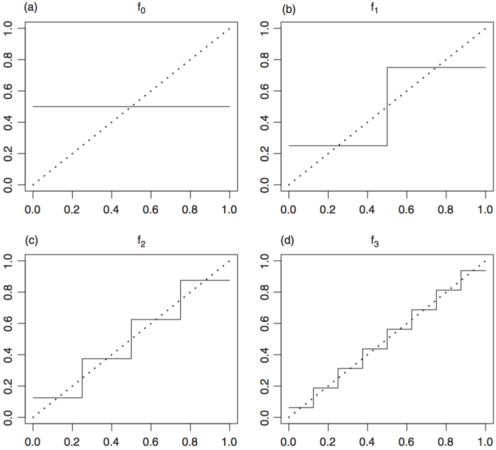


Figure 13.2 Cumulative approximations up to resolution levels $J = -1, 0, 1, 2$ from Example 13.1 using the Haar wavelet. The underlying function is $f(x) = x$, shown with a dotted line.

MRA Using the D2 wavelet

- Source **HWC**

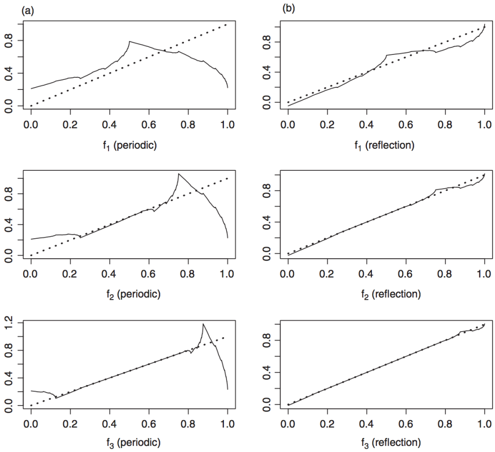


Figure 13.4 Approximations up to resolution levels $j = 0, 1, 2$ from Example 13.1 using the D2 wavelet. The left panels use periodic boundary handling, the right panel use reflection. The underlying function is $f(x) = x$, shown with a dotted line.

MRA Using the D2 wavelet

- To avoid boundary issues using D2
  - Specify using reflection at the boundaries, rather than periodicity.
  - increase the number of indices k that must be considered at each resolution level j.

Discrete wavelet transform

- Cascade algorithm provides MRA (Mallat 1989).
- Some restrictions
  - $J = \log_2(n)$.
  - The number of resolution levels in the wavelet series is truncated both above and below in practice, resulting in $J - j0 + 1$ series, each representing a resolution level.
- Commands in R that make use of the DWT are `dwt`, `idwt`, and `mra` in package `waveslim` (Whitcher (2010)).

Discrete wavelet transform (Example)

- $y_i = x_i = (i-1)/n, i = 1, 2, \cdots, n.$

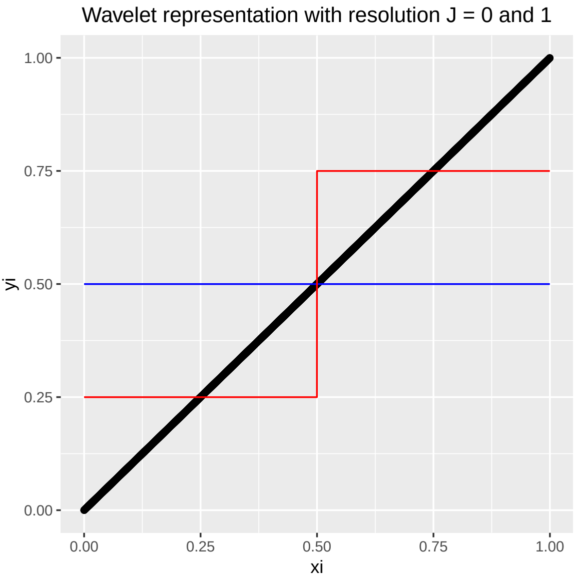
```
n = 2^12
xi  = (seq(1, n, by =1) - 1)/n
yi = xi
library(waveslim)
```

- Haar basis.
- Number of resolution levels $J = 12$.
- Decompose the sample data y.

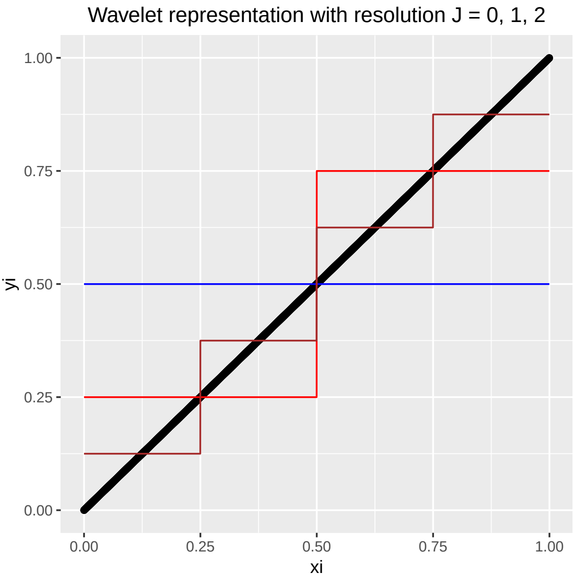
```
dwt.fit = mra(yi, method="dwt", wf="haar", J=12)
```

- Output is a list of 13 vectors.
- The first vector is the change necessary to go from the approximation f_{12} to f_{13}- approximation at the highest detail resolution level.
- The next to last vector is $f_1 - f_0$.
- The final, thirteenth vector is the smooth approximation f_0.
- Summing the thirteenth vector and the twelfth vector results in f_1.
- $f_{13} - f_{12}, f_{12} - f_{11}, \cdots, f_1 - f_0, f_0$.

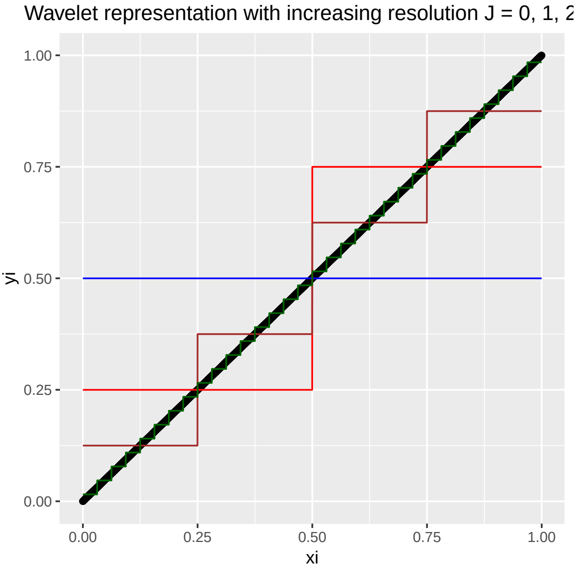
```
f0 = dwt.fit[[13]]
f1 = dwt.fit[[13]]+dwt.fit[[12]]
df = data.frame(x = xi, y = yi,
  f0=f0, f1 = f1)
p1 = ggplot() +
  geom_point(data  = df ,
    aes(x = xi, y = yi))+
  geom_line(data  = df ,
    aes(x = xi, y = f0), color = "blue") +
  geom_line(data = df,
    aes(x = xi, y = f1), color = "red")
```


Wavelet representation with resolution J = 0 and 1
1.00
0.75
0.50
0.25
0.00
yi
0.00
0.25
0.50
0.75
1.00
xi

```r
f2 = dwt.fit[[13]]+dwt.fit[[12]] + dwt.fit[[11]]
df = data.frame(x = xi, y = yi,
  f0 = f0, f1 = f1, f2 = f2)
p2 = ggplot() +
  geom_point(data  = df ,
    aes(x = xi, y = yi)) +
  geom_line(data = df,
    aes(x = xi, y = f0), color = "blue")+
  geom_line(data = df,
    aes(x = xi, y = f1), color = "red")+
  geom_line(data = df,
    aes(x = xi, y = f2), color = "brown")
```


Wavelet representation with resolution J = 0, 1, 2
1.00
0.75
0.50
0.25
0.00
yi
0.00
0.25
0.50
0.75
1.00
xi

```r
f5 = dwt.fit[[13]]+dwt.fit[[12]] + dwt.fit[[11]] + dwt.fit
df = data.frame(x = xi, y = yi,
  f0 = f0, f1 = f1, f2 = f2, f5 = f5)
p5 = ggplot() +
  geom_point(data  = df ,
    aes(x = xi, y = yi)) +
  geom_line(data = df,
    aes(x = xi, y = f0), color = "blue")+
  geom_line(data = df,
    aes(x = xi, y = f1), color = "red")+
  geom_line(data = df,
    aes(x = xi, y = f2), color = "brown")+
  geom_line(data = df,
    aes(x = xi, y = f5), color = "darkgreen")
```



Wavelet representation with increasing resolution J = 0, 1, 2
yi
xi
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00

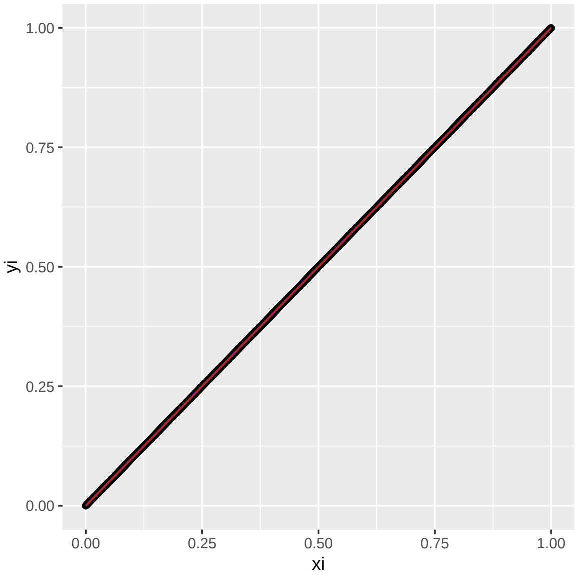
- What if we choose J is less than 12 for this example?
- Set $J = 3$
- $j0 > 0$, for example, when $J = 3$, $j0 = 9$.

```
dwt.fit.J3 = mra(yi, method="dwt", wf="haar", J=3)
```

```r
length(dwt.fit.J3)
```

```
## [1] 4
```

```r
f9 = dwt.fit.J3[[4]] # f0
f10 = dwt.fit.J3[[4]] + dwt.fit.J3[[3]] # f1
f11 = dwt.fit.J3[[4]] + dwt.fit.J3[[3]] + dwt.fit.J3[[2]]#
df = data.frame(x = xi, y = yi,
  f0 = f9, f1 = f10, f2 = f11)
p.J3 = ggplot() +
  geom_point(data  = df ,
    aes(x = xi, y = yi)) +
  geom_line(data = df,
    aes(x = xi, y = f0), color = "blue")+
  geom_line(data = df,
    aes(x = xi, y = f1), color = "red")+
  geom_line(data = df,
    aes(x = xi, y = f2), color = "brown")
```

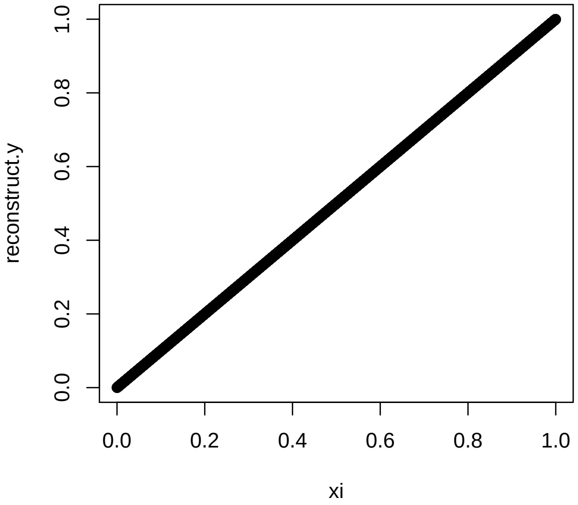

1.00
0.75
0.50
0.25
0.00
yi
0.00
0.25
0.50
0.75
1.00
xi

- `dwt` determines the wavelet coefficients at each resolution level.
  - `n.levels` - resolution levels to determine.
- Read Page 637 for more detail.

```
y.dwt <- dwt(yi, wf="haar", n.levels=12)
```

- The resulting R list of coefficients may be used to reconstruct the original vector of sampled data y.

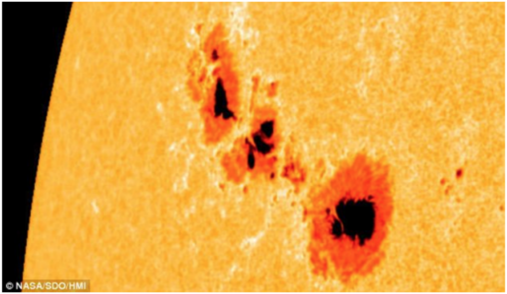
```
reconstruct.y = idwt(y.dwt)
plot(xi, reconstruct.y)
```



Wavelet Thresholding

- We saw how a function f may be represented with a wavelet basis.
    - DWT, a sample of length n from f may be decomposed into n wavelet coefficients making up a single smooth approximation and up to $J = \log_2(n)$. detail resolution levels.
- Sparsity - the ability of wavelets to represent a function by concentrating or compressing the information about f into a few large magnitude coefficients and many small magnitude coefficients.
- Compression (thresholding) is applied to the wavelet coefficients of a sampled function f prior to its reconstruction.
- Thresholding - provides a significant level of data reduction for the problem.

Sparsity of the Wavelet Representation

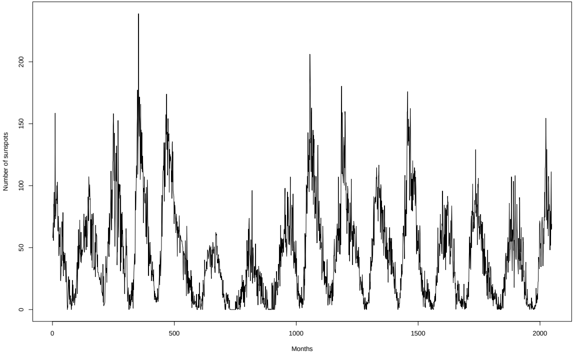


- **HWC** Example 13.3
- Monthly sunspot numbers from 1749 to 1983.
- Sunspots are temporary phenomena on the photosphere of the sun that appear visibly as dark spots compared to surrounding regions.

- Sunspots correspond to concentrations of magnetic field flux that inhibit convection and result in reduced surface temperature compared to the surrounding photosphere.
- The original data has length 2820, but only the first 2048 are used here to make it a dyadic number.
- So the filtered data is monthly sunspot data from January 1749 through July 1919.

```
library(datasets)
data(sunspots)
```

```r
plot.ts(sunspots[1:2048],
  ylab = "Number of sunspots",
  xlab = "Months")
```



- The DWT is applied to this data resulting in 2048 coefficients.

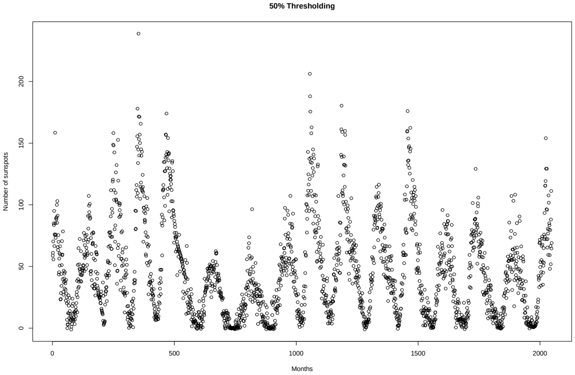
```
dwt.sunspot = dwt(sunspots[1:2048], n.levels = 4, wf = "la8
```

- These coefficients are sorted in magnitude and the smallest 50% (1024) are set to 0.
  - Reconstruction nearly indistinguishable from the original data.

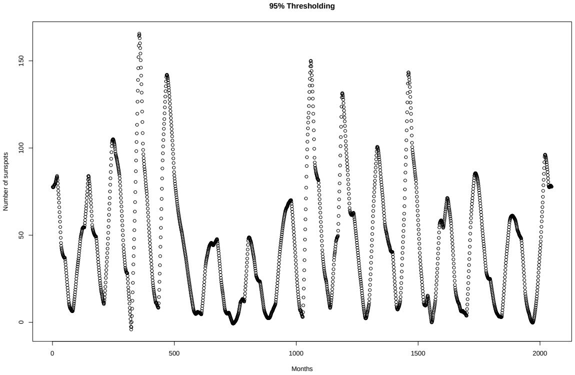
```
dwt.sunspot.coeff = unlist(dwt.sunspot)
dwt.sunspot.coeff = sort(dwt.sunspot.coeff,
  decreasing = T)
val = as.numeric(quantile(dwt.sunspot.coeff,
  p =.5))
manual.thresholding = manual.thresh(dwt.sunspot,
  value = val)
```

- The inverse DWT is applied to this compressed (50% thresholding) set of coefficients, resulting in the reconstruction.

```
y.idwt.manual.thresholding = idwt(manual.thresholding)
plot(y.idwt.manual.thresholding,
  ylab = "Number of sunspots",
  xlab = "Months",
  main = "50% Thresholding")
```


50% Thresholding
Number of sunspots
Months
0
50
100
150
200
0
500
1000
1500
2000

- Set smallest 95% of the coefficients to 0 prior to reconstruction.
    - Reconstruction with the basic shape of the original data, but with the very localized variability mostly removed.



95% Thresholding
Number of sunspots
Months
0
50
100
150
0
500
1000
1500
2000

Thresholding

- A drawback to compression - need to specify the amount of reduction.
- Thresholding specifies a data-driven compression.
- Many methods of thresholding are based on assuming that the errors are normally distributed.

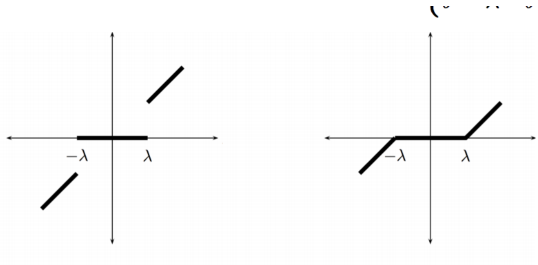
Thresholding

- Let θ is a coefficient estimated with the DWT and λ is a specified threshold value.
- Hard thresholding
  - sets a coefficient to 0 if it has small magnitude and leaves the coefficient unmodified otherwise.
- Soft thresholding
  - threshold sets small coefficients to 0 and shrinks the larger ones by λ toward 0.
- DWT operation may be represented as a matrix operator W

$$\tilde{\theta} = Wf + W\epsilon.$$

  - $\theta = Wf$ represents the wavelet coefficients of the unobserved sampled function f.
  - $\tilde{\epsilon} = W\epsilon$ represents the coefficients of the errors.

Thresholding



Hard thresholding

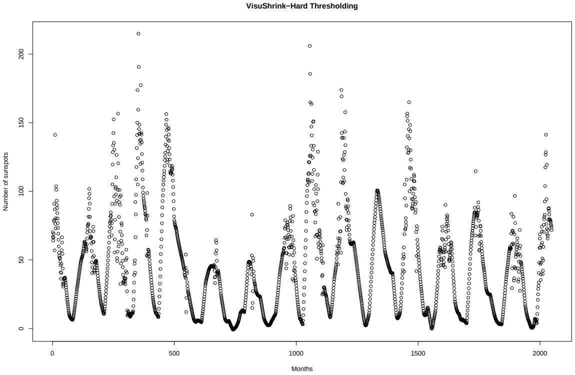
Soft thresholding

Source: Wasserman (2006)

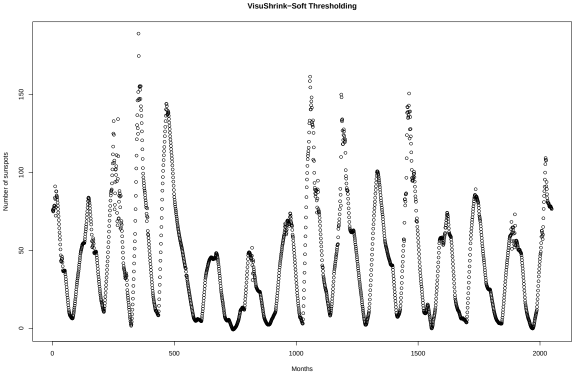
Thresholding - VisuShrink (Donoho and Johnstone (1994))

- Applying a single threshold λ (Donoho and Johnstone 1994).

```
y = sunspots[1:2048]
y.dwt = dwt(sunspots[1:2048])
y.visuShrink = universal.thresh(y.dwt, hard = TRUE)
y.idwt.visuShrink = idwt(y.visuShrink)
plot(y.idwt.visuShrink,
  ylab = "Number of sunspots",
  xlab = "Months",
  main = "VisuShrink-Hard Thresholding")
```


VisuShrink−Hard Thresholding
Number of sunspots
Months
0
50
100
150
200
0
500
1000
1500
2000

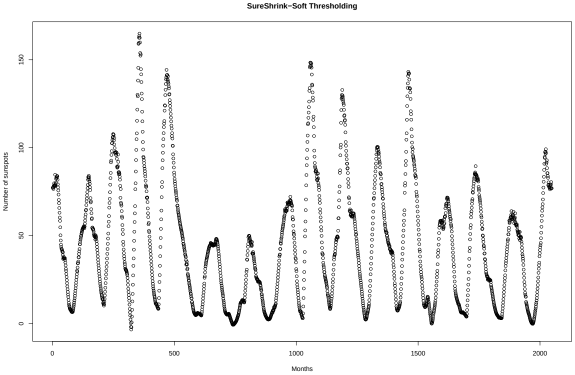
```r
y.visuShrink.soft = universal.thresh(y.dwt, hard = FALSE)
y.idwt.visuShrink.soft = idwt(y.visuShrink.soft)
plot(y.idwt.visuShrink.soft,
  ylab = "Number of sunspots",
  xlab = "Months",
  main = "VisuShrink-Soft Thresholding")
```


VisuShrink−Soft Thresholding
Number of sunspots
0
50
100
150
0
500
1000
1500
2000
Months

Thresholding - SureShrink (Donoho and Johnstone (1995))

- Uses a different threshold at each resolution level of the wavelet decomposition of f (Donoho and Johnstone 1995).
- SureShrink is actually a hybrid threshold method
  - certain resolution levels can be too sparse.
  - revert `SureShrink` to using the universal threshold of `VisuShrink` at the resolution level in question.

```
y.sureshrink = hybrid.thresh(y.dwt, max.level = 4)
y.sureshrink.idwt = idwt(y.sureshrink)
plot(y.sureshrink.idwt,
  ylab = "Number of sunspots",
  xlab = "Months",
  main = "SureShrink-Soft Thresholding")
```


SureShrink−Soft Thresholding
Number of sunspots
Months
0
50
100
150
0
500
1000
1500
2000

Other use of wavelets

- Nonparametric density estimation (Vidakovic (1999)).
- Use for understanding the properties of time series and random processes.

Notes

- Can do thresholding without strong distributional assumptions on the errors using cross-validation (Nason 1996).
- Practical, simultaneous confidence bands for wavelet estimators are not available (Wasserman 2006).
- Standard wavelet basis functions are not invariant to translation and rotations.
  - Recent work by (Mallat 2012) and (Bruna and Mallat 2013) extend wavelets to handle these kind of invariances.
  - Promising new direction for the theory of convolutional neural network.

References for this lecture

HWC Chapter 13 (Wavelets)

W Chapter 9

Bruna, Joan, and Stéphane Mallat. 2013. "Invariant Scattering Convolution Networks." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (8). IEEE: 1872–86.

Donoho, David L, and Iain M Johnstone. 1995. "Adapting to Unknown Smoothness via Wavelet Shrinkage." *Journal of the American Statistical Association* 90 (432). Taylor & Francis Group: 1200–1224.

Donoho, David L, and Jain M Johnstone. 1994. "Ideal Spatial Adaptation by Wavelet Shrinkage." *Biometrika* 81 (3). Oxford University Press: 425–55.

Mallat, Stéphane. 2012. "Group Invariant Scattering." *Communications on Pure and Applied Mathematics* 65 (10). Wiley Online Library: 1331–98.