FILTERS, CONVOLUTION and DFT

- First we need to recall how to find the matrix of a linear transformation
- So if $L: V \to V$ satisfies

$$L(\alpha \vec{v} + \beta \vec{w}) = \alpha L\vec{v} + \beta L\vec{w}$$

what is the matrix M so that

$$L(\vec{v}) = M\vec{v}$$

- The trick is to write $\vec{v}$ in terms of the standard basis $\vec{e}_1, \vec{e}_2, \dots, \vec{e}_N$ with $e_j = \begin{array}{cl} \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} & \begin{array}{l} \\ \\ \\ \leftarrow j \\ \\ \\ \\ \end{array} \end{array}$

$\vec{v} = \alpha_1 \vec{e}_1 + \alpha_2 \vec{e}_2 + \dots + \alpha_N \vec{e}_N$ so $\vec{v} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix}$

So using the properties of L

$$L(\vec{v}) = \alpha_1 L(\vec{e}_1) + \alpha_2 L(\vec{e}_2) + \dots + \alpha_N L(\vec{e}_N)$$

so if $\vec{\alpha} = [\alpha_1\ \alpha_2 \dots \alpha_N]^T$ then

$$L(\vec{\alpha}) = L(\vec{v}) = \begin{bmatrix} L(\vec{e}_1) & L(\vec{e}_2) & \dots & L(\vec{e}_N) \\ \\ \end{bmatrix} \vec{\alpha}$$

<u>example</u> if $L(\vec{e}_1) = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$, $L(\vec{e}_2) = \begin{bmatrix} 0 \\ 1 \\ 3 \end{bmatrix}$, $L(\vec{e}_3) = \begin{bmatrix} -1 \\ 0 \\ 2 \end{bmatrix}$

then $L(\vec{v}) = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ 3 & 3 & 2 \end{bmatrix} \vec{v}$

$$x \longrightarrow \boxed{L} \longrightarrow Lx$$

- Now x is a data stream. A linear filter is a process which takes x and yields Lx with L a linear transformation.

- We would like the filter to work the same no matter what point we designate as the 0 point

- The shift of a vector pushes element to the right by one and brings the last element to the front

$$S\begin{bmatrix}\alpha_1\\ \alpha_2\\ \vdots\\ \alpha_N\end{bmatrix} = \begin{bmatrix}\alpha_N\\ \alpha_1\\ \vdots\\ \alpha_{N-1}\end{bmatrix}$$

- Thus the shift is represented by the matrix

$$\begin{bmatrix} Se_1 & Se_2 & \cdots & Se_n \end{bmatrix} = \begin{bmatrix} 0 & 0 & & 0 & 1 \\ 1 & 0 & & 0 & 0 \\ 0 & 1 & & \vdots & \vdots \\ 0 & \vdots & & & \\ \vdots & & & & \\ 0 & \vdots & & 1 & 0 \end{bmatrix}$$

- A linear filter is shift invariant if shifting the input shifts the output by the same amount (also called translation invariant)

- In symbols $LS = SL$ (*)

- A LTI filter is one that is linear and translation invariant. They have special matrix representations which are connected to the DFT

Let's say $L\vec{e}_1 = \vec{g} = \begin{bmatrix} g_0 \\ \vdots \\ g_{N-1} \end{bmatrix}$

Then since $S\vec{e}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \vec{e}_2$

using equation (*)

$$L\vec{e}_2 = LS\vec{e}_1 = SL\vec{e}_1 = S\vec{g} = \begin{bmatrix} g_{N-1} \\ g_0 \\ \vdots \\ g_{N-2} \end{bmatrix}$$

For any j, let $S^j = SS..S$ (j-times)

$$Le_j = LS^j e_1 = S^j L e_1 = S^j \vec{g} = \begin{array}{l} \begin{bmatrix} \vdots \\ g_{N-1} \\ g_0 \\ \vdots \\ \vdots \end{bmatrix} \begin{array}{l} \leftarrow g_{N-j+1} \\ \\ \\ \\ \leftarrow g_{N-j} \end{array} \end{array}$$

So the matrix representing the LTI Filter is

$$M = \begin{bmatrix} g & Sg & \cdots & S^{N-1}g \end{bmatrix} = \begin{bmatrix} g_0 & g_{N-1} & & g_1 \\ \vdots & g_0 & & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ g_{N-1} & g_{N-2} & & g_0 \end{bmatrix}$$

This kind of matrix is called Toeplitz or circulent.

g is called the impulse response since it is the output from a pulse of magnitude 1 at time zero $L(\vec{e}_1)$. It determines the LTI Filter completely.

- To understand how Toeplitz matrices work or equivalently LTI Filers, we have to understand the initially unrelated notion of convolution.

- If $\vec{f}$ and $\vec{g}$ are both N-dimensional vectors indexed $0, \ldots, N-1$ Then the n^{th} component of their cyclic convolution $\vec{f} * \vec{g}$ is

$$(f * g)_n = \sum_{j=0}^{N-1} f_j \, g_{n-j}$$

where we always work with subscripts mod N

- Thus $f * g$ is another N-dimensional vector.

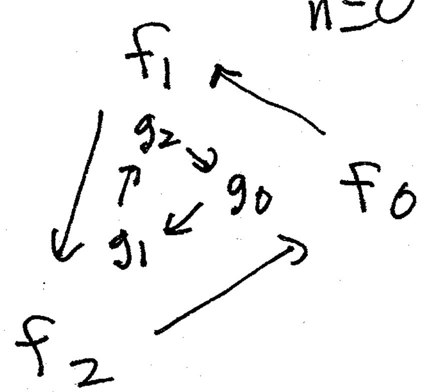
Example: Let $N = 3$

$$(f*g)_0 = \sum_{j=0}^{2} f_j g_{-j} = f_0 g_0 + f_1 g_{-1} + f_2 g_{-2}$$
$$= f_0 g_0 \quad f_1 g_2 + f_2 g_1$$

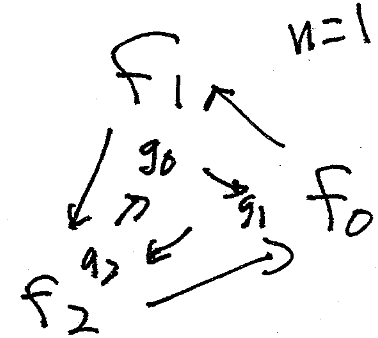
$$(f*g)_1 = \sum_{j=0}^{2} f_j \, g_{1-j} = f_0 g_1 + f_1 g_0 + f_2 g_{-1}$$
$$= f_0 g_1 + f_1 g_0 + f_2 g_2$$

$$(f*g)_2 = \sum_{j=0}^{2} f_j \, g_{\underset{\uparrow 2}{}-j} = f_0 g_2 + f_1 g_1 + f_2 g_0 \quad \text{- VISUALIZE}$$

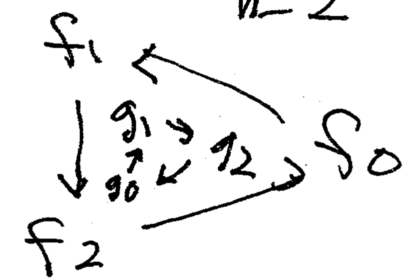
$n=0$



$n=1$



$n=2$



note that g is reversed and rotated

Now notice that convolution against g is Linear

$$L(\vec{v}) = \vec{v} * \vec{g}$$

$$L(\vec{v}+\vec{w}) = (\vec{v}+\vec{w}) * \vec{g} = \vec{v}*g + \vec{w}*\vec{g} = L(\vec{v}) + L(\vec{w})$$

$$L(\alpha\vec{v}) = (\alpha\vec{v}) * \vec{g} = \alpha(\vec{v}*g) = \alpha L(v)$$

What is its matrix? Using the formulas on the previous page

$$L(\vec{v}) = \begin{bmatrix} L(\vec{e}_1) & L(\vec{e}_2) & L(\vec{e}_3) \\ & & \end{bmatrix} \vec{v} = \begin{bmatrix} g_0 & g_2 & g_1 \\ g_1 & g_0 & g_2 \\ g_2 & g_1 & g_0 \end{bmatrix} \vec{v}$$

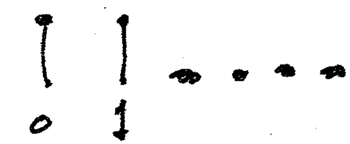
The circulant matrix determined by $\vec{g}$, the same as the LTI Filter with impulse $\vec{g}$.

<u>Theorem</u>: If $L: V \to V$ is a LTI Filter Than it is determined by an impulse response vector $\vec{g}$ and L acts by convolution

$$L(\vec{v}) = \vec{v} * g.$$

LTI Filters have many uses but let's consider how they are computed. For large N, $\vec{v} * \vec{g}$ is large to compute as is $M\vec{v}$. The fast way is via the DFT as implemented by the FFT.

First an example, let the impulse "function" g be $g_0 = 1/2$ $g_1 = 1/2$, $g_j = 0$ $j = 2, \ldots, N-1$



Then $(f * g)_n = \sum_{j=0}^{N-1} f_j \, g_{n-j} = f_n g_0 + f_{n-1} g_1$

$= \frac{1}{2}(f_n + f_{n-1})$.

So this filter replaces the point at place n with its average with the point to its left

The pointwise product of two vectors is

$$\vec{u} .* \vec{v} = \begin{bmatrix} u_1 v_1 \\ u_2 v_2 \\ \vdots \\ u_N v_N \end{bmatrix} \quad \text{so} \quad \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} .* \begin{bmatrix} -1 \\ 4 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ 8 \\ 6 \end{bmatrix}$$

(Convolution Theorem)

The main theorem says that the DFT turns convolutions into pointwise products

<u>Theorem</u> If x and y are data vectors of the same length and $\hat{x} = DFT(x)$ and $\hat{y} = DFT(y)$

$$\widehat{x * y} = \frac{1}{\sqrt{N}} \hat{x} .* \hat{y}$$

or $$DFT(x * y) = \frac{1}{\sqrt{N}} \left(DFT(x) .* DFT(y) \right)$$

- This gives a way to compute $*$ using the DFT and its inverse IDFT

$$x * y = \frac{1}{\sqrt{N}} \, IDFT\left(DFT(x) .* DFT(y) \right)$$

- note in matlab since $X_j = \sum_{k=1}^{N} x_k \, \omega^{-(j-1)(k-1)}$

In matlab notation
$$x * y = ifft\left(fft(x) .* fft(y) \right)$$

- the proof of the convolution Theorem is a calculation

13

Proof:

$$\hat{x}_j \hat{y}_j = \frac{1}{N}\left(\sum_{k=0}^{N-1} x_k \omega^{-jk}\right)\left(\sum_{\ell=0}^{N-1} y_\ell \omega^{-j\ell}\right)$$

$$= \frac{1}{N} \sum_{k=0}^{N-1} \sum_{\ell=0}^{N-1} x_k y_\ell \, \omega^{-j(k+\ell)}$$

$$\boxed{\text{let } n = k+\ell \text{ so } \ell = n-k}$$

$$= \frac{1}{N} \sum_{n=0}^{N-1} \sum_{k=0}^{N-1} x_k y_{n-k} \, \omega^{-jn}$$

$$= \frac{1}{\sqrt{N}} \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} \left(\sum_{k=0}^{N-1} x_k y_{n-k} \right) \omega^{-jn}$$

$$= \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} (x * y)_n \, \omega^{-jn} = \frac{1}{\sqrt{N}} \left(\widehat{x * y}\right)_j .$$

$\frac{1}{\sqrt{N}}$

$\boxed{\text{DEMO}}$