# `WiSEBoot` Vignette

Megan Heyman `heyma029@umn.edu`

October 28, 2015

## 1 WiSE Bootstrap Model Selection

Wild Scale-Enhanced (WiSE) bootstrap is a variant of the wild bootstrap where the model residuals are multiplied by an additional scaling factor. Theoretical details of the general WiSE bootstrap methodology may be found in (Chatterjee, 2015). This package is an implementation of the WiSE bootstrap for a specific case of the partial linear model. Namely, given an equally-spaced time series of length $T = 2^J$, $J \in \mathbb{I}^+$, we assume that a partial linear model adequately describes the time series.

$$\boldsymbol{Y}(\boldsymbol{t}) = \gamma_0 \mathbf{1} + \gamma_1 \boldsymbol{t} + \boldsymbol{\mu}(\boldsymbol{t}) + \boldsymbol{e}(\boldsymbol{t}) \tag{1}$$

$\boldsymbol{Y} \in \mathbb{R}^T$ is the observed data, $\boldsymbol{t}$ is a vector time indices, and $\boldsymbol{\mu}(\boldsymbol{t})$ is a general function (with some additional assumptions) in time. The nonparametric component, $\boldsymbol{\mu}(\boldsymbol{t})$, may contain the interesting cycles or signals within the data, so it is of interest to estimate. However, current methodology does not provide the ability to fully estimate this function. Thus, it is proposed that this function be approximated by a fixed basis. If $W$ is a $T-$ row matrix of a fixed wavelet basis for the discrete wavelet transform (DWT), and $\gamma$ contains the scaling and filter wavelet coefficients, then our approximate model is

$$\boldsymbol{Y}(\boldsymbol{t}) \approx \gamma_0 \mathbf{1} + \gamma_1 \boldsymbol{t} + W\boldsymbol{\gamma} + \boldsymbol{e}(\boldsymbol{t}) \tag{2}$$

Note, in many cases of the DWT, especially those implemented within `wavethresh`, the scaling coefficient is equivalent to $\gamma_0$. Thus, only the scaling coefficient or $\gamma_0$ should be estimated in any models where this occurs.

We propose that models of this type are useful descriptors of various time series, including climate model output. The quantity $W\boldsymbol{\gamma}$ is the approximation of $\boldsymbol{\mu(t)}$, which hypothetically contains 'interesting' signal within the data. Thus, the estimates of $\boldsymbol{\gamma}$ are of relevance to us. The parametric, linear component $(\gamma_0\mathbf{1}+\gamma_1\boldsymbol{t})$, would necessarily change by data application, but we claim it is adequate in the examples presented here. This parametric component is important to estimate, but may not contain interesting information for our application. Thus, in this approximation of the partial linear model, the set of population parameters $\gamma_0$, $\gamma_1$ and $\boldsymbol{\gamma}$ are estimated.

Estimating the full vector of wavelet coefficients, $\boldsymbol{\gamma}$, is problematic with current methodologies. However, it is typical that the wavelet filter coefficients are sparse, containing many zero or nearly zero-valued entries. The proposed methodology takes advantage of the sparsity and chooses a strong threshold criteria for the coefficients. With data of length $2^J$, filter wavelet coefficients exist for levels 0, 1, ..., $J-1$. Call $J_0$ the threshold. Then all coefficients occuring at levels greater than $J_0$ are set to 0. Essentially, our models assume all fine-level filter coefficients are 0-valued in expectation. Thresholding $\boldsymbol{\gamma}$ in this fashion decreases the number of parameters to estimate which implies consistent estimation is possible. Of course, this is subject to certain assumptions and conditions on the data and model, which are not discussed here.

Assume fixing the threshold, $J_0$, provides an adequate model. Then, we may estimate the population parameters $\gamma_0$, $\gamma_1$ and $\boldsymbol{\gamma}$ consistently. Using least squares estimation, $\hat{\gamma}_0$ and $\hat{\gamma}_1$ are obtained and residuals are defined as $\boldsymbol{r(t)} = \boldsymbol{Y(t)} - \hat{\gamma}_0\mathbf{1} - \hat{\gamma}_1\boldsymbol{t}$. The DWT is performed on $\boldsymbol{r(t)}$ to find a length-$T$ vector of wavelet coefficients. The threshold is then applied and $\boldsymbol{\gamma}$ is estimated by $\hat{\boldsymbol{\gamma}}_{J_0}$. Wavelet residauls are defined by $\boldsymbol{r_w(t)} = \boldsymbol{r(t)} - W\hat{\boldsymbol{\gamma}}_{J_0}$.

The discussion thus far has concerned parameter estimation. It is usually of interest to also estimate the error associated with the parameters in the model so that inference is possible. Although all of the assumptions of 2 have not been stated in this document, it is important to note that the following bootstrap methodology does *not* require that the errors, $\boldsymbol{e(t)}$, be homoscedastic or come from a normal distribution. The WiSE bootstrap is an excellent tool for estimation in situations where the typical, strong assumptions over models do not hold. Not only do we obtain consistent estimates of the parameters, but also their errors.

To perform the WiSE bootstrap for model 2 with set threshold $J_0$, create bootstrap series for $b = 1, 2, ..., B$

$$\boldsymbol{Y_b(t)} = \hat{\gamma}_0\mathbf{1} + \hat{\gamma}_1\boldsymbol{t} + W\hat{\boldsymbol{\gamma}}_{J_0} + \tau U_b \boldsymbol{r_w(t)} \tag{3}$$

where $\tau$ is a scaling parameter such that $\tau^2/T \to 0$ and $\tau \to \infty$. Define $U_b = diag(u_1, u_2, ..., u_T)$ for $b = 1, ..., B$. The $u_i, i = 1, 2, ..., T$ are independent, mean 0, variance 1, finite $8^{th}$ moment random variables, which are also independent across each bootstrap sample. The $\boldsymbol{Y_b(t)}$ are used to find the bootstrap estimates $\hat{\gamma}_{0b}$, $\hat{\gamma}_{1b}$, and $\hat{\boldsymbol{\gamma}}_{J_0b}$. The set of $B$ parameter estimates allows the user to examine the variability in each parameter. The WiSE bootstrap also allows for estimation of the variance associated with the errors, but this is not implemented within `WiSEBoot`. All details of theoretical conditions and consistency results are contained in (Chatterjee, 2015).

This WiSE bootstrap re-sampling technique is novel, as it allows for a model selection while concurrently providing (asymptotically) consistent estimates of the model parameters (Chatterjee, 2015). All prior description sets the threshold level, $J_0$, at a fixed level. The `WiSEBoot` package provides an automatic process to select the wavelet coefficient threshold level. The choice of the threshold determines the number of wavelet coefficients to be estimated, and thus, a model selection. Technically, speaking the models considered for selection follow a sieve selection scheme (i.e. estimate ALL wavelet coefficients at any levels less than or equal to $J_0$).

The model selection process is conducted as follows. For a data series of length $2^J$, a WiSE bootstrap sample is created for thresholds of $J_0 \in \{0, 1, ..., J-2\}$ as well as setting all filter wavelet coefficients to 0. The selected model minimizes the mean of the mean squared-error between the bootstrap estimated model and the provided data. That is, for any $J_0$, we calculate the mean of the mean-squared error

$$\overline{MSE}_{J_0} = \frac{1}{TB}\sum_{b=1}^{B} < \boldsymbol{Y(t)} - \hat{\gamma}_{0b}\mathbf{1} - \hat{\gamma}_{1b}\boldsymbol{t} - W\hat{\boldsymbol{\gamma}}_{J_0b}, \quad \boldsymbol{Y(t)} - \hat{\gamma}_{0b}\mathbf{1} - \hat{\gamma}_{1b}\boldsymbol{t} - W\hat{\boldsymbol{\gamma}}_{J_0b} >$$

where $< \boldsymbol{x}, \boldsymbol{y} >$ is defined as the inner product of vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. The selected wavelet coefficient threshold in the model is $J_0^*$ where

$$\overline{MSE}_{J_0^*} = \min_j \overline{MSE}_j$$

## 2 WiSE Bootstrap Model Selection: Simulation Example

To demonstrate the wavelet coefficient threshold selection, this example uses some simulated wavelet signals. What may be considered as a population-level signal is contained in `SimulatedSmoothSeries`, which is a matrix.

```
data(SimulatedSmoothSeries)
dim(SimulatedSmoothSeries)
```

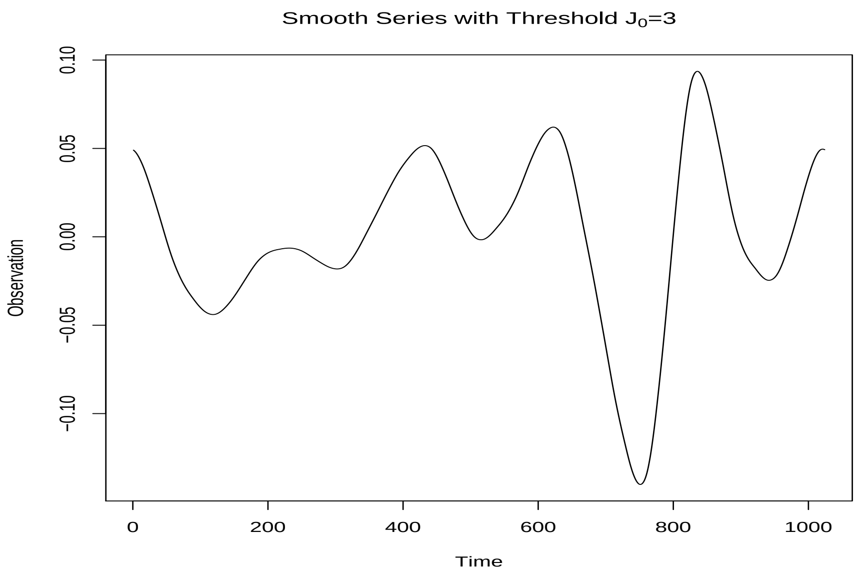
```
## [1] 1024    9
```

```
SimulatedSmoothSeries[1:3, ]
```

```
##              J0.0       J0.1        J0.2       J0.3        J0.4
## [1,] -0.009340405 0.01674716 0.004249168 0.04902028 -0.02750788
## [2,] -0.009343208 0.01676378 0.003836796 0.04871207 -0.02514671
## [3,] -0.009345534 0.01677582 0.003428975 0.04834426 -0.02252105
##              J0.5      J0.6      J0.7       J0.8
## [1,] -0.050660410 0.1438138 0.2153899 0.40223065
## [2,] -0.028013936 0.1572082 0.2122281 0.34668205
## [3,] -0.005142542 0.1555804 0.1611787 0.09866474
```

All signals in this matrix were generated using the `wavethresh` package with a `"DaubLeAsymm"` (Daubechies Least Asymmetric) wavelet with `8` vanishing moments and a `"periodic"` boundary condition. The column names of `SimulatedSmoothSeries` indicate the true wavelet coefficient threshold level. Let's consider the $4^{th}$ column. The wavelet coefficients were thresholded at the $J_0 = 3$ level. Thus, all finer-level coefficients (levels 4,5,6,7,8, and 9) were set to 0, to create this smooth signal.

Here is a plot of the signal in the $4^{th}$ column:

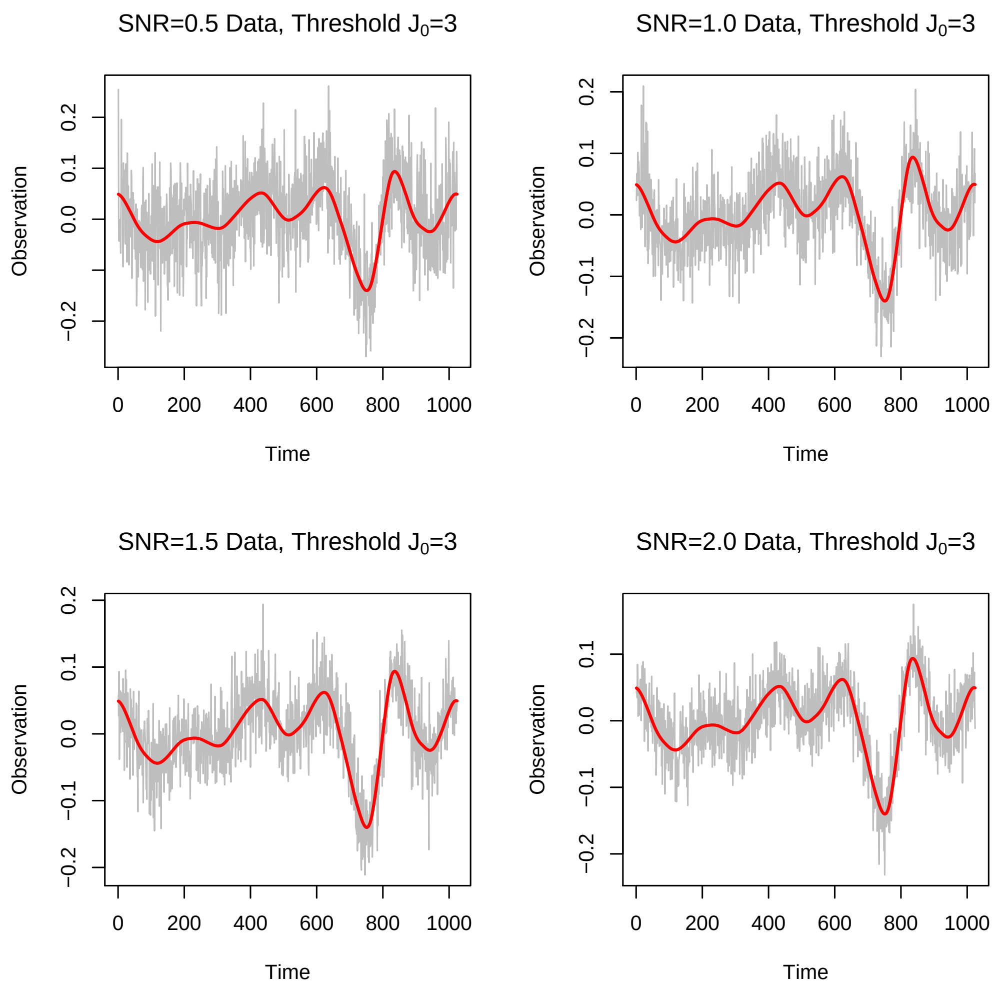


If data were observed from a population signal such as this, the data would be noisy. The matrices `SimulatedSNR0.5Series`, `SimulatedSNR1.0Series`, `SimulatedSNR1.5Series`, and `SimulatedSNR2.0Series` contain data series with population signals from `SimulatedSmoothSeries` and varying levels of added noise. Defining the signal-to-noise ratio (SNR) as

$$\text{SNR}=(\text{Variance of Signal})/(\text{Variance of Noise}),$$

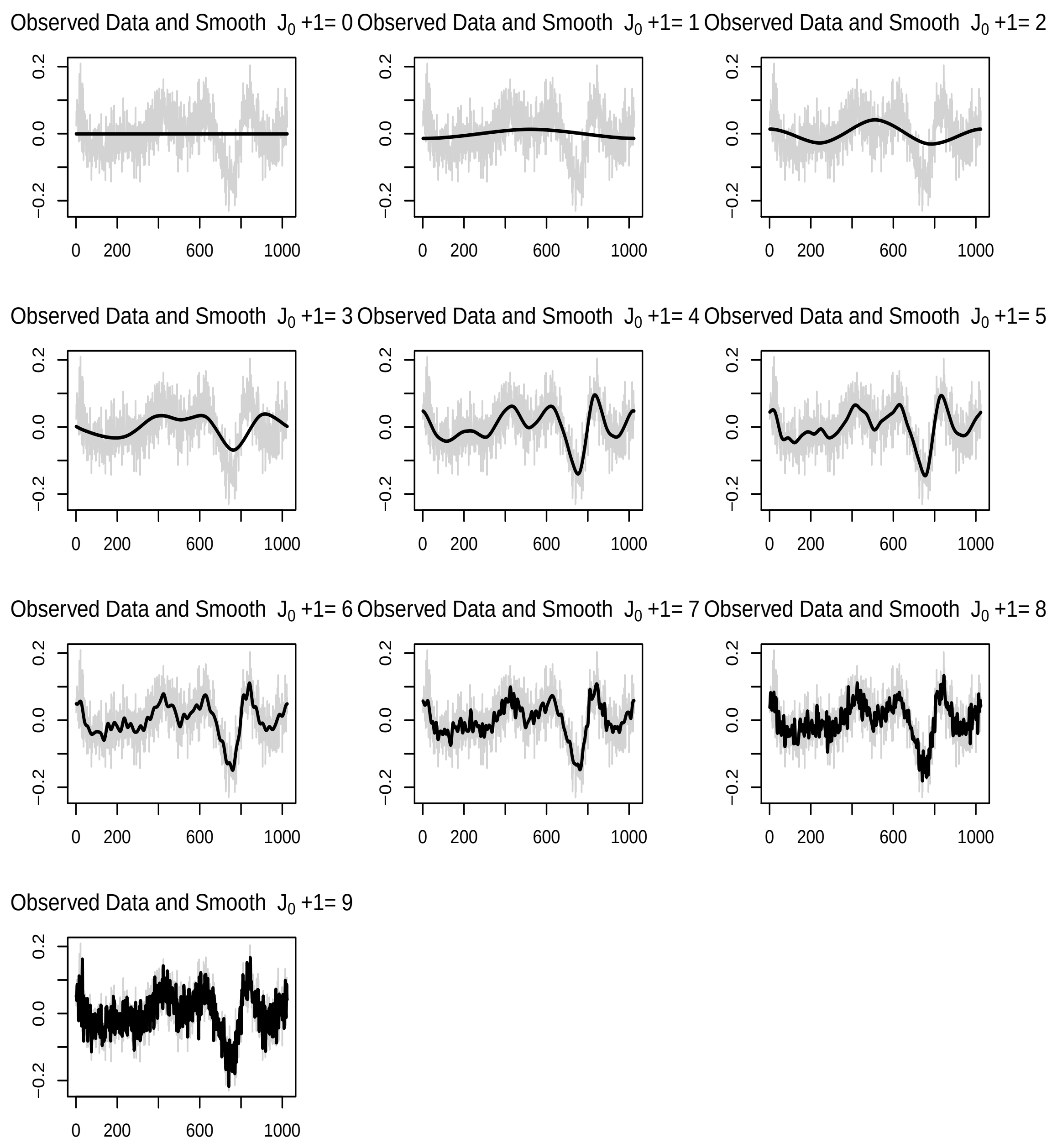
these matrices contain data series with SNR=0.5, 1.0, 1.5 and 2.0. It seems reasonable to assume that the signal would be more apparent in the SNR=2.0 series than the SNR=0.5 series.

Here is a look at the 'noisy data':



We can see that the true signal is fairly clear in each of these data series, but the lower SNR series are noisier. Typically, the analyst would not know the true signal threshold level. The `smoothTimeSeries` function allows for a quick visualization of all possible wavelet coefficient threshold levels. This is especially useful if the analyst would like to manually choose a wavelet coefficient threshold level. Using the SNR=1.0 series, we can demonstrate this capability. Note, the wavelet settings for these smooths generated below exactly match the true signals from `SimulatedSmoothSeries`. The user could change the wavelet family, filter, and boundary.

```
smoothPlot <- smoothTimeSeries(SimulatedSNR1.0Series[ ,4], plot="all")
```



Without knowing the truth, an analyst may guess that the wavelet threshold level could be any of $J_0 \in \{2,3,4\}$. We can use the `WiSEBoot` function to remove the guess-work and automatically choose a wavelet threshold. As discussed in Section 1, the model which achieves the minimum of the mean-squared error is selected. Let's leave the default wavelet settings again (which exactly match the true signal wavelet here) and look at the MSE criteria for B=`R=10` bootstrap samples. Typically, a larger number of bootstrap samples is used, as these may be generated in parallel. We choose a lower number in the vignette for the sake of time.

```
set.seed(1414)
SNR10Boot <- WiSEBoot(SimulatedSNR1.0Series[ ,4], R=10)
SNR10Boot$MSECriteria
```

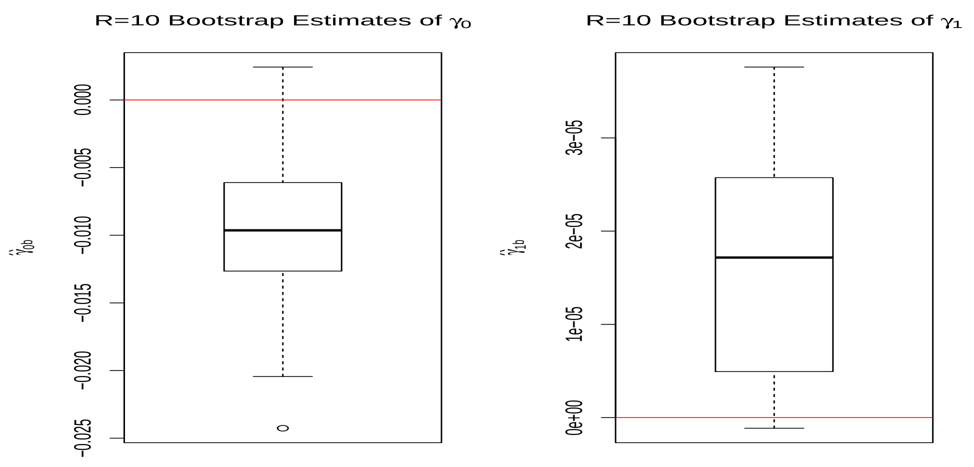
```
##       J0plusOne   meanOfMSE
##  [1,]         9 0.005253337
##  [2,]         8 0.004647180
##  [3,]         7 0.003526287
##  [4,]         6 0.002935839
##  [5,]         5 0.002609077
##  [6,]         4 0.002394103
##  [7,]         3 0.003768089
##  [8,]         2 0.004040207
##  [9,]         1 0.004421682
## [10,]         0 0.004503563
```

Even in this small bootstrap sample, the correct model was selected. We can see at $J_0 + 1 = 4$ the mean of the MSE is minimized with a value of 0.00239. The reader should keep in mind that this is a simulation example, so a desirable result is not surprising.

Now, we have an automatically selected model with threshold $J_0 = 3$. The output `WiSEBoot` object, called `SNR10Boot`, contains not only the model selection, but also the estimated parameters from each bootstrap sample in the selected model. Thus, the distributions of the $\gamma_0$, $\gamma_1$ and $\gamma$ parameters may be examined by visualizing the bootstrap estimates.

The boxplots below show the empirical distributions of the parametric linear parameter estimates $(\hat{\gamma}_{0b}, \hat{\gamma}_{1b})$. Because the data was simulated, the population parameters are known to be $\gamma_0 = 0, \gamma_1 = 0$.

```
par(mfrow=c(1,2))
boxplot(SNR10Boot$BootIntercept,
        main=expression(paste("R=10 Bootstrap Estimates of ", gamma[0])),
        ylab=expression(hat(gamma)[0][b]))
abline(h=0, col="red")
abline(h=)
boxplot(SNR10Boot$BootSlope,
        main=expression(paste("R=10 Bootstrap Estimates of ", gamma[1])),
        ylab=expression(hat(gamma)[1][b]))
abline(h=0, col="red")
```

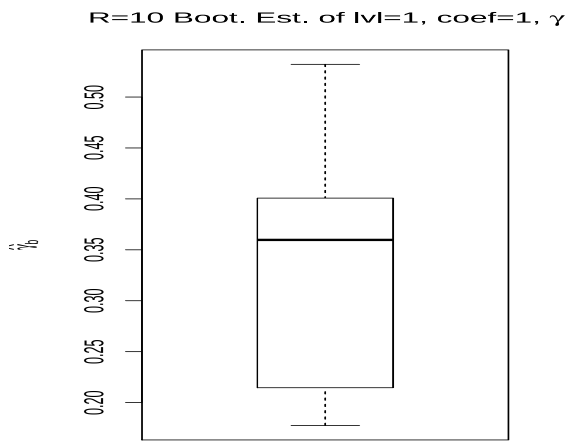


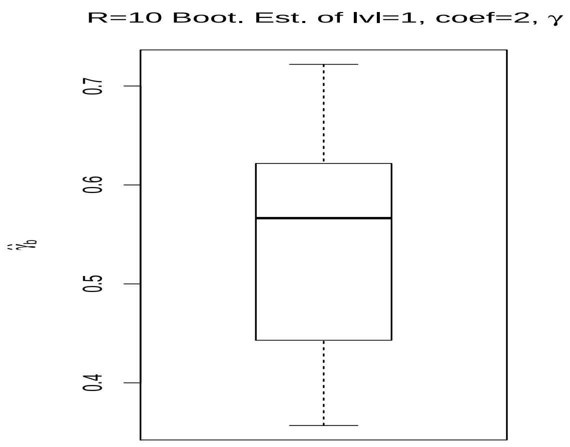
As always in any type of bootstrap, the bootstrap distribution will be centered around the original estimate from the data. We can see that these boxplots aren't exactly centered at 0 because of this centering issue associated with the bootstrap. (Keep in mind there are only `R=10` bootstrap samples too.)

Likewise, any individual non-thresholded filter wavelet coefficient may be visualized. Below are the bootstrap distributions of the level=1 filter coefficients.

```
par(mfrow=c(1,2))
boxplot(SNR10Boot$BootWavelet[,3],
        main=expression(paste("R=10 Boot. Est. of lvl=1, coef=1, ", gamma)),
        ylab=expression(hat(gamma)[b]))
```

```
boxplot(SNR10Boot$BootWavelet[,4],
        main=expression(paste("R=10 Boot. Est. of lvl=1, coef=2, ", gamma)),
        ylab=expression(hat(gamma)[b]))
```





## 3 WiSE Bootstrap Hypothesis Test

Next we provide a brief introduction to a hypothesis test of the wavelet coefficients from two data series of equal lengths. This methodology is discussed in detail in (Braverman, 2015). Here, the theoretical discussion is meant to give users of the `WiSEHypothesisTest` and `WiSEConfidenceRegion` a general idea of what mathematics are running in the background.

Consider two equally-spaced data series of length $T = 2^J, J \in \mathbb{I}^+$. Each data series may be modeled by eqn. 1 and approximated by eqn. 2. That is,

$$\boldsymbol{Y(t)} = \gamma_{y0}\mathbf{1} + \gamma_{y1}\boldsymbol{t} + W\boldsymbol{\gamma}_y + \boldsymbol{e(t)}_y$$

$$\boldsymbol{X(t)} = \gamma_{x0}\mathbf{1} + \gamma_{x1}\boldsymbol{t} + W\boldsymbol{\gamma}_x + \boldsymbol{e(t)}_x$$

Notice, the same wavelet basis, $W$, is used in each approximation, but separate parameters are allowed. Users of the `WiSEHypothesisTest` function may wish compare the ‘interesting’ signals within these two series. Recalling the discussion from Section 1, we may decide that the ‘interesting’ signals are $\boldsymbol{\mu(t)}_y$ and $\boldsymbol{\mu(t)}_x$. With the wavelet basis approximation, the user tests the linear relationship between the two sets of wavelet coefficients: $\boldsymbol{\gamma}_y = \alpha\mathbf{1} + \beta\boldsymbol{\gamma}_x$. The null hypothesis in `WiSEHypothesisTest` is

$$H_0 : \alpha = m, \beta = n, \quad m, n \in \mathbb{R}$$

In (Braverman, 2015), values of $m = 0$ and $n = 1$ are tested. If the null hypothesis in this specific scenario is not rejected, then we may say that there is not sufficient evidence to conclude that the signal within the climate model output does not match the observed climate. The algorithm to generate the WiSE bootstrap sample changes slightly so that we are sampling under the null hypothesis. Please see (Braverman, 2015) for the details.

## 4 WiSE Bootstrap for Hypothesis Testing: Climate Model Signals

An example analysis of climate model data is presented here. The full analysis of the climate models at some specific grid-cells is available in (Braverman, 2015). Within the `WiSEBoot` package, two sets of climate model outputs and observed climate are available. Here, we will look at the data in `CM20N20S60E`.

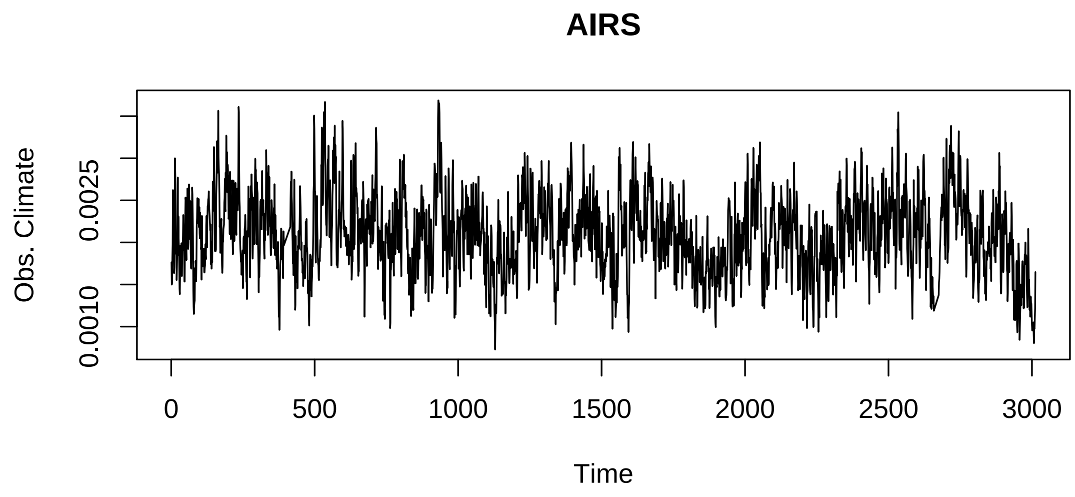
```
data(CM20N20S60E)
CM20N20S60E[1:3,]
```

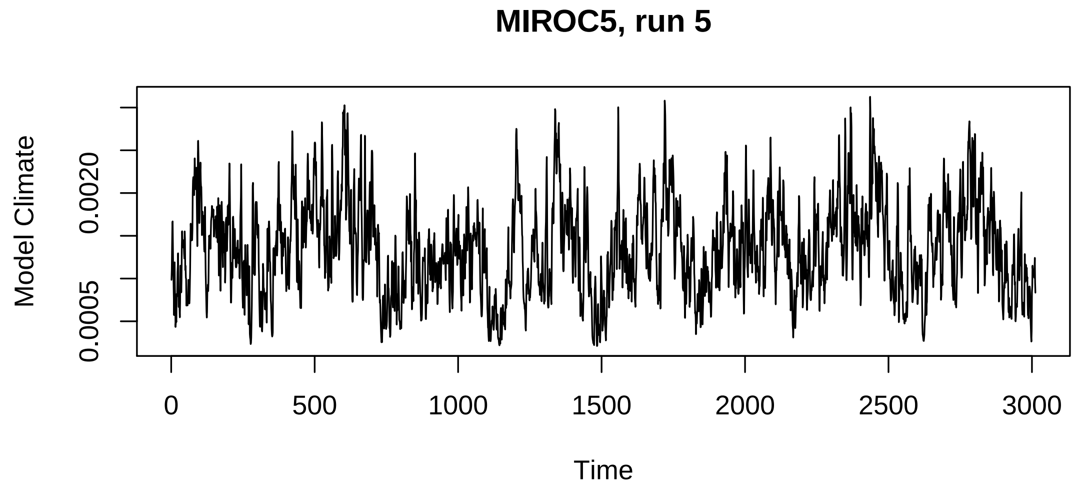
```
##                   AIRS    IPSLRun1    IPSLRun2     IPSLRun3    IPSLRun4
## 2002-10-01 0.001764203 0.001844154 0.002112894 0.0006225206 0.001579949
## 2002-10-02 0.001498882 0.001700833 0.001708422 0.0006596091 0.001716239
## 2002-10-03 0.001522536 0.001539518 0.001419565 0.0008227325 0.001948907
##             MIROC5Run1   MIROC5Run2   MIROC5Run3  MIROC5Run4  MIROC5Run5
## 2002-10-01 0.001653560 0.0017041005 0.0010692392 0.001497587 0.000984715
## 2002-10-02 0.001490860 0.0011596567 0.0010018310 0.001331268 0.001106074
## 2002-10-03 0.001315217 0.0008881064 0.0008363572 0.001092815 0.001381810
##             MIROC5Run6
## 2002-10-01 0.001133758
## 2002-10-02 0.001047236
## 2002-10-03 0.001029497
```

This data matrix contains data from AIRS (what we call 'observed climate'), 4 runs of the IPSL model, and 6 runs of the MIROC5 model. Each of the model runs is obtained by choosing different starting parameters. Specifically, this matrix contains a set of specific humidity observations/outputs between 20N and 20S at 60E and an altitude of 500 hPa. Observations are daily, from October 1, 2002 to December 29, 2010.

In this analysis, we'll compare the signals in AIRS and MIROC5, run 5. Here are some plots of the raw data.

```
par(mfrow=c(2,1))
plot.ts(CM20N20S60E[,1], main="AIRS", ylab="Obs. Climate")
plot.ts(CM20N20S60E[,10], main="MIROC5, run 5", ylab="Model Climate")
```





Before it is possible to use the WiSE bootstrap methodology with a wavelet basis approximation, the data must be of length $T = 2^J$ for a positive integer, $J$. The raw data contains 3012 observations. Thus, we begin by using the `padMatrix` function to lengthen each series simultaneously. Conversely, the user could truncate the data to the closest power of 2 in length, which may be more highly recommended when it is OK to exclude some of the data.

```r
pad60E <- padMatrix(CM20N20S60E)
dim(pad60E$xPad)


## [1] 4096   11
```

The code above uses the default options within `padMatrix` of padding at both sides of the data series by reflecting. The linear trend is not replaced (default) to the padded data matrix because we may easily input the estimated linear (parametric) parameters to the hypothesis testing function. This data is now of length $2^{12}$.

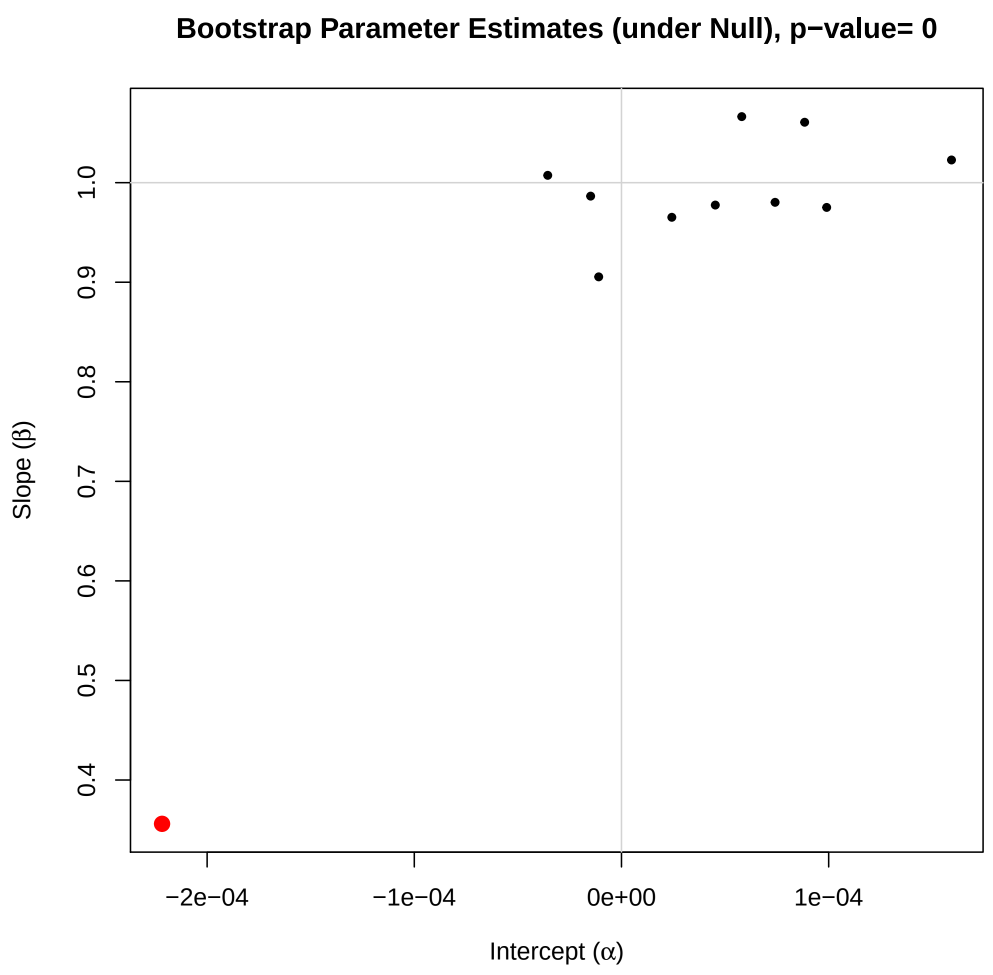
The hypothesis testing function requires that the user choose a wavelet coefficient threshold level. This may be done automatically with the `WiSEBoot` function first, by inputting both series as a 4096 x 2 matrix. Here, we choose to just set a threshold of $J_0 = 5$, as this corresponds to a cycle of 128 days.

Now that the data is of correct length, we may test the hypothesis

$$H_0 : \alpha = 0, \beta = 1$$

(i.e. the climate model signal matches the observed climate). Our 'X' series is AIRS and the 'Y' series is MIROC5, run 5. For demonstration purposes, we will choose to take `R=10` bootstrap samples. A higher number of samples is generally recommended.

```r
hypObj <- WiSEHypothesisTest(pad60E$xPad[,1], pad60E$xPad[,10], R=10, J0=5,
                             XParam=pad60E$linearParam[,1],
                             YParam=pad60E$linearParam[,10])
```



```
hypObj$AsymptoticPValue
```

```
##              [,1]
## [1,] 4.597697e-44
```

```
hypObj$BootstrapPValue
```

```
## [1] 0
```

The function calculates two p-values. The asymptotic p-value is based upon the distribution of Hotelling's $T^2$ and the test statistic utilizes the variance-covariance matrix from the bootstrap sample. The bootstrap p-value computes a Hotelling's $T^2$ test statistic for each bootstrap sample, and the quantile of the data-based Hotelling's $T^2$ is computed from the bootstrap sample. A plot of the bootstrap sample and observed data parameter estimates is also generated optionally.

We can see that both the asymptotic and bootstrap p-values indicate that the null hypothesis should be rejected. This is not surprising, especially after examining the generated plot. The $\hat{\alpha}$ and $\hat{\beta}$ estimated from the data (red point) is clearly outside of the cloud of bootstrap sample points (black). The gray vertical and horizontal lines represent the parameter values under the null hypothesis.

## Acknowledgments

Thank you to all of the collaborators on this project. Specifically, Amy Braverman (NASA-JPL) and Matthew Gunson (NASA-JPL) provided the re-gridded climate data within `CM20N20S60E` and `CM20N20S150W`. My thesis advisor, Snigdhansu Chatterjee (University of Minnesota), not only helped test this package's code, but also gave me much feedback and advice. Noel Cressie (Univeristy of Wollongong) and Stephen LeRoy (Harvard) also provided useful feedback and innovation to this project.

## References

Chatterjee, S. and Heyman, M. "WiSE Bootstrap for model selection", 2015 (in progress).

Braverman, A., Cressie, N., Chatterjee, S., et al. "Probabilistic Climate Model Evaluation", 2015 (in progress).