1. (a) Given the range of n, where $n = 0, 1, \ldots, \frac{N}{2} - 1$, $x(n)$ has the following symmetric property

$$x(n + \frac{N}{2}) = -x(n)$$

And $X(k)$ is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-\frac{j2\pi nk}{N}} = \sum_{n=0}^{\frac{N}{2}-1} x(n)e^{-\frac{j2\pi nk}{N}} + \sum_{n=\frac{N}{2}}^{N-1} x(n)e^{-\frac{j2\pi nk}{N}}$$

Calculation of $X(k)$ can be reduced to the range of $n = 0 \to \frac{N}{2} - 1$, as follows

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n)e^{-\frac{j2\pi nk}{N}} + \sum_{n=0}^{\frac{N}{2}-1} x(n + \frac{N}{2})e^{-\frac{j2\pi(n+\frac{N}{2})k}{N}}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(n)e^{-\frac{j2\pi nk}{N}} + x(n + \frac{N}{2})e^{-\frac{j2\pi(n+\frac{N}{2})k}{N}}$$

And due to the symmetric property of $x(n)$ outlined at the beginning

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n)e^{-\frac{j2\pi nk}{N}} - x(n)e^{-\frac{j2\pi(n+\frac{N}{2})k}{N}}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(n)\left[e^{-\frac{j2\pi nk}{N}} - e^{-\frac{j2\pi(n+\frac{N}{2})k}{N}}\right]$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(n)\left[e^{-\frac{j2\pi nk}{N}} - e^{-j\pi k}e^{-\frac{j2\pi nk}{N}}\right]$$

and since even harmonics of $X(k)$ are represented by even values of k, $e^{-j\pi k}$ will always be 1, meaning

$$X_{even}(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n)\left[e^{-\frac{j2\pi nk}{N}} - (1)e^{-\frac{j2\pi nk}{N}}\right]$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(n) \times 0$$

$$\therefore X_{even}(k) = 0$$

(b) Given $X(k)$ from (a)

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n)\left[e^{-\frac{j2\pi nk}{N}} - e^{-j\pi k}e^{-\frac{j2\pi nk}{N}}\right]$$

we can determine that the odd harmonics of $X(k)$ are when k is odd. So, $e^{-j\pi k}$ will always be -1, meaning

$$X_{odd}(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n)\left[e^{-\frac{j2\pi nk}{N}} - (-1)e^{-\frac{j2\pi nk}{N}}\right]$$
$$= \sum_{n=0}^{\frac{N}{2}-1} 2x(n)e^{-\frac{j2\pi nk}{N}}$$

And if $Y(m)$, the $N/2$-point DFT of $y(n)$, is equivalent to odd harmonics of $X(k)$, then

$$Y(m) = \sum_{n=0}^{\frac{N}{2}-1} y(n)e^{-\frac{j2\pi nm}{N}} = \sum_{n=0}^{\frac{N}{2}-1} 2x(n)e^{-\frac{j2\pi nk}{N}}, \qquad k \texttt{ mod } 2 = 1$$

Since k can only be odd, we can replace it with $(2m + 1)$ to keep the identity equivalent and reduce variables

$$Y(m) = \sum_{n=0}^{\frac{N}{2}-1} y(n)e^{-\frac{j2\pi nm}{N}} = \sum_{n=0}^{\frac{N}{2}-1} 2x(n)e^{-\frac{j2\pi n(2m+1)}{N}}$$
$$= \sum_{n=0}^{\frac{N}{2}-1} y(n)e^{-\frac{j2\pi nm}{N}} = \sum_{n=0}^{\frac{N}{2}-1} 2x(n)e^{-\frac{j4\pi nm}{N}}e^{-\frac{j2\pi n}{N}}$$

4π can be reduced to 2π in the right-side complex exponential, resulting in $e^{-\frac{j2\pi nm}{N}}$ being factored out

$$\sum_{n=0}^{\frac{N}{2}-1} y(n) = \sum_{n=0}^{\frac{N}{2}-1} 2x(n)e^{-\frac{j2\pi n}{N}}$$

If we assume element-wise equivalence, we get one solution for $y(n)$, being

$$y(n) = 2x(n)e^{-\frac{j2\pi n}{N}}, \qquad n = 0, 1, \ldots, \frac{N}{2} - 1$$

2.

$$x(n) = \{3,\ \ 0,\ \ -1,\ \ 2\}$$
$$y(n) = \{1,\ \ 5,\ \ 4,\ \ -2\}$$

Let's say the circular convolution of $x(n)$ and $y(n)$ equals $z(n)$

(a)

$$z(0) = \sum\{3 \times 1,\ \ 0 \times \text{-}2,\ \ \text{-}1 \times 4,\ \ 2 \times 5\} = 9$$
$$z(1) = \sum\{3 \times 5,\ \ 0 \times 1,\ \ \text{-}1 \times \text{-}2,\ \ 2 \times 4\} = 25$$
$$z(2) = \sum\{3 \times 4,\ \ 0 \times 5,\ \ \text{-}1 \times 1,\ \ 2 \times \text{-}2\} = 7$$
$$z(3) = \sum\{3 \times \text{-}2,\ \ 0 \times 4,\ \ \text{-}1 \times 5,\ \ 2 \times 1\} = -9$$
$$z(n) = \{9,\ \ 25,\ \ 7,\ \ -9\}$$

(b) Verification of part (a) in MATLAB, using `fft()` and `ifft()`

```
%% 2 part b

% init x(n) and y(n)
x = [3 0 -1 2];
y = [1 5 4 -2];

% use fft() and ifft() to compute circular convolution of x and y
ccirc = ifft(fft(x).*fft(y));
disp(ccirc);
```

```
>> ccirc =

     9    25     7    -9
```

The results are equivalent to those found in part (a).

(c) Computing the linear convolution of $x(n)$ and $y(n)$ using MATLAB

```
%% 2 part c

% init x(n) and y(n)
x = [3 0 -1 2];
y = [1 5 4 -2];

% use conv() to compute linear convolution of x and y
clin = conv(x, y);
disp(clin);
```

```
>> clin =

     3    15    11    -9     6    10    -4
```

(d) Computing the linear convolution of $x(n)$ and $y(n)$ using MATLAB `fft()` and `ifft()`

```
%% 2 part d

% init x(n) and y(n)
x = [3 0 -1 2];
y = [1 5 4 -2];

% pad x and y with appropriate 0's, both be length N
% where N = length(x) + length(y) -1
xpad = [x, zeros(1, length(y) - 1)];
ypad = [y, zeros(1, length(x) - 1)];

% use fft() and ifft() to compute linear convolution of x and y
clin_eq = ifft(fft(xpad).*fft(ypad));
disp(round(clin_eq));
```

```
>> clin_eq =

     3    15    11    -9     6    10    -4
```

The results are equivalent to those found in part (c).

3. (a) Input sequence $x(n)$ from `sampdata.m`

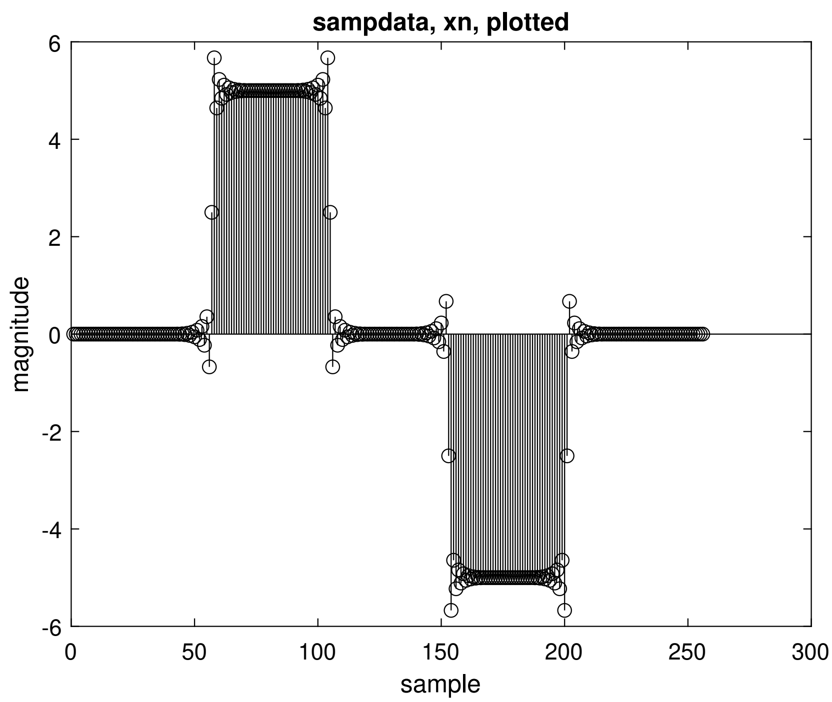

Figure 1: Stem plot of `sampdata`

(b) Created 32 coefficient FIR filter, as given in homework:

```
%% 3 part b
order = 32;
ws = 0.749;
wc = 0.85*ws;
F = [0.0 wc ws 1.0];
A = [1.0 0.95 0.01 0.0];
b = firpm(order, F, A);
```

(c) Plotting magnitude and phase response to filter `b` made in (b)

```
%% 3 part c
freqz(b);
```

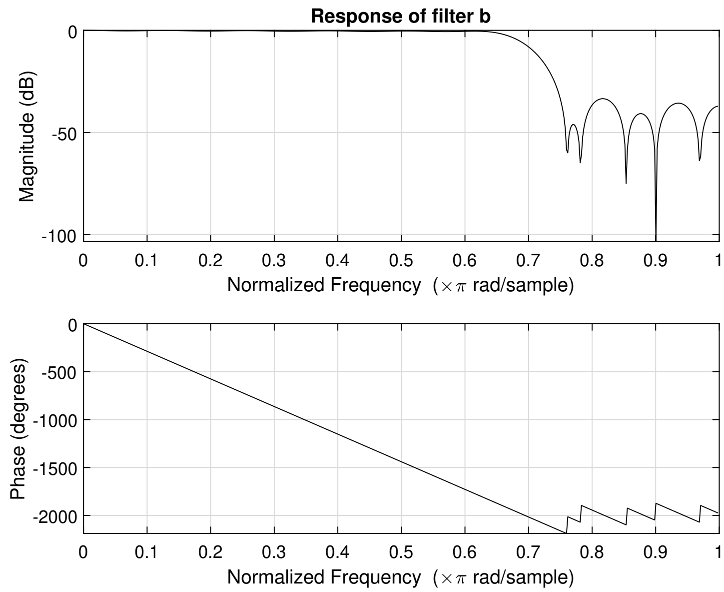

Figure 2: Magnitude and phase response of filter `b`

(d) Using MATLAB `conv` to calculate `y1`, output of filter when input $x(n)$ is applied

```
%% 3 part d
y1 = conv(xn, b); % where xn = sampdata
stem(y1);
```

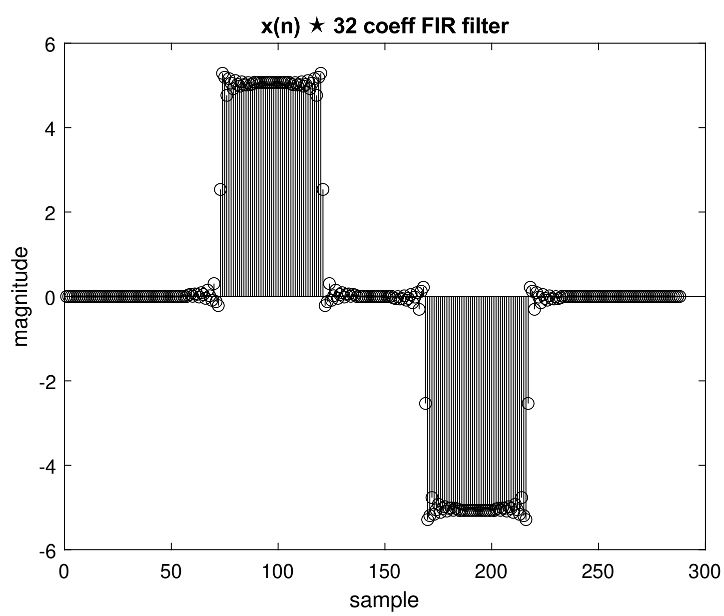

Figure 3: y1, the convolution of input and FIR filter, plotted using stem

(e) Similar to 2 (d), we will be performing linear convolution of the input sequence and the filter using IDFT and DFT algorithms provided by MATLAB

```
%% 3 part e
% appropriate padding, so N = Q + M - 1
xn_pad = [xn, zeros(1, length(b) - 1)];
b_pad = [b, zeros(1, length(xn) - 1)];

% linear convolution using ifft/fft, then plotting
y2 = ifft(fft(xn_pad).*fft(b_pad));
stem(y2);
```

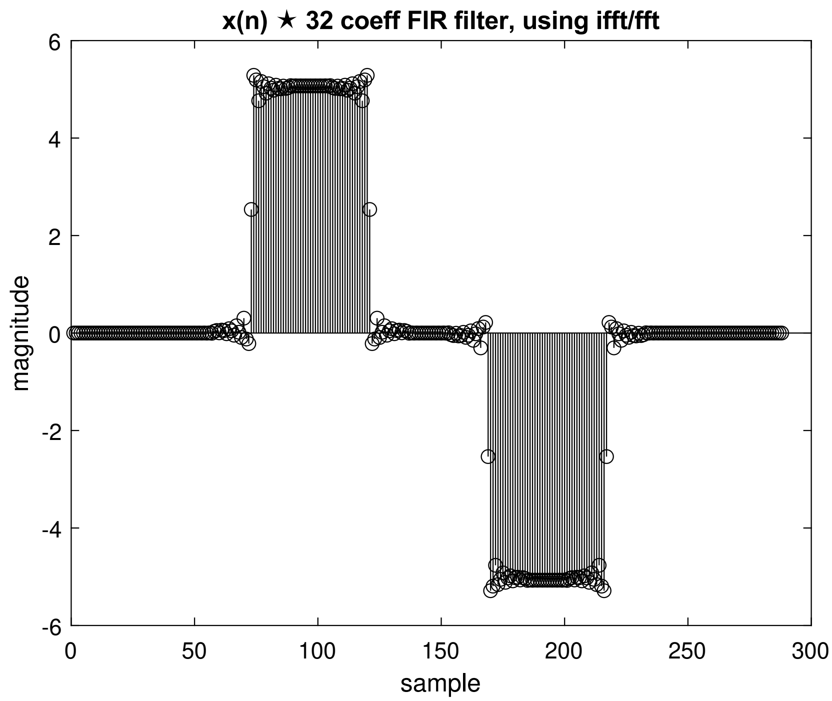

Figure 4: y2, the convolution of input and FIR filter, plotted using stem

(f) Calculate the absolute difference between `y1` and `y2`. Should be very small

```
%% 3 part f

stem(abs(y1 - y2));
```

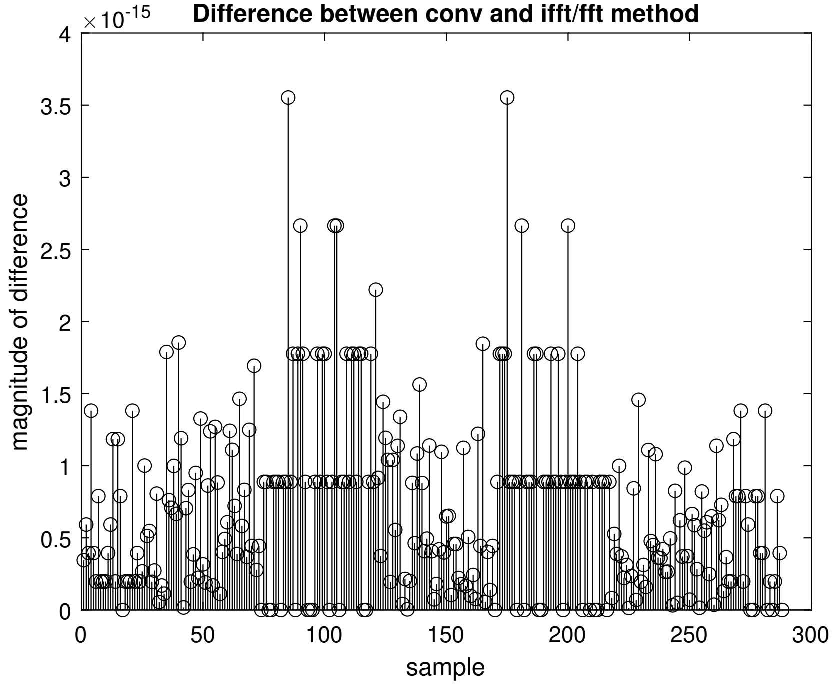

Figure 5: Absolute difference between `y1` and `y2`

4. Given $X(k)$ is the DFT sequence of $x(n)$, calculate the DFT sequences of $x_c(n)$ and $x_s(n)$ in terms of $X(k)$, where

$$x_c(n) = x(n)\cos\left(\frac{4\pi k_0 n}{N}\right), \quad 0 \le n \le N-1$$

and

$$x_s(n) = x(n)\sin\left(\frac{4\pi k_0 n}{N}\right), \quad 0 \le n \le N-1$$

First $x_c(n)$,

$$\begin{aligned} x_c(n) &= \frac{1}{2}x(n)\left[e^{\frac{j4\pi k_0 n}{N}} + e^{-\frac{j4\pi k_0 n}{N}}\right] \\ &= \frac{1}{2}x(n)e^{\frac{j2\pi(2k_0)n}{N}} + \frac{1}{2}x(n)e^{\frac{j2\pi(-2k_0)n}{N}} \end{aligned}$$

$$x_c(n) \Longleftrightarrow X_c(k) = \frac{1}{2}X(k-2k_0) + \frac{1}{2}X(k+2k_0)$$

Similarly,

$$\begin{aligned} x_s(n) &= \frac{1}{2j}x(n)\left[e^{\frac{j4\pi k_0 n}{N}} - e^{-\frac{j4\pi k_0 n}{N}}\right] \\ &= \frac{1}{2j}x(n)e^{\frac{j2\pi(2k_0)n}{N}} - \frac{1}{2j}x(n)e^{\frac{j2\pi(-2k_0)n}{N}} \end{aligned}$$

$$x_s(n) \Longleftrightarrow X_s(k) = \frac{1}{2j}X(k-2k_0) - \frac{1}{2j}X(k+2k_0)$$

5. There are 31912 samples of a speech audio waveform, given by Q, and we wish to use the overlap-save method filtering procedure with an FIR filter with 130 coefficients, given by M. N is restricted to a power of 2.

   (a) Without using the overlap-save method, the sampled speech and FIR filter must be linearly convoluted, which implies 0-padding. And since N must be a power of 2, $N \geq Q + M - 1 = 32042$. The closest power of 2 is $2^{15} = 32768 = N$. The number of complex multiplications is given by N^2, meaning there must be 2^{30} complex multiplications performed when applying the FFT algorithm.

   (b) Number of complex multiplications required using the overlap-save method with block size of 512 samples.
   The number of complex multiplications for the overlap-save method with any given block size is given in the following form

   $$n_{mult} = B[2N \log_2(N) + N] + N \log_2(N)$$

   where N is now the block size, in samples, and B is the number of blocks. The number of blocks is determined by

   $$B = \left\lceil \frac{Q}{N - M + 1} \right\rceil$$

   In our case,

   $$B = \left\lceil \frac{31912}{512 - 131 + 1} \right\rceil = 84$$

   So that

   $$n_{mult} = 84[1024 \log_2(512) + 512] + 512 \log_2(512) = 821760$$

   (c) Number of complex multiplications required using the overlap-save method with block size of 1024 samples.
   Similar to (b),

   $$B = \left\lceil \frac{31912}{1024 - 131 + 1} \right\rceil = 36$$

   So that

   $$n_{mult} = 36[2048 \log_2(1024) + 1024] + 1024 \log_2(1024) = 784384$$

   (d) Number of complex multiplications required using the overlap-save method with block size of 2048 samples.
   Similar to (b) and (c),

   $$B = \left\lceil \frac{31912}{2048 - 131 + 1} \right\rceil = 17$$

   So that

   $$n_{mult} = 17[4096 \log_2(2048) + 2048] + 2048 \log_2(2048) = 823296$$

   (e) There seems to be a minimum of this non-differentiable $f(N, Q, M)$, meaning that optimizing the overlap-save is certainly possible. Though an analytical method of finding the minimum would prove to be difficult, numerical methods can easily calculate the minimum of the function, and thus the optimal block size that results in the least complex multiplications. For our values of Q and M, the ideal block size would be 970.