Academic year
2024-2025

Faculty of Applied Engineering

Digital Signal Processing

Signals & Transforms – Solutions to the Exercises

Walter Daems

Cursusdienst
UNIVERSITAS
Prinsesstraat 16
2000 Antwerpen
T +32 3 233 23 72
F +32 3 233 65 81
E info@cursusdienst.be
W www.universitas.be

Bachelor of Science in de Industriële Wetenschappen

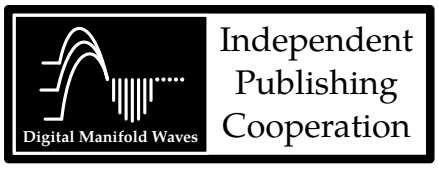
1514FTIDSP 6-Digital Signal Processing

University of Antwerp

This document has been typeset using LaTeX and the `uantwerpendocs` package.
This text is a special version for students suffering dyslexia. It has been typeset using the Sylexiad font. Please give feedback on your experience with this version using e-mail to walter.daems@uantwerpen.be.

Calculations have been performed using Matlab/Octave and generic programming and script languages (C/C++/Perl/Python).
Graphics have been composed using PGF, TiKZ and InkScape.
All this material has been prepared on a GNU/Linux workstation.

Typesetting of this document was enabled by:



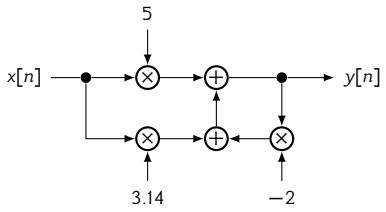
DSP-ST-2024-3.10-TB

Contents

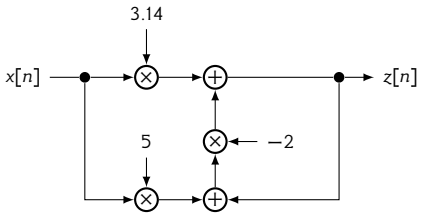
Chapter 2

Signals

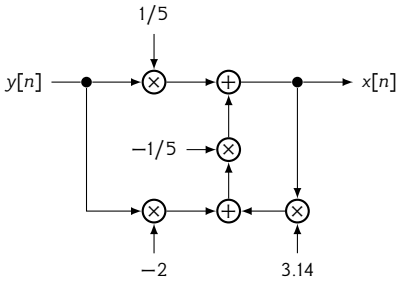
Solution 2.4.2-1:



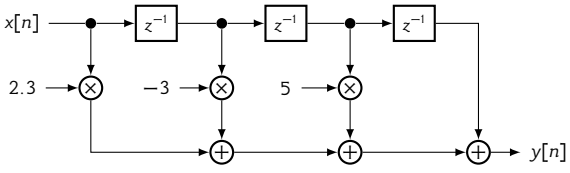
Solution 2.4.2-2:



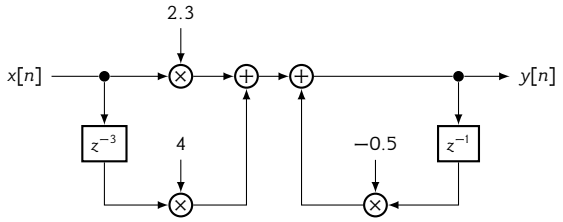
Solution 2.4.2-3:



Solution 2.4.2-4:



Solution 2.4.2-5:



Solution 2.4.2-6:

$$y[n] = a\,(x[n] - by[n-1])$$

Solution 2.4.2-7:

$$z[n] = x[n] + b_1 z[n-1] + b_2 z[n-2]$$
$$y[n] = a_0 z[n] + a_1 z[n-1] + a_2 z[n-2]$$

Solution 2.4.2-8:

$$z_2[n] = a_2 x[n-1] + b_2 y[n-1]$$
$$z_1[n] = a_1 x[n-1] + b_1 y[n-1] + z_2[n-1]$$
$$y[n] = a_0 x[n] + z_1[n]$$

Solution 2.6-1: Let's start with the observation that $x(t)$ is periodic with a period that is the smallest common multiple of the periods of the composing sine (with period $\pi/10$) and cosine (with period $\pi/5$). The period of $x(t)$ therefore is $\pi/5$.

If you don't care about a good exercise in basic algebra, you might want to skip 'the hard way' and continue with 'the easy way'.

The hard way Peak-to-peak value

x(t) reaches its extremes whenever $\mathrm{d}x(t)/\,\mathrm{d}t = 0$. Therefore, we solve:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = 0$$
$$6\cos(20t) + 2.5\sin(10t) = 0$$
$$\Big\downarrow \quad \cos(20t) = 1 - 2\sin^2(10t)$$
$$6 - 12\sin^2(10t) + 2.5\sin(10t) = 0$$

This is a quadratic form in $\sin(10t)$, and therefore:

$$\sin(10t) = \frac{-2.5 \pm \sqrt{2.5^2 - 4.(-12).6}}{2.(-12)} = \frac{-2.5 \pm \sqrt{294.25}}{-24} = \begin{cases} -0.61057 \\ 0.81890 \end{cases}$$

And therefore, we obtain:

$$t = \frac{1}{10}\begin{cases} \arcsin(-0.61057) \\ \arcsin(0.81890) \end{cases}$$

$$t = \frac{1}{10}\begin{cases} \begin{cases} 0.95950 + 2k\pi \\ \pi - 0.9590 + 2k\pi \end{cases} \\ \begin{cases} -0.65678 + 2k\pi \\ \pi + 0.65678 + 2k\pi \end{cases} \end{cases} \quad \text{with } k \text{ integer}$$

This leads to four primary extrema in the points $t = 0.095950, -0.065678, 0.218209$ and 0.379837. If we fill these points into the original function, we become the local extrema: $x(t) = 0.13851, -0.48812, -0.13851$ and 0.48812. This leads to the peak to peak value:

$$x_{PTP} = 0.48812 - (-0.48812) = 0.97624$$

Mean value
The mean value is easily obtained, as

$$\begin{aligned} x_{MEAN} &= \frac{1}{T}\int_0^T x(t)\,\mathrm{d}t \\ &= \frac{1}{\pi/5}\int_0^{\pi/5} (0.3\sin(20t) - 0.25\cos(10t))\,\mathrm{d}t \\ &= 0 \end{aligned}$$

The latter conclusion is easily drawn, knowing that the mean value of a sine and cosine over a multiple number of periods is zero.

Variance

$$\begin{aligned} x_{VAR}^2 &= \frac{1}{T}\int_0^T (x(t) - x_{MEAN})^2\,\mathrm{d}t \\ &= \frac{1}{\pi/5}\int_0^{\pi/5} (0.3\sin(20t) - 0.25\cos(10t))^2\,\mathrm{d}t \\ &= \frac{1}{\pi/5}\int_0^{\pi/5} \Big(0.09\sin^2(20t) - 0.15\sin(20t)\cos(10t) + 0.0625\cos^2(10t)\Big)\,\mathrm{d}t \end{aligned}$$

As we know (a) that the variance of a pure sine or cosine wave over an integral number of periods to be equal to 0.5 and (b) the integral of a product of sine and cosine wave over an integral number of periods is zero, we easily can simplify this further:

$$\begin{aligned} x_{VAR}^2 &= 0.045 - 0 + 0.03125 \\ &= 0.07625 \end{aligned}$$

RMS-value
The RMS value is easily obtained as it equals the square root of the variance, because the mean value equals zero. Therefore:

$$\begin{aligned} x_{RMS} &= \sqrt{x_{VAR}^2} \\ &= 0.27613 \end{aligned}$$

The easy way Though the above is a good exercise in algebra, it takes a while, whereas using Matlab/Octave, the result can more easily be obtained.

A graph of $x(t)$ covering the overall period $\pi/5$ is easily generated using:

```
t = linspace( 0, pi/5, 1000 );
function y = x( t )
  y = 0.3 * sin( 20 .* t ) - 0.25 * cos( 10.* t );
end
plot( t, x(t) );
```

Peak-to-peak value
The peak-to-peak value is easily derived:

```
xptp = max(x(t)) - min(x(t))
==> xptp =  0.97623
```

Of course, the correctness of the result depends on the resolution imposed by N. Increasing N will improve the correctness, but be aware that brute force comes at a price: calculation time.

Mean value

```
xmean = mean( x(t) )
==> xmean =  -2.5000e-04
```

Again, the same remark regarding the value of N can be made.

Variance

```
xvarsq = mean( ( x(t) - xmean ).^2 )
==> xvarsq =  0.076236
```

Note that `xvarsq` conrresponds to x_{VAR}^2.

RMS-value

```
xrms = sqrt( xvarsq + xmean^2 )
==> xrms =  0,27611
```

Solution 2.6-2: The problem with discrete periodic signals is that though their analog counterpart is periodic, the signal is sampled such that the periodicity does not reveal itself. This exercise poses such a case. Stated more clearly: the signal $x[n]$ is not periodic. Often it is said that $x[n]$ is pseudo-periodic with pseudo-period $\pi/5$, i.e. not an integer. In this case, the only thing to do is take a lot of pseudo-periods into account.

A graph of $x[n]$ is easily generated using:

```
N = 1000;
n = linspace(0, N, N+1);
function y = x( n )
  y = 0.3 * sin( 20 .* n ) - 0.25 * cos( 10 .* n );
end
plot( n, x(n) );
```

Peak-to-peak value
The peak-to-peak value is easily derived:

```
xptp = max(x(n)) - min(x(n))
==> xptp =  0.97526
```

Of course, the correctness of the result depends on N. Ideally N should approach infinity.

Mean value

```
xmean = mean( x(n) )
==> xmean =  1.1312e-04
```

Variance

```
xvarsq = mean( ( x(n) - xmean ).^2 )
==> xvarsq =  0.076386
```

Note that `xvarsq` conrresponds to x_{VAR}^2. Please, also note that we used the mean function here, which is not fully correct compared to the exact definition (see course notes), but for large N the deviation is neglectable.

RMS-value

```
xrms = sqrt( xvarsq + xmean^2 )
==> xrms =  0.27638
```

Solution 2.6-3: Let's start by entering $x[n]$ in Octave/Matlab.

```
x = [ -0.3 0.4 0.2 -0.3 0.5 0.7 0.25 0.1 -0.3 -0.4 -0.2 0 ];
```

Peak-to-peak value
The peak-to-peak value is easily derived:

```
xptp = max(x) - min(x)
==> xptp =  1.1000
```

Mean value

```
xmean = mean( x )
==> xmean =  0.054167
```

Variance

```
xvarsq = mean( ( x - xmean ).^2 )
==> xvarsq =  0.12061
```

Note that `xvarsq` conrresponds to x_{VAR}^2. Pease, note also that using the 'mean' function is fully correct here, as the signal is not a sampling of a larger population. It is the full signal to be analyzed.

RMS-value

```
xrms = sqrt( xvarsq + xmean^2 )
==> xrms =  0.35148
```

Solution 2.8.4-1: The formulae to determine the even/odd decomposition are straightforward:

$$f_e(t) = \frac{f(t) + f(-t)}{2}$$
$$f_o(t) = \frac{f(t) - f(-t)}{2}$$

and hence,

$$x_e(t) = -0.2\frac{\sin(20t) + \sin(-20t)}{2}$$
$$x_o(t) = -0.2\frac{\sin(20t) - \sin(-20t)}{2}$$

As the sine function itself is odd (i.e. $\sin(-x) = -\sin(x)$), we easily obtain:

$$x_e(t) = -0.1(\sin(20t) - \sin(20t)) = 0$$
$$x_o(t) = -0.1(\sin(20t) + \sin(20t)) = -0.2\sin(20t)$$

This result could have been obtained more easily by observing that the orginal function was odd to start with.

Solution 2.8.4-2: The formulae to determine the even/odd decomposition are straightforward:

$$f_e[n] = \frac{f[n] + f[-n]}{2}$$
$$f_o[n] = \frac{f[n] - f[-n]}{2}$$

and hence,

$$x_e[n] = \pi\frac{\cos[5n] + \cos[-5n]}{2}$$
$$x_o[n] = \pi\frac{\cos[5n] - \cos[-5n]}{2}$$

As the cosine function itself is even (i.e. $\cos(-x) = \cos(x)$), we easily obtain:

$$x_e[n] = \pi \frac{\cos[5n] + \cos[5n]}{2} = \pi \cos[5n]$$
$$x_o[n] = \pi \frac{\cos[5n] - \cos[5n]}{2} = 0$$

This result could have been obtained more easily by observing that the orginal function was even to start with.

Solution 2.8.4-3: The formulae to determine the even/odd decomposition are straightforward:

$$f_e[n] = \frac{f[n] + f[-n]}{2}$$
$$f_o[n] = \frac{f[n] - f[-n]}{2}$$

and hence $x_e[n]$ can be easily found to be:

$$\begin{aligned} x_e[n] = \frac{1}{2}(& [-0.3, 0.4, 0.2, -0.3, 0.5, 0.7, \underbrace{0.25}_{n=0}, 0.1, -0.3, -0.4, -0.2, 0, -0.2] \\ & + [-0.2, 0, -0.2, -0.4, -0.3, 0.1, \underbrace{0.25}_{n=0}, 0.7, 0.5, -0.3, 0.2, 0.4, -0.3]) \\ = & [-0.25, 0.2, 0, -0.35, 0.1, 0.4, \underbrace{0.25}_{n=0}, 0.4, 0.1, -0.35, 0, 0.2, -0.25] \end{aligned}$$

and $x_o[n]$ can be easily found to be:

$$\begin{aligned} x_o[n] = \frac{1}{2}(& [-0.3, 0.4, 0.2, -0.3, 0.5, 0.7, \underbrace{0.25}_{n=0}, 0.1, -0.3, -0.4, -0.2, 0, -0.2] \\ & - [-0.2, 0, -0.2, -0.4, -0.3, 0.1, \underbrace{0.25}_{n=0}, 0.7, 0.5, -0.3, 0.2, 0.4, -0.3]) \\ = & [-0.05, 0.2, 0.2, 0.05, 0.4, 0.3, \underbrace{0}_{n=0}, -0.3, -0.4, -0.05, -0.2, -0.2, 0.05] \end{aligned}$$

Chapter 3

The Fourier transform

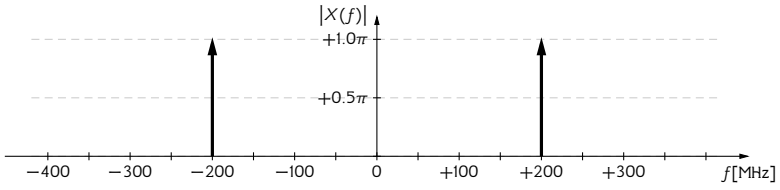
Solution 3.3.3-1: The maximum frequency that appears in the spectrum of $X(\omega)$ amounts to $\pm 75\,\text{Mrad}$. Therefore, according to Shannon's time-domain sampling theorem, we need to sample the signal at a rate ω_s of:

$$\omega_s = 2\omega_B = 2 \cdot 75\,\text{Mrad/s} = 150\,\text{Mrad/s} \sim 23.873\,\text{MHz}$$

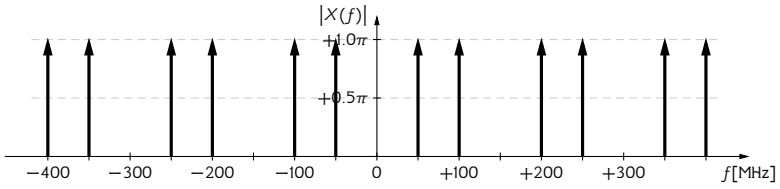
Solution 3.3.3-2: According to Shannon's time-domain sampling theorem, the maximal bandwidth f_B of the signal that can be sampled at a rate f_s without information loss equals:

$$f_B = \frac{f_s}{2} = \frac{44\,100\,\text{Hz}}{2} = 22\,050\,\text{Hz}$$

Solution 3.3.4-1: The (Fourier) spectrum before sampling looks like:



The (Fourier) spectrum after sampling looks like:



Solution 3.3.4-2: What your best friend hears is an alias of the whistle frequency f_w. This alias is due to the sampling at a frequency $f_s = 8\,\text{kHz}$. The aliases of the whistle's frequency are located at:

$$f_{alias,k} = f_w + k f_s \qquad \text{with } k \in \mathbb{Z}$$

The alias frequency within the cell phone's primary frequency range $[-f_s/2, f_s/2]$ is:

$$\begin{aligned} f_{alias,k} &= f_w - f_s \\ &= 8.25 \times 10^3\,\text{Hz} - 8 \times 10^3\,\text{Hz} = 250\,\text{Hz} \end{aligned}$$

Solution 3.4.3-1: A sampling rate $f_s = 10\,\text{kHz}$ corresponds to a sampling period $T_s = 1/f_s = 100\,\mu\text{s}$. To be able to sample the full 2 seconds, we need 20 000 samples, i.e. $N = 20 \times 10^3$
Therefore, the appropriate ω_f equals:

$$\omega_f = \frac{\omega_S}{N} = \frac{2\pi f_s}{N} = \frac{2\pi 10 \times 10^3\,\text{Hz}}{20 \times 10^3} = \pi\,\text{rad/s}$$

Solution 3.4.3-2: The maximal signal time length expressed as a number of points N is related to the sample rate f_S and the frequency pitch f_f by:

$$N = \frac{f_s}{f_f} = \frac{1 \times 10^6\,\text{Hz}}{1 \times 10^4\,\text{Hz}} = 100$$

This covers a time length of $100 \times 1\,\mu\text{s} = 0.1\,\text{ms}$.

Solution 3.5.1-1: Let's start with the right hand of the equation, but write $x(-t)$ as $y(t)$:

$$\begin{aligned}
x(t) \star y(t) &= \int_{-\infty}^{+\infty} x(\tau) y(t-\tau)\,\mathrm{d}\tau \\
&\downarrow\ y(t) = x(-t) \\
x(t) \star x(-t) &= \int_{-\infty}^{+\infty} x(\tau) x(-(t-\tau))\,\mathrm{d}\tau \\
&= \int_{-\infty}^{+\infty} x(\tau) x(\tau - t))\,\mathrm{d}\tau \\
&\downarrow \text{substitute: } \alpha = \tau - t \\
&= \int_{-\infty}^{+\infty} x(\alpha + t) x(\alpha)\,\mathrm{d}\alpha \\
&= x(t) \star x(t)
\end{aligned}$$

Solution 3.5.1-2: Let's apply the definition of the Fourier transform:

$$\begin{aligned}
\mathcal{F}\left(x(-t)\right) &= \int_{-\infty}^{+\infty} x(-t)\,\mathrm{e}^{-\mathrm{j}\omega t}\,\mathrm{d}t \\
&\downarrow \text{substitute } \tau = -t \\
&= \int_{-\infty}^{+\infty} x(\tau)\,\mathrm{e}^{\mathrm{j}\omega\tau}\,\mathrm{d}\tau \\
&= \int_{-\infty}^{+\infty} x(\tau)\,\overline{\mathrm{e}^{-\mathrm{j}\omega\tau}}\,\mathrm{d}\tau \\
&= \int_{-\infty}^{+\infty} \overline{x(\tau)\,\mathrm{e}^{-\mathrm{j}\omega\tau}}\,\mathrm{d}\tau \\
&= \overline{\int_{-\infty}^{+\infty} x(\tau)\,\mathrm{e}^{-\mathrm{j}\omega\tau}\,\mathrm{d}\tau} \\
&\downarrow \text{Assume: } x(t) \xrightarrow{\mathcal{F}} X(\omega) \\
&= X^*(\omega)
\end{aligned}$$

Note that for notational clarity we used the equivalent notation: $x^* = \bar{x}$, both denoting the complex conjugate of x.

Solution 3.5.1-3: Let's start with the right hand of the equation, but write $x^*(-t)$ as $y(t)$:

$$\begin{aligned}
x(t) \star y(t) &= \int_{-\infty}^{+\infty} x(\tau) y(t-\tau)\,\mathrm{d}\tau \\
&\downarrow\ y(t) = x^*(-t) \\
x(t) \star x^*(-t) &= \int_{-\infty}^{+\infty} x(\tau) x^*(-(t-\tau))\,\mathrm{d}\tau \\
&= \int_{-\infty}^{+\infty} x(\tau) x^*(\tau - t))\,\mathrm{d}\tau \\
&\downarrow \text{substitute: } \alpha = \tau - t \\
&= \int_{-\infty}^{+\infty} x(\alpha + t) x^*(\alpha)\,\mathrm{d}\alpha \\
&= x(t) \star x^*(t)
\end{aligned}$$

Solution 3.5.1-4: Let's apply the definition of the Fourier transform:

$$\begin{aligned}
\mathcal{F}\left(x^*(-t)\right) &= \int_{-\infty}^{+\infty} x^*(-t)\,\mathrm{e}^{-\mathrm{j}\omega t}\,\mathrm{d}t \\
&\downarrow \text{substitute } \tau = -t \\
&= \int_{-\infty}^{+\infty} x^*(\tau)\,\mathrm{e}^{\mathrm{j}\omega\tau}\,\mathrm{d}\tau \\
&= \int_{-\infty}^{+\infty} x^*(\tau)\,\overline{\mathrm{e}^{-\mathrm{j}\omega\tau}}\,\mathrm{d}\tau \\
&= \int_{-\infty}^{+\infty} \overline{x(\tau)\,\mathrm{e}^{-\mathrm{j}\omega\tau}}\,\mathrm{d}\tau \\
&= \overline{\int_{-\infty}^{+\infty} x(\tau)\,\mathrm{e}^{-\mathrm{j}\omega\tau}\,\mathrm{d}\tau} \\
&\downarrow \text{Assume: } x(t) \xrightarrow{\mathcal{F}} X(\omega) \\
&= X(\beta) = X^*(\omega)
\end{aligned}$$

Note that for notational clarity we used the equivalent notation: $x^* = \overline{x}$, both denoting the complex conjugate of x.

Solution 3.5.2-1: Let's start with the right hand of the equation, but write $x[-n]$ as $y[n]$ (periodic with period N as well):

$$\begin{aligned}
x[n] \star y[n] &= \sum_{m=0}^{N-1} x[m]y[n-m] \\
&\downarrow y[n] = x[-n] \\
x[n] \star x[-n] &= \sum_{m=0}^{N-1} x[m]x[-(n-m)] \\
&= \sum_{m=0}^{N-1} x[m]x[m-n] \\
&\downarrow \text{substitute: } l = m - n \\
&= \sum_{-n}^{N-n-1} x[l+n]x[l] \\
&\downarrow x[n] \text{ is periodic with period } N \\
&= \sum_{0}^{N-1} x[l+n]x[l] \\
&= x[n] \star x[n]
\end{aligned}$$

Solution 3.5.2-2: Let's apply the definition of the discrete Fourier transform:

$$\begin{aligned}
\text{DFT}\,(x[-n]) &= \sum_{n=0}^{N-1} x[-n]\, e^{-j\frac{2\pi kn}{N}} \\
&\downarrow \text{substitute } m = -n \\
&= \sum_{m=-(N-1)}^{0} x[m]\, e^{j\frac{2\pi km}{N}} \\
&\downarrow x[n] \text{ is periodic with period } N \\
&= \sum_{m=0}^{N-1} x[m]\, e^{j\frac{2\pi km}{N}} \\
&= \sum_{m=0}^{N-1} x[m]\, \overline{e^{-j\frac{2\pi km}{N}}} \\
&= \sum_{m=0}^{N-1} \overline{x[m]\, e^{-j\frac{2\pi km}{N}}} \\
&= \overline{\sum_{m=0}^{N-1} x[m]\, e^{-j\frac{2\pi km}{N}}} \\
&\downarrow \text{Assume: } x[n] \xrightarrow{\text{DFT}} X[k] \\
&= X^*[k]
\end{aligned}$$

Note that for notational clarity we used the equivalent notation: $x^* = \overline{x}$, both denoting the complex conjugate of x.

Solution 3.5.2-3: Let's start with the right hand of the equation, but write $x^*[-n]$ as $y[n]$ (periodic with period N as well):

$$\begin{aligned}
x[n] \star y^*[n] &= \sum_{m=0}^{N-1} x[m] y^*[n-m] \\
&\downarrow y[n] = x[-n] \\
x[n] \star x^*[-n] &= \sum_{m=0}^{N-1} x[m] x^*[-(n-m)] \\
&= \sum_{m=0}^{N-1} x[m] x^*[m-n] \\
&\downarrow \text{substitute: } l = m - n \\
&= \sum_{-n}^{N-n-1} x[l+n] x^*[l] \\
&\downarrow x[n] \text{ and } x^*[n] \text{ are periodic with period } N \\
&= \sum_{0}^{N-1} x[l+n] x^*[l] \\
&= x[n] \star x^*[n]
\end{aligned}$$

Solution 3.5.2-4: Let's apply the definition of the discrete Fourier transform:

$$
\begin{aligned}
\mathrm{DFT}\left(x^*[-n]\right) &= \sum_{n=0}^{N-1} x^*[-n]\, \mathrm{e}^{-\mathrm{j}\frac{2\pi kn}{N}} \\
&\downarrow \text{ substitute } m = -n \\
&= \sum_{m=-(N-1)}^{0} x^*[m]\, \mathrm{e}^{\mathrm{j}\frac{2\pi km}{N}} \\
&\downarrow x[n] \text{ is periodic with period } N \\
&= \sum_{m=0}^{N-1} x^*[m]\, \mathrm{e}^{\mathrm{j}\frac{2\pi km}{N}} \\
&= \sum_{m=0}^{N-1} x^*[m]\, \overline{\mathrm{e}^{-\mathrm{j}\frac{2\pi km}{N}}} \\
&= \sum_{m=0}^{N-1} \overline{x[m]\, \mathrm{e}^{-\mathrm{j}\frac{2\pi km}{N}}} \\
&= \overline{\sum_{m=0}^{N-1} x[m]\, \mathrm{e}^{-\mathrm{j}\frac{2\pi km}{N}}} \\
&\downarrow \text{ Assume: } x[n] \xrightarrow{\mathrm{DFT}} X[k] \\
&= X^*[k]
\end{aligned}
$$

Note that for notational clarity we used the equivalent notation: $x^* = \overline{x}$, both denoting the complex conjugate of x.

Chapter 4

The Discrete Fourier Transform in practice

Solution 4.1.2-1: The following table organizes the calculations in the same way as the calculations in the course:

		k					
		0	1	2	3	4	5
n	x[n]	W_6^{kn}					
0	−1	1.000	1.000	1.000	1.000	1.000	1.000
1	3.2	1.000	0.500 − 0.866j	−0.500 − 0.866j	−1.000 − 0.000j	−0.500 + 0.866j	0.500 + 0.866j
2	2	1.000	−0.500 − 0.866j	−0.500 + 0.866j	1.000 + 0.000j	−0.500 − 0.866j	−0.500 + 0.866j
3	5	1.000	−1.000 − 0.000j	1.000 + 0.000j	−1.000 − 0.000j	1.000 + 0.000j	−1.000 − 0.000j
4	−1.2	1.000	−0.500 + 0.866j	−0.500 − 0.866j	1.000 + 0.000j	−0.500 + 0.866j	−0.500 − 0.866j
5	1.3	1.000	0.500 + 0.866j	−0.500 + 0.866j	−1.000 − 0.000j	−0.500 − 0.866j	0.500 − 0.866j
		$X[k]$					
		9.300	−4.150 − 4.417j	1.350 + 1.126j	−9.700 − 0.000j	1.350 − 1.126j	−4.150 + 4.417j

Solution 4.1.2-2: The following code is very straightforward:

```
x = [ -1  3.2  2  5  -1.2  1.3 ];
X = fft( X )
```

Solution 4.1.2-3: We continue with Octave/Matlab in the state after the previous exercise

```
subplot(2,1,1);
plot( 0:5, abs(X), 's' );
subplot(2,1,2);
plot( 0:5, angle(X), 's' );
```

Solution 4.1.2-4: We continue with Octave/Matlab in the state after the previous exercise

```
xx = ifft(X)
```

The result looks like this:

```
x =
 Columns 1 through 4:
  -1.0000 + 0.0000i   3.2000 - 0.0000i   2.0000 - 0.0000i   5.0000 + 0.0000i
 Columns 5 and 6:
  -1.2000 + 0.0000i   1.3000 - 0.0000i
```

This is again the original sequence. Perfect.

Solution 4.1.2-5:

```
subplot(2,1,1);
plot( 0:5, abs(X), 's' );
title( "magnitude␣and␣phase␣of␣X␣=␣fft(x)" );
xlabel( "k" );
ylabel( "|X|" );
subplot(2,1,2);
plot( 0:5, angle(X), 's' );
xlabel( "k" );
ylabel( "angle(X)" );
```

Solution 4.1.2-6: Replace line 10 in the previous exercise by:

```
plot( 0:5, 20*log10( abs(X) ), 's' );
```

Solution 4.1.2-7: Issue the following command after plotting:

```
axis( 0, 5);
```

Solution 4.4.1-1: As the signal is causal, we are ready to pad it with zeroes,

```
x = [ -1, 3.2, 2, 5, -1.2, 1.3 ];
xx = [ x, 0, 0 ];
```

and calculate its DFT:

```
fft(xx)
==> ans =
 Columns 1 through 4:
   9.3000 + 0.0000i  -1.9920 - 6.8790i  -4.2000 + 0.5000i   2.3920 - 2.8790i
 Columns 5 through 8:
  -9.7000 + 0.0000i   2.3920 + 2.8790i  -4.2000 - 0.5000i  -1.9920 + 6.8790i
```

Solution 4.4.1-2: Let's set up our initial signal vector:

```
x = [2, 3, 5, 4, -1, 1, -5, -3];
```

For $M = 8$, we'll do the zero padding the hard way:

```
xx = [ x, 0, 0, 0, 0, 0, 0, 0 ];
XX = fft( x );
subplot(2,1,1);
plot( 0:length(XX)-1, abs(XX), 's' );
title( "FFT␣of␣the␣zero␣padded␣signal␣(M␣=␣8)" );
axis( [ 0, length(XX)-1 ] );
xlabel( "k" );
ylabel( "|XX|" );
subplot(2,1,2);
plot( 0:length(XX)-1, angle(XX), 's' );
axis( [ 0, length(XX)-1 ] );
xlabel( "k" );
ylabel( "angle(XX)" );
```

You can repeat the sequency above for higher values of M. However, the zero padding can be done with considerably less effort:

```
xx = [ x, zeros(1,M) ];
```

Assigning an appropriate value to M, and repeating the previous steps, generates the desired graphs. One can keep the previous plots on the screen, by opening new figures prior to generating the graphs. Just type with increasing positive integer values of i in between every graph you want to generate.

```
figure(i);
```

As you will observe, the plots with higher values of M show an increased detail of the original spectrum.

Solution 4.4.1-3: Taking a look at the transition diagram, we conclude that we can calculate a DtFT by calculating a DFT, and multiplying the DFT values with a comb function with overall weight ω_f. From an energy point of view, the dirac impulses can be replaced by rectangles of height equaling the DFT values and width the frequency pitch $omega_f$.

Let's start by defining the signal x, assuming that timepoint $t = 4\,\mu s$ is a repetition of $t = -4\,\mu s$,

$$x = [0 \quad -0.2 \quad 0 \quad 0.5 \quad \underbrace{1}_{n=0} \quad 0.5 \quad 0 \quad -0.2]$$

Let's enter this signal in Octave/Matlab:

```
x = [ 0, -0.2, 0, 0.5, 1, 0.5, 0, -0.2 ];
```

Now, let's zero pad the signal to make it of length 2^{10}. Making signals of length equaling a power of two, helps the overall computational performance.

```
xx = [ x, zeros(1, 1024 - length(x) ];
```

and let's only consider positive times (because the DFT/FFT assumes positive timepoints), keeping the periodicity of the time-signal intact:

```
xx = circshift( xx, -4 );
```

Next, we can calculate the DFT/FFT and again make it symmetrical around $\omega = 0$.

```
XX = fft(xx);
X = circshift( XX, 512 );
```

Finally, let's plot the DtFT. The abcissa of the graph ranges from $-\omega_s/2$ tot $\omega_s/2$, with $\omega_s = 2\pi/T$ with $T = 1\,\mu s$

```
subplot(2,1,1);
plot( linspace( -1000*pi, 1000*pi, 1025)(1:1024), abs( X ), '-' );
title( "DtFT(x(t)" );
axis( [-1000*pi, 1000*pi ] );
xlabel( "w␣(Hz)" );
ylabel( "|X|" );
subplot(2,1,2);
plot( linspace( -1000*pi, 1000*pi, 1025)(1:1024), angle( X ), '-' );
axis( [-1000*pi, 1000*pi ] );
xlabel( "w␣(Hz)" );
ylabel( "angle(X)" );
```

If you take a look at the phase graph, can you explain the jumps from 0 to a value slightly higher than 3 and back?

Solution 4.4.2-1: Let's enter the signal into Octave/Matlab:

```
X = [ 1, 2+j, 2-j];
```

As this spectrum has an even magnitude and an odd phase, we know it corresponds to a real signal. We can easily check this in Octave/Matlab:

```
x = ifft( X )
=> x =
   1.66667  -0.91068   0.24402
```

We must make sure the zero padding does not destroy this property.
We can do this by shifting the signal to reveal the symmetry, pad it with zeros and then undo the shifting. This corresponds to 'adding zeros in the middle'.
We start with the shifting:

```
circshift( X, 1 )
==> answer =
   2 - 1i   1 + 0i   2 + 1i
```

Then we add zeros:

```
XX = [ circshift( X, 1 ), zeros( 1, 5 ) ];
==> XX =
   2 - 1i   1 + 0i   2 + 1i   0 + 0i   0 + 0i   0 + 0i   0 + 0i   0 + 0i
```

And then, we undo the shifting:

```
circshift( XX, -1 )
==> answer =
   1 + 0i   2 + 1i   0 + 0i   0 + 0i   0 + 0i   0 + 0i   0 + 0i   2 - 1i
```

Now, we're ready to calculate the inverse DFT:

```
xx = ifft( circshift( XX, -1 ) )
==> xx =
 Columns 1 through 4:
   0.62500   0.30178  -0.12500  -0.40533
 Columns 5 through 8:
  -0.37500  -0.05178   0.37500   0.65533
```

Finally, we need to undo the scaling effect of the frequency-domain zero padding:

```
xxx = 8/3 * xx;
```

Solution 4.4.2-2: Let's set up our initial signal vector, and calculate it's DFT:

```
x = [2, 3, 5, 4, -1, 1, -5, -3];
X = fft(x)
==> X =
 Columns 1 through 3:
    6.00000 +  0.00000i   -0.53553 - 16.36396i    1.00000 -  3.00000i
 Columns 4 through 6:
    6.53553 +  3.63604i   -4.00000 +  0.00000i    6.53553 -  3.63604i
 Columns 7 and 8:
    1.00000 +  3.00000i   -0.53553 + 16.36396i
```

It's easy to verify that the spectrum has an even magnitude and an odd phase. When padding this spectrum with zeros, we must maintain the symmetry. For spectra with an even length, this is not so obvious.
Let's take a look at the problem that arises, when applying carelessly the method of the previous example:

```
N = length( X );
M = 8
XX = circshift( [ circshift( X, N/2 ), zeros( 1, M ) ], -N/2 );
==> XX =
 Columns 1 through 3:
    6.00000 +  0.00000i   -0.53553 - 16.36396i    1.00000 -  3.00000i
 Columns 4 through 6:
    6.53553 +  3.63604i    0.00000 +  0.00000i    0.00000 +  0.00000i
 Columns 7 through 9:
    0.00000 +  0.00000i    0.00000 +  0.00000i    0.00000 +  0.00000i
 Columns 10 through 12:
    0.00000 +  0.00000i    0.00000 +  0.00000i    0.00000 +  0.00000i
 Columns 13 through 15:
   -4.00000 +  0.00000i    6.53553 -  3.63604i    1.00000 +  3.00000i
 Column 16:
   -0.53553 + 16.36396i
```

We can easily check that the symmetry is broken for $XX(5)$ when comparing to $XX(13)$.[1]
We restore the symmetry by assigning the average of the two values to both cells, i.e.

```
XX(13) = XX(5) = XX(13) / 2;
```

Now, we can calculate the original time signal by applying the inverse DFT, and undoing the scaling caused by the frequency-domain zeropadding.

```
xx = 16/8 * ifft( XX )
==> xx =
 Columns 1 through 4:
   2.00000    2.68441    3.00000    3.67025
 Columns 5 through 8:
   5.00000    5.73108    4.00000    0.62062
 Columns 9 through 12:
  -1.00000    0.22980    1.00000   -1.46315
 Columns 13 through 16:
  -5.00000   -5.64530   -3.00000    0.17228
```

As you can check, the original signal values are present in this zero-padded version.

You can repeat the above for increasing values of M, and you will see an improved time-domain detail for increasing values of M.
Plotting makes the correspondence visually apparent. One can keep the previous plots on the screen, by opening new figures prior to generating the graphs. Just type with increasing positive integer values of i in between every graph you want to generate.

```
figure(i);
```

Solution 4.7.4.10-1: Consider the definition of the Blackman window:

$$\text{Blackman}_N[n] = \begin{cases} 0.42 - 0.5\cos\left(\frac{2\pi n}{N}\right) + 0.08\cos\left(\frac{4\pi n}{N}\right) & \text{if } 0 \leq n \leq N-1 \\ 0 & \text{otherwise} \end{cases}$$

Taking your calculator, setting $N = 10$ and a few calculations later, you obtain the following series:[2]

$$\text{Blackman}_{10}[n] = [0, 0.0402, 0.2008, 0.5098, 0.8492, 1, 0.8492, 0.5098, 0.2008, 0.0402]$$

Multiplying this window element by element with the original time-domain sequency $x[n]$ yields:

$$x_{\text{Blackman}}[n] = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$$

The same calculations can be made using Octave/Matlab. However, you need to take care of the somewhat tricky definition of the Blackman window in Octave/Matlab.
Let's start by entering the signal into Octave/Matlab:

```
x = [ 0, 24.86767, 9.96164, 5.88481, 4.71015, 5.00000, ...
      7.06522, 13.73122, 39.84656, 223.80899];
```

The Blackman window can be generated using:

```
wb = blackman(11)(1:10)'
==> wb =
 Columns 1 through 6:
  -1.3878e-17   4.0213e-02   2.0077e-01   5.0979e-01   8.4923e-01   1.0000e+00
 Columns 7 through 10:
   8.4923e-01   5.0979e-01   2.0077e-01   4.0213e-02
```

[1] An invariant to the case of spectra with an even lenght is that one of these values is zero and the other is real.
[2] Make sure your calculator considers angles to be in radians, not degrees!

The resulting windowed time-domain sequence is easily obtained as:

```
xwb = x .* wb
==> xwb =
 Columns 1 through 9:
  -0.00   1.00   2.00   3.00   4.00   5.00   6.00   7.00   8.00
 Column 10:
   9.00
```

Solution 4.7.4.10-2: Start by numbering the first and last element of the sequence in the time domain:

$$x[n] = [\underbrace{0}_{n=0}, 1, 2, 3, 4, 5, 4, 3, 2, 1, \underbrace{0}_{n=10}]$$

This makes it clear we need a Hann window with the extremities (equaling zero) at $n = 0$ and $n = 10$. Therefore, we pick $N = 10$.
As in the previous exercise, we can use our calculator, but here, we'll employ Octave/Matlab. The calculations are most easy. Take into account that in Octave/Matlab, we need to enter $N + 1$ as parameter to the window functions:

```
x = [ 0, 1, 2, 3, 4, 5, 4, 3, 2, 1, 0 ];
wh = hanning( 11 )';
xwh = x .* wh
==> xwh =
 Columns 1 through 6:
   0.00000   0.09549   0.69098   1.96353   3.61803   5.00000
 Columns 7 and 11:
   3.61803   1.96353   0.69098   0.09549   0.00000
```

We clearly can observe that the symmetry has been maintained.

Solution 4.7.4.10-3: Start by numbering the first and last element of the sequence in the time domain:

$$x[n] = [\underbrace{1}_{n=0}, 2, 3, 4, 4, 3, 2, \underbrace{1}_{n=7}]$$

This makes it clear we need a Nuttall window with the extremities (equaling zero) at $n = 0$ and $n = 7$. Therefore, we pick $N = 7$.
As in the first exercise, we can use our calculator, but here, we'll again employ Octave/Matlab. The calculations are most easy. Take into account that in Octave/Matlab, we need to enter $N + 1$ as parameter to the window functions:

```
x = [ 1, 2, 3, 4, 4, 3, 2, 1 ];
wn = nuttallwin( 8 )';
wxn = x .* wn
==> wxn =
 Columns 1 through 5:
   3.6280e-04   7.5552e-02   1.0282e+00   3.5674e+00   3.5674e+00
 Columns 6 and 8:
   1.0282e+00   7.5552e-02   3.6280e-04
```

We clearly can observe that the symmetry has been maintained.

Solution 4.7.4.10-4: Show your solution to the professor. He'll be happy to check your result.

Chapter 5

Sampling, Quantization and Reconstruction

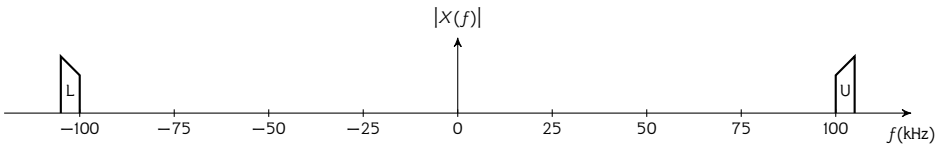
Solution 5.2.1-1: According to Shannon's sampling theory:

$$\omega_c = \frac{\omega_s}{2} = 10\,\text{Mrad/s}$$

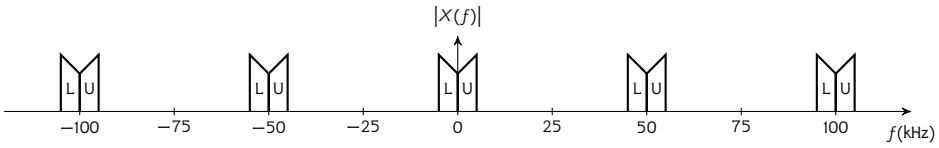
Solution 5.2.1-2: According to Shannon's sampling theory:

$$f_s = 2f_c = 40\,\text{kHz}$$

Solution 5.2.2-1: Rather than using the complicated equations above, the easiest way to solve this type of problem, is to make a small sketch of the spectrum at hand. Here's a sketch of the signal:

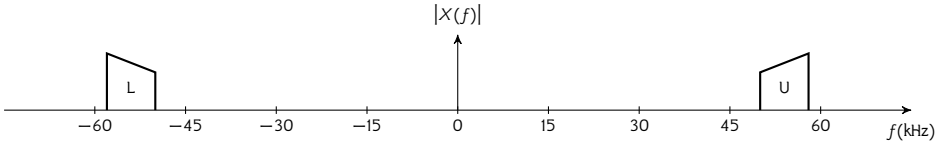


Sampling this signal at a rate of $f_s = 50\,\text{kHz}$, means replicating this spectrum shift over a distance equaling a multiple of f_S. This has been depicted in the sketch below:

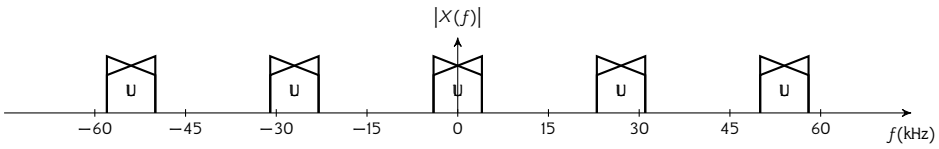


As one can clearly see, no destructive aliasing occurs and there is no spectral inversion.

Solution 5.2.2-2: Rather than using the complicated equations above, the easiest way to solve this type of problem, is to make a small sketch of the spectrum at hand. Here's a sketch of the signal:

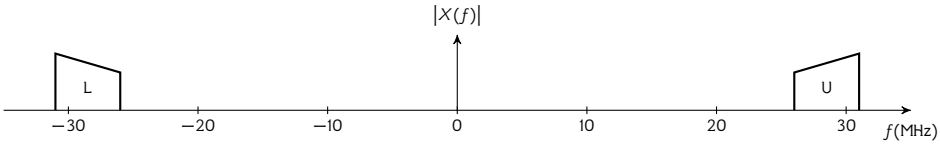


Sampling this signal at a rate of $f_s = 27\,\text{kHz}$, means replicating this spectrum shift over a distance equaling a multiple of f_S. This has been depicted in the sketch below:

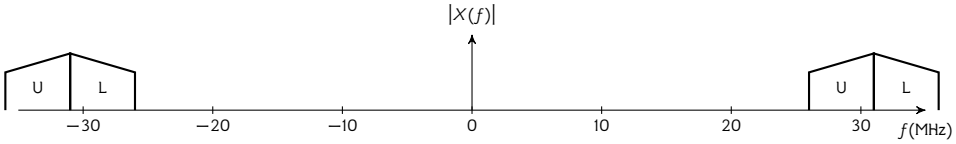


As one can clearly see, destructive aliasing occurs. This is not a good sampling frequency.

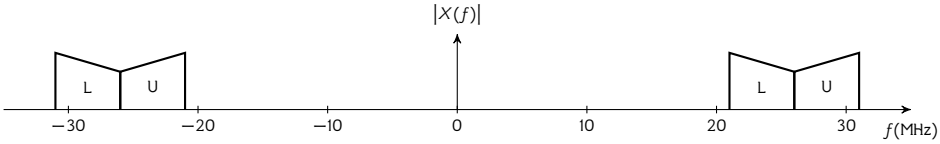
Solution 5.2.2-3: One possible solution strategy is to employ the equations outlined in the section on bandpass sampling. However, following the reasoning outlined there is not that difficult and avoids having to memorize the equations (or looking them up on the course, or in a formula collection).
Let's start by sketching the original signal:



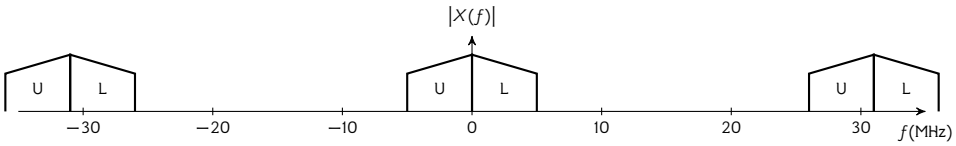
Shannon's (low-pass) frequency sampling theorem tells us to sample this signal with $\omega_s = 62\,\text{MHz}$. This corresponds to the graph below:



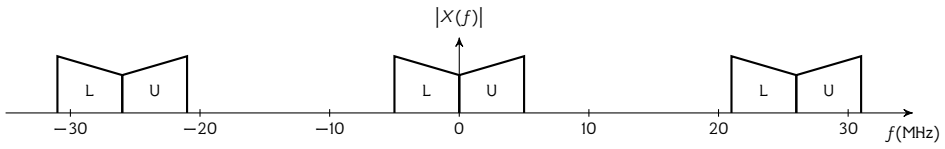
Let's try to lower the sampling frequency while looking for candidate optimal solutions. We can lower the sampling frequency introducing destructive overlap, but we can continue lowering the sampling frequency until the overlap ceases to exist. This corresponds to the situation below (with spectral inversion):



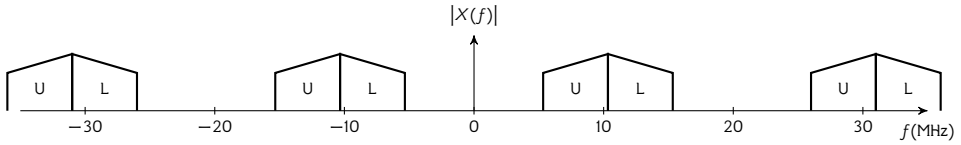
The sampling frequency, in this case, is easily determined to be 52 MHz.
However, in this case, we can further lower the sampling frequency without introducing overlap until a new abutment arises, still with spectral inversion.
This corresponds to the graph below:



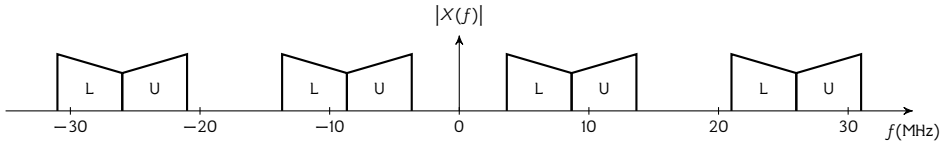
The sampling frequency in this case is easily determined to be $f_s = 31\,\text{MHz}$. This is a candidate solution (with spectral inversion).
Reducing the sampling frequency even further, creates destructive overlap until the we reach the following situation without spectral inversion:



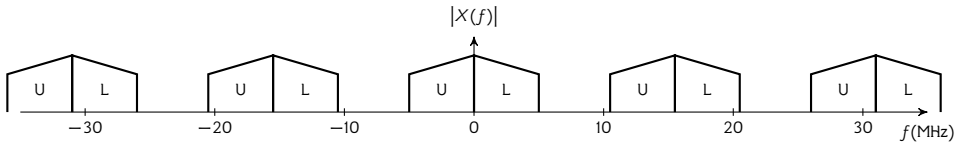
There's no destructive overlap. The sampling frequency can be easily determined to be $f_s = 26\,\text{MHz}$.
However, in this case we can further reduce the sampling frequency until a new abutment arises, still without spectral inversion:



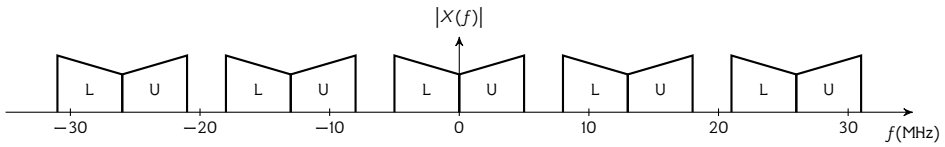
The sampling frequency corresponding to this situation is easily determined to be $f_s = 20.667\,\text{MHz}$. Again, this is a candidate solution (without spectral inversion).
Reducing the sampling frequency even further, creates destructive overlap until the we reach the following situation with spectral inversion:



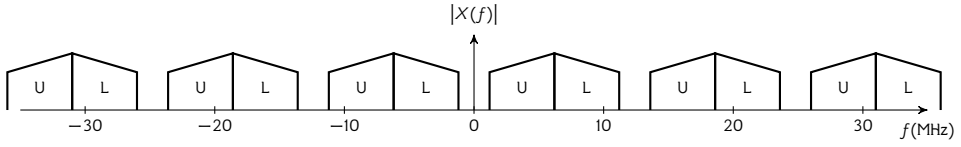
There's no destructive overlap. The sampling frequency can be easily determined to be $f_s = 17.333\,\text{MHz}$.
However, in this case we can further reduce the sampling frequency until a new abutment arises, still with spectral inversion:



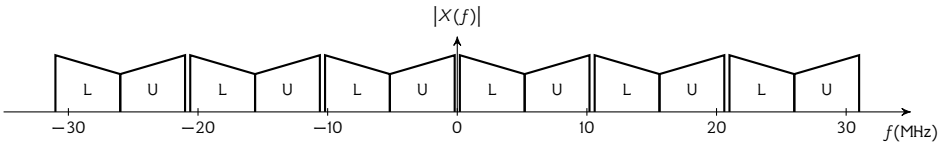
The sampling frequency corresponding to this situation is easily determined to be $f_s = 15.5\,\text{MHz}$. Again, this is a candidate solution (with spectral inversion).
Reducing the sampling frequency even further, creates destructive overlap until the we reach the following situation without spectral inversion:



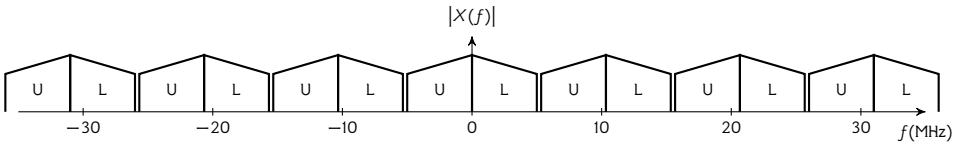
There's no destructive overlap. The sampling frequency can be easily determined to be $f_s = 13\,\text{MHz}$.
However, in this case we can further reduce the sampling frequency until a new abutment arises, still without spectral inversion:



The sampling frequency corresponding to this situation is easily determined to be $f_s = 12.4\,\text{MHz}$. Again, this is a candidate solution (without spectral inversion).
Reducing the sampling frequency even further, creates destructive overlap until the we reach the following situation with spectral inversion:



There's no destructive overlap. The sampling frequency can be easily determined to be $f_s = 10.4\,\text{MHz}$.
However, in this case we can further reduce the sampling frequency until a new abutment arises, still with spectral inversion:



The sampling frequency corresponding to this situation is easily determined to be $f_s = 10.333\,\text{MHz}$. Again, this is a candidate solution (with spectral inversion).
When reducing the sampling frequency even further, overlap can no longer be avoided. Conclusion: the latest candidate solution is the best one. The lowest bandpass sampling frequency is $f_s = 10.333\,\text{MHz}$. In this case spectral inversion does occur.
For practical reasons (to avoid spectral overlap at $f = 0$, very often one chooses a sampling frequency $f_s = \frac{10.333+10.4}{2}\,\text{MHz} = 10.367\,\text{MHz}$.

Solution 5.2.2-4: From the solution of the previous exercise, it is easy to see that the lowest frequency that avoid spectral inversion is $f_s = 12.4\,\text{MHz}$.
For practical reasons (to avoid spectral overlap at $f = 6.2\,\text{MHz}$, very often one chooses a sampling frequency $f_s = \frac{12.4+13}{2}\text{MHz} = 12.7\,\text{MHz}$.

Solution 5.5.1.1-1: We'll start by loading the audio fragment

```
[ x, fs, format ] = auload('filename.wav');
```

The samples are present in matrix `x` that has two columns, one for the left stereo channel, one for the right stereo channel. The values are in the range $[-1, 1]$.
You can check the result by playing the audio fragment:

```
sound( x, fs );
```

Requantizing can be done very easily (though computationally not so optimal, but we don't care about that):

```
N = 8;
y = fix( 2^(N-1) * x ) / 2^(N-1);
```

Listening to the requantized signal will clearly illustrate what quantization noise means.

```
sound( y, fs );
```

What you here is pure white noise on the background, solely due to the 8-bit quantization.

Solution 5.5.1.1-2: We'll start by loading the audio fragment

```
[ x, fs, format ] = auload('filename.wav');
```

The samples are present in matrix `x` that has two columns, one for the left stereo channel, one for the right stereo channel. The values are in the range $[-1, 1]$.
You can check the result by playing the audio fragment:

```
sound( x, fs );
```

Requantizing the fragment nonlinearly using mu-law can be done very easily using the `lin2mu` and `mu2lin` functions.

```
z = mu2lin( lin2mu( x ) ) / 128;
```

Listening to the requantized signal shows some improvement concerning signal to noise ratio, but the result is not very spectacular.

```
sound( z, fs )
```

This is due to the fact that the fragment you selected comes from a CD, that probably is mastered in such a way to optimally use the dynamic range that is available. If one takes a signal that is more quiet, the positive effect of the μ-law quantization becomes more clear.

Solution 5.5.1.1-3: We'll start by loading the audio fragment and scaling it down by a factor of 10.

```
[ x, fs, format ] = auload('filename.wav');
x /= 10;
```

Then, we will generate the linear and nonlinear quantization in y and z, respectively:

```
N = 8;
y = fix( 2^(N-1) * x ) / 2^(N-1);
z = mu2lin( lin2mu( x ) ) / 128;
```

Now comparing the three reveals a much clearer advantage of the μ-law sampling.

```
sound( x, fs );
sound( y, fs );
sound( z, fs );
```

This demo will without any doubt convince you of the advantage of nonlinear quantization for audio signals that are quantized with a low bit rate. You are using this nonlinear sampling every day when using your cell phone.

Solution 5.6.1-1: For a mid-rise converter, the quantization range $[-1, +1]$ is divided in 2^N intervals. Therefore, it's easy to line the ideal levels up next to the real levels

Level	Real Value	Ideal Value
t_{-3}	-0.80	-0.75
t_{-2}	-0.45	-0.50
t_{-1}	-0.25	-0.25
t_0	-0.01	0.00
t_{+1}	$+0.23$	$+0.25$
t_{+2}	$+0.49$	$+0.50$
t_{+3}	$+0.80$	$+0.75$

Offset The offset is easily calculated according to its definition

$$\begin{aligned} E_{offset,\text{ADC}} &= \frac{\sum_k \left(t_k - k\Delta\right)}{2^N - 1} \\ &= \frac{\sum_{k=-3} k = +3 \left(t_k - k0.25\right)}{2^3 - 1} \\ &= \frac{0.01}{7} = 0.0014286 \end{aligned}$$

In Octave this is calculated most easily:

```
tk = [ -0.80 -0.45 -0.25 -0.01 0.23 0.49 0.80 ];
ti = [ -0.75 -0.50 -0.25  0.00 0.25 0.50 0.75 ];
offset = mean( tk - ti )
==> offset = 0.0014286
```

Given the measurement accuracy in the original table, this offset is not significant, and can be considered zero.

Full-scale gain error We again revert to the definition:

$$\begin{aligned} E_{FSG,\text{ADC}} &= t_{\max} - t_{\min} - (2^N - 2)\Delta \\ &= 0.8 - (-0.8) - \left(2^3 - 2\right) 0.25 \\ &= 1.6 - 1.5 = 0.1 \end{aligned}$$

Again, in Octave this result is obtained most easily:

```
n = length( tk );
fsge = ( tk(n) - tk(1) ) - ( ti(n) - ti(1) )
==> fsge = 0.10000
```

This error is significant.

Solution 5.6.1-2: Let's start with offset and gain correction, and then calculate DNL and INL.

Offset and Gain correction The error values determined in the previous exercise allow for offset and gain correction, according to the appropriate equations. The table below shows in its columns the original values, the offset-corrected values, the offset and gain-corrected values

Level	Real Value	Offset-corrected	Offset/Gain-corrected	Ideal Value
t_{-3}	−0.80	−0.80	−0.75	−0.75
t_{-2}	−0.45	−0.45	−0.42	−0.50
t_{-1}	−0.25	−0.25	−0.23	−0.25
t_0	−0.01	−0.01	−0.01	0.00
t_{+1}	+0.23	+0.23	+0.21	+0.25
t_{+2}	+0.49	+0.49	+0.46	+0.50
t_{+3}	+0.80	+0.80	+0.75	+0.75

As you can see, the error at the end-points is reduced to zero, but the deviation at the interior levels is increased! This shows the weakness of the simple definitions of offset (which is based on all values) and gain error (which is based on the end-points only).

DNL The DNL is based on the maximum consecutive deviation between consecutive levels. The maximum consecutive deviation between any consecutive offset and gain-corrected acquisition levels occurs between t_{-2} and t_{-3}. Therefore:

$$\begin{aligned}\mathrm{DNL}_{\mathrm{ADC}} &= \frac{\max_k \left|t_k - t_{k-1} - \Delta\right|}{\Delta} \\ &= \frac{\left|t_{-2} - t_{-3} - \Delta\right|}{\Delta} \\ &= \frac{0.33 - 0.25}{0.25} = 0.32\mathrm{bit}\end{aligned}$$

Again, Octave makes this calculation most simple (and avoids the manual search):

```
tc = [ -0.75 -0.42 -0.23 -0.01 0.21 0.47 0.75 ];
ti = [ -0.75 -0.50 -0.25  0.00 0.25 0.50 0.75 ];
delta = 0.25;
n = length(tc);
DNL = max( abs( tc(2:n) - tc(1:n-1) - delta ) ) / delta
==> DNL = 0.32000
```

INL The INL is based on the maximum overall deviation between real (offset and gain corrected levels) levels and the ideal ones. The highest deviation occurs at t_{-2}, the lowest one at t_{+1}.

$$\begin{aligned}\mathrm{INL}_{\mathrm{ADC}} &= \frac{\max_k(t_k - k\Delta) - \min_k(t_k - k\Delta)}{\Delta} \\ &= \frac{(-0.42 - (-0.5)) - (0.21 - 0.25)}{0.25} = 0.48\mathrm{bit}\end{aligned}$$

Again, most simple in Octave:

```
dev = tc - ti;
INL = ( max(dev) - min(dev) ) / delta
==> INL = 0.48000
```

Solution 5.6.1-3: For a mid-rise converter, the dequantization interval $[-1, +1]$ is divided in 2^N intervals, of which one of the intervals occurs for half its length at the top and the other half at the bottom. Therefore, it's easy to line the ideal levels up next to the real levels.

Level	Real Value	Ideal Value
r_{-4}	-0.865	-0.875
r_{-3}	-0.611	-0.625
r_{-2}	-0.372	-0.375
r_{-1}	-0.131	-0.125
r_0	0.128	0.125
r_{+1}	0.384	0.375
r_{+2}	0.622	0.625
r_{+3}	0.886	0.875

Offset The offset is easily calculated according to its definition

$$\begin{aligned}E_{\mathit{offset},\mathrm{DAC}} &= \frac{\sum_k \left(r_k - (k+0.5)\Delta\right)}{2^N} \\ &= \frac{\sum_{k=-4}^{k=3} \left(r_k - (k+0.5)\Delta\right)}{2^3} \\ &= \frac{0.041}{8} = 0.0051\end{aligned}$$

In Octave this is calculated most easily:

```
rk = [ -0.865 -0.611 -0.372 -0.131 0.128 0.384 0.622 0.886 ];
ri = [ -0.875 -0.625 -0.375 -0.125 0.125 0.375 0.625 0.875 ];
offset = mean( rk - ri )
==> offset = 0.0051250
```

Given the measurement accuracy in the original table, this offset is significant.

Full-scale gain error We again revert to the definition:

$$E_{FSG,\text{DAC}} = r_{2^{N-1}-1} - r_{-2^{N-1}} - (2^N - 1)\Delta \qquad = 0.886 - (-0.865) - \left(2^3 - 1\right) 0.25$$
$$= 1.751 - 1.750 = 0.001$$

Again, in Octave this result is obtained most easily:

```
n = length( rk );
fsge = ( rk(n) - rk(1) ) - ( ri(n) - ri(1) )
==> fsge = 1.0000e-03
```

This error is on the border of being significant.

Solution 5.6.1-4: Let's start with offset and gain correction, and then calculate DNL and INL.

Offset and Gain correction The error values determined in the previous exercise allow for offset and gain correction, according to the appropriate equations. The table below shows in its columns the original values, the offset-corrected values, the offset and gain-corrected values

Level	Real Value	Offset-corrected	Offset/Gain-corrected	Ideal Value
r_{-4}	-0.865	-0.870	-0.870	-0.875
r_{-3}	-0.611	-0.616	-0.616	-0.625
r_{-2}	-0.372	-0.377	-0.377	-0.375
r_{-1}	-0.131	-0.136	-0.136	-0.125
r_0	0.128	0.123	0.123	0.125
r_{+1}	0.384	0.379	0.379	0.375
r_{+2}	0.622	0.617	0.617	0.625
r_{+3}	0.886	0.881	0.881	0.875

DNL The DNL is based on the maximum consecutive deviation between consecutive levels. The maximum consecutive deviation between any consecutive offset and gain-corrected acquisition levels occurs between r_2 and r_3. Therefore:

$$\text{DNL}_{\text{ADC}} = \frac{\max_k \left| r_k - r_{k-1} - \Delta \right|}{\Delta}$$
$$= \frac{\left| r_3 - r_2 - \Delta \right|}{\Delta}$$
$$= \frac{0.881 - 0.617}{0.25} = 0.056\text{bit}$$

Again, Octave makes this calculation most simple (and avoids the manual search):

```
rc = [ -0.870 -0.616 -0.377 -0.136 0.123 0.379 0.617 0.881 ];
ri = [ -0.875 -0.625 -0.375 -0.125 0.125 0.375 0.625 0.875 ];
delta = 0.25;
n = length(rc);
DNL = max( abs( rc(2:n) - rc(1:n-1) - delta  ) ) / delta
==> DNL = 0.056000
```

INL The INL is based on the maximum overall deviation between real (offset and gain corrected levels) levels and the ideal ones. The highest deviation occurs at r_{-3}, the lowest one at r_{-1}.

$$\begin{aligned} \mathrm{INL}_{\mathrm{DAC}} &= \frac{\max_k(r_k - (k+0.5)\Delta) - \min_k(r_k - (k+0.5)\Delta)}{\Delta} \\ &= \frac{(-0.616 - (-0.625)) - (-0.136 - (-0.125))}{0.25} = 0.08\mathrm{bit} \end{aligned}$$

Again, most simple in Octave:

```
dev = rc - ri;
INL = ( max(dev) - min(dev) ) / delta
==> INL = 0.080000
```

Solution 5.6.1-5: Submit your solution to your instructor. He will be happy to check your result.

Solution 5.6.1-6: Submit your solution to your instructor. He will be happy to check your result.

Solution 5.6.1-7: Submit your solution to your instructor. He will be happy to check your result.

Chapter 6

The Z-transform

Solution 6.6.2-1:

a) $x[n] = \sin\frac{n\pi}{2}$

b) $x[n] = \cos\frac{n\pi}{2}$

c)

$$\forall n \in \mathbb{N}: \begin{cases} x[4n] = 0 \\ x[4n+1] = 5^n \\ x[4n+2] = 0 \\ x[4n+3] = 0 \end{cases}$$

d) $x[n] = \frac{2}{3} \cdot \left(\frac{4}{3}\right)^n$

Solution 6.6.2-2:

a) $x[n] = \frac{1}{n!}$

b) $\forall n = 2, 3, 4, \ldots$:

$$\begin{cases} x[0] = 0 \\ x[1] = 0 \\ x[n] = \frac{(-1)^n}{n-1} \end{cases}$$

c) $\forall k \in \mathbb{N}$:

$$\left\{ x[2k+1] = 0 x[2k] \quad = \frac{(-1)^k}{(2k+1)!} \right.$$

d) $\forall k \in \mathbb{N}$:

$$\begin{cases} x[2k+1] = 0 \\ x[2k] = \frac{1}{(2k)!} \end{cases}$$

e) $\forall k \in \mathbb{N}$:

$$\begin{cases} x[2k+1] = \frac{-2}{2k+1} \\ x[2k] = 0 \end{cases}$$

Solution 6.9-1:

a) $y[4k] = y[4k+3] = 0$ and $y[4k+1] = y[4k+2] = 1$

b) $y[n] = n^2 + 1$

c) $y[n] = 3n + 1$

d) $y[n] = 12 \cdot 2^{-n}$

e) $y[n] = 10 \cdot \delta[n] + 2 \cdot \frac{3^n}{n!}$

f) $y[n] = 3n$

g) $y[n] = \frac{6}{(n+2)!}$

h) $y[n] = 3^{-n}$

i) $y[n] = 1 - 2^n$

j) $y[n] = \cos(n \arccos a)$