**Unit-II**

|  |  |  |
| --- | --- | --- |
| **Session Name** | **:** | Frequency Transformation |
| **Course Title** | **:** | CS2403 -Digital Signal Processing |
| **Semester** | **:** | VII Semester |
| **Programme Name** | **:** | B.E |
|  | | |
| **Author Name** | **:** | Mrs.V.GANDHIMATHI |
| **Department** | **:** | ECE |
| **Institution Name** | **:** | VRS College of Engg. & Tech |
| **Mobile Number** | **:** | 9245582632 |
| **E-mail** | **:** | [adhihema02@gmail.com](mailto:adhihema02@gmail.com) |

**Session -1**

**1. Introduction: Discrete Fourier Transform**

Since the learners have some basic understanding about Fourier transform, it would be a good idea to ask some basic questions to help the learners recall the concepts DFT.

<http://en.wikipedia.org/wiki/Discrete_Fourier_transform>

**Introduction to DFT**

In [mathematics](http://en.wikipedia.org/wiki/Mathematics), the **discrete Fourier transform** (**DFT**) converts a finite list of equally spaced [samples](http://en.wikipedia.org/wiki/Sampling_(signal_processing)) of a [function](http://en.wikipedia.org/wiki/Function_(mathematics)) into the list of [coefficients](http://en.wikipedia.org/wiki/Coefficient) of a finite combination of [complex](http://en.wikipedia.org/wiki/Complex_number) [sinusoids](http://en.wikipedia.org/wiki/Sine_wave), ordered by their [frequencies](http://en.wikipedia.org/wiki/Frequency), that has those same sample values. It can be said to convert the sampled function from its original domain (often [time](http://en.wikipedia.org/wiki/Time_domain) or position along a line) to the [frequency domain](http://en.wikipedia.org/wiki/Frequency_domain).

The input samples are [complex numbers](http://en.wikipedia.org/wiki/Complex_number) (in practice, usually [real numbers](http://en.wikipedia.org/wiki/Real_number)), and the output coefficients are complex too. The frequencies of the output sinusoids are integer multiples of a fundamental frequency, whose corresponding period is the length of the sampling interval. The combination of sinusoids obtained through the DFT is therefore [periodic](http://en.wikipedia.org/wiki/Periodic_function) with that same period. The DFT differs from the [discrete-time Fourier transform](http://en.wikipedia.org/wiki/Discrete-time_Fourier_transform) (DTFT) in that its input and output sequences are both finite; it is therefore said to be the Fourier analysis of finite-domain (or periodic) discrete-time functions.

The DFT is the most important [discrete transform](http://en.wikipedia.org/wiki/Discrete_transform), used to perform [Fourier analysis](http://en.wikipedia.org/wiki/Fourier_analysis) in many practical applications. In [digital signal processing](http://en.wikipedia.org/wiki/Digital_signal_processing), the function is any quantity or [signal](http://en.wikipedia.org/wiki/Signal_(information_theory)) that varies over time, such as the pressure of a [sound wave](http://en.wikipedia.org/wiki/Sound_wave), a [radio](http://en.wikipedia.org/wiki/Radio) signal, or daily [temperature](http://en.wikipedia.org/wiki/Temperature) readings, sampled over a finite time interval (often defined by a [window function](http://en.wikipedia.org/wiki/Window_function)). In [image processing](http://en.wikipedia.org/wiki/Image_processing), the samples can be the values of [pixels](http://en.wikipedia.org/wiki/Pixel) along a row or column of a [raster image](http://en.wikipedia.org/wiki/Raster_image). The DFT is also used to efficiently solve [partial differential equations](http://en.wikipedia.org/wiki/Partial_differential_equations), and to perform other operations such as [convolutions](http://en.wikipedia.org/wiki/Convolution) or multiplying large integers.

Since it deals with a finite amount of data, it can be implemented in [computers](http://en.wikipedia.org/wiki/Computer) by [numerical algorithms](http://en.wikipedia.org/wiki/Numerical_algorithm) or even dedicated [hardware](http://en.wikipedia.org/wiki/Digital_circuit). These implementations usually employ efficient [fast Fourier transform](http://en.wikipedia.org/wiki/Fast_Fourier_transform) (FFT) algorithms;[[1]](http://en.wikipedia.org/wiki/Discrete_Fourier_transform" \l "cite_note-colley-1) so much so that the terms "FFT" and "DFT" are often used interchangeably. The terminology is further blurred by the (now rare) synonym [finite Fourier transform](http://en.wikipedia.org/wiki/Finite_Fourier_transform) for the DFT, which apparently predates the term "fast Fourier transform" but has the same [initialism](http://en.wikipedia.org/wiki/Initialism" \o "Initialism).

**Suggested Activity: Introduces & Questions**

**Questions**

1. Define Fourier.
2. Write the formula for Fourier transform.
3. Define IFT.
4. Differentiate FT for CT and DT signals.
5. **DFT and IDFT**

<http://www.dspguide.com/ch8/2.htm>

**Suggested Activity: Explains and Problems**

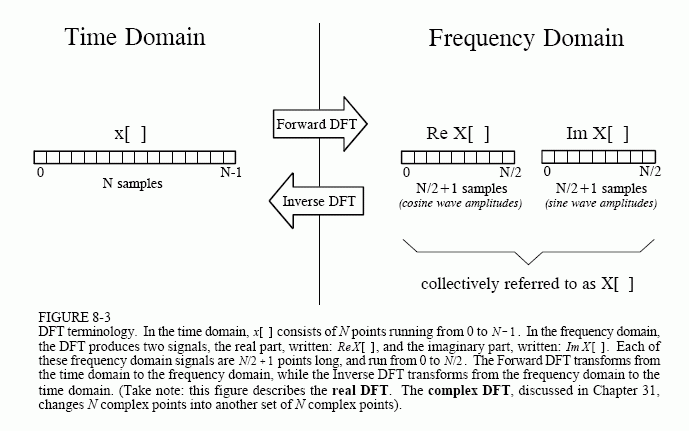
The DFT and IDFT can be explained on the board .Write the formula for DFT and IDFT and solve problems using formulas.

**3**. **Conclusion –Mind map**

a**.** Define DFT and IDFT.

b. Give the formulas for DFT and IDFT

c. Determine DFT for x (n) = {1,1,0,0}



The frequency domain contains exactly the same information as the time domain, just in a different form. If you know one domain, you can calculate the other. Given the time domain signal, the process of calculating the frequency domain is called **decomposition**, **analysis,** the **forward DFT***,* or simply, **the DFT**. If you know the frequency domain, calculation of the time domain is called **synthesis**, or the **inverse DFT**. Both synthesis and analysis can be represented in equation form and computer algorithm.The number of samples in the time domain is usually represented by the **variable *N***. While *N* can be any positive integer, a power of two is usually chosen, i.e., 128, 256, 512, 1024, etc. There are two reasons for this. First, digital data storage uses binary addressing, making powers of two a natural signal length. Second, the most efficient algorithm for calculating the DFT, the Fast Fourier Transform (FFT), usually operates with *N* that is a power of two. Typically, *N* is selected between 32 and 4096. In most cases, the samples run from 0 to *N*-1, rather than 1 to *N*.

**Session -2**

1. **Recap: DFT**

**Suggested Activity: Quiz**

We can conduct a quiz to check the learners what things they learned in previous session.

# Properties of DFT

<http://en.wikipedia.org/wiki/Discrete_Fourier_transform>

# Suggested Activity: Chalk and talk, PPT

* + - Periodicity, Linearity
    - Time & frequency shifting
    - Time reversal of a sequence

**Periodicity**

If the expression that defines the DFT is evaluated for all integers *k* instead of just for k = 0, \dots, N-1 , then the resulting infinite sequence is a periodic extension of the DFT, periodic with period *N*.

The periodicity can be shown directly from the definition:

X_{k+N} \ \stackrel{\mathrm{def}}{=} \ \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} (k+N) n} =
\sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} k n}  \underbrace{e^{-2 \pi i n}}_{1} = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} k n} = X_k. 

Similarly, it  **DF**\mathbf{F}^{-1}=\frac{1}{N}\mathbf{F}^***T**

Another way of looking at the DFT is to note that in the above discussion, the DFT can be expressed as a [Vandermonde matrix](http://en.wikipedia.org/wiki/Vandermonde_matrix" \o "Vandermonde matrix):

\mathbf{F} =
\begin{bmatrix}
 \omega_N^{0 \cdot 0}     & \omega_N^{0 \cdot 1}     & \ldots & \omega_N^{0 \cdot (N-1)}     \\
 \omega_N^{1 \cdot 0}     & \omega_N^{1 \cdot 1}     & \ldots & \omega_N^{1 \cdot (N-1)}     \\
 \vdots                   & \vdots                   & \ddots & \vdots                       \\
 \omega_N^{(N-1) \cdot 0} & \omega_N^{(N-1) \cdot 1} & \ldots & \omega_N^{(N-1) \cdot (N-1)} \\
\end{bmatrix}


where

\omega_N = e^{-2 \pi i/N}\,

is a primitive [Nth root of unity](http://en.wikipedia.org/wiki/Roots_of_unity). The inverse transform is then given by the inverse of the above matrix:

With [unitary](http://en.wikipedia.org/wiki/Unitary_operator) normalization constants 1/\sqrt{N}, the DFT becomes a [unitary transformation](http://en.wikipedia.org/wiki/Unitary_transformation), defined by a unitary matrix:

1. **Conclusion :**

**Suggested Activity: Chart**

**Session-3**

1. **Recap: DFT**

**Suggested Activity: Recall**

DFT concepts can be recall by using DFT and IDFT formulas

1. **Properties of DFT**

**Suggested Activity: Presentation, chalk and talk**

1. **Conclusion and summary:**

**Suggested Activity: Word Puzzles**

**Words:**

1. **DFT**
2. **IDFT**
3. **Periodicity**
4. **Linear**
5. **Time shift**
6. **Frequency shift**

**Session-4**

# Filtering methods based on DFT

**Suggested Activity: Introduces**

<https://engineering.purdue.edu/~ee538/DFTbasedLinearFiltering.pdf>

1. **Overlap save and add method**

**Suggested Activity: chalk and talk**

**We can explain overlap and save method on the board**

1. **Conclusion and summary:**

**Suggested Activity: Problems solved**

**Example:**

**x(n)={ 1,-1,2,-2,3,-3,4,-4} and h(n) ={-1,1}**

**Session-5**

**1. Introduction – FFT Algorithms**

**Suggested Activity: Introduces**

###### 2. Radix-2 and Radix-4 algorithms

**Suggested Activity: Writing board**

**3.** Conclusion:

**Suggested Activity: Rapid fire**

1. Define FFT.
2. Compare DFT and FFT
3. Advantages of FFT.

**Session -6**

1. **Decimation in time Algorithms:**

**Suggested Activity: Introduces**

**FFT Algorithm Decimation-in-time-Algorithm**

### [Decimation-in- Time FFT Algorithms To achieve the dramatic ...](http://www.google.co.in/url?sa=t&rct=j&q=fft%20algorithm%20decimation%20in%20time&source=web&cd=1&cad=rja&ved=0CCgQFjAA&url=http%3A%2F%2Fwww.systems.caltech.edu%2FEE%2FCourses%2FEE32b%2Fhandouts%2FFFT.pdf&ei=6Mr3UYGYCo-nrAe06oFQ&usg=AFQjCNFE3hvdGS-RKtqZxFSqqG99VgDYBg&bvm=bv.49967636,d.bmk)

###### 2. DIT-Procedure

**Suggested Activity: Writing board**

**3.** Conclusion:

**Suggested Activity: Formulas**

**Session -7**

1. **Decimation in frequency Algorithms:**

**Suggested Activity: Introduces**

The radix-2 decimation-in-frequency algorithm rearranges the [discrete Fourier transform (DFT) equation](http://cnx.org/content/m12018/latest/#eq:dft) into two parts: computation of the even-numbered discrete-frequency indices X(k) for k=[0,2,4,…,N−2] (or X(2r)  and computation of the odd-numbered indices k=[1,3,5,…,N−1] (or X(2r+1) )

X(2r)=∑n=0N−1x(n)W2rnN∑n=0N2−1x(n)W2rnN+∑n=0N2−1x(n+N2)W2r(n+N2)N∑n=0N2−1x(n)W2rnN+∑n=0N2−1x(n+N2)W2rnN1∑n=0N2−1(x(n)+x(n+N2))WrnN2DFTN2[x(n)+x(n+N2)]

X(2r+1)=∑n=0N−1x(n)W(2r+1)nN∑n=0N2−1(x(n)+WN2Nx(n+N2))W(2r+1)nN∑n=0N2−1((x(n)−x(n+N2))WnN)WrnN2DFTN2[(x(n)−x(n+N2))WnN]

The mathematical simplifications reveal that both the even-indexed and odd-indexed frequency outputs X(k) can each be computed by a length-N2 DFT. The inputs to these DFTs are sums or differences of the first and second halves of the input signal, respectively, where the input to the short DFT producing the odd-indexed frequencies is multiplied by a so-called **twiddle factor** term WkN=e−(i2πkN). This is called decimation **in frequency** because the frequency samples are computed separately in alternating groups and a **radix-2** algorithm because there are two groups. . This conversion of the full DFT into a series of shorter DFTs with a simple preprocessing step gives the decimation-in-frequency FFT its computational savings.

###### 2. DIT-Procedure

**Suggested Activity: chalk and talk**

**3.** Conclusion:

**Suggested Activity: Learner led presentation**

We can asked to any one of the learner to summarize the DIT and the procedure of DIF

**Session -8**

1. **Introduction-Linear filtering:**

**Suggested Activity: Introduces**

###### Use of FFT algorithm in linear filtering

###### Use of FFT in Linear Filtering

[A **Linear Filtering** Approach to the Computation of Discrete Fourier **...**](http://www.google.co.in/url?sa=t&rct=j&q=use%20of%20fft%20in%20linear%20filtering&source=web&cd=13&ved=0CDIQFjACOAo&url=http%3A%2F%2Fwww.ingelec.uns.edu.ar%2Fpds2803%2FMateriales%2FArticulos%2F01162132.pdf&ei=ps73UdeUEI7KrAf77IDAAg&usg=AFQjCNG49sLnyjF2c2xQwy2bpLeYpSIQDQ&bvm=bv.49967636,d.bmk)

###### Suggested Activity: presentation

**3.** Conclusion:

**Suggested Activity: Rapid fire**

Questions:

1. What is linear filtering?
2. What are the convolution sectioned methods?
3. Differentiate overlap add and save method.
4. Give the procedure for convolution method.

**Session -9**

1. **Introduction-DCT :**

**Suggested Activity: Introduces and questions**

<http://www.cs.cf.ac.uk/Dave/Multimedia/node231.html>

###### 2. DCT- Formulas and definitions

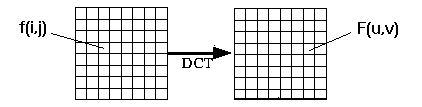
**Suggested Activity: Chalk and talk**

**3.** Conclusion:

**Suggested Activity: Formulas**

## The Discrete Cosine Transform (DCT)

   The discrete cosine transform (DCT) helps separate the image into parts (or spectral sub-bands) of differing importance (with respect to the image's visual quality). The DCT is similar to the discrete Fourier transform: it transforms a signal or image from the spatial domain to the frequency domain (Fig [7.8](http://www.cs.cf.ac.uk/Dave/Multimedia/node231.html#DCTenc)).



**DCT Encoding**

The general equation for a 1D (*N* data items) DCT is defined by the following equation:

\begin{displaymath}
F(u) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1}
\Lambda(i).cos\left[
\frac{\pi.u}{2.N}(2i+1)
\right]f(i)\end{displaymath}

and the corresponding **inverse** 1D DCT transform is simple *F-1*(*u*), i.e.:

where

\begin{displaymath}
\Lambda(i) = \left\{ \begin{array}
{ll} \frac{1}{\sqrt{2}} & {\rm
for}
\xi = 0\ 1 & {\rm otherwise}\end{array} \right.\end{displaymath}

The general equation for a 2D (*N* by *M* image) DCT is defined by the following equation:

\begin{displaymath}
F(u,v) = \left(\frac{2}{N}\right)^{\frac{1}{2}}
\left(\frac{...
 ...}(2i+1)
\right]cos\left[ \frac{\pi.v}{2.M}(2j+1) \right].f(i,j)\end{displaymath}

and the corresponding **inverse** 2D DCT transform is simple *F-1*(*u*,*v*),

\begin{displaymath}
\Lambda(\xi) = \left\{ \begin{array}
{ll} \frac{1}{\sqrt{2}} & {\rm
for}
\xi = 0 \ 1 & {\rm otherwise}\end{array} \right.\end{displaymath}