Earth 460 Classnotes (Set 6)

II.) Time Series Analysis – Discrete Signals

C.) z-Transform

1.) Consider a uniformly sampled time signal defined in mathematical terms as

$$g_s(t) = \sum_{n=-\infty}^{+\infty} g(n\Delta t)\,\delta(t - n\Delta t) = \sum_{n=-\infty}^{+\infty} g_n\;\delta(t - n\Delta t) \text{ where } g_n = g(n\Delta t)$$

a.) The discretely sampled information can be represented as a sequence or row vector where the element indices denote the time of the samples.

$$\{g_k\}_{k=-\infty}^{+\infty} = \{\cdots, g_{-2}, g_{-1}, g_0, g_1, g_2, \cdots\} \text{ or } \mathbf{g} = (\cdots, g_{-2}, g_{-1}, g_0, g_1, g_2, \cdots)$$

b.) For finite length signals, the sequence/row vector is truncated to its significant elements. For example, suppose we have a causal signal that is five terms in length:

$$\mathbf{g} = (\cdots, g_{-2}, g_{-1}, g_0, g_1, g_2, g_3, g_4, g_5, g_6, \cdots) = (\cdots, 0, 0, -1, 4, 0, -1, 3, 0, 0, \cdots)$$

It can be completely summarized by

$$\mathbf{g} = (g_0, g_1, g_2, g_3, g_4) = (-1, 4, 0, -1, 3)$$

c.) This uniformly sampled time signal is commonly referred to as a *time series*.

2.) The Fourier transform of uniformly sampled time signal is

$$G_s(f) = \int_{-\infty}^{+\infty} g_s(t)e^{-2\pi i f t}\,dt = \int_{-\infty}^{+\infty}\left[\sum_{n=-\infty}^{+\infty} g_n\;\delta(t - n\Delta t)\right]e^{-2\pi i f t}\,dt$$

$$= \sum_{n=-\infty}^{+\infty} g_n\left[\int_{-\infty}^{+\infty}\delta(t - n\Delta t)e^{-2\pi i f t}\,dt\right]$$

Using the sifting property of the δ function, $G_s(f) = \sum_{n=-\infty}^{+\infty} g_n\; e^{-2\pi i n\Delta t f}$

3.) If we use $z = e^{-2\pi i \Delta t f}$, then we can rewrite the last result as

$$G_s(f) = \sum_{n=-\infty}^{+\infty} g_n\, z^n$$

4.) This result leads to the definition of the z-transform for a time series:

$$\mathbf{g} = \left(\cdots, g_{n-1}, g_n, g_{n+1}, \cdots\right) \Leftrightarrow G(z) = \sum_{n=-\infty}^{+\infty} g_n\, z^n = \cdots + g_{-1}\, z^{-1} + g_0\, z^0 + g_1\, z^1 + \cdots$$

a.) The form of the z-transform is a polynomial in powers of z. The coefficient of z^n is the time series element at the corresponding time $n\,\Delta t$.

Examples:

1.) Given $\mathbf{g} = \left(g_{-1}, g_0, g_1, g_2\right) = \left(-1, 0, 2, 4\right)$ →

$$\mathbf{g} \Leftrightarrow G(z) = g_{-1}\, z^{-1} + g_0\, z^0 + g_1\, z^1 + g_2\, z^2 = -1 z^{-1} + 0\, z^0 + 2\, z^1 + 4\, z^2$$

2.) Given $\mathbf{g} = \left(g_0, g_1, g_2, g_3, \cdots\right) = \left(1, a^{-2}, a^{-4}, a^{-6}, \cdots\right)$ →

$$\mathbf{g} \Leftrightarrow G(z) = g_0\, z^0 + g_1\, z^1 + g_2\, z^2 + g_3\, z^3 + \cdots = 1 z^0 + a^{-2}\, z^1 + a^{-4}\, z^2 + a^{-6}\, z^3 + \cdots$$

b.) The inverse z-transform requires reconstruction of the time series elements from the z polynomial coefficients and assigning them to the appropriate sampling times.

Examples:

1.) Given

$$G(z) = 2\, z^{-2} + 3\, z^{-1} + 1 z^0 - 4\, z^1 - 3\, z^2 = g_{-2}\, z^{-2} + g_{-1}\, z^{-1} + g_0\, z^0 + g_1\, z^1 + g_2\, z^2 \rightarrow$$

$$\mathbf{g} = \left(g_{-2}, g_{-1}, g_0, g_1, g_2\right) = \left(2, 3, 1, -4, -3\right) \Leftrightarrow G(z)$$

2.) Given $G(z) = \cdots + a^2\, z^{-3} + a\, z^{-2} + 1 z^{-1} = \cdots + g_{-3}\, z^{-3} + g_{-2}\, z^{-2} + g_{-1}\, z^{-1}$ →

$$\mathbf{g} = \left(\cdots, g_{-3}, g_{-2}, g_{-1}\right) = \left(\cdots, a^2, a, 1\right) \Leftrightarrow G(z)$$

c.) A unit impulse located at $t = n\,\Delta t$ has the z-transform pair

$\boldsymbol{\delta}_{\mathbf{n}} = \left(\cdots, g_{n-1}, g_n, g_{n+1}, \cdots\right) = \left(\cdots, 0, 1, 0, \cdots\right) \Leftrightarrow z^n$. In particular, $\boldsymbol{\delta}_0 \Leftrightarrow 1$.

D.) Convolution of Discrete Signals

1.) Definition of convolution of two discrete time series is

$$\mathbf{d} = \mathbf{g} * \mathbf{h} \rightarrow d_k = \sum_{n=-\infty}^{n=+\infty} g_n\, h_{k-n} = \sum_{n=-\infty}^{n=+\infty} h_n\, g_{k-n}$$

2.) If $\mathbf{g}$ and $\mathbf{h}$ are finite length signals, then the summation can be done in long hand form for each output element.

Example:

$\mathbf{g} = (g_{-1}, g_0, g_1) = (\,1, -1, 2)$ and $\mathbf{h} = (h_1, h_2, h_3) = (\,3, -1, 2)$

$$\mathbf{d} = \mathbf{g} * \mathbf{h} \rightarrow d_k = \sum_{n=-\infty}^{+\infty} g_n\, h_{k-n}, \quad k = 0, 1, 2, 3, 4$$

$$k = 0: \quad d_k = \sum_{n=-1}^{+1} g_n\, h_{k-n} = g_{-1} h_1 + g_0 h_0 + g_1 h_{-1} = 1\cdot 3 - 1\cdot 0 + 2\cdot 0 = 3$$

$$k = 1: \quad d_k = \sum_{n=-1}^{+1} g_n\, h_{k-n} = g_{-1} h_2 + g_0 h_1 + g_1 h_0 = -1\cdot 1 - 1\cdot 3 + 2\cdot 0 = -4$$

$$k = 2: \quad d_k = \sum_{n=-1}^{+1} g_n\, h_{k-n} = g_{-1} h_3 + g_0 h_2 + g_1 h_1 = 1\cdot 2 + 1\cdot 1 + 2\cdot 3 = 9$$

$$k = 3: \quad d_k = \sum_{n=-1}^{+1} g_n\, h_{k-n} = g_{-1} h_4 + g_0 h_3 + g_1 h_2 = 1\cdot 0 - 1\cdot 2 - 2\cdot 1 = -4$$

$$k = 4: \quad d_k = \sum_{n=-1}^{+1} g_n\, h_{k-n} = g_{-1} h_5 + g_0 h_4 + g_1 h_3 = 1\cdot 0 - 1\cdot 0 + 2\cdot 2 = 4$$

$$\therefore \mathbf{d} = (d_0, d_1, d_2, d_3, d_4) = (\,3, -4, 9, -4, 4)$$

3.) Convolution can be performed using a mechanical procedure.

a.) One of the time series is folded (i.e.,reversed).

b.) The folded time series is completely moved to the left end of the unfolded time series. It is then shifted right along one position at a time past the unfolded time series.

c.) For each shifted position, the products for each column are summed to obtain the corresponding element of the convolved signal. The time position of convolved element is equal to the sum of indices along any column.

$$
\begin{array}{cccccc}
 & & & (\,g_{-1} & g_0 & g_1\,) \\
 & & & (\,1 & -1 & 2\,) \\
 & (\,2 & -1 & 3\,) & & \\
 & (\,h_3 & h_2 & h_1\,) & & \\
\hline
d_0 = & & & 3 & &
\end{array}
$$

$$
\begin{array}{ccccccc}
 & & (\,g_{-1} & g_0 & g_1\,) & & \\
 & & (\,1 & -1 & 2\,) & & \\
 & (\,2 & -1 & 3\,) & & & \\
 & (\,h_3 & h_2 & h_1\,) & & & \\
\hline
d_1 = & & -1 & -3 & = -4 & &
\end{array}
$$

$$
\begin{array}{cccccc}
 & (\,g_{-1} & g_0 & g_1\,) & \\
 & (\,1 & -1 & 2\,) & \\
 & (\,2 & -1 & 3\,) & \\
 & (\,h_3 & h_2 & h_1\,) & \\
\hline
d_2 = & 2 & 1 & 6 & = 9
\end{array}
$$

$$
\begin{array}{cccccc}
 & (\,g_{-1} & g_0 & g_1\,) & & \\
 & (\,1 & -1 & 2\,) & & \\
 & & (\,2 & -1 & 3\,) & \\
 & & (\,h_3 & h_2 & h_1\,) & \\
\hline
d_3 = & & -2 & -2 & = -4 &
\end{array}
$$

$$
\begin{array}{cccccc}
 & (\,g_{-1} & g_0 & g_1\,) & & \\
 & (\,1 & -1 & 2\,) & & \\
 & & & (\,2 & -1 & 3\,) \\
 & & & (\,h_3 & h_2 & h_1\,) \\
\hline
d_4 = & & & 4 & &
\end{array}
$$

4.) Convolution can be done using the z-transforms (probably the easiest method to compute the convolution of two discrete signals.

a.) Similar to the Fourier transform, convolution is converted to multiplication with the z-transform. Given $\mathbf{g} \Leftrightarrow G(z)$ and $\mathbf{h} \Leftrightarrow H(z)$, then $\mathbf{g} * \mathbf{h} \Leftrightarrow G(z) \cdot H(z)$

b.) This procedure is equivalent to polynomial multiplication.

Example:

$\mathbf{g} = (g_{-1}, g_0, g_1) = (\ 1, -1, 2) \Leftrightarrow G(z) = 1z^{-1} - 1z^0 + 2z^1$ and

$\mathbf{h} = (h_1, h_2, h_3) = (\ 3, -1, 2) \Leftrightarrow H(z) = 3z^1 - 1z^2 + 2z^3$

$\mathbf{d} = \mathbf{g} * \mathbf{h} \Leftrightarrow D(z) = G(z) \cdot H(z)$

$$
\begin{array}{rrrrrrl}
 & 1\,z^{-1} & -\,1\,z^{0} & +\,2\,z^{1} & & & (= G(z)) \\
\times & 3\,z^{1} & -\,1\,z^{2} & +\,2\,z^{3} & & & (= H(z)) \\
\hline
 & 3\,z^{0} & -\,3\,z^{1} & +\,6\,z^{2} & & & \\
 & & -\,1\,z^{1} & +\,1\,z^{2} & -\,2\,z^{3} & & \\
 & & & 2\,z^{2} & -\,2\,z^{3} & +\,4\,z^{4} & \\
\hline
 & 3\,z^{0} & -\,4\,z^{1} & +\,9\,z^{2} & -\,4\,z^{3} & +\,4\,z^{4} & (= D(z))
\end{array}
$$

$\mathbf{d} = (d_0, d_1, d_2, d_3, d_4) = (\ 3, -4, 9, -4, 4) \Leftrightarrow D(z) = 3z^0 - 4z^1 + 9z^2 - 4z^3 + 4z^4$

E.) Deconvolution of Discrete Signals

1.) Design of deconvolution operators / inverse filters for discrete time series is easily accomplished using z-transforms.

2.) If $\mathbf{g}$ is the discrete impulse response of a filter, the impulse response of its inverse filter $\mathbf{g}'$ has the following property: $\mathbf{g} * \mathbf{g}' = \boldsymbol{\delta}_\mathbf{0}$.

3.) Given the z-transform pairs $\mathbf{g} \Leftrightarrow G(z)$ and $\mathbf{g}' \Leftrightarrow G'(z)$, then $\mathbf{g} * \mathbf{g}' = \boldsymbol{\delta}_\mathbf{0} \Leftrightarrow G(z) \cdot G'(z) = 1$. It follows that $G'(z) = 1/G(z)$.

4.) Hence, the z-transform of an inverse filter can be determined using polynomial division. . (Manually, this means long division!)

5.) Consider the simple, two sample time series

$$\mathbf{g} = (g_0, g_1) = (1, k) \Leftrightarrow G(z) = 1z^0 + kz^1 = 1 + kz$$

a.) The polynomial division can be performed in two different ways. It can be done in a manner that produces progressively larger powers of z or it can be done such that progressively smaller powers of z are obtained. These two techniques correspond to constructing impulse responses for inverse filters that extend into future or past time, respectively.

For increasing powers of z:

$$
\begin{array}{r|l}
 & 1 - kz + k^2z^2 - k^3z^3 + \ldots\ldots \\ \hline
1 + kz & 1 \\
 & \underline{-1 - kz} \\
 & \quad -kz \\
 & \quad \underline{\ \ kz + k^2z^2} \\
 & \qquad\quad k^2z^2 \\
 & \qquad\quad \underline{-k^2z^2 - k^3z^3} \\
 & \qquad\qquad\quad -k^3z^3 \\
 & \qquad\qquad\quad \underline{\ \ k^3z^3 + k^4z^4} \\
 & \qquad\qquad\qquad\quad \ldots\ldots
\end{array}
$$

$$\mathbf{g}' = (g'_0, g'_1, g'_2, g'_3, \cdots) = (1, -k, k^2, -k^3, \cdots) \Leftrightarrow G'(z) = 1 - kz + k^2z^2 - k^3z^3 + \cdots$$

(Note: this version of $\mathbf{g}'$ is purely causal)

For decreasing powers of z:

$$
\begin{array}{r|l}
 & k^{-1}z^{-1} - k^{-2}z^{-2} + k^{-3}z^{-3} - \ldots\ldots \\ \hline
kz+1 & 1 \\
 & \underline{-1 \qquad -k^{-1}z^{-1}} \\
 & \qquad\quad -k^{-1}z^{-1} \\
 & \qquad\quad \underline{k^{-1}z^{-1} + k^{-2}z^{-2}} \\
 & \qquad\qquad\qquad k^{-2}z^{-2} \\
 & \qquad\qquad\quad \underline{-k^{-2}z^{-2} - k^{-3}z^{-3}} \\
 & \qquad\qquad\qquad\qquad \ldots\ldots
\end{array}
$$

$$\mathbf{g}' = \left(\cdots, g'_{-3}, g'_{-2}, g'_{-1}\right) = \left(\cdots, k^{-3}, -k^{-2}, k^{-1}\right) \Leftrightarrow G'(z) = \cdots + k^{-3}z^{-3} - k^{-2}z^{-2} + k^{-1}z^{-1}$$

(Note: this version of $\mathbf{g}'$ is purely non-causal)

b.) The choice of time direction is based on a number of factors (e.g., causality). In particular, stability of the inverse filter is important.

For our example, consider $k = 1/2$:

$\mathbf{g}' = \left(g'_0, g'_1, g'_2, g'_3, \cdots\right) = \left(1, -1/2, 1/4, -1/8, \cdots\right)$ is stable.

$\mathbf{g}' = \left(\cdots, g'_{-4}, g'_{-3}, g'_{-2}, g'_{-1}\right) = \left(\cdots, -16, 8, -4, 2\right)$ is unstable

6.) It is impractical to use an inverse filter with an infinitely long sequence of terms. It is necessary to truncate its impulse response at some point where a reasonable approximation is obtained.

For example, we could truncate $\mathbf{g}'$ after seven terms:

$$\mathbf{g}'' = \left(g''_0, g''_1, g''_2, g''_3, g''_4, g''_5, g''_6\right) = \left(1, -1/2, 1/4, -1/8, 1/16, -1/32, 1/64\right)$$

a.) The goodness of the truncated / approximate inverse filter can be evaluated using the concept of squared error / error energy between the desired output $\boldsymbol{\delta}_0$ and the actual output $\boldsymbol{\delta}'' = \mathbf{g} * \mathbf{g}''$.

The error energy E is defined as $E = \sum_n \left(\delta_n - \delta_n''\right)^2$ where the summation index n covers all the non-zero terms of $\boldsymbol{\delta}_\mathbf{0}$ and $\boldsymbol{\delta}''$.

Example:

With $\mathbf{g} = \left(g_0, g_1\right) = \left(1, 1/2\right)$ and $\mathbf{g}'' = \left(g_0'', g_1'', g_2''\right) = \left(1, -1/2, 1/4\right)$, then

$\boldsymbol{\delta}'' = \mathbf{g} * \mathbf{g}'' = \left(\delta_0'', \delta_1'', \delta_2'', \delta_3''\right) = \left(1, 0, 0, 1/8\right)$. Compared with $\boldsymbol{\delta}_\mathbf{0} = \left(\delta_0\right) = \left(1\right)$,

$$E = \sum_{n=0}^{3} \left(\delta_n - \delta_n''\right)^2 = \left(1-1\right)^2 + \left(0-0\right)^2 + \left(0-0\right)^2 + \left(0-1/8\right)^2 = 1/64$$

Let us now consider the four term truncation

$\mathbf{g}'' = \left(g_0'', g_1'', g_2'', g_3''\right) = \left(1, -1/2, 1/4, -1/8\right)$. Then

$\boldsymbol{\delta}'' = \mathbf{g} * \mathbf{g}'' = \left(\delta_0'', \delta_1'', \delta_2'', \delta_3'', \delta_4''\right) = \left(1, 0, 0, 0, -1/16\right)$ and

$$E = \sum_{n=0}^{4} \left(\delta_n - \delta_n''\right)^2 = \left(1-1\right)^2 + \left(0-0\right)^2 + \left(0-0\right)^2 + \left(0-0\right)^2 + \left(0-1/16\right)^2 = 1/256$$

Increasing the truncation length of the approximate inverse filter commonly decreases the error energy.

b.) For a set inverse operator length, the simple truncated filter does not give the best possible results in terms of error energy. The optimal inverse filter can be determined using a least-squares criterion (the basis for least squares / Wiener filtering).

F.) Least Squares / Wiener Filtering

1.) This technique is used to design a filter that changes an input signal as closely as possible into a desired output signal. This objective is achieved in a least squares sense by minimizing the error energy.

2.) Derivation of the least squares method

a.) This process involves the following discrete causal signals:

$\mathbf{g} = (g_0, g_1, \cdots, g_N)$ is the (N+1) length input signal

$\mathbf{f} = (f_0, f_1, \cdots, f_M)$ is the (M+1) length filter

$\mathbf{d} = (d_0, d_1, \cdots, d_{M+N})$ is the (M+N+1) length desired output signal

$\mathbf{c} = \mathbf{g} * \mathbf{f} = (c_0, c_1, \cdots, c_{M+N})$ is the (M+N+1) length actual output signal

where $c_i = \sum_{j=0}^{M} f_j g_{i-j}$ for $i = 0, 1, \cdots, M+N$.

b.) The error energy $E = \sum_{i=0}^{M+N} (d_i - c_i)^2 = \sum_{i=0}^{M+N} \left[d_i - (\mathbf{g} * \mathbf{f})_i \right]^2$

$$= \sum_{i=0}^{M+N} \left[d_i - \left(\sum_{j=0}^{M} f_j \, g_{i-j} \right) \right]^2 .$$

c.) The minimum error is attained by setting the partial derivative of E with respect to each f_k to zero: $\frac{\partial E}{\partial f_k} = 0$ for $k = 0, 1, \cdots, M$.

$$\frac{\partial E}{\partial f_k} = \sum_{i=0}^{M+N} 2 \left[d_i - \left(\sum_{j=0}^{M} f_j \, g_{i-j} \right) \right] (-g_{i-k}) = 0 \Rightarrow -\sum_{i=0}^{M+N} d_i \, g_{i-k} + \sum_{i=0}^{M+N} \left(\sum_{j=0}^{M} f_j \, g_{i-j} \right) (g_{i-k}) = 0$$

$$\Rightarrow \sum_{j=0}^{M} f_j \left(\sum_{i=0}^{M+N} g_{i-j} \, g_{i-k} \right) = \sum_{i=0}^{M+N} d_i \, g_{i-k} \quad \text{for } k = 0, 1, \cdots, M$$

(These relationships are called the *normal equations*)

d.) The linear system of normal equations can be expressed in the following matrix form:

$$\begin{bmatrix} \phi_0 & \phi_1 & \phi_2 & \cdots & \phi_M \\ \phi_{-1} & \phi_0 & \phi_1 & & \vdots \\ \phi_{-2} & \phi_{-1} & \phi_0 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \phi_1 \\ \phi_{-M} & \cdots & \cdots & \phi_{-1} & \phi_0 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix} = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ \vdots \\ h_M \end{bmatrix}$$

where $\phi_{j-k} = \sum_{i=0}^{M+N} g_{i-j}\, g_{i-k}$ and $h_k = \sum_{i=0}^{M+N} d_i\, g_{i-k}$.

e.) It can be shown that $\phi_{j-k} = \phi_{k-j}$ →

$$\begin{bmatrix} \phi_0 & \phi_1 & \phi_2 & \cdots & \phi_M \\ \phi_1 & \phi_0 & \phi_1 & & \vdots \\ \phi_2 & \phi_1 & \phi_0 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \phi_1 \\ \phi_M & \cdots & \cdots & \phi_1 & \phi_0 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix} = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ \vdots \\ h_M \end{bmatrix}$$

f.) Different types of deconvolution operators can be obtained by selecting the appropriate desired output $\mathbf{d}$.

1.) Spiking deconvolution is done using $\mathbf{d} = \boldsymbol{\delta}_0 = (1, 0, \cdots, 0)$.

2.) A delayed spike $\boldsymbol{\delta}_n$ $(n \geq 1)$ can produce better results (i.e., lower E and less high frequency noise amplification) if the input wavelet is not minimum phase.

3.) Predictive deconvolution attempts to remove multiple events which can be predicted from knowledge of its corresponding primary reflection arrival time.

This is done by subtracting the predicted trace from the observed trace: $\mathbf{d}_t = \mathbf{g}_t - \mathbf{g}_{t-L} * \mathbf{f}_t$ where L is the time lag between the primary and multiple event arrivals.

g.) Prewhitening is performed by adding a small, positive constant to the diagonal of the matrix:

$$\begin{bmatrix} (1+\varepsilon)\phi_0 & \phi_1 & \phi_2 & \cdots & \phi_M \\ \phi_1 & (1+\varepsilon)\phi_0 & \phi_1 & & \vdots \\ \phi_2 & \phi_1 & (1+\varepsilon)\phi_0 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \phi_1 \\ \phi_M & \cdots & \cdots & \phi_1 & (1+\varepsilon)\phi_0 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix} = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ \vdots \\ h_M \end{bmatrix}$$

Example:

Consider $\mathbf{g} = (g_0, g_1) = (1, 1/2)$. From above, it can be seen that the polynomial division process gives the two term deconvolution operator $\mathbf{f}' = (f_0', f_1') = (1, -1/2)$. The corresponding actual output is $\mathbf{c}' = (c_0', c_1', c_2') = (1, 0, -1/4)$ with $E' = 1/16$.

Using the least squares method with $\mathbf{d} = (d_0, d_1, d_2) = \boldsymbol{\delta}_0 = (1, 0, 0)$, then deconvolution operator $\mathbf{f} = (f_0, f_1)$ is the solution to the matrix equation

$$\begin{bmatrix} \phi_0 & \phi_1 \\ \phi_1 & \phi_0 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix} = \begin{bmatrix} h_0 \\ h_1 \end{bmatrix}$$

where $\phi_0 = \sum_{i=0}^{2} g_i\, g_i = g_0\, g_0 + g_1 g_1 + g_2\, g_2 = 1\cdot 1 + \frac{1}{2}\cdot\frac{1}{2} + 0\cdot 0 = \frac{5}{4}$,

$$\phi_1 = \sum_{i=0}^{2} g_{i-1} g_{i-2} = g_{-1} g_{-2} + g_0\, g_{-1} + g_1 g_0 = 0\cdot 0 + 1\cdot 0 + \frac{1}{2}\cdot 1 = \frac{1}{2},$$

$$h_0 = \sum_{i=0}^{2} d_i\, g_i = d_0\, g_0 + d_1 g_1 + d_2\, g_2 = 1\cdot 1 + 0\cdot\frac{1}{2} + 0\cdot 0 = 1 \text{ and}$$

$$h_1 = \sum_{i=0}^{2} d_i\, g_{i-1} = d_0\, g_{-1} + d_1 g_0 + d_2\, g_1 = 1\cdot 0 + 0\cdot 1 + 0\cdot\frac{1}{2} = 0.$$

Solving the resulting system

$$\begin{bmatrix} 5/4 & 1/2 \\ 1/2 & 5/4 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \Rightarrow \quad \mathbf{f} = (f_0, f_1) = (20/21, -8/21).$$

The actual output is $\mathbf{c} = \mathbf{g} * \mathbf{f} = (1, 1/2) * (20/21, -8/21) = (20/21, -2/21, -4/21)$.

The corresponding error energy is

$$E=\sum_{i=0}^{2}\left(d_i-c_i\right)^2=\left(1/21\right)^2+\left(-2/21\right)^2+\left(-4/21\right)^2=1/21<E'=1/16$$

G.) Correlation of Discrete Time Signals

1.) Definition of correlation two discrete time series is

$$\boldsymbol{\varphi}^{\mathbf{gh}}=\mathbf{g}\otimes\mathbf{h}\rightarrow\phi_k^{gh}=\sum_{n=-\infty}^{n=+\infty}g_n\,h_{k+n}=\sum_{n=-\infty}^{n=+\infty}g_{n-k}\,h_n$$

2.) Cross-correlation of $\mathbf{g}$ and $\mathbf{h}$ can be done using a mechanical procedure similar to the one used to perform convolution.

a.) In the case of correlation, there is no folding / flipping.

b.) The time index of the correlation is determined by the amount of shift relative to the reference (i.e., zero shift) position.

Example:

$\mathbf{g}=\left(g_0,g_1,g_2\right)=\left(1,-2,3\right)$ and $\mathbf{h}=\left(h_0,h_1,h_2\right)=\left(4,1,-1\right)$

ϕ_0^{gh} : zero shift between signals (i.e., g and h indices identical down column)

	(	g_0	g_1	g_2	)
	(	1	-2	3	)
	(	4	1	-1	)
	(	h_0	h_1	h_2	)
$\phi^{gh}{}_0$ =		4	-2	-3	= -1

ϕ_1^{gh} : $\mathbf{h}$ shifted one increment left (i.e., h indices +1 down column)

		(	g_0	g_1	g_2	)
		(	1	-2	3	)
	(	4	1	-1	)	
	(	h_0	h_1	h_2	)	
$\phi^{gh}{}_1$ =			1	2		= 3

ϕ_{-2}^{gh} : $\mathbf{h}$ shifted two increment right (i.e., h indices -2 down column)

$$
\begin{array}{rccccc}
 & (\ g_0 & g_1 & g_2\ ) & & \\
 & (\ 1 & -2 & 3\ ) & & \\
 & & (& 4 & 1 & -1\ ) \\
 & & (& h_0 & h_1 & h_2\ ) \\
\hline
\phi^{gh}_{-2} = & & & 12 & &
\end{array}
$$

3.) Autocorrelation has the same form as cross correlation with $\mathbf{h} = \mathbf{g}$:

$$\boldsymbol{\varphi}^{\mathbf{gg}} = \mathbf{g} \otimes \mathbf{g} \rightarrow \phi_k^{gg} = \sum_{n=-\infty}^{n=+\infty} g_n\, g_{k+n}$$

H.) Discrete Fourier Transform and Fast Fourier Transform

1.) The Fourier transform of a discrete time signal $G_s(f) = \sum_{n=-\infty}^{+\infty} g_n\, e^{-2\pi i n \Delta t f}$ (from above) is a continuous function in frequency f. It is necessary to develop a discrete (i.e., uniformly sampled) version of the Fourier transform for digital analysis and processing.

2.) The motivation for the discrete Fourier transform (DFT) comes from the form of the Fourier series.

a.) A periodic signal has a Fourier series with discrete frequency components that are integer multiples of the fundamental frequency $f_{fundamental} = 1/T$.

b.) Consider the finite length, discrete time signal $\mathbf{g} = (g_0, g_1, g_2, \cdots, g_{N-1})$. It can be converted into a periodic signal by repeating the elements ad infinitum. This process produces a periodic discrete time signal with period $T = N\,\Delta t$ where N is the number of time series elements and Δt is the sampling interval of $g(t)$.

c.) In terms of frequency content, this period time signal contains:

1.) Zero frequency (i.e., DC bias) component.

2.) Fundamental frequency component $f_{fundamental} = 1/T = 1/N\,\Delta t$.

3.) Harmonic frequency components $f_{nth\ harmonic} = n/T = n/N\,\Delta t$ where $n = 2, 3, 4, 5, \cdots$.

d.) At first glance, it would appear that while we have achieved the goal of discrete frequency components, we have produced an infinitely long sequence of elements.

However, because $\mathbf{g}$ is a discrete time signal, its Fourier transform is periodic with period $1/\Delta t$ due to the replication effects of time sampling.

1.) Therefore, the frequency components $f_n = n/N\,\Delta t$ $(n = 0, 1, 2, 3, \cdots, N-1)$ completely characterize the discrete frequency spectrum.

2.) N time samples give N frequency samples.

3.) We do not need $f_N = 1/\Delta t$; due to periodicity, $f_0 = f_N$.

3.) Given the above motivation, the discrete Fourier transform (DFT) pair $\mathbf{g} = (g_0, g_1, g_2, \cdots, g_{N-1}) \Leftrightarrow \mathbf{G} = (G_0, G_1, G_2, \cdots, G_{N-1})$ is defined in the following manner:

a.) Forward DFT: $G_k = G(f = k/N\Delta t) = \sum_{n=0}^{N-1} g_n\, e^{-2\pi i n k/N}$ for $k = 0, 1, 2, \cdots, N-1$

b.) Inverse DFT: $g_k = g(t = k\Delta t) = \frac{1}{N}\sum_{n=0}^{N-1} G_n\, e^{+2\pi i n k/N}$ for $k = 0, 1, 2, \cdots, N-1$

4.) An algorithm called the Fast Fourier transform (FFT) can be used to very rapidly compute the DFT if N (the number of samples in the time series) is a power of 2 (i.e., 2, 4, 8, 16,…).

a.) The reason for the efficient computation is called “doubling”; this phenomenon occurs when $N = 2^n$. This property allows us to organize computation so the number of multiplications and exponential terms used is reduced.

b.) Two point FFT: $\mathbf{g} = (g_0, g_1) \Leftrightarrow \mathbf{G} = (G_0, G_1)$ where

$$G_k = \sum_{n=0}^{1} g_n e^{-2\pi i n k/N} = g_0 + g_1 e^{\pi i k} = \begin{cases} g_0 + g_1, k = 0 \\ g_0 - g_1, k = 1 \end{cases}$$

Because $e^{\pi i k} = (-1)^k$, the two point FFT requires no multiplications or exponentials, just different ordering of the g_k terms.

c.) Four point FFT: $\mathbf{g} = (g_0, g_1, g_2, g_3) \Leftrightarrow \mathbf{G} = (G_0, G_1, G_2, G_3)$ where

$$G_k = \sum_{n=0}^{3} g_n e^{-2\pi i n k/N} = g_0 + g_1 e^{-\pi i k/2} + g_2 e^{-\pi i k} + g_3 e^{-3\pi i k/2}$$

$$= \left(g_0 + g_2 e^{-\pi i k}\right) + e^{-\pi i k/2}\left(g_1 + g_3 e^{-\pi i k}\right)$$
$$= \left[g_0 + (-1)^k g_2\right] + e^{-\pi i k/2}\left[g_1 + (-1)^k g_3\right] \text{ for } k = 0, 1, 2, 3$$

We have effectively reorganized the computation into two point operations. Further, we need only a single exponential computation (vs. three if we had done the straight summation definition) for each k value.

d.) Eight point FFT:

$$\mathbf{g} = (g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7) \Leftrightarrow \mathbf{G} = (G_0, G_1, G_2, G_3, G_4, G_5, G_6, G_7)$$

Following the same approach, we can show that

$$G_k = \left(g_0 + g_4 e^{-\pi i k}\right) + e^{-\pi i k/4}\left(g_1 + g_5 e^{-\pi i k}\right) + e^{-\pi i k/2}\left(g_2 + g_6 e^{-\pi i k}\right)$$
$$+ e^{-3\pi i k/4}\left(g_3 + g_7 e^{-\pi i k}\right)$$

$$= \left[g_0 + (-1)^k g_4\right] + e^{-\pi i k/4}\left[g_1 + (-1)^k g_5\right] + e^{-\pi i k/2}\left[g_2 + (-1)^k g_6\right]$$
$$+ e^{-3\pi i k/4}\left[g_3 + (-1)^k g_7\right] \text{ for } k = 0, 1, 2, 3, 4, 5, 6, 7$$

Once again, the transform is converted into a series of two point operations with a reduced number of multiplications and exponentials.

e.) The above strategy, combined with an efficient decomposition process ("bit reversal") and merging process with the exponential terms (called a "butterfly operation") defines the FFT.

f.) To obtain $\mathbf{g}$ with $N = 2^n$, zero padding is done before performing the FFT.