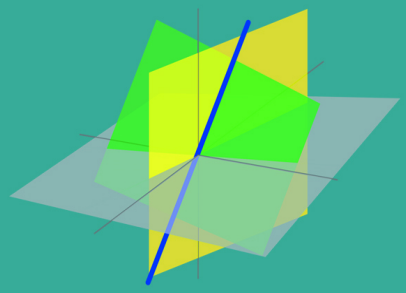

LINEAR ALGEBRA

for

SIGNAL PROCESSING - I

Soman K.P
Sachin Kumar S
Prabaharan P
Neethu Mohan

Centre for Computational Engieering and Networking
Amrita Vishwa Vidyapeetham, Coimbatore

Contents

Chapter 1

Introduction

Democratizing the signal processing with two mathematical tools/concepts taught in high school. These are the concept of resolution of vectors in orthogonal directions and the concept of least-square. If we want to make everything smart, clean and green; the areas that is going to be vital are signal processing, information processing and pattern classification (statistical and or deep learning). These are also the foundation for burgeoning Data-Science. The computational foundation for all these is linear algebra. This implies that if do not give strong emphasis to this topic in our undergraduate curricula, education will become lopsided. In this article we are showing how linear algebra/matrix algebra course can be enhanced by incorporating straight forward applications from various engineering disciplines. At first we will show how the concept of orthogonality in linear algebra (also vector algebra) is extended to give strong foundation for signal processing and classification. It can be taught in the first year of engineering itself. Imagine how much a student will be excited if he is able to process his own vital signals and signals from the surroundings (audio, video, mechanical vibrations etc) using the mathematics he is learning. We took signal processing as our

first topic because,

- This is required for all engineering, though at present the subject signal processing is mainly taught to electrical sciences. This situation should change.
- Many consider this as one of the toughest subject to master.

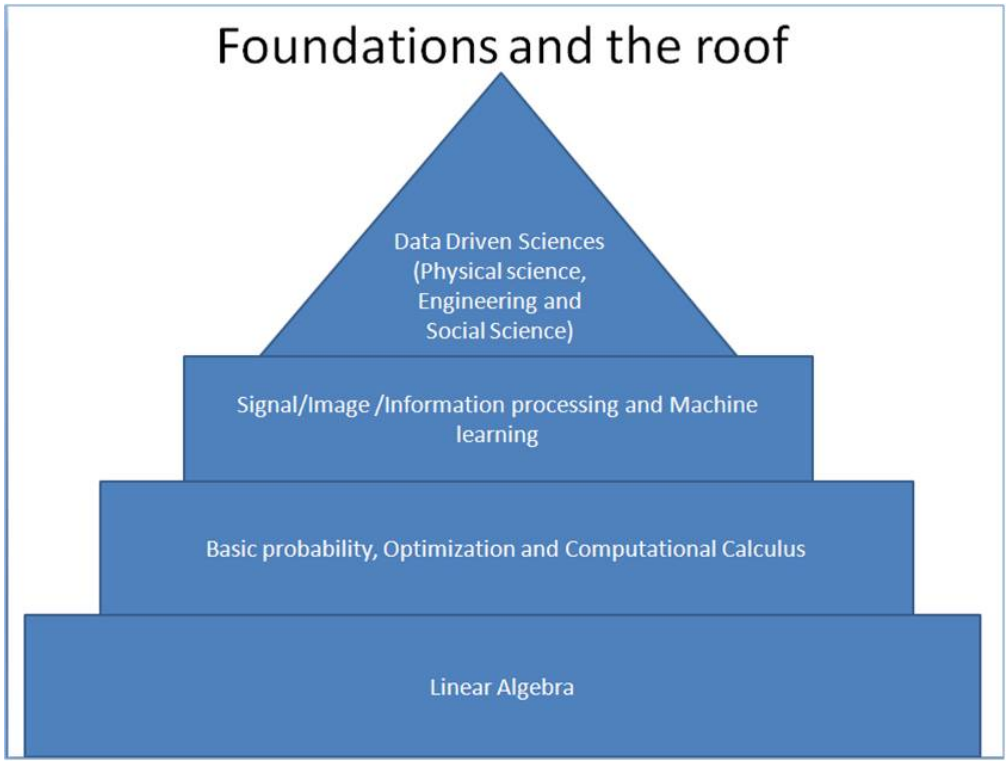

Figure 1.1: Foundation and roof of signal processing

There are several reason for this feeling,

- A system point view and the associated terms like linearity, time invariance, linear time invariance, convolution etc is introduced at the beginning. Though this

is important, it need not be taught at the beginning. These concepts and associated exercise problems turn away the students from the core concepts of signal processing. At least digital signal representation can be understood even without any of those concepts. What is more important is digital signal representation using orthogonal bases and signal content modification.

- A complex exponential base is used to represent a signal. Most books (and also teachers) are silent about why at all one should use complex quantities to analyze and manipulate a real signal. All signals are real in nature. The appearance of associated totally confuses even a smart student in electrical and communication engineering. Complex numbers, in many books, appears out of the blue and stay forever in the remaining part of the text leaving the student trembling throughout the course. Instead, representation using real bases like DCT, Walsh Hadamard, Haar etc should be introduced first.

- No geometrical and visual interpretation is given to the DFT formula. Most students don't know that the formula represent an inner product. The concept of orthogonal basis is totally alien to undergraduate students.

- DTFT and Z-transform is introduced as a formula in a definition without telling its need in digital signal processing.

In high school every kid is learning about resolution of forces. Forces in a plane are resolved into two orthogonal directions, along x and y directions. This is how it is,

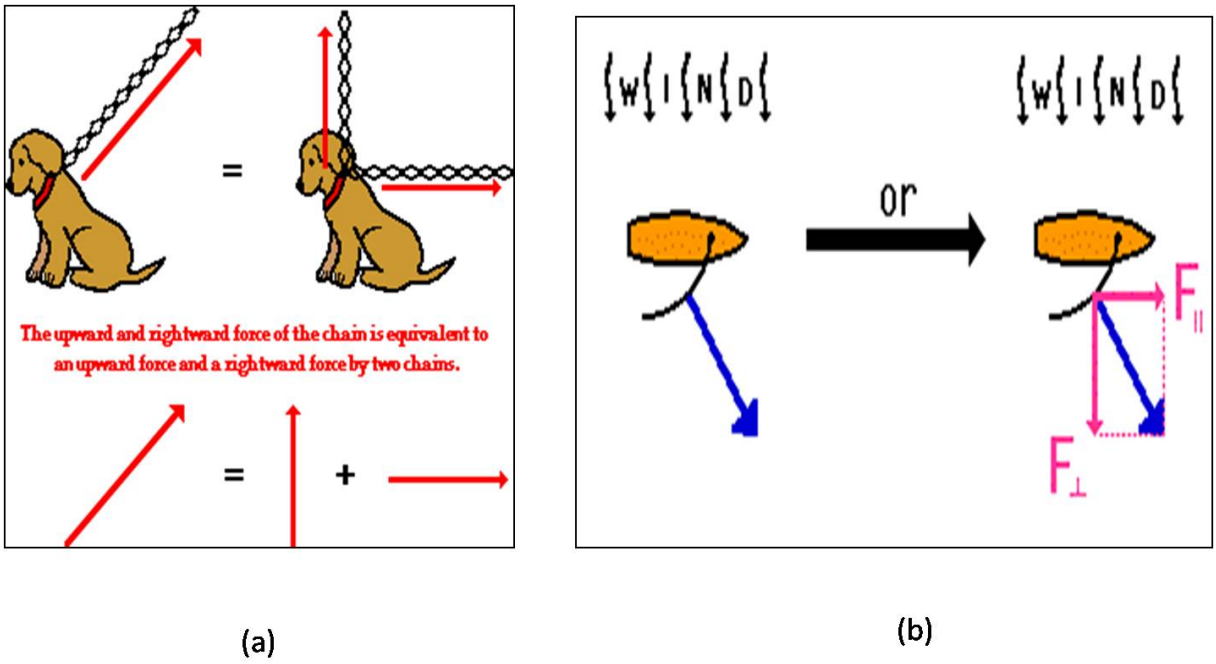

Figure 1.2: Illustration of force

The force can be represented as

$$F = F_x + F_y$$

This can be represented as

$$F = \langle F, base_1 \rangle \, base_1 + \langle F, base_2 \rangle \, base_2$$

Inorder to understand this with an example, consider a point in 2-dimension space (3,4). This point (or vector in 2-dimension) can be represented using two trivial basis $\begin{bmatrix} 1 & 0 \end{bmatrix}^T$ and $\begin{bmatrix} 0 & 1 \end{bmatrix}^T$ as,

$$\begin{bmatrix} 3 \\ 4 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 4 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Analysis of a signal (a vector in n-dimension space) can be done by changing the basis. Consider another sets of basis,

$$\left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \end{bmatrix} \right\}$$

In normalized form as,

$$\left\{ \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}, \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix} \right\}$$

The vector (3,4) can be represented as,

$$\begin{bmatrix} 3 \\ 4 \end{bmatrix} = a \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} + b \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix}$$

We show that most of the transforms appearing in signal processing is basically this resolution operation extend to a vector in n-dimensional space. The concepts requited from linear algebra are,

- Linear combination
- Inner product
- Orthogonality of vectors
- Vector space
- Orthogonal basis for vector space

- Change of basis

From linear algebra point of view, an n-tuple data vector $x \in R^n$ can be represented using a given orthonormal basis set as $\{base_1, base_2, base_3, ..., base_n\}$ as,

$$x = (x^T base_1)base_1 + (x^T base_2)base_2 + (x^T base_3)base_3 + ... + (x^T base_n)base_n$$

In the above expression, $x^T base_i$ is a scalar quantity.

Question now is how to create useful orthogonal bases. Usefulness is decided according to,

- Qualitative interpretation of the signal (signal content in terms of frequencies)
- Sparse representation of the signal for storing and transmission.
- Separating signal into useful components (need not be in terms of frequency components)

1.1 Concept and creation of orthogonal bases

A continuous signal is sampled at uniform rate/interval. This ordered array is a vector in n-dimensional vector space where n is the length of the signal (number of sampled values). Its basis represented by columns of matrix is identity matrix of size $n \times n$. We can represent (or resolve) the vector using another set of basis (another set of n unit orthogonal vectors). Let the basis vectors be real and form columns of a matrix W. Then $X = W^T x$ represents the transform coefficients. Let,$w_1, w_2, w_3, ...w_n$ be the columns of W. Then, $X(k) = W_k^T x$. That is, $X(k)$ is the inner product of x with k^{th}

basis vector in W. If the inner product is zero, then the vectors are orthogonal to each other.

1.1.1 Sampling full sine and cosine functions

We can create discrete orthogonal bases by sampling orthogonal bases in continuous domain. One such set of bases is one that emerged from famous Fourier series. Fourier series is about expressing signals in terms of sine and cosine harmonic functions. Readers may refer Calculus book for more details. Consider a 1-D domain with duration T and a single cosine wave that exactly fit the distance T. The frequency associated with such a wave is 1/T. We call this wave as fundamental wave and the frequency as fundamental frequency. Consider sine waves and cosine waves of frequencies which are integer multiple of fundamental frequency 1/T. We can show that all these waves are orthogonal in the range 0 to T as per the following definition of orthogonality. Two functions $f_1(t)$,and $f_2(t)$ are orthogonal in the range T if,

$$\int_0^T f_1(t) f_2(t) dt = 0$$

It can be easily shown that the proposed harmonic functions satisfy above requirement. Now, suppose we want 4 bases for representing a vector/signal in R^4. Since we are not specifying T, we develop a methodology which is independent of T. We know our fundamental should make one full wave in time T, it is equal to angular movement of 2π. So, we sample the range 0 to 2π at intervals $^{2\pi}/_{4}$ (in general,$^{2\pi}/_{N}$ for N-tuple data). Let, this be vector θ,

$$\hat{\theta} = \begin{pmatrix} 0 & 1 \cdot \frac{2\pi}{4} & 2 \cdot \frac{2\pi}{4} & 3 \cdot \frac{2\pi}{4} \end{pmatrix} = \begin{pmatrix} 0 & \frac{\pi}{2} & \pi & \frac{3\pi}{2} \end{pmatrix}$$

Then the vectors, $\cos(0 \cdot \hat{\theta}), \cos(1 \cdot \hat{\theta}), \cos(2 \cdot \hat{\theta}), \sin(1 \cdot \hat{\theta})$ form the basis for R^4. That is, $\cos(\hat{\theta}) = \begin{pmatrix} \cos(0) & \cos(\frac{\pi}{2}) & \cos(\pi) & \cos(\frac{3\pi}{2}) \end{pmatrix}$. Therefore, the orthogonal bases (row wise) are,

$$\begin{aligned}
\cos(0 \cdot \hat{\theta}) &= \begin{pmatrix} \cos(0) & \cos(0) & \cos(0) & \cos(0) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \\
\cos(1 \cdot \hat{\theta}) &= \begin{pmatrix} \cos(0) & \cos(\frac{\pi}{2}) & \cos(\pi) & \cos(\frac{3\pi}{2}) \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 & 0 \end{pmatrix} \\
\cos(2 \cdot \hat{\theta}) &= \begin{pmatrix} \cos(0) & \cos(\pi) & \cos(2\pi) & \cos(3\pi) \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 & -1 \end{pmatrix} \\
\sin(1 \cdot \hat{\theta}) &= \begin{pmatrix} \sin(0) & \sin(\frac{\pi}{2}) & \sin(\pi) & \sin(\frac{3\pi}{2}) \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & -1 \end{pmatrix}
\end{aligned}$$

Now we can normalize (make the norm to unity) by dividing each vector by its norm. Then the orthonormal transform matrix (transpose of basis matrix) is created by writing each normalized vector as the **column of the matrix** as follows.

$$\begin{bmatrix} \frac{1}{2} & \frac{1}{\sqrt{2}} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{-1}{2} & \frac{1}{\sqrt{2}} \\ \frac{1}{2} & \frac{-1}{\sqrt{2}} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{=1}{2} & \frac{-1}{\sqrt{2}} \end{bmatrix}$$

Now consider the case of 8-tuple data. The vector θ is,

$$\hat{\theta} = \begin{pmatrix} 0 & 1 \cdot \frac{2\pi}{4} & 2 \cdot \frac{2\pi}{4} & 3 \cdot \frac{2\pi}{4} & 4 \cdot \frac{2\pi}{4} & 5 \cdot \frac{2\pi}{4} & 6 \cdot \frac{2\pi}{4} & 7 \cdot \frac{2\pi}{4} \end{pmatrix} = \begin{pmatrix} 0 & \frac{\pi}{4} & \frac{\pi}{2} & \frac{3\pi}{4} & \pi & \frac{5\pi}{4} & \frac{3\pi}{2} & \frac{7\pi}{4} \end{pmatrix}$$

Now the 8 bases can be created as follows,

$$\cos(0 \cdot \hat{\theta}), \cos(1 \cdot \hat{\theta}), \cos(2 \cdot \hat{\theta}), \cos(3 \cdot \hat{\theta}), \cos(4 \cdot \hat{\theta}), \sin(1 \cdot \hat{\theta}), \sin(2 \cdot \hat{\theta}), \sin(3 \cdot \hat{\theta})$$

$$\begin{aligned}
\cos(0 \cdot \hat{\theta}) &= \begin{pmatrix} \cos(0) & \cos(0) & \cos(0) & \cos(0) & \cos(0) & \cos(0) & \cos(0) & \cos(0) \end{pmatrix} \\
\cos(1 \cdot \hat{\theta}) &= \begin{pmatrix} \cos(0) & \cos(\frac{\pi}{4}) & \cos(\frac{\pi}{2}) & \cos(\frac{3\pi}{4}) & \cos(\pi) & \cos(\frac{5\pi}{4}) & \cos(\frac{3\pi}{2}) & \cos(\frac{3\pi}{4}) \end{pmatrix} \\
\cos(2 \cdot \hat{\theta}) &= \begin{pmatrix} \cos(0) & \cos(\frac{\pi}{2}) & \cos(\pi) & \cos(\frac{3\pi}{2}) & \cos(2\pi) & \cos(\frac{5\pi}{2}) & \cos(3\pi) & \cos(\frac{7\pi}{2}) \end{pmatrix} \\
\cos(3 \cdot \hat{\theta}) &= \begin{pmatrix} \cos(0) & \cos(\frac{3\pi}{4}) & \cos(\frac{3\pi}{2}) & \cos(\frac{9\pi}{4}) & \cos(3\pi) & \cos(\frac{15\pi}{4}) & \cos(\frac{9\pi}{2}) & \cos(\frac{21\pi}{4}) \end{pmatrix} \\
\cos(4 \cdot \hat{\theta}) &= \begin{pmatrix} \cos(0) & \cos(\pi) & \cos(2\pi) & \cos(3\pi) & \cos(4\pi) & \cos(5\pi) & \cos(6\pi) & \cos(7\pi) \end{pmatrix} \\
\sin(1 \cdot \hat{\theta}) &= \begin{pmatrix} \sin(0) & \sin(\frac{\pi}{4}) & \sin(\frac{\pi}{2}) & \sin(\frac{3\pi}{4}) & \sin(\pi) & \sin(\frac{5\pi}{4}) & \sin(\frac{3\pi}{2}) & \sin(\frac{7\pi}{4}) \end{pmatrix} \\
\sin(2 \cdot \hat{\theta}) &= \begin{pmatrix} \sin(0) & \sin(\frac{\pi}{2}) & \sin(\pi) & \sin(\frac{3\pi}{2}) & \sin(2\pi) & \sin(\frac{5\pi}{2}) & \sin(3\pi) & \sin(\frac{7\pi}{2}) \end{pmatrix} \\
\sin(3 \cdot \hat{\theta}) &= \begin{pmatrix} \sin(0) & \sin(\frac{3\pi}{4}) & \sin(\frac{3\pi}{2}) & \sin(\frac{9\pi}{4}) & \sin(3\pi) & \sin(\frac{15\pi}{4}) & \sin(\frac{9\pi}{2}) & \sin(\frac{21\pi}{4}) \end{pmatrix}
\end{aligned}$$

That is, the orthogonal bases in row wise are,

$$\begin{aligned}
\cos(0\cdot\hat{\theta}) &= \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \\
\cos(1\cdot\hat{\theta}) &= \begin{pmatrix} 1 & \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} & -1 & \frac{-1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{pmatrix} \\
\cos(2\cdot\hat{\theta}) &= \begin{pmatrix} 1 & 0 & -1 & 0 & 1 & 0 & -1 & 1 \end{pmatrix} \\
\cos(3\cdot\hat{\theta}) &= \begin{pmatrix} 1 & \frac{-1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & -1 & \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} \end{pmatrix} \\
\cos(4\cdot\hat{\theta}) &= \begin{pmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{pmatrix} \\
\sin(1\cdot\hat{\theta}) &= \begin{pmatrix} 0 & \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} & -1 & \frac{-1}{\sqrt{2}} \end{pmatrix} \\
\sin(2\cdot\hat{\theta}) &= \begin{pmatrix} 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 \end{pmatrix} \\
\sin(3\cdot\hat{\theta}) &= \begin{pmatrix} 0 & \frac{1}{\sqrt{2}} & -1 & \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} & 1 & \frac{-1}{\sqrt{2}} \end{pmatrix}
\end{aligned}$$

Now by normalizing each vector and arrange as the **rows of a matrix**, the orthonormal basis matrix is obtained as,

$$\begin{bmatrix}
\frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} \\
\frac{1}{2} & \frac{1}{2\sqrt{2}} & 0 & -\frac{1}{2\sqrt{2}} & -\frac{1}{\sqrt{2}} & -\frac{1}{2\sqrt{2}} & 0 & \frac{1}{2\sqrt{2}} \\
\frac{1}{2} & 0 & -\frac{1}{2} & 0 & \frac{1}{2} & 0 & -\frac{1}{2} & 0 \\
\frac{1}{2} & -\frac{1}{2\sqrt{2}} & 0 & \frac{1}{2\sqrt{2}} & -\frac{1}{2} & \frac{1}{2\sqrt{2}} & 0 & -\frac{1}{2\sqrt{2}} \\
\frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} \\
0 & \frac{1}{2\sqrt{2}} & \frac{1}{2} & \frac{1}{2\sqrt{2}} & 0 & -\frac{1}{2\sqrt{2}} & -\frac{1}{2} & -\frac{1}{2\sqrt{2}} \\
0 & \frac{1}{2} & 0 & -\frac{1}{2} & 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\
0 & \frac{1}{2\sqrt{2}} & -1 & \frac{1}{2\sqrt{2}} & 0 & -\frac{1}{2\sqrt{2}} & 1 & -\frac{1}{2\sqrt{2}}
\end{bmatrix}$$

We can call these matrices as real DFT transform matrices. The following Matlab code explains the creation of real DFT transform matrices. The basis matrix, D represents the row wise orthogonal bases and hence DD^T is diagonal matrix. Then for making the normalized bases, divide each row by its norm. *Dnormalized* represents the orthonormal bases as column wise.

Matlab code

```
clc;clear all;close all;

N = 8;

theta = (0:N-1)*(2*pi/N);
```

D = [cos((0:N/2)'*theta);sin((1:N-(N/2+1))'*theta)]; *%row wise orthogonal bases (D*D'=diagonal)*

Dnormalized = [D(1,:)*sqrt(1/N); D(2:(N/2),:)*sqrt(2/N);D((N/2)+1,:)*sqrt(1/N);...

D((N/2)+2:end,:)*sqrt(2/N)]' *% column wise orthonormal bases*

1.1.2 Sampling half cosine functions

Instead of sine and cosines, we can use multiples of half-cosine waves. By sampling cosines functions which are integer multiples, half cosine waves (fundamental is created by sampling the range 0 to Π). The sampling can be understood from the figure. In the figure, red line shows the sampling location and the height value corresponds to the elements of basis. If we have N-tuple data vector, we need N bases. These N bases are created by taking N-equispaced samples from N cosine waves. It start sampling from $\frac{\pi}{2N}$ with intervals of $\frac{\pi}{N}$. We also need to normalize each bases to make it orthonormal bases.

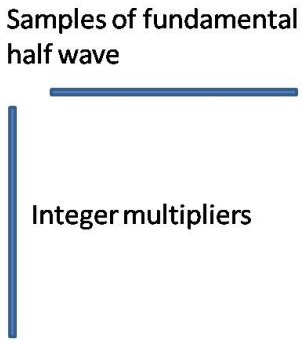

Figure 1.3: Visualization of discrete cosine transform

The following Matlab code explains the generation of discrete cosine bases. The basis matrix D represents the row-wise orthogonal bases (DD^T is diagonal matrix). Then by dividing each row by its norm, the normalized bases are generated. Another

single line command for generating DCT bases are also given. The reader can verify both using examples.

Matlab code

clc; clear all; close all;

N = 8; *% length of the signal*

k = zeros(N,1);

b = (0:N-1)*pi/N+pi/(2*N); *% row vector*

mul = (0:N-1)'; *% column vector*

D = cos(mul*b); *% rowwise bases (D*D'=diagonal)*

D = [D(1,:); D(2:end,:);]; *% rowwise normalized bases*

D'*D ;*% Identity matrix*

The intial step, D = cos(mul*b);, for cosine basis formation can be understood as an outer-product operation as $D = \cos(mul \otimes b)$. For $N = 4, b = \begin{bmatrix} 0.3927 & 1.1781 & 1.0635 & 2.7489 \end{bmatrix}$, $mul = \begin{bmatrix} 0 & 1 & 2 & 3 \end{bmatrix}^T$,. The outer-product operation $mul \otimes b$ gives a matrix

$$mul \otimes b = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0.3927 & 1.1781 & 1.0635 & 2.7489 \end{array} \\ \end{array}$$

That is,

$$mul \otimes b = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.3927 & 1.1781 & 1.9635 & 2.7489 \\ 0.7854 & 2.3562 & 3.9270 & 5.4978 \\ 1.1781 & 3.5343 & 5.8905 & 8.2467 \end{bmatrix}$$

Hence, the cosine operation will look like,

$$D = \cos\left(\begin{bmatrix} \\ mul \otimes b \\ \\ \end{bmatrix}\right)$$

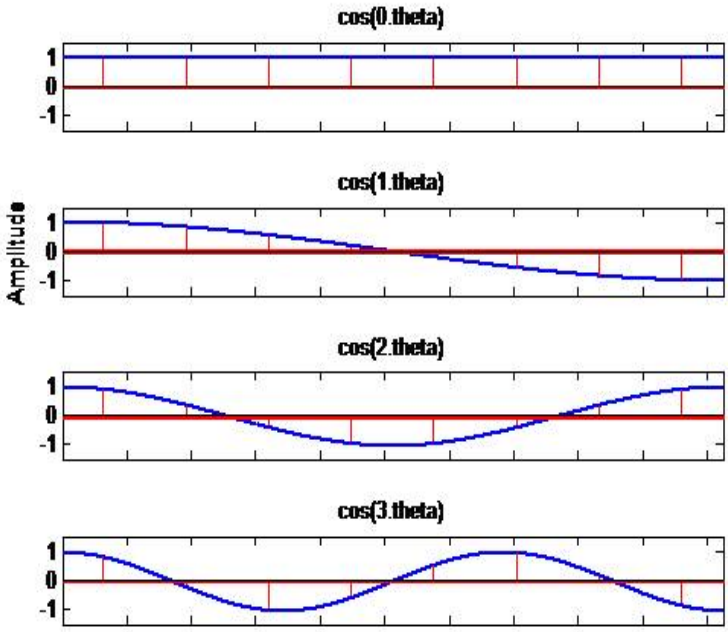

Figure 1.4: Plot for discrete cosine transform

Another method-Matlab code

N = 4; *% length of the signal*

DCT = dct(eye(N)) *% rowwise normalized bases*

% OR

DCT = dctmtx(N) *% rowwise normalized bases*

1.1.3 Sampling half sine functions

Matlab code

clc; clear all; close all;

N = 8; *% length of the signal*

k = zeros(N,1);

b = (0:N-1)*pi/N+pi/(2*N); *% row vector*

mul = (0:N-1)'; *% column vector*

D = sin(mul*b); *% rowwise bases (D*D'=diagonal)*

D = [D(1,:); D(2:end,:);]; *% rowwise normalized bases*

D'*D ;*% Identity matrix*

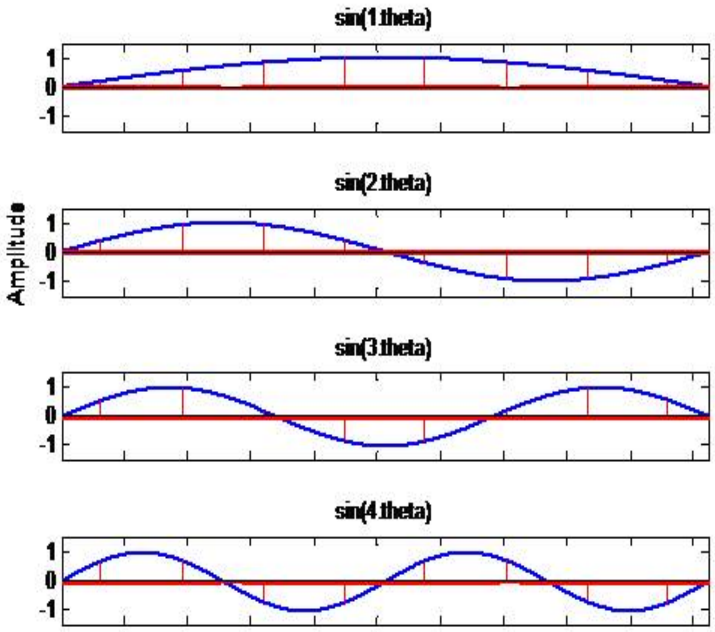

Figure 1.5: Plot for discrete sine transform

1.1.4 Walsh-Hadamard transform

Kronecker product maps two arbitrarily dimensioned matrices into a larger block matrix. Given an $m \times n$ matrix A and $k \times l$ matrix B,

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}_{m \times n} \text{ and } B = \begin{bmatrix} b_{11} & \cdots & b_{1l} \\ \vdots & \ddots & \vdots \\ b_{k1} & \cdots & b_{kl} \end{bmatrix}_{k \times l}$$

their Kronecker product, $A \otimes B$ is a larger block matrix of size $mk \times nl$ as follows,

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ b_{m1}B & \cdots & b_{mn}B \end{bmatrix}_{mk \times nl}$$

For example, given $A = \begin{bmatrix} 1 & 0 \\ -2 & 1 \end{bmatrix}_{2\times 2}$ and $B = \begin{bmatrix} 1 & 2 & 0 \\ -1 & 4 & 3 \end{bmatrix}_{2\times 3}$, their Kronecker product $A \otimes B$ of size 4×6 is,

$$A \otimes B = \begin{bmatrix} 1\begin{bmatrix} 1 & 2 & 0 \\ -1 & 4 & 3 \end{bmatrix} & 0\begin{bmatrix} 1 & 2 & 0 \\ -1 & 4 & 3 \end{bmatrix} \\ -2\begin{bmatrix} 1 & 2 & 0 \\ -1 & 4 & 3 \end{bmatrix} & 1\begin{bmatrix} 1 & 2 & 0 \\ -1 & 4 & 3 \end{bmatrix} \end{bmatrix}_{4\times 6} = \begin{bmatrix} 1 & 2 & 0 & 0 & 0 & 0 \\ -1 & 4 & 3 & 0 & 0 & 0 \\ -2 & -4 & 0 & 1 & 2 & 0 \\ 2 & -8 & -6 & -1 & 4 & 3 \end{bmatrix}_{4\times 6}$$

Next, we are going to see how the concept of Kronecker product is used for generating orthogonal Walsh - Hadamard bases of R^8.

Let,$A = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}_{2\times 2}$, then $B = A \otimes A = \begin{bmatrix} 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & -1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \end{bmatrix}_{4\times 4} =$

$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}_{4\times 4}$ Now,

$$H = B \otimes A = \begin{bmatrix} 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & -1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & -1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & -1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & -1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & -1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & -1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 1\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \end{bmatrix}_{8\times 8}$$

and the Walsh-Hadamard base matrix is obtained as follows,

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}_{8\times 8}$$

Dividing each row by $\frac{1}{\sqrt{N}}$ gives the normalized bases. The following Matlab code explains the Walsh Hadamard basis creation using kronecker product.

Matlab code

```
N = 8;

A = [1 1;1 -1];

B = kron(A,A);

H = kron(B,A); % orthogonal bases

Hnor = (1/sqrt(N))*H; % orthonormal bases

% The in-built command for creating Walsh Hadamard bases is,

H = hadamard(N)
```

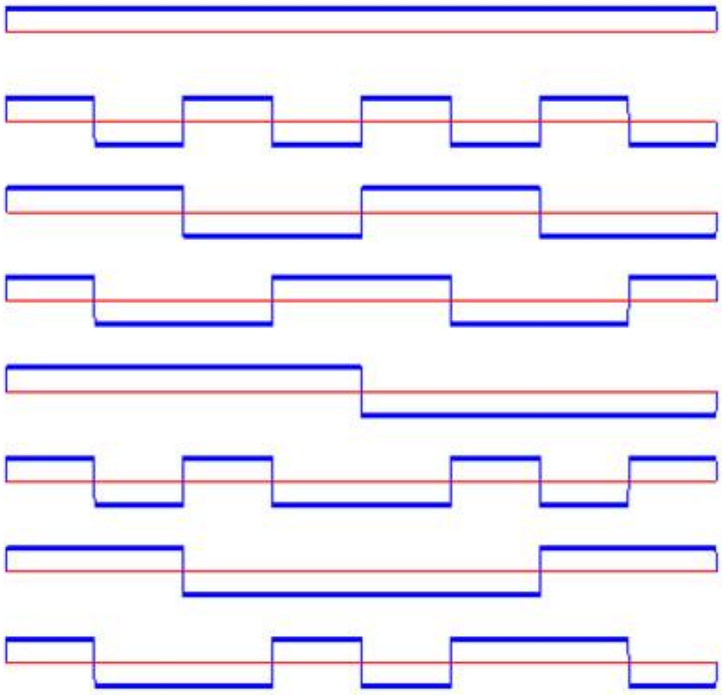

Figure 1.6: Plot for Walsh-Hadamard bases

1.1.5 Orthogonal wavelet matrix using Haar scaling and wavelet functions

Wavelet bases are sparse and localized bases (bases which are not spread across the domain) and they allow the multi-resolution decomposition of signals. The 'Haar' wavelet transform bases for representing a vector/signal in R^8 is as follows,

$$W = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}_{8\times 8}$$

The orthogonal bases are given as the rows of the above matrix W, and dividing each

term by $\sqrt{2}$ will give the orthonormal 'Haar' wavelet bases. For example, consider an 8-point signal, . Using the 'Haar' wavelet transform matrix, the first-level decomposition coefficients are obtained as, $x = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \end{bmatrix}^T$. Using the 'Haar' wavelet transform matrix, the first-level decomposition coefficients are obtained as,

$$\begin{bmatrix} s(1) \\ s(2) \\ s(3) \\ s(4) \\ d(1) \\ d(2) \\ d(3) \\ d(4) \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix}$$

Here, $\begin{bmatrix} s(1) & s(2) & s(3) & s(4) \end{bmatrix}$ represents the low-pass filter coefficients and $\begin{bmatrix} d(1) & d(2) & d(3) & d(4) \end{bmatrix}$ $x = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix}$ represents the high-pass filter coefficients. The second level decomposition coefficients are obtained as,

$$\begin{bmatrix} ss(1) \\ ss(2) \\ ds(1) \\ ds(2) \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} s(1) \\ s(2) \\ s(3) \\ s(4) \end{bmatrix}$$

While going for the third level,

$$\begin{bmatrix} sss(1) \\ dss(1) \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} ss(1) \\ ss(2) \end{bmatrix}$$

and the final decomposed structure is given as,

$$\begin{bmatrix} sss(1) & dss(1) & ds(1) & ds(2) & d(1) & d(2) & d(3) & d(4) \end{bmatrix}^T$$

Let, 'k' denotes the scale of transformation. For 'k=1', the wavelet transformation matrix will be,

$$W = \begin{bmatrix} 0.7071 & 0.7071 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.7071 & 0.7071 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7071 & 0.7071 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.7071 & 0.7071 \\ 0.7071 & -0.7071 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.7071 & -0.7071 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7071 & -0.7071 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.7071 & -0.7071 \end{bmatrix}$$

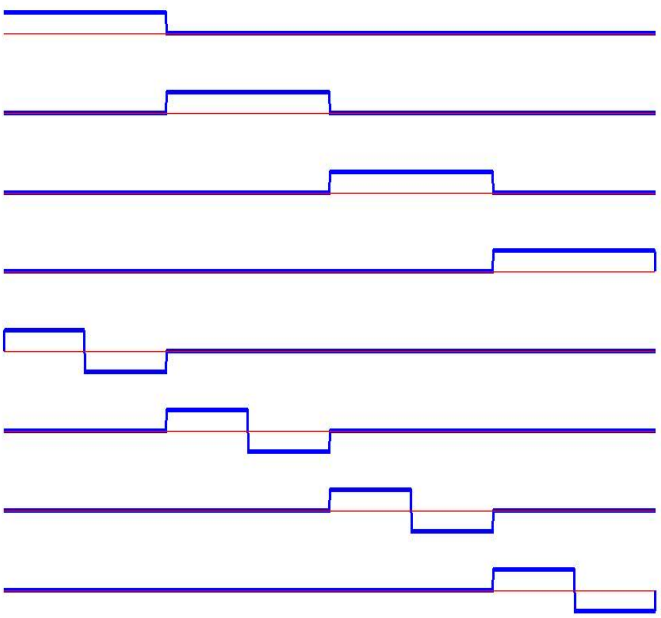

Figure 1.7: Plot for Haar wavelet bases,k=1

For 'k=2',

$$W = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & -0.5 & -0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & -0.5 & -0.5 \\ 0.7071 & -0.7071 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.7071 & -0.7071 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7071 & -0.7071 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.7071 & -0.7071 \end{bmatrix}$$

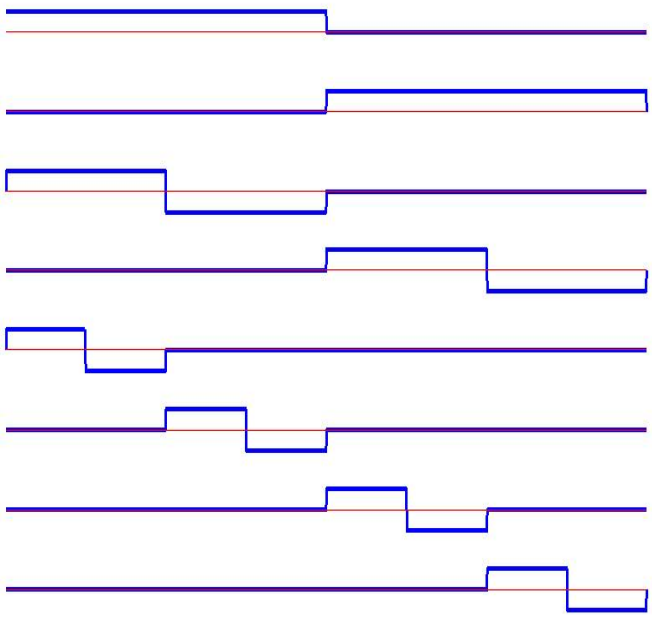

Figure 1.8: Plot for Haar wavelet bases,k=2

For 'k=3',

$$W = \begin{bmatrix} 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 \\ 0.3536 & 0.3536 & 0.3536 & 0.3536 & -0.3536 & -0.3536 & -0.3536 & -0.3536 \\ 0.5 & 0.5 & -0.5 & -0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & -0.5 & -0.5 \\ 0.7071 & -0.7071 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0.7071 & -0.7071 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7071 & -0.7071 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.7071 & -0.7071 \end{bmatrix}$$

In general, for an the 8-point signal, the coefficients can be obtained as (k =3),

$$\begin{bmatrix} sss(1) \\ dss(1) \\ ds(1) \\ ds(2) \\ d(1) \\ d(2) \\ d(3) \\ d(4) \end{bmatrix} = \begin{bmatrix} 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 \\ 0.3536 & 0.3536 & 0.3536 & 0.3536 & -0.3536 & -0.3536 & -0.3536 & -0.3536 \\ 0.5 & 0.5 & -0.5 & -0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & -0.5 & -0.5 \\ 0.7071 & -0.7071 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0.7071 & -0.7071 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7071 & -0.7071 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.7071 & -0.7071 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix}$$

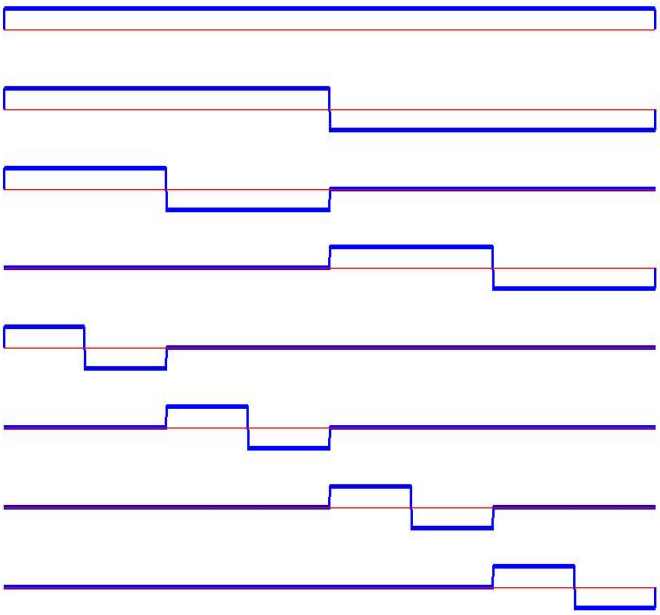

Figure 1.9: Plot for Haar wavelet bases,k=3

Matlab code

clc;clear all;close all;

N = 8;

% Set wavelet name. wname = 'haar'

% % [LoD,HiD,LoR,HiR] = wfilters(wname);

%WAVMAT returns a wavelet transformation matrix

% Input:

% N: matrix dimension, which is required to be power of 2.

% h: analysis lowpass filter

% g: analysis highpass filter

% k: number of scales of transformation

% Output:

% wmat: wavelet transformation matrix

h = LoD;

g = HiD;

k = 1;

W = wavmat(N, h, g, k)

W'*W; % Identity matrix

1.1.6 Orthogonal wavelet matrix using Daubechies scaling and wavelet functions

$$W = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & -0.5 & -0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & -0.5 & -0.5 \\ 0.7071 & -0.7071 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.7071 & -0.7071 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7071 & -0.7071 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.7071 & -0.7071 \end{bmatrix}$$

1.1.7 Complex exponential functions (DFT)

Orthogonal bases can be created by equispaced samples of complex exponentials. Earlier, we created bases by sampling sine and cosine harmonic functions. We can combine cosine and sine function with same frequency to create complex exponential function. The 'exponential function' play very important role in conventional signal processing. The advantage of using complex exponential basis is that there exist an algorithm for

finding the transform (matrix-vector multiplication operation) in a very fast way. It is called FFT (Fast Fourier Transform) and it is a matrix-free method. That is, the algorithm never creates a basis matrix for transform operation. We will explain with an example for vectors in ^{4}. Consider a 4-tuple data, we sample the domain 0 to 2π as before and find bases as rows in figure 1.10.

$\theta =$	$0\times\frac{2\pi}{4}$	$1\times\frac{2\pi}{4}$	$2\times\frac{2\pi}{4}$	$3\times\frac{2\pi}{4}$	
$e^{j\times0\times\theta} =$	1	1	1	1	base1
$e^{j\times1\times\theta} =$	1	0 +1j	-1	0 -1j	base2
$e^{j\times2\times\theta} =$	1	-1	1	-1	base3
$e^{j\times3\times\theta} =$	1	0-1j	-1	0+1j	base4

Figure 1.10: Complex Fourier basis

The above Fourier bases creation can also be understood using an outer-product operation, denoted by$\otimes$. Let 'index' be a vector denoting the basis number and 'theta' be a vector denoting angles.

$$index = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$

and

$$theta = \begin{bmatrix} 0 & \frac{2\pi}{4} & \pi & \frac{6\pi}{4} \end{bmatrix}$$

Let $M \in {}^{4\times4}$be a matrix, formed using outer-product operation, $M = index \otimes theta$. That is,

$$M = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1.5708 & 3.1416 & 4.7124 \\ 0 & 3.1416 & 6.2832 & 9.4248 \\ 0 & 4.7124 & 9.4248 & 14.1372 \end{bmatrix}$$

basis matrix will be formed using a single command in Matlab as $\exp(j \times M)$. $j$denotes the complex number.

$$\exp\left(j \times \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1.5708 & 3.1416 & 4.7124 \\ 0 & 3.1416 & 6.2832 & 9.4248 \\ 0 & 4.7124 & 9.4248 & 14.1372 \end{bmatrix}\right) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{matrix} \leftarrow 1st basis \\ \leftarrow 2nd basis \\ \leftarrow 3\text{rd} basis \\ \leftarrow 4\text{th} basis \end{matrix}$$

For normalizing these basis, divide each row by its norm and hence the normalized bases are given as,

$$\frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

It can be shown that, for this theta, $e^{j3\theta} = e^{-j1\theta}$. The integer multiplier in the exponent corresponds to number of full waves within sampling period T. (later we will relate this new identity to aliasing). Now explain the process for N=8 samples. The bases for N=8 is as shown in figure. Bases are shown as column vectors. We may interpret the integer multiplier in the exponent as wave number. Note how bases with different wave numbers are arranged in a DFT matrix (in the figure it is in columns). Try to understand the physical interpretaion of negative wave number from the figure 1.11.

So complex form of DFT is based on the following fact

1. Continous domain Fourier series bases, $b_m = \frac{1}{\sqrt{T}} e^{jm\Omega t}, b_n = \frac{1}{\sqrt{T}} e^{jn\Omega t}$ with $\Omega = \frac{2\pi}{T}$is orthonormal since
$$\int_0^T b_m \bar{b}_n dt = 0 if m \neq n, 1 if m = n$$

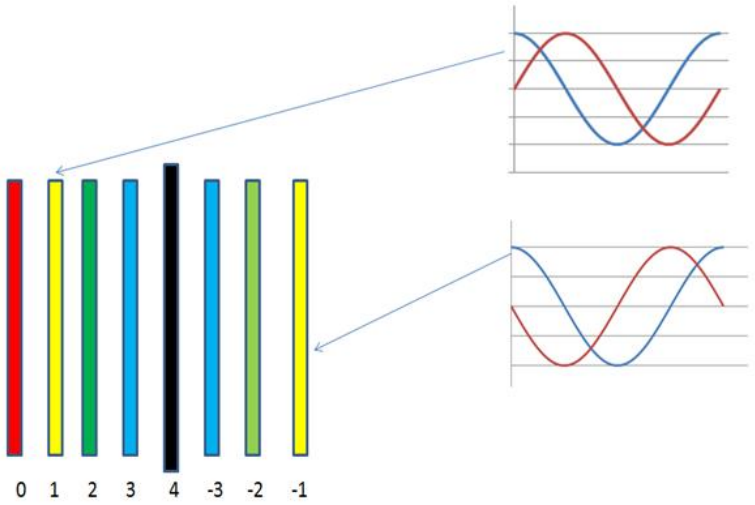

Figure 1.11: basis for N=8

Note that in case complex quantity is involved inner product definition is modified as, $\langle b_m, b_n \rangle = \int_0^T b_m \bar{b}_n dt$

2. Sampling such bases lead to discrete domain complex bases.

3. N-point DFT is created with N bases starting from wave number 0 to wave number N-1.

However we treat bases starting from N/2 +1 onwards as negative wave numbers as in 1.11. This can be easily verified. Alternately we can think of bases after N/2th basis are negative wave number basis. This in fact a necessity for representing real signals. We know $\cos \Omega t = \frac{e^{j\Omega t}+e^{-j\Omega t}}{2}$. That is we need positive wave number bases and negative wave number bases to represent a real signal. For N=8, the sampling instances for creating the bases are given as, $\widehat{\theta} = \left[\begin{array}{cccccc} 0 \cdot \frac{2\pi}{8} & 1 \cdot \frac{2\pi}{8} & 2 \cdot \frac{2\pi}{8} & \cdots & 7 \cdot \frac{2\pi}{8} \end{array} \right]$ and the following matrix represents the normalized bases as columns,

$$\frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & C & j & -\sqrt{2}+j\sqrt{2} & -1 & -\sqrt{2}-j\sqrt{2} & -j & \sqrt{2}-j\sqrt{2} \\ 1 & j & -1 & -j & 1 & j & -1-j & -j \\ 1 & -\sqrt{2}+j\sqrt{2} & -j & \sqrt{2}+j\sqrt{2} & -1 & \sqrt{2}-j\sqrt{2} & j & -\sqrt{2}-j\sqrt{2} \\ 1 & -1+j & 1-j & -1+j & 1 & -1-j & 1+j & -1-j \\ 1 & -\sqrt{2}-j\sqrt{2} & j & \sqrt{2}-j\sqrt{2} & -1 & \sqrt{2}+j\sqrt{2} & -j & -\sqrt{2}+j\sqrt{2} \\ 1 & -j & -1-j & j & 1 & -j & -1+j & j \\ 1 & \sqrt{2}-j\sqrt{2} & -j & -\sqrt{2}-j\sqrt{2} & -1 & -\sqrt{2}+j\sqrt{2} & j & \sqrt{2}+j\sqrt{2} \end{bmatrix}$$

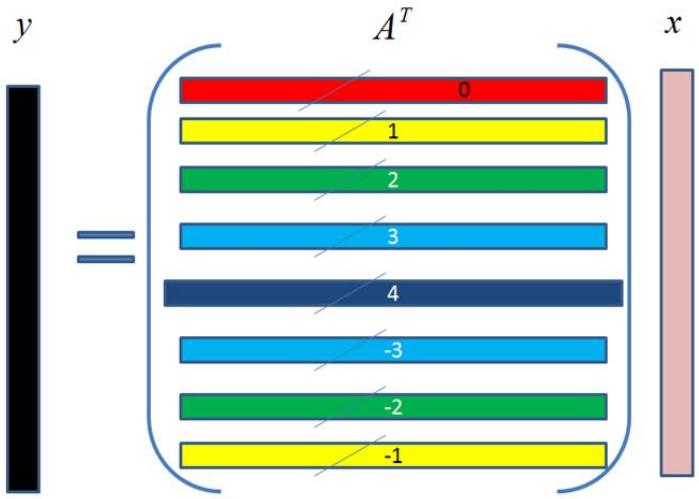

Figure 1.12: Illustation

The following Matlab code explains the generation of complex DFT matrix as rows of basis vector.

Matlab code for creating DFT matrix as rows as basis vectors

N = 4;

b = [(0:N-1)*2*pi/N] ;

% row vector

mul = [0:N-1]' ;

% column vector

```
bm = exp(i*mul*b);
% row wise basis vectors
DFTmatrix =(1/sqrt(N)*bm) % rowwise normalized bases
```

The following Matlab code explains the generation of complex DFT matrix as rows as conjugate of basis vector. The single line command for the same complex DFT matrix is also given. reader can verify both through examples.

Matlab code for creating DFT matrix as rows as conjugate of basis vectors

```
clc;clear all;close all;
N = 4;
% length of the signal
b = (0:N-1)*2*pi/N;
% row vector
mul = (0:N-1)' ;
% column vector
bm = exp(-i*mul*b)
% conjugate as basis vectors (row wise)
DFTmatrix = sqrt(1/N)*bm;
% Rows as conjugate of basis vectors (normalized)
```

Another method - dft = fft(eye(N))

1.1.8 Applications of DFT

1. Getting approximate spectral content of a signal

2. Convolution operation- doing covolution of two time domian signals by multiplication of corresponding elements in Fourier domain and then inverse transforming

3. Digiatal filtering

1.2 Creation of Bi-Orthogonal bases

1.2.1 Bi-Orthogonal Wavelet matrix using Daubechies bi-orthogonal scaling and wavelet functions

wname = 'bior3.3';

% Set wavelet name
$\mathrm{Lo}_D, Hi_D, Lo_R, Hi_R$

$= wfilters(wname);$

$N = 8;$

$k = 2;$

$W1 = wavmat(N, Lo_D, Hi_D, k);$

$W2 = wavmat(N, Lo_R, Hi_R, k);$

$W1 * W2'$

$\%Identitymatrix$

1.2.2 Bi-Orthogonal Wavelet matrix using spline scaling and wavelet functions

Now the question is how do we use these bases and how do we interpret transforms.

We can interpret matrix vector multiplication in two ways.

(a) A linear combination

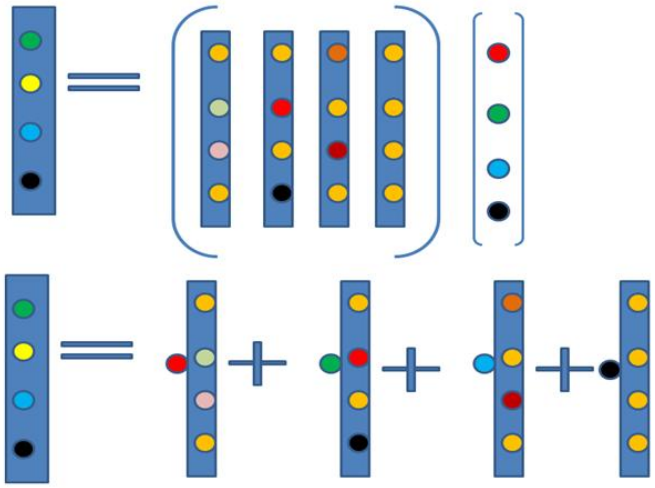

Figure 1.13: Illustration of matrix-vector multiplication as linear combination

(b) A series of dot products (we call it as projections)

We can relate this operation with forward and inverse transform in signal processing. When we have basis in columns we take linear combination interpretation. When we have basis in rows we go for dot product interpretation. Therefore, the linear combination interpretation correspond to inverse transform and the dot product interpretation correspond to forward transform in signal processing. Consider an example in R^4. Let x represents the signal, A represents the basis

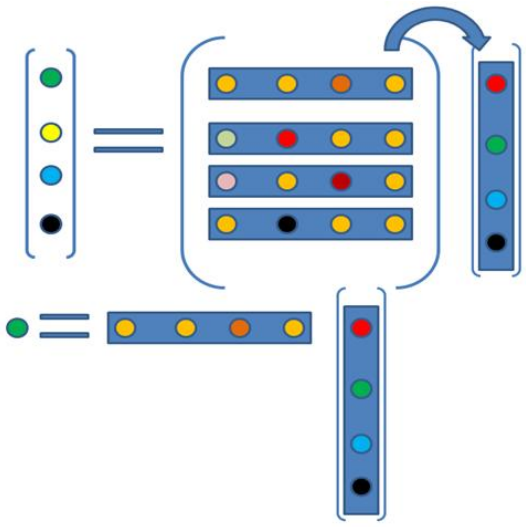

Figure 1.14: Illustration of matrix-vector multiplication as a series of dot products

transform matrix and $y$represents the transform coefficients. The normalized sine and cosine basis created earlier are taken and we put as columns of a matrix A as follows,

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

Here, we have basis as column and hence we go for linear combination interpretation. Now try to express $x$as a linear combination of bases and be the coefficient vector. Then $x = Ax$, that is,

$$\begin{array}{cccccccc} | & & | & & | & & | \\ x & = y_1 & b_1 & + y_2 & b_2 & + \cdots + y_n & b_n \\ | & & | & & | & & | \end{array}$$

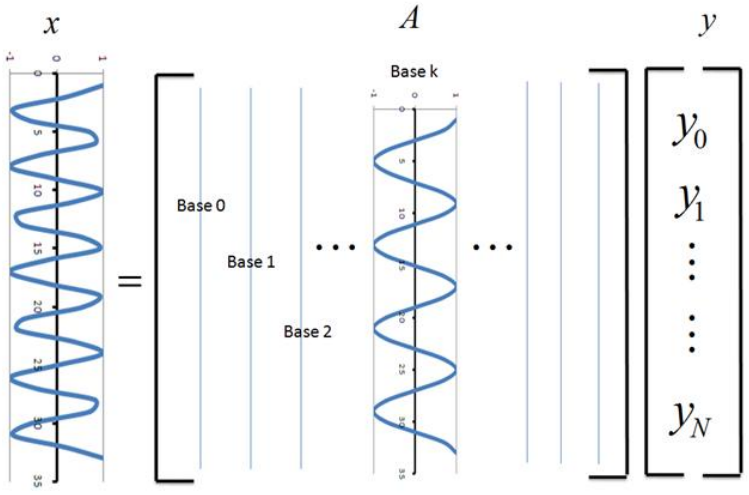

Figure 1.15: Illustration of $x = Ax$

That is, $\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{\sqrt{2}} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{-1}{2} & \frac{1}{\sqrt{2}} \\ \frac{1}{2} & \frac{-1}{\sqrt{2}} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{-1}{2} & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = y_1 \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix} + y_2 \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{-1}{\sqrt{2}} \\ 0 \end{bmatrix} + y_3 \begin{bmatrix} \frac{1}{2} \\ \frac{-1}{2} \\ \frac{1}{2} \\ \frac{-1}{2} \end{bmatrix} + y_4 \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ 0 \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$$

Since A is orthogonal to $y = A^T x$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \langle x, b_1 \rangle \\ \langle x, b_2 \rangle \\ \langle x, b_3 \rangle \\ \langle x, b_4 \rangle \end{bmatrix}$$

Note that in A^T, the basis are in rows. So here we give dot product interpretation.

That is, $\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} & 0 \\ \frac{1}{2} & \frac{-1}{2} & \frac{1}{2} & \frac{-1}{2} \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$

Now, combining the above two equation we get

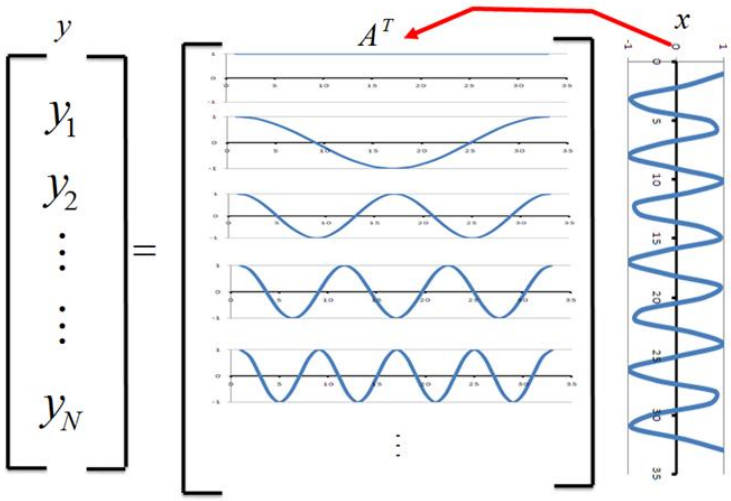

Figure 1.16: Illustration of $x = Ax$

$$x = y_1 b_1 + y_2 b_2 + \cdots + y_n b_n$$

Note that in case bases b are discretized version of

(a) real function then $\langle x, b_i \rangle = x^T b_i$

(b) complex exponential function, then $\langle x, b_i \rangle = x^T \bar{b}_i$, where superscript - stand for complex conjugation (of each element of vector)

It is left to reader to find the reason for necessity of such a twist in the definition of inner product. Readers can verify that the equation $x = y_1 b_1 + y_2 b_2 + \cdots + y_n b_n$ is true for real signal x if and only if we define inner product in that way. If you are able to appreciate the formula $x = y_1 b_1 + y_2 b_2 + \cdots + y_n b_n$ you have already studied a big part of conventional signal processing.

1.2.3 Example 1: Real function

Consider a vector,

$x = \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}$, the real DFT transform matrix, $A = \begin{bmatrix} \frac{1}{2} & \frac{1}{\sqrt{2}} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{-1}{2} & \frac{1}{\sqrt{2}} \\ \frac{1}{2} & \frac{-1}{\sqrt{2}} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{-1}{2} & \frac{-1}{\sqrt{2}} \end{bmatrix}$

, here the basis are the columns of A and $y$be the transform coefficients. Now express vector $x$as the linear combination of bases as follows, $x = Ay$and which represents the forward transform.

$$\begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{\sqrt{2}} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{-1}{2} & \frac{1}{\sqrt{2}} \\ \frac{1}{2} & \frac{-1}{\sqrt{2}} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{-1}{2} & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix} = y_1 \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix} + y_2 \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{-1}{\sqrt{2}} \\ 0 \end{bmatrix} + y_3 \begin{bmatrix} \frac{1}{2} \\ \frac{-1}{2} \\ \frac{1}{2} \\ \frac{-1}{2} \end{bmatrix} + y_4 \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ 0 \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$$

The transform coefficients can be obtained by inverse transform as, $y = A^T x$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} & 0 \\ \frac{1}{2} & \frac{-1}{2} & \frac{1}{2} & \frac{-1}{2} \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}$$

Here, we go for the dot product interpretation as follows,

$$y_1 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}, y_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix},$$

$$y_3 = \begin{bmatrix} \frac{1}{2} & \frac{-1}{2} & \frac{1}{2} & \frac{-1}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}, y_4 = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}$$

On solving this, the coefficients are obtained as, $\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} \frac{11}{2} \\ -2\sqrt{2} \\ \frac{1}{2} \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$

Now,

$$\begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix} = \frac{11}{2} \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix} + -2\sqrt{2} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{-1}{\sqrt{2}} \\ 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \frac{1}{2} \\ \frac{-1}{2} \\ \frac{1}{2} \\ \frac{-1}{2} \end{bmatrix} + \frac{-1}{\sqrt{2}} \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ 0 \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$$

That is,

$$\begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix} = \begin{matrix} | \\ x \\ | \end{matrix} = \begin{matrix} | \\ \langle x, \bar{b}_1 \rangle \\ | \end{matrix} \begin{matrix} | \\ b_1 \\ | \end{matrix} + \begin{matrix} | \\ \langle x, \bar{b}_2 \rangle \\ | \end{matrix} \begin{matrix} | \\ b_2 \\ | \end{matrix} + ... + \begin{matrix} | \\ \langle x, \bar{b}_n \rangle \\ | \end{matrix} \begin{matrix} | \\ b_n \\ | \end{matrix}$$

1.2.4 Example 2: Complex function

Consider a vector, $x = \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}$, the complex DFT transform matrix, $A = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{i}{2} & \frac{-1}{2} & \frac{-i}{2} \\ \frac{1}{2} & \frac{-1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{-i}{2} & \frac{-1}{2} & \frac{-i}{2} \end{bmatrix}$, here the basis are the columns of A and $y$be the transform coefficients. Now express vector $x$as the linear combination of bases as follows, $x = Ay$and which represents the forward transform.

$$\begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{i}{2} & \frac{-1}{2} & \frac{-i}{2} \\ \frac{1}{2} & \frac{-1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{-i}{2} & \frac{-1}{2} & \frac{-i}{2} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix} = y_1 \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix} + y_2 \begin{bmatrix} \frac{1}{2} \\ \frac{i}{2} \\ \frac{-1}{2} \\ \frac{-i}{2} \end{bmatrix} + y_3 \begin{bmatrix} \frac{1}{2} \\ \frac{-1}{2} \\ \frac{1}{2} \\ \frac{-1}{2} \end{bmatrix} + y_4 \begin{bmatrix} \frac{1}{2} \\ \frac{-i}{2} \\ \frac{-1}{2} \\ \frac{-i}{2} \end{bmatrix}$$

Now the transform coefficients can be obtained by inverse transform as, $y = A^T x$.

Since $A$is the complex DFT matrix, A^Trepresents the complex conjugate basis vectors of Aas row-wise.

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{i}{2} & \frac{-1}{2} & \frac{-i}{2} \\ \frac{1}{2} & \frac{-1}{2} & \frac{1}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{-i}{2} & \frac{-1}{2} & \frac{-i}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}$$

Here we go for the dot product interpretation as follows,

$$y_1 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}, y_2 = \begin{bmatrix} \frac{1}{2} & \frac{-i}{2} & \frac{-1}{2} & \frac{i}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix},$$

$$y_3 = \begin{bmatrix} \frac{1}{2} & \frac{-1}{2} & \frac{1}{2} & \frac{-1}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}, y_4 = \begin{bmatrix} \frac{1}{2} & \frac{i}{2} & \frac{-1}{2} & \frac{-i}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}$$

On solving, the coefficients obtained are,

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 5.5 \\ -2 + 0.5i \\ 0.5 \\ -2 - 0.5i \end{bmatrix}$$

Now,

$$\begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix} = 5.5 \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix} + (-2 + 0.5i) \begin{bmatrix} \frac{1}{2} \\ \frac{i}{2} \\ \frac{-1}{2} \\ \frac{-i}{2} \end{bmatrix} + 0.5 \begin{bmatrix} \frac{1}{2} \\ \frac{-1}{2} \\ \frac{1}{2} \\ \frac{-1}{2} \end{bmatrix} + (-2 - 0.5i) \begin{bmatrix} \frac{1}{2} \\ \frac{-i}{2} \\ \frac{-1}{2} \\ \frac{i}{2} \end{bmatrix}$$

That is,

$$\begin{bmatrix} 1 \\ 2 \\ 5 \\ 3 \end{bmatrix} = \begin{matrix} | \\ x \\ | \end{matrix} \begin{matrix} = \langle x, b_1 \rangle \\ \end{matrix} \begin{matrix} | \\ b_1 \\ | \end{matrix} \begin{matrix} + \langle x, b_2 \rangle \\ \end{matrix} \begin{matrix} | \\ b_2 \\ | \end{matrix} \begin{matrix} + ... + \langle x, b_n \rangle \\ \end{matrix} \begin{matrix} | \\ b_n \\ | \end{matrix}$$

1.3 Least Squares in Signal Processing

1.3.1 Concept of Least Square fitting

If the function is convex and differentiable, we can obtain optimum point - the location x^* at which the function value is minimum by applying first order condition that

$$\nabla f(x^*) = 0(vector)$$

. We start with single variable linear regression problem. Let,

$$(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$$

be the given data points. The scatter diagram be as shown in figure 1.18. We have to find slope m and intercept c of the regression line given by, $y = mx + c$. In matrix form the problem is formulated as,

$$z^* = \begin{pmatrix} m \\ c \end{pmatrix} = \arg\min_z \|Az - y\|_2^2$$

where

$$Az = y + e\text{is} \underbrace{\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ . & . \\ x_n & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} m \\ c \end{bmatrix}}_{z} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ . \\ y_n \end{bmatrix}}_{y} + \underbrace{\begin{bmatrix} e_1 \\ e_2 \\ . \\ e_n \end{bmatrix}}_{e}$$

The objective function is basically

$$e^T e = (Az - y)^T (Az - y) = \|Az - y\|_2^2$$

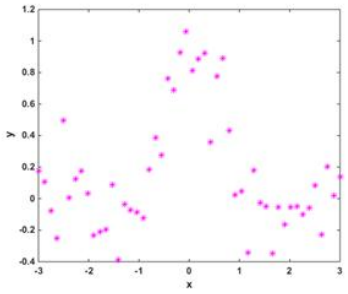

Figure 1.17: Non-linear data

wehre

$$e^T e = e_1^2 + e_2^2 + ... + e_n^2$$

is sum of square of errors. This in turn depends on m and c which in compact form is the vector z. The optimal value of m and c should minimize the sum of square of errors. Note that error term e_1, e_2etc can assume positive and negative values. Therefore we go for minimization of error sum of square rather than error sum. Now how do we find optimal$z$?. We apply first order condition. We have

$$f(z) = (Az - y)^T (Az - y) = \|Az - y\|_2^2$$

$$\nabla f(z) = 0 \Rightarrow 2A^T (Az - y) = 0 \Rightarrow z^* = (A^T A)^{-1} A^T y$$

If columns of A are independent $A^T A$is invertible. If not

$$z^* = A^+ y = pinv(A) * y$$

Let us have another example where the regression function is non-linear. Let us consider a case where relationship between x and $y$is not linear. For example,

consider the data shown in figure 1.18. Here, $y = mx + c$will not be a proper fit.

Let us consider fitting 6th degree polynomial of the form

$$y = a_0 + a_1 x + a_2 x^2 + ... + a_6 x^6$$

For each x and $y$value. We formulate a matrix equation of the form,

$$Az = y + e$$

where,

$$A = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 & x_1^4 & x_1^5 & x_1^6 \\ 1 & x_2 & x_2^2 & x_2^3 & x_2^4 & x_2^4 & x_n^6 \\ . & . & . & . & . & . & . \\ 1 & x_n & x_n^2 & x_n^3 & x_n^4 & x_n^4 & x_n^6 \end{bmatrix}, z = \begin{bmatrix} a_0 \\ a_2 \\ \vdots \\ a_6 \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, e = \begin{bmatrix} e \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

and obtain

$$z^* = \arg\min_z \|Az - y\|_2^2$$

The solution again is

$$z^* = (A^T A)^{-1} A^T y$$

. Let us find for the following data.

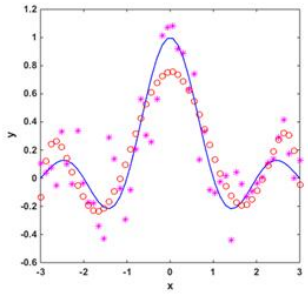

Figure 1.18: Non-linear regression

Matlab code % Sinc function data (for non-linear regression) generation

% data generation

x = linspace(-3,3,50);

y = sin(pi*x)./(pi*x);

noise = 0.2*randn(1,50);

yn = y+noise;

x = x'; yn = yn';

% parameter estimation

A = [ones(50,1) x x.^{2}x.^{3}x.^{4}x.^{5}x.^{6}];

$z = pinv(A) * yn;$

$yp = A * z;$

$\%Plotting$

$plot(x, y); holdon;$

$plot(x, yn, '*'); holdon; xlabel('x'); ylabel('y')$

$plot(x, yp, 'ro'); xlabel('x'); ylabel('y')$

Least square method can provide approximate solution to linear system of equations. It can be applied to several signal processing problems such as smoothing, deconvolution, missing data estimation, linear prediction etc. The method of least squares (LS) is extremely important and it makes good sense for many practical problems. The approach dates back to Gauss who in 1795 introduced the method to study planetary motions.

Thinking in term of statistical signal processing, we are assuming a static model with error signal orthogonal to model space. Or in other words no probabilistic assumptions are made about the data, only a signal model is assumed. This static model many times are expressed in terms of sparsity of the signal with respect to its transform coefficients or sparsity in terms of the local derivatives or smoothness. Usually the algorithm is easy to implement, either in a blockbased or sequential manner. This amounts to the minimization of a quadratic cost function. Within the least squares approach we attempt to minimize the squared difference between the observed data and the assumed model of noiseless data. One drawback of this approach is that rigorous statistical performance cannot be assessed without some specific assumptions about probabilistic structure in the data. We can solve least squares problem in block or in sequential. Weighted least squares method allow to assign confidence to samples. We can also use a forgetting factor to deal with time-varying statistics. A number of applications of least squares theory: adaptive noise cancellation, digital filter design, Prony type spectral estimation, and many more. This session explains the solutions to the linear system of equations by least squares method and its applications in solving some important signal processing problems like signal denoising, deconvolution, missing sample estimation. Also these problems are extended to images and the solution is obtained via least square method.

1.3.2 Over and Underdetermined Systems

Consider an over-determined system $y = Ax$, where $A$is a tall matrix with linearly independent columns. The solution to this over determined system is obtained by

minimizing the energy of the error, $x^* = \arg\min_x \|y - Ax\|_2^2$

$$\begin{aligned} f(x) &= \|y - Ax\|_2^2 = (y - Ax)^T(y - Ax) \\ &= y^T y - y^T Ax - x^T A^T y + x^T A^T Ax \\ &= y^T y - 2y^T Ax + x^T A^T Ax \ \left((y^T Ax)^T = x^T A^T y\right) \end{aligned}$$

Differentiating with respect to x and equating to zero gives,

$$\frac{\partial}{\partial x} f(x) = -2A^T y + 2A^T Ax \qquad \left((A^T y)^T = y^T A \Rightarrow y^T Ax = (A^T y)^T x\right)$$

$$\frac{\partial}{\partial x} f(x) = 0 \quad \Rightarrow \quad A^T Ax \ = A^T y$$

Then the least square solution for $x$is obtained as, $x = (A^T A)^{-1} A^T y$

Now, consider an under-determined system of equations,$y = Ax$, where A contains linearly independent rows. A in this case will be a wide matrix. The minimum norm solution can be obtained as

$$\begin{aligned} &\min_x \|x\|_2^2 \\ &such\, that\ y = Ax \end{aligned}$$

The Lagrangian function corresponds to above optimization problem is defined as,

$L(x, \mu) = \|x\|_2^2 + \mu^T(y - Ax)$

Take the derivatives of the Lagrangian and equating to zero gives,

$$\begin{aligned} &\tfrac{\partial}{\partial x} L(x) = 2x - A^T \mu \Rightarrow x = \tfrac{1}{2} A^T \mu \\ &\tfrac{\partial}{\partial \lambda} L(x) = y - Ax \ \Rightarrow y = Ax \end{aligned}$$

By substituting the value of in yields, $y = \frac{1}{2} AA^T \mu \Rightarrow \mu = 2(AA^T)^{-1} y$

Now the least norm solution for x is obtained as, $x = A^T(AA^T)^{-1} y$

References

[1] Adelfio, G. (2012). Change-point detection for variance piecewise constant models. *Communications in Statistics - Simulation and Computation.* 41(4): 437 - 448.

[2] Aharon, M., Elad, M. and Bruckstein, A. (2006). On the uniqueness of over-complete dictionaries, and a practical way to retrieve them. *Linear Algebra and its Applications.* 416: 48 - 67.