UNFOLD

MingJian Hong
hongmingjian@gmail.com

November 7, 2011

Theory

The UNFOLD recovers the image sequence from the k-t space sequence, which is under-sampled in phase direction, by labeling and resolving the overlapped components in the time axis. The idea behind UNFOLD is short and sweet, and shows that the authors had a profound understanding of the Fourier transform.

To simplify the subsequent description, the frequency-encoding direction is omitted.

With a full sampling, the DFT pair is

$$X[k] = \sum_{n=0}^{N-1} x[n] exp(-j\frac{2\pi}{N}nk) \tag{1}$$

$$x[n] = \frac{1}{N}\sum_{n=0}^{N-1} X[k] exp(j\frac{2\pi}{N}nk) \tag{2}$$

Suppose that the N is even, two under-sampling patterns are defined as follows.

$$\delta_o[n] = \begin{cases} 1 & : \quad n = 2k+1 \\ 0 & : \quad n = 2k \end{cases}$$

$$\delta_e[n] = \begin{cases} 0 & : \quad n = 2k+1 \\ 1 & : \quad n = 2k \end{cases}$$

where k $\in [0, \frac{N}{2} - 1]$

Obviously, $\delta_o[n] = \delta_e[((n-1))_N]$ holds true. According to the property of the DFT, we have

$$IDFT(\delta_o)[n] = IDFT(\delta_e)[n]exp(j\frac{2\pi}{N}n) \tag{3}$$

The $IDFT(\delta_e)$ can be expanded as follows.

$$\begin{aligned} IDFT(\delta_e)[n] &= \frac{1}{N}\sum_{k=0}^{N-1}\delta_e[k]exp(j\frac{2\pi}{N}kn) \\ &= \frac{1}{2}\frac{1}{N/2}\sum_{k=0}^{N/2-1}exp(j\frac{2\pi}{N/2}kn) \end{aligned} \tag{4}$$

which is reduced to

$$IDFT(\delta_e)[n] = \begin{cases} \frac{1}{2} & : \quad n = 0, \frac{N}{2} \\ 0 & : \quad otherwise \end{cases} \tag{5}$$

by using the equation in the Problem 3.54 of [3]. Accordingly, we have

$$IDFT(\delta_o)[n] = \begin{cases} \frac{1}{2} & : \quad n = 0 \\ -\frac{1}{2} & : \quad n = \frac{N}{2} \\ 0 & : \quad otherwise \end{cases} \tag{6}$$

The reconstructions from the k-space under-sampled by $\delta_e[n]$ and $\delta_o[n]$ are as follows.

$$\begin{aligned} x_e[n] &= IDFT(X\delta_e) \\ &= IDFT(X) * IDFT(\delta_e) \end{aligned} \tag{7}$$

$$\begin{aligned} x_o[n] &= IDFT(X\delta_o) \\ &= IDFT(X) * IDFT(\delta_o) \end{aligned} \tag{8}$$

Based on the property of the convolution, $x_e[n]$ and $x_o[n]$ are the sum of replications of $\frac{1}{2}x[n]$ and $\frac{(-1)^k}{2}x[n]$ centered at $k\frac{N}{2}, k = 0, \pm1, \pm2, ...$, respectively, as shown in Fig.(**??**). Mathematically, we have

$$x_e[n] = \frac{1}{2}(x[n] + x[((n + \frac{N}{2}))_N]) \tag{9}$$

$$x_o[n] = \frac{1}{2}(x[n] - x[((n + \frac{N}{2}))_N]) \tag{10}$$

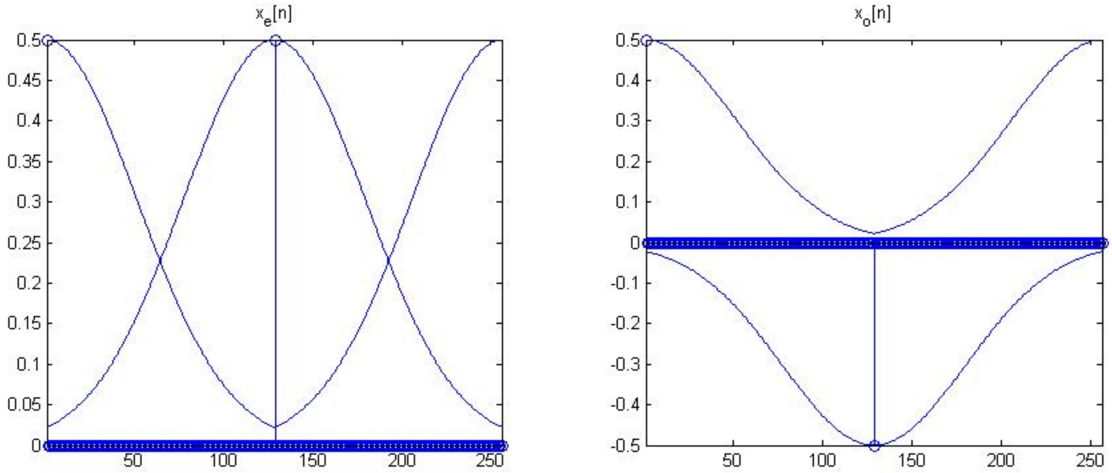

Figure 1: replications - x_e[n] and x_o[n] is the sum of two curves.

So far, we do not consider the time axis of the image sequence. Suppose that the k-space of the even frame is under-sampled by δ_e and odd frame by δ_o.

If the points at n_0 and $n_1 = ((n_0 + \frac{N}{2}))_N$ are constant in time, the value of aliased point where they overlap will oscillate between $\frac{1}{2}(x[n_0] + x[n_1])$ and $\frac{1}{2}(x[n_0] - x[n_1])$ frame by frame, as shown in the Fig.(**??**a).

The UNFOLD resolves $x[n_0]$ and $x[n_1]$ out of the aliased point by Fourier transform in the time axis. As shown in the Fig.(**??**b), the spatial overlapped points are discriminable in the Fourier domain along the time axis. By applying a predefined filter (the red curve), the $x[n_0]$ and $x[n_1]$ can be separated and recovered.

In the case of non-constant $x[n_0]$ and $x[n_1]$ in time (Fig.(**??**c)), the spectra associated with them contains a range of frequencies instead of a single component, as shown in the Fig.(**??**d).

Matlab code for UNFOLD

```
clear
load d:\my\work\007\sol_yxzt_pat1

% 0 ———————— UNFOLDing a real cardiac cine
```

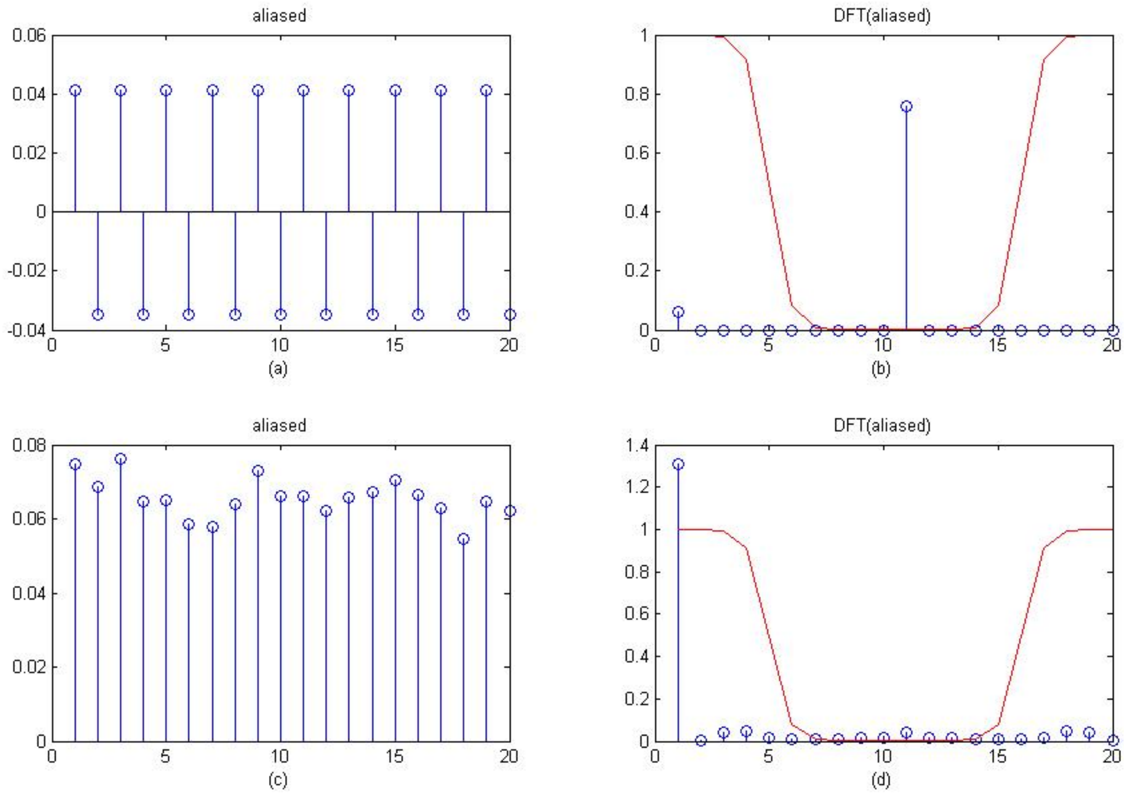

Figure 2: UNFOLD

```
% non—zero — showing the idea behind the UNFOLD
ideaonly=1;

% Used ONLY when `ideaonly' is non—zero
% 0 ———————— UNFOLDing constant signal
% non—zero — UNFOLDing time—varying signal
dynamic = 0;


T=20; % number of frames
N=256;

for i=1:T
    tmp=sol_yxzt(:, :, 1, i);
    tmp=tmp./max(tmp(:));
    xtref(:, :, i)=tmp;
end

% patterns
p1=zeros(N, 1); p1(1:2:end)=1;
p2=zeros(N, 1); p2(2:2:end)=1;

%Fermi filter to preserve the DC—component
tmp=linspace(0, 1, 20);
tmp=tmp(:);
f1=1./(1+exp((tmp—0.79)./0.022));
f2=1./(1+exp((flipud(tmp)—0.79)./0.022));
f=abs(f1—f2);

if ideaonly == 0
    % zero—fill recovery
    for i=1:T
        ktref(:, :, i)=fft2(xtref(:, :, i));
        if mod(i, 2) == 1
            p=repmat(p1, 1, N);
        else
            p=repmat(p2, 1, N);
        end;
        xtalias(:, :, i)=2*real(ifft2(ktref(:, :, i).*p));
    end

    % UNFOLD pixel by pixel
    for i=1:N
        for j=1:N
            t=xtalias(i,j, :);
            t=t(:);
            Ft=fft(t);

            Ft=Ft.*f;
```

```

            xthat(i, j, :)=real(ifft(Ft));
        end
    end
    implay(xthat)
else %if ideaonly ≠ 0
    col=128; % take 128th column as the example
    if dynamic == 0
        % each column is the same
        tmp=xtref(:, col, 1);
        X=repmat(tmp(:), 1, T);
    else
        % columns change with time
        for i=1:T
            tmp=xtref(:, col, i);
            X(:, i)=tmp(:);
        end
    end

    % to k-space
    K=fft(X); % cannot use the fft2!!!

    % under-sampling
    Ku=K.*repmat([p1 p2], 1, T/2);

    % zero-fill recovery
    Xu=2*real(ifft(Ku));

    % take 128th point as the example
    n0=128; %n1=mod(N/2+n0, N);

    if dynamic == 0
        x=X(:, 1); % any column suffices

        x1=Xu(:, 1);
        x2=Xu(:, 2);

        % just for verification, MBZ
        norm((x+circshift(x, N/2)) - x1)
        norm((x-circshift(x, N/2)) - x2)

        % to Fourier domain
        t=Xu(n0, :)';
        Ft=fft(t);
        figure;  stem(Ft);
        hold on; stem(f, 'r');
        legend('Spectrum', 'Fermi filter');
        xlabel('frame');

```

```
        % verify that Ft(1) is the DC—component
        %mean(t)—mean(ifft(Ft.*f))

        % XXX — how to verify the Ft(T/2+1) is the ...
            Nyquist—component?
    else
        % to Fourier domain
        t=Xu(n0, :)';
        Ft=fft(t);
        figure;  stem(Ft);
        hold on; stem(f, 'r');
        legend('Spectrum', 'Fermi filter');
        xlabel('frame');
    end

    % filter the Nyquist—component out
    dc=Ft.*f;

    % recover x from the DC—component
    norm(real(ifft(dc))—X(n0, :)') % show the error(owing ...
        to frequency leakage)

    figure;  stem(real(ifft(dc)));
    hold on; stem(X(n0, :), 'r');
    legend('recovered', 'reference');
    xlabel('frame');
end
```

References

[1] B. Madore, G. H. Glover, and N. J. Pelc, "Unaliasing by Fourier-encoding the overlaps using the temporal dimension (UNFOLD), applied to cardiac imaging and fMRI", Magnetic Resonance in Medicine 42, no. 5 (1999): 813-828.

[2] Jeffrey Tsao, "On the UNFOLD method", Magnetic Resonance in Medicine 47, no. 1 (2002): 202-207.

[3] A. V. Oppenheim, A. S. Willsky, and with S. Hamid, Signals and Systems, 2nd ed. Prentice Hall, 1996.