APPLICATIONS IN SIGNAL PROCESSING

Section 9 Digitizing a Signal

This document digitizes continuous or sampled signals. Your input:

- **F(t)**, the calculated or sampled continuous-time signal
- Δ, the sampling interval
- ω, the bandwidth limit of the signal

The output is a vector of integers representing the quantized level assigned to each digitized sample. The document plots the original samples, the quantized output, and the quantizing noise.

References

William McC. Siebert, *Circuits, Signals, and Systems*, The MIT Press (Cambridge, 1986).

Background

It is often desirable to convert a continuous-time signal into a set of digital samples. The signal can now be processed by digital circuitry, and will be easier to store and recall. This type of sampling is used in compact disk recording technology and other analog-to-digital conversion applications.

Nyquist Sampling Theorem and Aliasing

There are two issues in digitization. The first is the sampling rate. A signal is completely characterized by sampling it at twice its bandwidth. This is known as the Nyquist sampling theorem. The signal can be oversampled, which may be done to generate a smoother reconstruction of the signal.

If fewer samples are used, then signal components of higher frequencies will resemble additional components at lower frequencies. This is a phenomenon called aliasing, which can distort the information carried by a signal.

Quantization Levels

The second criterion for effective digitization is the size of the smallest quantized step in amplitude, known as the sampling interval. In theory, the greater the number of levels, the more accurately the amplitude of the signal can be represented. However, limitations on storage size and calculation dictate that levels be chosen at some reasonable size: one tenth of the amplitude or smaller. Sufficiently small intervals ensure that the error in quantization will be independent of the trends in the signal. For a sufficiently large number of levels, the maximum error will be restricted to approximately one interval.

Mathcad Implementation

Mathcad's histogram function **hist** is used to digitize a continuous signal. The signal is sampled at a given rate, and each sample is assigned a quantized amplitude level. To use the example, define an input function **F(t)** (the signal to be digitized) with a specified bandwidth, **w**. Define the length of time, **L**, for the sampled output, and a value for **N**, the number of levels in the quantized amplitude.

Enter signal and sampling information.

bandwidth, shown as highest frequency component (10 kHz as an example):

$$\omega := 10^4$$

input signal:

$$F(t) := .3 \cdot \cos(\omega \cdot t) + .4 \cdot \cos\left(\frac{\omega}{4} \cdot t\right) + \cos\left(\frac{\omega}{6} \cdot t\right)$$

The length of the sample is given by 20 times the smallest frequency component. This is entered as a multiple of the bandwidth:

length of sample:

$$x := 6 \qquad L := 20 \cdot \left(\frac{x}{\omega}\right)$$

number of quantizing levels:

$$N := 10$$

Calculate the Digitizing Parameters and the Vector of Samples

sampling frequency (4X oversampling):

$$f := 4 \cdot \left(\frac{\omega}{\pi}\right)$$

sampling rate:

$$T := \frac{1}{f} = 7.854 \cdot 10^{-5}$$

vector of samples:

$$n := 0 .. f \cdot L \qquad s_n := F(n \cdot T)$$

maximum and minimum values of the function (used to calculate the sample interval):

$$A := \max(s) = 1.7$$

$$B := min(s) = -1.449$$

The application now constructs a vector **a** which stores the **N** possible quantized levels for amplitude. The vector **codes** contains the numbers 0 through **N** - 1, which identify the levels.

sampling interval:

$$\Delta := \frac{A-B}{N} = 0.315$$

vector of levels:

$$k := 0..N \qquad a_k := B + k \cdot \Delta \qquad a_N := A + 1$$

vector of codes:

$$i := 0..N-1 \qquad code_i := i$$

A straightforward application of the **hist** function returns a vector **q** giving the level (number of sampling intervals) for each sample value.

coded sample:

$$q_n := \mathrm{hist}\left(a, s^{\langle n \rangle}\right) \cdot code$$

You can use **q** to reconstruct the digitized approximation to the original signal. The first plot shows this approximation **o** together with the original signal **F**. The second plot shows the quantizing noise, that is, the difference between the original sample values and the quantized amplitude values in **o**.

step-function output:

$$o := B + \frac{\Delta}{2} + q \cdot \Delta$$

graphing parameters:

$$step := \frac{L}{200} \qquad t := 0, step..L$$

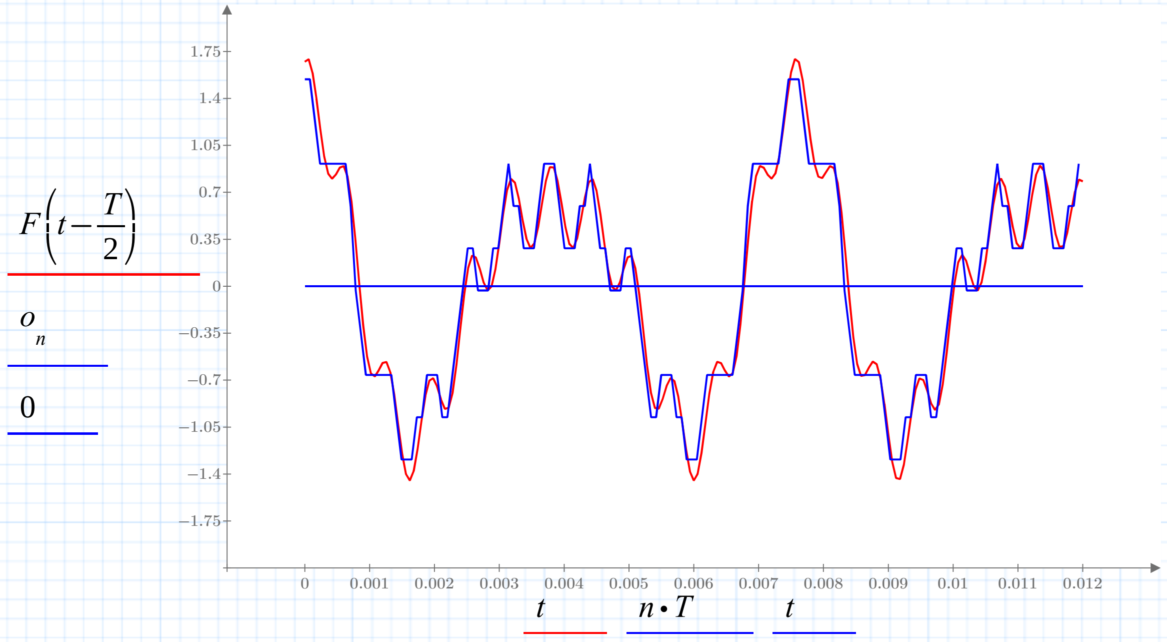

Fig. 9.1 Plot of output and original signal

The quantization noise is given by:

$$e := s - o$$

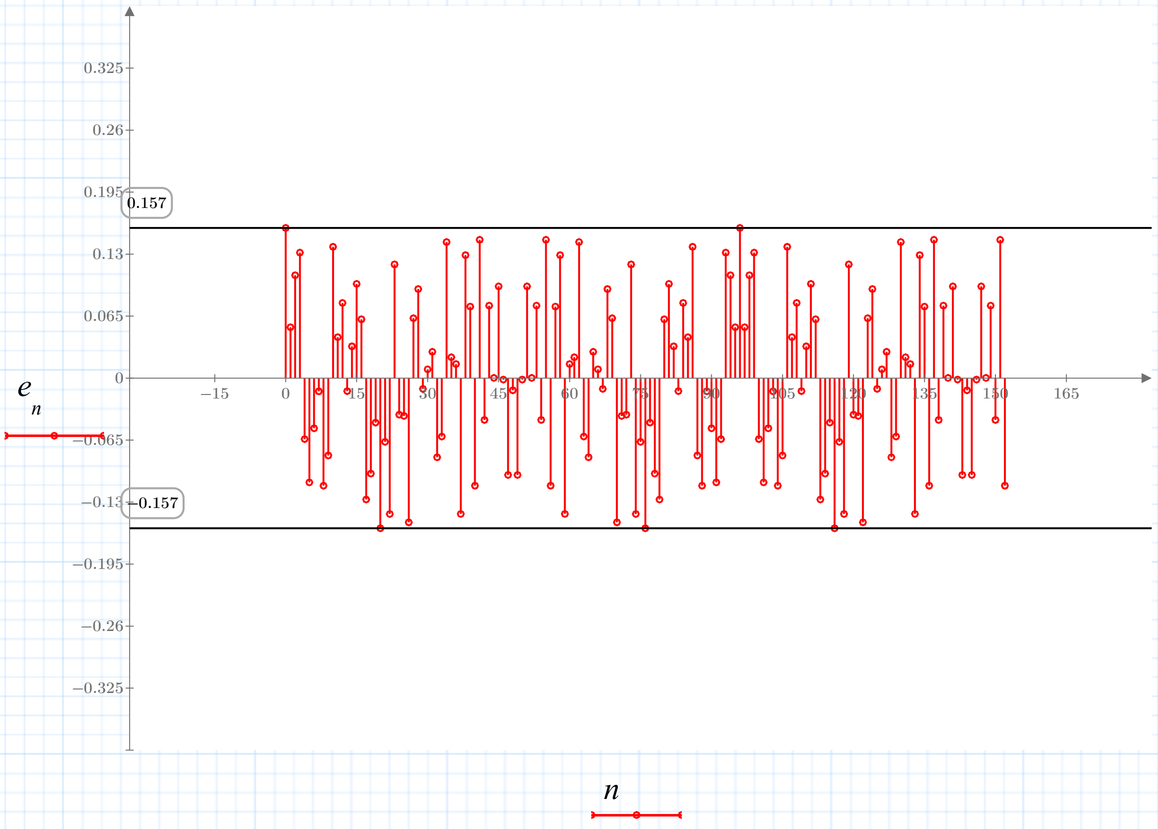

Fig. 9.2 Plot of quantization noise

The space between the dotted lines is one sampling interval, **D**.

To quantize any set of sample data, read the data into the vector **s** by using one of the functions in the **File Access** category, such as READCSV, READEXCEL, READFILE, READPRN, or READTEXT. Alternately, you could use the READEXCEL button or the Excel Component button. These are located on the Mathcad Ribbon: Input/Output tab.

Once your data is stored in the vector **s**, let **n** run from 0 to **length(s)** – 1. You can now delete the original definition of **s**, and the associated definitions for sampling time and sample length.