DFT and Matlab

with some examples

Fourier transform

- Fourier transform is defined as:

$$X(\omega) = \int_{-\infty}^{+\infty} x(t) \cdot e^{-j\omega t} dt$$

- with $\omega = 2 \cdot \pi \cdot f$ Rad/s
- And x(t) a signal in the time domain.

Fourier series

- In periodic signals only harmonics are present and we can use the Fourier series for time domain signal x(t) with period T:

$$x(t) = \sum_{k=-\infty}^{\infty} \alpha_k e^{jk\omega_0 t} \quad \text{with} \quad \alpha_k = \tfrac{1}{T} \int_{-T/2}^{T/2} x(t) \cdot e^{-jk\omega_0 t} dt$$

- In which *k* is the harmonic index and ω_0 is the fundamental angular frequency.

Discrete Fourier transform (I)

- The DFT can be used to describe discrete signals in the frequency domain.
- The DFT $X(e^{j\omega t})$ of an arbitrary discrete signal x[nT] (or (x[n] when sampling frequency T normalized to 1 Hz), is defined as

$$X(\omega) \equiv X(e^{j\omega T}) = \sum_{n=-\infty}^{\infty} x[nT] \cdot e^{-jn\omega T}$$

ω is on the left hand side replaced by $e^{j\omega T}$ to indicate
X is periodic with a period 2π/T

- $T=1/f_{sample}$, and ω is 2π f, which is the x-axis frequency variable, and n is the index of samples taken each $1/f_{sample}$ seconds.

Discrete Fourier transform (II)

- By defining $\omega = k \cdot 2\pi \cdot f_{sample}/N_{DFT}$, with k the index of discrete frequency bins running from k=0 to $k=(N_{DFT}-1)$. N_{DFT} is the length of the DFT (i.e. the total number of samples used for the calculation of the DFT), we can write:

$$X(\omega) \equiv X(k) = \sum_{n=0}^{N_{DFT}-1} x[n] \cdot e^{-j\frac{2\pi \cdot n \cdot k}{N_{DFT}}}$$

- Notice that we have assumed that the length of the signal sequence x(n) is equal to N_{DFT}. If N_{DFT} is larger than the sequence length, tools, in general, will be zero-padding. If it is smaller you are throwing away samples. Both are normally undesirable. Also notice that $f_{sample} = 1/T$ cancels against T in the DFT definition from the previous slide.
- The calculated spectrum is made up of individual components in the frequency bins:

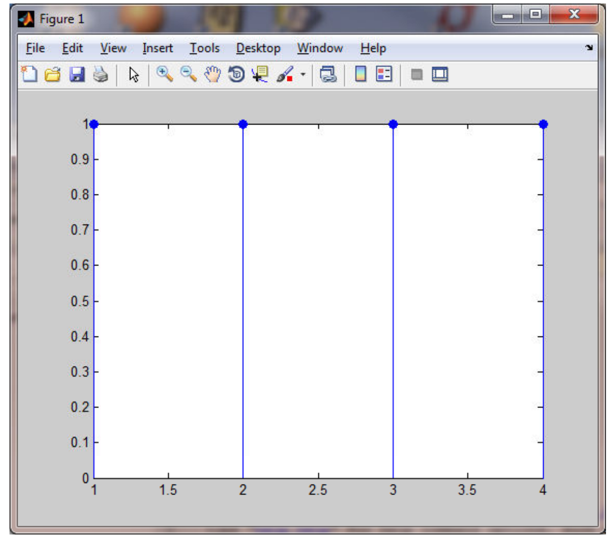
$$f(k) = k \cdot \frac{f_{sample}}{N_{DFT}} = k \cdot \frac{1}{N_{DFT} \cdot T_{sample}}$$

Discrete Fourier transform (III)

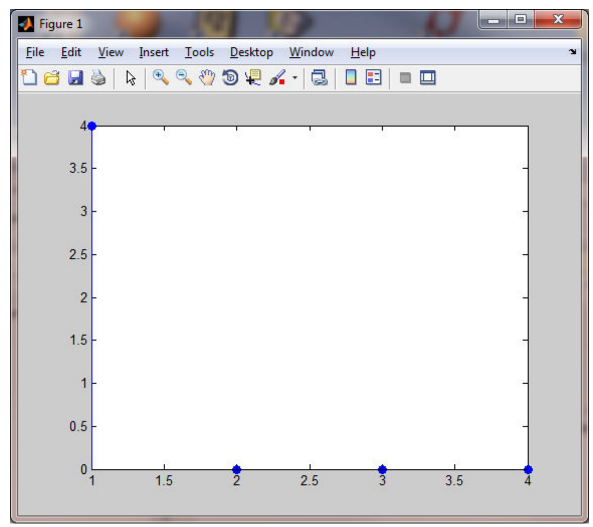
- **A few observations & remarks**:
  - Spacing or frequency resolution is $1/(N_{DFT}\ T_{sample})$. So the required observation time increases with the desired resolution.
  - $N_{DFT}\ T_{sample}$ is to simulation time required (DFT period) to get the required number samples N_{DFT} from a time continuous signal.
  - Normally we want an exact calculation of the spectrum: frequency bins in the discrete frequency spectrum corresponds exactly to the signal frequency.
    - For that to happen: the signal must be periodic (e.g. period T_0)
    - The observation time $N_{DFT} \cdot T_{sample}$ must be an integer multiple of the period T_0 of the signal
    - Effectively we can then do an DFT using the boxcar window (i.e. no windowing, which is best normally, if we have control over the input signal).
  - $f_{nyquist}$ is $f_{sample}/2$: be aware of components folding into the first Nyquist band.
  - Make sure the system you are simulating is settled: cut away settling effects before doing the DFT (otherwise your signal is not periodic).

Matlab example (I)

- X1[1 1 1 1]
- Stem(X1, ‘fill’) gives:



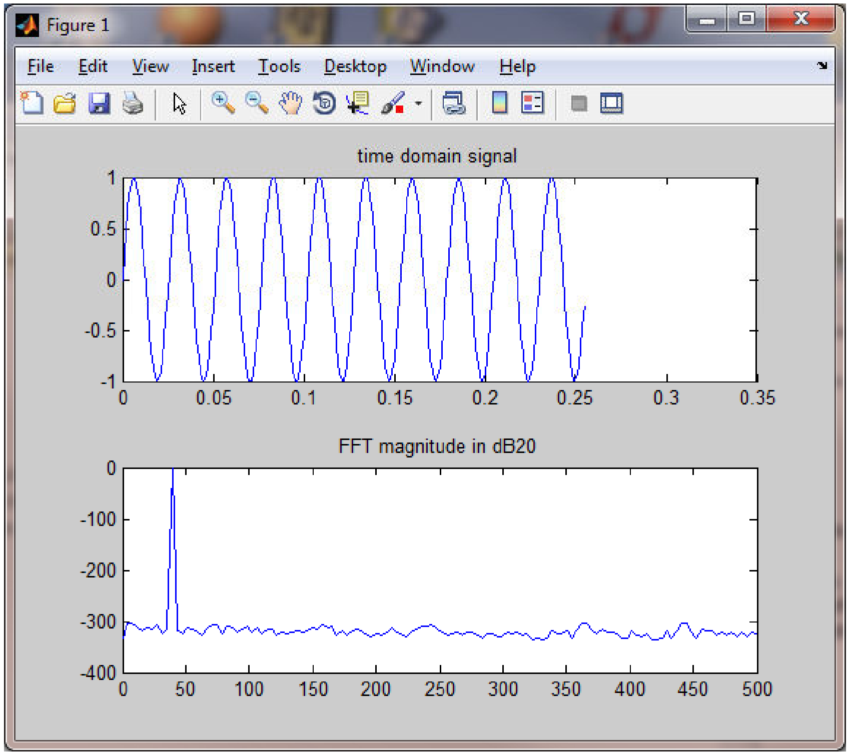
- Y1=FTT(X1,4)
- Plot(abs(Y1)) gives:



Note that Matlab doesn’t normalize: to get the right amplitude divide the magnitude by N_{DFT}

Matlab example (II)

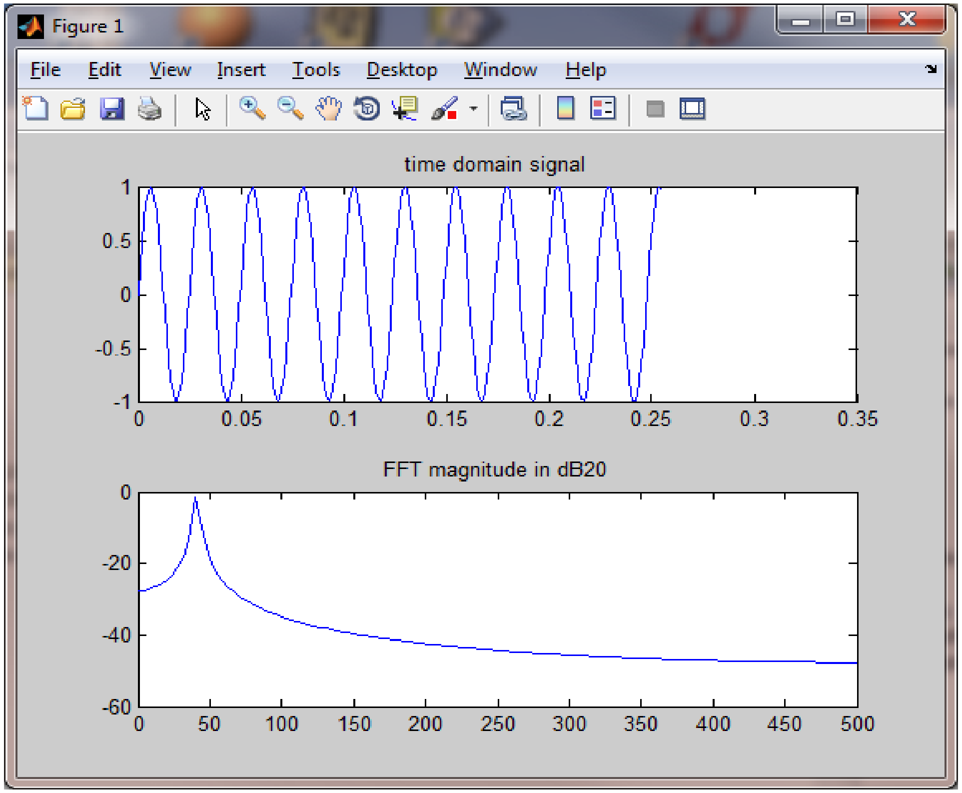
```
Fs=1000;
T=1/Fs;
NFFT=256;
FFTperiod=256*T;
t=(0:NFFT-1)*T;
str=['FFT period is ',num2str(FFTperiod), ' seconds'];
Freq=10*1/FFTperiod
strFreq =['Frequency is ', num2str(Freq), ' Hz'];
Freqbin =1/FFTperiod
strFreqbin =['FFT frequency bin is ',num2str(1/FFTperiod), ' Hz'];
strFreqNyquist =['FNyquist is ',num2str(1/FFTperiod*256/2), ' Hz'];
disp(str);
disp(strFreq);
disp(strFreqbin);
disp(strFreqNyquist);
x=sin(2*pi*Freq*t);
subplot(2,1,1)
plot(t,x);
TITLE('time domain signal')
Y=fft(x,NFFT)/NFFT;
f=Fs/2*linspace(0,1,NFFT/2);
subplot(2,1,2);
plot(f,20*log10(2*abs(Y(1:NFFT/2))))
TITLE('FFT magnitude in dB20')
```



Good example: The observation time $N_{DFT} \cdot T_{sample}$ is an integer multiple of the period T_0 of the signal.

Matlab example (III)

```
Fs=1000;
T=1/Fs;
NFFT=256;
FFTperiod=256*T;
t=(0:NFFT-1)*T;
str=['FFT period is ',num2str(FFTperiod), ' seconds'];
Freq=10.33*1/FFTperiod
strFreq =['Frequency is ', num2str(Freq), ' Hz'];
Freqbin =1/FFTperiod
strFreqbin =['FFT frequency bin is ',num2str(1/FFTperiod), ' Hz'];
strFreqNyquist =['FNyquist is ',num2str(1/FFTperiod*256/2), ' Hz'];
disp(str);
disp(strFreq);
disp(strFreqbin);
disp(strFreqNyquist);
x=sin(2*pi*Freq*t);
subplot(2,1,1)
plot(t,x);
TITLE('time domain signal')
Y=fft(x,NFFT)/NFFT;
f=Fs/2*linspace(0,1,NFFT/2);
subplot(2,1,2);
plot(f,20*log10(2*abs(Y(1:NFFT/2))))
TITLE('FFT magnitude in dB20')
```



Bad example: The observation time $N_{DFT} \cdot T_{sample}$ is not an integer multiple of the period T_0 of the signal.

References

- CMOS Mixed-signal Circuit design, R. J. Baker
- Digital Signal Processing, A practical Approach, E. Ifeachor, B. W. Jervis
- Discrete-Time Signal Processing, A. van den Enden, N. Verhoeckx