# PULSAR MODE CHANGE DETECTION

**IRES Report**

**August 19, 2019**

Min Young Kim
University of Washington Seattle
Advisor: Bhal Chandra Joshi

# Contents

## 0.1 Pulsars

Pulsars are the remnants of a star that has gone supernova at the end of its life cycle. It leaves behind an extremely dense neutron star, with strong gravitational and magnetic fields. The rotational axis and magnetic axis are not necessarily aligned. When electrons flow along the magnetic field lines at the polar cap, it acts as a source for radio emission. This radio signal is distributed over a range of observing frequencies, and it is thought that higher frequency signals are typically generated closer to the surface of the pulsar, in what is known as radius-to-frequency mapping.

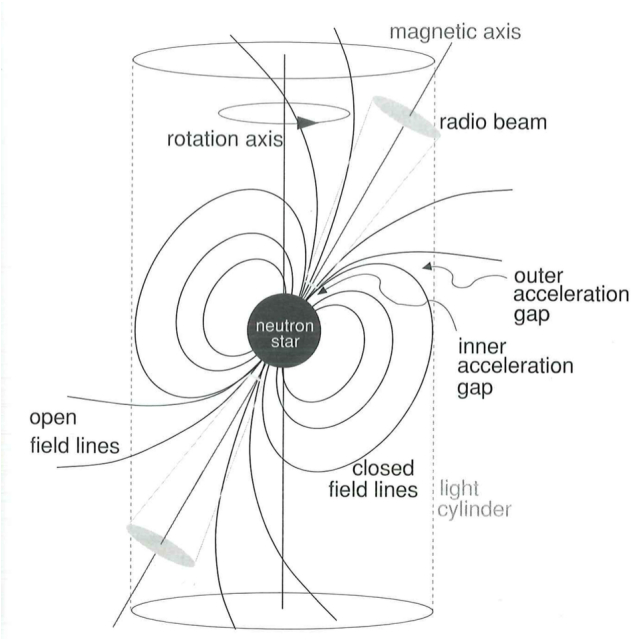



**Figure 1:** A simplified pulsar model. Taken from Handbook

Pulsar signals are extremely faint, hence a commonly used unit of flux density is the Jansky. $1Jy = 10^{-26} Wm^{-2} Hz^{-1}$. The median flux density for pulsars is around $0.8mJy$. Hence, a technique called folding is employed to obtain an average pulsar profile. One first identifies the period of the pulsar in the Fourier frequency domain. Then, given a continuous "time series" of pulsar signals, one cuts the data into chunks of length equal to the pulsar period, and sums them up together. Because random Gaussian noise/white noise (WN) has a zero mean, for $N$ folds of data chunks, the signal increases proportional to $N$, while WN increase proportional to $\sqrt{N}$. The end result is an average pulsar profile with a much higher signal to noise ratio (SNR). A more detailed guide to folding can be found in Pulsar Handbook 7.1.

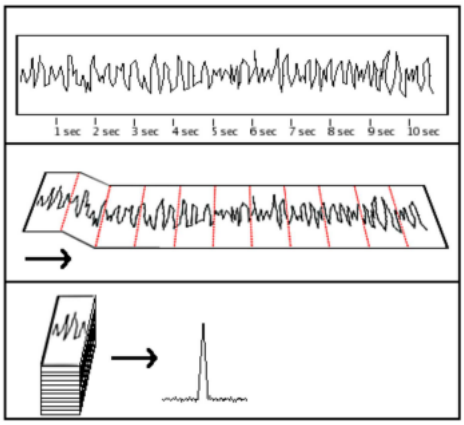



**Figure 2:** Folding of a pulsar signal to increase SNR.

## 0.2 Radio Astronomy

How are these signals detected? Typically, pulsar signals are distributed across a range of radio frequencies. The most basic method is the use of a filter bank. A filter bank has a large number of narrow band filters, essentially dividing the range of radio frequencies observed into discrete bins. Each band filter ideally only measures flux corresponding to its own observing frequency. Thus, a telescope receiver is able to measure radio signals at each specified frequency alone.

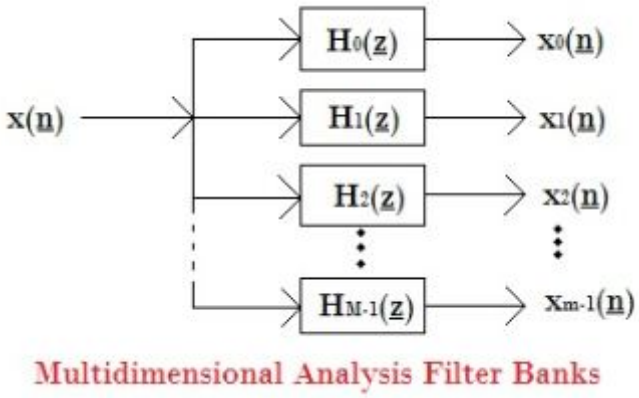



**Figure 3:** A diagram of a filter bank.

Radio telescopes generally have a paraboloid shape. Pulsar signals are so far away that they can be approximated as plane waves, and when such plane waves encounter a paraboloid reflector, the waves are focused at the focal spot of the dish. The cross

section of the wave power at the focal point is not a point, but produces a single slit interference pattern. The width of the focal spot is typical $\frac{F\lambda}{D}$, where $F$ is the focal length, $\lambda$ is the wavelength, and $D$ is the diameter of the aperture. It is only logical that one would want to increase the diameter of the aperture to increase the focal spot, or resolution. This leads to the idea of a number of radio telescopes distributed over a large area, effectively increasing its diameter. One such example is the square kilometer array. At the Giant Meter Radio Telescope (GMRT) in India, 30 radio telescopes are distributed in a Y shape, giving an effective total diameter of a 25km dish. Unlike the Greenbank telescope in West Virginia, which has 2000 metal reflectors which are motor controlled to compensate for deformations due to wind, gravity, and temperature, GMRT uses a wire mesh with varying densities as a function of radius of the dish. The varying density of the mesh corresponds to a different frequency band. The use of wire mesh also means less tension required to compensate for gravitational stress.


**Figure 4:** Location of all 30 GMRT telescopes.

## 0.3 J0332+5434

The goal of the project is to detect and characterize mode changing phenomenon of pulsars. Mode changing is a phenomenon in which a pulsar average profile changes for some time into its abnormal mode, then reverts back to its original or normal mode. Typically, a pulsar spends majority of its time in its normal mode. This was first

observed in B1937+21. The pulsar whose data was used in this report, J0332+3435, is a regular pulsar with a period of .714 seconds, and a dispersion measure of 26.76 $cm^3$ $pc^{-1}$. It is one of the brightest pulsars in the northern hemisphere, and its high SNR makes it ideal to test different mode detection methods on. Its normal mode consists primarily of 3 peaks, the central being the tallest. It's abnormal mode primarily features a decrease in amplitude of the third peak.

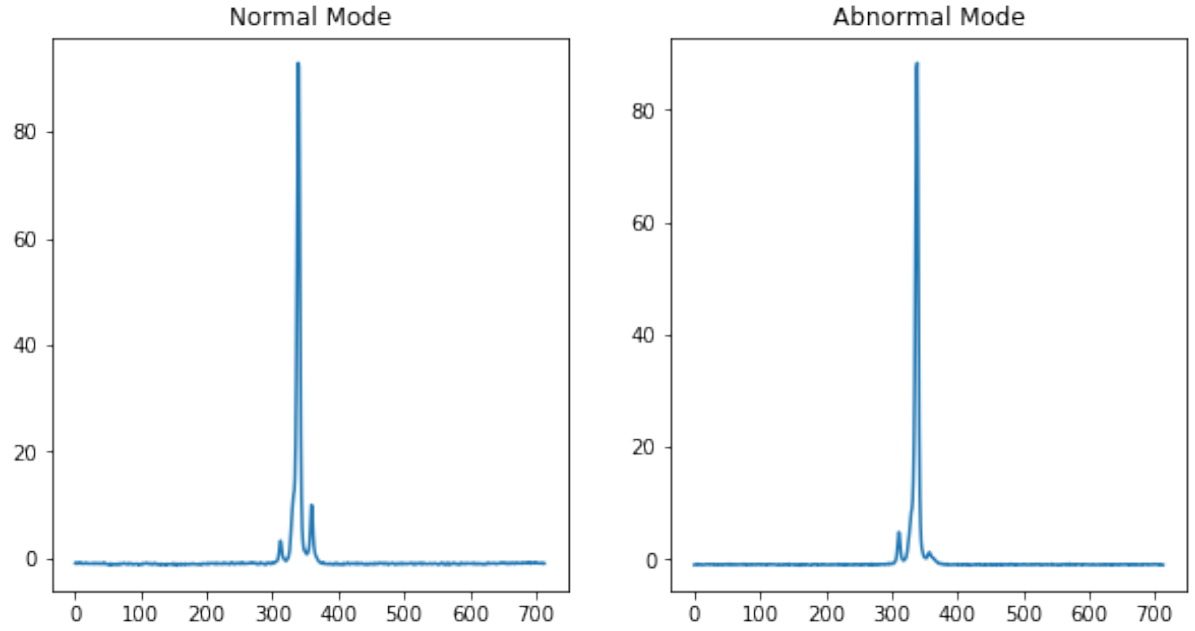



**Figure 5:** Pulse profiles of J0332+5434. There is a clear change in profile shape, specifically the decrease in amplitude of the third peak. [Python 'matplotlib']

The data used was gathered from Ooty telescope in South India. After removing noisy data which did not show any pulsar signals, there remained 62 average profiles to work with. Each profile had 1250 pulses folded. The profiles had varying amounts of white noise. Each profile had a length of 714 bins, corresponding to the pulsar period of .714 seconds. Hence, the sampling period was 1ms.

## 0.4 Data Preparation

First, the profiles needed to be aligned in phase. While one can arbitrarily choose a standard profile to align the remaining 61 profiles to, it is crucial to choose the standard profile to have its peaks occur near the middle of the observed bins. The reason is

explained in Section 0.6. This aligning of phase is done in the Fourier frequency domain, with the use of Fourier Shift Theorem.

First, the Discrete Fourier Transform (DFT) of a signal $x$ is defined by

$$X(\omega_k) \equiv \sum_{n=0}^{N-1} x(t_n) e^{-i\omega_k t_n}, \quad k = 0, 1, 2, ..., N-1$$

where
$x(t_n) \equiv$ input signal amplitude at time $t_n$,
$T \equiv$ sampling interval,
$t_n \equiv nT =$ n$^{th}$ sampling instant, $n \in \mathbb{N}$,
$X(\omega_k) \equiv$ spectrum of x at frequency $\omega_k$,
$\omega_k \equiv k\Omega = \frac{k2\pi f_s}{N} = \frac{k2\pi}{NT} =$ k$^{th}$ frequency sample,
$\Omega \equiv \frac{2\pi}{NT} =$ radian frequency sampling interval,
$f_s \equiv \frac{1}{T} =$ sampling rate (samples/time unit, $Hz$ if seconds, and
$N =$ number of time/frequency samples.

Likewise, the Inverse DFT is defined by

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{i2\pi nk/N}, \quad n = 0, 1, 2, ..., N-1$$

The Fourier Shift Theorem states,

$$x(n - \Delta) \leftrightarrow e^{-i\omega\Delta} X(\omega_k)$$

where $\omega_k \equiv \frac{2\pi k}{N}, \quad k = 0, 1, 2, ..., N-1,$
and the signal in time $x(n)$ is delayed by $\Delta$ *samples*, hence $\Delta \in \mathbb{Z}$.

In words, the Fourier Shift theorem can be interpreted as such: a shift in time domain by $\Delta$ corresponds to a multiplication by a linear phase term $e^{-i\omega_k\Delta}$ in the Fourier domain. Hence, given the profile data, one can calculate the time delay $\Delta$ in units of samples/bins against some reference profile, apply DFT to the profile, multiply by the linear phase factor $e^{-i\omega_k\Delta}$, and then take the inverse DFT. The end result should be a profile that is now aligned with some reference profile in phase.
How does one calculate $\Delta$? Ideally, one would use apply a root-finding algorithm known

as Brent's method to equations (A 7) and (A 8) to solve for $\tau$, and convert it to number of samples. However, due to the high SNR of J0332+5434 data, it was enough to simply calculate $\Delta$ by taking the difference in number of samples between two given profiles' maximum amplitude. For other pulsar profiles with a lower SNR, or a profile shape that doesn't have a single extreme peak, it would be best to implement Brent's method.

After aligning all the profiles against some reference profile, one needs to calibrate and normalize each profile. Calibration is done by subtracting the mean of each profile to itself. Normalization is done simply by dividing each profile by its integrated area.

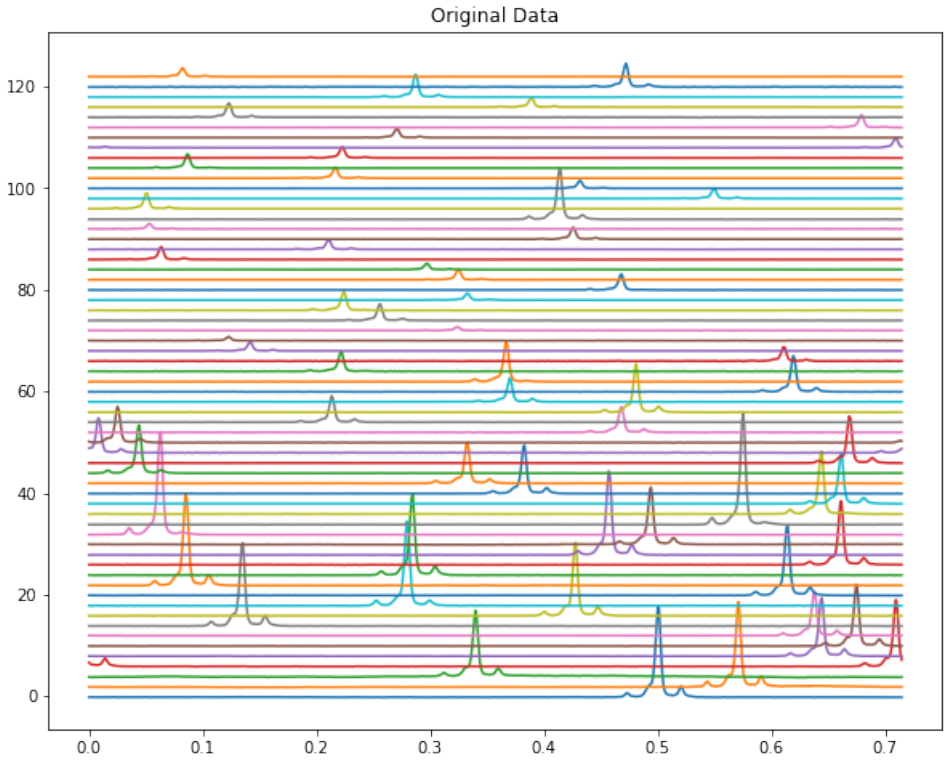



**Figure 6:** Original data of 62 profiles. [Python 'matplotlib']

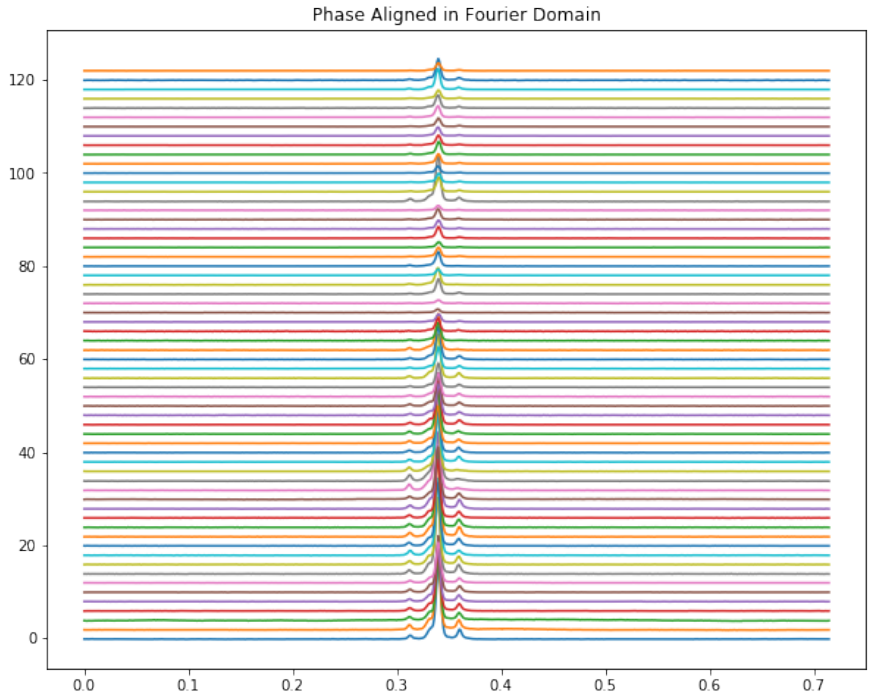



**Figure 7:** Phase aligned in Fourier domain. [Python 'matplotlib']

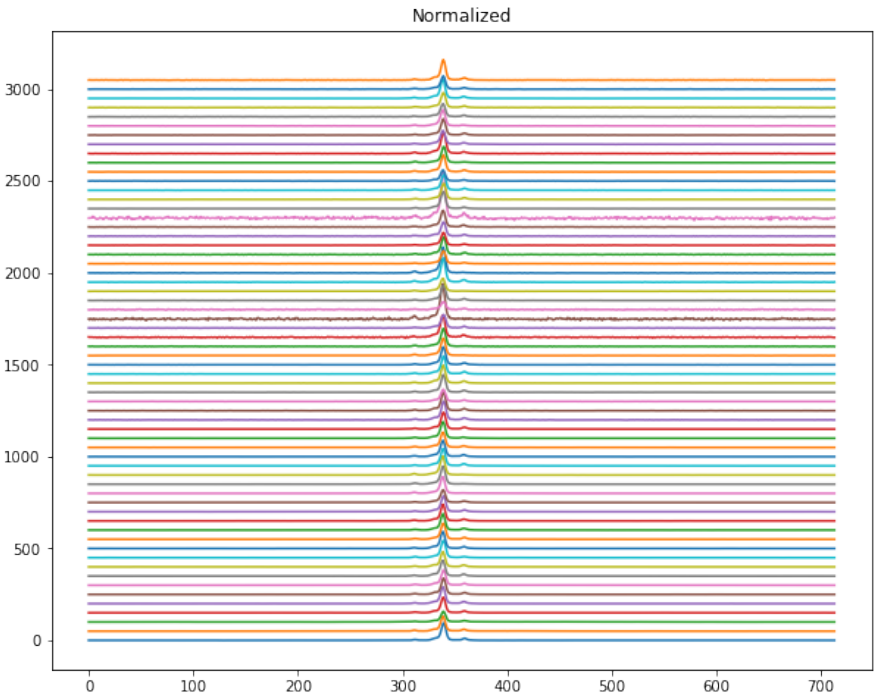



**Figure 8:** Finally, normalized profiles to 1.[Python 'matplotlib']

## 0.5 Gaussian Fitting

One of the most basic methods is Gaussian. One models a profile as a sum of Gaussian components, then for each profile produce a Gaussian fit. By observing how the amplitude of the fits vary over epoch, one can detect mode changes.

The Gaussian model used composed of five components. They were produced simply using the Gaussian probability density function

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \; e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Where $\sigma$ is the standard deviation, $\mu$ is the mean. It does take some trial and error to get correct parameters for the Gaussian components, as a single model might not adequately fit all the profiles.

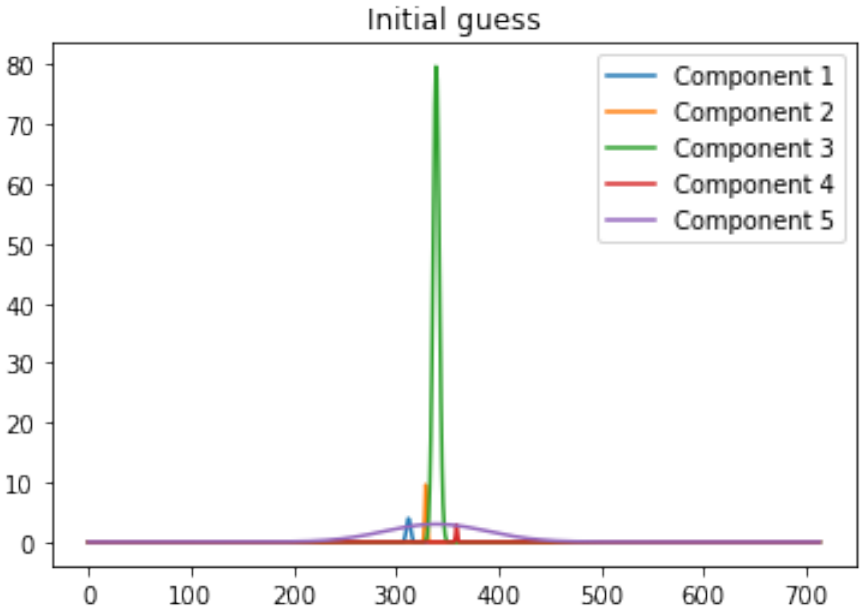



**Figure 9:** Initial guess of five Gaussian components. [Python 'matplotlib']

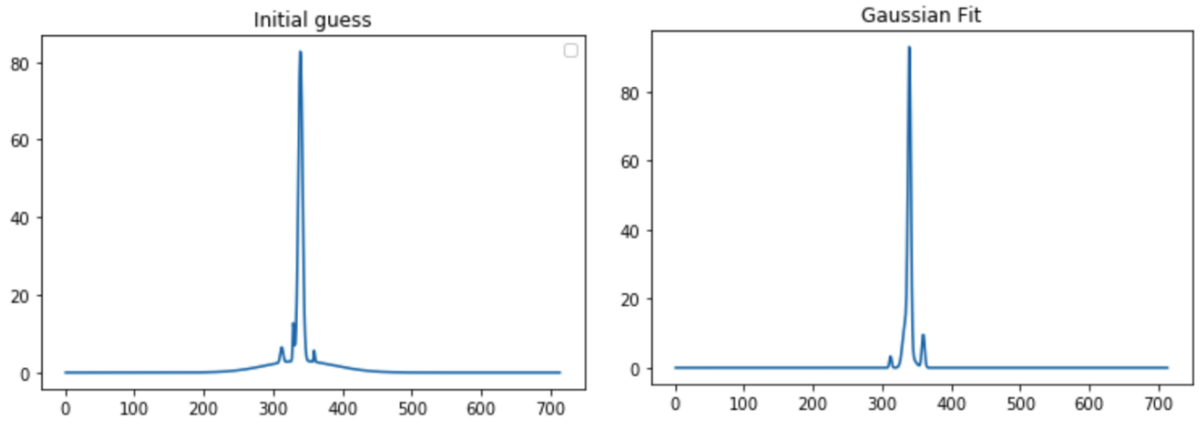



**Figure 10:** Gaussian model pre and post fit [Python 'matplotlib]

.

## 0.6 Kolmogorov-Smirnov Test

### 0.6.1 Discrete KS Test

The Kolmogorov-Smirnov (KS) test is a non-parametric, goodness-of-fit test. A one-sample KS test attempts to determine if a set of observed values are generated from some reference distribution. A two-sample KS test attempts to determine if two observed data sets come from the same distribution. In this project, we use a one-sample KS test. A continuous, one-sample KS test is defined as follows.

Let $F_0(x)$ be the hypothesized distribution, $F_{data}(x)$ be the empirical distribution of observed data. The test statistic used is

$$D = \sup_x |F_0(x) - F_{data}(x)|$$

where $\sup_x$ denotes the supremum. Once one obtains the D-statistic, one can calculate the corresponding p-value. The p-value is the probability of observing the given data under the assumption that $H_0$, the null hypothesis, is true. In this context, $H_0$ would be that $F_{data}(x)$ is sampled from the hypothesized distribution $F_0(x)$. The alternative hypothesis, $H_\alpha$, would be that given data does not come from the hypothesized distribution. Given some significance level $\alpha$, say $\alpha = .05$, if the p-value $< \alpha$, then we would reject $H_0$ and favor $H_\alpha$. Note that failure to reject $H_0$ does not necessarily imply

the data is drawn from the null distribution. It simply means there was not enough evidence to reject the $H_0$.

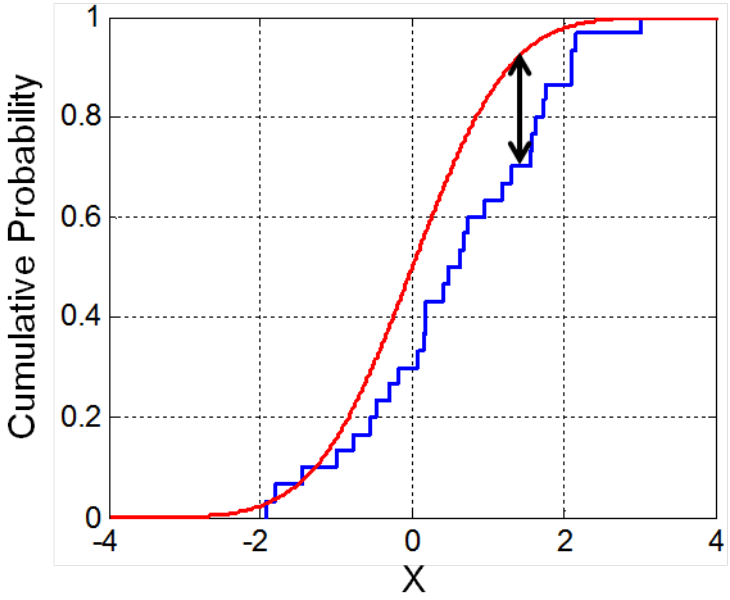



**Figure 11:** A one sample continuous KS test. Hypothetical CDF in red, empirical CDF in blue. The arrow denotes the D statistic at the given point.

### 0.6.2 Application to J0332+5434

First, we picked a reference profile, whose cumulative distribution function (CDF) will serve as our null distribution. The very first profile was chosen arbitrarily. Note that a KS test is more sensitive to deviations near the center of the distribution. This is why when phase aligning in the data preparation step, it was necessary to align all profiles such that the pulse signals occurred near the center. Once a have picked our null distribution, we simulate 1000 profiles by adding WN on top of the reference profile. Then, performing the KS test of the 1000 simulated profiles' CDF against the reference profile's CDF returns 1000 D-statistics, whose distribution is shown in the histogram. If we desire a significance level $\alpha_{ks}$ when performing the KS test on the real profiles, the critical D-statistic, $D_{crit}$ corresponding to $\alpha_{ks}$ will be equal to the $1-\alpha_{ks}$ th percentile of the simulated distribution. A significance level of $\alpha_{ks} = .05$ was used.

Next, we perform the KS test of the remaining profiles' CDF against the null distribution, and obtain D-statistics. The distribution of the D-statistics are shown. If a

given profile's D-statistic is larger than $D_{crit}$, then we conclude that that profile is significant at the $\alpha_{ks}$ level, and hence displays a mode different from that of the reference profile.

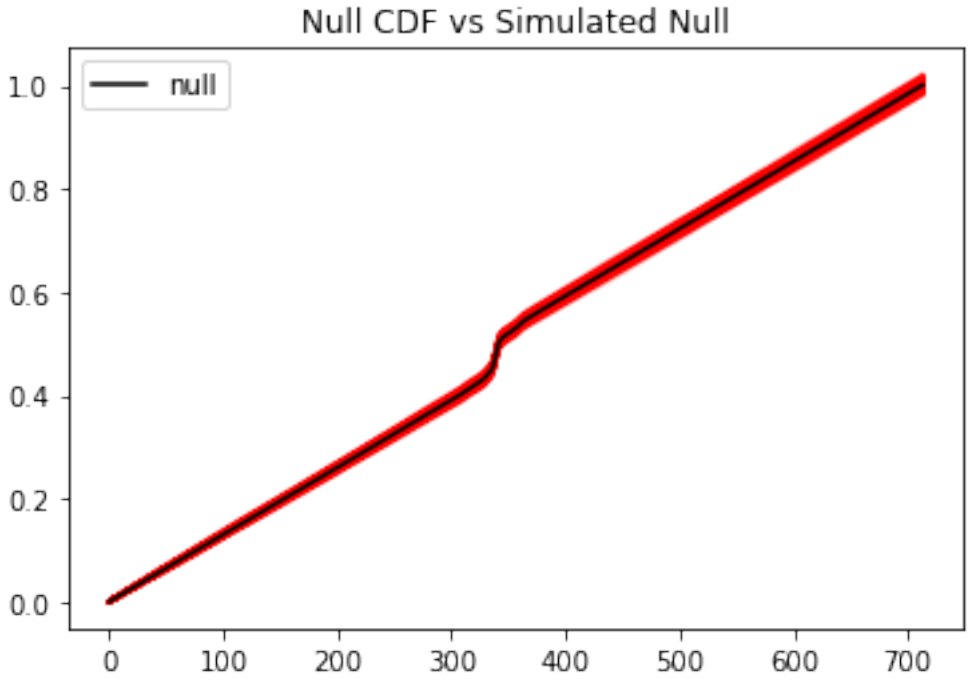



**Figure 12:** CDF of the null profile in black. CDFs of 1000 WN simulations in red. [Python 'matplotlib']

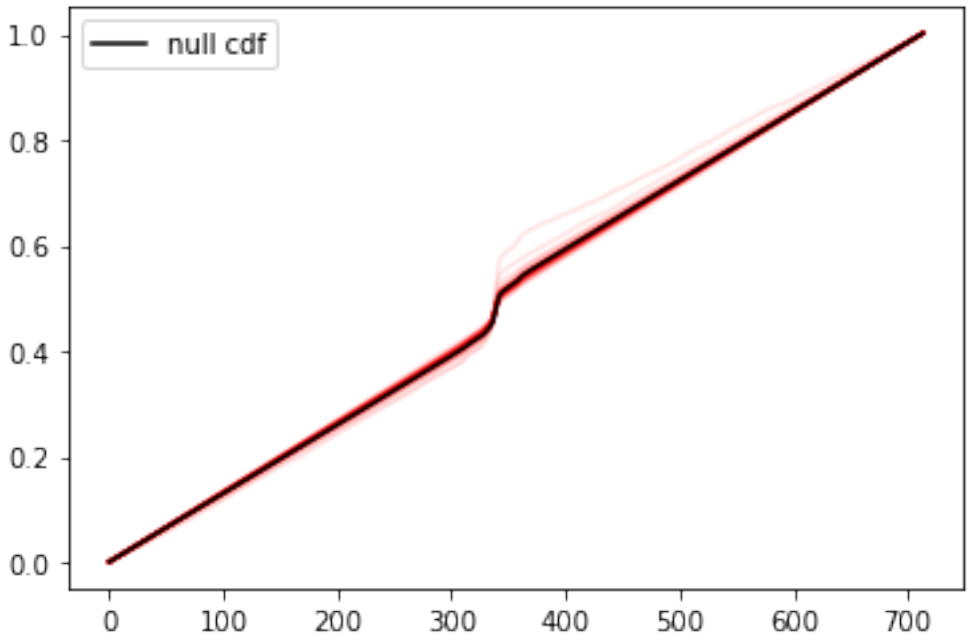



**Figure 13:** CDF of the null profile in black. CDFs of other 61 profiles in red. [Python 'matplotlib']

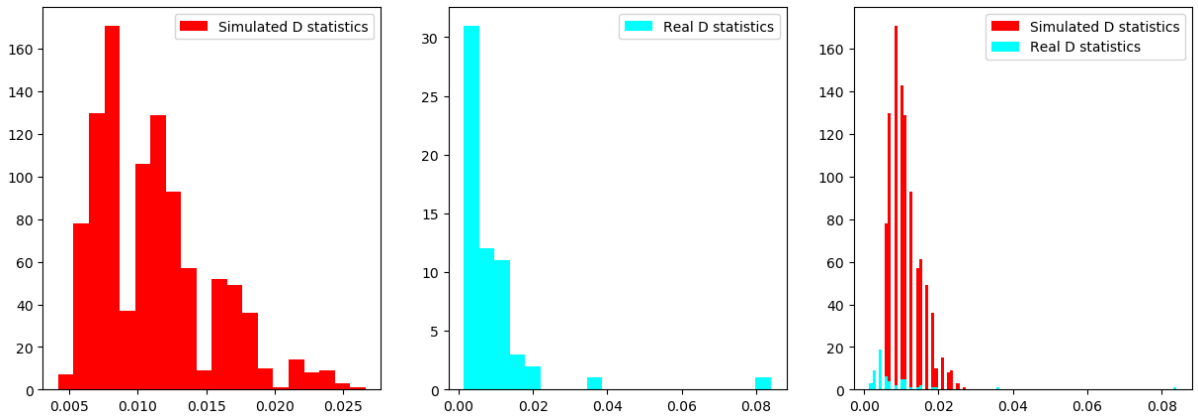



**Figure 14:** Distributions of D-statistics from (a) 1000 simulations (b) 61 profiles (c) The two distributions over plotted. [Python 'matplotlib']

One can then plot the mode changes according to the KS test as a function of epoch to visually identify a potential characteristic time of mode changes.

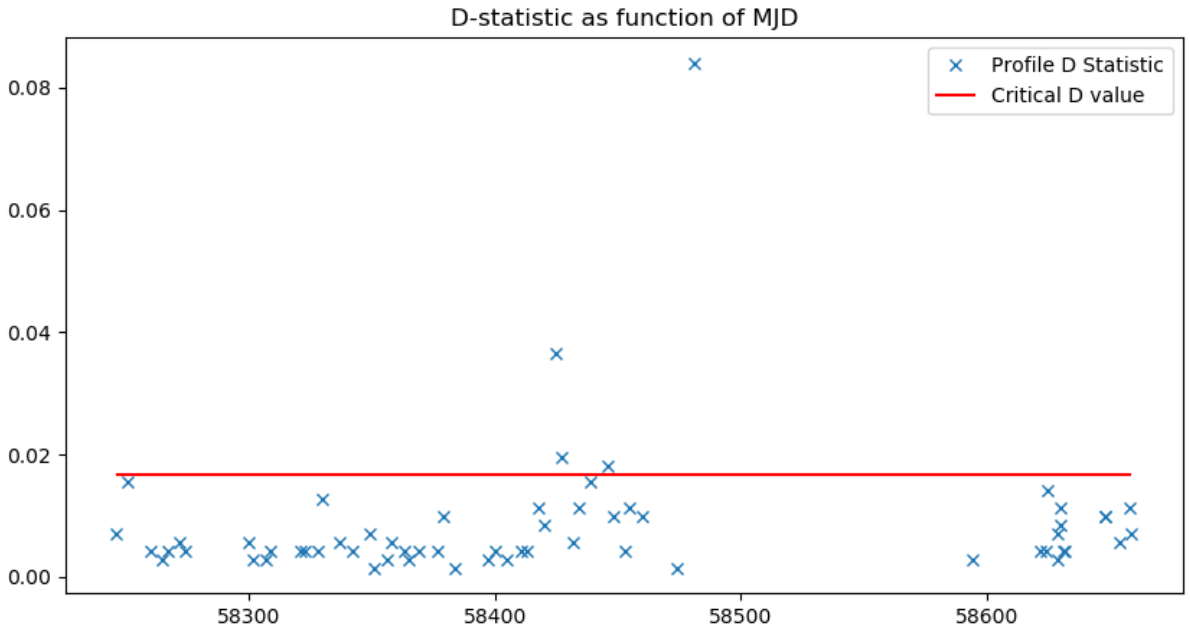



**Figure 15:** D-statistics of profiles plotted as a function of epoch. Red line indicates $D_{crit}$. Profiles with D-statistics above this line is significant at $\alpha ks = .05$. [Python 'matplotlib']

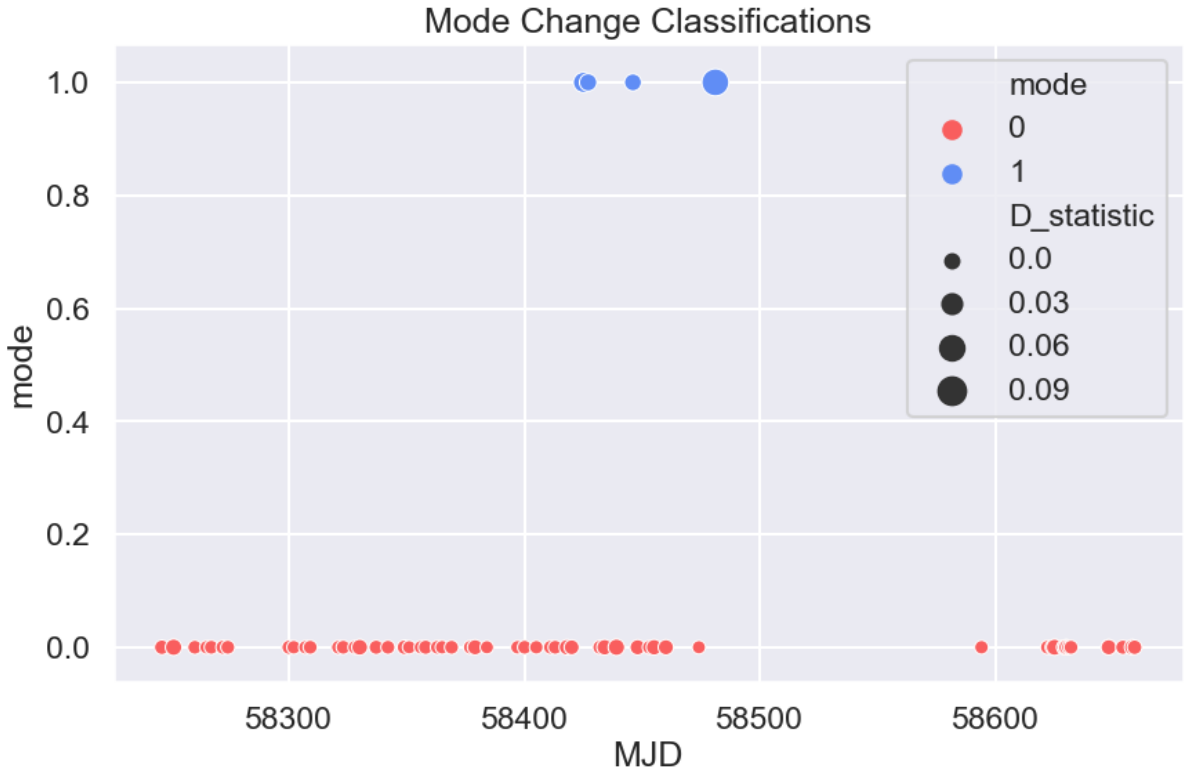



**Figure 16:** Plotting mode changes as a function of epoch. 0 indicates same mode as null profile. 1 indicates a different mode. Size of marker indicates D-statistic value. [Python 'matplotlib', 'seaborn']

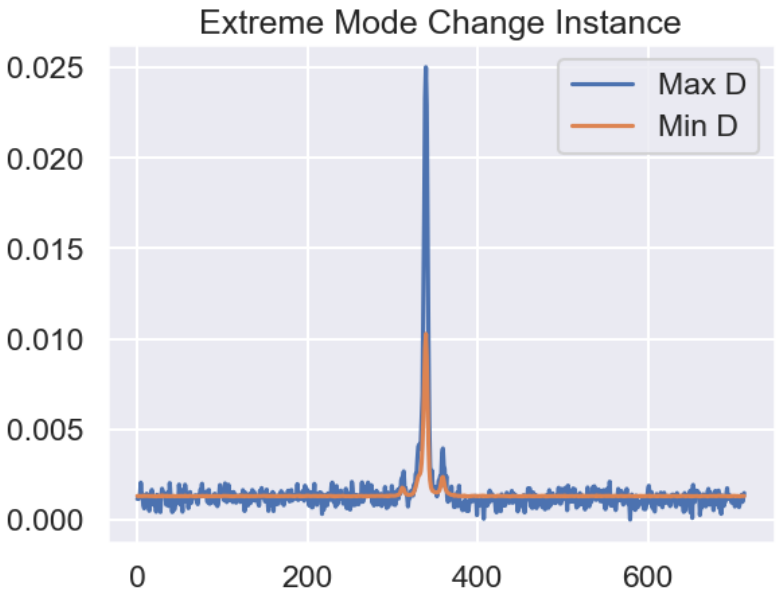



**Figure 17:** Plotting profiles with maximum and minimum values of D-statistic. There is a clear difference in profile shape. [Python 'matplotlib', 'seaborn']

## 0.7 Wavelet Analysis

The Fourier transform is a widely used method of analyzing signals. However, its downside is that one can have information about a signals frequency content or time content (as a time series), but not both. There are workarounds around this, such as the windowed Fourier Transform. This still has its disadvantages. This motivates the use of a wavelet transform, which compromises between time and frequency content. At higher frequencies, we achieve a precise time localization at the expense of frequency localization. At lower frequencies, we achieve a precise frequency localization at the expense of time localization. This popular visualization is commonly used.

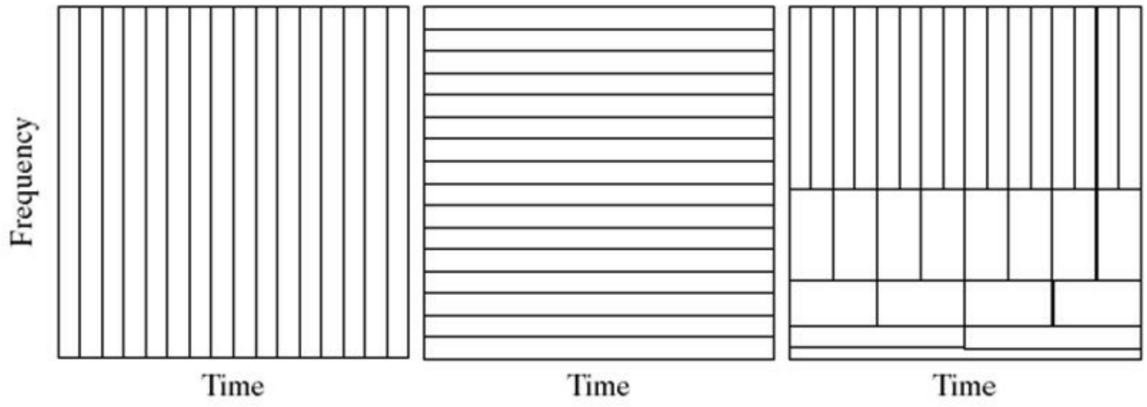



**Figure 18:** In the time-frequency domain, (a) Time series (b) Fourier transform (c) Wavelet Transform

### 0.7.1 Continuous Wavelet Transform

In wavelet analysis, wavelets are functions of two parameters, a translation parameter $b$ and a dilation parameter $a$, both of which vary continuously over $\mathbb{R}$ ($a \neq 0$). We define $\psi$, the "mother" wavelet. Then, its family of wavelets is defined as

$$\psi^{a,b}(x) = \frac{1}{\sqrt{a}}\psi\left(\frac{x-b}{a}\right), \quad a, b \in \mathbb{R}, a \neq 0$$

And it is normalized to unity, $||\psi^{a,b}|| = ||\psi|| = 1$.
But first, for a function to be a wavelet, it must pass the admissibility criterion, ie, the constant used in inverse continuous wavelet transform (CWT), must satisfy

$$C_\psi = \int_{-\infty}^{\infty} \frac{|\hat{\psi}(\omega)|^2}{|\omega|} d\omega < \infty$$

Simply put, a wavelet must have a zero mean and be localized in time and space. Now, the definition of CWT is as follows.

$$(T^{wav} f)(a,b) = < f, \psi^{a,b} > = \int dx f(x) |a|^{-1/2} \psi^* \left(\frac{x-b}{a}\right)$$

where $^*$ denotes the complex conjugate. An interpretation is that a function $f$ can be written as a linear combination of wavelets $\psi^{a,b}$ with coefficients given by the wavelet

transform. The inverse CWT is then defined as

$$f = C_\psi^{-1} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{dadb}{a^2} (T^{wav} f)(a,b) \psi^{a,b}$$

### 0.7.2 Haar wavelet

The most basic wavelet is the Haar wavelet. First, we define the dyadic interval.
For each pair of integers $j, k \in \mathbb{Z}$, we define the interval

$$I_{j,k} = [2^{-j}k,\ 2^{-j}(k+1)) = \left[\frac{k}{2^j},\ \frac{k}{2^j} + \frac{1}{2^j}\right)$$

The collection of all such intervals is called the collection of all dyadic subintervals of $\mathbb{R}$. It is helpful to note that dyadic interval at scale $j$ always has length $|I| = 2^{-j}$, and so a larger scale $j$ means a smaller interval width. Also note that $I_{j,k} = I_{j+1,2k} \cup I_{j+1,2k+1}$. This is saying that a dyadic interval at scale $j$ is the union of two dyadic intervals at scale $j+1$, and that the two adjacent subintervals with scale $j+1$ split the larger interval into left and right halves.

Dyadic intervals have special properties that come into play when defining the Haar orthonomal system.
Given $j_0, k_0, j_1, k_1 \in \mathbb{Z}$, with either $j_0 \neq j_1$ or $k_0 \neq k_1$, then either
1. $I_{j_1,k_1} \cap I_{j_0,k_0} = \emptyset$
2. $I_{j_1,k_1} \subseteq I_{j_0,k_0}$, or
3. $I_{j_0,k_0} \cap I_{j_1,k_1}$

Property 1 leads to the idea of orthonormality of the Haar wavelet bases which are defined on dyadic intervals. Properties 2 and 3 state that the smaller dyadic subinterval must either be contained in the left half or right half of a later dyadic interval.

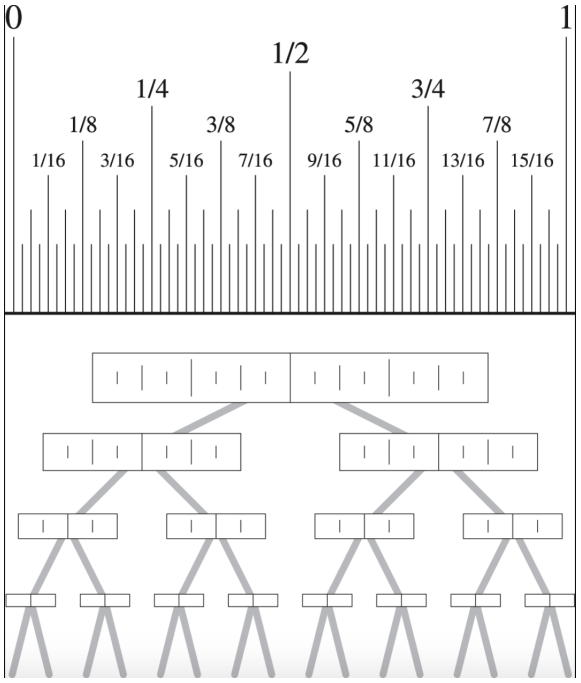



**Figure 19:** A dyadic interval on $[0,1]$. It can also be thought of as a binary tree.

Now we can define the Haar scaling function (on the interval $[0,1)$), of the Haar "father" wavelet. Let $p(x) = \chi_{[0,1)}(x)$, where $\chi$ is an indicator function (value 1 on interval, 0 elsewhere), and for each $j, k \in \mathbb{Z}$, define

$$p_{j,k}(x) = 2^{j/2} P(2^j x - k) = D_{2^j} T_k p(x)$$

where $D$ is the dilation operator, and $T$ is the translation operator. Then the collection $\{p_{j,k}(x)\}_{j,k\in\mathbb{Z}}$ is the system of Haar scaling functions, and we say for each $j \in \mathbb{Z}$, the collection $\{p_{j,k}(x)\}_{k\in\mathbb{Z}}$ is called the system of scale $j$ Haar scaling functions. Essentially, the system of Haar scaling functions is a system of dilated and translated $\chi_{[0,1)}$ that exist on dyadic intervals, and have a constant area of 1.

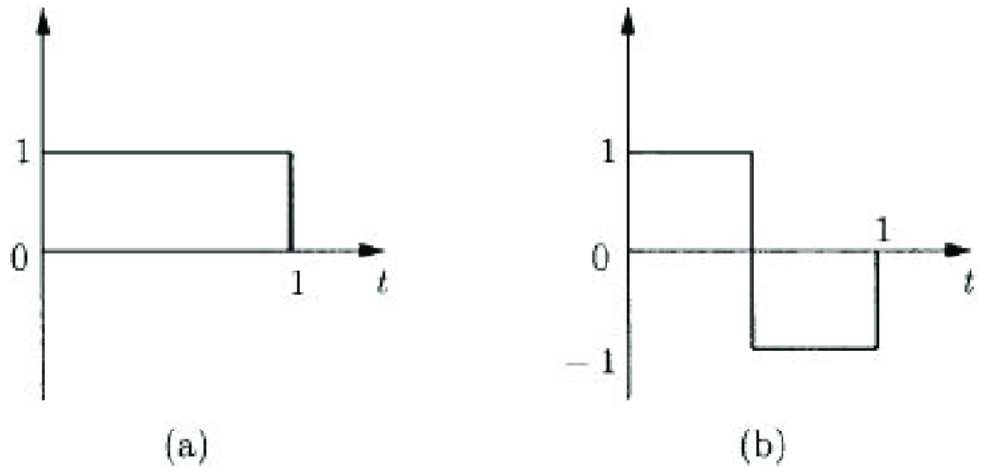



**Figure 20:** (a) Haar scaling function, $p_{0,0}$ (b) Haar wavelet function, $h_{0,0}$

Now we can define the Haar wavelet, or the "mother" wavelet.
Let $h(x) = \chi_{[0,1/2)}(x) - \chi_{[1/2,1)}(x)$, and for each $j, k \in \mathbb{Z}$, define

$$h_{j,k}(x) = 2^{j/2} h(2^j x - k) = D_{2^j} T_k h(x)$$

The collection $\{h_{j,k}(x)\}_{j,k\in\mathbb{Z}}$ is the Haar system on $[0,1)$, and for each $j \in \mathbb{Z}$, the collection $\{h_{j,k\in\mathbb{Z}}(x)\}_{k\in\mathbb{Z}}$ is the system of scale $j$ Haar functions. We see that the Haar wavelet $h_{j,k}(x)$ is supported on the dyadic interval $I_{j,k}$, and so we refer to the Haar function $h_{j,k}(x)$ as being associated with the interval $I_{j,k}$
We have only defined both the Haar scaling function and Haar function on $[0,1)$. Now we can define the Haar system on $\mathbb{R}$.
Let a fixed scale $J \in \mathbb{Z}$ be given. The collection

$$\{p_{j,k}(x), h_{j,k}(x) : j \geq J, k \in \mathbb{Z}\}$$

is called the scale $J$ Haar system on $\mathbb{R}$. It can be shown that the scale $J$ Haar system is a complete orthonormal system on $\mathbb{R}$. This allows us to represent any signal as a superposition of the Haar functions and a single Haar scaling function. More detail on how the discrete wavelet transform (DWT) is achieved is shown in 0.7.5 using the Haar system.

### 0.7.3 Other Wavelet Examples

The Morlet wavelet function is defined as

$$\psi_0(\eta) = \pi^{-1/4} e^{i\omega_0 \eta} e^{-\eta^2/2}$$

where $\eta$ is a non-dimensional "time" parameter, and $\omega_0$ is the wave number. One can observe that a Morlet wavelet is essentially a sinusoid modulated by a Gaussian to become a "packet" and be localized in time.

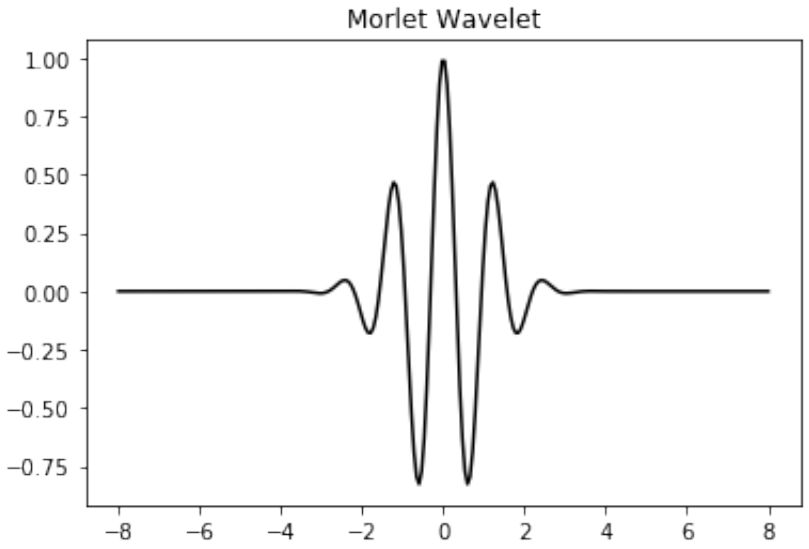



**Figure 21:** A example Morlet wavelet. [Python 'PyWavelets']

The first order derivative of Gaussian wavelet is defined as

$$\psi(t) = -te^{-t^2/2}$$

Because a Gaussian is infinitely differentiable, one can use any derivative of Gaussians (DOG) when performing wavelet transforms. The second-order DOG is also known as the Mexican-hat wavelet.

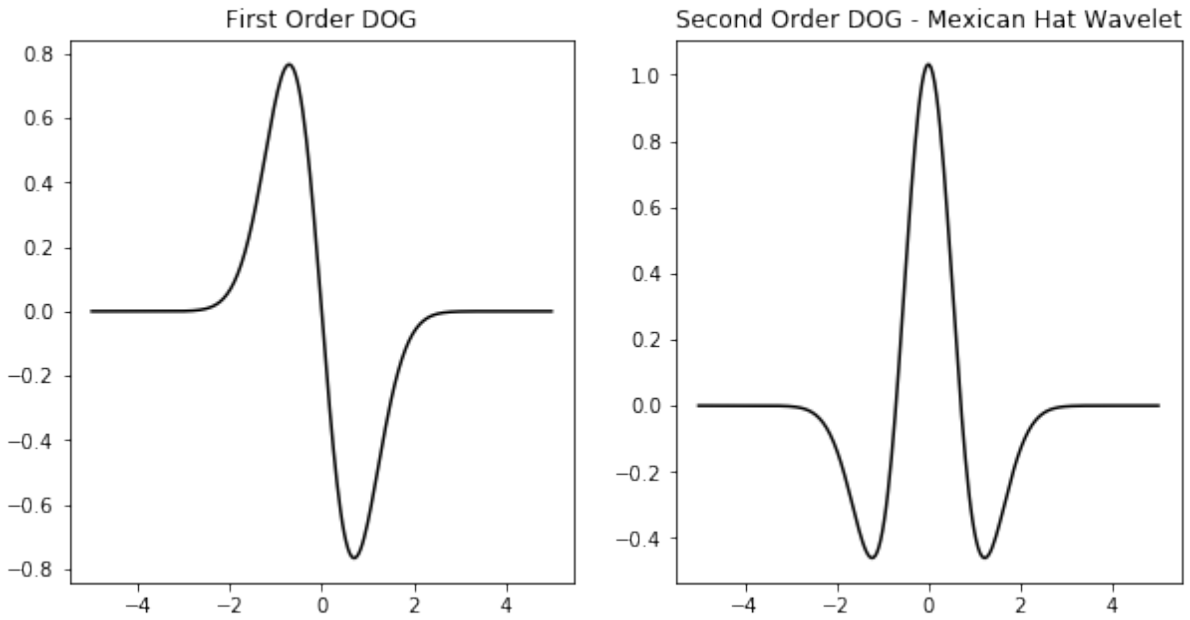



**Figure 22:** First and second order DOG wavelets. [Python 'PyWavelets']

Some other examples include Paul are Debauchies wavelets. There is no clear rule on which mother wavelet to choose when analyzing a signal. As a general rule, one would choose a wavelet that best approximates the features in a desired signal. Also, many wavelet transform software use Morlet wavelets as a default. This is because a Morlet wavelet is conservative middle in terms of localization in time and frequency. For example, Paul $<$ Morlet $<$ Gaussian in terms of frequency resolution, while Gaussian $<$ Morlet $<$ Paul in terms of time resolution. If one were to use a wavelet packet decomposition of a signal (not explained in this report, but the basic idea is to decompose not just the approximation coefficients as in a DWT, but the detail coefficients as well, completing the full decomposition binary tree. The binary tree figure for a DWT is shown in 0.7.4.), then there exists a Best-Basis algorithm that is designed to find the most optimal wavelet packet basis by minimizing a cost function.

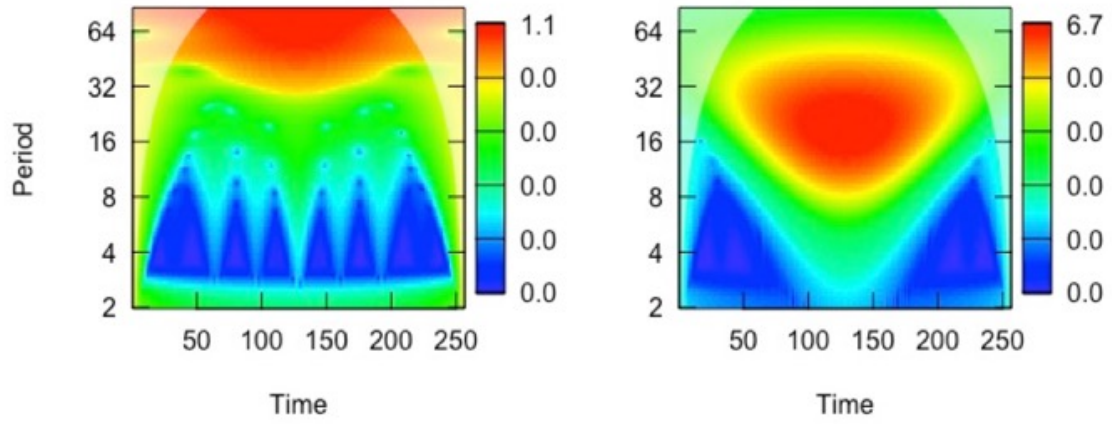



**Figure 23:** Wavelet power spectrum of a (a) Morlet wavelet (b) first order DOG wavelet. [R 'WaveletComp'].

### 0.7.4 Discrete Haar Transform

First, a motivational example. Suppose one has a signal of length 8, say $x(t) = \{1,5,7,5,7,3,1,7\}$ Now if one had to share this signal with another person, but could only do so with a signal of length 4, what would be the best way? A logical solution would be to average each pair of values, and send the signal $a(t) = \{3,6,5,4\}$. Now if we had to reconstruct the original signal $x(t)$ from $a(t)$, what information would we need? Because each value in $a(t)$ is an average of a pair of values, this means each value is equidistant from the original pair of values. Hence, if we had a signal to store the pairwise differences, $d(t) = \{-2,1,2,-3\}$, then one could fully restore the signal. Specifically, we can recover the original signal by $x(t) = \{3+(-2),\ 3-(-2),\ 6+(1),\ 6-(1),\ 5+(2),\ 5-(2),\ 4+(-3),\ 4-(-3)\} = \{1,5,7,5,7,3,1,7\}$. This is the basic idea behind the Discrete Haar Transform, and more loosely, any DWT.

In a Discrete Haar Transform (DHT), the averaging of pairwise values in a signal is achieved by the Haar scaling function $\varphi(x)$, which is analogous to a low-pass filter of the signal. The pairwise difference operation is performed by the Haar wavelet function, $\psi(x)$, which is analogous to a high-pass filter of the signal. The goal is to represent a function f(x), say defined on the interval $[0,1)$, in terms of the Haar system. Given an

integer $J \geq 0$, we want to achieve

$$f(x) = \sum_{j=J}^{\infty} \sum_{k=0}^{2^j-1} \langle f, h_{j,k} \rangle h_{j,k}(x) + \sum_{k=0}^{2^J-1} \langle f, p_{J,k} \rangle p_{J,k}(x)$$

The 2nd term is a function approximation with a scaling function at scale $J$, where this scale corresponds to a low resolution approximation. Then, adding all the detail/difference terms at all scales and positions in the signal, we can reconstruct $f(x)$.

Now we formally define the DHT. Define the approximation coefficient at scale $j$ and position $k$ as $c_j(k)$ and the detail coefficient at scale $j$ and position $k$ as $d_j(k)$ as follows.

$$c_j(k) = \langle f, p_{N-j,k} \rangle, \;\; d_j(k) = \langle f, h_{N-j,k} \rangle$$

Note that while for the Haar scaling function $\varphi$ and the Haar wavelet $\psi$, a larger $j$ corresponds to a smaller dyadic interval on which it is supported. Now, due to the indexing $N-j$, a larger $j$ for the coefficients $c_j$ and $d_j$ now correspond to a larger width, or lower resolution. Now, given $J, N \in \mathbb{N}$ with $J < N$ and a finite sequence $c_0 = \{c_0(k)\}_{k=0}^{2^N-1}$, the DHT is defined by

$$\{d_j(k) : 1 \leq j \leq J, 0 \leq k \leq 2^{N-j} - 1\} \cup \{c_j(k) : 0 \leq k \leq 2^{N-j} - 1\}$$

, where

$$c_j(k) = \frac{1}{\sqrt{2}} \left[ c_{j-1}(2k) + c_{j-1}(2k+1) \right], \;\; d_j(k) = \frac{1}{\sqrt{2}} \left[ c_{j-1}(2k) - c_{j-1}(2k+1) \right]$$

and the Inverse DHT is given by

$$c_{j-1}(2k) = \frac{1}{\sqrt{2}} \left[ c_j(k) + d_j(k) \right], \;\; d_{j-1}(k) = \frac{1}{\sqrt{2}} \left[ c_j(k) - d_j(k) \right]$$

This recursion relation allow the DHT to be implemented through an algorithm Recall the example earlier where we wanted to convey a signal $x(t)$ to another person by storing information of pairwise averages and pairwise differences. We can clearly see the analogy in the formal definition. In the DHT, we take pairwise $(2k, 2k+1)$ approximation coefficients at a scale $j-1$ and produce a single value $c_j(k)$ which is at a scale $j > j-1$, hence at a lower resolution (taking a size 8 signal and averaging it to a signal of size 4). Likewise, for the detail coefficient $d_j(k)$, we take pairwise approximations of a function

at a smaller scale, but take the difference this time.

The DHT can be expressed as a matrix operation, which makes it more intuitive to understand. GIven $L \in \mathbb{N}$ even, define $\left(\frac{L}{2}\right)$ x $L$ matrices $\mathbf{H_L}, \mathbf{G_L}$ by

$$\mathbf{H_L} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 1 & 0 & \dots & 0 \\ & & & & & \vdots & \\ 0 & 0 & 0 & \dots & 0 & 1 & 1 \end{pmatrix} \quad \mathbf{G_L} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & -1 & 0 & \dots & 0 \\ & & & & & \vdots & \\ 0 & 0 & 0 & \dots & 0 & 1 & -1 \end{pmatrix}$$

$\mathbf{H_L}$ is the approximation matrix, thought of as computing pairwise averages for the Haar wavelet. $\mathbf{G_L}$ can be thought of as the detail matrix, computing pairwise differences for the Haar wavelet. Then, we define the $L$ x $L$ matrix $\mathbf{W_L}$, the wavelet matrix, by

$$\mathbf{W_L} = \begin{pmatrix} \mathbf{H_L} \\ \mathbf{G_L} \end{pmatrix}$$

Then, given $J, N \in \mathbb{N}$ with $J < N$ and length $2^N$ vector $\mathbf{c_0} = (c_0(0), c_0(1), ..., c_0(2^N - 1))$, the DHT of $\mathbf{c_0}$ is the vector $(\mathbf{d_1}, \mathbf{d_2}, \mathbf{d_3}, ..., \mathbf{d_J}, \mathbf{c_J})$, where

$$\begin{pmatrix} \mathbf{c_j} \\ \mathbf{d_j} \end{pmatrix} = \begin{pmatrix} \mathbf{H} \\ \mathbf{G} \end{pmatrix} \mathbf{c_{j-1}}$$

with $\mathbf{H}, \mathbf{G}$ are $2^{N-j}$ x $2^{N-j+1}$, with $1 \leq j \leq J$. The Inverse DHT is given by

$$\mathbf{c_{j+1}} = \mathbf{H}^* \mathbf{c_j} + \mathbf{G}^* \mathbf{d_j}$$

The DHT (and DWT in general) is visualized by a tree diagram for DHT. First, given a signal $\mathbf{c_0}$ of length $N$, we apply the approximation and detail matrix $\mathbf{H}, \mathbf{G}$ combined in the form of $\mathbf{W}$. The $\mathbf{H}$ gives us approximation coefficients $c_1$ of length $\frac{N}{2}$, and $\mathbf{G}$ gives us the detail coefficients $d_1$ of length $\frac{N}{2}$. Now, we save the detail coefficient $d_1$, and apply $\mathbf{W}$ to $\mathbf{c_1}$ only. This gives us coefficients $\mathbf{c_1}, \mathbf{d_1}$, both length $\frac{N}{2^2}$. We save $\mathbf{d_1}$ and repeat. We continue this wavelet "decomposition" to a desired level or until the length of the approximation coefficient can no longer be halved.

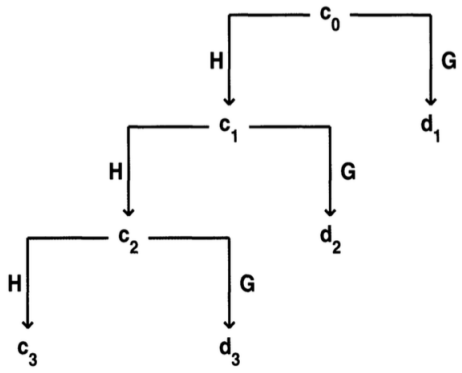



**Figure 24:** A tree diagram illustrating the DHT. Taken from [1].

Hence, you end up with a single approximating function at a fixed low resolution

$$c_J = \sum_{k=0}^{2^J-1} \langle f, p_{J,k} \rangle p_{J,k}(x)$$

plus details at varying scales

$$\sum_{j=J}^{\infty} \sum_{k=0}^{2^j-1} \langle f, h_{j,k} \rangle h_{j,k}(x)$$

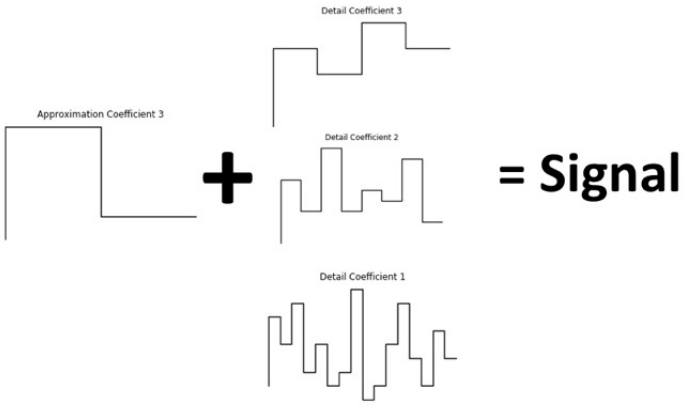



**Figure 25:** A visual interpretation of a DHT and its coefficients [Python 'matplotlib'].

The approximation coefficients of the DHT of a signal of length $2^4$ is shown below.

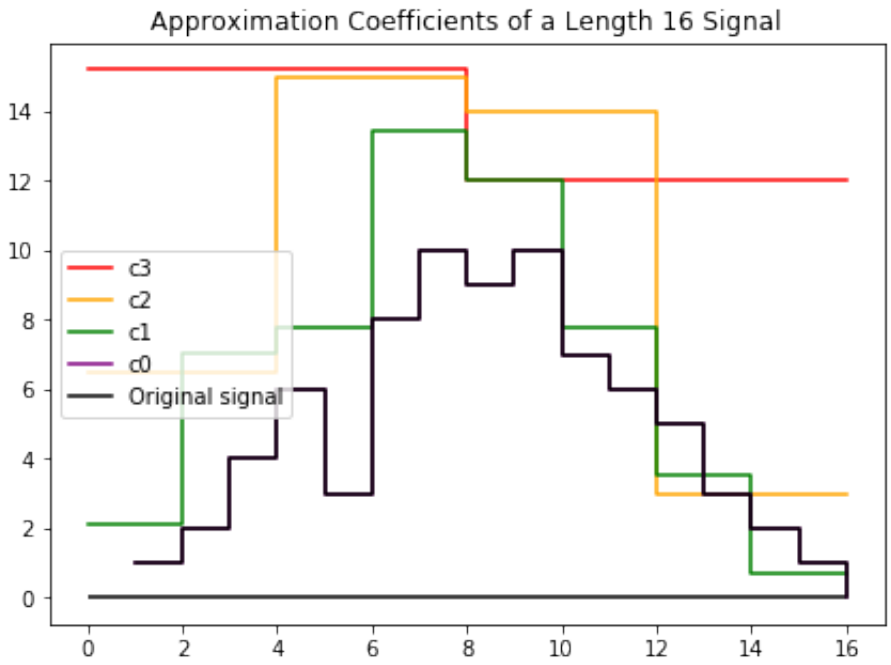



**Figure 26:** Approximation coefficients at each level of decomposition. c0 is identical to the original signal. DHT would progress from c0 to c3, as the high resolution details are stored in detail coefficients and the progressively low resolution information is stored in the approximation coefficients. Inverse DHT, given all detail coefficients and c3, reconstructs the original signal. [Python 'matplotlib', 'PyWavelets']

### 0.7.5 Discrete Wavelet Transform

(This section might have errors, I had to go through the Introduction to Wavelet Analysis quickly so I still do not perfectly understand MRA and the DWT. This is my best interpretation.) Before going into DWT, one should study multi-resolution analysis (MRA, Chapter 7 of [1]). MRA allows one to construct their own wavelets (satisfying some conditions), and apply DWT. The discrete wavelet transform (DWT) essentially uses the similar idea outlined in the DHT. The DWT of a lenth $M = 2^N$ signal can be thought of as a linear transformation of the M-vector

$$\mathbf{c_0} = [c_0(0), c_0(1), ..., c_0(M-1)]$$

into the M-vector

$$\mathbf{d} = \left[\mathbf{d_1}, \mathbf{d_2}, ..., \mathbf{d_J}, \mathbf{c_J}\right]$$

with

$$\mathbf{d_j} = \left[d_j(0), d_j(1), ..., d_j(2^{-j}M-1)\right], \ \mathbf{c_J} = \left[c_J(0), c_J(1), ..., c_j(2^{-J}M-1)\right]$$

In DWT, $l^2$ sequence of coefficients $\{h(k)\}$ which satisfies

$$\varphi(x) = \sum_k h(k) 2^{1/2} \varphi(2x-k)$$

on $\mathbb{R}$, where $\varphi$ is the scaling function associated with a given MRA $V_j$, is called the scaling filter associated with $\varphi$. Then, define the wavelet filter

$$g(n) = (-1)^n \overline{h(1-n)}$$

. Once we have the scaling filter $h(k)$ and the wavelet filter $h(n)$, we can define the scaling function and the wavelet function as follows.

$$\varphi(x) = \sum_n h(n) \varphi_{1,n}(x)$$

and the corresponding wavelet function

$$\psi(x) = \sum_n g(n) \varphi_{1,n}(x)$$

. Now, we can formally define the DWT.
Defining scaling function $\varphi$, scaling filter $h(k)$, wavelet filter $g(n)$ and the wavelet function $\psi(x)$ as above, given $f(x), L^2$ on $\mathbb{R}$, define for $k \in \mathbb{Z}$,

$$c_0(k) = \langle f, \varphi_{0,k} \rangle$$

and for every $j \in \mathbb{N}, k \in \mathbb{Z}$,

$$c_j(k) = \langle f, \varphi_{-j,k} \rangle, \ \ d_j(k) = \langle f, \psi_{-j,k} \rangle$$

. Then, the recursion relations between coefficients used in DWT are

$$c_{j+1}(k) = \sum_n c_j(n) \overline{h(n-2k)}, \ \ d_{j+1}(k) = \sum_n c_j(n) \overline{g(n-2k)}$$

and

$$c_j(k) = \sum_n c_{j+1}(n)h(k-2n) + \sum_n d_{j+1}(n)g(k-2n)$$

Again, it is more intuitive to visibly see the approximation and detail matrices. Let $h(k)$ be a real valued scaling filter of length 4, define the wavelet filter $g(k) = (-1)^k h(3-k)$, and a signal of length 8. Then

$$\mathbf{H_8} = \begin{pmatrix} h(0) & h(1) & h(2) & h(4) & 0 & 0 & 0 & 0 \\ 0 & 0 & h(0) & h(1) & h(2) & h(4) & 0 & 0 \\ 0 & 0 & 0 & 0 & h(0) & h(1) & h(2) & h(4) \\ h(2) & h(3) & 0 & 0 & 0 & 0 & h(1) & h(2) \end{pmatrix}$$

and

$$\mathbf{G_8} = \begin{pmatrix} g(0) & g(1) & g(2) & g(4) & 0 & 0 & 0 & 0 \\ 0 & 0 & g(0) & g(1) & g(2) & g(4) & 0 & 0 \\ 0 & 0 & 0 & 0 & g(0) & g(1) & g(2) & g(4) \\ g(2) & g(3) & 0 & 0 & 0 & 0 & g(1) & g(2) \end{pmatrix},$$

Then, constructing the wavelet matrix $\mathbf{W}$ as in DHT, a first level decomposition of length M-vector $\mathbf{c_0}$ yields coefficients at level 1,

$$\mathbf{W_M c_0} = \begin{pmatrix} \mathbf{d_1} \\ \mathbf{c_1} \end{pmatrix}$$

In the second step/level,

$$\begin{pmatrix} \mathbf{I_{(M/2)}} & 0 \\ 0 & \mathbf{W_{M/2}} \end{pmatrix} \begin{pmatrix} \mathbf{d_1} \\ \mathbf{c_1} \end{pmatrix} = \begin{pmatrix} \mathbf{d_1} \\ \mathbf{d_2} \\ \mathbf{c_2} \end{pmatrix}$$

where $\mathbb{I}$ is a $M/2$ x $M/2$ identity matrix, there to preserve information on the detail matrix, and only apply the wavelet matrix only to the approximation coefficients. In general, the $j^{th}$ step is

$$\begin{pmatrix} \mathbf{I_{1-2^{-j}}} & 0 \\ 0 & \mathbf{W_{2^{-j}M}} \end{pmatrix} \begin{pmatrix} \mathbf{d_1} \\ \vdots \\ \mathbf{d_{j-1}} \\ \mathbf{c_j} \end{pmatrix} = \begin{pmatrix} \mathbf{d_1} \\ \vdots \\ \mathbf{d_{j-1}} \\ \mathbf{d_j} \\ \mathbf{c_j} \end{pmatrix}$$

So far we have only worked with signals that are of length $2^N$ for some $N \in \mathbb{N}$. What

happens if our signal is not a power of 2? There are several ways to deal with this. The two main methods are zero padding and periodization.

In zero padding, we simply make the signal to have a length of power of 2 by appending 0's on both ends, as such: $\{0,0,0,0,5,2,7,4,7,43,4,7,2,0,0,0\}$. This introduces artificially diminished wavelet powers at the ends of signal, leading to the requirement of the cone of influence, described in the next section. Periodization is, as it suggests, extending the signal such that it is a periodic signal of length that is a power of 2.

### 0.7.6 Wavelet Power Spectrum

The wavelet power spectrum is a powerful visual tool, but more than that, it is used in the significance testing described in the next section. The wavelet powerspectrum displays the wavelet power of a signal, with frequency/period in the vertical axis, and time in the horizontal axis. Recall the definition of the CWT. Defining $a$ as the scale parameter and $b$ as the translation parameter. Then, the CWT of the time series $x(t)$ is

$$W(b,a) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t) \psi^* \left( \frac{t-b}{a} \right) dt$$

The wavelet power is then the quantity $|W(b,a)|^2$. In implementation, the integral is a summation over $N$ samples in the time series, and the computation is actually done in the Fourier domain. A step-by-step guide is in Schulte.

The cone of influence (COI) is a region over the spectrum, especially prominent in the lower frequency (larger scale parameter $a$). This is a region where the wavelet extends beyond the observed region of the signal and edge effects come into play. One must be cautious when interpreting spectrum features in the COI region. For example, if one zero-pads a finite signal when performing DWT, then when the daughter wavelet is being multiplied with the signal near the ends, the zeros diminish the power within the COI.

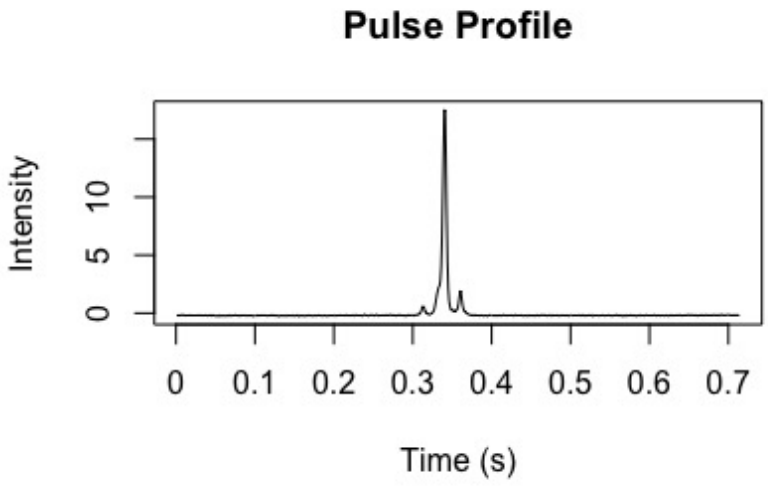



**Figure 27:** J0332+5434 pulse profile with period of 0.714 seconds. [R]

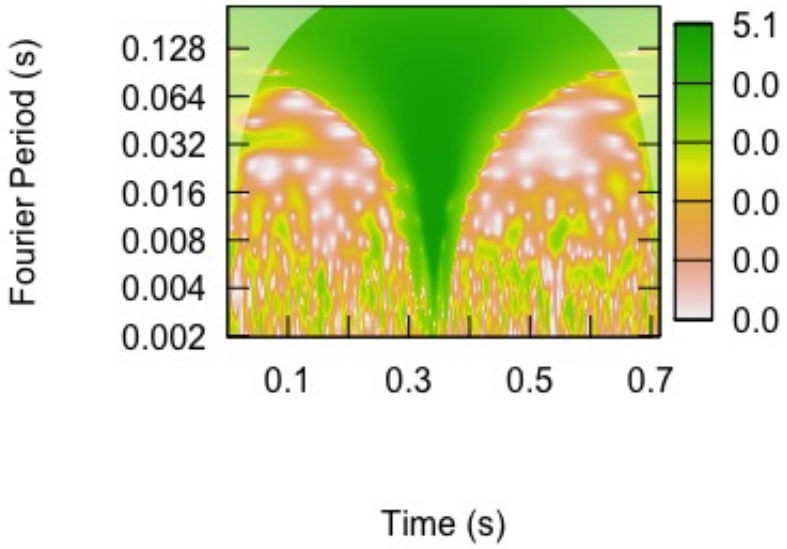



**Figure 28:** Wavelet power spectrum of the pulse profile. COI is the light gray shading at the corners. [R 'WaveletComp']

Interpreting the horizontal axis, time, is pretty straightforward. The vertical axis, in this figure shown as period, is sometimes confusing. First, remember that this is a plot of the timescale plane, with scale converted to period. Hence, the period on the vertical axis denotes the length of the daughter wavelet in time units. Now recall that a wavelet transform obtains good frequency localization at lower frequencies at the expense of time

localization, while obtaining good time localization at higher frequencies at the expense of frequency localization. This is clearly visible in the spectrum above. First, note that if the vertical axis were converted into frequency (1/period), as we go up, frequency would decrease. Now observe that at higher frequencies (lower on the vertical axis), the contour patches are stretched along the frequency direction. Thus it has very precise time localization, but not so much in the frequency domain. Meanwhile, at lower frequencies (higher on the vertical axis), patches begin to stretch along the time direction. Hence at lower frequencies, wavelet transform has precise frequency localization but not so much in the time domain.

## 0.8 Wavelet Significance Testing

### 0.8.1 Pointwise Testing

The first step-by-step significance testing in wavelet analysis was first introduced by Torrence and Compo (1998). The method introduced is commonly referred to as pointwise significance testing. The basic idea is this: define the set containing all points in a wavelet spectrum as $H$. One then picks a background noise representing the null hypothesis, performs thousands of Monte Carlo simulations of the noise, takes wavelet transforms, and plots the distribution of wavelet powers (or whatever the desired wavelet quantity is, eg., wavelet coherency). The critical power corresponding to a pointwise significance level $\alpha_{pw}$ is simply the $1-\alpha_{pw}$ th percentile of the Monte Carlo distributions. Then, one compares each point in $H$ against this critical value, hence the name "pointwise".

Here is a more detailed outline. I will only consider the wavelet power from here on. For each point in $H$, assign a p-value, $\rho_{pw}$, the probability of observing an equal or more extreme wavelet power if the null hypothesis (say pure WN) is true. The test then returns a subset of $H$,

$$P_{pw} = \{(b,a) : \rho_{b,a}(b,a) < \alpha_{pw}\}$$

which is the set of all points that are pointwise significant. Typically, in a power spectrum, the groups of pointwise significant points are encircled by a contour which is a function of $\alpha_{pw}$. The encircled area is referred to as patches.

The issue with pointwise testing is that it identifies many spurious patches as significant. The following sections describe areawise, geometric, and cumulative areawise testing pro-

cedures, which were designed to reduce the number of false positives.

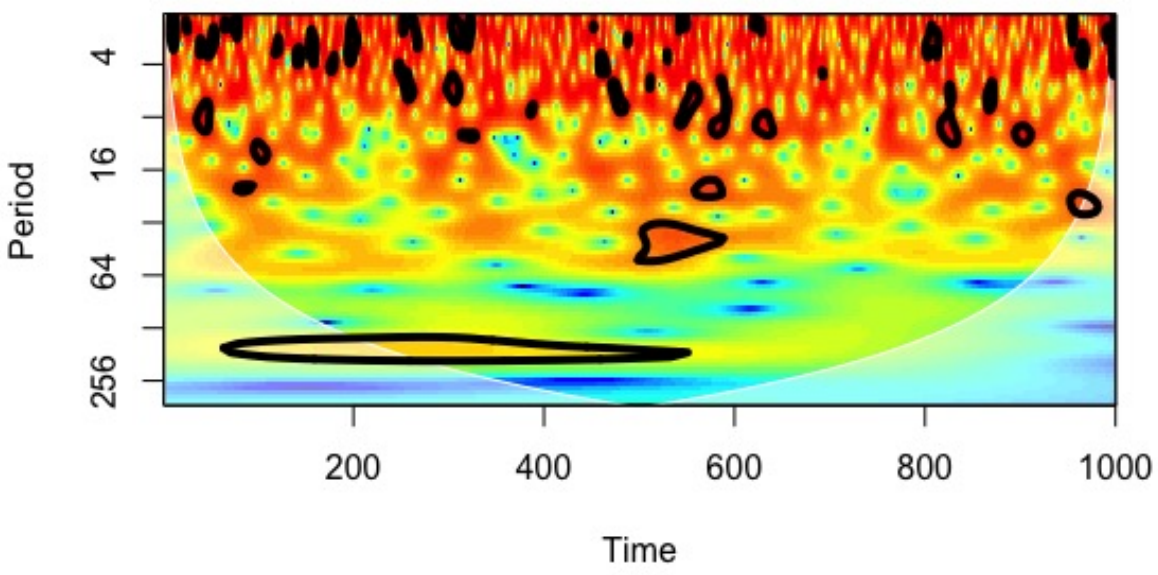



**Figure 29:** Power spectrum of pure WN signal against a null hypothesis of pure WN. Pointwise testing falsely identifies many patches as significant at $\alpha_{pw} = .05$. [R 'biwavelet']

### 0.8.2 Areawise Testing

Maraun et al. (2007) developed the areawise testing to reduce spurious patches. Maraun and Kurth (2004) noted that points in $H$ in the timescale plane are correlated. Given two points, $(b, a), (b', a') \in H$, the correlation between the two points' coefficients is represented by the reproducing kernel

$$K(b, a, b', a') = \frac{1}{C_\psi \sqrt{(aa')}} \int \left[ \psi\left(\frac{t-b}{a}\right) \psi^*\left(\frac{t-b'}{a'}\right) \right] dt$$

where $C_\psi$ is the admissibility constant, and $\psi$ is the mother wavelet. Then, the wavelet coefficient at $(b, a)$ is

$$W(b, a) = \int\int K(b, a, b', a') W(b', a') \frac{da' db'}{a'^2}$$

where $K(b, a, b', a')$ can be thought of as a weight multiplying the neighboring point's wavelet coefficient $W(b', a')$. Hence, information from neighboring points is captured. This leads to the idea that spurious patches arise in clusters due to the correlations in coefficients. We can use the fact that typical spurious patches have an area similar to that of the kernel at the position $(b, a)$ to reduce false positives.

First, we define areawise significant patches. Given all pointwise significant patches $P_{pw}$, choose a critical area $P_{crit} \subset H$ for which the reproducing kernel, when dilated and translated to $(b, a)$, exceeds some critical level $K_{crit}$,

$$P_{crit}(b, a) = \{(b', a') : K(b, a, b', a') > K_{crit}\}$$

Then the set of all points whose wavelet power is areawise significant (on top of being pointwise significant) is

$$P_{aw} = \bigcup_{P_{crit}(b,a) \subset P_{pw}} P_{crit}(b, a)$$

. Essentially, given a pointwise significant patch, a point $(b, a)$ inside this patch is also areawise significant if the reproducing kernel, dilated and translated to $(b, a)$, fits completely in the patch. An issue with areawise testing, however, is that determining the critical area corresponding to $\alpha_{aw}$ requires a root finding algorithm which is computationally expensive.

### 0.8.3 Geometric Testing

In order to avoid this root finding algorithm, Schulte et al. (2015) developed a geometric test. The key was to define a new test statistic, $A_{norm}$, which is the normalized area of a pointwise significant patch area, $A_{patch}$.

$$A_{norm} = \frac{A_{patch}}{\hat{a}^2}$$

where $\hat{a}^2$ is the patch's mean scale coordinate, detailed in his paper. The division accounts for a patch's expansion in time and scale direction as scale increases (toward lower frequency). Hence, once can compare area of any patch anywhere in $H$ through this normalization. Areawise significance testing steps are as follows.

First, generate many wavelet spectrum under some null hypothesis, say white noise,

through Monte Carlo simulation. Next, create a null distribution of of $A_{norm}$ using patches in the spectra. Then, one can find the critical $A_{norm}$ corresponding to some geometric significance level $\alpha_{geo}$ by computing $1-\alpha_{geo}$ th percentile of the null distribution.

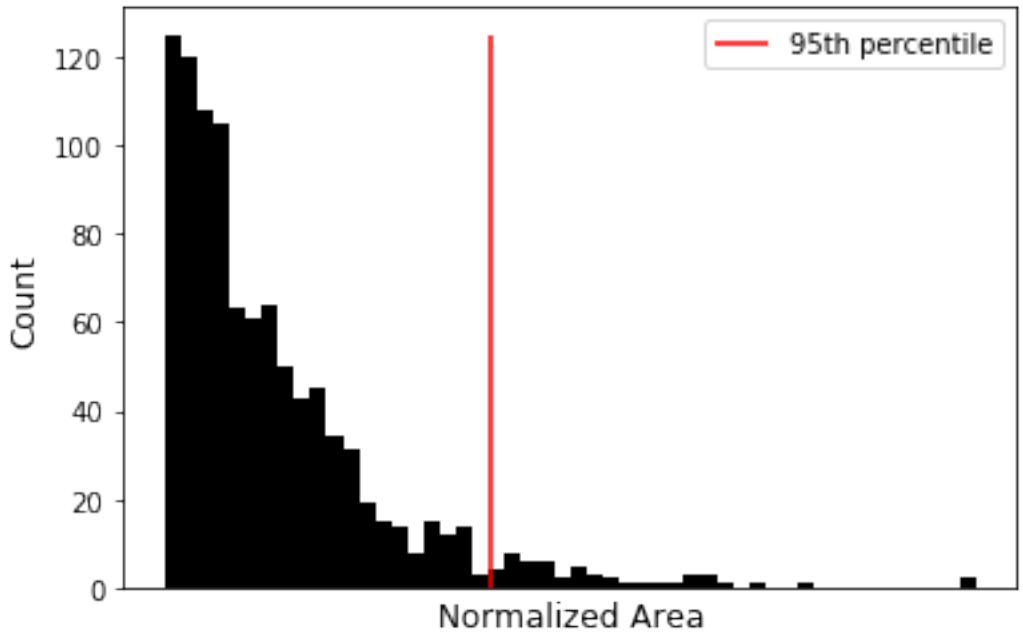



**Figure 30:** Value of normalized area at the 95th percentile of Monte Carlo simulations is the critical $A_{norm}$ corresponding to $\alpha_{geo} = .05$. To be rigorous, one would fit a distribution to the histogram and then take the 95th percentile. [Python 'matplotlib'].

### 0.8.4 Cumulative Areawise Testing

The issue with both the areawise and geometric testing is that one needs to select two different significant levels, $\alpha_{pw}$ and $\alpha_{aw}$ or $\alpha_{geo}$. Thus, interpretation of the significance of a patch is not clear. For example, is a patch that is significant at $(\alpha_{pw} = .05, \alpha_{geo} = .01)$ more significant then one at $(\alpha_{pw} = .01, \alpha_{geo} = .05)$? Hence, an improvement to the geometric testing was introduced in Schulte (2016). The idea was to study how the geometric and topological characteristics of a patch changes as $\alpha_{pw}$ varied. I will try to explain the test here, but there are a lot of details and excellent figures that explain the topological ideas in the original paper that won't be discussed in this report. Figure 4 depicting topological evolution and figure 9 depicting geometric pathways in the paper are especially helpful in understanding the test.

First, given pointwise significant patches, the goal of this test is to assess changes in

the normalized area $A_{norm}$ defined previously for each point in $H$, as **pointwise** significance $\alpha$ varies over $\alpha_1, \alpha_2, ..., \alpha_N$. If we choose $N$ pointwise significance levels such that $\alpha_1 < \alpha_2 < ... < \alpha_N$, then the sets

$$P_{pw}^{i} = \{(b, a) : \rho_{pw}(b, a) < \alpha_i\}$$

form a filtration of H,

$$\emptyset = P_{pw}^{1} \subseteq P_{pw}^{2} \subseteq ... \subseteq P_{pw}^{N} = H$$

for sufficiently small $\alpha_1$ and sufficiently large $\alpha_N$. Given a single point $x \in H$, the local filtration about $x$ forms a geometric pathway. Formally, a geometric pathway $G_x$, corresponding to a point $x \in H$, is a nested sequence

$$\emptyset = P_1^x \subseteq P_2^x \subseteq ... \subseteq P_N^x = H$$

with

$$P_i^x = \{(b, a) : (b, a) \in P_{pw}^{i}, (b, a) \sim x\}$$

where ~ denotes an equivalence relation. The meaning of equivalence relation between two points $x, (b, a) \in H$ is defined in the paper as well. In words, given a single point $x \in H$, begin at the lowest significance level, say $\alpha_1 = .10$. Many pointwise significant patches may appear. We sum up the normalized area $A_{norm}$ of only the patch with which $x$ belongs in. Now repeat this for increasing set of pointwise significance levels, say $\alpha_2 = .01, \alpha_3 = .03, ..., \alpha_N = .15$, for all points in $H$. Then the total sum of $A_{norm}$, or cumulative normalized area, for each point is the test statistic to be used. Each point is assigned a total normalized area integrated over $G_x$, or a "set of discrete significance levels". Hence, the name cumulative areawise testing. A visual interpretation of the test in shown in the next page.

Now, a step by step procedure for cumulative areawise testing. First, pick a set of discrete set of pointwise significance levels, $\alpha_1, \alpha_2, ..., \alpha_N$, and a point $x \in H$, to perform the test on. Second, calculate normalized areas $A_1^x, A_2^x, ..., A_N^x$ corresponding to $N$ members of $G_x$. Assume $A_i^x = 0$ if $P_i^x = \emptyset$ (point $x$ does not belong to any patches) or $P_i^x = \{x\}$ (the point $x$ is a patch by itself. Third, compute the sum $A^x = \sum_{k=1}^{N} A_x^k$, and compare this to some critical area $A_{crit}$ obtained from Monte Carlo method. Finally, repeat this $\forall x \in H$. To obtain $A_{crit}$, similar to previous tests, generate thousands of wavelet spectra and geometrical pathways under some null hypothesis, calculate the null distribution, and

calculate the $1-\alpha_c$ th percentile, where $\alpha_c$ is the desired significance of the cumulative test.

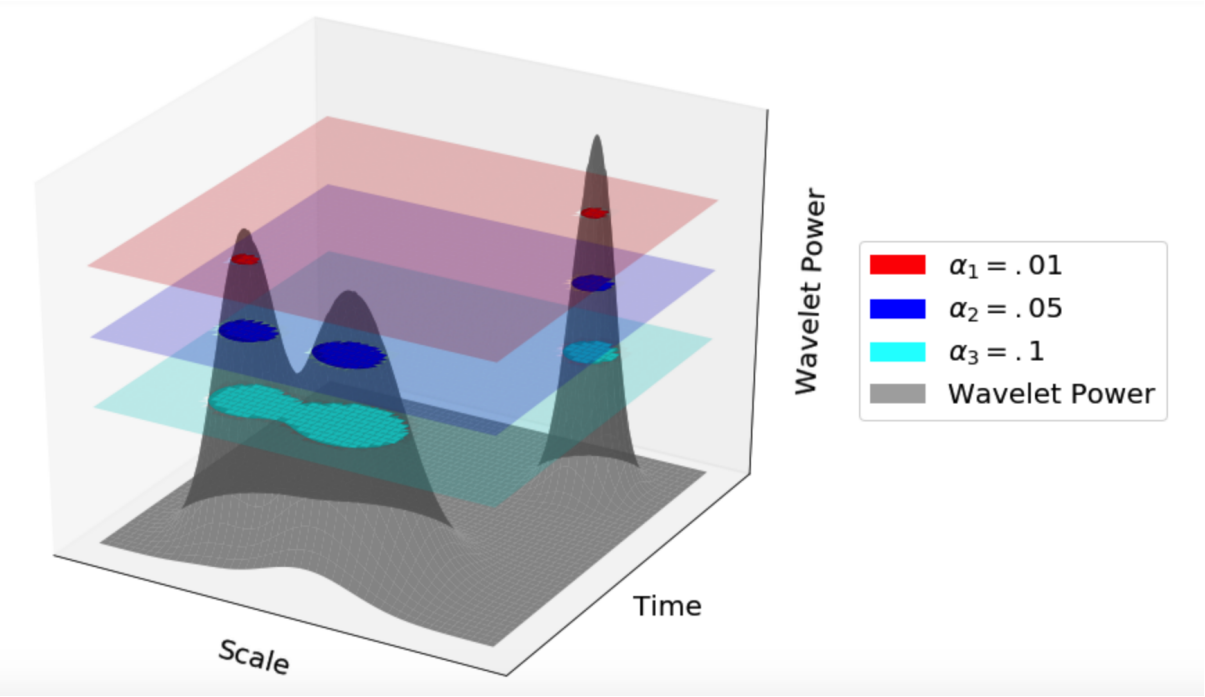



**Figure 31:** Interpretation of cumulative areawise testing as a filtration of $H$. Each colored area of intersection corresponds to a pointwise significant patch at the associated $\alpha_{pw}$ shown in legend. [Python 'matplotlib']

Let's imagine we applied cumulative areawise testing to the figure above. The grey surface represents the wavelet power of a signal in the timescale plane. Hence, a bivariate Gaussian with a tall height corresponds to a significant feature. Suppose we pick a point $x = (b, a)$ on the timescale plane that is at the center of the leftmost Gaussian, and a set of pointwise significance values $\alpha_1 = .01$, $\alpha_2 = .05$, and $\alpha_3 = .1$. Begin by applying pointwise testing with $\alpha_3 = .1$. The test returns cyan patches shown in the figure as significant. Because $x$ belongs in the left cyan patch, we add the normalized area, $A^3_x$, to our test statistic $A^x$. However, we *do not* add the area of the cyan patch on the right, as the point $x$ does not belong to the patch. Now, we apply pointwise testing to the spectrum with $\alpha_2 = .05$. Our significance level has decreased from .1 to .05, hence we are applying a more strict condition. This can be visualized as the blue plane, which is higher than the cyan plane and hence requires wavelet power to be larger to be considered significant. As such, the pointwise significant patches, colored in blue, have smaller area. In fact, the leftmost cyan patch has now separated into two separate blue patches. When we now add the normalized area $A^2_x$ to our test statistic $A^x$, we only add that of the leftmost blue

patch, as our point $x$ only belongs to that patch. Repeat for $\alpha_1 = .01$, which introduces even smaller patches.

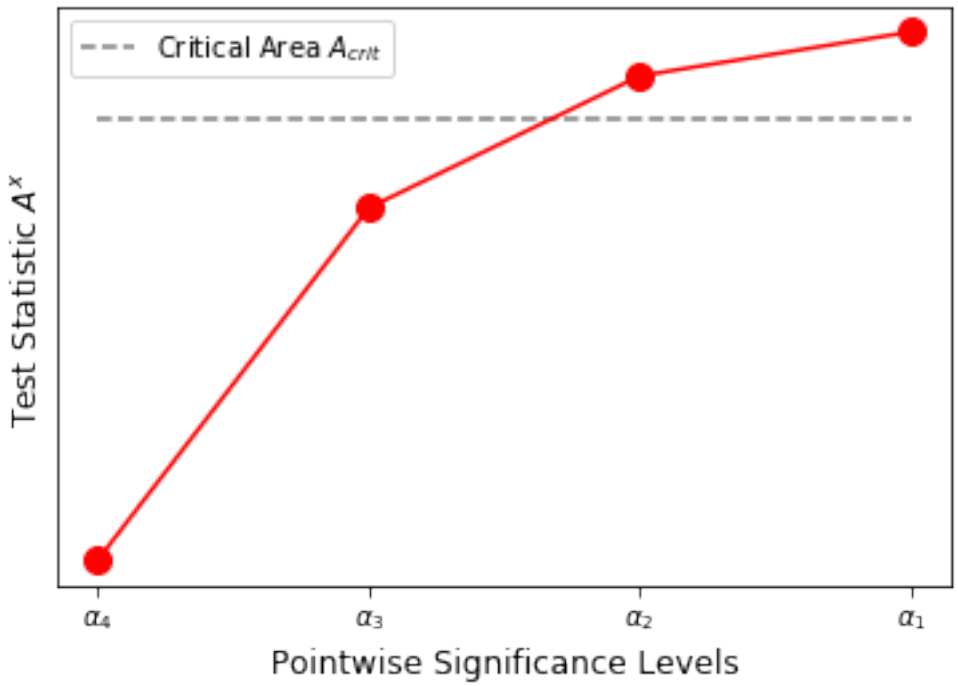



**Figure 32:** $A^x$ as it iterates over some specified pointwise significance levels. [Python 'matplotlib']

As we perform the filtration procedure, $A^x$, the test statistic of the point, might vary as shown in figure. If the final $A^x > A_{crit}$, then that point is considered cumulative areawise significant.

So how does one pick the set of pointwise significance levels? Schulte (2016) suggested $\alpha_1 = 0.02$, $\alpha_N = 0.18$,, $\Delta\alpha = 0.02$ to increase computational efficiency. It was also noted that $\Delta\alpha_1 < .02$ did not increase detection of true positives, but $\Delta\alpha_1 > 0.03$ decreased true positive detection. Also note that Schulte (2019) outlines a simplified version of the cumulative areawise testing which makes it computationally more efficient to implement. I won't discuss it in this report, but it's useful to understand how the Advanced Biwavelet R package detects contours and patches.

### 0.8.5 Other Significance Tests

Even though I didn't apply them, there are two more tests that can be applied. The first is cumulative arc-wise testing, introduced in Schulte (2019). This test is designed solely for detecting periodicities in a signal, and outperforms areawise and cumulative areawise testing in this task. The test acts on the assumption that periodicities in a signal produce long, narrow patches stretched along the time axis. Thus, we test based on the length of a significant patch, hence the name arc-wise.

This test could potentially be used to identify the characteristic period of mode changes, if it exists. However, one would have to connect numerous profile data as a single time series. Not only that, it would be best if the observation epochs are relatively continuous in time, but this is not the case for most pulsar observations.

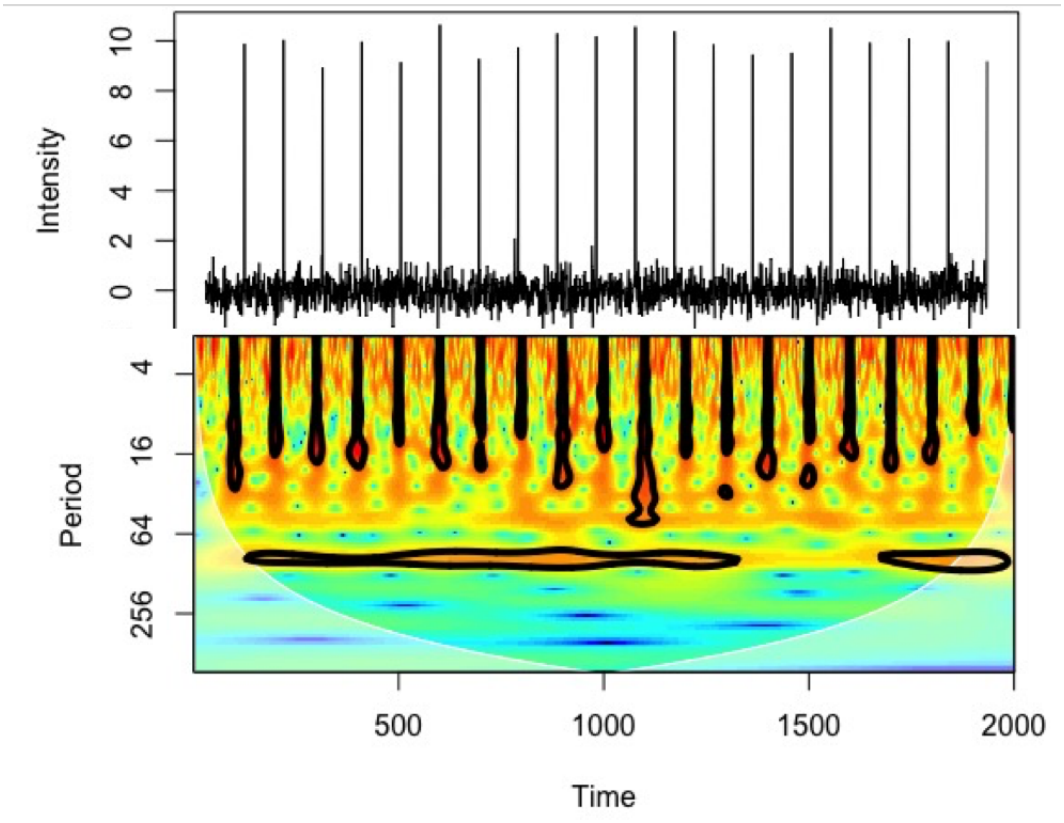



**Figure 33:** A white noise signal with a periodic singularity every 100s. A singularity produces a narrow patch stretched along the scale axis. The periodicity of the signal produces an arc along the time axis, which we know must be located at a period of 100s. Note that the contour shown is not from the application of a cumulative arcwise testing, but only pointwise. [R 'advbiwavelet']

First, one fixes a scale $a$, a pointwise significance level $\alpha_{pw}$. One applies the pointwise significance testing and identifies pointwise significant arcs at this fixed scale. Second, compute the normalized arc length, which is equal to $\frac{\text{Number of points in the arc}}{\text{fixed scale } b}$. Dividing scale accounts for the fact that at larger scales (lower frequencies), patches tend do stretch in both time and scale direction. Third, we repeat this for all scales. Finally, we repeat the previous three steps for a finite set of pointwise significance levels $\alpha_1, \alpha_2, ..., \alpha_N$ and sum them. Thus, for each $x \in H$, we have assigned a cumulative arc length, which is our test statistic. Similar to the other tests, a Monte Carlo method can be used to determine a critical arc-length.

The second test is topological significance testing, first introduced in Schulte 2015 and very well summarized in Schulte 2019. This test is ideally used together with cumulative areawise testing to extract extra information from a signal. While I won't go in detail here, there two main statistics of concern. One of them is Betti number $\beta_1$, which is the number of holes on a topological surface. Holes are regions in $H$ which are fully surrounded by points in $P_{pw}$. One can plot the mean number of holes from Monte Carlo simulation and a given signal as a function of varying pointwise significance levels to determine if the signal is distinguishable from the null distribution.

Holes are of particular interest in this report. It is explained in section WHATEVER . First, recall that spurious patches typically arise from the reproducing kernel. which is smooth. Thus, especially at low pointwise significance levels, kernels do not produce holes. However, when two fluctuations are located nearby in time and frequencies, it may cause holes in a patch. Hence, holes in a significant patch can distinguish between false positives and true positives.

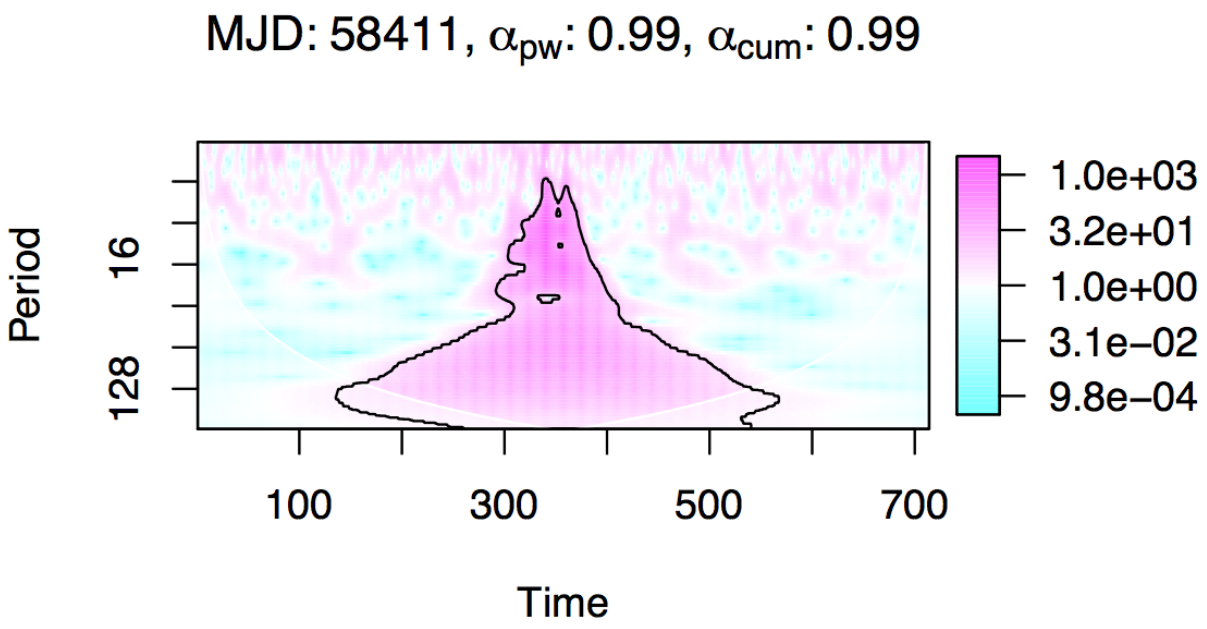



**Figure 34:** Power spectrum of the difference between two J0332+5434 profiles. Holes are visibly present, indicating a significant feature exists in the profile differences. [R 'advbiwavelet']

## 0.9 Application to J0332+5434

The papers discussed in the previous sections applied wavelet analysis in the context of climatology, whose typical null distribution is modeled by a red noise (RN). Hence, the Monte Carlo simulations of RN was generated by a first-order auto regressive model, $x_n = \rho x_{n-1} + w_n$. In general, AR($p$), an autoregressive model or order $p$, is defined as

$$x(t) = c + \sum_{i=1}^{p} \rho_i x_{t-i} + w_n$$

where $c$ is a constant, $\rho$ denotes the coefficients/parameters of the model, and $w_n$ is WN. Pure WN, which is the background noise we want in our context, can be achieved then by simply setting the coefficients to zero.

Wavelet analysis was applied in two different ways, on the ratio of profiles and the difference of profiles. The test was applied using R 'biwavelet' and 'adv_biwavelet_package'.

### 0.9.1 Ratio of Profiles

First, a null reference profile was arbitrarily chosen. Then, we took the ratio of null profile against the remaining profiles.. We then performed cumulative areawise testing. It was assumed that the profile data only contained WN, hence the null distribution of cumulative areas was generated using the ratio of WN. More specifically, the R package used only supported an autoregressive model of order 1, hence it was made as close to WN as possible by setting the lag1 autocorrelation coefficient as close to zero as possible (0.0000000001). The set of pointwise significance levels used were $\alpha_{pw} = \{.18, .16, .14, .12, .10, .08, .06, .04, .02\}$. By default, the length of noise realizations used to create null distribution of null cumulative area was 100, and the number of data points used in then null distribution was 100 as well. Only a single test was performed with $\alpha_{cum} = 0.01$

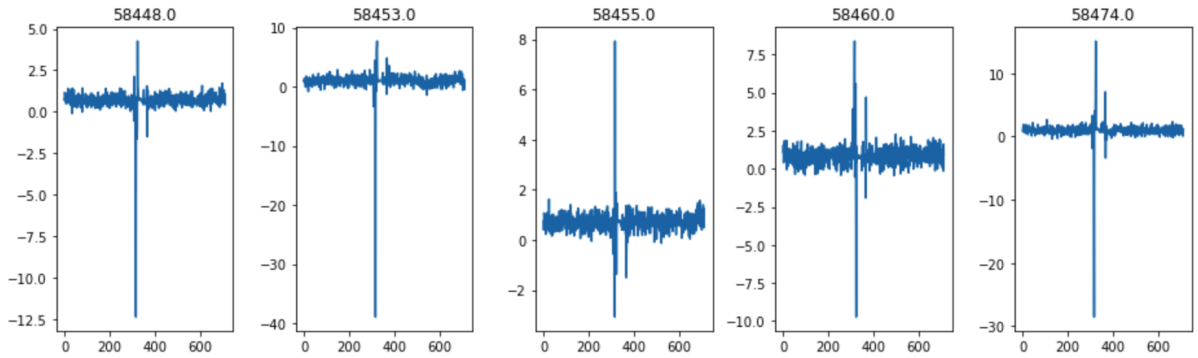



**Figure 35:** Few of the pulse profile ratios. [Python 'matplotlib']

The test identifying a majority of the profiles as significantly different from null profile (when it isn't really visually) is most likely due to the ratio data itself. As shown above, by nature of taking ratios, small deviations are exaggerated. Naturally, wavelet testing identifies many as significant. This could possibly be fixed by taking ratio of WN with greater standard deviation when calculating null distribution, or maybe not.

### 0.9.2 Difference of Profiles

Using the same null profile, we now took the difference between the null profile and the remaining profiles. All assumptions/defaults are same as the ratio testing except using the difference of WN to produce our null distribution of cumulative areas, and we tested it for three cases: $\alpha_{cum} = 0.01, 0.05, 0.1$.

A sanity check we can do is to observe the result for ratio of profile at MJD 58453. It was observed that the profile at MJD 58453 is almost identical to our null profile, as shown below. Ideally, we would expect no significant patches in the spectrum.

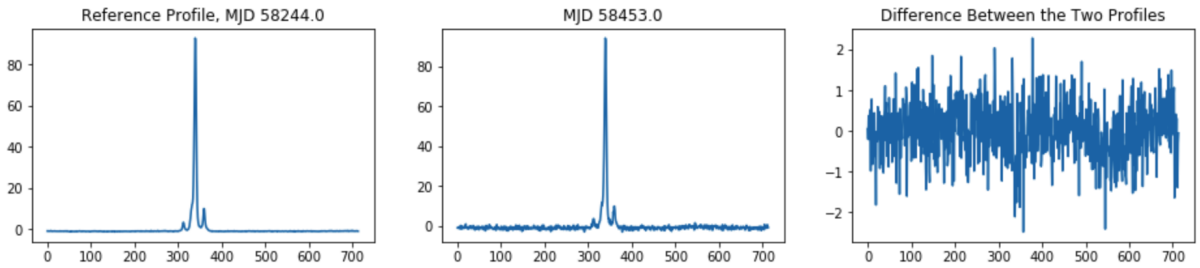



**Figure 36:** (a) Null Profile (b) Profile at MJD 58453 (c) The difference between the two profiles. The two profiles, when overplotted, are almost identical. [Python 'matplotlib']

As expected, the spectrum at MJD 58453 below shows no significant patches at $\alpha_{cum} = .01$. However, it does begin producing patches at $\alpha_{cum} = .05$ and $\alpha_{cum} = .1$. The test isn't failing, it is doing its job. As we saw in the difference plot above, there are features that aren't due to pure WN. Thus, we can conclude it is necessary to have a very strict $\alpha_{cum}$.

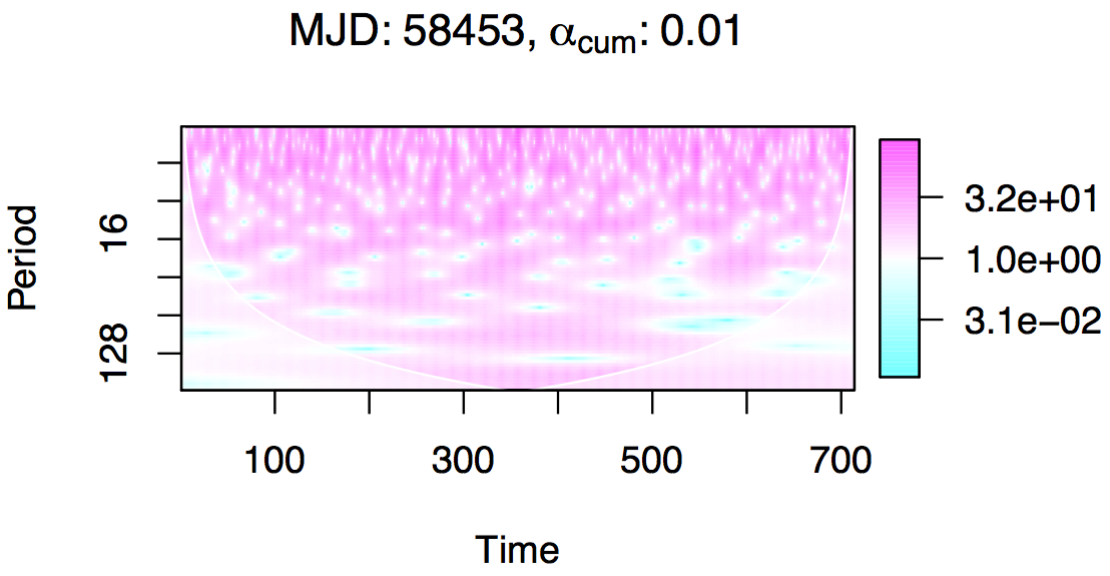



**Figure 37:** Spectrum of difference of MJD 58453 profile at $\alpha_{cum} = .01$. No significant patches as expected. [R 'advbiwavelet']

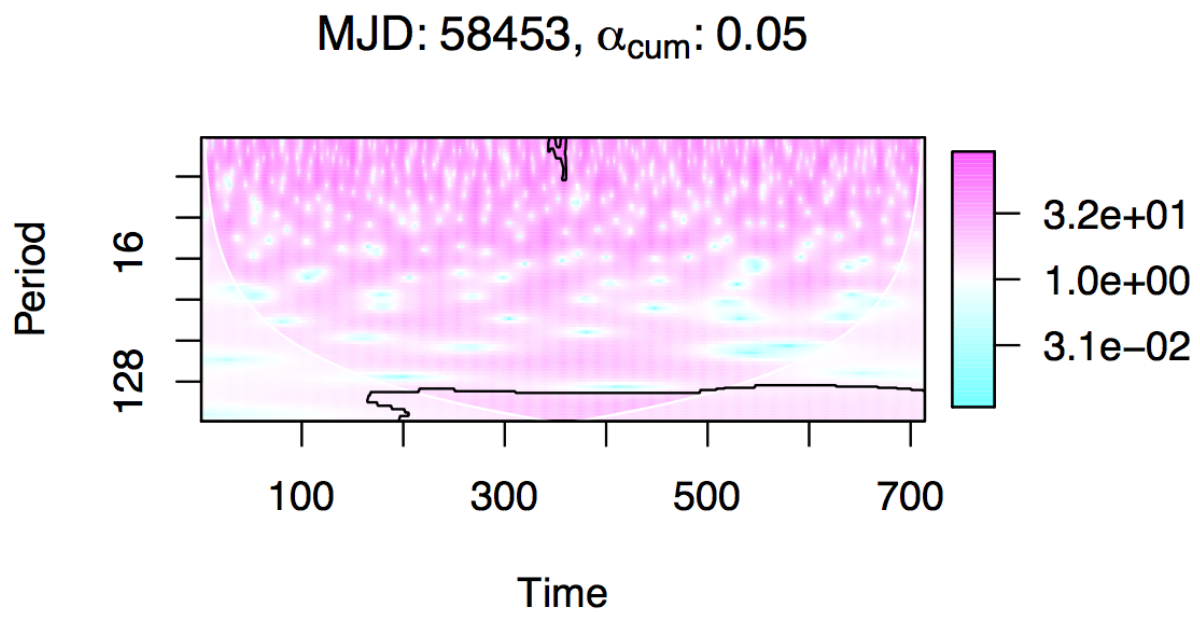



**Figure 38:** Spectrum of difference of MJD 58453 profile at $\alpha_{cum} = .05$. Spurious patches begin to appear. [R 'advbiwavelet']

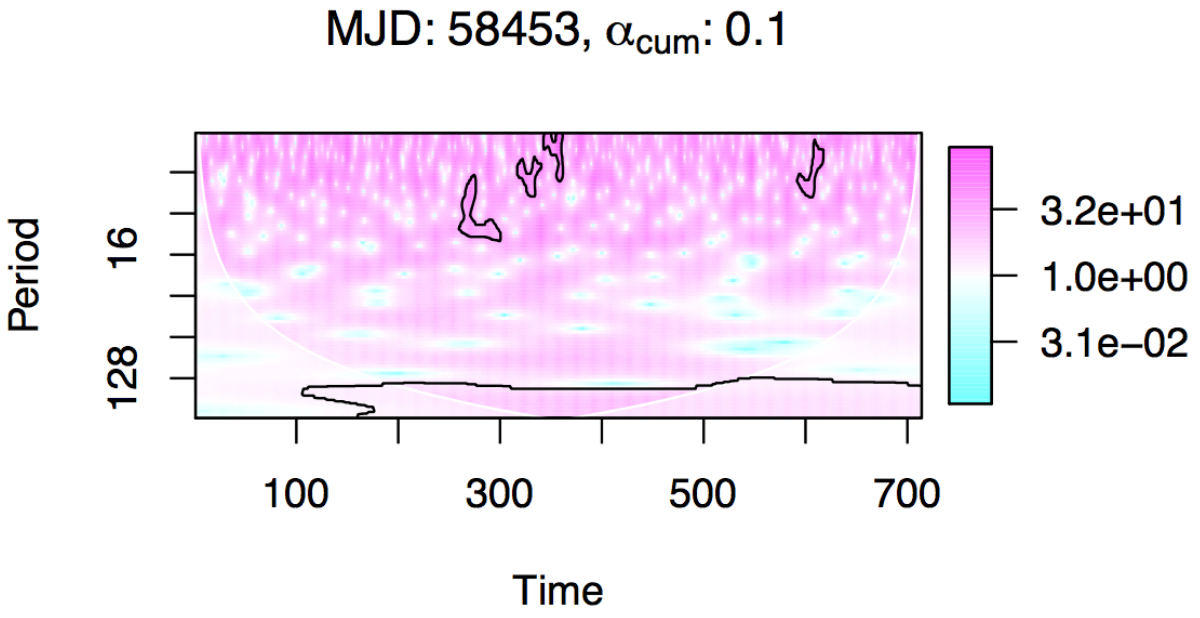



**Figure 39:** Spectrum of difference of MJD 58453 profile at $\alpha_{cum} = 0.1$. More spurious patches are born. [R 'advbiwavelet']

Now, what about the other profiles? First, take a look at the differences of profiles. Many of them have several very sharp features s on top of white noise.

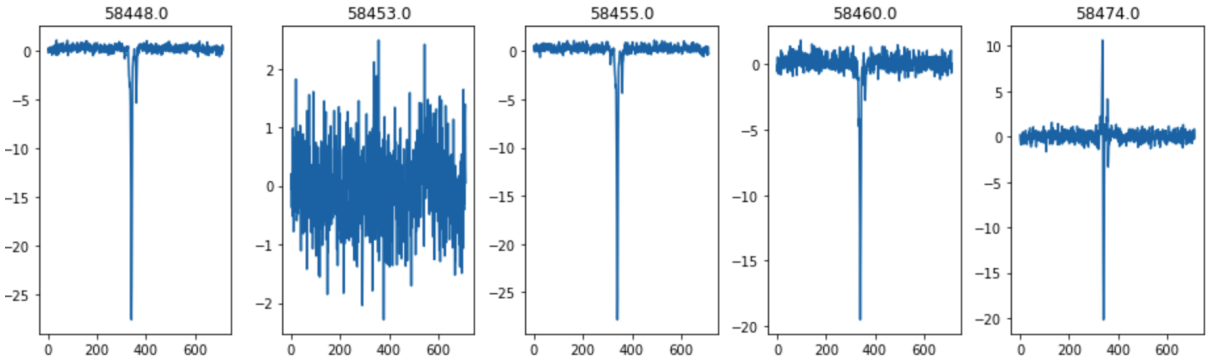



**Figure 40:** Differences in Profiles. MJD 58453 was an exception. [Python 'matplotlib']

After applying the Morlet wavelet transform and the cumulative areawise testing, almost all of the spectra displayed patch shapes which are a variation of a triangle with a stretched base. The typical spectrum is shown below.

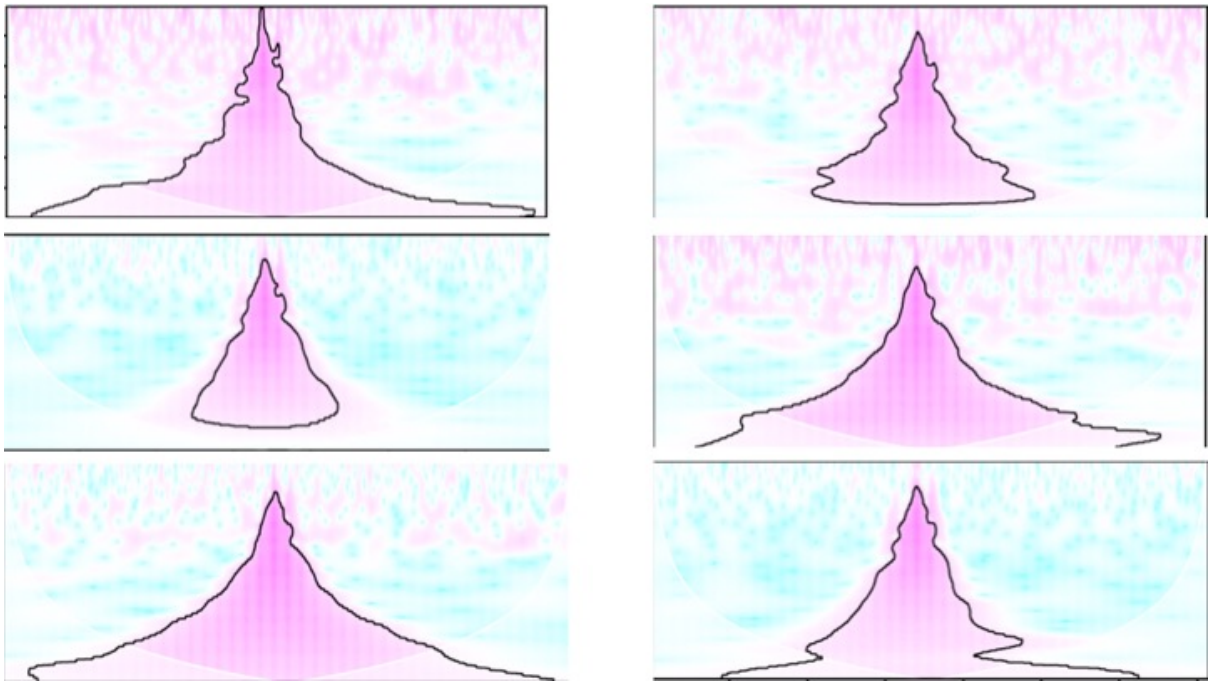
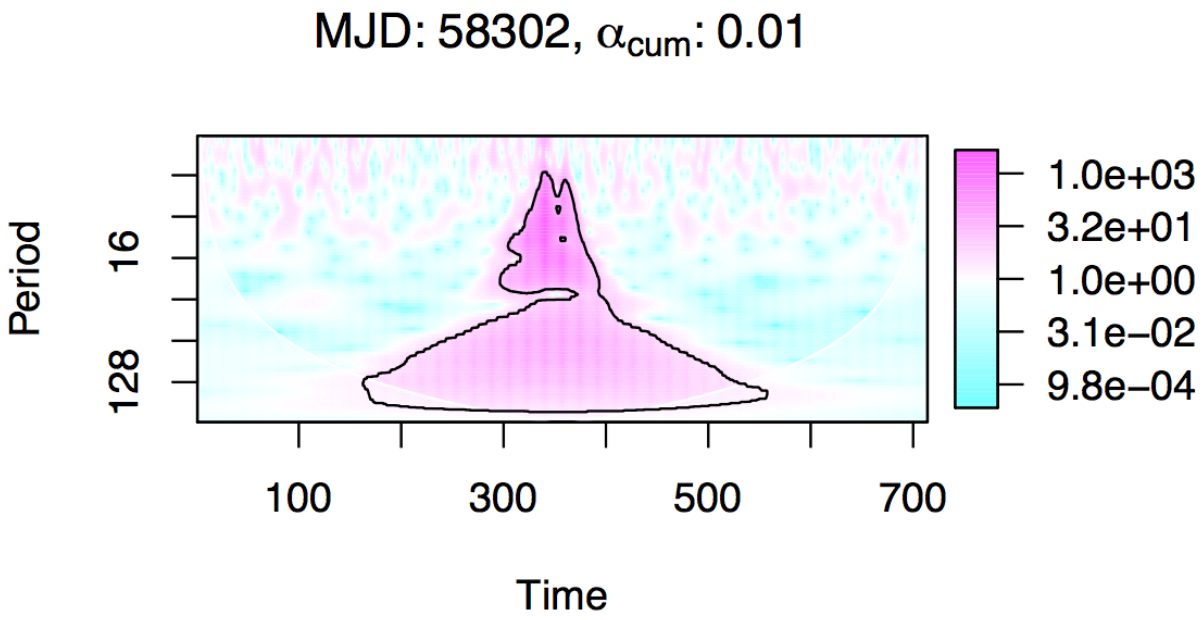
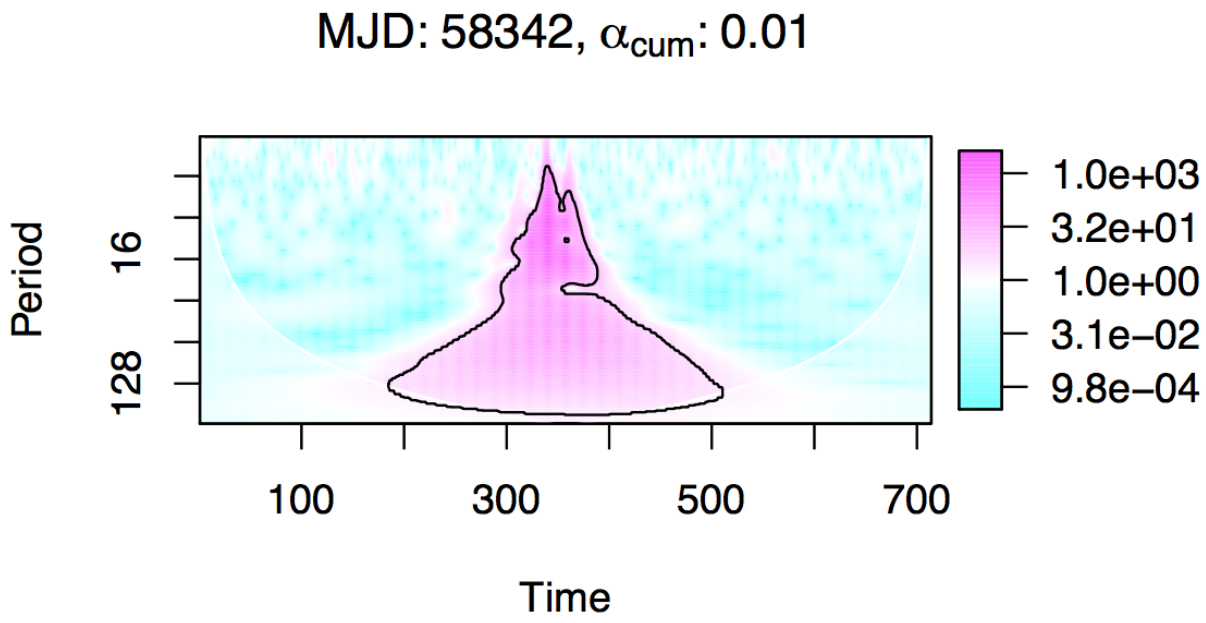


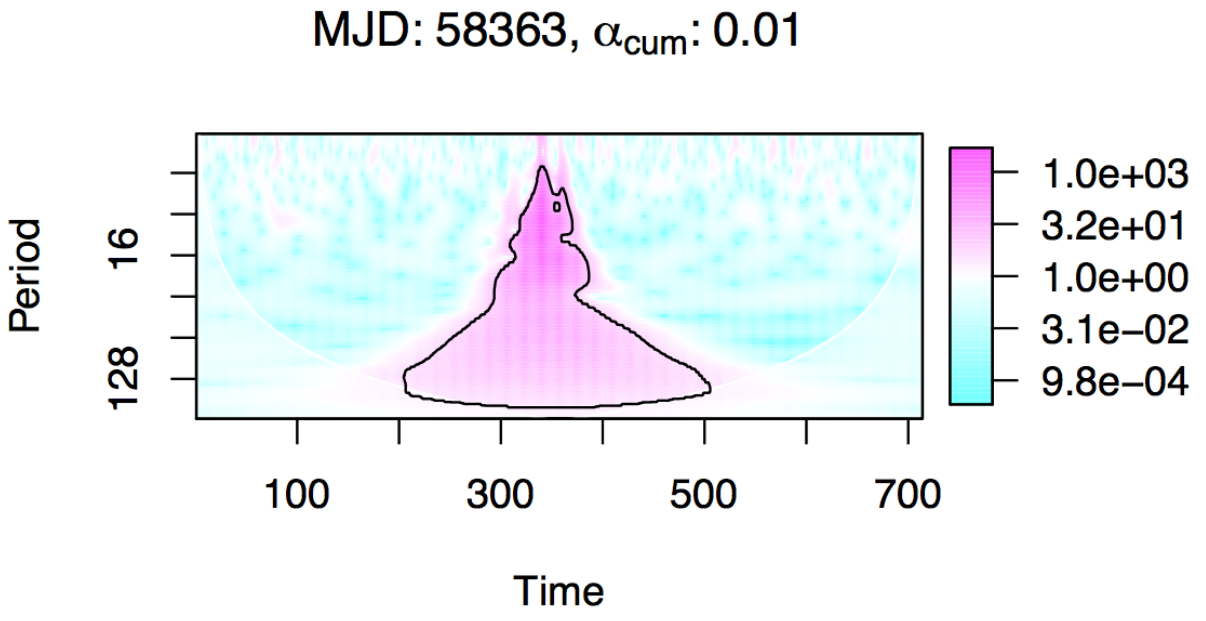
**Figure 41:** Typical spectrum of profile differences at $\alpha_{cum} = .01$. They all share some form of triangular patch. [R 'advbiwavelet']

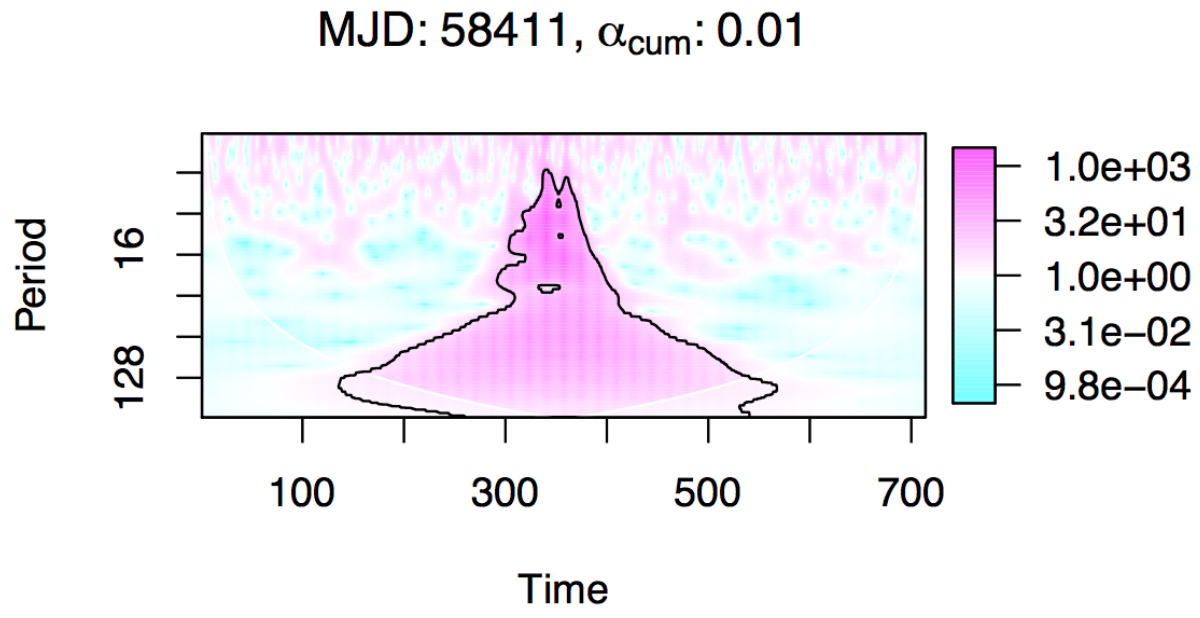
So far, not very promising. Like the testing on profile ratios, wavelet spectra don't seem to distinguish abnormal profiles. However, there is a big difference between these

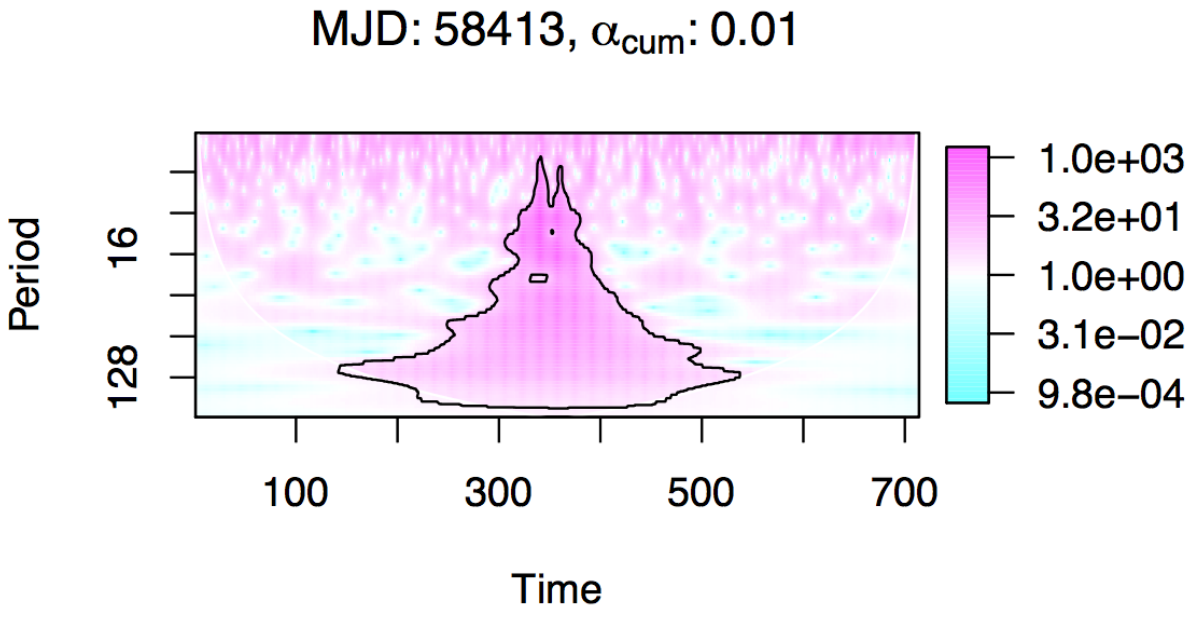
spectra and that of the ratio of profiles. While the majority of ratio spectra had holes in its patches, that is not the case for difference in profiles. In fact, there are 11 out of 61 profiles with holes, and another 4 that almost form holes. The numbers are the same for $\alpha_{cum} = .01, .05, .1$. Remembering that holes are often an indication of a significant feature, what happens if we impose a somewhat arbitrary condition that patches must have holes to be significant? It turns out then that 6 out of the 11 spectra with holes identify 6 out of 13 profiles visually classified to have a decrease in the third peak amplitude. Also, 1 out of the 4 spectra that almost form holes identify another 1 out of the 13 as well. The next six figures show the spectra which correctly identified profiles whose third peak decreases in amplitude. The four spectra which almost formed holes are also shown after. All made using R 'advbiwavelet'.




MJD: 58342, $\alpha_{cum}$: 0.01
Period
16
128
100
300
500
700
Time
1.0e+03
3.2e+01
1.0e+00
3.1e−02
9.8e−04


MJD: 58363, $\alpha_{cum}$: 0.01
Period
16
128
100
300
500
700
Time
1.0e+03
3.2e+01
1.0e+00
3.1e−02
9.8e−04


MJD: 58411, $\alpha_{cum}$: 0.01
Period
16
128
100
300
500
700
Time
1.0e+03
3.2e+01
1.0e+00
3.1e−02
9.8e−04


MJD: 58413, $\alpha_{cum}$: 0.01
Period
16
128
100
300
500
700
Time
1.0e+03
3.2e+01
1.0e+00
3.1e−02
9.8e−04

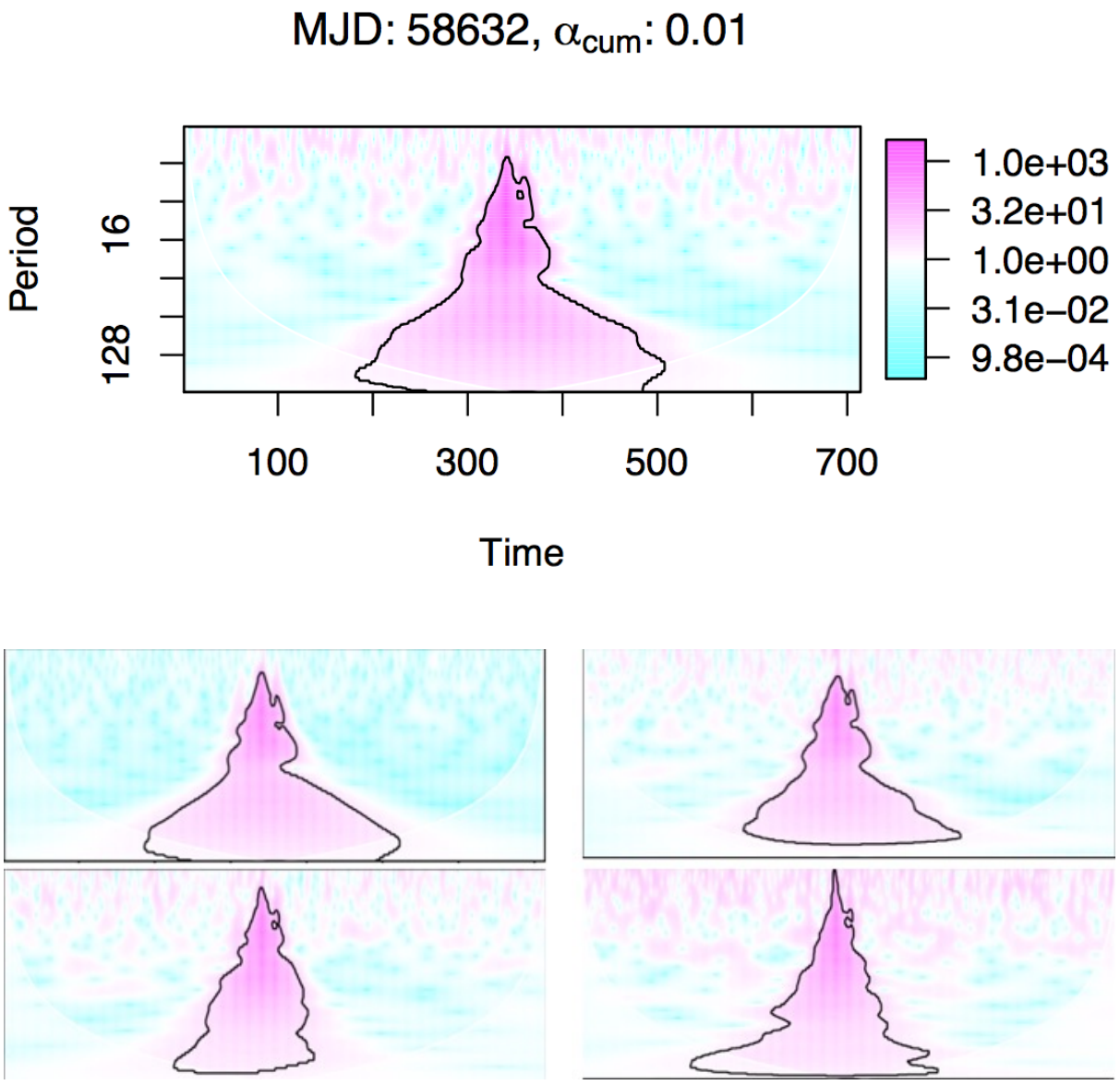



**Figure 42:** Four spectra that almost formed holes. One of them corresponds to a decrease in third peak of profile.

So does a hole indicate a mode change? Maybe, maybe not. Correctly identifying more than 50% of visually abnormal profiles seems like more than a coincidence. However, visually comparing, the other 5 profiles identified as significant don't differ from the null profile much. It's not easy to form a conclusion without fully understanding what sort of features cause holes, and what it means in terms of the difference of profiles.

However, what does seem to cause the triangular patch common to almost all spectra of profile differences seems to be the sharp Gaussian-like features in the profile differ-

ences. As standard deviation of a Gaussian feature decreases, its significance patch in the timescale domain seems to stretch more sharply along the scale domain, producing a triangle with a wide base and sharp top.

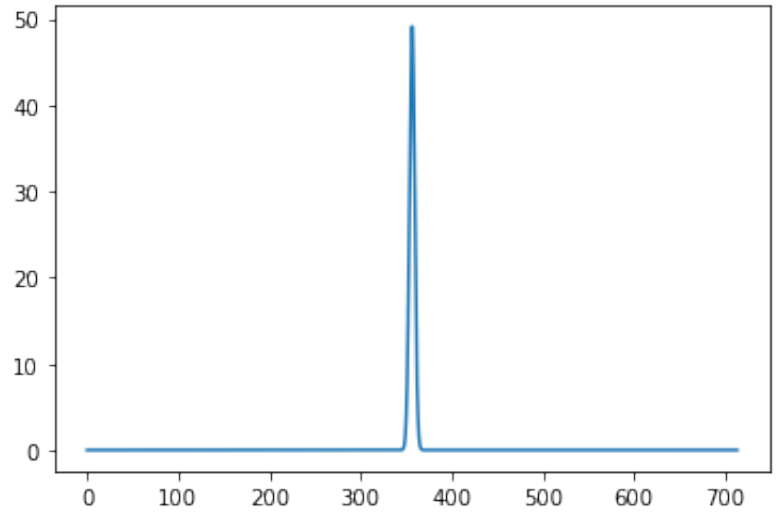



**Figure 43:** A sharp Gaussian feature similar to features commonly seen in FIGURE [Python 'matplotlib']

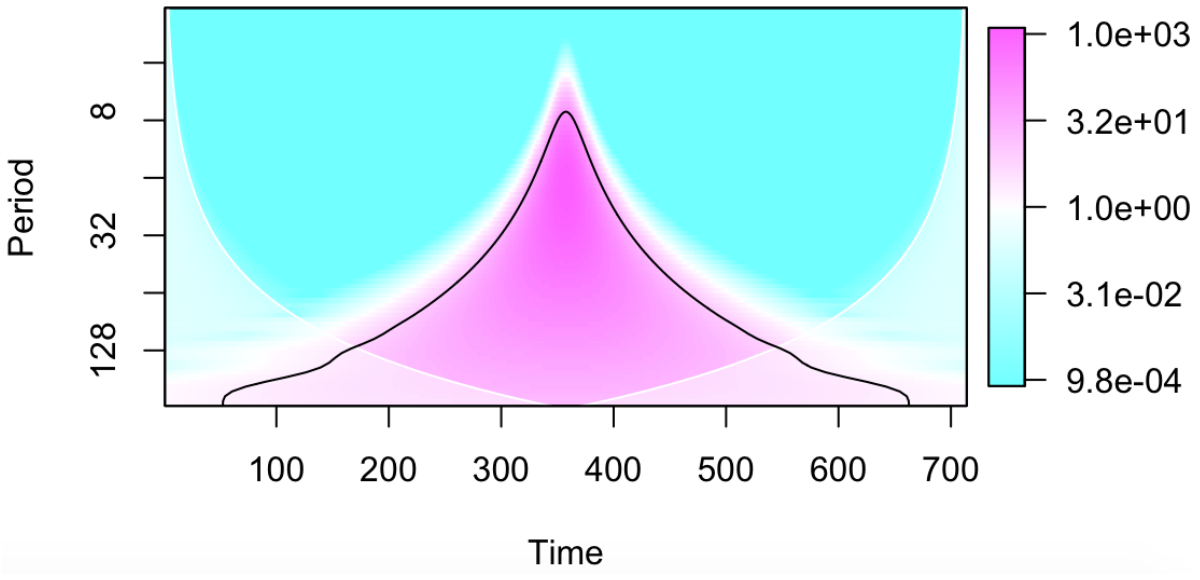



**Figure 44:** The sharp Gaussian's corresponding spectra with a patch of $\alpha_{cum} = 0.01$. It seems to reproduce the common triangle shaped patches seen in the J0332 difference spectra. [R 'advbiwavelet']

Another question we can ask is, what features in the profile differences cause holes?

There may be other, more complex causes, but one explanation is simply that holes are byproducts of several Gaussians nearby each other in the profile differences. This results in the triangular shaped patches "merging" in the timescale plane, as shown below. Figures produced with Python 'matplotlib' and R 'advbiwavelet'.

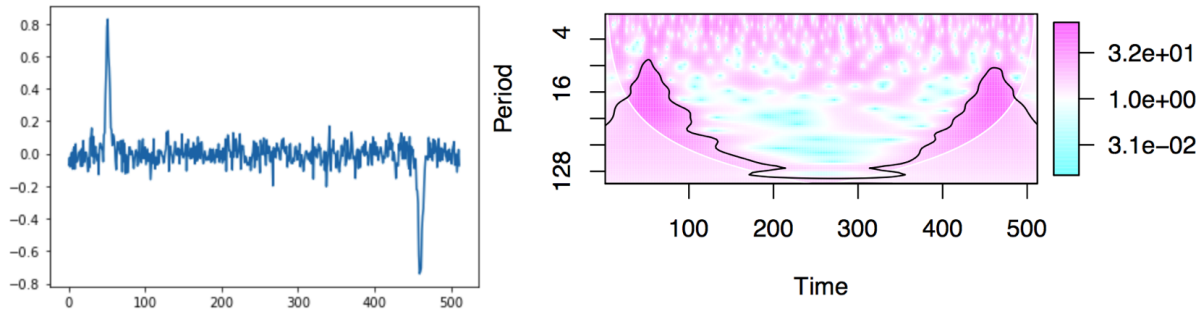



**Figure 45:** Difference between two simulated profiles, with large separation (left). Corresponding spectrum with $\alpha_{cum} = .05$ (right).

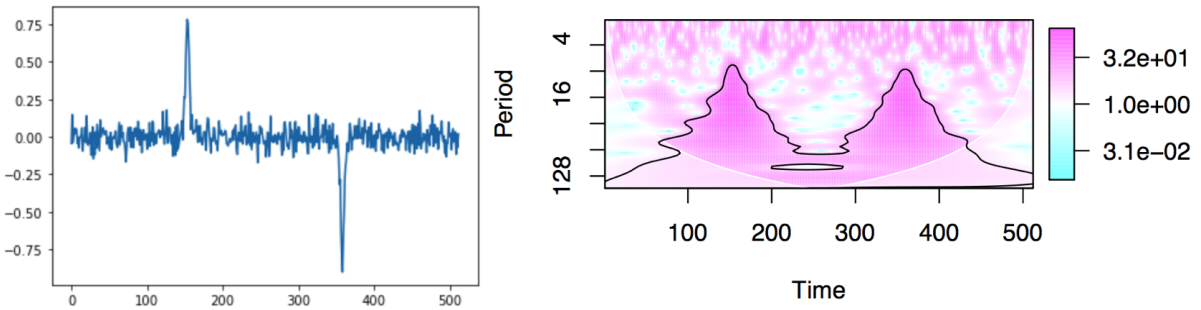



**Figure 46:** Difference between two simulated profiles, with medium separation (left). Corresponding spectrum with $\alpha_{cum} = .05$ (right).

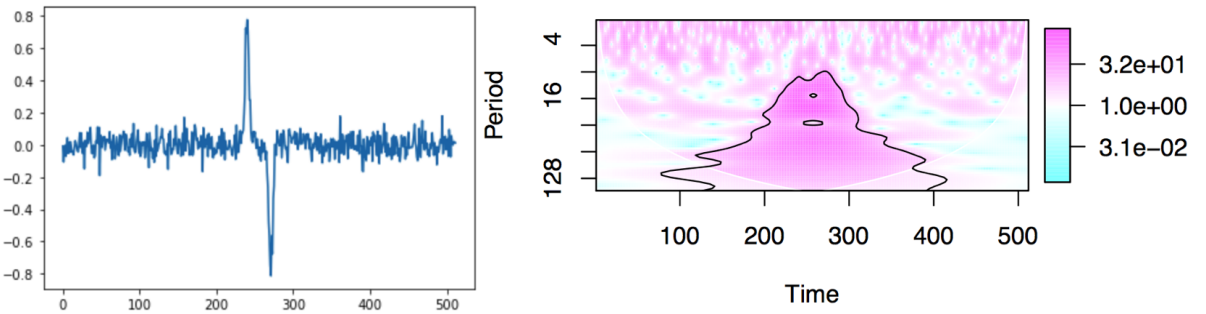



**Figure 47:** Difference between two simulated profiles, with small separation (left). Corresponding spectrum with $\alpha_{cum} = .05$ (right).

Holes form between two merging patches. A smaller standard deviation of Gaussian components in the difference results in smaller holes higher at shorter periods. Larger standard deviation of Gaussian components in the difference results in holes elongated along time, and occurring at longer periods. J0332 profile differences contained Gaussian features with small standard deviations, and small holes at shorter periods (near the top of triangular patches), supporting the idea that these features are one cause of holes.

Gaussian features in the profile differences which cause holes could truly be due to mode changes. However, it was noticed that even the smallest phase misalignment introduces holes as well, as this also produces features in the difference.

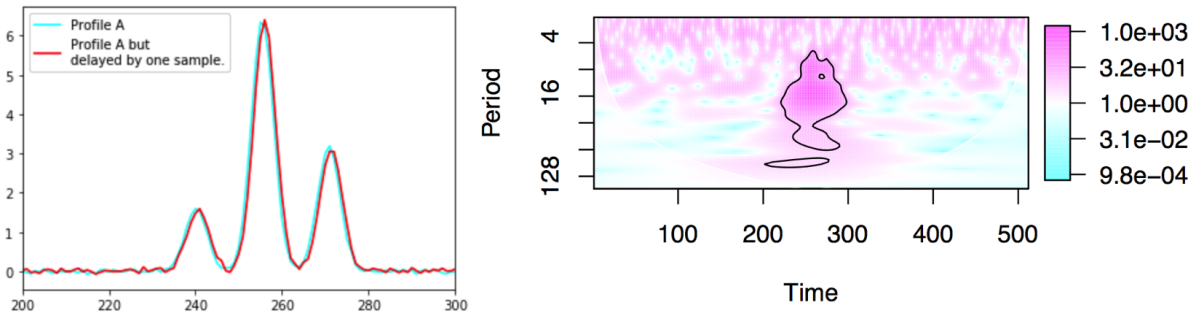



**Figure 48:** A profile but delayed one sample (left), and the spectrum with $\alpha_{cum} = .05$ (right). [Python 'matplotlib', R 'advbiwavelet']

Out of the 11 profiles identified by wavelet analysis (contains holes), one of them seems to have a minor phase misalignment. Visually, the profile's central peak does seem to have a decreased amplitude. It could be both phase misalignment and amplitude change contributing to the hole.

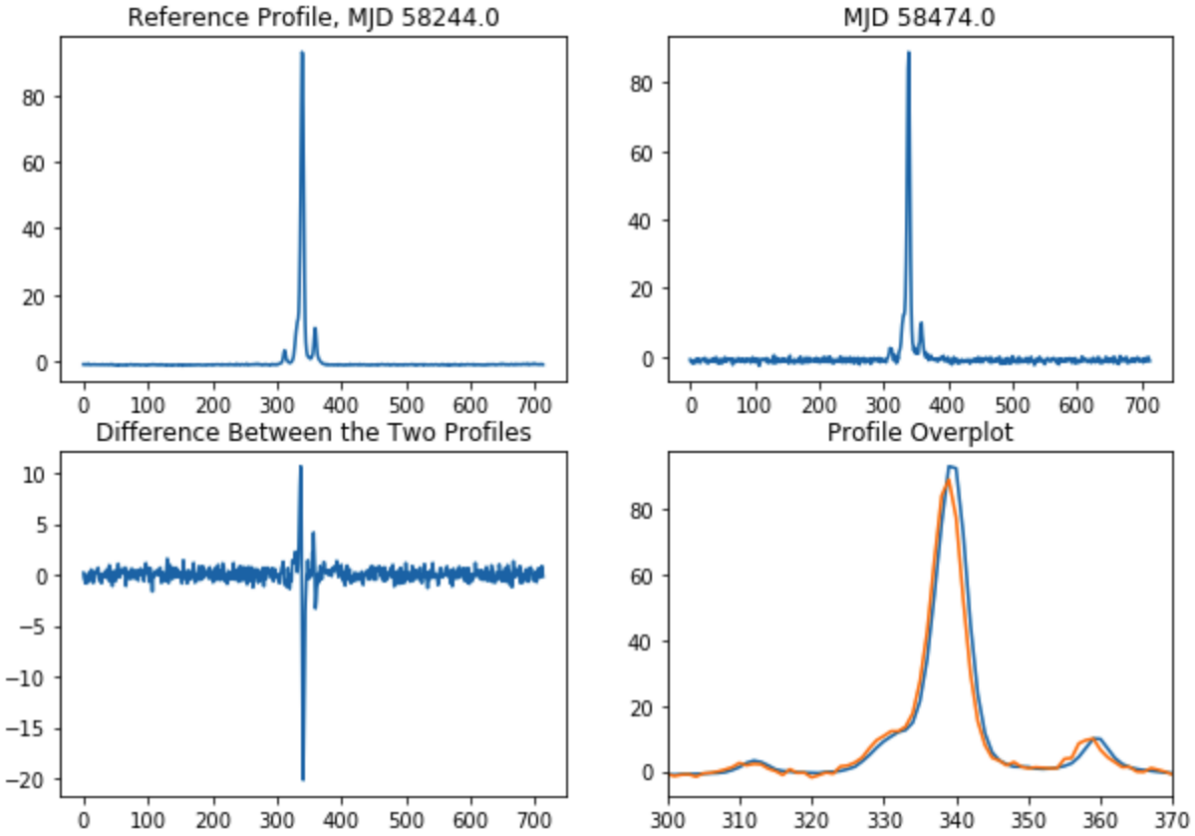



**Figure 49:** While central peak amplitude does change, a minute phase misalignment may have contributed to a hole in spectrum. [Python 'matplotlib']

Again, it's not clear whether holes indicate a mode change, is due to an artificial error, or a mix of both. The interpretation of holes is still not straightforward. However, the fact that requiring holes to be considered significant correctly identified mode changes $\frac{7}{13} > 50\%$ of the time is interesting. Also note that we are comparing all test results against visually classified mode changes, which looked for changes in amplitude of only the 3rd peak.

A way to improve results might be to think of another form of data to perform wavelet analysis on instead of difference or ratio of profile. These methods were chosen because the null distribution was being generated from WN, and so the profile data had to be manipulated in a way such that "no mode change" was equivalent to manipulated data exhibiting pure WN.

# References

[1] Walnut, David F. An Introduction to Wavelet Analysis. BirkhaÌĺuser, 2004.