Solutions: Homework Set # 6

Problem 1

(a) Using the inverse DTFT formula, we can compute the impulse response as

$$\begin{aligned} h[n] &= \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{j\omega}e^{j\omega n}d\omega = \frac{1}{2\pi}\int_{-\frac{\pi}{3}}^{\frac{\pi}{3}} e^{j\omega n}d\omega \\ &= \frac{1}{2\pi jn}e^{j\omega n}\Big|_{-\frac{\pi}{3}}^{\frac{\pi}{3}} = \frac{\sin(\frac{\pi}{3}n)}{\pi n} = \frac{1}{3}\operatorname{sinc}\left(\frac{n}{3}\right). \end{aligned}$$

(b) We want to minimize $\|H(e^{j\omega}) - H_K(e^{j\omega})\|_2$. Using the Parseval's Theorem, we have

$$\begin{aligned} \|H(e^{j\omega}) - H_K(e^{j\omega})\|_2^2 &= \frac{1}{2\pi}\int_{-\pi}^{\pi} |H(e^{j\omega}) - H_K(e^{j\omega})|^2 = \sum_n |h[n] - h_K[n]|^2 \\ &= \sum_{n:|n|\le K} |h[n] - h_K[n]|^2 + \sum_{n:|n|>K} |h[n] - h_K[n]|^2 \\ &= \sum_{n:|n|\le K} |h[n] - h_K[n]|^2 + \sum_{n:|n|>K} |h[n]|^2. \end{aligned}$$

Note that the second term does not depend on the choice of $h_K[n]$, and in order to minimize the first term, we have to choose $h_K[n] = h[n]$ (to make all terms in the summation equal zero). Thus

$$h_K[n] = \begin{cases} h[n] & |n| \le K \\ 0 & \text{else.} \end{cases} = \begin{cases} \frac{1}{3}\operatorname{sinc}(\frac{\pi n}{3}) & |n| \le K \\ 0 & \text{else.} \end{cases} =$$

(c) Here is a MATLAB file which produces the vector $h[n]$ and computes the corresponding Fourier transform.

```
function [h,H]=sinc_approx(K)
n=-K:K;
h=sinc(n/3)/3;
w=-pi:0.01:pi;
H=0;
for k=-K:K
    H=H+h(k+K+1)*exp(-j*k*w);
end
plot(w,abs(H));
```

Note that one can write only the first part of the file, and then use `fft` to compute the Fourier transform. However, `fft` considers the vector $h[n]$ as a vector on $n = 1, 2, \ldots, 2K+1$. You need to compensate this shifting by a complex exponential multiplication in frequency domain.

(d) Fig. 1 shows the frequency response of the filter for different length. It can be seen that the larger length FIR filter gives better approximation of $H(e^{j\omega})$. We can see a jump(ripple) at the cut-off frequencies ($\omega = \pm\frac{\pi}{3}$), which corresponds to the Gibbs phenomenon. You can think of it as approximating a discontinuity with linear combination of continuous functions, which is impossible for any finite K.

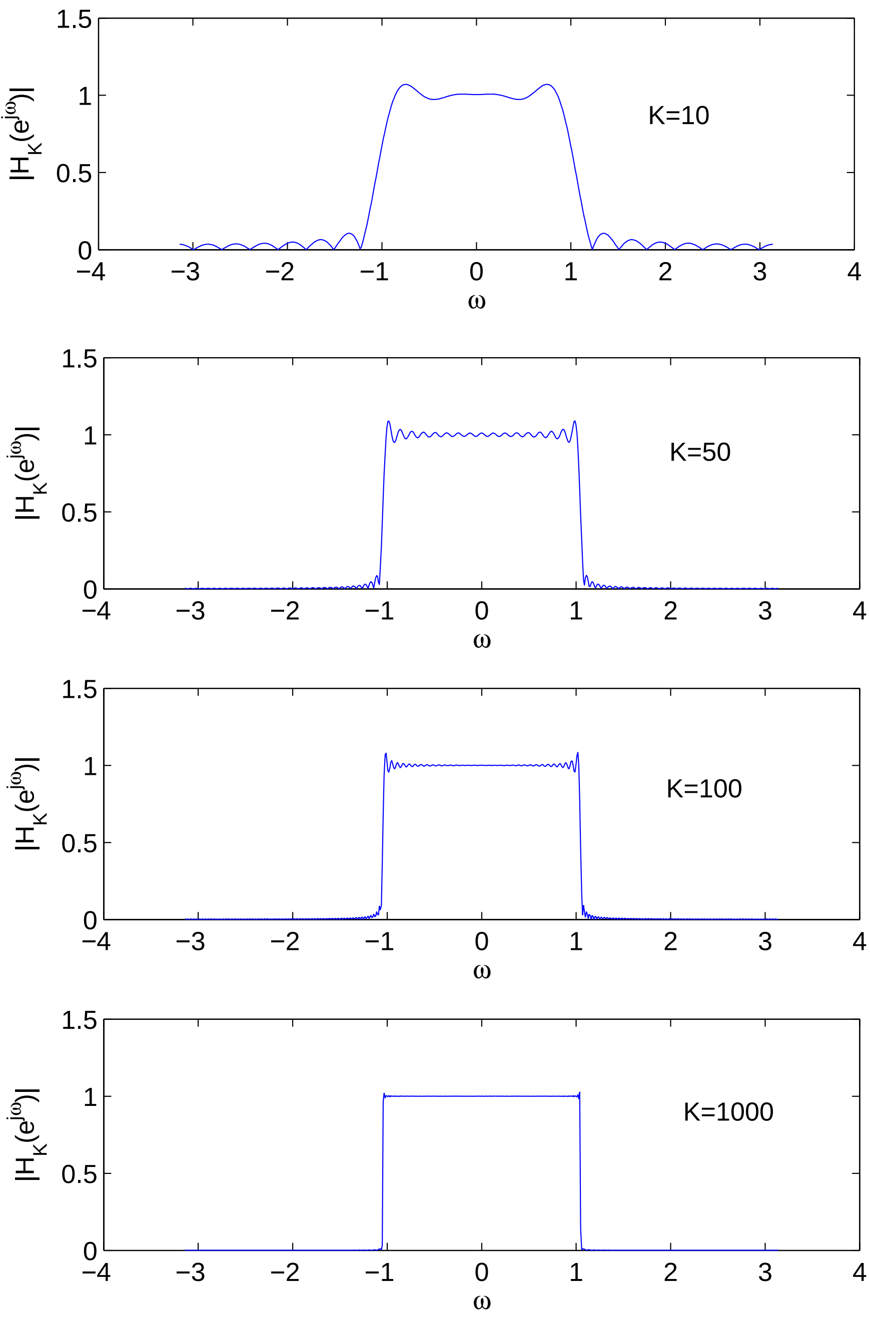

Figure 1: Approximation of ideal low-pass filter with FIR filters of length $2K + 1$. .

Problem 2 (Weighted Least-Squares Filter Design)

(a) W is a diagonal matrix, with entry $(W)_{kk} = \delta_p/\delta_s$ if $\omega_k \in [0, \omega_s]$, $(W)_{kk} = 1$ if $\omega_k \in [\omega_p, \pi]$, and $(W)_{kk} = 0$ otherwise.

$$U = \begin{bmatrix} 1 & \cos\omega_1 & \dots & \cos(\frac{M-1}{2}\omega_1) \\ 1 & \cos\omega_2 & \dots & \cos(\frac{M-1}{2}\omega_2) \\ & \ddots & & \\ 1 & \dots & & \cos(\frac{M-1}{2}\omega_k) \end{bmatrix}.$$

$$d_k = \begin{cases} 0 & \text{if } \omega_k \in [0, \omega_s] \\ 1 & \text{if } \omega_l \in [\omega_p, \pi] \\ \text{any} & \text{otherwise} \end{cases}$$

(b) We can write the condition as

$$\begin{aligned} \mathbf{e} &\le \delta_p \mathbf{1} \\ -\mathbf{e} &\le \delta_p \mathbf{1}, \end{aligned}$$

where $\mathbf{1} = [1 \dots 1]^{\mathrm{T}}$. From $\mathbf{e} = W(\mathbf{d} - U\mathbf{h})$, we get

$$\begin{aligned} -WU\mathbf{h} &\le \delta_p \mathbf{1} - W\mathbf{d} \\ WU\mathbf{h} &\le -(\delta_p \mathbf{1} - W\mathbf{d}), \end{aligned}$$

and so we find

$$A = \begin{bmatrix} -WU \\ WU \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} \delta_p \mathbf{1} - W\mathbf{d} \\ -(\delta_p \mathbf{1} - W\mathbf{d}) \end{bmatrix}.$$

(c)

Problem 3

(a) You should hear a snippet of the song "All You Need is Love" by the Beatles. For the plot of the DFT of "song", type

```
>> plot(abs(fft(song)));
```

The result is given in Figure 2. If we assume that the DTFT is defined in the interval $[-\pi, \pi]$, then the right half of the plot corresponds actually to negative frequencies. The labels of the points 0, π and $-\pi$ in Figure 2 indicate this fact. For a more convenient representation, one can use the function *fftshift*, which converts the DFT of a discrete sequence into a form which is suitable for plotting:

```
>> plot(abs(fftshift(fft(song))));
```

or, if you want to label the ω-axis correctly:

```
>> plot(linspace(-pi,pi,length(song)),abs(fftshift(fft(song))));
>> xlabel('\omega');
>> ylabel('|X(e^{j \omega})|');
```

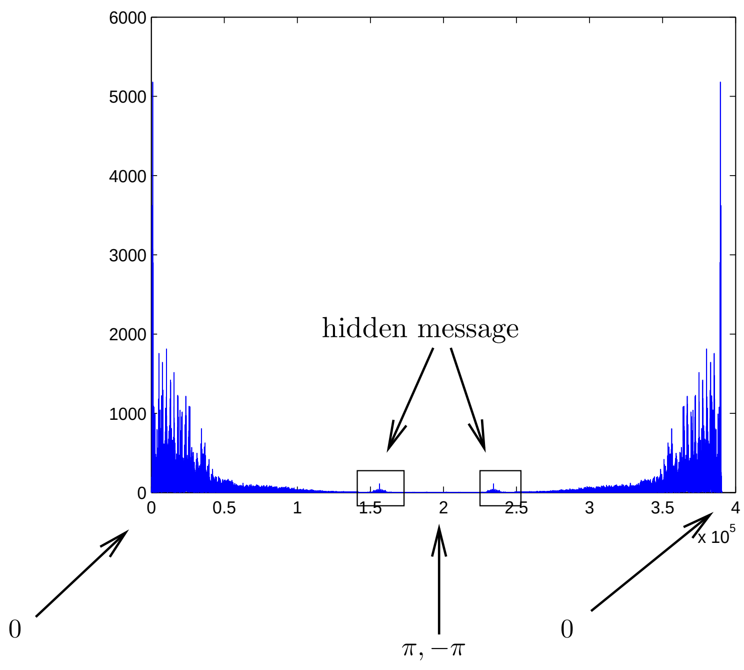

Figure 2: DFT of the sequence "song".

The result is shown in Figure 3.

(b) Yes, one can see it in the frequency-plot. The secret message is indicated in Figure 2. It appears that the energy of the hidden message is contained in the high-frequency range of the music sequence. Hence, a high-pass filter would extract the message, and cut away most of the energy of the song.

(c) By looking at Figure 3, we see that the band-edges $\omega_s = 0.6\pi$ and $\omega_p = 0.75\pi$ are fine for our purpose. Hence, we can use the filter provided by the file "wls.m" as-is. If your version of "wls.m" does not seem to work, you can generate a filter using the matlab function *firpm* (Parks-McClellan FIR filter):

```
>> [hpm, err]=firpm(40,[0 0.6 0.75 1],[0 0 1 1],[10,1]);
```

(d) Note that in Matlab, we can directly do the convolution of two sequences, even if they are of different length. The function *conv* will automatically zero-pad the shorter sequence before taking the convolution. We type

```
>> songFilt = conv(h,song);
>> plot(linspace(-pi,pi,length(songFilt)),abs(fftshift(fft(songFilt))));
>> xlabel('\omega');
>> ylabel('|X(e^{j \omega})|');
```

and obtain the plot in Figure 4. We can verify that the filter really cut off most of the low frequencies.

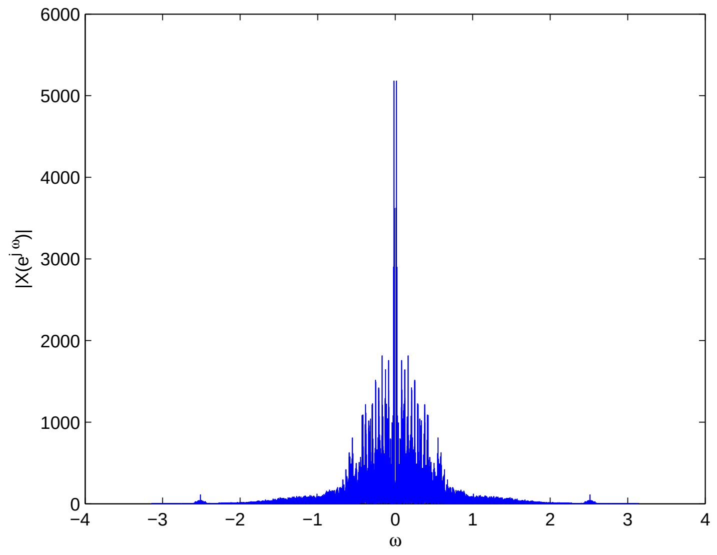

Figure 3: Nicer plot of the DFT of the sequence “song”.

(e) If we write the cosine as $\cos(0.8\pi n) = \frac{1}{2}(e^{j0.8\pi n} + e^{-j0.8\pi n})$, we see that the shifted signal is

$$m'[n] = \frac{1}{2}m[n]e^{j0.8\pi n} + \frac{1}{2}m[n]e^{-j0.8\pi n}.$$

By the shift property of the DTFT, we see that if $m'[n]$ was an infinite sequence, then the corresponding DTFT would be

$$M'(e^{j\omega}) = \frac{1}{2}M(e^{j(\omega-0.8\pi)}) + \frac{1}{2}M(e^{j(\omega+0.8\pi)}).$$

Here, we are actually dealing with the DFT. By the shift property of the DFT, and because $0.8\pi = \frac{2\pi}{N}\frac{0.8N}{2}$, we see that

$$M'[k] = \frac{1}{2}M[(k-0.4N) \mod N] + \frac{1}{2}M[(k+0.4N) \mod N].$$

This results in circularly shifting the two “halfs” of the spectrum of $m[n]$ by $0.4N = 156240$ samples in opposite directions.

(f) If we want to find the strict inverse of $m'[n]$, we would have

$$m[n] = \frac{m'[n]}{\cos(0.8\pi n)}.$$

A simpler inverse transformation is

$$\hat{m}[n] = m'[n]\cos(0.8\pi n),$$

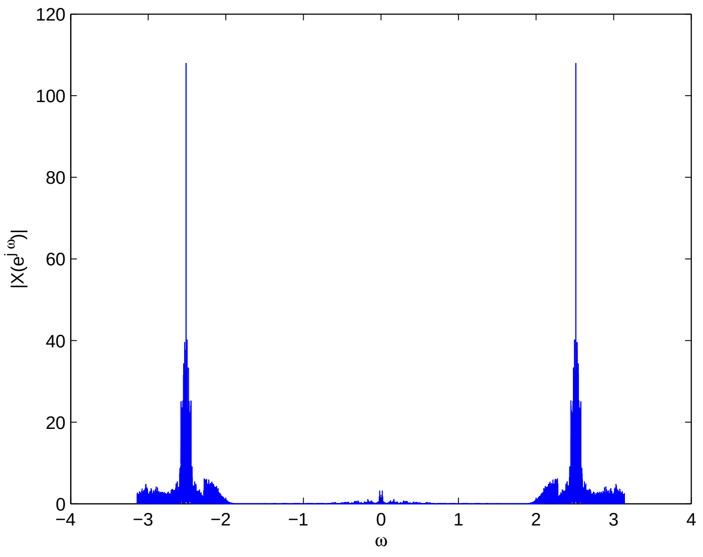

Figure 4: DFT of the sequence "songFilt".

i.e., we reapply the same cosine-shift. Note that $\hat{m}[n]$ is not exactly equal to $m[n]$, but in the low frequencies, $\hat{m}[n]$ and $m[n]$ are very similar. Figure 5 shows the spectrum of the original message $m[n]$. Figure 6 shows the spectrum of $m'[n]$. Finally, Figure 7 shows the spectrum of the reconstruction $\hat{m}[n]$. One can see that for $\omega \in [-\frac{\pi}{3}, \frac{\pi}{3}]$, the spectra in Figures 5 and 7 are quite similar.

Now, we use this operation on the sequence "songFilt":

```
>> n=1:length(songFilt);
>> songFiltShift = songFilt'.*cos(0.8*pi*n);
>> plot(linspace(-pi,pi,length(songFiltShift)),abs(fftshift(fft(songFiltShift))));
>> xlabel('\omega');
>> ylabel('|X(e^{j \omega})|');
```

The result is shown Figure 8. The result looks more or less like a sound spectrum, except for the high peaks at about $|\omega| = 1.3$. These peaks are there, because we used the cosine to shift the voice signal. (Compare with Figure 7.)

(g) You should be able to hear the message. It should say something about helicopters, grandmothers, birds etc...

(h) As mentioned before, the high-frequency noise peaks in Figure 8 are there because we used the cosine to shift the message to lower frequencies. In addition, there are some remainders of the music (very high frequencies like snare-drums), that have also been shifted down to low frequencies and can be heard as noise. We would use a low-pass filter to get rid of the noise.

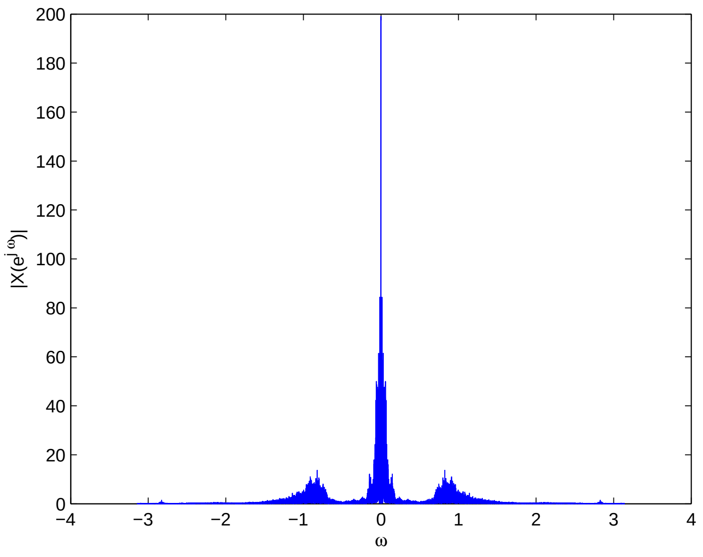

Figure 5: DFT of $m[n]$.

(i) In Figure 8, we see that the frequencies that interest us are more or less $\omega \in [-1, 1]$. This corresponds more or less to $|\omega| \leq 0.3\pi$. Hence, we would use *firpm* to design a low-pass filter that has a transition at more or less $\frac{0}{.}3\pi$:

```
>> [hpm, err]=firpm(40,[0 0.25 0.3 1],[1 1 0 0],[1,10]);
>> message = conv(songFiltShift,hpm);
>> plot(linspace(-pi,pi,length(message)),abs(fftshift(fft(message))));
>> xlabel('\omega');
>> ylabel('|X(e^{j \omega})|');
```

The corresponding plot is shown in Figure 9. We see that the high-frequency noise has disappeared. If we play the message, it can be understood more easily.

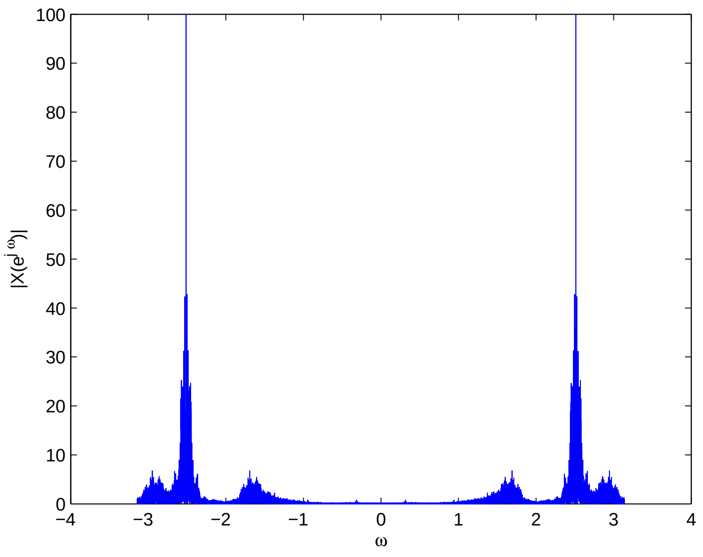

Figure 6: DFT of $m'[n]$.

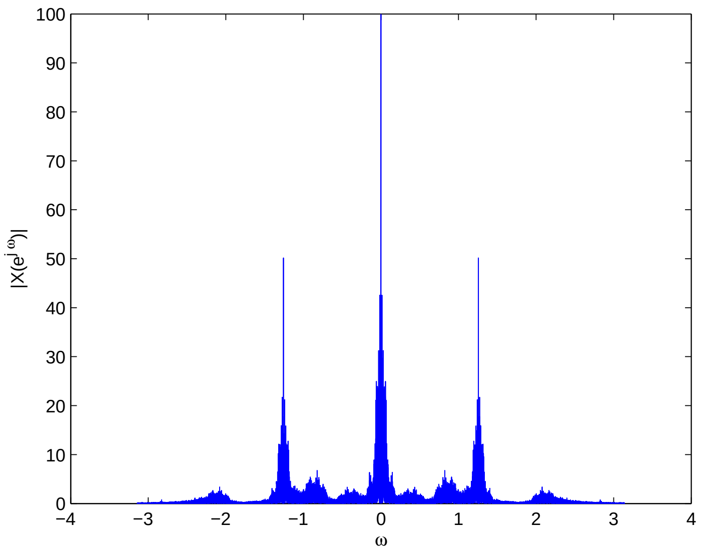

Figure 7: DFT of $\hat{m}[n]$.

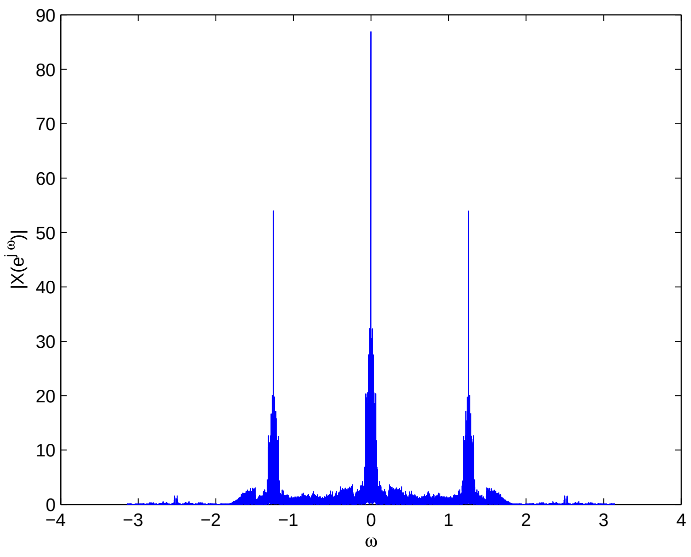

Figure 8: DFT of the sequence “songFiltShift”.

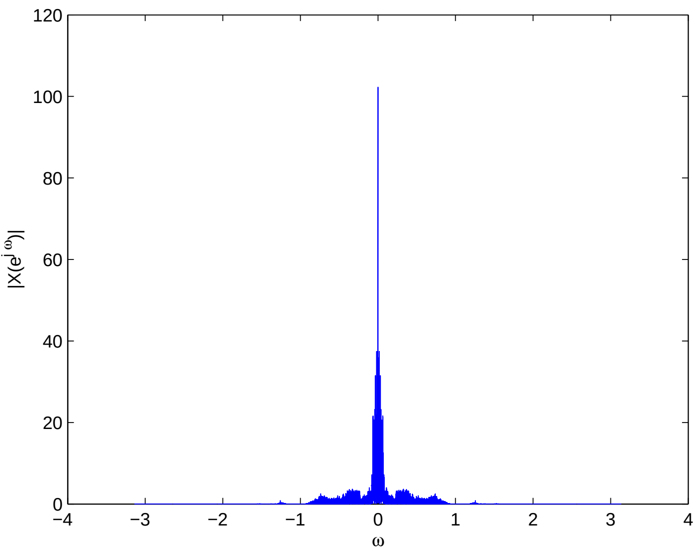

Figure 9: DFT of the sequence “message”.