TIME SERIES & SPECTRAL ANALYSIS

Design, Simulation, and Interpretation

Dao Feng
Department of Mathematics

ESSE4060
Prof. Spiros Pagiatakis
Lassonde School of Engineering
York University
December 6, 2022

Contents

1 FIRST LEVEL ANALYSIS, TIME SERIES SIMULATIONS, CODE DESIGN

Time Series Construction, Simulation, Visualization

We begin by constructing two synthetic time series from a collection of sinusoidal waves, a linear trend, and random noise.The requirements for the sinusoids are as follows:

* Frequency bands of the sinusoids are at least four octaves wide.
* Ratio between the largest and lowest amplitudes are at least 10.
* Sampling rate satisfies the inequality $\Delta < \frac{1}{2} f_{min}$, where f_{min} is the smallest frequency used.
* Length of the series should be larger than the largest period used to create the series.

Time Series I

In constructing the first time series, we begin by creating five sinusoidal waves in the form of $f_n = Acos(2\pi ft + \phi)$.

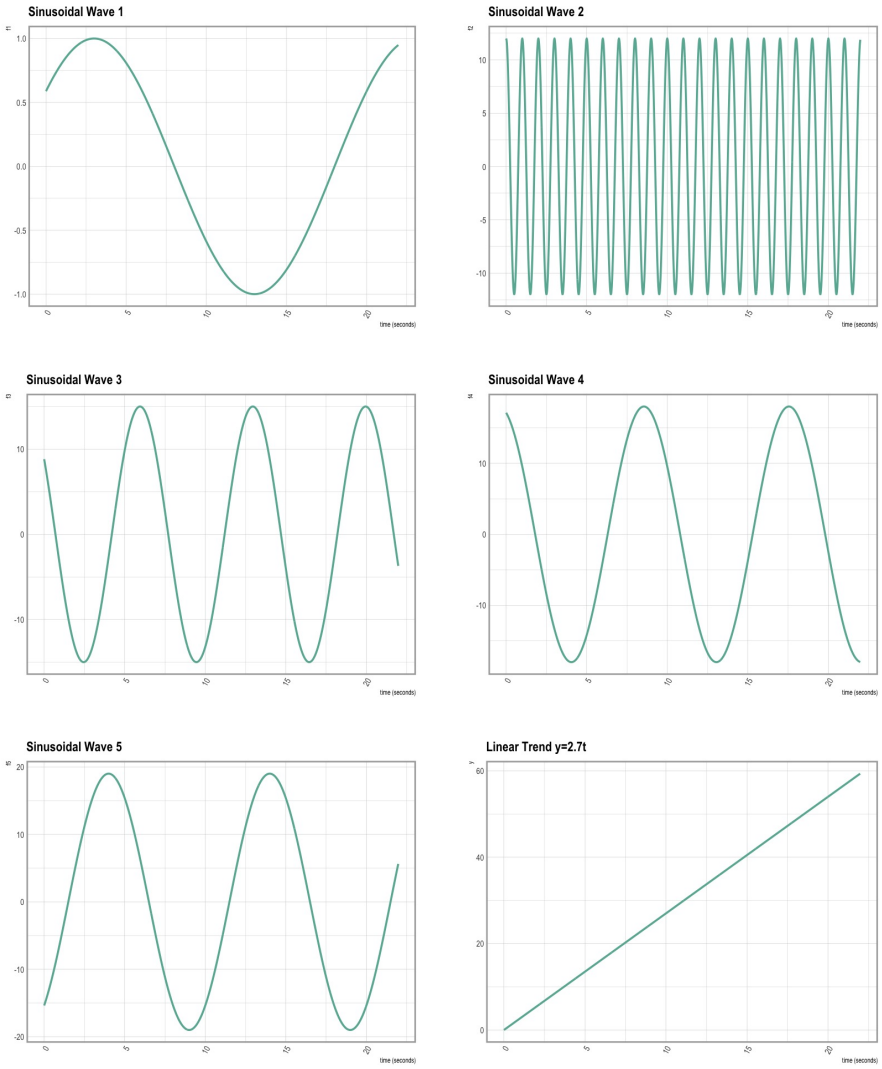
$$
\begin{aligned}
f_1 &= cos(2\pi t/20 + 1.7\pi) \\
f_2 &= 12cos(2\pi t) \\
f_3 &= 15cos(2\pi t/7 + 0.3\pi) \\
f_4 &= 18cos(2\pi t/9 + 0.1\pi) \\
f_5 &= 19cos(2\pi t/10 + 1.2\pi)
\end{aligned}
$$

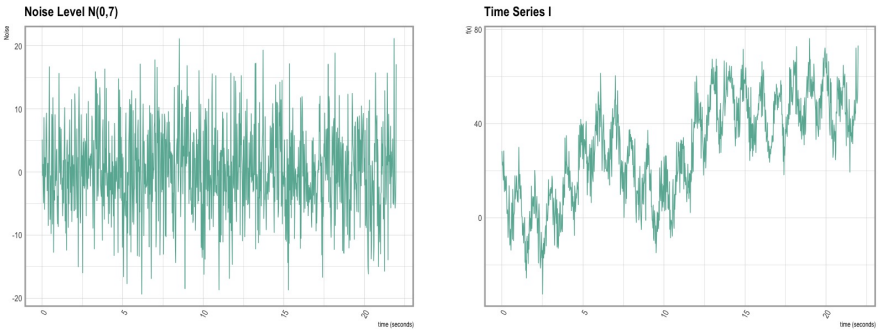
We construct a table detailing the characteristics of the waves.

f_n	Frequency (f)	Period (T)	Amplitude (A)	Phase Shift (ϕ)
f_1	1/20	20	1	1.7π
f_2	1	1	12	0
f_3	1/7	7	15	0.3π
f_4	1/9	9	18	0.1π
f_5	1/10	10	19	1.2π

We create our synthetic data using a sampling interval of $\Delta = 1/45$ satisfying the condition $\Delta < \frac{1}{2} f_{min}$. The length of the series is $t \in [0, 7\pi)$ which contains the largest period of 20 time units. Furthermore, it is evident from the table that the frequency bands of the sinusoids span over four octaves and that the ratio between A_{min} and A_{max} is greater than 10.

In addition, we will construct a linear trend of $y = 2.7t$ and randomly generate a Gaussian Noise of $N(0, 7)$.
Each sinusoidal wave can be shown graphically as follows:





Following this same process, we proceed to construct a second time series in the same manner.

Time Series II

The five sinusoidal waves in the second time series are:

$$
\begin{aligned}
g_1 &= cos(2\pi t/14 + 1.3\pi) \\
g_2 &= 5cos(2\pi t/4 + 2.5\pi) \\
g_3 &= 10cos(2\pi t/7 + 0.6\pi) \\
g_4 &= 12cos(2\pi t/8 + 1.9\pi) \\
g_5 &= 21cos(2\pi t)
\end{aligned}
$$

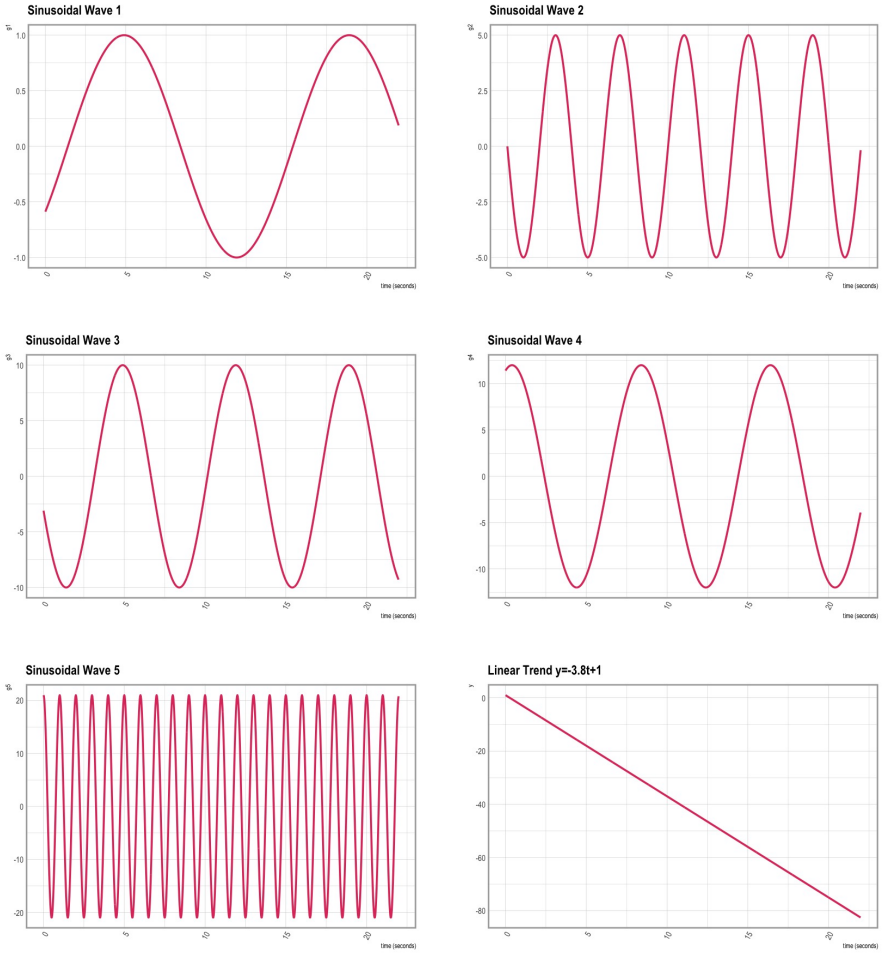
The Table of Wave Characteristics

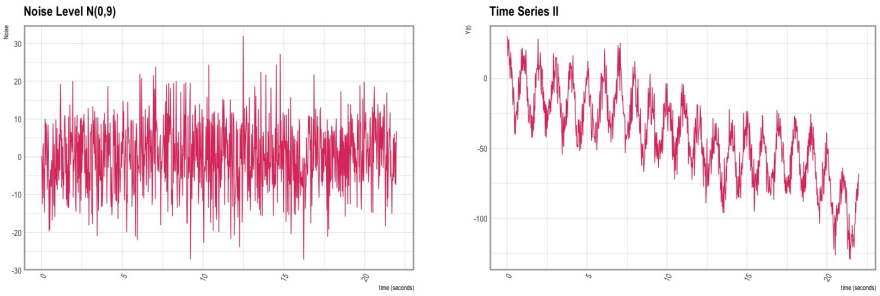
g_n	Frequency (f)	Period (T)	Amplitude (A)	Phase Shift (ϕ)
g_1	1/14	14	1	1.3π
g_2	1/4	4	5	2.5π
g_3	1/7	7	10	0.6π
g_4	1/8	8	12	1.9π
g_5	1	1	21	0

For our second time series, the sampling interval will also be $\Delta = 1/45$ in order be consistent with the previous time series. Otherwise, we will not be able to calculate cross-covariance and cross correlation. Sampling at $\Delta = 1/45$ still satisfies the condition $\Delta < \frac{1}{2} f_{min}$. The length of the series is $t[0, 7\pi)$ which contains the largest period of 14 time units. The frequency bands of the sinusoids span over four octaves and that the ratio between A_{min} and A_{max} is greater than 10.

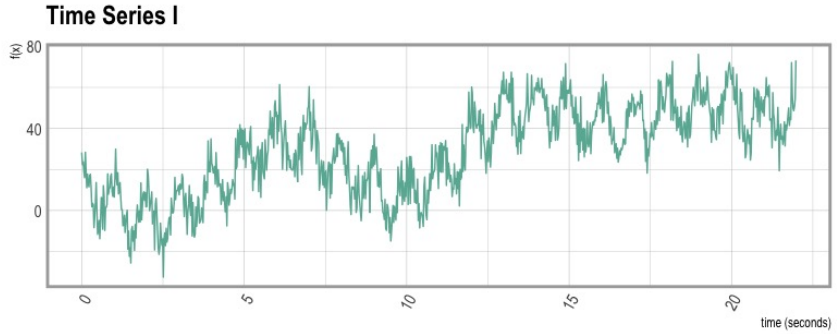
The linear trend used is $y = -3.8t + 1$ and the noise used is a randomly generated Gaussian Noise of $N(0, 9)$. The noise used for this second time series is outside the range of the two smallest amplitudes of the sinusoidal waves. Upon testing different levels, a value outside the suggested range seemed more suiting.

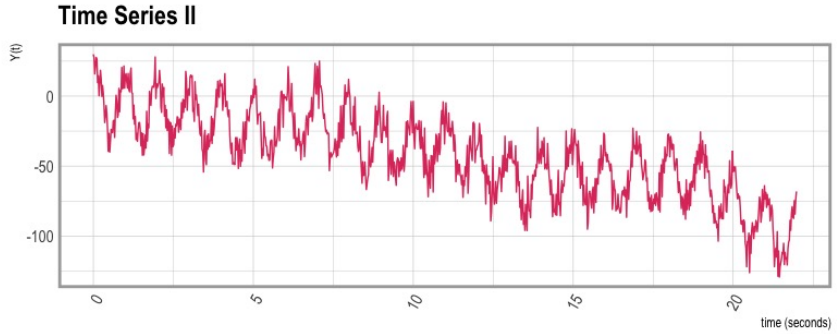
Each sinusoidal wave can be shown graphically as follows:





First Level Analysis





We will denote the first series as $x(t_i)$ and the second series $y(t_i)$ in the following analysis.

Mean & Variance

$$M_x = \frac{1}{N}\sum_{i=1}^{N} x(t_i)$$

With $\sum_{i=1}^{N} x(t_i) = 28988.04$ and $N = 990$,

$$M_x = 29.30$$

Similarily, with $\sum_{i=1}^{N} y(t_i) = -41104.21$ and $N = 990$

$$M_y = -41.52$$

$$S_x^2 = \frac{1}{N}\sum_{i=1}^{N}[x(t_i) - M_x]^2$$

$$S_x^2 = 464.94$$

$$S_y^2 = 944.15$$

Quadratic Norm & Power

$$P = \frac{1}{N}\sum_{i=1}^{N} x^2(t_i)$$

Power of $x(t_i)$:

$$P = 1322.31$$

Power of $y(t_i)$:

$$P = 2668.01$$

Auto-Covariance & Auto-Correlation Function

Auto-Covariance of $x(t_i)$:

$$C_{xx}(\tau) = \frac{1}{N-\tau}\sum_{i=1}^{N-\tau}[x(t_i) - M_x][x(t_i+\tau) - M_x]$$

$$C_{xx}(\tau) = 416.55$$

Auto-Correlation $x(t_i)$:

$$C_{xx}(0) = \frac{1}{N}\sum_{i=1}^{N-\tau}[x(t_i) - M_x][x(t_i + 0) - M_x] = \frac{1}{N}\sum_{i=1}^{N}[x(t_i) - M_x]^2 = S_x^2$$

$$C_{xx}(0) = 464.94$$

$$-1 \leq \frac{C_{xx}(\tau)}{C_{xx}(0)} \leq 1$$

$$-1 \leq \frac{416.5476}{464.94} \leq 1$$

$$-1 \leq 0.8959103 \leq 1$$

$$\rho_{xx} = 0.896$$

Auto-Covariance of $y(t_i)$:

$$C_{yy}(\tau) = \frac{1}{N-\tau}\sum_{i=1}^{N-\tau}[y(t_i) - M_y][y(t_i + \tau) - M_y]$$

$$C_{yy}(\tau) = 863.68$$

Auto-Correlation $y(t_i)$:

$$C_{yy}(0) = \frac{1}{N}\sum_{i=1}^{N-\tau}[y(t_i) - M_y][y(t_i + 0) - M_y] = \frac{1}{N}\sum_{i=1}^{N}[y(t_i) - M_y]^2 = S_y^2$$

$$C_{yy}(0) = 944.15$$

$$-1 \leq \frac{C_{yy}(\tau)}{C_{yy}(0)} \leq 1$$

$$-1 \leq \frac{863.6845}{944.1492} \leq 1$$

$$-1 \leq 0.9147755 \leq 1$$

$$\rho_{yy} = 0.915$$

Cross-Covariance & Cross-Correlation

$$C_{XY}(\tau) = \frac{1}{N} \sum_{i=1}^{N-\tau} [x(t_i) - M_x][y(t_i + \tau) - M_y]$$

$$C_{XY}(\tau) = -249.73$$

$$\rho_{XY} = \frac{C_{XY}(\tau)}{\sqrt{C_{xx}(0)C_{yy}(0)}}$$

where

$$\rho_{XY} = \frac{-249.73}{\sqrt{(464.94)(944.15)}}$$

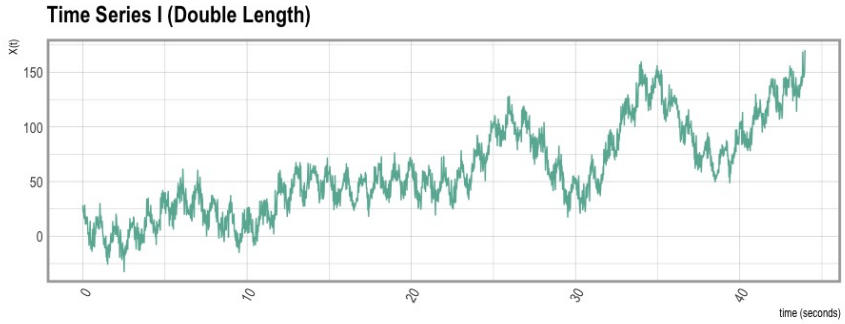
$$\rho_{XY} = -0.377$$

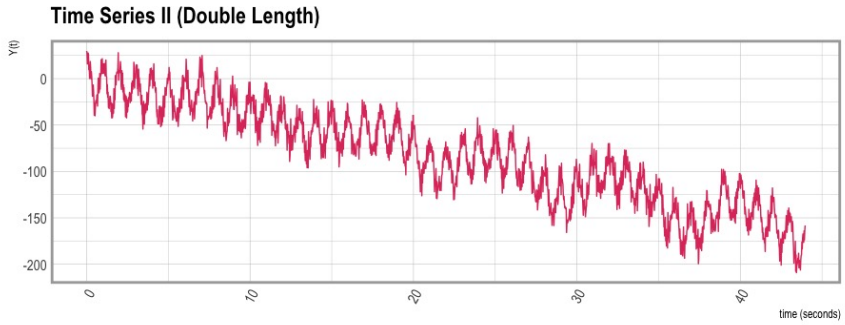
such that:

$$-1 \leq \rho_{XY} \leq 1$$

$$-1 \leq -0.377 \leq 1$$

Extended Length Time Series & Comparisons





By increasing the duration of both time series to double the original length, we repeat the calculations from above for both time series.

Mean and Variance

$$M_x = 59.50181 \qquad M_y = -82.61135$$
$$S_x^2 = 1662.984 \qquad S_y^2 = 22909.95$$

Power

$$M_x = 59.50181 \qquad M_y = -82.61135$$
$$S_x^2 = 1662.984 \qquad S_y^2 = 22909.95$$

Power of $x(t_i)$:

$$P = 5203.45$$

Power of $y(t_i)$:

$$P = 9538.43$$

Auto-Covariance

Auto-Covariance of $x(t_i)$:

$$C_{XX}(\tau) = 1612.533$$

Auto-Covariance of $y(t_i)$:

$$C_{YY}(\tau) = 2632.069$$

Auto-Correlation

Auto-Correlation of $x(t_i)$:

$$\rho_{XX} = 0.9696625$$

Auto-Correlation of $y(t_i)$:

$$\rho_{YY} = 0.1148876$$

Cross-Covariance

Cross-Covariance of $x(t_i)$ and $y(t_i)$:

$$C_{XY}(\tau) = -1493.801$$

Cross-Correlation

Cross-Correlation of $x(t_i)$ and $y(t_i)$:

$$\rho_{XY} = -0.242012$$

Comparisons and Interpretations

It is noticeable that as the duration is increased by double the original amount, there are drastic changes to the summary statistics of our time series. We see that in the first series $x(t_i)$, there are more intense vertical fluctuations in the extended duration.

For the second time series $y(t_i)$, we do not observe obvious fluctuations like the first one after extending the duration. We notice a continuous trend downwards which is largely contributed by the negative linear trend included in the construction. If we isolate other components of the series to exclude the linear trend component, the graph of $y(t_i)$ displays more stable cyclical or periodical fluctuation which suggests a more patterned behavior compared to $x(t_i)$.

What is interesting is that upon increasing the duration, the auto-correlation of $x(t_i)$ exhibited stronger auto-correlation while $y(t_i)$ showed drastically lower auto-correlation. Furthermore, after extending the duration, we arrive at a cross-correlation value that suggests the two series are becoming less correlated. This is visually shown from the graph that $x(t_i)$ is trending upwards while $y(t_i)$ is trending downwards. As the two series diverge in opposite directions, this can be expected due to the stable pattern of $y(t_i)$ and the increased fluctuations in $x(t_i)$ after the first half of the duration observed.

For both series $x(t_i)$ and $y(t_i)$, there is a increase in Power of the series. This is not surprising as by doubling the length of the series, we have twice as many points of observations and thus the term $\sum_{i=1}^{N} x^2(t_i)$ would be comparatively larger and thus yielding a larger power.

In the original duration frame of both series, we notice a high variance in both cases. This suggests that statistically these two series could be hard to generalize and that the values vary from the mean to a high extent. The second time series $y(t_i)$ has a high auto-correlation of $\rho_{YY} = 0.915$ which suggests that there is a high degree of similarity between the series to itself with the lag factor of τ. Furthermore, given that the two series have trends in opposite directions, it is as expected that in our calculations we have a negative value for cross-correlation between the two series.

2 FOURIER ANALYSIS

Fourier Transformation

Using the two Time Series created from the previous section, we apply a Fast Fourier Transform Algorithm (FFT) to the two series' at both original-length and the extended double-length.

The wave characteristics used in the construction of the two series are as follows:

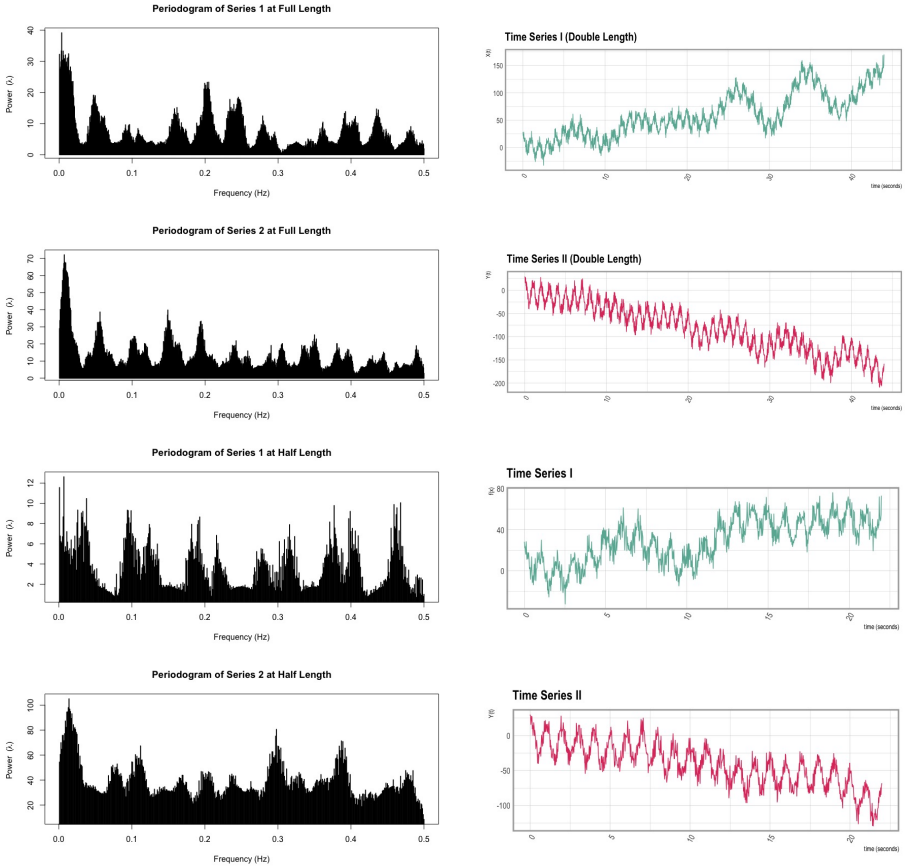
Time series I

f_n	Frequency (f)	Period (T)	Amplitude (A)	Phase Shift (ϕ)
f_1	1/20	20	1	1.7π
f_2	1	1	12	0
f_3	1/7	7	15	0.3π
f_4	1/9	9	18	0.1π
f_5	1/10	10	19	1.2π

Time series II

g_n	Frequency (f)	Period (T)	Amplitude (A)	Phase Shift (ϕ)
g_1	1/14	14	1	1.3π
g_2	1/4	4	5	2.5π
g_3	1/7	7	10	0.6π
g_4	1/8	8	12	1.9π
g_5	1	1	21	0

Periodograms



When doubling the length of the series we introduce more data points. Theoretically, the sample of the discrete time series will capture the original continuous series more accurately when the number of sampled points are increased.

A relatively higher value of power indicates more significance for the frequency in explaining the oscillation of the series.

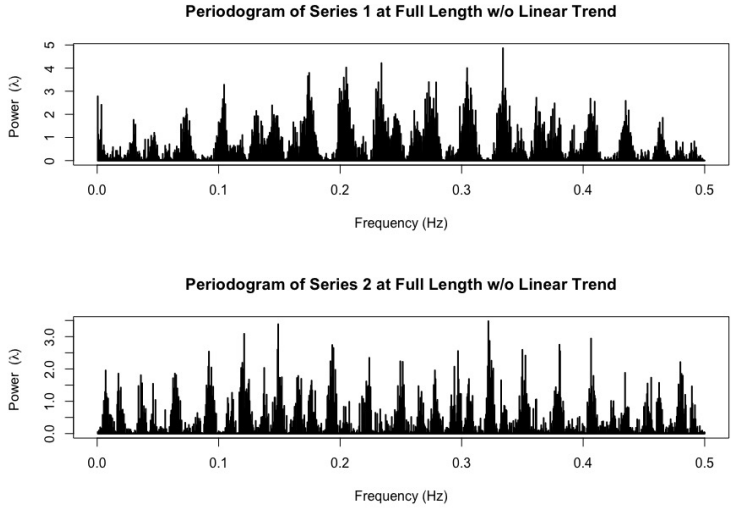
In our case, for the first series $x(t_i)$ at extended length, we are able to see the dominant peaks of the sinusoidal waves at different frequency values. Despite not being able to observe frequencies of f_2, the other frequencies of our sinusoids are observable. It is noticeable that our fundamental frequency f_1 at $0.05Hz$ is the most dominant peak. Similarily, this is also the case for the series in its

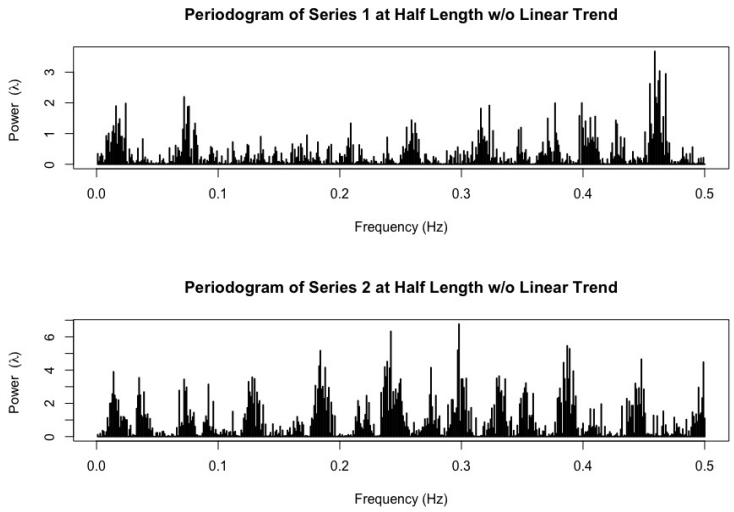
original length.

For the second series at extended length, we are not able to observe the frequency of g_5. The most dominant frequency can be observed in its fundamental frequency at approximately $1/14Hz$.

We can observe that by doubling the length of the series, there is a noticeable difference in smoothing of the periodograms. This difference is observable in both Series I and Series II. Furthermore, by doubling the lengths we see that the series' at extended lengths have more apparent peaks in comparison to the original length periodograms. We can also see a significant decrease in the number of ripples around the main dominant peaks when the length is increased. Therefore, in our case the duration of the series does contribute to the resolvability of peaks. If we were to change frequencies in our simulation to be similar to another, then we are likely to see a peak resolve in a manner such that they either cancel each other out or stack together and create a greater amplitude at that frequency.

Removing Linear Trends





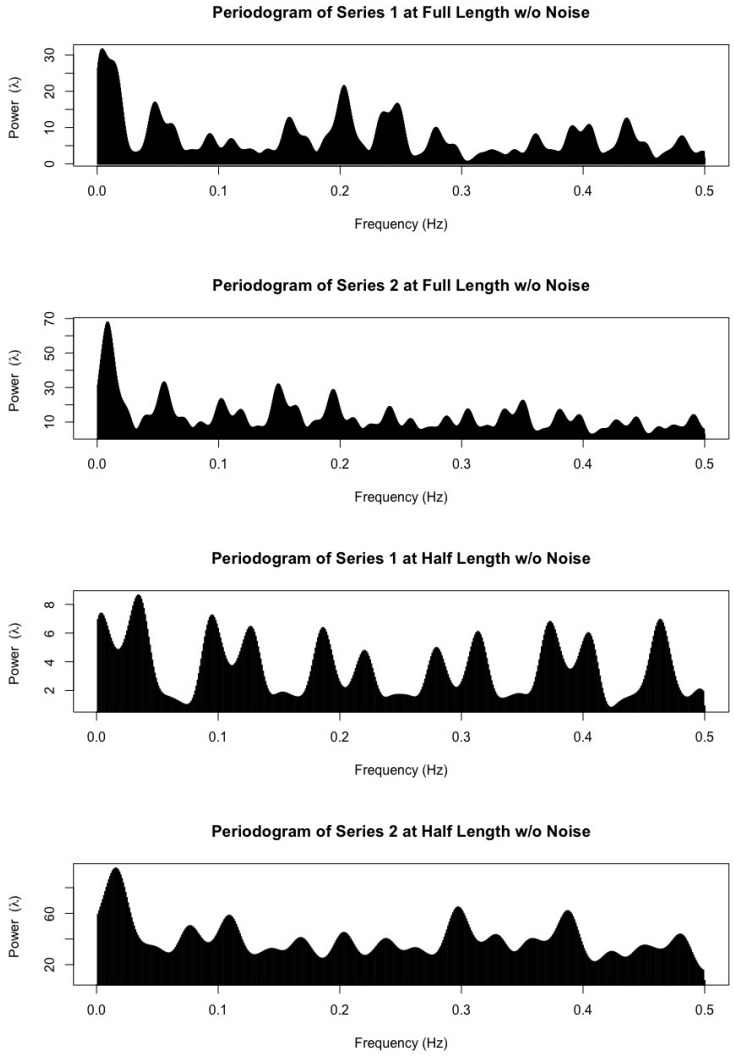
Upon Removing the linear trend, we look to re-examine the periodograms.

For the first series at extended length, after removing the linear trend, there is a significant difference compared to the original length graph. We see that the most dominant peak is now shifted to about $0.33Hz$ and that the graph is overall more observable in terms of oscillation pattern and trends. At half-length we see that the most dominant peak is at about $0.47Hz$ which is drastically different from our previous observations. The extended length series have peaks that are relatively more resolved while more ripples can be observed at original length. Compared to our graph previously which includes the linear trend, the ripples are less intense when doubled.

For the second series at extended length, we are able to observe the waves with less intense ripples around the main peaks and that the peaks are high in resolvability. The periodogram at double length seems to fill in uncaptured peaks in the original length graph. This is particularly noticeable between the frequency of $0.1Hz$ to $0.2Hz$.

Removing Noise

We proceed to remove the noise in the series and re-evaluate the periodogram of the two series at varying lengths.



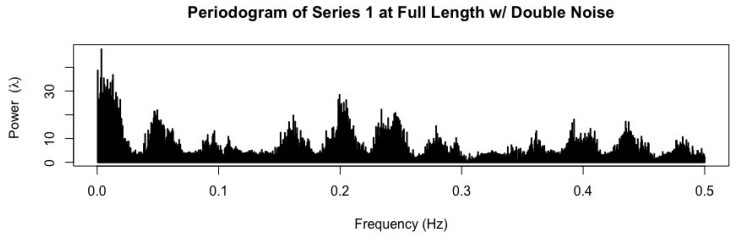
Upon removing the noise we are able to observe a smoothing over all the series. The periodogram now shows the series in bi-model and uni-modal distributions where the peaks are presented in a cyclical pattern that can be observed

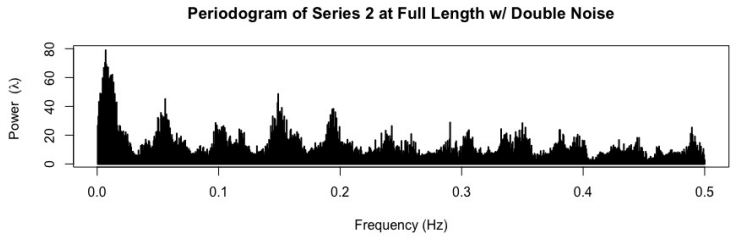
throughout the frequencies. By removing the noise, the repetitive nature of the series is better captured in the periodogram.

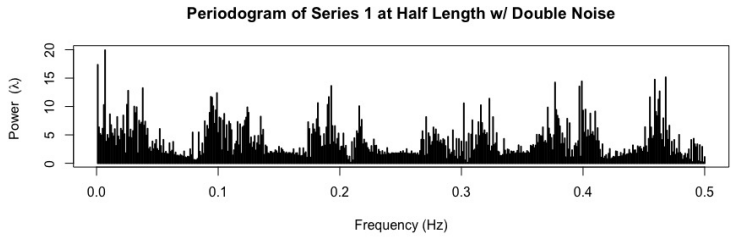
For the first series we see that the cyclical pattern is more apparent at original length. In the case of the second series, we see that the pattern can be observed at both durations.

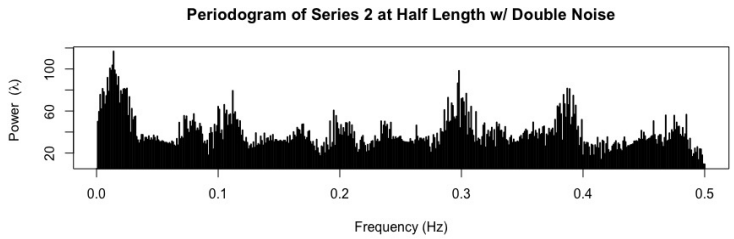
Double Noise

In contrast to the previous section, we will now proceed to double the noise effect present in all the series' and re-evaluate their respective periodograms at varying lengths.







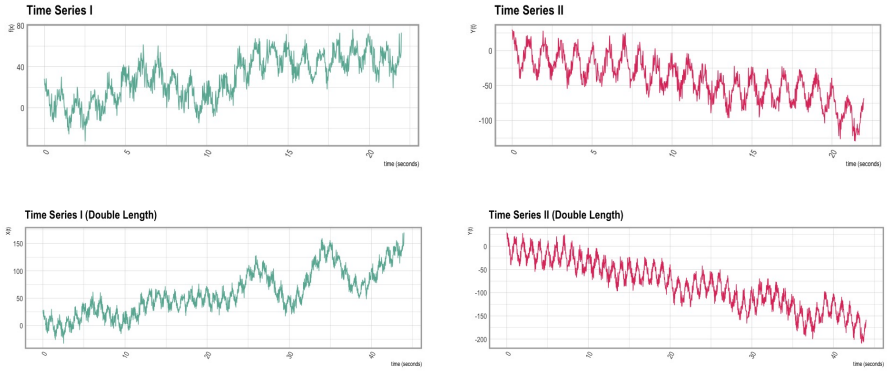


By increasing the random noise effect by a factor of two, the smoothing effect presented previously by removing the noise has disappeared. There is a significant increase in terms of the rippling effect. In all our graphs we can see an increase in ripples when the noise is doubled. Comparing the original length to the extended length graphs, we notice that the ripples are more intense for the original length series.

3 WIENER-KHINCHIN THEOREM AND POWER SPECTRAL DENSITY

Power Spectra

Using the same two Series we have constructed in previous sections (as shown below) of both original length and extended length, we will apply the Wiener-Khinchin Thereom using the auto-covariance function to find the Power Spectra of the series'.

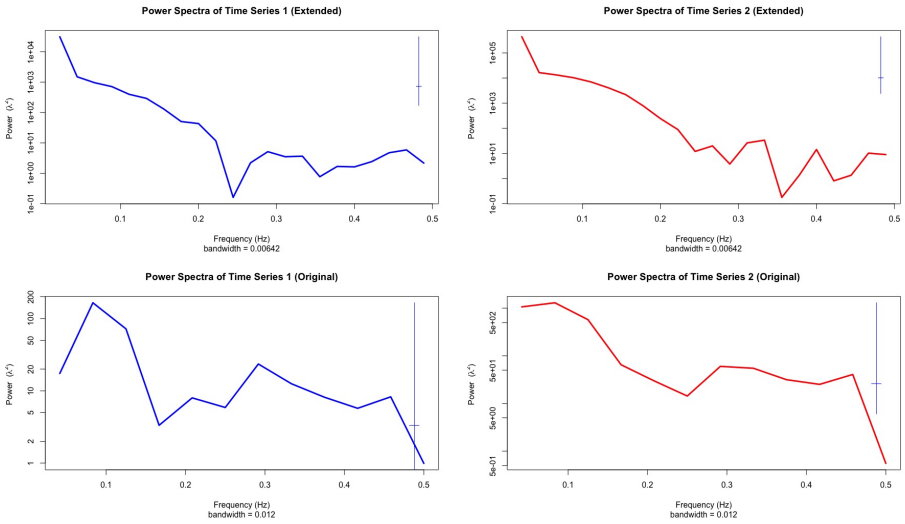


By the Wiener-Khinchin Thereom,

$$PS(f) = C_{xx}(f) = \int_{-\infty}^{\infty} C_{xx}(\tau) e^{-i2\pi f\tau} d\tau$$

We calculate the auto-covariance functions for all our series' and transform them from the time domain to the frequency domain using the Fast Fourier Transform algorithm to obtain the Power Spectra.

The results are as follows:

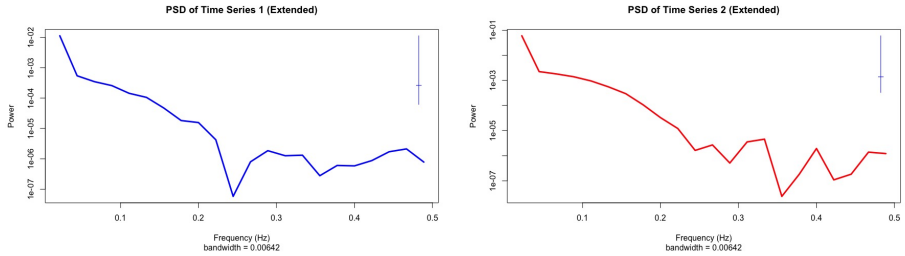


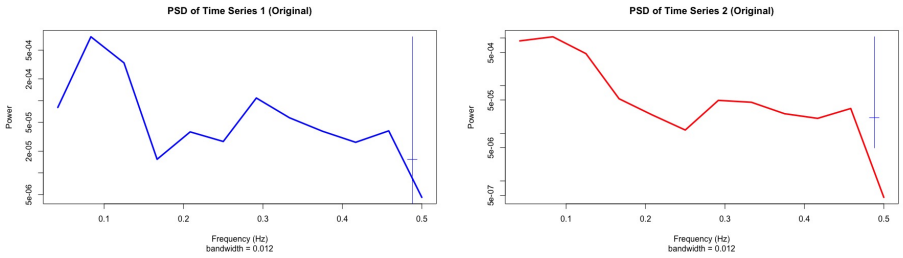
Power Spectral Density

We use the Power Spectra from the previous section and normalize the power of the series by the variance to obtain the Power Spectral Density (PSD) in terms of the auto-correlation using the Wiener-Khinchin Thereom,

$$PSD(f) = \gamma_{xx}(f) = \int_{-\infty}^{\infty} \rho_{xx}(\tau) e^{-i2\pi f\tau} d\tau$$

the results are as follows:

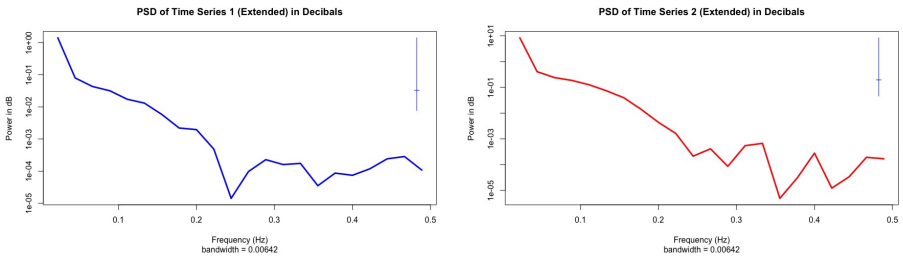




We notice that the Power Spectral Densities are similar to the Power Spectra, however, the y-scale of the graphs of the Power Spectral Densities are slightly lower, such that if the graphs were super-imposed, the Power Spectral Densities would fall within the area of the Power Spectra. This characteristic is attributed to the fact that we normalized the Power Spectra using the variance to yield the Power Spectral Densities.

Signal-to-Noise Ratio

Use the power spectral densities (PSD) of the two extended series, we determine the bandwidth of the random noise (white noise) and transform the y-scale frpm Power to decibels (dB).



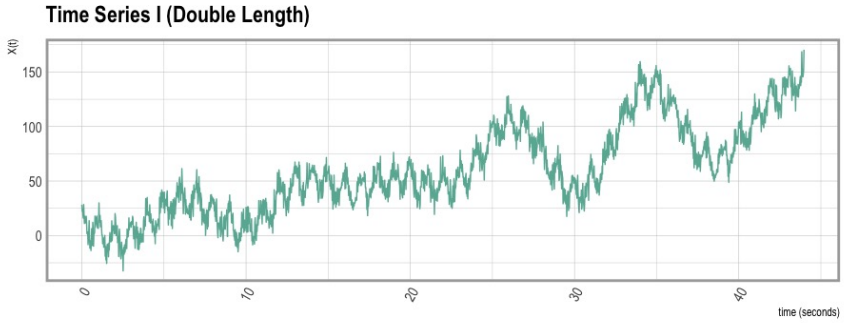
4 FILTER DESIGN

Introducing Filters

In this section we will explore the results of applying various filters to the two time series at extended (double) length. We will explore the effects of low-pass, high-pass, and band-pass filters with respect to each series where their characteristics are as follows:

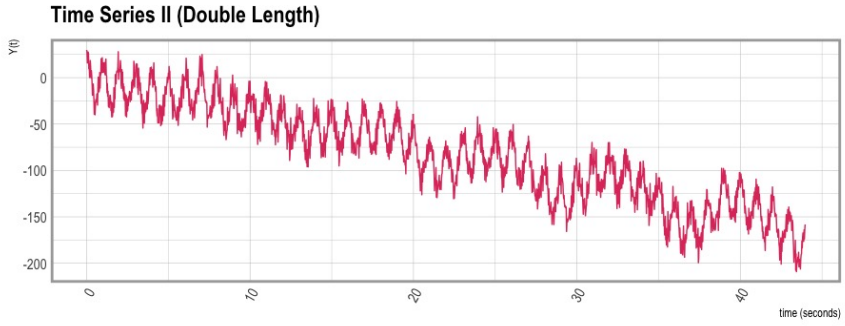
The Table of Wave Characteristics for Series I

f_n	Frequency (f)	Period (T)	Amplitude (A)	Phase Shift (ϕ)
f_1	1/20	20	1	1.7π
f_2	1	1	12	0
f_3	1/7	7	15	0.3π
f_4	1/9	9	18	0.1π
f_5	1/10	10	19	1.2π



The Table of Wave Characteristics for Series II

g_n	Frequency (f)	Period (T)	Amplitude (A)	Phase Shift (ϕ)
g_1	1/14	14	1	1.3π
g_2	1/4	4	5	2.5π
g_3	1/7	7	10	0.6π
g_4	1/8	8	12	1.9π
g_5	1	1	21	0

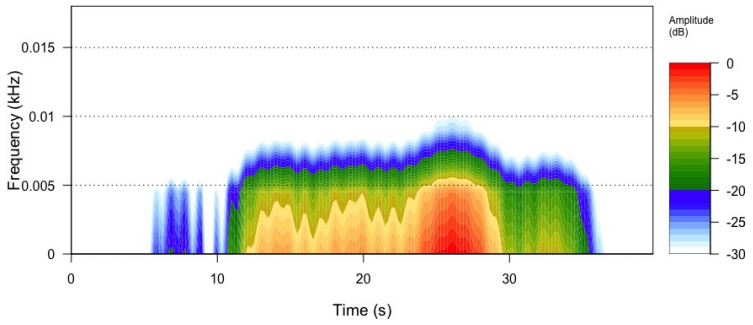


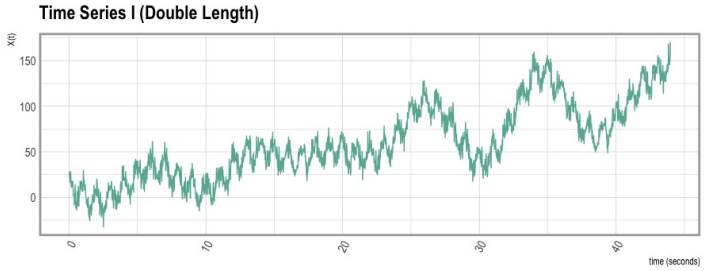
Low Pass Filters

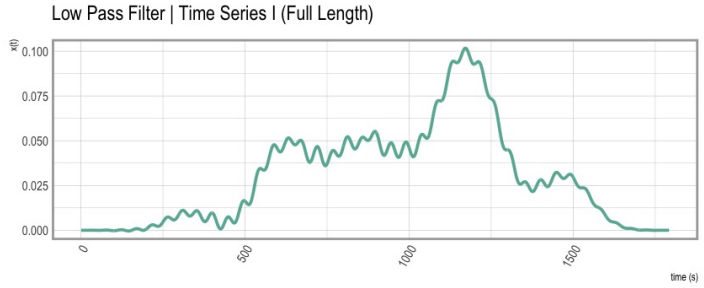
We will create low-pass filters to filter both time series. For both time series,

we will choose a cut-off frequency of 1Hz. We can see from the table of characteristics that for both our time series, the highest frequency for each of the five sinusoidal waves is 1Hz, therefore we want to set a low pass cut-off at 1 so that all five frequencies are captured and the noise (anything above the frequency of 1Hz) is filtered out in the time series.

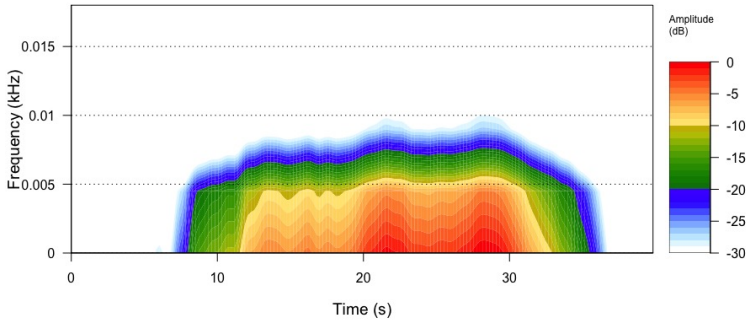
For Time Series I we have:

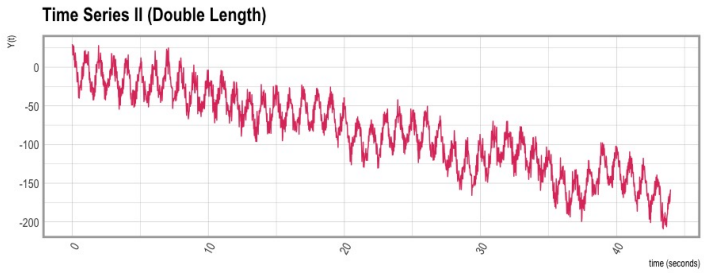


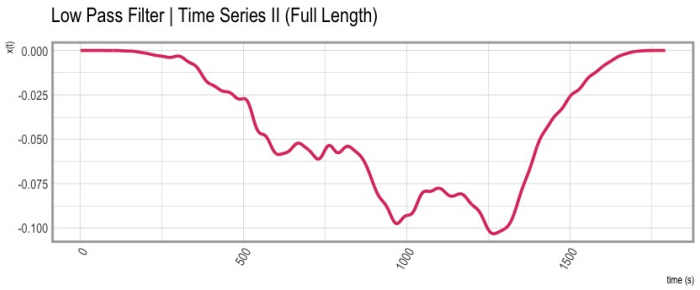




For Time Series II we have:





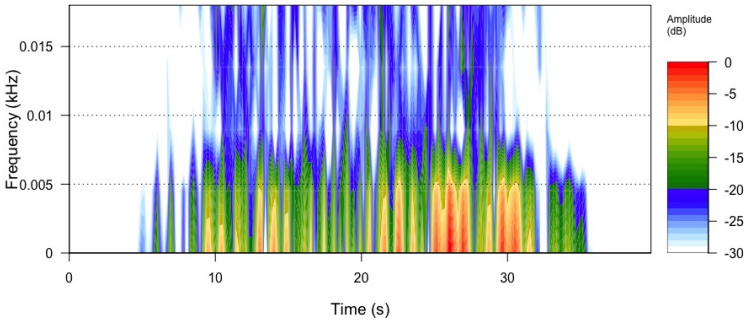


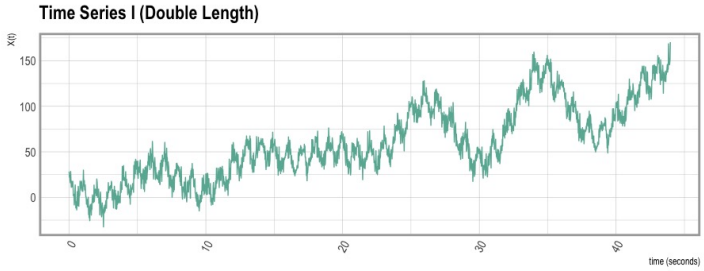
We notice that for both Time Series, the low-pass filter reduces the noise to the signals and creates a smoothing effect. Overall, it preserved the general trend of the series. However, at reduced sampling rates, there are some differences noticeable in the general pattern, this is particularly obvious in time series II where we see the points between 1000-1500 in the domain start to trend upwards.

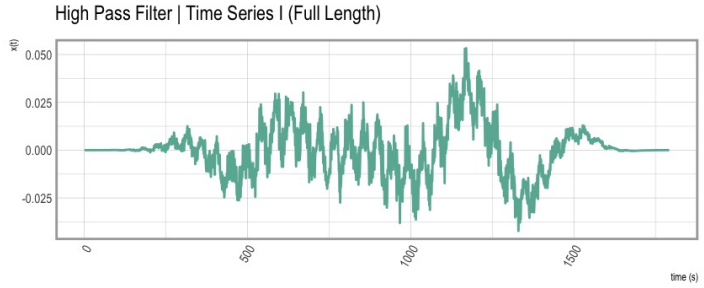
High Pass Filters

Applying high-pass filters to both series to filter out the two sinusoids with the lowest frequency, we set the cut-off frequency for time series I to be above $1/10Hz$.

For Time Series I,

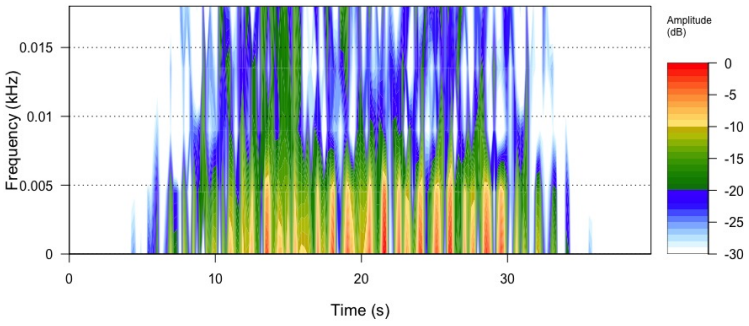


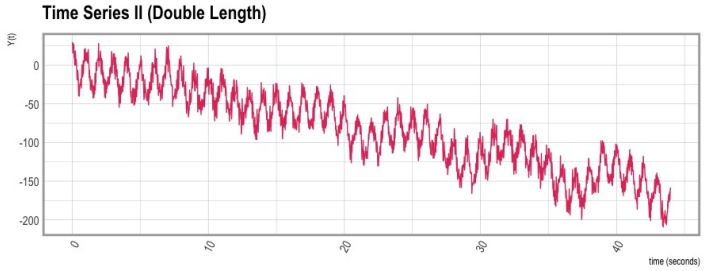


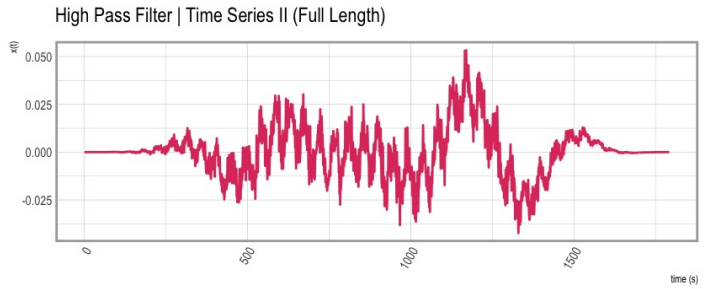


The cut-off frequency for time series II is set to be above $1/8Hz$.

For Time Series II,



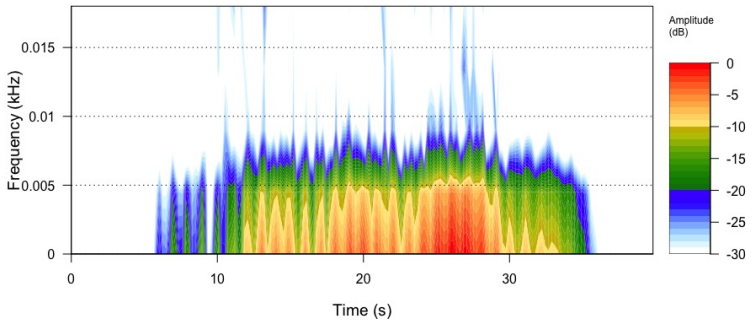


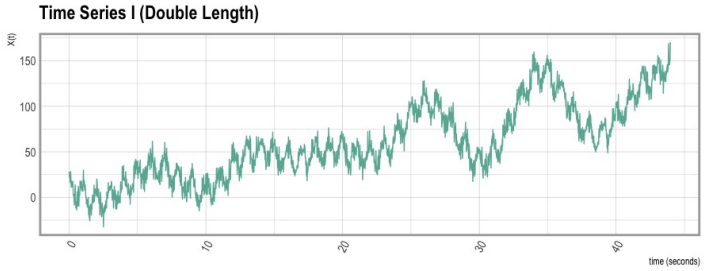


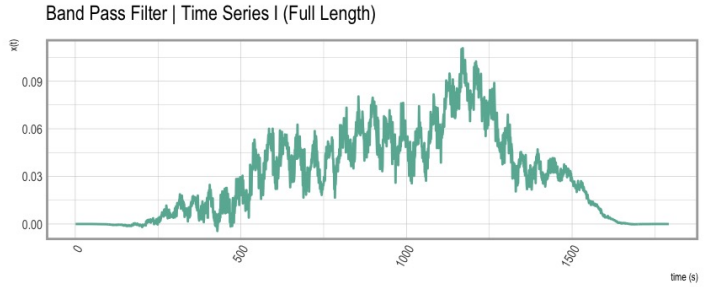
We notice that upon filtering both time series through the high-pass filter, the two lowest frequencies are eliminated. The effect of the linear trend is no longer present. This is especially obvious in Time Series II. We notice that the kHz graph shows that the higher frequencies are reduced by observing the dB reduction and comparing between the kHz ranges 0.010 to 0.015 and 0 to 0.005.

Band Pass Filters

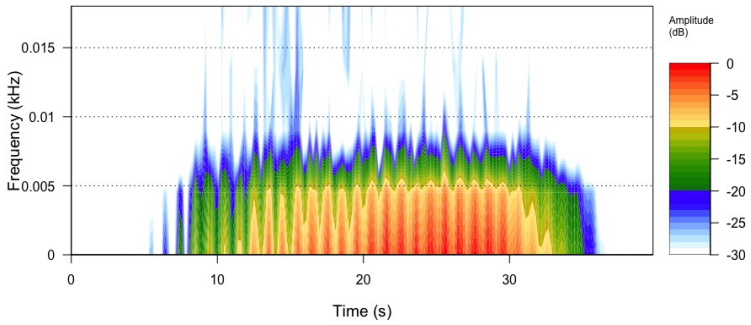
Lastly, we apply band-pass filters to both time series. In order to bypass the intermediate middle frequencies, we set the band-pass to stop frequencies between $1/9Hz$ and $1/6Hz$ for Time Series I.

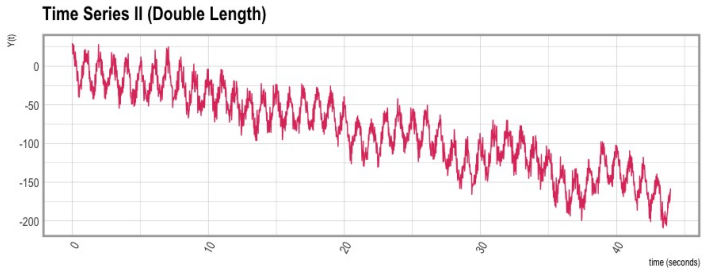


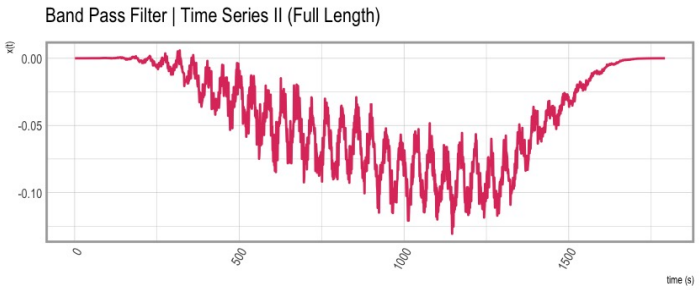




In order to bypass the intermediate frequencies, we set the band-pass to stop frequencies between $1/9Hz$ and $1/5Hz$ for Time Series II.







We notice that the band filter preserved the linear trend to some degree and did not fully filter out the linear trend, this is particularly obvious in Time Series II. In both cases, the high and low frequencies appear to be more dominant at the two ends of the window. We can also observe that the kHz to time plot reveals some low amplitude waves bleeding into the higher frequencies. This means that the band pass filter cut-off was not as clean at frequencies around the band-pass ranges, however, it was able to cut out most so further precision in adjusting the frequency ranges would yield more accurate results.

5 CROSS-SPECTRAL ANALYSIS

Modifications to Original Time Series for Cross-Spectra Analysis

For this section of the analysis we will perform cross-spectra analysis upon making the following adjustments the two original time series':

1. Replace one of the five frequencies in the second series to coincide precisely with one of the frequencies of the first series. It is advisable that one of the two amplitudes be much smaller than the other.
2. Replace a second frequency in the second series to be very close to one of the other frequencies in the first series. One of the amplitudes should be half of the other.

Upon modification to the second series $g(x)$, we have modified the frequency of wave g_2 to coincide precisely with one of the frequencies of the first series f_4. Furthermore, we also modified the frequency of g_4 with respect to f_5 such that they are close and that the amplitude of g_4 is precisely half of f_5.

The updated characteristics table is presented as follows:

The Table of Wave Characteristics for Series I $f(x)$

f_n	Frequency (f)	Period (T)	Amplitude (A)	Phase Shift (ϕ)
f_1	1/20	20	1	1.7π
f_2	1	1	12	0
f_3	1/7	7	15	0.3π
f_4	1/9	9	18	0.1π
f_5	1/10	10	19	1.2π

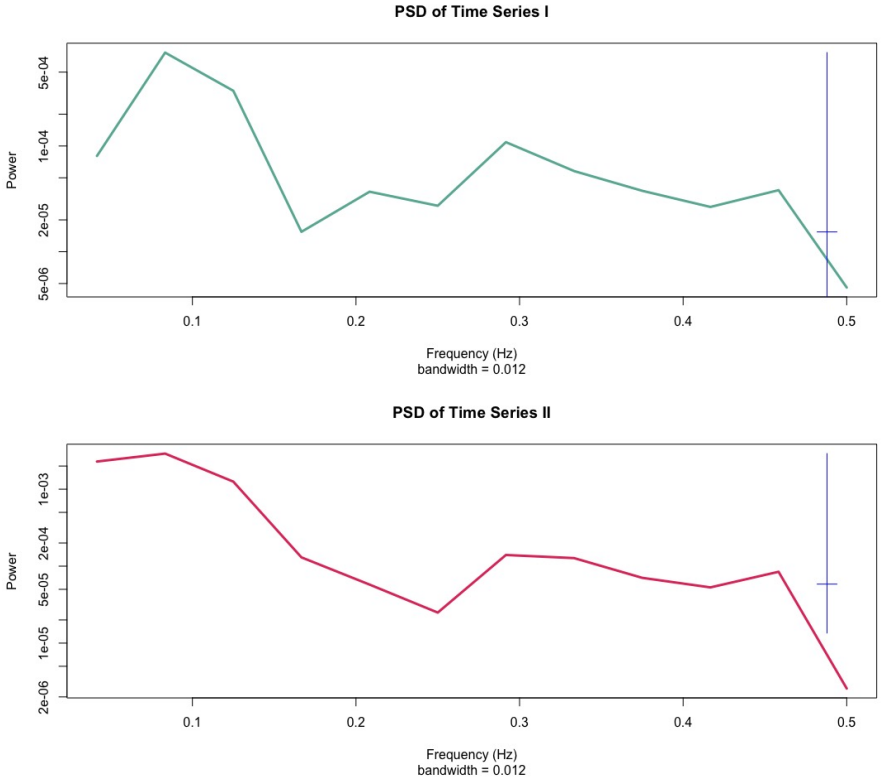
The Table of Wave Characteristics for Series II $g(x)$

g_n	Frequency (f)	Period (T)	Amplitude (A)	Phase Shift (ϕ)
g_1	1/14	14	1	1.3π
g_2	1/9	4	5	2.5π
g_3	1/7	7	10	0.6π
g_4	2/19	8	9.5	1.9π
g_5	1	1	21	0

We will now continue to use Fast Fourier Transformation algorithms to conduct the Cross-Spectra Analysis.

Power Spectrum Density

First we will plot the power spectral density of both time series.

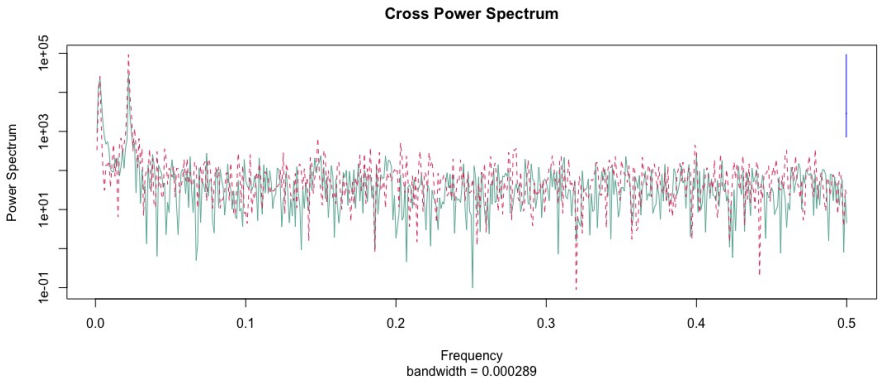


It is noticeable that for both series the frequency of $0.1Hz$ has the most power and that the frequency of $0.5Hz$ has the least influence.

Cross-Power Spectrum

The Cross-Power Spectrum of two series $f(x)$ and $g(x)$ can be defined as the Fourier transform of their cross-covariance function c_{xy}:

$$CrossPS(\omega) = C_{xy}(\omega) = \int_{-\infty}^{\infty} C_{xy}(\tau)e^{-i\omega\tau}d\tau$$

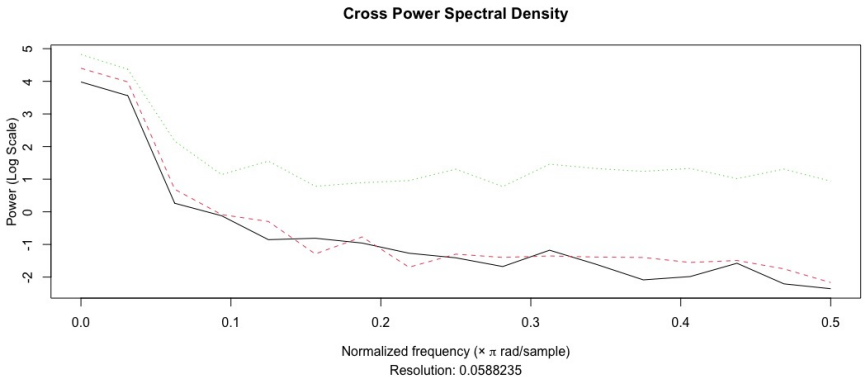


It is noted that the Cross Power Spectrum tells us, in terms of the cross covariance, that the power distribution shared among the power of the two series peak between frequencies 0.0 Hz and 0.1 Hz.

Cross-Power Spectral Density

The Cross-Power Spectral Density of two series $f(x)$ and $g(x)$ is the Fourier transform of their cross-correlation function $\rho_{xy}(\tau)$

$$CrossPSD(\omega) = \Gamma_{xy}(\omega) = \int_{-\infty}^{\infty} \rho_{xy}(\tau) e^{-i\omega\tau} d\tau$$



As the Cross Power Spectral Density would reveal, in terms of the cross-correlation of the two series, we see a maximized power at the low frequencies between 0.0

and 0.1Hz. A significant drop-off is observed between these frequencies as well and both time series decline to a more stable distribution of power around 0.06Hz.

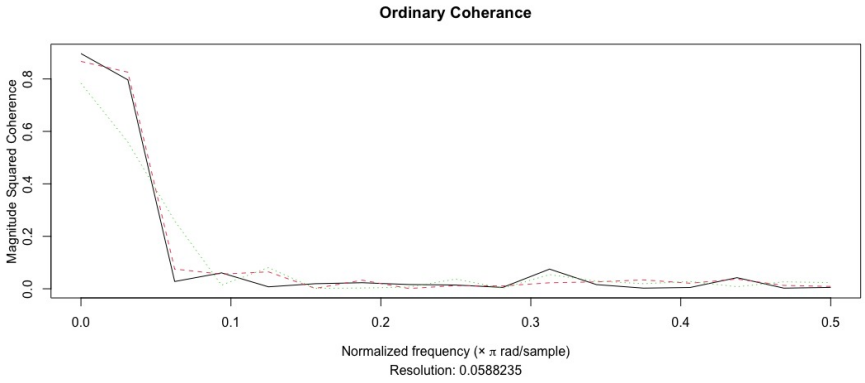
Square Coherence Spectrum

Ordinary Coherence is defined as:

$$|C_{xy}(\omega)|^2 = \frac{|\Gamma_{xy}(\omega)|^2}{\Gamma_{xx}(\omega)\Gamma_{xy}(\omega)}$$

where $\Gamma_{xx}(\omega)$ and $\Gamma_{yy}(\omega)$ are the PSDs of the individual series $f(x)$ and $g(x)$ such that:

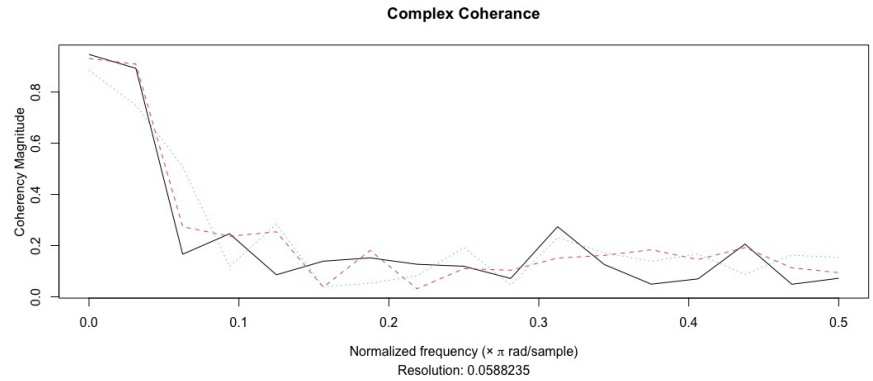
$$0 \leq |C_{xy}(\omega)|^2 \leq 1$$

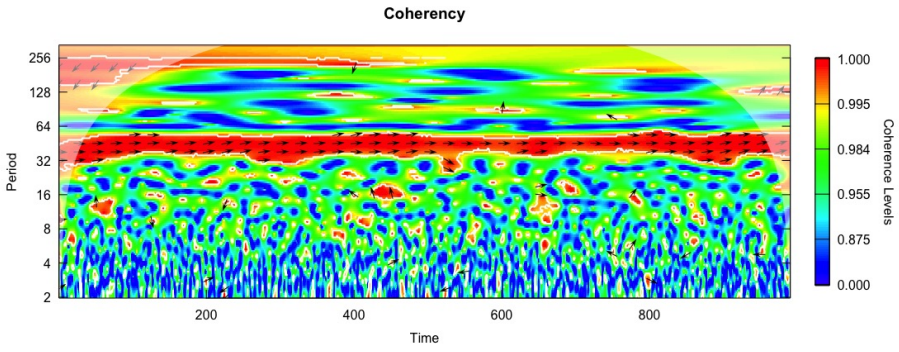


The graph above shows that the ordinary coherence of both series'.

Complex Coherency Spectrum

$$C_{xy}(\omega) = \frac{\Gamma_{xy}(\omega)}{\sqrt{(\Gamma_{xx}(\omega)\Gamma_{xy}(\omega))}}$$





6 REFERENCES

1. Pagiatakis, S. D. (n.d.). *Time Series and Spectral Analysis Lecture.*
2. Shumway, R. H., & Stoffer, D. S. (2010). *Time series analysis and its applications: With R examples (3rd ed.).* Springer.

7 APPENDIX

```
#Section I


# DEPENDENCIES
library(ggplot2)
library(dplyr)
library(hrbrthemes)


######## TIME SERIES 1

### PART 1 // SIMULATION

## GENERATING 5 SIGN WAVES

# A*cos(2*pi*w*t+phi)
pi=3.1415926
t<-seq(0,7*pi,1/45)
length(t)


f1=cos(2*pi*t/20 + 1.7*pi)
plot(t,f1, type="l")

f2=12*cos(2*pi*t)
plot(t,f2, type="l")

f3=15*cos(2*pi*t/7 + 0.3*pi)
plot(t,f3, type="l")

f4=18*cos(2*pi*t/9 + 0.1*pi)
plot(t,f4, type="l")

f5=19*cos(2*pi*t/10 + 1.2*pi)
plot(t,f5, type="l")

# linear / quadratic trend
y=2.7*t


# Random Noise
set.seed(10)
N<- rnorm(length(t),0,7)

#time series 1
fx<-cos(2*pi*t/20 + 1.7*pi)+
```

```
  12*cos(2*pi*t)+
  15*cos(2*pi*t/7 + 0.3*pi)+
  18*cos(2*pi*t/9 + 0.1*pi)+
  19*cos(2*pi*t/10 + 1.2*pi)+N+y

plot(t,fx, type="l")


# creating a dataframe
mat<-matrix(c(t,f1,f2,f3,f4,f5,y,N),ncol = 8, byrow = FALSE)
dat<-as.data.frame(mat)
colnames(dat)<- c("t", "f1","f2","f3","f4","f5","y","Noise")
dat<-dat%>%mutate(fx = f1+f2+f3+f4+f5+N+y)
head(dat)

## PART 2 // PLOTS

#plots of sinusoidal waves
#F1
F1P<-dat%>%ggplot(aes(x=t, y=f1)) +
  geom_line( color="#69b3a2", size=1.3) +
  ggtitle("Sinusoidal Wave 1")+
  xlab("time (seconds)") +
  ylab("f1") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
F1P

#F2
F2P<-dat%>%ggplot(aes(x=t, y=f2)) +
  geom_line( color="#69b3a2", size=1.3) +
  ggtitle("Sinusoidal Wave 2")+
  xlab("time (seconds)") +
  ylab("f2") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
F2P

#F3
F3P<-dat%>%ggplot(aes(x=t, y=f3)) +
  geom_line( color="#69b3a2", size=1.3) +
  ggtitle("Sinusoidal Wave 3")+
  xlab("time (seconds)") +
  ylab("f3") +
```

```
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
F3P

#F4
F4P<-dat%>%ggplot(aes(x=t, y=f4)) +
  geom_line( color="#69b3a2", size=1.3) +
  ggtitle("Sinusoidal Wave 4")+
  xlab("time (seconds)") +
  ylab("f4") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
F4P

#F5
F5P<-dat%>%ggplot(aes(x=t, y=f5)) +
  geom_line( color="#69b3a2", size=1.3) +
  ggtitle("Sinusoidal Wave 5")+
  xlab("time (seconds)") +
  ylab("f5") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
F5P

#y
Y<-dat%>%ggplot(aes(x=t, y=y)) +
  geom_line( color="#69b3a2", size=1.3) +
  ggtitle("Linear Trend y=2.7t")+
  xlab("time (seconds)") +
  ylab("y") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
Y

#Noise
NP<-dat%>%ggplot(aes(x=t, y=N)) +
  geom_line( color="#69b3a2") +
  ggtitle("Noise Level N(0,7)")+
```

```
  xlab("time (seconds)") +
  ylab("Noise") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
NP

#Plot of Time Series I
FXP<-dat%>%ggplot(aes(x=t, y=fx)) +
  geom_line( color="#69b3a2") +
  ggtitle("Time Series I")+
  xlab("time (seconds)") +
  ylab("f(x)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
FXP



####### TIME SERIES 2
pi=3.1415926
t<-seq(0,7*pi,1/45)
# ti<-seq(0,5*pi,1/30)
length(t)


g1=cos(2*pi*t/14 + 1.3*pi)
plot(t,g1, type="l")

g2=5*cos(2*pi*t/4 + 2.5*pi)
plot(t,g2, type="l")

g3=10*cos(2*pi*t/7 + 0.6*pi)
plot(t,g3, type="l")

g4=12*cos(2*pi*t/8 + 1.9*pi)
plot(t,g4, type="l")

g5=21*cos(2*pi*t)
plot(t,g5, type="l")

# linear / quadratic trend
y2=-3.8*t+1
plot(t,y2, type="l")
```

```r
# Random Noise
set.seed(10)
N2<- rnorm(length(t),0,9)

# second series

gx<-cos(2*pi*t/14 + 1.3*pi)+
  5*cos(2*pi*t/4 + 2.5*pi)+
  10*cos(2*pi*t/7 + 0.6*pi)+
  12*cos(2*pi*t/8 + 1.9*pi)+
  21*cos(2*pi*t)+N2+y2

plot(t,gx, type="l")

## creating dataframe

mat2<-matrix(c(t,g1,g2,g3,g4,g5,y2,N2),ncol = 8, byrow = FALSE)
dat2<-as.data.frame(mat2)
colnames(dat2)<- c("t", "g1","g2","g3","g4","g5","y","Noise")
dat2<-dat2%>%mutate(gx = g1+g2+g3+g4+g5+N2+y2)
head(dat2)


#Plotting 2
#G1P
G1P<-dat2%>%ggplot(aes(x=t, y=g1)) +
  geom_line( color="#DF3E6B", size=1.3) +
  ggtitle("Sinusoidal Wave 1")+
  xlab("time (seconds)") +
  ylab("g1") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))

G2P<-dat2%>%ggplot(aes(x=t, y=g2)) +
  geom_line( color="#DF3E6B", size=1.3) +
  ggtitle("Sinusoidal Wave 2")+
  xlab("time (seconds)") +
  ylab("g2") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))


G3P<-dat2%>%ggplot(aes(x=t, y=g3)) +
  geom_line( color="#DF3E6B", size=1.3) +
```

```
  ggtitle("Sinusoidal Wave 3")+
  xlab("time (seconds)") +
  ylab("g3") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))


G4P<-dat2%>%ggplot(aes(x=t, y=g4)) +
  geom_line( color="#DF3E6B", size=1.3) +
  ggtitle("Sinusoidal Wave 4")+
  xlab("time (seconds)") +
  ylab("g4") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))


G5P<-dat2%>%ggplot(aes(x=t, y=g5)) +
  geom_line( color="#DF3E6B", size=1.3) +
  ggtitle("Sinusoidal Wave 5")+
  xlab("time (seconds)") +
  ylab("g5") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))

G1P
G2P
G3P
G4P
G5P

Y2<-dat2%>%ggplot(aes(x=t, y=y)) +
  geom_line( color="#DF3E6B", size=1.3) +
  ggtitle("Linear Trend y=-3.8t+1")+
  xlab("time (seconds)") +
  ylab("y") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
Y2
```

```
#Noise
N2<-dat2%>%ggplot(aes(x=t, y=Noise)) +
  geom_line( color="#DF3E6B") +
  ggtitle("Noise Level N(0,9)")+
  xlab("time (seconds)") +
  ylab("Noise") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
N2

#Plot of Time Series II
GXP<-dat2%>%ggplot(aes(x=t, y=gx)) +
  geom_line( color="#DF3E6B") +
  ggtitle("Time Series II")+
  xlab("time (seconds)") +
  ylab("Y(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
GXP
FXP



## ANALYSIS

# For Time Series 1
head(dat)

#Mean
n<-length(dat$fx)
Mx<-(1/n)*sum(dat$fx)
Mx

#Var

varx<-(1/n)*sum((dat$fx-Mx)^2)
varx

#quadratic norm & power of series
n
P<-(1/n)*sum((dat$fx)^2)
P
```

```
#auto-covariance
Cxx<-(1/(n-1))*sum((dat$fx[1:n-1] - Mx)*(dat$fx[2:n]-Mx))

Cxx0<- (1/n)*sum((dat$fx - Mx)^2)
Cxx0

#auto-correlation

Cxx/Cxx0


# For time series 2 y(t_i)

head(dat2)

#Mean
n2<-length(dat2$gx)
n2
sum(dat2$gx)
My<-(1/n2)*sum(dat2$gx)
My

#Variance


vary<- (1/n2)*sum((dat2$gx-My)^2)
vary

#power
n2
P_y<-(1/n2)*sum((dat2$gx)^2)
P_y

#auto-covariance
Cyy<-(1/(n2-1))*sum((dat2$gx[1:n2-1] - My)*(dat2$gx[2:n2]-My))
Cyy

#auto-correlation

Cyy0<- (1/n2)*sum((dat2$gx - My)^2)
Cyy0

Cyy/Cyy0


#Cross Covariance

Cxy<- (1/(n))*sum((dat$fx[1:n-1] - Mx)*(dat2$gx[2:n]-My))
Cxy
```

```r
#Cross Correlation

Cxy/sqrt(Cxx0 * Cyy0)


#doubling the length of both series.


d<-seq(7*pi,14*pi,1/45)

f12=cos(2*pi*d/20 + 1.7*pi)

f22=12*cos(2*pi*d)

f32=15*cos(2*pi*d/7 + 0.3*pi)

f42=18*cos(2*pi*d/9 + 0.1*pi)

f52=19*cos(2*pi*d/10 + 1.2*pi)

# linear / quadratic trend
y2=2.7*d


# Random Noise
set.seed(10)
Nd<- rnorm(length(t),0,7)



#time series 1
fx2<-cos(2*pi*d/20 + 1.7*pi)+
  12*cos(2*pi*d)+
  15*cos(2*pi*d/7 + 0.3*pi)+
  18*cos(2*pi*d/9 + 0.1*pi)+
  19*cos(2*pi*d/10 + 1.2*pi)+Nd+y2

plot(d,fx2, type="l")

matxd<-matrix(c(d,f12,f22,f32,f42,f52,y2,Nd),ncol = 8, byrow = FALSE)
datxd<-as.data.frame(matxd)
colnames(datxd)<- c("t", "f1","f2","f3","f4","f5","y","Noise")
datxd<-datxd%>%mutate(fx = f12+f22+f32+f42+f52+Nd+y2)
head(datxd)

# binding rows of 2 dataframes

doublex<-bind_rows(dat,datxd)
plot(doublex$t, doublex$fx, type="l")
```

```
### doubling length time series 2
d<-seq(7*pi,14*pi,1/45)

g12=cos(2*pi*d/14 + 1.3*pi)

g22=5*cos(2*pi*d/4 + 2.5*pi)

g32=10*cos(2*pi*d/7 + 0.6*pi)

g42=12*cos(2*pi*d/8 + 1.9*pi)

g52=21*cos(2*pi*d)

# linear / quadratic trend
lt=-3.8*d+1
plot(d,y2, type="l")

# Random Noise
set.seed(10)
Nd2<- rnorm(length(t),0,9)

gx<-cos(2*pi*d/14 + 1.3*pi)+
  5*cos(2*pi*d/4 + 2.5*pi)+
  10*cos(2*pi*d/7 + 0.6*pi)+
  12*cos(2*pi*d/8 + 1.9*pi)+
  21*cos(2*pi*d)+Nd2+lt

plot(d,gx, type="l")

## Dataframe of double length time series y

mat2d<-matrix(c(d,g12,g22,g32,g42,g52,lt,Nd2),ncol = 8, byrow = FALSE)
dat2d<-as.data.frame(mat2d)
colnames(dat2d)<- c("t", "g1","g2","g3","g4","g5","y","Noise")
dat2d<-dat2d%>%mutate(gx = g12+g22+g32+g42+g52+Nd2+lt)
head(dat2d)

doubley<-bind_rows(dat2,dat2d)
plot(doubley$t, doubley$gx, type="l")

###### Plots of Doubled Time Series:
X2D<-doublex%>%ggplot(aes(x=t, y=fx)) +
  geom_line( color="#69b3a2") +
  ggtitle("Time Series I (Double Length)")+
  xlab("time (seconds)") +
```

```
  ylab("X(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
X2D


Y2D<-doubley%>%ggplot(aes(x=t, y=gx)) +
  geom_line( color="#DF3E6B") +
  ggtitle("Time Series II (Double Length)")+
  xlab("time (seconds)") +
  ylab("Y(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
Y2D


# Analysis

# Mean

n<-length(doublex$t)
n
sum(doublex$fx)
Mx<-(1/n)*sum(doublex$fx)
Mx

My<-(1/n)*sum(doubley$gx)
My

# Var
sx<-(1/n)*sum((doublex$fx-Mx)^2)
sx

sy<-(1/n)*sum((doubley$gx-Mx)^2)
sy

#Power


PX2<-(1/n)*sum((doublex$fx)^2)
PY2<-(1/n)*sum((doubley$gx)^2)

PX2
PY2
```

```
# Auto Cov
#X(ti)
Gammaxx<-(1/(n-1))*sum((doublex$fx[1:n-1] - Mx)*(doublex$fx[2:n]-Mx))
Gammaxx

#Y(ti)
Gammayy<-(1/(n-1))*sum((doubley$gx[1:n-1] - My)*(doubley$gx[2:n]-My))
Gammayy

# Auto Cor

#X(ti)
Gammaxx/sx

#Y(ti)
Gammayy/sy


#Cross-COV Cross-Cor

Gammaxy<- (1/(n))*sum((doublex$fx[1:n-1] - Mx)*(doubley$gx[2:n]-My))
Gammaxy

#Cross Correlation

Gammaxy/(sqrt(sx * sy))


# Section II

#### DEPENDENCIES
library(ggplot2)
library(dplyr)
library(hrbrthemes)
library(TSA)

# read csv file
getwd()
main<-read.csv("main.csv")

#dropping rowname index
main<-(main[,2:18])
dim(main)
colnames(main)
head(main)


#indexing "main" dataframe for f(x) and g(x)
```

```
## full extended duration 1980 rows for f(x)
ffx<-main[ ,1:9]

## full extended duration 1980 rows for g(x)
fgx<-main[ ,c(1,10:17)]

## half duration first 990 rows for f(x)
hfx<-main[1:990,1:9]

## half duration first 990 rows for g(x)
hgx<-main[991:1980,c(1,10:17)]

# Use FFT or other Fourier algorithms to analyse the series simulated

#1. FOR ffx
fft(ffx$fx)/1980
tr_ffx<-(fft(ffx$fx)/1980)

#1b Periodogram
P1<-periodogram(tr_ffx,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 1 at Full Length")

#P1

#2. FOR fgx
fft(fgx$gx)/1980
tr_fgx<-(fft(fgx$gx)/1980)

#2b Periodogram
P2<-periodogram(tr_fgx,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 2 at Full Length")

#P2

#3. FOR hfx
fft(hfx$fx)/990
tr_hfx<-(fft(hfx$fx)/990)

#3b Periodogram
P3<-periodogram(tr_hfx,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 1 at Half Length")

#4. FOR hgx
fft(hgx$gx)/990
```

```
tr_hgx<-(fft(hgx$gx)/990)

#4b Periodogram
P4<-periodogram(tr_hgx,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 2 at Half Length")


#### PLOTS

P1
P2
P3
P4
#### 2 Removing the LT

ffx$fx4<-ffx$fx-ffx$LTx
fgx$gx4<-fgx$gx-fgx$LTy
hfx$fx4<-hfx$fx-hfx$LTx
hgx$gx4<-hgx$gx-hgx$LTy

# fft after removing the LT
fft(ffx$fx4)/1980
tr_ffx4<-(fft(ffx$fx4)/1980)

fft(fgx$gx4)/1980
tr_fgx4<-(fft(fgx$gx4)/1980)


fft(hfx$fx4)/990
tr_hfx4<-(fft(hfx$fx4)/990)

fft(hgx$gx4)/990
tr_hgx4<-(fft(hgx$gx4)/990)



#plots

L1<-periodogram(tr_ffx4,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 1 at Full Length w/o Linear
                    Trend")

L2<-periodogram(tr_fgx4,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
```

```
                main="Periodogram of Series 2 at Full Length w/o Linear
                    Trend")


L3<-periodogram(tr_hfx4,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 1 at Half Length w/o Linear
                    Trend")


L4<-periodogram(tr_hgx4,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 2 at Half Length w/o Linear
                    Trend")







#### 3 Removing the random noise

ffx$fx2<-ffx$fx-ffx$Nx
fgx$gx2<-fgx$gx-fgx$Ny
hfx$fx2<-hfx$fx-hfx$Nx
hgx$gx2<-hgx$gx-hgx$Ny


# fft after reducing noise
fft(ffx$fx2)/1980
tr_ffx2<-(fft(ffx$fx2)/1980)

fft(fgx$gx2)/1980
tr_fgx2<-(fft(fgx$gx2)/1980)


fft(hfx$fx2)/990
tr_hfx2<-(fft(hfx$fx2)/990)

fft(hgx$gx2)/990
tr_hgx2<-(fft(hgx$gx2)/990)


#Periodograms

P5<-periodogram(tr_ffx2,
               ylab=expression(paste("Power (", lambda, ") ")),
```

```
              xlab="Frequency (Hz)",
              main="Periodogram of Series 1 at Full Length w/o Noise")
P5

P6<-periodogram(tr_fgx2,
              ylab=expression(paste("Power (", lambda, ") ")),
              xlab="Frequency (Hz)",
              main="Periodogram of Series 2 at Full Length w/o Noise")
P6

P7<-periodogram(tr_hfx2,
              ylab=expression(paste("Power (", lambda, ") ")),
              xlab="Frequency (Hz)",
              main="Periodogram of Series 1 at Half Length w/o Noise")
P7

P8<-periodogram(tr_hgx2,
              ylab=expression(paste("Power (", lambda, ") ")),
              xlab="Frequency (Hz)",
              main="Periodogram of Series 2 at Half Length w/o Noise")
P8


# 4 doubling the noise level


ffx$fx3<-ffx$fx+ffx$Nx
fgx$gx3<-fgx$gx+fgx$Ny
hfx$fx3<-hfx$fx+hfx$Nx
hgx$gx3<-hgx$gx+hgx$Ny

# fft after doubling noise
fft(ffx$fx3)/1980
tr_ffx3<-(fft(ffx$fx3)/1980)

fft(fgx$gx3)/1980
tr_fgx3<-(fft(fgx$gx3)/1980)


fft(hfx$fx3)/990
tr_hfx3<-(fft(hfx$fx3)/990)

fft(hgx$gx3)/990
tr_hgx3<-(fft(hgx$gx3)/990)

#plots

P9<-periodogram(tr_ffx3,
              ylab=expression(paste("Power (", lambda, ") ")),
              xlab="Frequency (Hz)",
```

```
                main="Periodogram of Series 1 at Full Length w/ Double
                    Noise")

P10<-periodogram(tr_fgx3,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 2 at Full Length w/ Double
                    Noise")


P11<-periodogram(tr_hfx3,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 1 at Half Length w/ Double
                    Noise")


P12<-periodogram(tr_hgx3,
                ylab=expression(paste("Power (", lambda, ") ")),
                xlab="Frequency (Hz)",
                main="Periodogram of Series 2 at Half Length w/ Double
                    Noise")


#Section III

#### DEPENDENCIES
library(ggplot2)
library(dplyr)
library(hrbrthemes)
library(TSA)
library(bspec)

# read csv file
getwd()
main<-read.csv("main.csv")

#dropping rowname index
main<-(main[,2:18])
dim(main)
colnames(main)
head(main)

#indexing "main" dataframe for f(x) and g(x)

## full extended duration 1980 rows for f(x)
ffx<-main[ ,1:9]

## full extended duration 1980 rows for g(x)
fgx<-main[ ,c(1,10:17)]
```

```
## half duration first 990 rows for f(x)
hfx<-main[1:990,1:9]

## half duration first 990 rows for g(x)
hgx<-main[991:1980,c(1,10:17)]


#### FFT to obtain Fourier Series

N<-1980
delta<- 1/45

n<-N*delta


Cffx<-acf(ffx$fx, type = "covariance", plot=FALSE, lag.max =n )
class(Cffx)
attributes(Cffx)
cffx<-Cffx$acf
class(cffx)

tcffx<-fft(cffx)

P1<-spectrum(cffx,
             xlab="Frequency (Hz)",
             ylab=expression(paste("Power (", lambda^2, ") ")),
             main="Power Spectra of Time Series 1 (Extended)",
             col="blue",
             lwd=3,
             )



Cfgx<-acf(fgx$gx, type = "covariance", plot=FALSE, lag.max=n )
cfgx<-Cfgx$acf
fft(cfgx)

tcfgx<-fft(cfgx)

P2<-spectrum(cfgx,
             xlab="Frequency (Hz)",
             ylab=expression(paste("Power (", lambda^2, ") ")),
             main="Power Spectra of Time Series 2 (Extended)",
             col="red",
             lwd=3,
)


N<-1980/2
```

```r
delta<- 1/45

n<-N*delta


Chfx<-acf(hfx$fx, type = "covariance", plot=FALSE, lag.max =n )
chfx<-Chfx$acf

tchfx<-fft(chfx)

P3<-spectrum(chfx,
             xlab="Frequency (Hz)",
             ylab=expression(paste("Power (", lambda^2, ") ")),
             main="Power Spectra of Time Series 1 (Original)",
             col="blue",
             lwd=3,
)



N<-1980/2
delta<- 1/45

n<-N*delta


Chgx<-acf(hgx$gx, type = "covariance", plot=FALSE, lag.max =n )
chgx<-Chgx$acf

tchgx<-fft(chgx)

P4<-spectrum(chgx,
             xlab="Frequency (Hz)",
             ylab=expression(paste("Power (", lambda^2, ") ")),
             main="Power Spectra of Time Series 2 (Original)",
             col="red",
             lwd=3,
)



## PSD

N<-1980
delta<- 1/45

n<-N*delta
```

```
Pffx<-acf(ffx$fx, type = "correlation", plot=FALSE, lag.max =n )
class(Pffx)
attributes(Pffx)
pffx<-Pffx$acf
class(pffx)

tpffx<-fft(pffx)

PSD1<-spectrum(pffx,
            xlab="Frequency (Hz)",
            ylab="Power",
            main="PSD of Time Series 1 (Extended)",
            col="blue",
            lwd=3,
)



Pfgx<-acf(fgx$gx, type = "correlation", plot=FALSE, lag.max=n )
pfgx<-Pfgx$acf
fft(pfgx)

tpfgx<-fft(pfgx)

PSD2<-spectrum(pfgx,
            xlab="Frequency (Hz)",
            ylab="Power",
            main="PSD of Time Series 2 (Extended)",
            col="red",
            lwd=3,
)



N<-1980/2
delta<- 1/45

n<-N*delta


Phfx<-acf(hfx$fx, type = "correlation", plot=FALSE, lag.max =n )
phfx<-Phfx$acf

tphfx<-fft(phfx)

PSD3<-spectrum(phfx,
            xlab="Frequency (Hz)",
            ylab="Power",
            main="PSD of Time Series 1 (Original)",
```

```
            col="blue",
            lwd=3,
)



N<-1980/2
delta<- 1/45

n<-N*delta


Phgx<-acf(hgx$gx, type = "correlation", plot=FALSE, lag.max =n )
phgx<-Phgx$acf

tphgx<-fft(phgx)

PSD4<-spectrum(phgx,
            xlab="Frequency (Hz)",
            ylab="Power",
            main="PSD of Time Series 2 (Original)",
            col="red",
            log="yes",
            lwd=3,
)



######### 3
N<-1980
delta<- 1/45

tau<-N*delta

# PSD of extended series 1

PSD1$bandwidth

SNR1<-10*log(pffx)
SNR1

PSDfx<- spectrum(SNR1,
                xlab="Frequency (Hz)",
                ylab="Power in dB",
                main="PSD of Time Series 1 (Extended) in Decibals",
                col="blue",
                log="yes",
                lwd=3,
)
```

```
snr(ffx$fx, PSD1)

             # PSD1$
             #
             # fx<-ffx$fx
             # fxacf <- acf(fx,type="covariance", lag=tau)
             # ftfx <- fft(fxacf$acf)
             # freqfx <- (1:nrow(ftfx))*N/nrow(ftfx) # 1000 hz
             # Afx <- (Re(ftfx)^2 + Im(ftfx)^2)^.5 #amplitude is
                 magnitude
             # Pfx<-
             # PSDfx <- cbind.data.frame(freqfx,Afx)
             # ggplot(PSDfx, aes(x=freqfx,y=Afx)) + geom_smooth()

# PSD of extended series 2

PSD2$bandwidth

SNR2<-10*log(pfgx)
SNR2

PSDgx<- spectrum(SNR2,
                 xlab="Frequency (Hz)",
                 ylab="Power in dB",
                 main="PSD of Time Series 2 (Extended) in Decibals",
                 col="red",
                 log="yes",
                 lwd=3,
)

# determine the bandwidth of the random noise (white noise) as well as
    its power (variance)

attributes(PSD1)

## Transformation to SNR
# use the noise power level to transform the PSD into another form in
    which the y-axis will be signal-to-noise ratio (SNR) expressed in
    decibels (dB)


sfx<-PSD1$freq
sfy<-PSD1$spec
```

```
###### 4
#we will consider time series 1
head(ffx)

# adding 2 more waves with frequency that are higher than the Nyquist

# nyquist freq.
nyq<-(2*delta)^(-1)
```

```
#Section IV

#### DEPENDENCIES
library(ggplot2)
library(hrbrthemes)
library(dplyr)
library(seewave)
library(smoother)
library(clampSeg)

# read csv file
getwd()
main<-read.csv("main.csv")

#dropping rowname index
main<-(main[,2:18])
dim(main)
colnames(main)
head(main)

#indexing "main" dataframe for f(x) and g(x)

## full extended duration 1980 rows for f(x)
ffx<-main[ ,1:9]

## full extended duration 1980 rows for g(x)
fgx<-main[ ,c(1,10:17)]

## half duration first 990 rows for f(x)
hfx<-main[1:990,1:9]

## half duration first 990 rows for g(x)
hgx<-main[991:1980,c(1,10:17)]
```

```
# PART 1 LOW PASS FILTER

  # TIME SERIES 1 FULL LENGTH



LPF1<-ffilter(ffx$fx,f=45,to=1.05)
S1<-spectro(LPF1,f=45,wl=10)


P1D<-as.data.frame(matrix(c(1:length(LPF1),LPF1), ncol = 2, byrow = TRUE
    ))


P1<-P1D%>%ggplot(aes(x=1:length(LPF1), y=LPF1)) +
  geom_line( color="#69b3a2", size=1.3) +
  ggtitle(expression("Low Pass Filter | Time Series I (Full Length)"))+
  xlab("time (s)") +
  ylab("x(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
P1

  # TIME SERIES 2 FULL LENGTH

LPF2<-ffilter(fgx$gx,f=45,to=1)
S2<-spectro(LPF2,f=45,wl=10)


P2D<-as.data.frame(matrix(c(1:length(LPF2),LPF2), ncol = 2, byrow = TRUE
    ))


P2<-P2D%>%ggplot(aes(x=1:length(LPF2), y=LPF2)) +
  geom_line( color="#DF3E6B", size=1.3) +
  ggtitle(expression("Low Pass Filter | Time Series II (Full Length)"))+
  xlab("time (s)") +
  ylab("x(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
P2
```

```
# PART 2 HIGH PASS FILTER

  # TIME SERIES 1 FULL LENGTH

HPF1<-ffilter(ffx$fx,f=45,from=0.11)
SS1<-spectro(HPF1,f=45,wl=10)

P3D<-as.data.frame(matrix(c(1:length(HPF1),HPF1), ncol = 2, byrow = TRUE
    ))
P3<-P3D%>%ggplot(aes(x=1:length(HPF1), y=HPF1)) +
  geom_line( color="#69b3a2", size=1) +
  ggtitle(expression("High Pass Filter | Time Series I (Full Length)"))+
  xlab("time (s)") +
  ylab("x(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
P3
  # TIME SERIES 2 FULL LENGTH

HPF2<-ffilter(fgx$gx,f=45,from=0.126)
SS2<-spectro(HPF2,f=45,wl=10)

P4D<-as.data.frame(matrix(c(1:length(HPF2),HPF2), ncol = 2, byrow = TRUE
    ))
P4<-P4D%>%ggplot(aes(x=1:length(HPF2), y=HPF1)) +
  geom_line( color="#DF3E6B", size=1) +
  ggtitle(expression("High Pass Filter | Time Series II (Full Length)"))+
  xlab("time (s)") +
  ylab("x(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
P4

# PART 3 BAND PASS FILTER

# filter out the two intermediate frequencies
# TIME SERIES 1 FULL LENGTH
BPF1<-ffilter(ffx$fx,f=45,from=1/9.5, to=1/6.5, bandpass = FALSE)
SSS1<-spectro(BPF1,f=45,wl=10)

P5D<-as.data.frame(matrix(c(1:length(BPF1),BPF1), ncol = 2, byrow = TRUE
    ))
P5<-P5D%>%ggplot(aes(x=1:length(BPF1), y=BPF1)) +
```

```
  geom_line( color="#69b3a2", size=1) +
  ggtitle(expression("Band Pass Filter | Time Series I (Full Length)"))+
  xlab("time (s)") +
  ylab("x(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
P5

# TIME SERIES 2 FULL LENGTH

BPF2<-ffilter(fgx$gx,f=45,from=1/9, to=1/5, bandpass = FALSE)
SSS2<-spectro(BPF2,f=45,wl=10)

P6D<-as.data.frame(matrix(c(1:length(BPF2),BPF2), ncol = 2, byrow = TRUE
    ))
P6<-P6D%>%ggplot(aes(x=1:length(BPF2), y=BPF2)) +
  geom_line( color="#DF3E6B", size=1) +
  ggtitle(expression("Band Pass Filter | Time Series II (Full Length)"))+
  xlab("time (s)") +
  ylab("x(t)") +
  theme_ipsum() +
  theme(axis.text.x=element_text(angle=60, hjust=1))+
  theme(panel.border = element_rect(color = "darkgrey",
                                    fill = NA,
                                    size = 2))
P6


#Section V

library(dplyr)
library(ggplot2)
library(hrbrthemes)
library(gsignal)
library(ClamR)
library(surveillance)
library(tester)
library(IRISSeismic)
library(WaveletComp)

# Importing Data from assignment 1

  # read csv file
getwd()
main<-read.csv("main.csv")

  #dropping rowname index
```

```
main<-(main[,2:18])
dim(main)
colnames(main)
head(main)

# indexing "main" dataframe for f(x) and g(x)

  ## half duration first 990 rows for f(x)
hfx<-main[1:990,1:9]

  ## half duration first 990 rows for g(x)
hgx<-main[991:1980,c(1,10:17)]


# Time Series 1 f(x)
head(hfx)

# Time Series 2 g(x)
head(hgx)


# Modification of original time series g(x)

pi=3.1415926
t<-seq(0,7*pi,1/45)
length(t)

# A*cos(2*pi*w*t+phi)

# Mod 1

# we will make changes to waves g_2 coincide precisely with one of the
# frequencies of the first series f_4.
# It is noted that one of the two amplitudes be much smaller than the
    other.

  # original g2=5*cos(2*pi*t/4 + 2.5*pi)

mg2= 5*cos(2*pi*t/9 + 2.5*pi)

# Mod 2

# Change g_4 with respect to f_5
# to have a amplitude that is 1/2 of f_5 and a frequency that is close


  # original g4=12*cos(2*pi*t/8 + 1.9*pi)

mg4= 9.5*cos(2*pi*t/9.5 + 1.9*pi)
```

```
# Wrangling
# modify g(x) data frame to add new values for g_2 and g_4
# and new sum of g(x) values

head(hgx)


mods<-cbind(mg2,mg4)

colnames(hgx)

hgx<-hgx%>%select("t","g1","g3","g5","LTy","Ny")

hgx<-cbind(hgx,mods)

hgx<-hgx%>%select("t","g1","mg2","g3","mg4","g5","LTy","Ny")

hgx<-cbind(hgx,rowSums(hgx))

names(hgx)<-c("t","g1","g2","g3","g4","g5","LTy","Ny", "gx")

head(hgx)


# I - PSD

#### FFT to obtain Fourier Series

N<-990
delta<- 1/45
n<-N*delta

# PSD of f(x)

Phfx<-acf(hfx$fx, type = "correlation", plot=FALSE, lag.max =n )
class(Phfx)
attributes(Phfx)
phfx<-Phfx$acf
class(phfx)

tphfx<-fft(phfx)

PSD1<-spectrum(phfx,
               xlab="Frequency (Hz)",
               ylab="Power",
               main="PSD of Time Series I",
               col="#69b3a2",
               lwd=3,
)
```

```
# PSD of g(x)
Phgx<-acf(hgx$gx, type = "correlation", plot=FALSE, lag.max =n )
class(Phgx)
attributes(Phgx)
phgx<-Phgx$acf
class(phgx)

tphgx<-fft(phgx)

PSD2<-spectrum(phgx,
               xlab="Frequency (Hz)",
               ylab="Power",
               main="PSD of Time Series II",
               col="#DF3E6B",
               lwd=3,
)


PSD1
PSD2


#Further Wrangling
#create a matrix such that each column is a signal
hdx<-cbind(hfx,hgx)
inp<-hdx%>%select("t","fx","gx")
names(inp)<-c("window", "fx", "gx")
head(inp)
inp<-as.matrix(inp)


ts<-inp[1:nrow(inp),2:3]
ts

# II - Cross Power Spectrum

csd(
  inp,
  window = NextPow2(sqrt(NROW(inp))),
  overlap = 0.5,
  nfft = ifelse(is_scalar(window), window, length(window)),
  fs = 1,
  detrend = "none"
)

CPS<-crossSpectrum(ts, spans = NULL, kernel = NULL, taper = 0.1,
              pad = 0, fast = TRUE,
              demean = FALSE, detrend = TRUE,
```

```
            na.action = stats::na.fail)

attributes(CPS)
head(CPS)


# Calculate the transfer function
transferFunction <- CPS$Pxy / CPS$Pxx
transferAmp <- Mod(transferFunction)
transferPhase <- pracma::mod(Arg(transferFunction) * 180/pi,360)

# 2 rows
layout(matrix(seq(2)))

# Plot
plot(1/CPS$freq,transferAmp,type='l',log='x',
     xlab="Period (sec)",
     main="Transfer Function Amplitude")

plot(1/CPS$freq,transferPhase,type='l',log='x',
     xlab="Period (sec)", ylab="degrees",
     main="Transfer Function Phase")



layout(1)

XPS<-spec.pgram(ts, spans = NULL, kernel=NULL, taper = 0,
          pad = 0, fast = TRUE, demean = TRUE, detrend = TRUE,
          plot = TRUE, na.action = na.fail, col=c("#69b3a2","#DF3E6B"),
          main="Cross Power Spectrum",
          ylab="Power Spectrum",
          xlab="Frequency")

spec.pgram(inp, spans = NULL, kernel=NULL, taper = 0,
          pad = 0, fast = TRUE, demean = FALSE, detrend = TRUE,
          plot = TRUE, na.action = na.fail)



# III CPSD

cpsd( inp,
      window = NextPow2(sqrt(NROW(inp))),
      overlap = 0.5,
      nfft = ifelse(is_scalar(window), window, length(window)),
      fs = 1,
      detrend = "none")
```

```
XPSD<-pwelch(
  inp,
  window = NextPow2(sqrt(NROW(inp))),
  overlap = 0.5,
  nfft = if (is_scalar(window)) window else length(window),
  fs = 1,
  detrend = "none",
  range = if (is.numeric(x)) "half" else "whole"
)

plot(
  XPSD,
  xlab = NULL,
  ylab = "Power (Log Scale)",
  main = "Cross Power Spectral Density",
  plot.type = "cross-spectrum",
  yscale = "log")


plot(
  XPSD,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  plot.type = "spectrum",
  yscale = "log")


# IV Squared COH

plot(
  XPSD,
  xlab = NULL,
  ylab = NULL,
  main = "Ordinary Coherance",
  plot.type = "coherence",
  yscale = "log")


# V COMPLEX COH


attributes(XPSD)

CO<-XPSD$coh
co<-sqrt(CO)
head(co)
head(CO)

XPSD2<-XPSD
```

```
XPSD2$coh<-co

plot(
  XPSD2,
  xlab = NULL,
  ylab = "Coherency Magnitude",
  main = "Complex Coherance",
  plot.type = "coherence",
  yscale = "log")

head(XPSD$freq)
imp<-as.data.frame(inp)
dt<-1
lp<-2*dt
up<-floor(nrow(imp)/3)*dt

CH<-analyze.coherency(imp, my.pair = c("fx", "gx"), loess.span = 0.75,
                dt = 1, dj = 1/20,
                lowerPeriod = 2,
                upperPeriod = 330,
                window.type.t = 1, window.type.s = 1,
                window.size.t = 5, window.size.s = 1/4,
                make.pval = TRUE, method = "white.noise", params = NULL,
                n.sim = 100,
                date.format = NULL, date.tz = NULL,
                verbose = TRUE)

CH$Coherency

wc.image(CH$Coherency)

wc.image(CH, which.image="wc", timelab="Time", periodlab="Period",
        main="Coherency",
        legend.params=list(lab="Coherence Levels", lab.line=3.5,
            label.digits=3))
```