Exploring Ramanujan Sums in Digital Signal Processing

Subhojit Sarkar

Exploring Ramanujan Sums in Digital Signal Processing

Dissertation submitted in partial fulfillment of the requirements for the degree of

Master of Technology
in
Computer Science

by

Subhojit Sarkar
[Roll No: CS-1507]

under the guidance of

Dr. Sarbani Palit
Assistant Professor
Computer Vision and Pattern Recognition Unit

Indian Statistical Institute
Kolkata-700108, India
July 2017

Abstract

The problem of estimation of period lengths from a “mixture” of periodic signals is a well-studied topic in the field of digital signal processing. Various methods have been proposed in the past, to extract signal periods. The extraction of periods of sinosoidal components in particular has attracted much attention. The more general problem of period determination when the mixture is of periodic but non-sinosoidal signals was first addressed in 1997 to separate foetal ECG from the maternal ECG. The algorithm used the **Singular Value Decomposition** method, but it had the stringent requirement of a large difference in the strengths of the individual signals. That is, if we have two signals having similar amplitude levels added together, the SVD method cannot be used. (Since the foetal ECG is generally much weaker than the maternal one, this method was sufficient for the problem.) Recently, using a summation proposed by the great Indian mathematician S. Ramanujan in 1918, certain methods have been developed. These algorithms also have good noise performance, which was usually absent in the older methods. We examine these algorithms in the first part of the thesis. We also point out some of the drawbacks of a few of the older methods.

While one of the main problems of signal processing is the identification of periods from a signal mixture, another equally important problem is their separation, that is, **reconstructing** the original signals, the “mixture” of which produced the signal under consideration. We will consider two such “mixing” operations, **addition** and **multiplication** in this thesis. There are methods such as filtering which have been used in the past to reconstruct the signals. Even Singular Value Decomposition has been used for this purpose. We propose an alternative method to “almost” separate the signals given a few conditions. The conditions are the mutual co-primality amongst the signal periods and a signal length of the LCM of the periods. We prove that “exact” reconstruction is never possible, for any given added or point by point multiplied signals. We then go on to show that using very little computation time and hardware support, we can very simply get back the original signals, with nominal changes. For the additive case, we get the initial signal with an offset and for the multiplicative case, we get a scaled version. We also show that this method gives better noise performance for white noise, under some conditions. Finally we show how, under few more strict conditions, even exact reconstruction of the initial signals in the multiplicative case is possible.

To my close ones. You know who you are.

CERTIFICATE

This is to certify that the dissertation entitled **"Exploring Ramanujan Sums in Digital Signal Processing"** submitted by **Subhojit Sarkar** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of **Master of Technology in Computer Science** is a bonafide record of work carried out by him under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in my opinion, has reached the standard needed for submission.

Dr. Sarbani Palit
Assistant Professor,
Computer Vision and Pattern Recognition Unit,
Indian Statistical Institute,
Kolkata-700108, INDIA.

Acknowledgements

I am indebted to my mentor and guide Dr. Sarbani Palit of the Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, Kolkata for her constant encouragement. I am thankful to Mr. Ankan Bhattacharya, former research fellow at ISI, for long hours of brainstorming sessions in the laboratory. Without their help, I would not have been able to do this work. A big thank you to all my family members at home for their constant encouragement. Finally, I wish to thank Prof. Bivas Dam, Department of Instrumentation and Electronics Engineering, Jadavpur University, for instilling the love of signal processing in me during my undergraduate years.

Contents

List of Figures **7**

1 Introduction **9**

2 Period Estimation **12**
- 2.1 Successive Differencing . . . 12
- 2.2 SVD . . . 12
- 2.3 Ramanujan Sums . . . 15
  - 2.3.1 Ramanujan Representation . . . 22
  - 2.3.2 Nested Periodic Matrices . . . 23

3 Signal Reconstruction **28**
- 3.1 Addition . . . 28
  - 3.1.1 Preliminaries . . . 28
  - 3.1.2 The Periodic Case . . . 28
  - 3.1.3 A Possible Solution? . . . 29
  - 3.1.4 The "Almost" Solution . . . 34
  - 3.1.5 Algorithm . . . 35
  - 3.1.6 Complexity . . . 35
  - 3.1.7 Constraints . . . 36
  - 3.1.8 More than two signals? . . . 36
  - 3.1.9 Noise Analysis . . . 36
- 3.2 Multiplication . . . 37
  - 3.2.1 Preliminaries . . . 37
  - 3.2.2 The Periodic Case . . . 37
  - 3.2.3 A Possible Solution? . . . 38
  - 3.2.4 The "Almost" Solution . . . 38
  - 3.2.5 Algorithm . . . 38
  - 3.2.6 Complexity . . . 39

4 Experiments and Results **40**
- 4.1 Addition . . . 40
  - 4.1.1 Two signals . . . 40
  - 4.1.2 Three signals . . . 40
  - 4.1.3 Noise Performance . . . 42
  - 4.1.4 Comparison with SVD based method . . . 43
- 4.2 Multiplication . . . 45
  - 4.2.1 Two signals . . . 45
  - 4.2.2 Three signals . . . 45

4.2.3 Exact reconstruction 46

**5 Conclusion and
Scope for Future Work 48**

6 Bibliography 49

List of Figures

1.1 Monthly mean total sunspot number [1/1749 - 12/2016]. Source: WDC-SILSO, Royal Observatory of Belgium, Brussels. 10
1.2 EEG data of a patient with an onset of a seizure. Source: UPenn and Mayo Clinic's Seizure Detection Challenge. 10
1.3 A "continuous" time sinosoid and its sampled version. 11

2.1 A random signal having period length 9, repeated over 11 periods. . . 13
2.2 Power spectrum of the function $d_k(n)$ in the previous figure. We notice that the minimas are at multiples of 9. 13
2.3 Two random signals of period length 5 and 9, added and truncated to length 40. 14
2.4 Power spectrum of the function $d_k(n)$ in the previous figure. We cannot draw any conclusion from this. 14
2.5 A periodic signal of length 99, having period 9. 15
2.6 The $\frac{\sigma_1}{\sigma_2}$ values for candidate period lengths from 3 to 33. 16
2.7 Two periodic signals having periods 5 and 9, added with 40dB white Gaussian noise, taken over 99 samples. Signals are of comparable strengths. 16
2.8 $\frac{\sigma_1}{\sigma_2}$ values from SVD based method applied to the previous signal. We seen no distinct peaks. 17
2.9 Two periodic signals having periods 5 and 9, added with 40dB white Gaussian noise, taken over 99 samples. The signal strengths are in the ratio 1:100 respectively. 17
2.10 $\frac{\sigma_1}{\sigma_2}$ values of candidate periods from SVD based method applied to the previous signal. Peaks at candidate period lengths of multiples of 9 are clearly found out. 18
2.11 Extracting periods from sunspot numbers. Works only when signal length is a multiple of 132(3216 in this case). 24
2.12 Period detection of a truncated signal having hidden periods 7 and 11. Signal strengths are comparable. 26
2.13 Period detection of a truncated signal having hidden periods 7 and 11. Signal strengths are in the ratio 1:4 respectively. 27

3.1 The model assumed for addition 29
3.2 Proof of the dependency in equations 33
3.3 The noise model assumed. N_1 and N_2 are white noises. 37

4.1 Reconstructing 2 signals in additive case 41
4.2 Reconstructing 3 signals 42

4.3 Noise analysis in the additive case 43
4.4 Comparison between our method and SVD for the additive 2 signal case. 44
4.5 Separation of 2 point by point multiplied signals. 45
4.6 Separation of 3 point by point multiplied signals. 46
4.7 Exact reconstruction in the point by point multiplicative case. 47

Chapter 1

Introduction

A signal, in layman's terms, can be defined as something which provides *information*. There are signals all around us, natural and man-made. Any quantity exhibiting variation (or lack of it) in time or in space is potentially a signal that might provide information on the status of a physical system, among other possibilities. The temperature of a day can be regarded as a signal. The variation of share prices of a company can be a signal. Speech, image, video and text are all signals. Human body is home to a plethora of electrical signals; the brain operates mostly on electrical signals. A good basis for signals and their properties are found in [1], [6].

It has been reported [2] that the seizures in an epileptic patient are usually preceded by some changes in the patient's EEG activity. If we can somehow manage to process the EEG signal in a patient, we can take measures to control the situation. Another potent area of research is non-invasive diagnosis of blood sugar, wherein photos of a patient's skin can be taken to be a signal and processed accordingly.

It is no wonder, therefore, that the processing of these signals is a wide field of study.

Basics

A signal may be continuous, or discrete. Intuitively, all signals that vary continuously are called **continuous** signals. On the other hand, those which vary only at specified points, remaining steady at other times are called **discrete** signals.
Continuous time signals, to be stored, processed and studied would require an *infinite* amount of resources. Indeed, how can we store infinite precision in a modern computer which has finite memory? For example, if we were to continuously record the temperature in a room (temperature change being a random process) for even a second, we would need infinite space to store the infinite amount of data recorded! Hence, all processing of continuous time signals done in a computer are in the discrete domain. In particular, we sample these continuously varying signals at *specified* instances, and treat the signal to be constant between these instances. In general, the more densely we sample the signal, the more precisely can our sampled discrete wave mimic the continuous signal. In Fig. 1.3, though we claim that the sinosoid is "continuous", it is actually produced by a computer and hence discrete. However, it serves for illustration purposes.

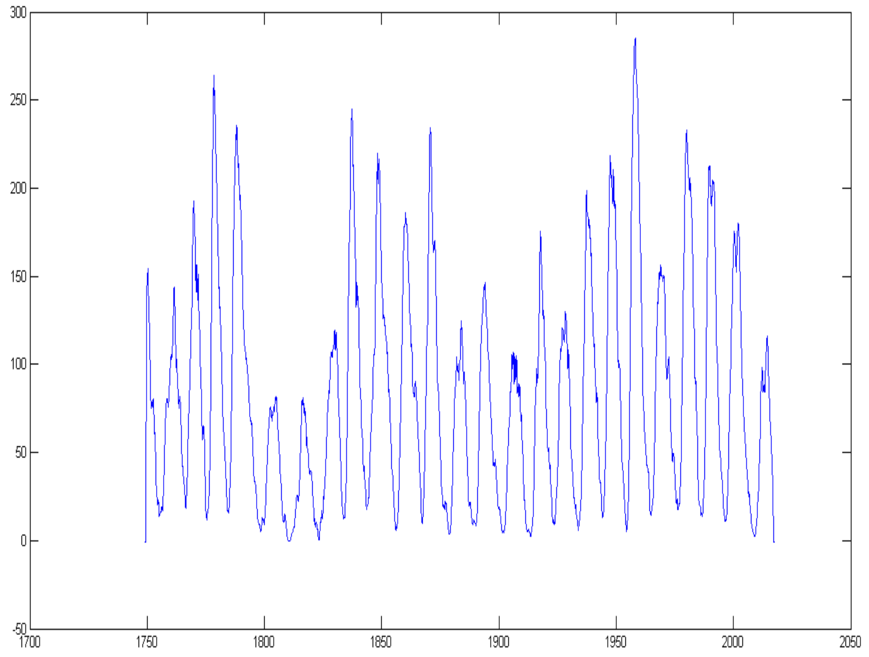


Figure 1.1: Monthly mean total sunspot number [1/1749 - 12/2016]. Source: WDC-SILSO, Royal Observatory of Belgium, Brussels.

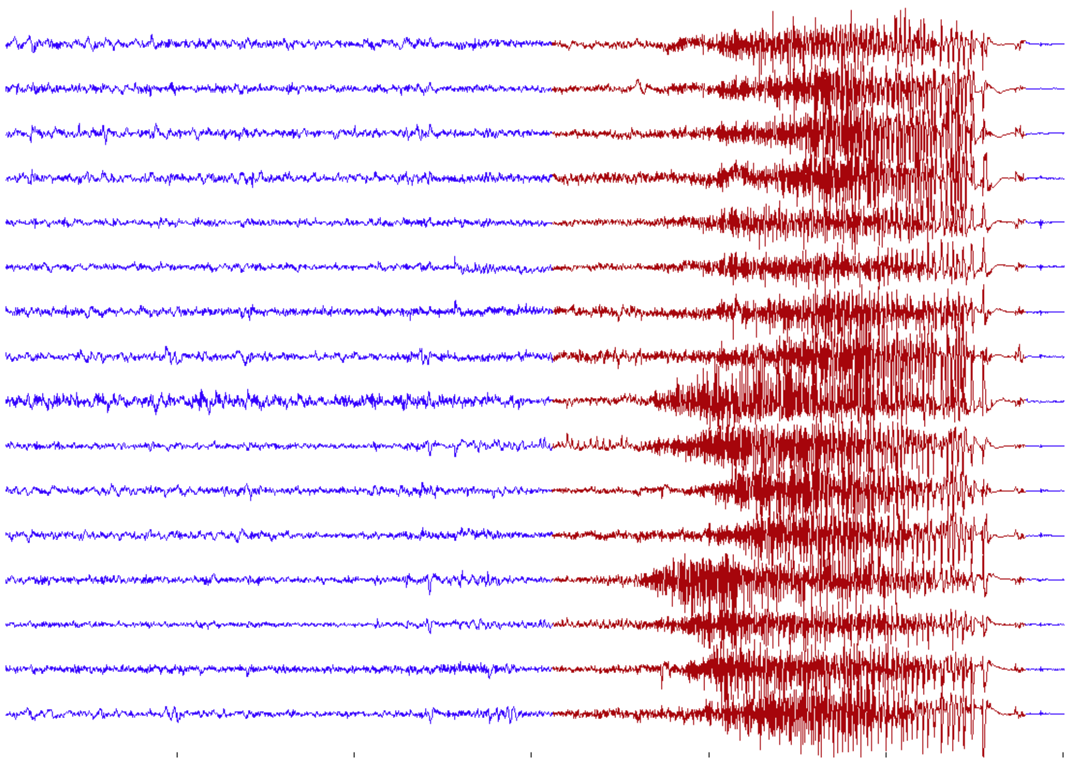


Figure 1.2: EEG data of a patient with an onset of a seizure. Source: UPenn and Mayo Clinic's Seizure Detection Challenge.

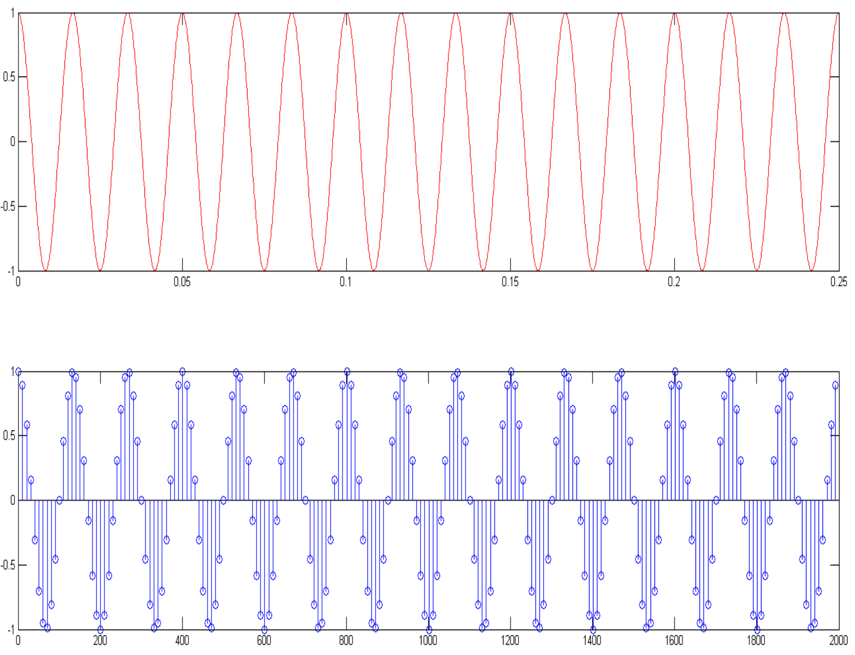


Figure 1.3: A "continuous" time sinosoid and its sampled version.

Again, in a computer, the value sampled at a particular instance has to be stored with finite precision. Hence, there appears another error, called the **quantization** error, in the processing of signals in a digital computer. Quantization error can be defined as process of mapping a large set of input values to a (countable) smaller set, usually attained by rounding off. Thus, digital signal processing suffers from two inaccuracies; one due to finite instances that can be sampled, and the rounding off of sampled values.

Periodicity

It has been observed that many real world signals tend to *repeat* themselves after some finite amount of time. If this time period is *constant*, then the signals are **periodic signals** and the time interval is called its **period**. If $x(n)$ is the given signal and N is the *smallest* number for which

$$x(n+N) = x(n)$$

then N is the *period* of $x(n)$.
The sunspot data, shown in the first figure, however exhibit some *variation* in its period, that is, in this case, N is not constant. It varies around 11 years, but is **not** fixed. Such signals are called **quasi-periodic signals**.

Chapter 2

Period Estimation

Estimation of periods is one of the most well-studied problems in the field of signal processing. Various methods have been proposed in the past, of which we will briefly discuss a few here.

2.1 Successive Differencing

Given a signal $x(n)$ of length N, we construct a function

$$d_k(n) = x(n+k) - x(n) \quad \forall n$$

Now, we see that if the period happens to be the selected k, or a factor of k, $d_k(n) = 0$. Thus, we can use this method to take the power spectrum of $d_k(n)$ and the the minimum value of k corresponding to the minima of the power spectrum is the period.

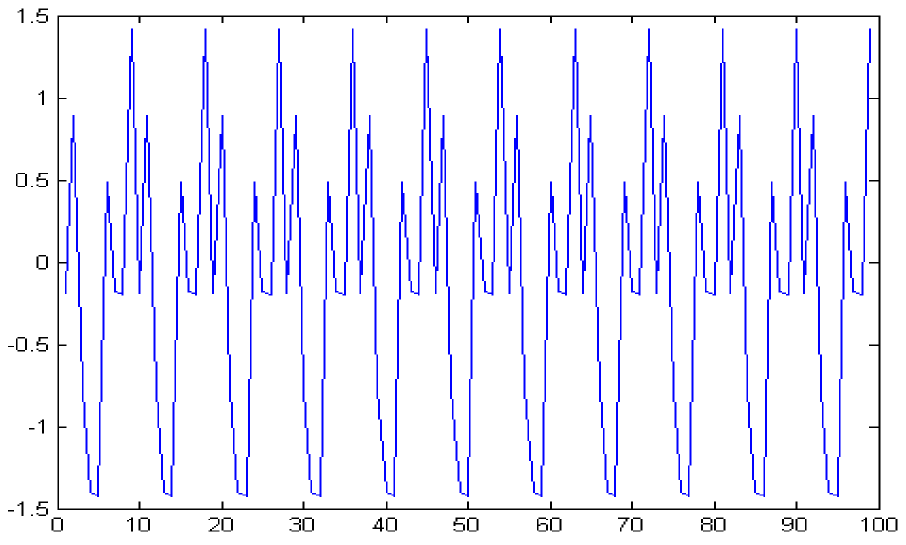
This method is a brute force method, and hence is not preferable in practice. Besides, this will not work well for signals which have multiple hidden components, but whose lengths are less than the LCM of the periodicities. For example, in the Fig. 2.3, we have taken two signals of period 9 and 5, added them and taken a truncated added signal of length 40. Since the effective period length of the periodic signal is 45, no detection of period is possible.

2.2 SVD

Singular Value Decomposition (SVD) can be carried out on the given signal to get the period. The method was proposed in [4]. It was found that if a periodic signal was *reshaped* into a matrix such that the number of columns became the period length, the **singular values** of that matrix were all very close to zero except the first one. This way,the periods can be found.
Let us assume that A is a signal of length N, having period length $m = \frac{N}{n}$, i.e., we have n cycles of A. **SVD** is defined as

$$A_{m\times n} = U_{m\times m}.\sigma_{m\times n}.V^T_{n\times n}$$

The method followed is as follows.
We *arrange* the signal in the form of a matrix, varying the row size as our **candidate periods**, *i.e.*, if we know that $m_1 \leq m \leq m_2$, we can arrange the signal as



[H]

Figure 2.1: A random signal having period length 9, repeated over 11 periods.

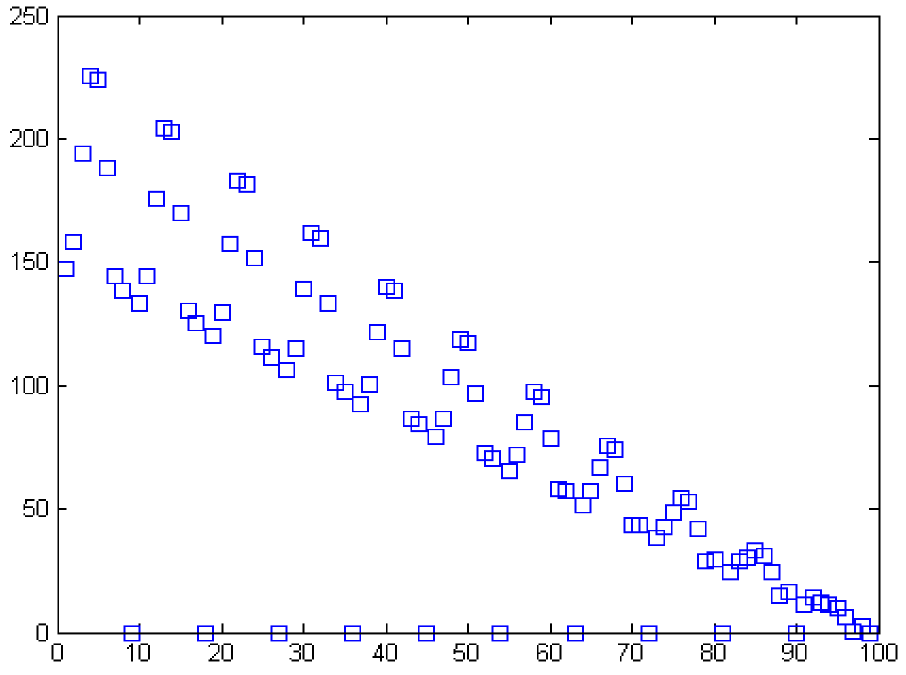


Figure 2.2: Power spectrum of the function $d_k(n)$ in the previous figure. We notice that the minimas are at multiples of 9.

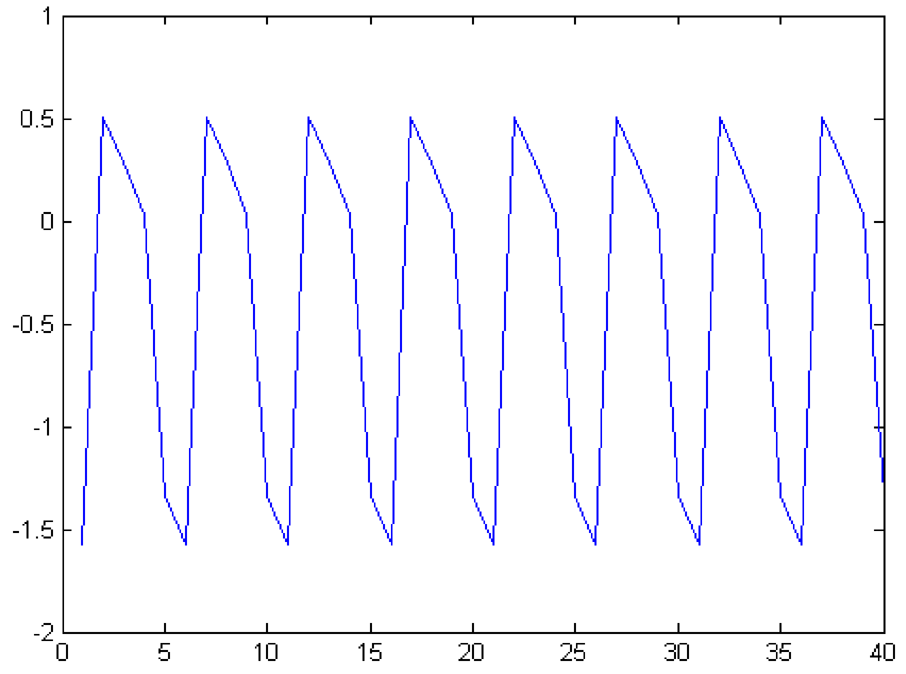


Figure 2.3: Two random signals of period length 5 and 9, added and truncated to length 40.

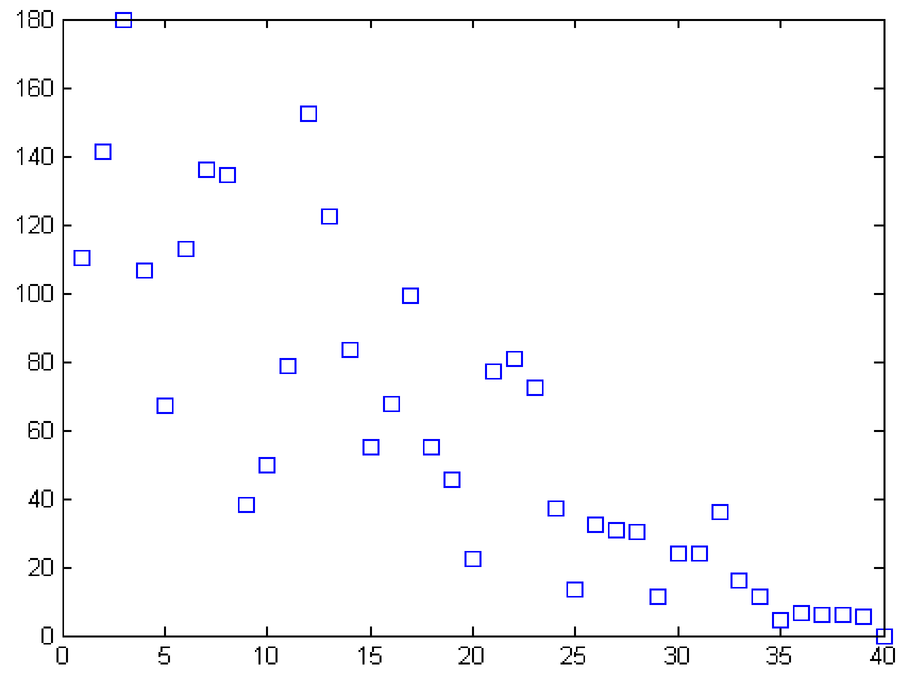


Figure 2.4: Power spectrum of the function $d_k(n)$ in the previous figure. We cannot draw any conclusion from this.

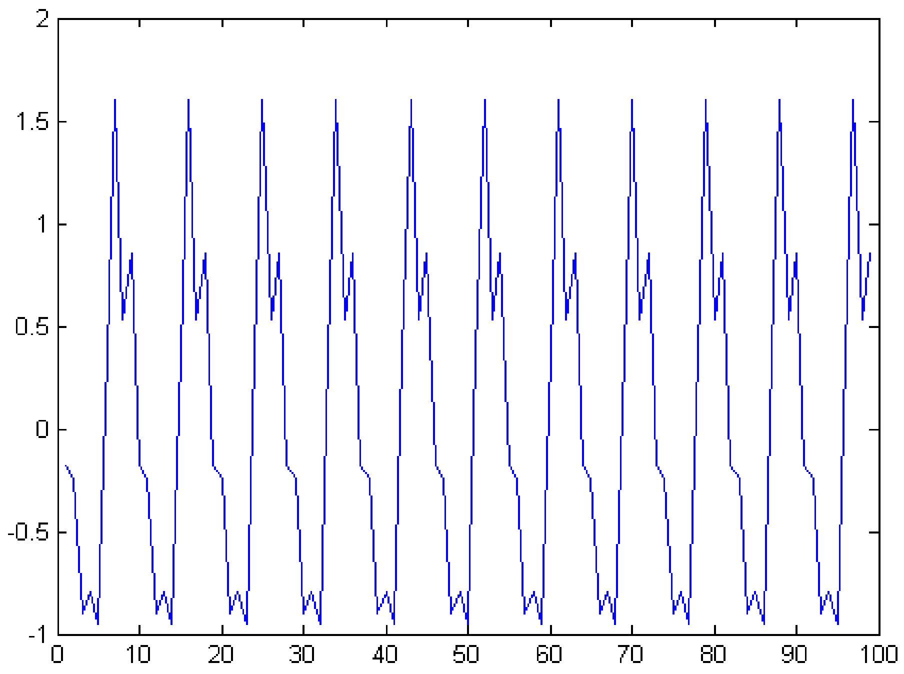


Figure 2.5: A periodic signal of length 99, having period 9.

matrices having row sizes varying from m_1 to m_2. It has been shown that after this decomposition, if the row size happens to be the period length m, the ratio of the first two diagonal elements of σ, $\frac{\sigma_1}{\sigma_2}$ is *very* high (if the signal has more than 1 signal, then the ratio of subsequent $\frac{\sigma_i}{\sigma_{i+1}}$ is high). In Fig. 2.5, we take a signal of period length 9, taken over 11 cycles. We try to find m, starting from a possible *candidate* period of 3 and going upto $99/3 = 33$. We see that the very high peaks are at 9,18 and 27. This means that when the signal is *arranged* as a matrix having number of columns 9, 18 and 27, we get a very high $\frac{\sigma_1}{\sigma_2}$, thus letting us know that 9 is a period. If there are multiple periodic signals contained in the signal under consideration, this method only works when the individual signals are of *considerable* different strengths. This is shown in Fig. 2.8.

2.3 Ramanujan Sums

In the year 1918, the Indian mathematician Srinivas Ramanujan came up with a summation [5], which he showed could be used to represent several well-known arithmetic functions.
The sum, now called the **Ramanujan Sum** has the following form

$$c_q(n) = \sum_{\substack{k=1 \\ (k,q)=1}}^{q} e^{j2\pi kn/q}$$

where $(k, q) = 1$ signifies that k and q are relatively co-prime. In other words, (k, q) denotes the **Greatest Common Divisor** of k and q. For example, if $q = 12$,

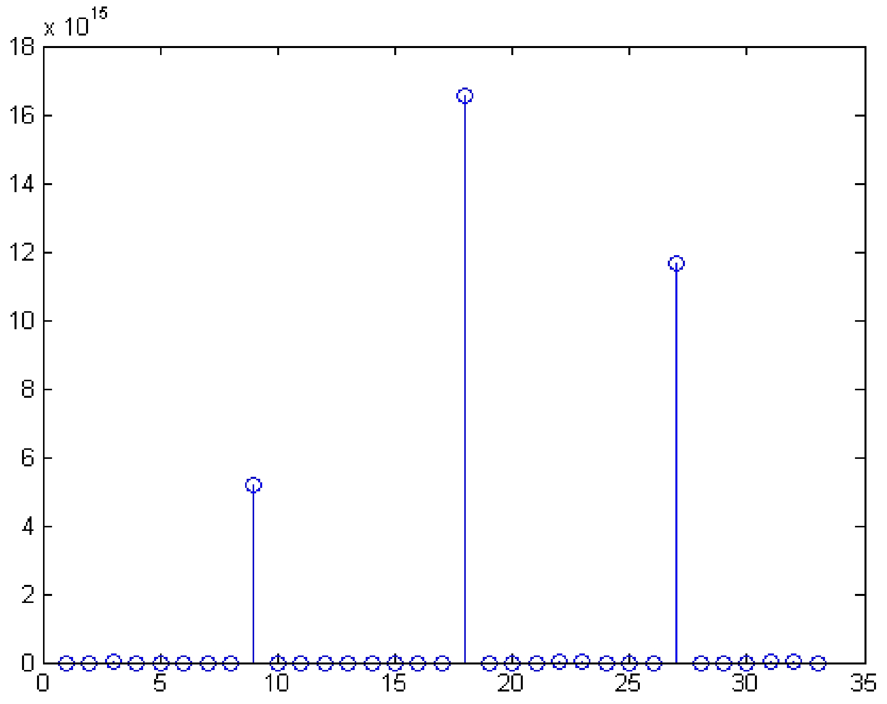


Figure 2.6: The $\frac{\sigma_1}{\sigma_2}$ values for candidate period lengths from 3 to 33.

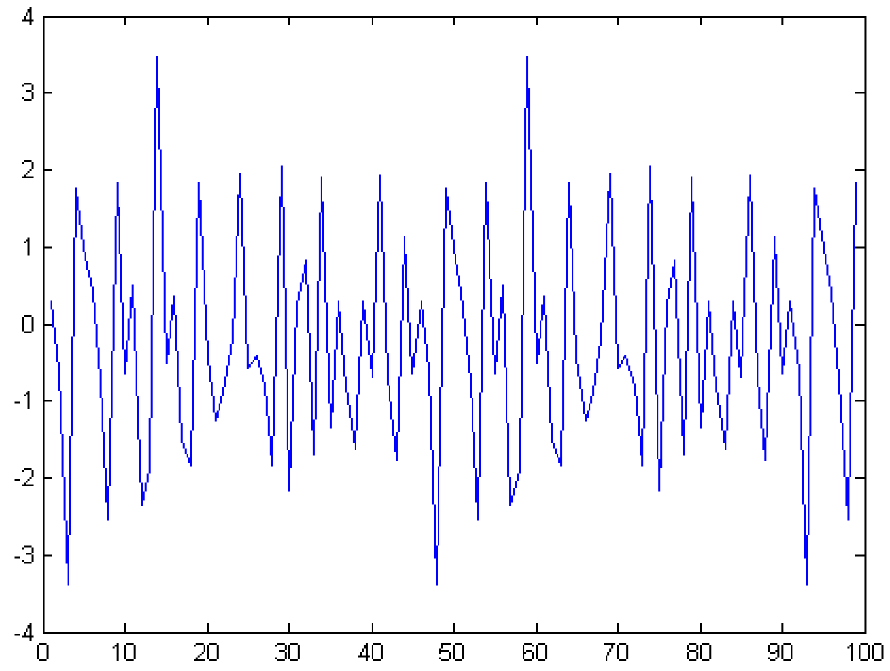


Figure 2.7: Two periodic signals having periods 5 and 9, added with 40dB white Gaussian noise, taken over 99 samples. Signals are of comparable strengths.

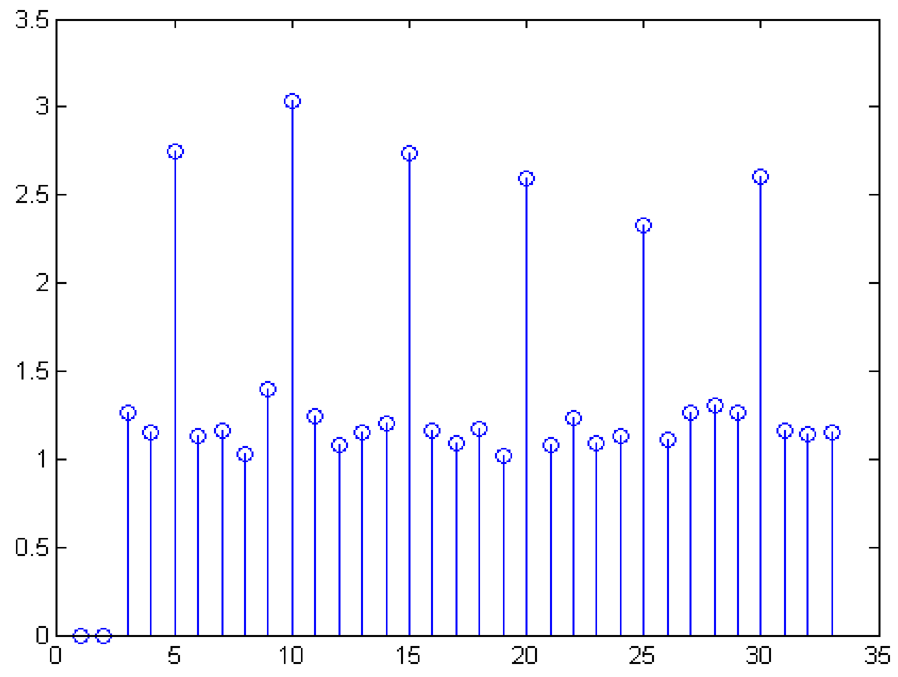


Figure 2.8: $\frac{\sigma_1}{\sigma_2}$ values from SVD based method applied to the previous signal. We seen no distinct peaks.

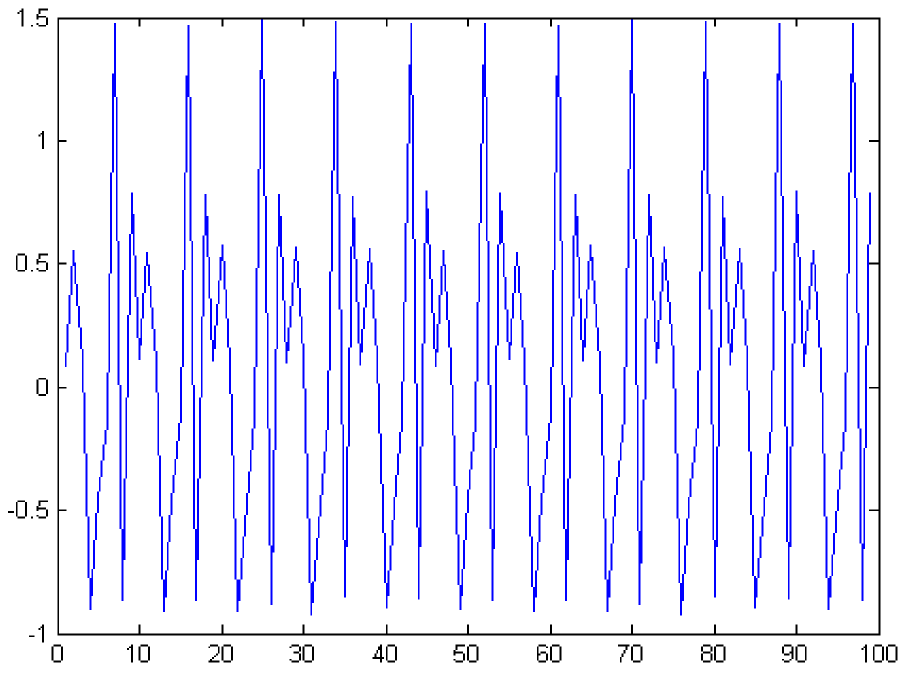


Figure 2.9: Two periodic signals having periods 5 and 9, added with 40dB white Gaussian noise, taken over 99 samples. The signal strengths are in the ratio 1:100 respectively.

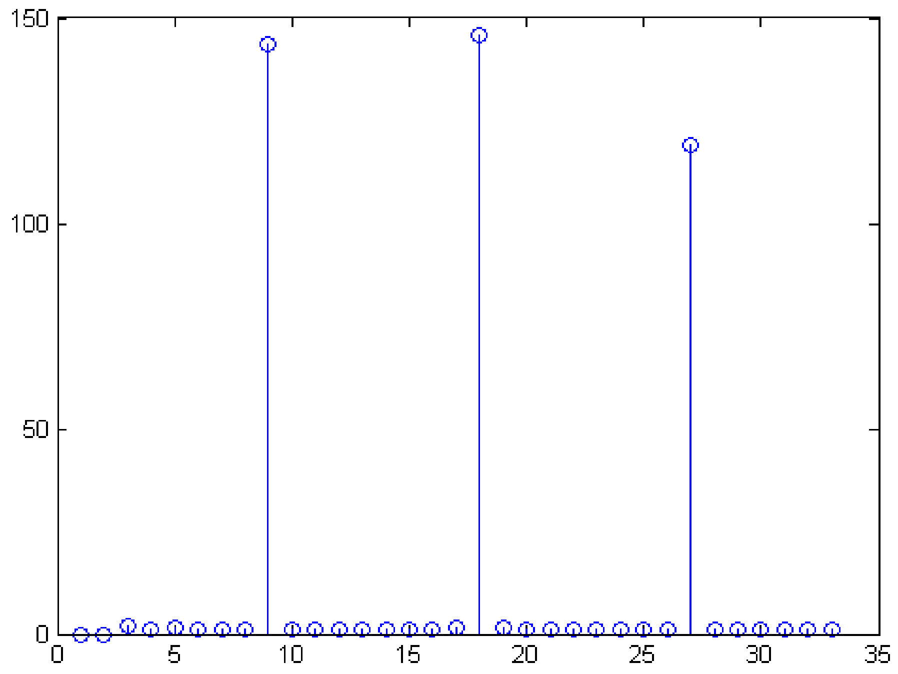


Figure 2.10: $\frac{\sigma_1}{\sigma_2}$ values of candidate periods from SVD based method applied to the previous signal. Peaks at candidate period lengths of multiples of 9 are clearly found out.

then the values of k co-prime to q are 1, 5,7 and 11. Hence,

$$c_{10}(n) = e^{j2\pi n/12} + e^{j10\pi n/12} + e^{j14\pi n/12} + e^{j22\pi n/12}$$

Ramanujan observed that a lot of *arithmetic* functions could be represented as a linear combination of $c_q(n)$s, ie,

$$x(n) = \sum_{q=1}^{\infty} \alpha_q c_q(n), n \geq 1$$

An *arithmetic* function is usually an infinite sequence of numbers, defined for $1 \leq n < \infty$ and usually integer valued. For example, **Euler's Totient function**, $\phi(n)$, is a very well-known arithmetic function. For a given positive integer n, it is defined as the *number of integers between 1 and n that are co-prime to n.* For example, $\phi(10) = 4$, since 1,3, 5 and 10 are co-prime to 10.
We will now point out some useful properties of this summation.
For ease of writing, we will often represent $e^{-j2\pi/q}$ as W_q. Also, (k, q) represents the GCD of k and q

Periodicity
We can easily observe that the sequence $c_q(n)$ repeats itself after every q values. This is because

$$\begin{aligned}
c_q(n+q) &= \sum_{\substack{k=1\\(k,q)=1}}^{q} e^{j2\pi k(n+q)/q} \\
&= \sum_{\substack{k=1\\(k,q)=1}}^{q} e^{j2\pi kn/q}.e^{j2\pi q/q} \\
&= \sum_{\substack{k=1\\(k,q)=1}}^{q} e^{j2\pi kn/q} \\
&= c_q(n)
\end{aligned}$$

Thus, the sequence $c_q(n)$ repeats after every q values.

Symmetric property
The sums are symmetric, because

$$\begin{aligned}
c_q(-n) &= \sum_{\substack{k=1\\(k,q)=1}}^{q} e^{j2\pi k(-n)/q} \\
&= \sum_{\substack{k=1\\(k,q)=1}}^{q} e^{j2\pi qn/q}.e^{-j2\pi kn/q} \\
&= \sum_{\substack{k=1\\(k,q)=1}}^{q} e^{j2\pi (q-k)n/q}
\end{aligned}$$

Let $(q-k) = k'$. Also, if $(k,q) = 1$, then $(q, q-k) = 1$. Therefore,

$$\begin{aligned}
c_q(-n) &= \sum_{\substack{k'=0\\(k',q)=1}}^{q} e^{j2\pi k'n/q} \\
&= c_q(n)
\end{aligned}$$

Armed with the *periodic* and *symmetric* properties, we can prove a stronger property, namely

$$c_q(n) = c_q(q-n)$$

Number of terms
The number of terms to be summed up for each $c_q(n)$ is of course equal to the number of integers between 1 and q that are *relatively co-prime* to q. Thus, the number of terms in the summation $c_q(n)$ is the **Euler's Totient function**, $\phi(q)$. Another

observation is that $c_q(0)=\phi(q)$.

Real terms
Another special property of **Ramanujan sums** is that despite the presence of the complex j in the summation, the numbers are **all** real. This is because since

$$(e^{j\theta})* = e^{-j\theta}$$

and because of the symmetric property of Ramanujan Sums

$$c_q(-n) = c_q(n)$$

we have

$$c_q^*(n) = c_q(n)$$

Hence, the sums are all real.
The Discrete Fourier Transform
We have the DFT equation of a length N function $x(n)$ given by

$$X(k) = \sum_{n=0}^{N-1} x(n).e^{-j2\pi kn/N}$$

Thus, for the Ramanujan sum $c_q(n)$, we have

$$\begin{aligned} C_q(k) &= \sum_{n=0}^{q-1} c_q(n).e^{-j2\pi kn/q} \\ &= \sum_{n=0}^{q-1}(\sum_{\substack{l=1 \\ (l,q)=1}}^{q} e^{j2\pi ln/q}).e^{-j2\pi kn/q} \\ &= \sum_{\substack{l=1 \\ (l,q)=1}}^{q} e^{j2\pi ln/q}.\sum_{n=0}^{q-1} e^{-j2\pi kn/q} \end{aligned}$$

If $(k,q) = 1$, then $\exists$ an $l = k$ in first summation. Thus, $e^{j2\pi ln/q}.e^{-j2\pi kn/q} = 1$
$\therefore \sum_{n=0}^{q-1} e^{j2\pi ln/q}.e^{-j2\pi kn/q} = q$ When $l \neq k$, then $\sum_{n=0}^{q-1} e^{-j2\pi(l-k)n/q} = 0$.
Thus,

$$C_q(k) = \begin{cases} q, & \text{if } (k,q) = 1 \\ 0, & \text{otherwise} \end{cases}$$

Integers
All the terms in the summation happen to be integers. Ramanujan showed a proof[5] by using the Mobius Function. We will show the proof as shown in [9], not only because this gives us a fast way to compute the sums, but also for the elegance of it.
In the definition of Ramanujan Sum, we have a set of fractions $\frac{k}{q}$ such that $(k,q) = 1$. This set of irreducible rationals has only $\phi(q)$ elements, unlike the set of all rationals

$$S_1 = \left\{\frac{h}{q} \mid 1 \leq h \leq q\right\}$$

which has q elements. We can write S_1 as a union of irreducible rationals. Let

$$S_2 = \left\{ \frac{a}{d} \mid 1 \leq a \leq d \, (a,d) = 1, d \mid q \right\}$$

Now, if $x \in S_1$, then $x = \frac{h}{q}$ and can be written as $\frac{a}{d}$ by cancelling the GCD between h and q. So $x \in S_2$.
Conversely, let $y \in S_2$. Then $y = \frac{a}{d}$ where $a \leq d$ and $d \mid q$. So we can rewrite $y = \frac{a.l}{q}$ for some integer l. Clearly, $al \leq q$, so $y \in S_1$. So, $S_1 = S_2$.
For example, if q=6, then

$$S_1 = \left\{ \frac{1}{6}, \frac{2}{6}, \frac{3}{6}, \frac{4}{6}, \frac{5}{6}, \frac{6}{6} \right\}$$

and

$$S_2 = \left\{ 1 \right\} \cup \left\{ \frac{1}{2} \right\} \cup \left\{ \frac{1}{3}, \frac{2}{3} \right\} \cup \left\{ \frac{1}{6}, \frac{5}{6} \right\}$$

Now, let $F(x)$ be any function in x, and let us consider evaluating the *sum* of the values of the function at uniformly spaced samples at $x = h/q$, where $1 \leq h \leq q$, and q is a fixed positive integer. Then,

$$\sum_{h=1}^{q} F\Big(\frac{h}{q}\Big) = \sum_{d|q} \sum_{\substack{a=1 \\ (a,d)=1}}^{d} F\Big(\frac{a}{d}\Big)$$

Thus, we can compute the sum of samples in two stages.
For example if $F(x) = 1 \forall x$, then the left hand side is q and the inner sum on the right side is $\phi(d)$. Thus,

$$\sum_{d|q} \phi(d) = q$$

a well-known result which is also used in the following sections.
Again, let $F(x)$=$e^{j2\pi nx}$, where n is a fixed integer. Now,

$$\begin{aligned} \sum_{h=1}^{q} F\Big(\frac{h}{q}\Big) &= \sum_{h=1}^{q} e^{j\frac{2\pi}{q}n} \\ &= \sum_{d|q} \sum_{\substack{a=1 \\ (a,d)=1}}^{d} e^{j\frac{2\pi a}{d}}n \end{aligned}$$

The left side is $q\delta((n))_q$ where

$$\delta((n))_q = \begin{cases} 1, & \text{if } q|n \\ 0, & \text{otherwise} \end{cases}$$

The inner sum on the right side is precisely the Ramanujan sum $c_d(n)$. Thus,

$$\sum_{d|q} c_d(n) = q\delta((n))_q$$

Hence,

$$c_q(n) = q\delta((n))_q - \sum_{\substack{q_k|q \\ q_k<q}} c_{q_k}(n)$$

where $q_k|q$ denotes that q_k are divisors of q.
Hence, we have done two things. **1.** Proved that if $c_1(n)$ is an integer, *all* $c_q(n)$ are integers.
2. Generated a fast recursive way to produce Ramanujan sums. **Examples**
A few examples of the summation are shown. One cycle is shown for each case.

$$\begin{aligned}
c_1(n) &= 1 \\
c_2(n) &= 1, -1 \\
c_3(n) &= 2, -1, -1 \\
c_4(n) &= 2, 0, -2, 0 \\
c_5(n) &= 4, -1, -1, -1, -1 \\
c_6(n) &= 2, 1, -1, -2, -1, 1 \\
c_7(n) &= 6, -1, -1, -1, -1, -1, -1 \\
c_8(n) &= 4, 0, 0, 0, -4, 0, 0, 0 \\
c_9(n) &= 6, 0, 0, -3, 0, 0, -3, 0, 0 \\
c_2(n) &= 4, 1, -1, 1, -1, -4, -1, 1, -1, 1
\end{aligned}$$

Many other properties were proved in [9], which we will not prove again here. We will instead delve into the problem solving part.
One thing was pointed out in [8]. In the equation

$$x(n) = \sum_{q=1}^{\infty} \alpha_q c_q(n), n \geq 1$$

the co-efficients α are usually done as

$$\alpha_q = \frac{1}{\phi(q)}(\lim_{M\to\infty} \frac{1}{M} \sum_{n=1}^{M} x(n)c_q(n))$$

However, most signals we come across in real life are *finite*. Hence, $x(n)$ equals zero for all n except possibly $1 \leq n \leq N$, N being the signal length.
$\therefore \lim_{M\to\infty} \sum_{n=1}^{M} x(n)c_q(n)/M = \lim M \to \infty \sum_{n=1}^{N} x(n)c_q(n)/M \to 0$
Hence, new representations are to be found to apply this summation to finite signals.

2.3.1 Ramanujan Representation

One approach was proposed in [8]. We consider the following expansion

$$x(n) = \sum_{q=1}^{N} \alpha_q c_q(n), 0 \leq n \leq N-1$$

where the first N sequences $c_q(n)$ are all used. In vector form,

$$X = A.\alpha$$

where

$$\begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} = A_N \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix}$$

It was shown that the matrix A is of full rank and the representation always holds good. But it is of little use if we want to find the periodicity.

Vaidyanathan [9] proposed the **Ramanujan Subspace** and proved certain properties. The subspace is developed as follows.
1. Form a circulant matrix of order $q \times q$ from the Ramanujan sum $c_q(n)$. Column space of this matrix is called the **Ramanujan subspace**
2. The first $\phi(q)$ columns of the the circulant matrix are linearly independent and they form the basis of the **Ramanujan Subspace** of dimension $\phi(q)$.
For example, the circulant matrix for Ramanujan sum of order 5 is as follows.

$$\begin{bmatrix} 4 & -1 & -1 & -1 & -1 \\ -1 & 4 & -1 & -1 & -1 \\ -1 & -1 & 4 & -1 & -1 \\ -1 & -1 & -1 & 4 & -1 \\ -1 & -1 & -1 & -1 & 4 \end{bmatrix}$$

$\because \phi(5) = 1$,
$\therefore$ the corresponding basis is

$$\begin{bmatrix} 4 & -1 & -1 & -1 \\ -1 & 4 & -1 & -1 \\ -1 & -1 & 4 & -1 \\ -1 & -1 & -1 & 4 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

Using this, [8] proposed another representation, which decomposed the signal down into orthogonal projections of Ramanujan subspaces of orders being the factors of the signal length. However, this method has the drawback of not finding any periods which are not divisors of the signal length. That is, only periods which are divisors of the signal length are identified. This is shown in Fig. 2.11, where we show the peak at the well-known 132 month period is present when the signal length is truncated to a length of a multiple of 132.

2.3.2 Nested Periodic Matrices

Vaidyanathan proposed an algorithm in [7]. Let us take a period P and all its divisors d_i, $1 \leq i \leq K$ in increasing order, including 1 and P. Let us consider a matrix of the form

$$A = [C_{d_1} \quad C_{d_2} \quad ...C_{d_k}]$$

with the following properties.

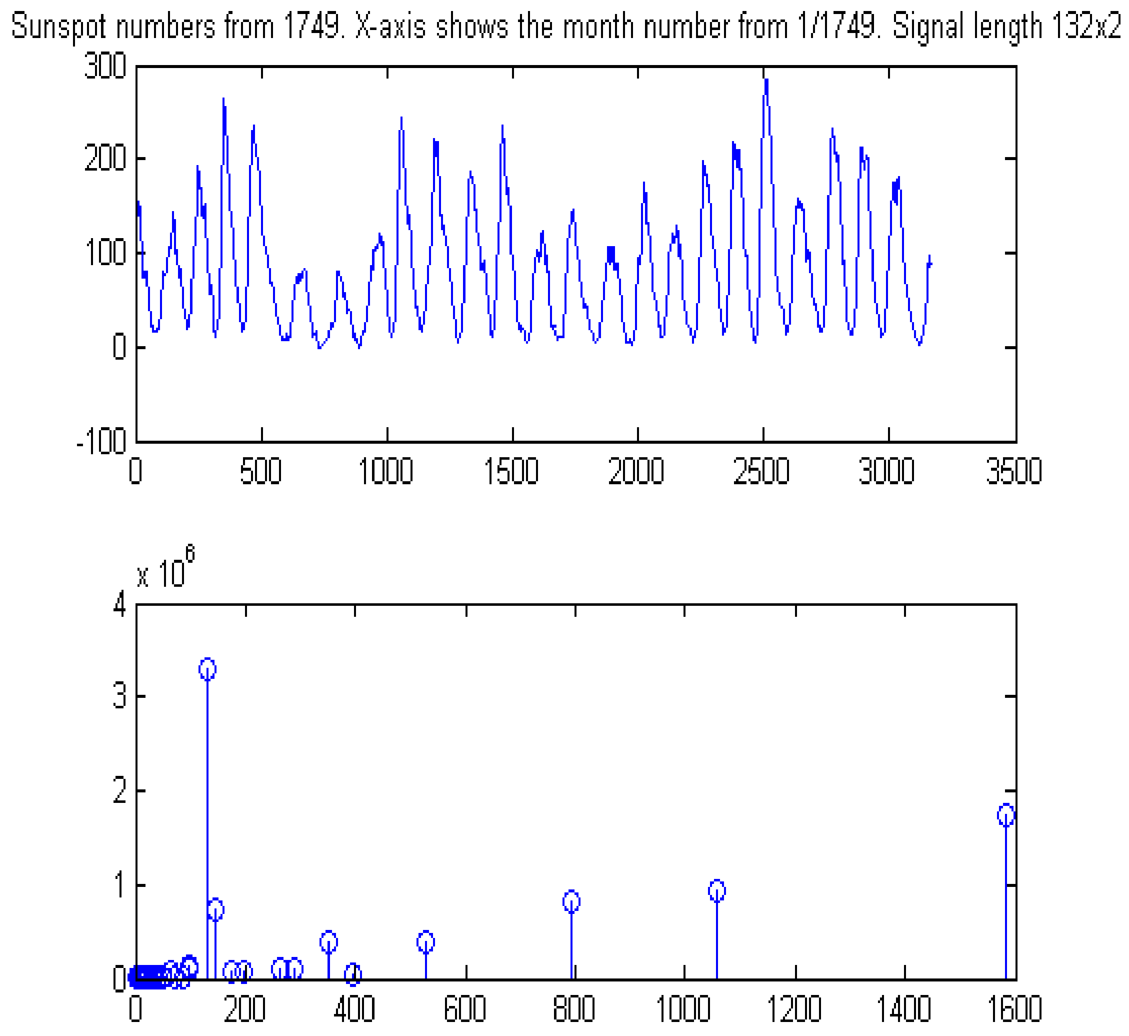


Figure 2.11: Extracting periods from sunspot numbers. Works only when signal length is a multiple of 132(3216 in this case).

1. Each C_{d_i} is a $P \times \phi(d_i)$ matrix, so that the total number of columns in $\sum_{d_i|P} \phi(d_i) = P$. Thus **A** is a $P \times P$ matrix.
2. Each column of C_{d_i} is a length P sequence of period d_i.
3. **A** has full rank P.

Such a matrix is called a **Nested Periodic Matrix**.

It has the useful property that *given any nested periodic matrix A, any $P \times 1$ vector y with period $q|P$ can be expressed as y=Ac where all those components of c are zero, that do not pair up with the q columns a_{k_j} that have periods equal to or a divisor of q.*

We can fill up such a matrix using **Ramanujan sums**. It was shown in [7] that there are a number of ways to fill up the matrix. However, Ramanujan Sums gave the best results. For example, if P=8,

$$A = \begin{bmatrix} 1 & 1 & 2 & 0 & 4 & 0 & 0 & 0 \\ 1 & -1 & 0 & 2 & 0 & 4 & 0 & 0 \\ 1 & 1 & -2 & 0 & 0 & 0 & 4 & 0 \\ 1 & -1 & 0 & -2 & 0 & 0 & 0 & 4 \\ 1 & 1 & 2 & 0 & 4 & 0 & 0 & 0 \\ 1 & -1 & 0 & 2 & 0 & 4 & 0 & 0 \\ 1 & 1 & -2 & 0 & 0 & 0 & 4 & 0 \\ 1 & -1 & 0 & -2 & 0 & 0 & 0 & 4 \end{bmatrix}$$

We now construct a dictionary as follows. Suppose the length of signal under consideration is N. If we have an idea that the maximum periodicity is P_{max}, then for all d going from 1 to P_{max}, we make a $d \times d$ nested periodic matrix, extending the columns upto N truncating the last matrix if necessary. We then select the $\phi(d))$ columns of this matrix which have period d. We form a fat dictionary for all d. For example, if $N = 5$ and P_{max}=5, we have our dictionary as

$$D = \begin{bmatrix} 1 & 1 & 2 & -1 & 2 & 0 & 4 & -1 & -1 & -1 \\ 1 & -1 & -1 & 2 & 0 & 2 & -1 & 4 & -1 & -1 \\ 1 & 1 & -1 & -1 & -2 & 0 & -1 & -1 & 4 & -1 \\ 1 & -1 & 2 & -1 & 0 & -2 & -1 & -1 & -1 & 4 \\ 1 & 1 & -1 & 2 & 0 & 0 & -1 & -1 & -1 & -1 \end{bmatrix}$$

If the given signal $x(n)$ is periodic in any value less than P_{max}, then it has to be a linear combination of the columns of the dictionary. So the following equation must have a solution y

$$x = Dy$$

However, since the matrix is fat, there maybe multiple solutions. We can try to find the least l_2 norm solution to an over-determined linear system. This will work because if we consider the following optimization problem,

$$min||Ey||_2 \quad \text{such that} \quad x = Dy$$

where D is a diagonal matrix whose ith diagonal entry $f(P_i)$ where P_i is the period of the ith column of D and $f(.)$ is some increasing function. Typically, $F(P) = P^2$ gives good results.

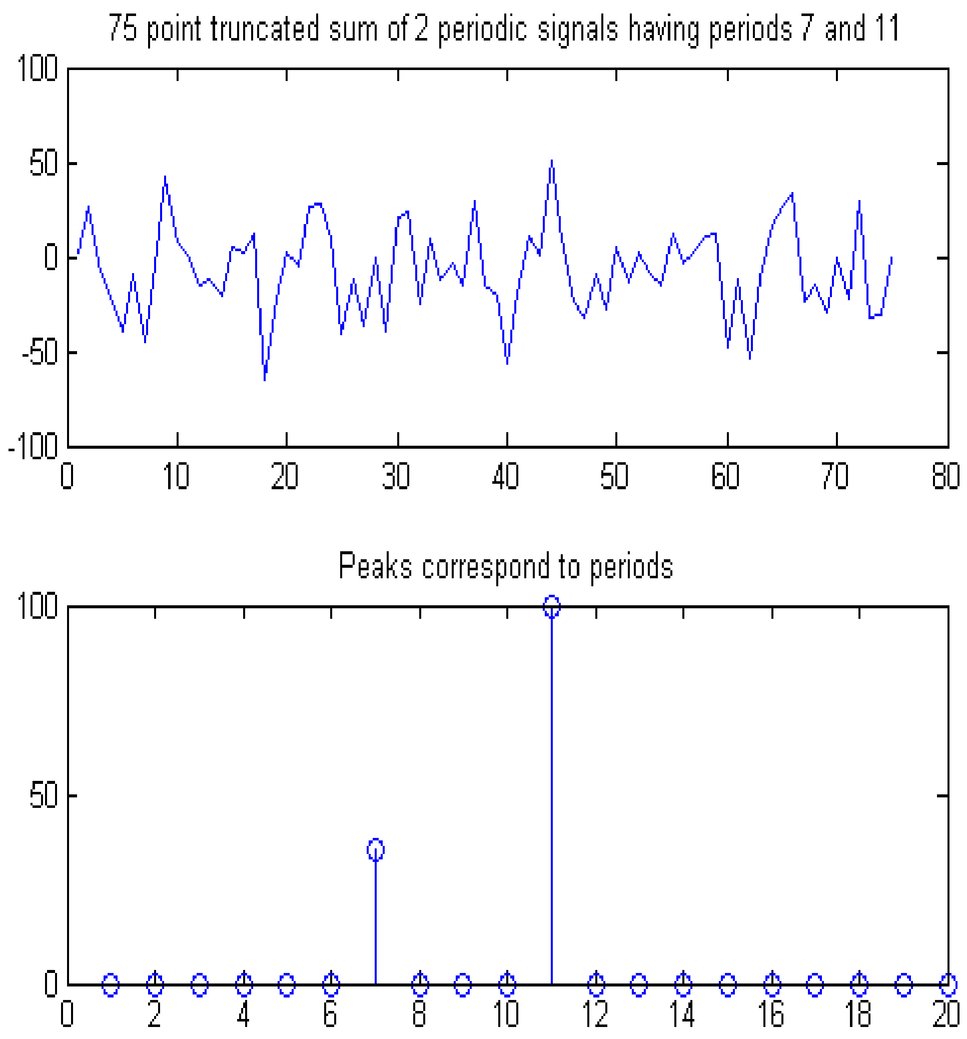


Figure 2.12: Period detection of a truncated signal having hidden periods 7 and 11. Signal strengths are comparable.

We see that the method gives good results for the case where the signal strengths are comparable, as shown in Fig 2.12.

However, in the case of different signal strengths, as shown in Fig. 2.13, the detection of the signals with smaller strength diminishes rapidly. For ratios less than 1:10, the smaller signal period is not determined.

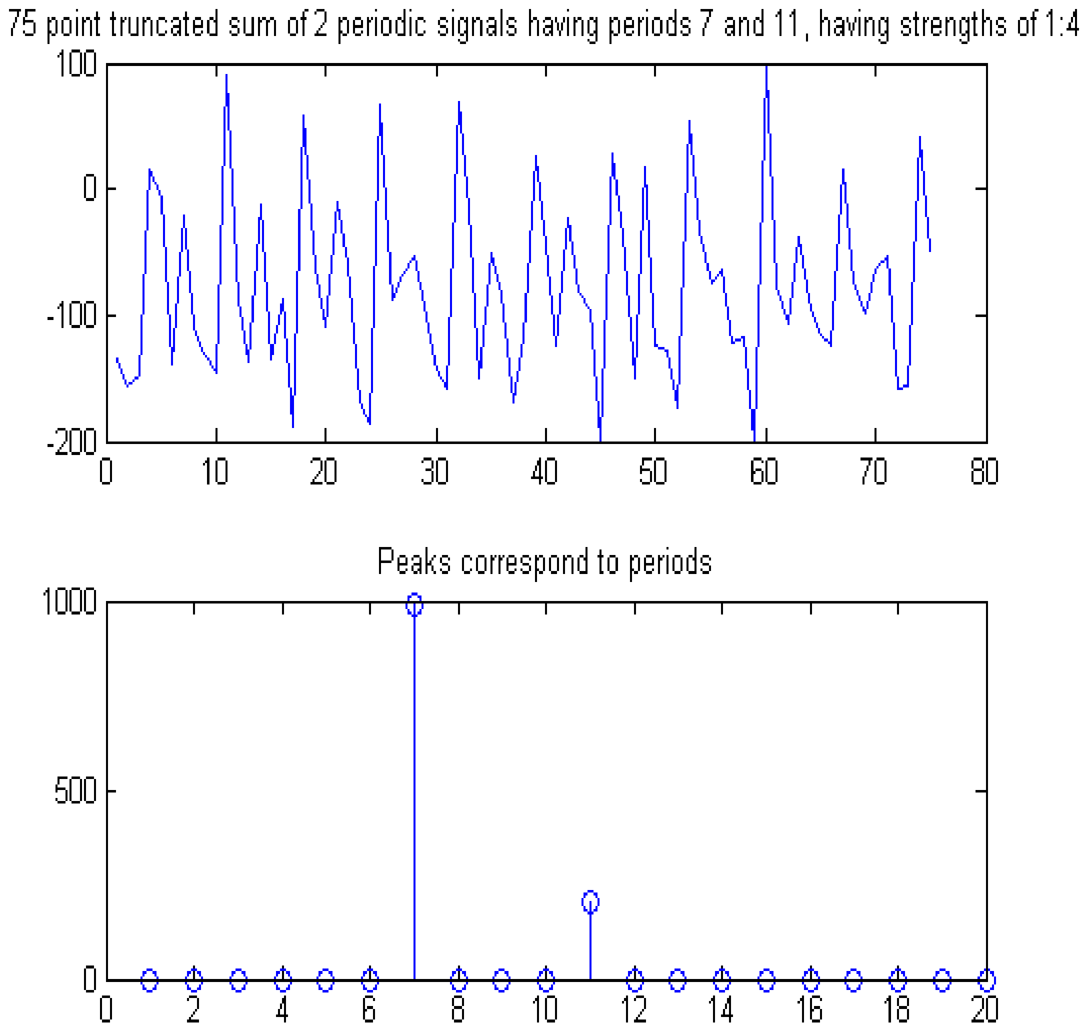


Figure 2.13: Period detection of a truncated signal having hidden periods 7 and 11. Signal strengths are in the ratio 1:4 respectively.

Chapter 3

Signal Reconstruction

One of the main problems of signal processing is the identification of periods in a given signal. Yet another problem is their separation, that is, reconstructing the original signals, the "mixture" of which produced the signal under consideration. We will consider two such "mixing", **addition** and **multiplication**.

3.1 Addition

3.1.1 Preliminaries

Let us assume two signals $x(n)$ and $y(n)$, each of length N, without loss of generality. We can always assume that they are of equal length, because if not, we can always pad the shorter one with zeros to make them of equal length. We define $z(n)$ as

$$z(i) = x(i) + y(i)$$

for $1 \leq i \leq N$.
Consequently $z(n)$ is also of length N. Hence, we will get exactly N equations of the form

$$x(i) + y(i) = z(i)$$

for $1 \leq i \leq N$. The simplest idea that comes to mind is that of solving a system of linear equations.
The number of unknowns is $P_1 + P_2$. Hence, we have N linearly independent equations and $N + N = 2N$ unknowns. Hence, this approach is unsuitable for non-periodic signals.

3.1.2 The Periodic Case

For example, let us take the example of a signal $z(n)$ which is the sum of two periodic signals, $x(n)$ and $y(n)$, having periodicities of P_1 and P_2. That is

$$x(n) = x(n + P_1)$$

$$y(n) = y(n + P_2)$$

and

$$z(n) = x(n) + y(n)$$

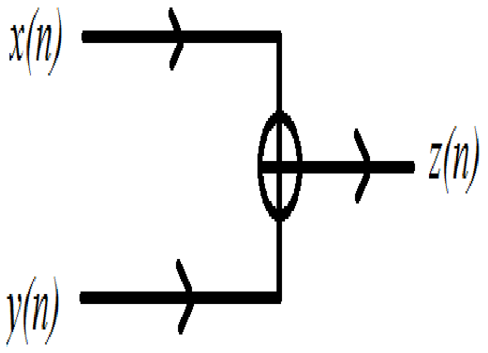


Figure 3.1: The model assumed for addition

Suppose we have been able to find P_1 and P_2 by any suitable method, or in our case, using the **Nested Periodic Matrices** and we now want to reproduce $x(n)$ and $y(n)$. Can we find a solution then?

3.1.3 A Possible Solution?

Assuming

$$x(n) = x(1), x(2), ..., x(P_1), x(1), x(2), ...$$

and

$$y(n) = y(1), y(2), ..., y(P_2), y(1), y(2), ...$$

we can try to solve for each x_i and each y_i given $z(n)$ of "sufficient" length.

An interesting fact was pointed out by Vaidyanathan [9]. The period length of $z(n)$ is always a divisor of $P_1.P_2$. For example, if $x(n) = \{1, 2, 3, 1, 2, 3\}$ and $y(n) = \{1, 1, 1, 1, 1, 1\}$, then the period of $z(n)$ remains 3. However, if we have $y(n) = \{3, 2, 1, 3, 2, 1\}$, then the period becomes 1. Thus, in both cases, the period of $z(n)$ becomes a divisor of $P_1.P_2$=3.

Here we prove a useful theorem.

Theorem 1: To separate two signals with known periodicities, given their added version, by "solving" a system of linear equations, is impossible. (*However, there is a very simple silver lining, if we are willing to compromise a bit.*)

Proof:
Case 1: $P_2 = kP_1$ *ie,* the length of $z(n)$ is P_2.
Therefore, we have only P_2 distinct equations available to us. Even if they are all linearly independent, their number is less than the number of unknowns, which in this case is $P_1 + P_2$. Hence, the equations cannot be solved.

Case 2: P_1 and P_2 are relatively co-prime.
We find a proof of this in [10]. However, we prove this in an entirely different way. We will prove a few lemmas for this case.

Lemma 1: If P_1 and P_2 are co-prime to each other, each value of $x(n)$ gets added to each value of $y(n)$ **exactly** once, in $P_1.P_2$ samples of $z(n)$.

Proof: The proof is by contradiction.
Let us consider the sample $x(i)$, where $0 < i \leqslant P_1$.
In $P_1.P_2$ samples of $z(n)$, $x(n)$ completed exactly P_2 periods.
Therefore, $x(i)$ appears exactly P_2 times.
Likewise, in $P_1.P_2$ samples of $z(n)$, $y(n)$ completed exactly P_1 periods.
Now, for $0 < l < P_2$ and $0 < k < P_1$ where k and l are integers, we can write

$$l.P_1 + i = k.P_2 + r.....(1)$$

where i is an integer such that $0 < i < P_1$ and r is an integer such that $0 < r < P_2$.
Now, let us assume that there exist two values of r, corresponding to a single i.
Therefore, by the same logic, there will be two different equations, each with different l and k (the properties of l and k are maintained).
Let the two equations be

$$l_1.P_1 + i = k_1.P_2 + r$$

and

$$l_2.P_2 + i = k_2.P_2 + r$$

.
Now subtracting one from the other, we get

$$(l_1 - l_2).P_1 = (k_1 - k_2).P_2$$

That is,

$$\frac{(l_1 - l_2)}{(k_1 - k_2)} = \frac{P_2}{P_1}$$

Now, $\because$ both l_1 and l_2 are less than P_2,
$\therefore$ $l_1 - l_2$ is also less than P_2.
Similar arguments can be put for k_1 and k_2.
However, we have assumed that P_1 and P_2 are relatively co-prime.
$\therefore$ $(l_1 - l_2) \geq P_2$, as any factor cancellation of P_2 is impossible.
However, we had proved above that $(l_1 - l_2) < P_2$.
Therefore, we have a contradiction. Thus, there can be a **single** r for a particular i.
This concludes the proof.

An immediate consequence of *Lemma 1* is the next well-known property, which van be proved in other ways as well. This is because the period of $z(n)$ is a divisor of $P_1.P_2$ and there is no repetition within the first $P_1.P_2$ values.

Property If P_1 and P_2 are co-prime to each other, then $z(n)$ has period $P_1.P_2$.

Corollary: All possible pairs of i and j, $1 \leq i \leq P_1$ and $1 \leq j \leq P_2$ form equations of the type

$$x(i) + y(j) = z(k), \quad 1 \leq z \leq P_1.P_2$$

Now, we can prove Theorem 1 for the co-prime case.
We know that $z(n)$ has a period $P_1.P_2$. Therefore, we have $P_1.P_2$ equations and exactly $P_1 + P_2$ unknowns. Now, we can form groups of equations as follows:
Let $z(\alpha) = x(r) + y(s)$.
Now, we can find some $u \neq r$ and $v \neq s$ such that $z(\beta) = x(u) + y(v)$.
Thus, we have

$$\begin{aligned} z(\alpha) &= x(r) + y(s) \\ z(\beta) &= x(u) + y(v) \\ z(\gamma) &= x(r) + y(v) \\ z(\delta) &= x(u) + y(s) \end{aligned}$$

Hence, we can write

$$z(\alpha) + z(\beta) = z(\gamma) + z(\delta)$$

Thus, from this set of equations, we have found one dependency.
More precisely, we can find exactly $(P_1 - 1).(P_2 - 1)$ pairs of u and v satisfying the required relation. Now, we claim that for *each* set of 4 equations selected as above, we will have one *linear dependency.* We first show this by an example and generalize later.
Example: Let $P_1 = 3$ and $P_2 = 4$.
Thus, $x(n)$, $y(n)$ and $z(n)$ respectively goes as

$$x(1), x(2), x(3), x(1), x(2), x(3), x(1), x(2), x(3), x(1), x(2), x(3)$$

$$y(1), y(2), y(3), y(4), y(1), y(2), y(3), y(4), y(1), y(2), y(3), y(4)$$

$$z(1), z(2), z(3), z(4), z(5), z(6), z(7), z(8), z(9), z(10), z(11), z(12)$$

Let us take $r = 1$ and $s = 4$.
Thus, we have $(3-1).(4-1)$ ways of selecting u and v. A quick observation reveals that the possible pairs are $(2, 1)$, $(2, 2)$, $(2, 3)$, $(3, 1)$, $(3, 2)$ and $(3, 3)$.

So, we have 6 possible quads when each is paired with $z(4) = x(1) + y(4)$. We can write the resulting 6 equations as follows.

$$\begin{aligned} z(4)+z(5) &= z(1)+z(8) \\ z(4)+z(2) &= z(10)+z(8) \\ z(4)+z(11) &= z(7)+z(8) \\ z(4)+z(9) &= z(1)+z(12) \\ z(4)+z(6) &= z(10)+z(12) \\ z(4)+z(3) &= z(7)+z(12) \end{aligned}$$

We see that $z(5)$, $z(2)$, $z(11)$, $z(9)$, $z(6)$ and $z(3)$ appear *exactly* once in the equations. Hence, these 6 equations are *linearly independent.*
Thus, though we have 12 equations initially, we have shown that at least 6 of them are linearly dependent on the others. Thus, we are left with at maximum $12-6=6$ linearly independent equations. However, our number of unknowns was $3+4=7$. Hence, the system of linear equations cannot be solved.
Now we move on to the general proof.

Let (i,j) denote a node of a graph. Thus, each possible pair of (i,j) is a node in the graph. Hence there are exactly $P_1.P_2$ nodes in the graph. There are also some unconnected vertices.
Now, from one (r,s) let us add edges to all (u,v) such that $r \neq u$ and $s \neq v$. Thus, we have exactly $(P_1-1).(P_2-1)$ edges. Now, let us consider a neighbours of (r,s), namely (u_1,v_1). Let

$$\begin{aligned} z(\alpha) &= x(r)+y(s) \\ z(\beta) &= x(u_1)+y(v_1) \\ z(\gamma) &= x(r)+y(v_1) \\ z(\delta) &= x(u_1)+y(s) \end{aligned}$$

Evidently,

$$z(\alpha)+z(\beta)=z(\gamma)+z(\delta)$$

Like this, we can form exactly $(P_1-1).(P_2-1)$ quads, thus eliminating an equation with each quad. Each quad contains (r,s), some (u_i,v_i), and (r,v_i), (u_1,s). However, there is no saying if these $(P_1-1).(P_2-1)$ eliminated equations are linearly dependent among themselves. We will now prove that they are indeed independent. Let us take a new quad, involving say (u_2,v_2). We then have

$$\begin{aligned} z(\alpha) &= x(r)+y(s) \\ z(\theta) &= x(u_2)+y(v_2) \\ z(\lambda) &= x(r)+y(v_2) \\ z(\phi) &= x(u_2)+y(s) \end{aligned}$$

As before,

$$z(\alpha)+z(\theta)=z(\lambda)+z(\phi)$$

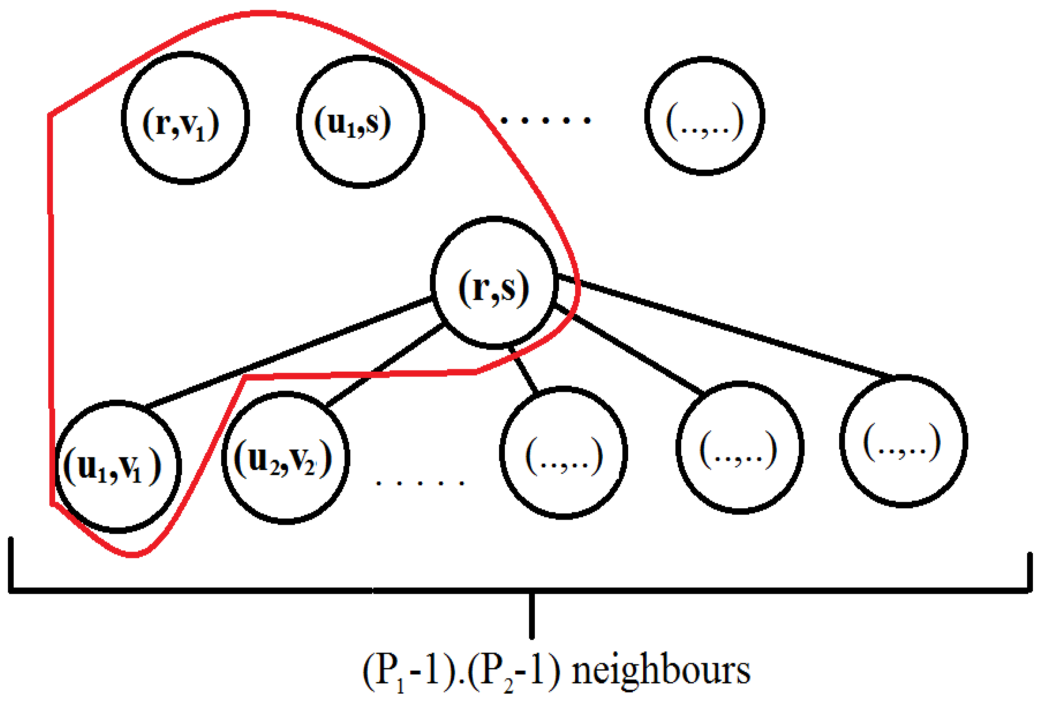


Figure 3.2: Proof of the dependency in equations

We observe that the new quad can never contain (u_1, v_1). Or indeed, no quad will contain (u_i, v_i) and (u_j, v_j), $i \neq j$. Thus, all the quads we form will be *linearly independent* as they all contain one term that is not present in the equations from other quads.
Hence, there are $(P_1 - 1.P_2 - 1)$ dependent equations, amongst the total of $P_1.P_1$. Hence the total number of linearly independent equations can be at most $P_1.P_2 - (P_1 - 1).(P_2 - 1) = P_1 + P_1 - 1$, which is 1 less than the number of unknowns. Hence, the equations cannot be solved.

Case 3: P_1 and P_2 are not co-prime, nor is one the multiple of the other.
Let $d = gcd(P_1, P_2)$, $1 < d < min(P_1, P_2)$. We know that in this case, the period of $z(n)$ is $\frac{P_1.P_2}{d}$. There are $P_1 + P_2$ unknowns, as in the above two cases.
We downsample the signal $z(n)$ by a factor of d, in such a way as follows. If $z(n) = z(1), z(2), z(3), z(4), z(5), z(6), z(z), z(8), z(9), z(10), z(11), z(12)$ with $d = 2$, we get *two* signals, $z_1(n) = z(1), z(3), z(5), z(7), z(9), z(11)$ and
$z_2(n) = z(2), z(4), z(6), z(8), z(10), z(12)$.
In general, we will get d versions of downsampled $z(n)$. Now, for each downsampled $z_i(n)$, we will find $x_i(n)$ and $y_i(n)$, each having period $P_1' = \frac{P_1}{d}$ and $P_2' = \frac{P_2}{d}$ respectively. Here, P_1' and P_2' are obviously co-prime to each other. Now, by **Case 2**, the set of equations given by $x_i(n)$, $y_i(n)$ and $z_i(n)$ are not solvable. Also, we claim that any two $x_i(n)$ and $x_j(n)$, $i \neq j$, will not contain common terms. This proof is by contradiction.
Proof: Let us assume that $x_i(n)$ was generated by starting from the i-th sample and $x_j(n)$ was generated by starting from the j-th sample of $x(n)$. That is, $x_i(n) = x(i + k.d)$ and $x_j(n) = x(j + k.d)$. Without loss of generality, we assume

$1 \leq i < j < d$. Now suppose

$$x_i(h) = x_j(g)$$

That is, $x_i(n)$ and $x_j(n)$ have a common term. Then,

$$\begin{aligned} i + k.d &= j + l.d \\ j - i &= (k - l).d \end{aligned}$$

But $(j - i)$ is a positive number less than d. That is $1 \leq (j - i) < d$. But the right hand side is a multiple of d. Hence, our assumptions was wrong and by contradiction, no $x_i(n)$ and $x_j(n)$ have common terms. Hence, each system $X_i(n) + y_i(n) = z_i(n)$, $1 \leq i < d$, is unsolvable. Thus the system of case 3 is also unsolvable.

3.1.4 The "Almost" Solution

As evident from the discussion above, we cannot decompose the two signals completely. However, if we are prepared to accept slightly modified versions of the signals $x(n)$ and $y(n)$, we can do so, provided some conditions are met. A mention is made of a possible solution in [10], however no formal algorithm or its analysis was done.

We propose a very simple algorithm by which we can "almost" decompose the input signal into its additive components, under some restrictions.

Armed with the results, we are now ready to state the following theorem:

Theorem 2: If the periods are known to be co-prime to each other, we can always find the P_1 points of $x(n)$ added with the **same** $y(i)$, where $1 \leqslant i \leqslant P_2$.
In other words, P_2 exactly offset versions of signal $x(n)$ is distributed across $z(n)$. Similarly, there are P_1 exactly offset versions of $y(n)$ spread out in $z(n)$.
The trick is now to find them.
While proving lemma 1, we found an equation that sort of mapped each $x(i)$ with exactly one value of $y(n)$. So, we already have a mapping of the various values of $x(n)$ spread across $z(n)$.
Let us take an example. Let $P_1 = 2$ and $P_2 = 5$. Also, let $x(n)$=$\{x_1, x_2\}$ and $y(n) = \{y_1, y_2, y_3, y_4, y_5\}$. In this case, the period of $z(n)$ is 10. So let us have a look at $z(n)$.

$$\begin{gathered} z(n) = ..., (x_1 + y_1), (x_2 + y_2), (x_1 + y_3), (x_2 + y_4), (x_1 + y_5), \\ (x_2 + y_1), (x_1 + y_2), (x_2 + y_3), (x_1 + y_4), (x_2 + y_5), ... \end{gathered}$$

We observe that all the samples of $y(n)$ appear to be summed up with each sample of $x(n)$. Again, we see that equation (1) gives us a way of getting the sample of $x(n)$ to which values of $y(n)$ are mapped. In this case, $z(1), z(3), z(5), z(7), z(9)$ maps $y(1), y(3), y(5), y(2)$ and $y(4)$ respectively, but offset by $x(1)$. Knowing this ordering, we can of course sort these values by a standard algorithm and solve the problem in additional $O(P_2 log P_2)$ time. However, we can do better.

3.1.5 Algorithm

We will reconstruct $y(n)$, knowing $z(n)$, P_1 and P_2. The first algorithm assumes that there is hardware support for integer division. We will reconstruct just P_2 here.

```
Result: Reconstructed signal with an offset
input $z(n)$,$P_1$,$P_2$;
Create an array $R$ of size $P_2$;
Set $i$= 1,$l$=0;
while $l < P_2$ do
    Calculate k=floor($\frac{l.P_1+i}{P_2}$);
    Calculate $r$=$l.P_1 + i - k.P_2$;
    Set $R(r)$=$z(k.P_2 + r)$;
    $i$=$i$+1;
end
```

Algorithm 1: Signal Reconstruction with Division

The reconstructed signal has an offset of k, where k is a sample of $x(n)$. If the hardware does not support fast division, we do the following.

```
Result: Reconstructed signal with an offset
input $z(n)$,$P_1$,$P_2$;
i=1,$p_1$=1,$p_2$=1;
initialize $R$ of size $P_2$
while $i \leq P_1.P_2$ do
    if $p_1$=1 then
        R($p_2$)=z(i);
    end
    if $p_1 > P_1$ then
        $p_1$=0;
    end
    if $p_2 > P_2$ then
        $p_2$=0;
    end
    i=i+1;
    $p_1$=$p_1$+1;
    $p_2$=$p_2$+1;
end
```

Algorithm 2: Signal Reconstruction

3.1.6 Complexity

We will assume that P_1 and P_2 are of the same order. The complexity of the first algorithm depends on the hardware support available. However, it is not less than of $O(P^3)$. The second algorithm is of complexity $O(P^2)$.

3.1.7 Constraints

However, there are a few conditions that have to be met for this algorithm to work.
(1) The periods must be relatively co-prime to each other.
(2) The total length of the input signal $z(n)$ must be at least the LCM of the individual period lengths. In view of **(1)**, this is equal to the length of the individual periods.

3.1.8 More than two signals?

Another point that can be mentioned is that this method can be used to separate any number of additive signals, provided the two conditions above are met. If there are a signals, having periods P_1, P_2, ..., P_a, we can use the same algorithm as above, with a slight modification. Suppose we want to extract the signal corresponding to period P_i. We treat P_i as P_2 in the procedure and the sum of all other signals as P_1'. This is because here, $P_1' = \sum_{\substack{b=1 \\ b\neq i}}^{a} P_b$, a signal of period $\frac{\prod_{b=1}^{a} P_b}{P_i}$. Thus, P_i can be easily separated, but the offset will be different, or more precisely the sum of a single sample of the other $(a-1)$ signals.

3.1.9 Noise Analysis

In terms of noise analysis, the algorithm is bound to give poor results, as the reconstructed signals are directly mapped from the output, without any other processing. However, if we assume that the noise corrupting the initial signal period is zero mean noise, we can improve the output in case of noisy signals as well.

In our *reconstruct* procedure, we were taking only a single value of i. However, if we were to take **all** possible values of i and carry out the procedure, we would have got P_1 signals. The average of these will give the reconstructed signal $y(n)$ without any effect of the initial noise in $x()$.
Proof: Let each $x(i) = x'(i) + b(i)$, where $x'(n)$ is the uncorrupted signal and $b(n)$ is a zero mean noise. By the modified **procedure** *reconstruct*, we will get P_1 copies of $y(n)$, each added with a single $x(i)$. Now, adding these P_1 signals and taking the average, we get

$$Y(n) = \frac{\sum_{i=1}^{P_1} y(n) + x(i)}{P_1} + \frac{\sum_{i=1}^{P_1} b(i)}{P_1}$$

Since $b(n)$ is a zero mean noise, $Y(n)$ is independent of the noise component of $x(n)$. The noise component of $y(n)$ cannot be removed by this process.

In summary, we propose a simple, fast algorithm to decompose $z(n)$ into its components, albeit they are reconstructed with some non-zero offset. The algorithm provides conditional noise removal for zero mean noise, but at the cost of higher complexity. If the periods extracted by the **Nested Periodic Matrices** happen to be relatively prime and we have sufficient length of signal available, we can reconstruct the signals very fast, with an offset.
This method overcomes the major hurdle of the **Singular Value Decomposition**

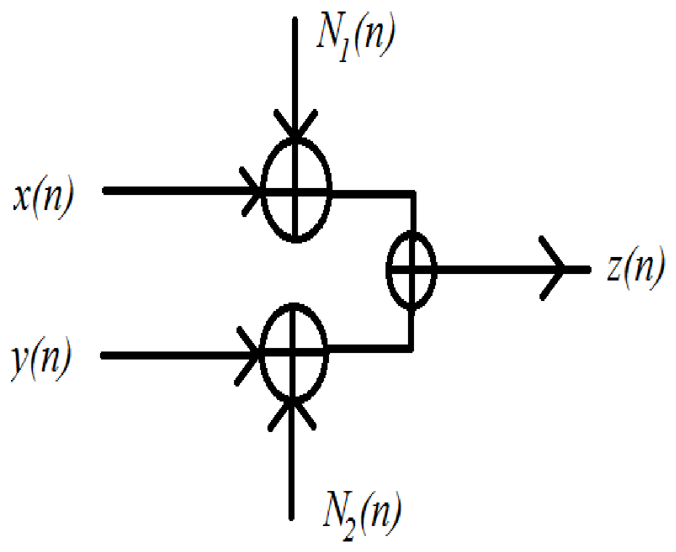


Figure 3.3: The noise model assumed. N_1 and N_2 are white noises.

based method[4], which requires that one of the signals be dominant over the other. In our method, there is no such condition.

3.2 Multiplication

3.2.1 Preliminaries

Similar to addition, we may encounter signals which have undergone point by point multiplication. The analysis of this section is very similar to the previous section, barring a few subtle instances. So we omit the parts that are common to both the analyses.
Let us take 2 non-periodic signals, $x(n)$ and $y(n)$, each of length N and define $z(n)$ to be

$$z(i) = x(i).y(i)$$

for $1 \leq i \leq N$. Now, we have as before $2N$ unknowns in $x(n)$ and $y(n)$ and only N values of $z(n)$, giving us N equations, which are all linearly independent. However, the number of unknowns exceed the number of solutions and "exact" separation of the signals cannot be achieved.

3.2.2 The Periodic Case

We assume $x(n)$, $y(n)$, P_1 and P_2 are as defined in the previous section. We can show as before that an "exact" solution can never be attained.

3.2.3 A Possible Solution?

The usual approach will not work. That is, even in the case of multiplication, it can be shown that whatever be the relation (eg. multiple, co-primality, or otherwise) between P_1 and P_2, we can never exactly solve the equations and separate the unknowns. The proof can be done in a way similar to the previous section. However, intuitively, the proof can be shown by the following way.
Given a set of equations of the form

$$x(i).y(j) = z(k)$$

we can (assuming that none of them are zero, or negative) take the logarithm of both the sides of each equation and *reduce* the proof to the proof of the addition case. That is, we instead have,

$$\log(x(i)) + \log(y(j)) = \log(z(k))$$

Hence, the proof reduces to the previous case, assuming none of the terms is zero, or negative. The formal proof can of course be done as in the additive case.

3.2.4 The "Almost" Solution

This algorithm is almost the same as above. The variation is due to the fact is that whereas in the addition case, we were getting the reconstructed signal $x_r(n)$ as

$$x_r(n) = k + x(n)$$

where k was a sample of $y(n)$, in the multiplication case, we will get

$$x_r(n) = k.x(n)$$

where k is some sample of $y(n)$. Here, there will be 2 things possible.
1. $k = 0$. Here, the reconstructed signal will be all zeros. To bypass this problem, we will simply scan through all samples of $y(n)$ till we get a $k \neq 0$. The complexity increases in this case. One thing to note is that $\exists q$, for which $y(q) \neq 0$, as otherwise, $y(n)$ would not be periodic
2. Samples of $y(n)$ are very large as compared to samples of $x(n)$, i.e., $x(i) \ll y(j) \forall i, j$. In this case, the reconstructed signal will be a scaled by a large factor. If such is not acceptable, we can offer a "weaker" alternative. We call it weaker because it introduces two more conditions for it to work. However, in this case, we will have *exact* reconstruction.

3.2.5 Algorithm

The algorithm is the same as the addition, except for the "zero-checking" case. So we will not repeat that here, However, for the second case in the previous subsection, we will write the conditions.
Condition 1: All the samples must be integers.
Condition 2: The greatest common divisors of the signals $x(n)$ and $y(n)$ individually, must be 1.

The reasons for imposing these two conditions are as follows. We intend to *divide* the samples of the reconstructed signal $x_r(n)$ by the greatest common divisor is itself. So, if the *maximum* factor common to *each* of the samples of $x_r(n)$ is k, where $k \neq 0$ is a sample of $y(n)$, we divide $x_r(n)$ by k, so that the final reconstructed signal $x_f(n)$ is

$$\begin{aligned} x_f(n) &= \frac{x_r(n)}{k} \\ &= \frac{k.x(n)}{k} \\ &= x(n) \end{aligned}$$

Thus, we get **exact** reconstruction. The samples must be **integers** because the concept of greatest common divisors does not exist for fractions.

3.2.6 Complexity

The complexity of this algorithm is higher than its additive counterpart. This is because, we need an additional $O(P)$ time to find a non-zero k, as defined above. So, the complexity of the non-exact case is $O(P^3)$. For the exact case, we will need an additional $O(\log(P))$ time for the Euclidean algorithm. However, we will have no effect on the final complexity as the initial complexity is itself greater.

Chapter 4

Experiments and Results

We will be dealing with pseudorandom signals, generated by software. However, for the rest of the chapter, we will only be referring to them as *random* signals, interchangeably.

4.1 Addition

4.1.1 Two signals

In the Fig. 4.1, we have taken two pseudorandomly generated signals x and y of period 7 and 17 and repeated each one so that the length of each becomes 119. The signals are then added to form the signal in the 3rd graph. We carry out the reconstruct algorithm and the 4th plot shows the reconstructed x' and the 5th shows the reconstructed y'.

4.1.2 Three signals

We have taken 3 randomly generated signals of period 7,9 and 11 respectively, and extended each upto 693 samples by repetition. Then we have created the added signal adding the 3 signals thus got. We then run the "modified" algorithm to reconstruct the signals. The simulation is shown in Fig. 4.2.

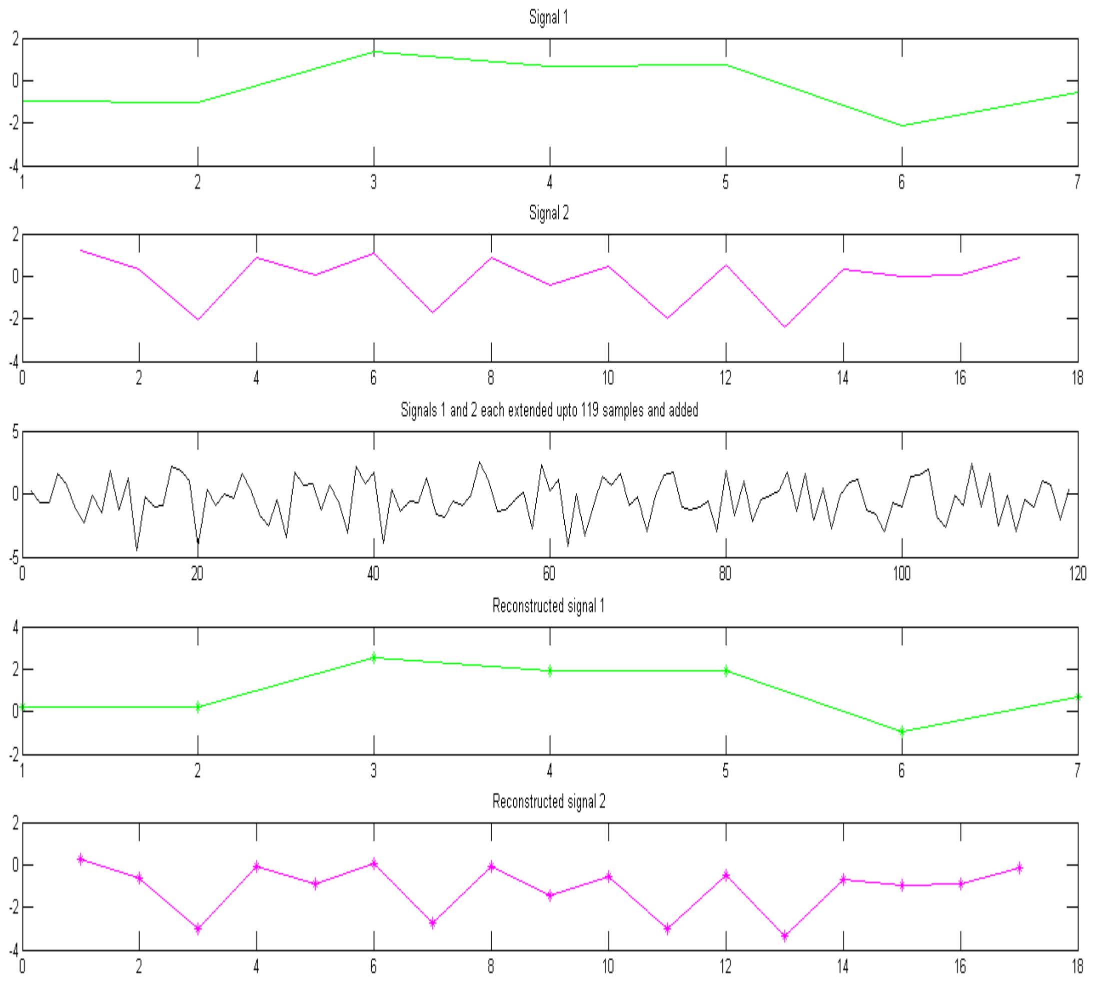


Figure 4.1: Reconstructing 2 signals in additive case

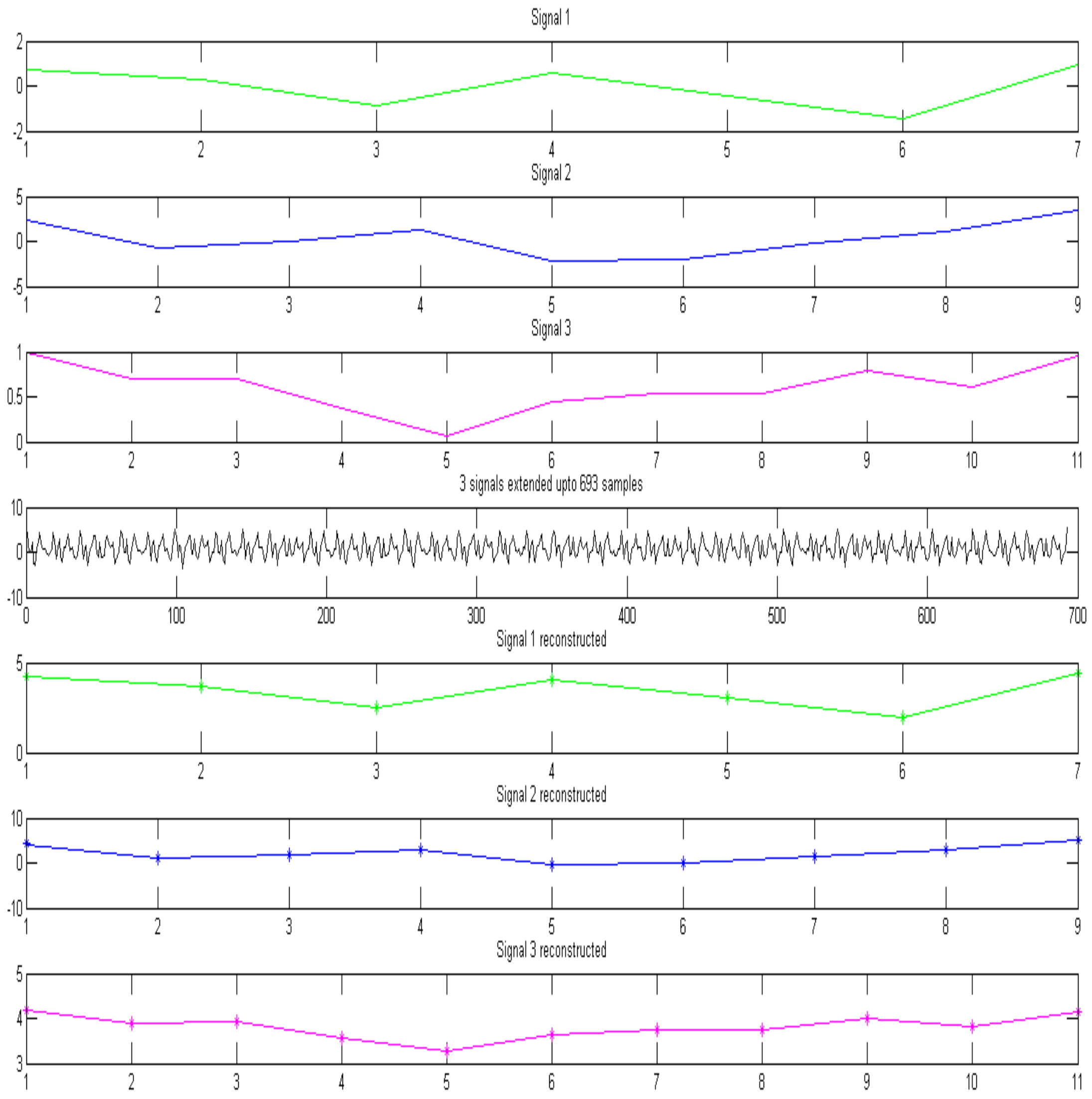


Figure 4.2: Reconstructing 3 signals

4.1.3 Noise Performance

In Fig. 4.3, we show the results of testing our algorithm by taking two random signals, of period lengths 12 and 65 respectively. We have added **20 dB** *white Gaussian noise* to each, separately. Then we have extended each signal to 780 samples each and added them. Then we have run our reconstruction algorithm as before. We show the errors in the two reconstructions.

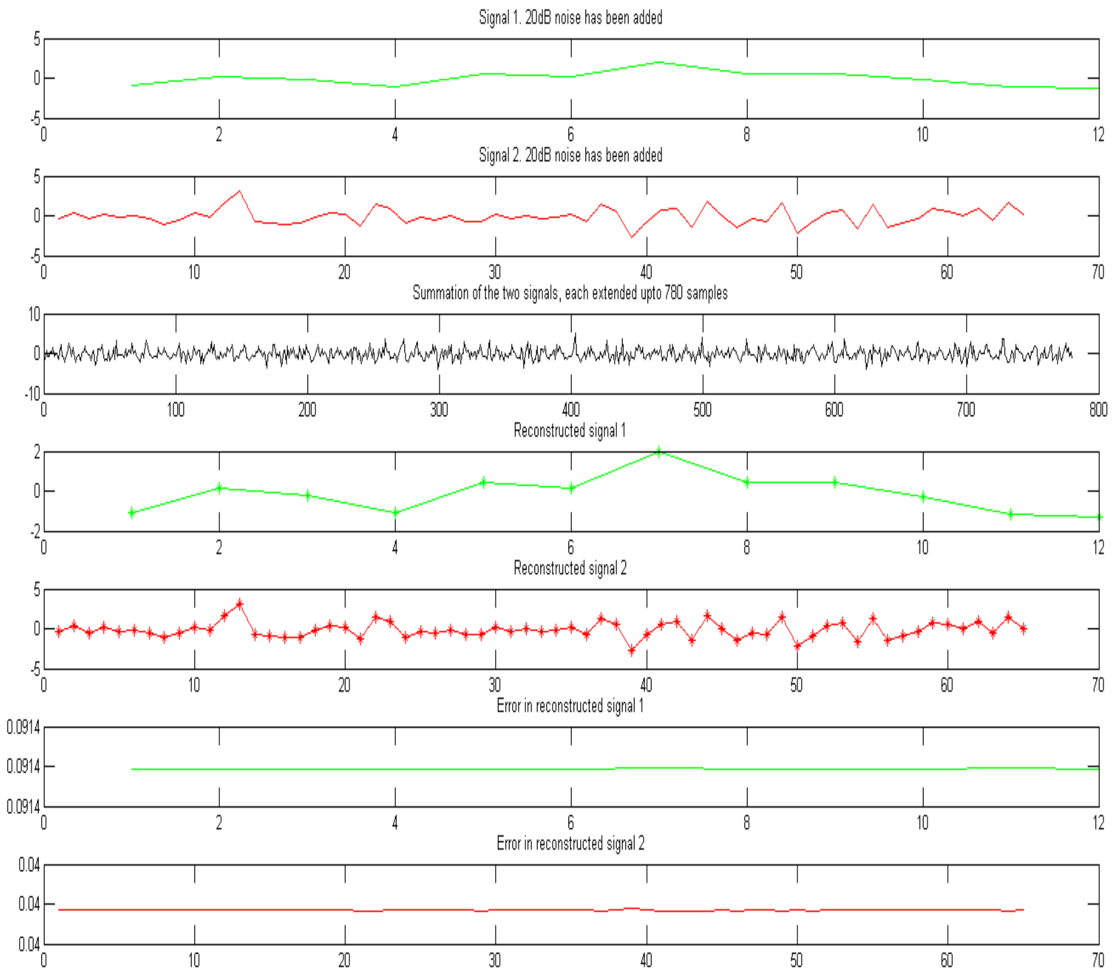


Figure 4.3: Noise analysis in the additive case

4.1.4 Comparison with SVD based method

We pit our method against the SVD based method on the 2 signal case, generating the signals as before. The results are visually similar, but the SVD method of course gives better results. However, as pointed out in [3], the SVD algorithm takes at least $O(m^n+n^3)$ time, where m and n are he dimensions of the matrix involved. Assuming that our signals have periods in $O(P)$, the time taken will be $O(P^3)$. However, our method takes only $O(P^2)$ time. The simulation is shown in Fig. 4.4.

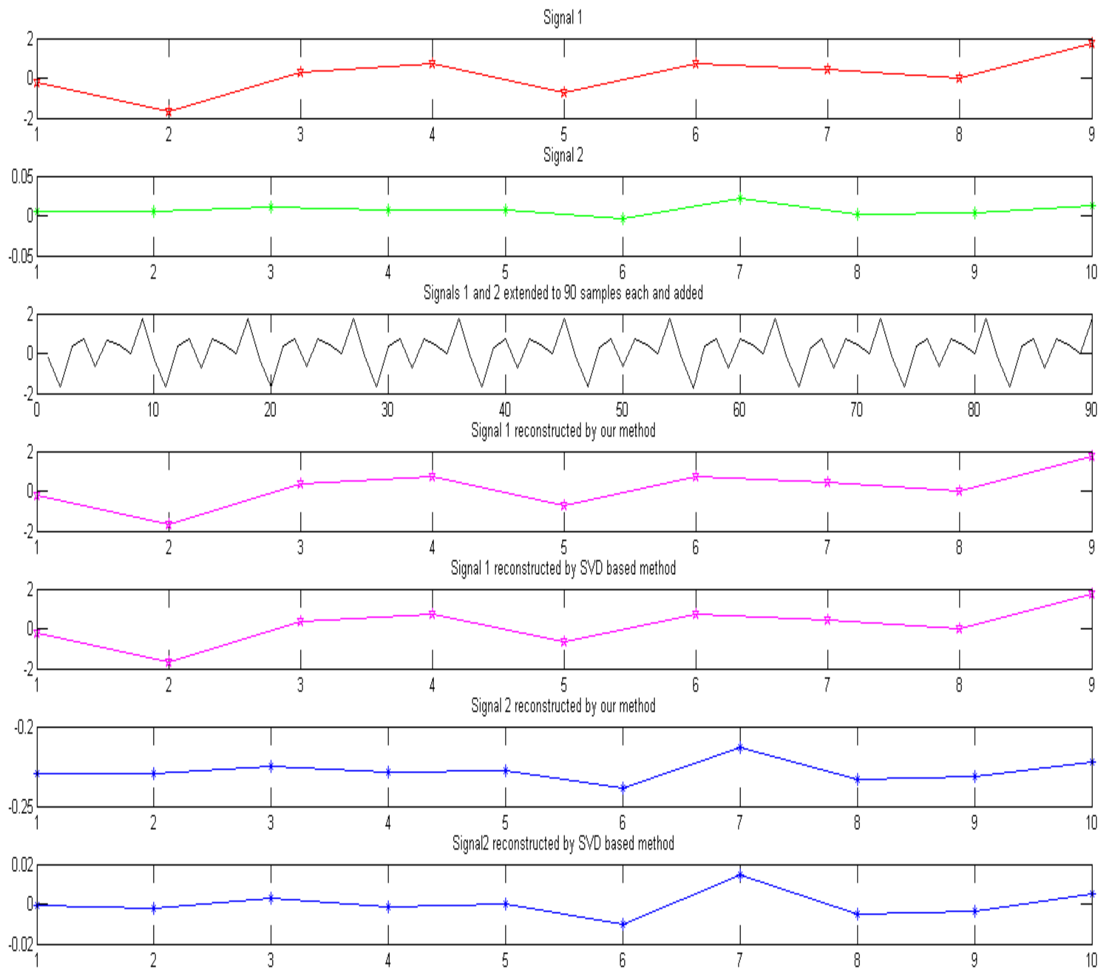


Figure 4.4: Comparison between our method and SVD for the additive 2 signal case.

4.2 Multiplication

4.2.1 Two signals

We take two randomly generated signals of period lengths 11 and 15, and as in the additive case, we have multiplied them together. In Fig. 4.5, we have the initial signals, the multiplied signal which acts as our input, and the two separated signals.

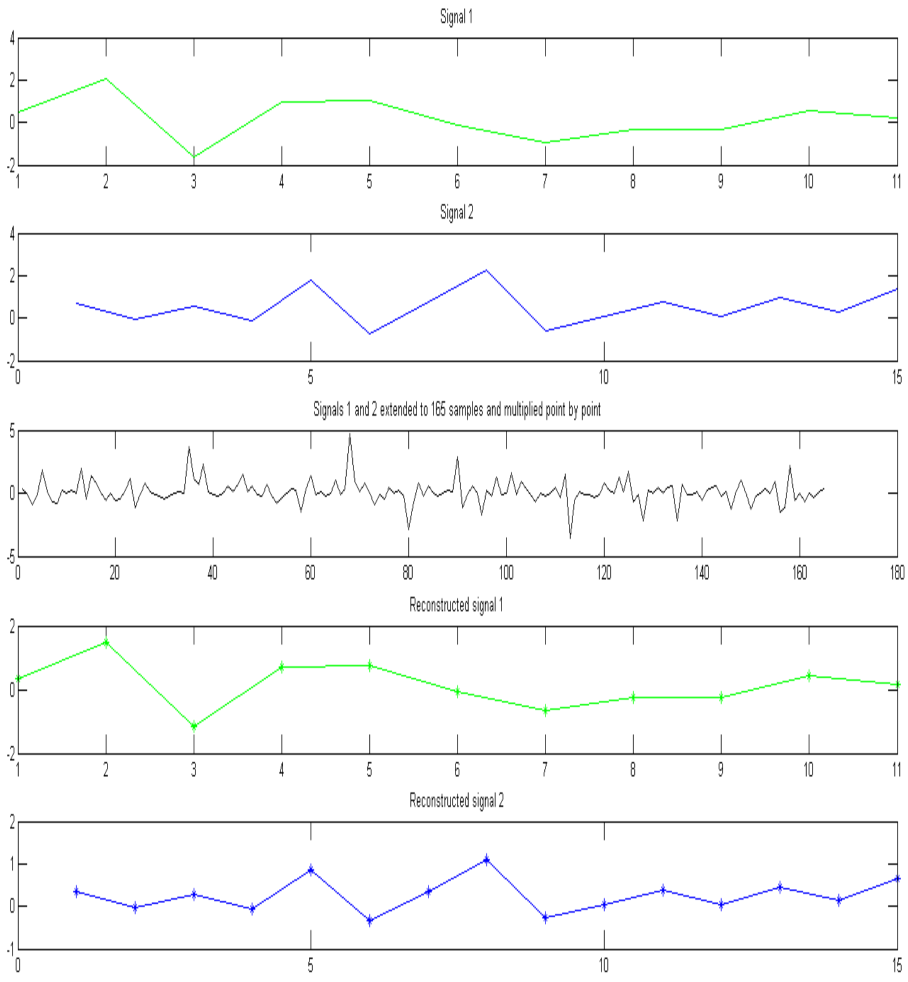


Figure 4.5: Separation of 2 point by point multiplied signals.

4.2.2 Three signals

We create 3 signals of periods 5,8 and 11 respectively and as before, extend each upto 440 samples by repetition. We then multiply them point by point and run our algorithm on it. The results are shown in Fig. 4.6.

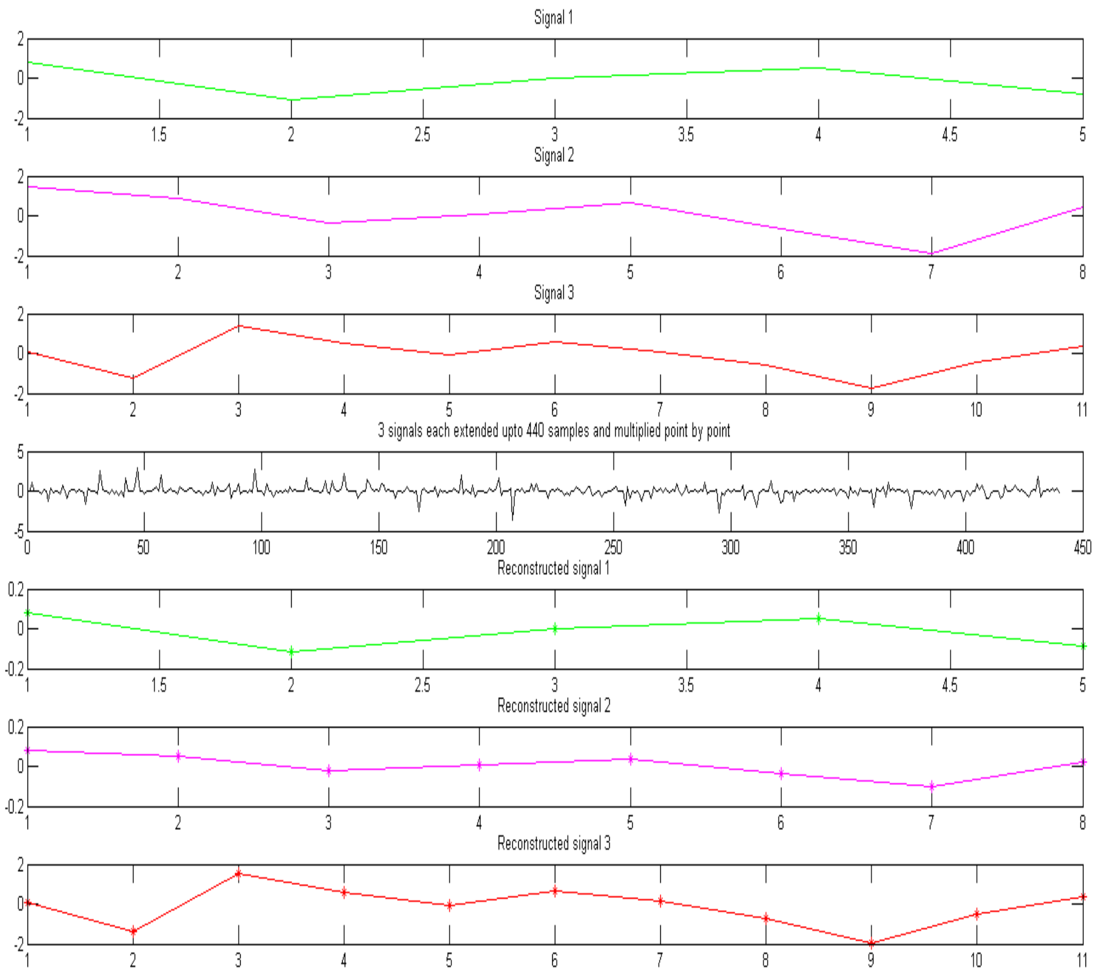


Figure 4.6: Separation of 3 point by point multiplied signals.

4.2.3 Exact reconstruction

As pointed out previously, we will have **two** strict conditions for the algorithm to give **exact** reconstruction. In Fig. 4.7, we take two signals of periods 9 and 10 respectively, such that all the samples are integers between 1 and 100, and the GCD of samples in the individual signals is 1. Then we run the modified reconstruction algorithm.

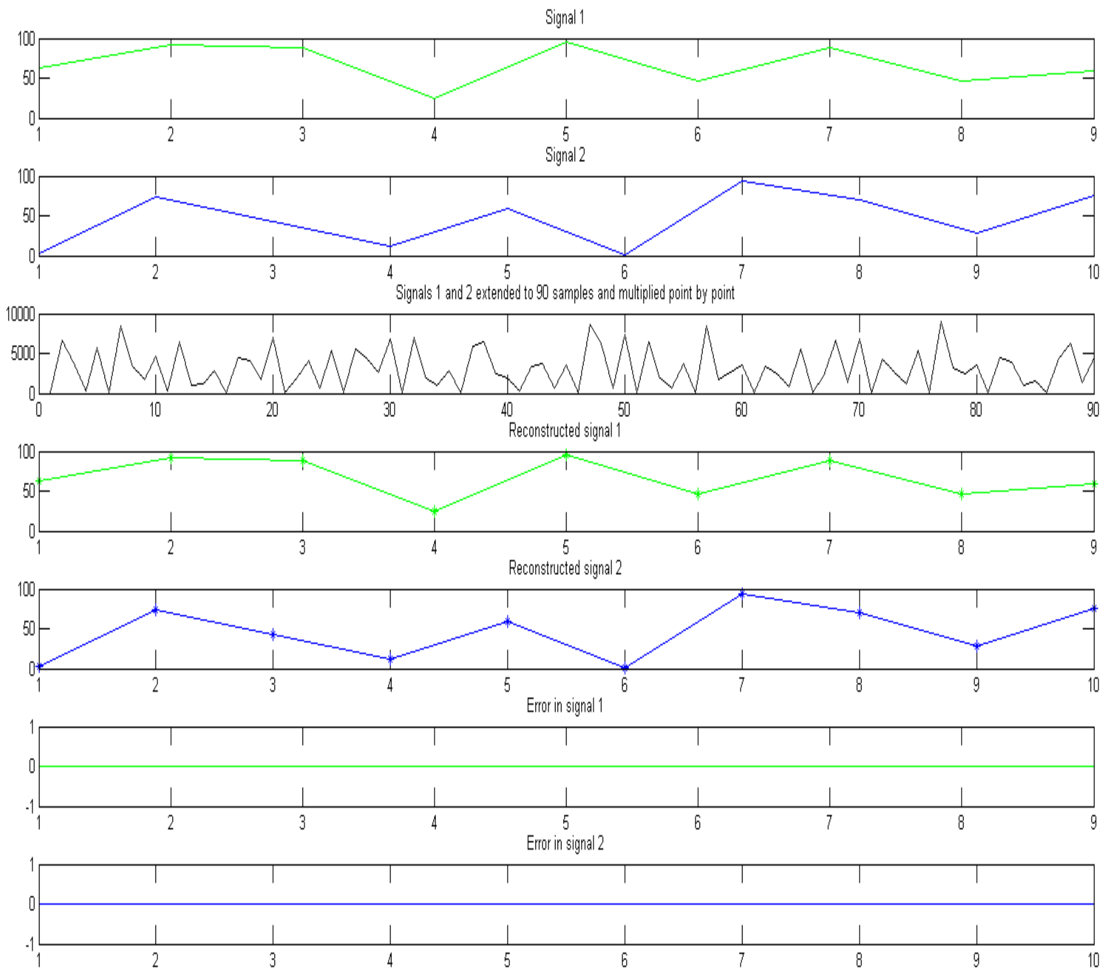


Figure 4.7: Exact reconstruction in the point by point multiplicative case.

Comparison

To the best of our knowledge, no algorithm has been published to separate periodic signals which have undergone point by point multiplication. Thus we have been unable to compare the proposed algorithm with any other.

Table 4.1: Comparison between our algorithm and the SVD based approach

Property	SVD based method (1997)	Our Method (2017)
Possible periodicities	All, except when $P_2 = kP_1$	Only co-prime periodicities
Noise Performance	Good	Conditional
Complexity	$O(P^3)$	$O(P^2)$
Multiplicative reconstruction	No	Yes

Chapter 5

Conclusion and Scope for Future Work

We examine and compare two period estimation algorithms against the **Ramanijan sums** based method. We find that the Ramanujan sums overcome some of the main barriers of the older methods. However, there happens to be instances where this method is also weak.

Our main contribution in this thesis is the reconstruction of signals with known periods which have undergone any of two "mixing" operations, namely **addition** and **multiplication**. We show that we can **never** fully reconstruct the said signals even after knowing the periods. However, (i) if the periods are mutually co-prime and (ii) the "mixed" signal is **at least** of the length of the individual periods (in this case, the number being the product of the period lengths, given the previous condition), we can get back the original signals with nominal changes. For the additive case, the original signal is determined but with an *offset* and in the case of multiplication, the input signal is found to be *scaled*. This method can be applied to as many signals as we like (limited only by hardware constraints) provided the two conditions are met. The process provides some noise performance if the signals are corrupted with *zero mean* noise *before* being repeated and "mixed". This algorithm with some modification and two more strict conditions is able to recover the signals which have undergone point by point multiplication. Finally, if the periods are *not* mutually co-prime, we can get back the original signal, but a *downsampled* version, with loss in information.

Further studies are required to pit this algorithm against other algorithms when it comes to real world data. Possible relaxation of the strict conditions of the algorithm will have to be studied. Also, reasons why Ramanujan sum based method fails for some real world data need to be analysed.

Chapter 6

Bibliography

[1] Alan V. Oppenheim, Ronald W. Schafer, John R. Buck. *Discrete-Time Signal Processing.* Prentice Hall, 2005.

[2] Amir B. Geva, Dan H. Kerem. Forecasting generalized epileptic seizures from the eeg signal by wavelet analysis and dynamic unsupervised fuzzy clustering. *IEEE Transactions on Biomedical Engineering*, 45(10):1205–1216, • 1998.

[3] Gene H. Golub, Charles F. Van Loan. *Matrix Computations.* The Johns Hopkons University Press, 1996.

[4] P.P. Kanjilal , S. Palit. Fetal ecg extraction from single-channel maternal ecg using singular value decomposition. *IEEE Transactions on Biomedical Engineering*, 44(1):51–59, January 1997.

[5] S. Ramanujan. On certain trigonometrical sums and their applications in the theory of numbers. *Trans. Cambridge Philosoph. Soc.*, XXII(13):259–276, 1918.

[6] Li Tan. *Digital Signal Processing.* Academic Press, 2012.

[7] S. V. Tenneti and P.P. Vaidyanathan. Nested periodic matrices and dictionaries: New signal representations for period estimation. *IEEE Transactions on Signal Processing*, 63(14):3736–3750, 2015.

[8] P. P. Vaidyanathan. Ramanujan sums in the context of signal processing 2014;part ii: Fir representations and applications. *IEEE Transactions on Signal Processing*, 62(16):4158–4172, Aug 2014.

[9] P.P. Vaidyanathan. Ramanujan sums in the context of signal processing- part i: Fundamentals. *IEEE Transactions on Signal Processing*, 62(16):4145–4157, 2014.

[10] Zou Mouyan, R. Unbehauen, Chai Zhenming. Separation of periodic signals by using an algebraic method. *Proc. Intl. Symp. on Circuits and Systems*, 5:2427–2430, 1991.