The fast Fourier transform (FFT) is an efficient and accurate tool for numerically filtering, integrating, and differentiating time-series data. For FTT calculations on segments of generally nonperiodic signals, the accuracy of these calculations is improved if:

- the time series amplitude tapers to zero at the beginning and end of the record,
- the time series is zero-padded at the beginning and/or the end of the record, and
- the filter transfer function is smooth.

In the FFT-based digital signal processing algorithm described here, a time series $u_k$ ($k = 1, \cdots, N$), with a sample interval of $\Delta t$, is first detrended, then windowed

$$\hat{u}_k = w_k(u_k - aj - b),$$

where $j = k - N/2$, $a = \sum (ju_k)/\sum j^2$, $b = \sum u_k/N$, and $w_k$ is a tapered windowing function,

$$w_k = \begin{cases} 
\frac{1}{2} [1 - \cos(\pi (k - 1)10/N)] & 1 \leq k \leq N/10 + 1 \\
1 & N/10 + 1 \leq k \leq N - N/10 \\
\frac{1}{2} [1 + \cos(\pi (k - N + N/10)10/N)] & N - N/10 \leq k \leq N 
\end{cases}$$

FFT’s are most efficient when applied to time series of lengths that are a power of 2, (e.g., 1024). In the algorithm described here, $N_{\text{fft}}$, is the integer power of two that is just larger than $N$,

$$N_{\text{fft}} = 2^{\lceil \log N/\log 2 \rceil}$$

where $\lceil \cdot \rceil$ rounds its argument up to the next largest integer. The frequency increment of a discrete Fourier transform computed from a time series of $N_{\text{fft}}$ points with a sample interval of $\Delta t$ is $\Delta f = 1/(N_{\text{fft}} \Delta t)$. According to the frequency sorting convention of the FFT, the frequency corresponding to the $k$-th Fourier coefficients, $f_k$, is

$$f_k = \begin{cases} 
(k - 1)\Delta f & 1 \leq k \leq N_{\text{fft}}/2 + 1 \\
(k - N_{\text{fft}} - 1)\Delta f & N_{\text{fft}}/2 + 2 \leq k \leq N_{\text{fft}} 
\end{cases}$$

The frequency $f_1$ is set to zero, the frequency $f_{N_{\text{fft}}/2+1}$ is the Nyquist frequency, $1/(2\Delta t)$, the frequency $f_{N_{\text{fft}}/2+2}$ is $\Delta f$ greater than $-1/(2\Delta t)$, and the frequency $f_{N_{\text{fft}}}$ is $-\Delta f$. 
If the sampled, detrended, and windowed signal $\hat{u}_k$ is to be band-pass filtered between $f_{lo}$ and $f_{hi}$ ($0 \leq f_{lo} \leq f_{hi} \leq 1/(2\Delta t)$), then the filter transfer function, $H(f_k)$ is set to zero for $k \leq k_{lo+}, k_{hi+} \leq k \leq k_{lo-}$, and $k_{hi-} \leq k$, where

$$k_{lo+} = \max(\lfloor f_{lo}/\Delta f \rfloor + 1, 1) \quad (5)$$

$$k_{hi+} = \min(\lfloor f_{hi}/\Delta f \rfloor + 1, N_{fft}/2 + 1) \quad (6)$$

$$k_{lo-} = \min(\lceil -f_{lo}/\Delta f \rceil + 1 + N_{fft}, N_{fft}) \quad (7)$$

$$k_{hi-} = \max(\lceil -f_{hi}/\Delta f \rceil + 1 + N_{fft}, N_{fft}/2 + 2) \quad (8)$$

and where $\lfloor \cdot \rfloor$ rounds its argument down to the next smallest integer. For the filter transfer function to vary smoothly, transition bandwidth values are specified as

$$N_{lo} = \min(\lceil k_{hi+} - k_{lo+} \rceil/10, k_{lo+} + 1) \quad (9)$$

$$N_{hi} = \min(\lceil k_{hi+} - k_{lo+} \rceil/10, k_{hi+} + 1) \quad (10)$$

For band-bass filtering, the filter transfer function is then

$$H(f_k) = 0 \quad (k \leq k_{lo+}, k_{hi+} \leq k \leq k_{lo-}, k_{hi-} \leq k) \quad (11)$$

$$H(k_{lo+} + k) = H(k_{lo-} - k) = (1 - \cos(\pi k/N_{lo}))/2 \quad (0 \leq k \leq N_{lo}) \quad (12)$$

$$H(k_{hi+} - k) = H(k_{hi-} + k) = (1 - \cos(\pi k/N_{hi}))/2 \quad (0 \leq k \leq N_{hi}) \quad (13)$$

$$H(f_k) = 1 \quad (k_{lo+} + N_{lo} \leq k \leq k_{hi+} - N_{hi}) \quad (14)$$

$$H(f_k) = 1 \quad (k_{hi-} + N_{hi} \leq k \leq k_{lo-} - N_{lo}) \quad (15)$$

If sampled signal, detrended, and windowed signal $\hat{u}_k$ is to be integrated or differentiated using the transfer function, then the filter transfer function is multiplied by

$$I(f_k) = (2\pi if_k)^{-n}, \quad (16)$$
where \( n \) is the number of integrations \((n < 0 \text{ means differentiation})\), \( i = \sqrt{-1} \) and \( I(f_1) = I(0) = 1 \).

![Figure 2](image1.png)

**Figure 2.** A smooth band-pass filter transfer function and a filtered integrator transfer function.

FFT-based digital signal processing is then carried out using FFT’s of length \( N_{\text{fft}} \). Values of \( \hat{u}_k \) beyond \( N \) \((N + 1 \leq k \leq N_{\text{fft}})\) are zero. The FFT of \( \hat{u}_k \) is computed from the forward FFT, \( U = \mathcal{F}\mathcal{F}\mathcal{T}[\hat{u}] \); these Fourier coefficients are multiplied by the filter transfer function to obtain the Fourier coefficients of the filtered signal, \( Y(f_k) = H(f_k)I(f_k)U(f_k) \), and the filtered signal is recovered from the inverse FFT, \( y = \mathcal{I}\mathcal{F}\mathcal{F}\mathcal{T}[Y] \). If \( u \) is a real-valued sequence then the imaginary part of \( y \) is essentially zero and the filtered signal is returned as the real part of \( y_k \), \((1 \leq k \leq N)\).

![Figure 3](image2.png)

**Figure 3.** The original signal, \( u \), and a band-pass filtered and integrated signal, \( y \).
function y = ftdsp(u,sr,flo,fhi,ni)
% y = ftdsp(u,sr,flo,fhi,ni)
% band-pass filter and integrate a discrete-time signal, u
% u : the discrete-time signals to be filtered/integrated
% sr : the sample rate
% flo : the low frequency limit for the bandpass filter (>= 0)
% fhi : the high frequency limit for the bandpass filter (<= sr/2);
% ni : the number of integrations (may be zero or negative for differentiation)

[P,m] = size(u); if ( m > P ), Tpose = 1; u = u'; else Tpose = 0; end

% de-trending and windowing the data can help with numerical accuracy
u = detrend(u); % detrend or base-line correction
Pw = floor(P/20); % number of window points
w = [0.5*(1-cos(pi*[0:Pw]/Pw)) ones(1,P-2*Pw-2) 0.5*(1+cos(pi*[0:Pw]/Pw))];
u = u.* (w.*ones(1,m)); % comment out this line for no windowing

NF = 2^ceil(log(P)/log(2)); % use 2^n points for FFT calculations
delta_f = sr/NF; % frequency resolution
f = [ [0: NF/2] [-NF/2+1: -1]' ] * delta_f; % frequency data

kloP = max(floor(flo/delta_f) + 1, 1 );
khiP = min(floor(fhi/delta_f) + 1, NF/2+1 );
kloN = min(ceil(-flo/delta_f) + 1 + NF, NF );
khiN = max(ceil(-fhi/delta_f) + 1 + NF, NF/2+2 );

Nband_lo = round(abs(khiP-kloP)/10); % low frequency transition bandwidth
Nband_hi = round(abs(khiP-kloP)/10); % high frequency transition bandwidth
if Nband_lo > kloP, Nband_lo = kloP+1; end
if Nband_hi > khiP, Nband_hi = khiP+1; end

H = zeros(NF,1); % initialize filter transfer function
H([kloP:khiP]) = 1; % positive band pass frequencies
H([khiN:kloN]) = 1; % negative band pass frequencies

if flo > delta_f
    for k = 0:Nband_lo -1 % taper in frequency domain
        H([kloP+k kloN-k]) = 0.5*(1-cos(k*pi/Nband_lo));
    end
end
if fhi < sr/2-delta_f
    for k = 0:Nband_hi -1 % taper in frequency domain
        H([khiP-k khiN+k]) = 0.5*(1-cos(k*pi/Nband_hi));
    end
end

ID = (i*2*pi*').*(-ni); ID(1) = 1; % integration/differentiation filter
U = fft(u,NF); % take the FFT of the real signal, u
Y = [H.ID*ones(1,m)].*U; % convolution with the filter transfer function
y = ifft(Y,NF); % Inverse FFT

if ( (max(norm(imag(y)) ./ norm(real(y))')) > 1e-4 )
disp( 'ftdsp: uh-oh, the imaginary part should be practically zero' );
end
y = real(y(1:P,:)); % retain only the original N data points
if ( Tpose ) y = y'; end

% H.P. Gavin, Dept. Civil and Environ. Eng’g, Duke Univ., Jul. 2007

[CC BY-NC-ND] H.P. Gavin September 20, 2018