Signal, Image and Data Processing
(236200)

Tutorial 8

Processing of Discrete Signals
Consider the discrete signal

\[ \overline{\varphi} = \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_N \end{bmatrix} \]

which is an \( N \)-length column vector of real values, i.e., \( \overline{\varphi} \in \mathbb{R}^N \).

Obviously, we can express \( \overline{\varphi} \) also as

\[ \overline{\varphi} = \sum_{k=1}^{N} \varphi_k \overline{\beta}^S_k = \varphi_1 \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \varphi_2 \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} + \cdots + \varphi_N \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \]

where \( \{\overline{\beta}^S_k\}_{k=1}^{N} \) is the standard basis for \( \mathbb{R}^N \).
Discrete Signals: Representation in the Standard Basis

The standard basis vector is defined as \( \overline{\beta}_k^S = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \) \( \Rightarrow \) the \( k^{th} \) component

and accordingly, \( \langle \overline{\beta}_k^S, \overline{\beta}_l^S \rangle = \overline{\beta}_k^S \overline{\beta}_l^S = \delta_{kl} = \begin{cases} 1 & \text{if } k = l \\ 0 & \text{if } k \neq l \end{cases} \)

Moreover, arranging \( \{\overline{\beta}_k^S\}_{k=1}^N \) in a matrix shows that \( \begin{bmatrix} \overline{\beta}_1^S & \overline{\beta}_2^S & \ldots & \overline{\beta}_N^S \end{bmatrix} = I_{N \times N} \)

letting us to rewrite the standard representation of the signal as

\[
\overline{\varphi} = \sum_{k=1}^{N} \varphi_k \overline{\beta}_k^S = \begin{bmatrix} \overline{\beta}_1^S & \overline{\beta}_2^S & \ldots & \overline{\beta}_N^S \end{bmatrix} \overline{\varphi} = I_{N \times N} \overline{\varphi}
\]

“coefficients” vector
Discrete Signals: Representation using Unitary Matrices

Consider the $N \times N$ unitary matrix $\mathbf{U}$.

By definition, $\mathbf{U} \mathbf{U}^* = \mathbf{U}^* \mathbf{U} = \mathbf{I}_{N \times N}$ ,

Hence, its columns $\{\mathbf{u}_k\}_{k=1}^N$ are a basis for $\mathbb{R}^N$ or in the complex-valued case for $\mathbb{C}^N$.

Accordingly, we can write

$$
\bar{\varphi} = \mathbf{I}_{N \times N} \bar{\varphi} = \mathbf{U} \mathbf{U}^* \bar{\varphi} = \mathbf{U} \begin{bmatrix}
\langle \mathbf{u}_1, \bar{\varphi} \rangle \\
\langle \mathbf{u}_2, \bar{\varphi} \rangle \\
\vdots \\
\langle \mathbf{u}_N, \bar{\varphi} \rangle
\end{bmatrix} = \sum_{k=1}^{N} \langle \mathbf{u}_k, \bar{\varphi} \rangle \mathbf{u}_k
$$

We define the $k^{th}$ representation coefficient as $\varphi_k^{(U)} \triangleq \langle \mathbf{u}_k, \bar{\varphi} \rangle$.

The $K$-term approximation with respect to $\mathbf{U}$ is

$$
\hat{\varphi}_K^{(U)} = \sum_{k=1}^{K} \langle \mathbf{u}_k, \bar{\varphi} \rangle \mathbf{u}_k
$$

Here the first $K$ coefficients are taken, therefore, the approximation is not necessarily optimal.
Discrete Signals: Representation using Unitary Matrices

The total energy (which is the squared-length) of $\phi$ is

$$\langle \phi, \phi \rangle = \phi^* \phi = \sum_{k=1}^{N} \varphi_k^2$$

$$\langle \phi, \phi \rangle = \phi^* \phi = \phi^* I_{N \times N} \phi = \phi^* UU^* \phi = (U^* \phi)^* U^* \phi = \phi^{(u)^*} \phi^{(u)} = \langle \phi^{(u)}, \phi^{(u)} \rangle$$

The total energy of the representation is preserved under a unitary transformation!

where

$$\langle \phi^{(u)}, \phi^{(u)} \rangle = \begin{bmatrix} \langle \bar{u}_1, \phi \rangle \\ \langle \bar{u}_2, \phi \rangle \\ \vdots \\ \langle \bar{u}_N, \phi \rangle \end{bmatrix}^* \begin{bmatrix} \langle \bar{u}_1, \phi \rangle \\ \langle \bar{u}_2, \phi \rangle \\ \vdots \\ \langle \bar{u}_N, \phi \rangle \end{bmatrix} = \sum_{k=1}^{N} \langle \bar{u}_k, \phi \rangle^2$$

Note however that if one considers $K$-term approximations for $K < N$, then $\sum_{k=1}^{K} \langle \bar{u}_k, \phi \rangle^2$ may differ from $\sum_{k=1}^{N} \varphi_k^2$. 

236200, CS Department, Technion
Consider the $K$-term approximation

$$\hat{\phi}_K^{(u)} = \sum_{k=1}^{K} \langle \vec{u}_k, \phi \rangle \vec{u}_k$$

The corresponding error vector is

$$\vec{e}_K = \vec{\phi} - \hat{\phi}_K^{(u)} = \sum_{k=1}^{N} \langle \vec{u}_k, \phi \rangle \vec{u}_k - \sum_{k=1}^{K} \langle \vec{u}_k, \phi \rangle \vec{u}_k = \sum_{k=K+1}^{N} \langle \vec{u}_k, \phi \rangle \vec{u}_k = \sum_{k=K+1}^{N} \phi_k^{(u)} \vec{u}_k$$

and the respective squared-error is the squared-length of the vector $\vec{e}_K$:

$$\langle \vec{e}_K, \vec{e}_K \rangle = \vec{e}_K^* \vec{e}_K = \left( \sum_{k=K+1}^{N} \phi_k^{(u)} \vec{u}_k \right)^* \left( \sum_{k=K+1}^{N} \phi_k^{(u)} \vec{u}_k \right) = \sum_{k=K+1}^{N} |\phi_k^{(u)}|^2 \vec{u}_k^* \vec{u}_k = \sum_{k=K+1}^{N} |\phi_k^{(u)}|^2$$

Using the orthonormality of $\{\vec{u}_k\}_{k=1}^{N}$.
Operations on Discrete Signals

An operator/system that processes or improves the signal. Examples: noise cleaning, sharpening, etc.

The output signal is defined as $\vec{\phi}^{out} \triangleq \mathcal{H}\{\vec{\phi}^{in}\}$
Linear Systems

Linearity:

\[ \mathcal{H}\{a_1 \varphi_1 + a_2 \varphi_2\} = a_1 \cdot \mathcal{H}\{\varphi_1\} + a_2 \cdot \mathcal{H}\{\varphi_2\} \]

where \( a_1 \) and \( a_2 \) are real or complex scalars.

Accordingly, for a linear operator we can write:

\[
\mathcal{H}\{\varphi\} = \mathcal{H}\left\{ \sum_{k=1}^{N} \varphi_k \bar{\beta}_k^S \right\} = \sum_{k=1}^{N} \varphi_k \mathcal{H}\{\bar{\beta}_k^S\} = \begin{bmatrix} \mathcal{H}\{\bar{\beta}_1^S\} & \mathcal{H}\{\bar{\beta}_2^S\} & \cdots & \mathcal{H}\{\bar{\beta}_N^S\} \end{bmatrix} \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_N \end{bmatrix} = \varphi_{out}
\]
Linear Systems

\[ \mathcal{H}\{\varphi\} = \mathcal{H}\left\{\sum_{k=1}^{N} \varphi_k \beta_k^S\right\} = \sum_{k=1}^{N} \varphi_k \mathcal{H}\{\beta_k^S\} = \begin{bmatrix} \mathcal{H}\{\beta_1^S\} & \mathcal{H}\{\beta_2^S\} & \ldots & \mathcal{H}\{\beta_N^S\} \end{bmatrix}\begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_N \end{bmatrix} = \overline{\varphi}^{out} \]

A matrix defining the linear operator (any discrete linear operator can be expressed as a matrix!)

We define \( \overline{h}_k \triangleq \mathcal{H}\{\beta_k^S\} \) for \( k = 1, \ldots, N \) and then we can write

\[ \overline{\varphi}^{out} = \begin{bmatrix} \overline{h}_1 & \overline{h}_2 & \ldots & \overline{h}_N \end{bmatrix} \overline{\varphi}^{in} \]

where \( \overline{\varphi}^{in} \) was denoted above also as \( \overline{\varphi} \).
A **cyclical shift** of a vector $\vec{\varphi}$ is applied by the operator $\mathcal{T}\{\cdot\}$ and defined as

$$
\mathcal{T}\left\{\begin{bmatrix}
\varphi_1 \\
\varphi_2 \\
\varphi_3 \\
\vdots \\
\varphi_N
\end{bmatrix}\right\} = \begin{bmatrix}
\varphi_N \\
\varphi_1 \\
\varphi_2 \\
\vdots \\
\varphi_{N-1}
\end{bmatrix}
$$

The operator $\mathcal{T}\{\cdot\}$ is linear (prove!), hence, it is described by the matrix

$$
\mathbf{T} = \begin{bmatrix}
\mathcal{T}\{\vec{\beta}_1^S\} & \mathcal{T}\{\vec{\beta}_2^S\} & \cdots & \mathcal{T}\{\vec{\beta}_N^S\}
\end{bmatrix} = \begin{bmatrix}
\mathcal{T}\left\{\begin{bmatrix}1 \\
0 \\
0 \\
\vdots \\
0
\end{bmatrix}\right\} & \mathcal{T}\left\{\begin{bmatrix}0 \\
1 \\
0 \\
\vdots \\
0
\end{bmatrix}\right\} & \cdots & \mathcal{T}\left\{\begin{bmatrix}0 \\
0 \\
0 \\
\vdots \\
1
\end{bmatrix}\right\}
\end{bmatrix} = \begin{bmatrix}0 & 0 & 1 \\
1 & 0 & 0 \\
0 & 1 & \vdots & \vdots \\
\vdots & \vdots & \ddots & \ddots \\
0 & 0 & \cdots & 0
\end{bmatrix}
$$
Shift-Invariant Systems

An operator \( \mathcal{H}\{\cdot\} \) is shift-invariant if it commutes with the cyclical shift operator \( \mathcal{T}\{\cdot\} \), namely,

**Shift invariance:** \[ \mathcal{H}\left\{ \mathcal{T}\{\overline{\varphi^{\text{in}}} \} \right\} = \mathcal{T}\left\{ \mathcal{H}\{\overline{\varphi^{\text{in}}} \} \right\} \quad \text{for any } \overline{\varphi^{\text{in}}} \]

This means that shifting the input signal to a shift-invariant system \( \mathcal{H} \) will result in the same output as if the input to \( \mathcal{H} \) was non-shifted and the respective output \( \mathcal{H}\{\overline{\varphi^{\text{in}}} \} \) was shifted.

We often denote a linear shift-invariant (LSI) system as \( \mathcal{H}_{\text{LSI}}\{\cdot\} \) or the matrix \( \mathbf{H}_{\text{LSI}} \).
Shift-Invariant Systems

We further express the matrix of a linear shift-invariant operator $\mathcal{H}_{LSI}\{\cdot\}$ as:

$$H_{LSI} = \begin{bmatrix}
\mathcal{H}_{LSI}\{\beta_1^S\} & \mathcal{H}_{LSI}\{\beta_2^S\} & \ldots & \mathcal{H}_{LSI}\{\beta_N^S\}
\end{bmatrix}$$

Note that $\mathcal{H}_{LSI}\{\cdot\}$ is applied here on shifts of the same column! Therefore the columns in the resulting matrix are cyclically shifted versions of the first column! A circulant matrix!
Circulant Matrices: Properties

Property #1:

Two circulant matrices, $H^{(1)}_{LSI}$ and $H^{(2)}_{LSI}$, commute: $H^{(1)}_{LSI}H^{(2)}_{LSI} = H^{(2)}_{LSI}H^{(1)}_{LSI}$

Moreover, $H^{(1)}_{LSI}H^{(2)}_{LSI}$ is also a circulant matrix.

(You will prove this property in homework #4)

Property #2:

All the circulant matrices have the same eigenvectors!

The $[DFT]^*$ matrix diagonalizes any circulant matrix!

(A detailed proof is available in the handwritten lecture notes #9 available on the course website).
Circulant Matrices: Diagonalization via \([DFT]^*\)

Consider a circulant matrix \(H_{LSI}\) that, by the second property, \(H_{LSI}\) is diagonalized by the \([DFT]^*\) matrix denoted here as \(\beta \triangleq [DFT]^*:\)

\[
\beta^* H_{LSI} \beta = \Lambda_H
\]

where \(\Lambda_H\) is a diagonal matrix with diagonal values \(\{\lambda_k^H\}_{k=1}^N\)

or equivalently

\[
H_{LSI} = \beta \Lambda_H \beta^*
\]

or equivalently

\[
H_{LSI} \bar{\beta}_k = \lambda_k^H \bar{\beta}_k \quad \text{for} \quad k = 1, \ldots, N
\]

where \(\{\bar{\beta}_k\}_{k=1}^N\) are the columns of the \(\beta\) matrix.
Circulant Matrices: Diagonalization via $[DFT]^*$

$$H_{LSI} \bar{\beta}_k = \lambda_k^H \bar{\beta}_k \quad \text{for } k = 1, \ldots, N$$

Moreover, it can be shown that the eigenvalues $\{\lambda_k^H\}_{k=1}^N$ can be computed as:

$$\lambda_k^H = \langle \text{first row of } H_{LSI}, \ k^{th} \text{ row of } [DFT]^* \rangle \quad \text{for } k = 1, \ldots, N$$

$$= \langle \begin{bmatrix} h_0 & h_{N-1} & h_{N-2} & \cdots & h_1 \end{bmatrix}, \begin{bmatrix} W^{k\cdot0} & W^{k\cdot1} & W^{k\cdot2} & \cdots & W^{k\cdot(N-1)} \end{bmatrix} \rangle$$

or in a matrix form as

$$\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \vdots \\ \lambda_N \end{bmatrix} = [DFT]^* \begin{bmatrix} h_0 \\ h_{N-1} \\ h_{N-2} \\ \vdots \\ h_1 \end{bmatrix}$$

(Proofs for the results in this slide are given in the handwritten lecture notes #9 available on the course website).
Applying LSI Operators in the DFT Domain

A linear shift-invariant system $H_{LSI}$ processes the input signal $\phi^{in}$ to provide the system output: $\phi^{out} = H_{LSI} \phi^{in}$

We can utilize the diagonalized form of $H_{LSI} = \beta \Lambda_H \beta^*$ and write:

$$\phi^{out} = [DFT]^* \Lambda_H [DFT] \phi^{in}$$

meaning that

$$[DFT] \phi^{out} = [DFT][DFT]^* \Lambda_H [DFT] \phi^{in}$$

$$[DFT] \phi^{out} = \Lambda_H ([DFT] \phi^{in})$$

236200, CS Department, Technion
Applying LSI Operators in the DFT Domain

Noticing that

\[
[DFT][\phi]_{out} = \Lambda_H([DFT][\phi]_{in})
\]

is actually

\[
[\phi]_{out,DFT} = \begin{bmatrix}
\lambda_1^H & \cdots & \\
\vdots & \ddots & \\
\cdots & & \lambda_N^H
\end{bmatrix}
[\phi]_{in,DFT} = \begin{bmatrix}
\lambda_1 \phi_1^{in,DFT} \\
\lambda_2 \phi_2^{in,DFT} \\
\vdots \\
\lambda_N \phi_N^{in,DFT}
\end{bmatrix}
\]

The computation in the DFT-domain is a componentwise product of the eigenvalues of $H_{LSI}$ and the DFT coefficients of $[\phi]_{in}$.

This is much simpler than the computation $[\phi]_{out} = H_{LSI} [\phi]_{in}$ in the signal domain.

(The overall computational benefit relies also on existing fast implementations of DFT computations).
Discrete-Time Systems: Examples

Consider discrete-time signals and systems defined for \( n \in \{0,1,\ldots,N-1\} \) for some integer \( N > 1 \).

Here \( x[n] \) is the input signal and \( y[n] \) is the output signal of the system. These signals also take the following vector form of

\[
\mathbf{x} = \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix} \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} y[0] \\ y[1] \\ \vdots \\ y[N-1] \end{bmatrix}
\]

If \( \mathbf{y} = \mathbf{Hx} \) holds for some \( N \times N \) matrix \( \mathbf{H} \), the system is linear.

In case that \( \mathbf{H} \) is also a circulant matrix, then the linear system is also shift-invariant.
Determine whether the following system is linear and/or shift-invariant.

**System #1:**

\[ y[n] = \begin{cases} x[n] & , n \text{(mod 2)} = 0 \\ 0 & , else \end{cases} \]

Clearly this is a **linear system** (prove it by yourself).

It can be formulated in a matrix form as

\[
\begin{bmatrix}
    y[0] \\
    y[1] \\
    \vdots \\
    y[N - 1]
\end{bmatrix} =
\begin{bmatrix}
    1 & 0 & & \\
    0 & 1 & & \\
    & & \ddots & \\
    & & & 0
\end{bmatrix}
\begin{bmatrix}
    x[0] \\
    x[1] \\
    \vdots \\
    x[N - 1]
\end{bmatrix}
\]

\( \mathbf{H} \) is a diagonal matrix with alternating values of ones and zeros, hence, it is not circulant → the system is **not shift invariant**.
Determine whether the following system is linear and/or shift-invariant.

System #2:

\[ y[n] = x[n + 1 \mod N] - x[n] \]

This is a linear system (prove it by yourself).
It can be formulated in a matrix form as

\[
\begin{bmatrix}
    y[0] \\
    y[1] \\
    \vdots \\
    y[N-1]
\end{bmatrix} =
\begin{bmatrix}
    -1 & 1 & 1 & \cdots & 1 \\
    -1 & -1 & 1 & \cdots & 1 \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    1 & 1 & 1 & \cdots & -1
\end{bmatrix}
\begin{bmatrix}
    x[0] \\
    x[1] \\
    \vdots \\
    x[N-1]
\end{bmatrix}
\]

\( H \) has two fixed-valued diagonals forming a circulant matrix structure → the system is shift invariant.
Determine whether the following system is linear and/or shift-invariant.

**System #3:**

\[ y[n] = \sum_{k=0}^{n} x[k] \]

This is a **linear system** (prove it by yourself).

It can be formulated in a matrix form as

\[
\begin{bmatrix}
  y[0] \\
  y[1] \\
  \vdots \\
  y[N - 1]
\end{bmatrix} =
\begin{bmatrix}
  1 & 0 & 0 & 0 & \cdots & 0 \\
  1 & 1 & 0 & 0 & \cdots & 0 \\
  1 & 1 & 1 & 0 & \cdots & 0 \\
 1 & 1 & 1 & 1 & 0 & 0 \\
  \vdots \\
  1 & 1 & 1 & \cdots & 1 & 1
\end{bmatrix}
\begin{bmatrix}
  x[0] \\
  x[1] \\
  \vdots \\
  x[N - 1]
\end{bmatrix}
\]

\( \mathbf{H} \) is a lower triangular matrix of ones, hence, it is not a circulant matrix → the system is **not shift invariant**.
Determine whether the following system is linear and/or shift-invariant.

**System #4:**

\[ y[n] = \sum_{k=0}^{M} x[n - k \pmod{N}] \]

This is a **linear system** (prove it by yourself).

It can be formulated in a matrix form as (example for \( M = 3 \))

\[
\begin{bmatrix}
 y[0] \\
 y[1] \\
 \vdots \\
 y[N-1]
\end{bmatrix} =
\begin{bmatrix}
 1 & 0 & \cdots & 0 & 1 & 1 \\
 1 & 1 & 0 & \cdots & 0 & 1 \\
 1 & 1 & 1 & 0 & \cdots & 0 \\
 0 & 1 & 1 & 1 & 0 & \cdots \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 0 & \cdots & 0 & 1 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
 x[0] \\
 x[1] \\
 \vdots \\
 x[N-1]
\end{bmatrix}
\]

\( H \) is a circulant matrix
→ the system is **shift invariant**.