Práctica 1: Sampling and Quantization

Renato Bedriñana Cárdenas   Hugo Blanco Demelo

24 de octubre de 2025

1 Task 1

Question: Give your interpretation of the resulting graphs. Do the quantization levels correspond with the values you had expected?

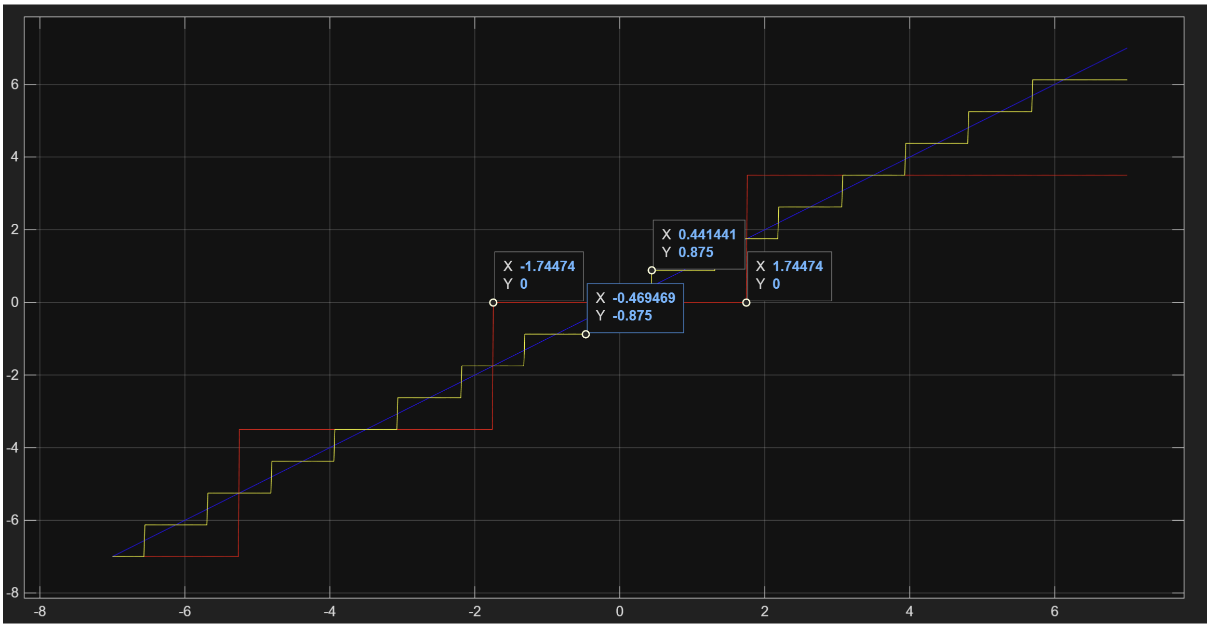
We represent the original continuous signal x (blue) and the 2 quantized version using 2 (red) and 4 (yellow) bits. As expected, the 2 bit quantization produces fewer discrete levels than the 4 bits quantization. Increase the number of bits decreases Δ, resulting in smaller steps and a quantized signal that follows the input more closely.

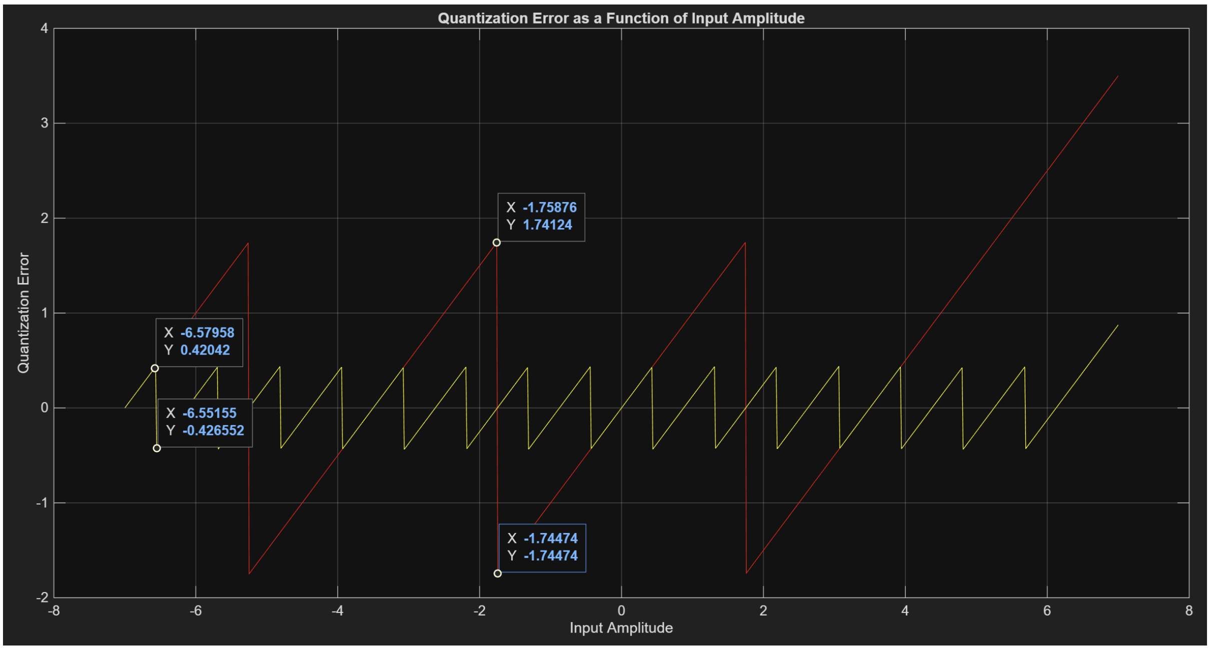
Question: For both cases, represent the quantization error as a function of input amplitude in the range $[-7, +7]$ and comment on your results. Is this error always within the $[-\Delta/2, +\Delta/2]$ interval?

The magnitude of the error decreases as the number of bits increases, since a smaller quantization step Δ reduces the maximum deviation between the input and its quantized version. The $[-\Delta/2, +\Delta/2]$ in each case is as follows:

- For N = 2 the Δ value we get is $\Delta = 3{,}5$, so the interval should be $[-1{,}75, 1{,}75]$.
- For N = 4 the Δ value we get is $\Delta = 0{,}875$, so the interval should be $[-0{,}4375, 0{,}4375]$.

In both cases, the error remains bounded within the theoretical interval $[-\Delta/2, +\Delta/2]$.

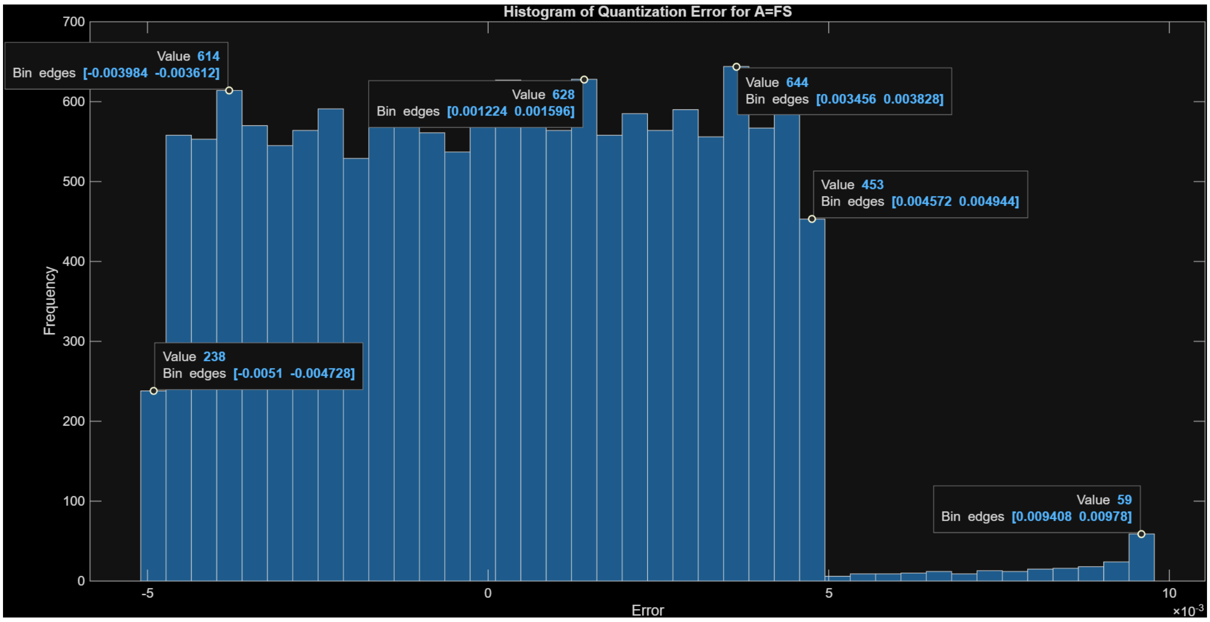




2 Task 2

Question: Assume a full-scale sinusoidal input and plot the histogram of the quantization error. Do you observe what you expected, or not?

Due we have an amplitude equal to FS we can expect clipping. We have $\Delta = \frac{2*FS}{2^N} = 0{,}0098$, the $[-\frac{\Delta}{2}, +\frac{\Delta}{2}]$ interval should be uniformly distributed (while the input does not get clipped) between $[-0{,}0049, +0{,}0049]$. In the histogram we can see that in that interval the error is uniformly distributed, but there is an error tail in the positive extreme. It means that there is **clipping** in the positive.



Question: Explain the operation of the Matlab command var. Estimate the variance of the quantization error using var, and compare it to its theoretical value. Estimate the value (in dB) of the Signal to-Quantization Noise Ratio (SQNR) and compare it to its theoretical value.

The MATLAB command `var` computes the variance of a set of values. For a vector x, it calculates: $\mathrm{var}(x) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$, where $\bar{x}$ is the mean of the values in x and n is the total number of samples. We used `var(x,1)` to compute the population variance (divide by n).

The empiral value of the variance of the quantization error we got is $8{,}72123e-06$, and the theoretical value is $\frac{\Delta^2}{12} = 7{,}94729e-06$.

The estimated value of the SQNR in dB we got is 61,5633 dB, and the theoretical value is 61,9597 dB.

Question: Repeat the previous steps for sinusoids with different amplitudes, and with decreasing resolutions of 12, 10, 8, 6 and 4 bits, in order to fill Table 1, rounding the SQNR values (in dB) to two decimal places. Comment on your results.

	$A = 0{,}5 \cdot \texttt{FS}$		$A = 0{,}75 \cdot \texttt{FS}$		$A = \texttt{FS}$		$A = 1{,}03 \cdot \texttt{FS}$	
	SQNR (dB)		SQNR (dB)		SQNR (dB)		SQNR (dB)	
N	theory	measured	theory	measured	theory	measured	theory	measured
12	67.98	68.03	71.5	71.54	74.00	73.7	74.26	38.47
10	55.94	56.01	59.46	59.53	61.96	61.56	62.22	38.19
8	43.90	44.04	47.42	47.53	49.92	49.15	50.18	36.97
6	31.86	32.13	35.38	35.6	37.88	36.52	38.14	32.27
4	19.82	20.37	23.34	23.78	25.8397	23.63	26.1	22.53

Cuadro 1: Pertaining to Task 2.

For amplitudes below FS (0.5*FS and 0.75*FS) the empirical SQNR values closely match the theoretical predictions. For an amplitude equal to FS, the empirical values still align well with theory, indicating minimal clipping effects. However, as the amplitude exceeds FS (1.03*FS), discrepancies arise due to clipping effects, SQNR collapses and even adding more bits does not solve the problem.

As the number of bits decreases the variance of the error grows roughly as expected and SQNR drops approximately 6 dB/bit. For moderate amplitudes the theory remains a good approximation down to mid-low N (but deviations increases as N gets smaller).

3 Task 3

Question: Suppose that you have an N-bit A/D converter with tunable FS, and you know that your input samples follow a symmetric triangular pdf in some interval $[-x_0, x_0]$. Intuitively, how would you set the FS value of your converter? What would the resulting rms value σ_x in dBFS be?

If you set $FS < x_0$ any imput $|x|$ greater than FS will be clipped. If $FS > x_0$,we would be wasting the converter's since the signal would never reach the limits. Therefore, the value of FS should be x_0.

To reach the variance of a symmetric triangular distribution we need to make some calculations:

$$Var(x) = E[x^2] - (E[x])^2 = E[x^2] + 0 = \int_{-x_0}^{x_0} x^2 f(x)dx = x_0^2/6$$

$\sigma_x = \sqrt{var(x)} = \frac{x_0}{\sqrt{6}}$ and in dBFS (with $x_0 = \text{FS}$) would be $20\log_{10}(1/\sqrt{6}) = -7,78$ dBFS.

Question: Explain how to generate in Matlab samples of a random variable following a symmetric triangular pdf with zero mean and rms value σ_x. Check the histogram and use the commands mean and var to validate your approach

We have two options to do it:

- Option 1: The easiest way to generate a random variable with triangular pdf is using the function *makedist* from Matlab. The function spects the parameters A, B and C that define the triangular distribution. So we set the function parameters to get a symmetric triangular distribution centered at 0: `makedist('Triangular','A',-x0,'B',x0,'C',0)`

  The result of the mean and var commands are as follows:

  - Empirical mean: -4.58885e-05, wich is near to 0, very close to our target mean.
  - Empirical rms value [dBFS]: -7.78072, wich is very close to our target variance.

  To do it, we can use the following code:

```
x0 = 2;
A = -x0; B = 0; C = +x0; % simetria = media 0

pd = makedist('Triangular','A',A,'B',B,'C',C);
N = 100000;
```

```
            samples = random(pd, N, 1);

            % comprobaciones rapidas
            emp_mean = mean(samples);
            emp_var = var(samples);
            emp_desv_std = std(samples);

            % valores teoricos
            % theo_mean = 0; % simetria centrado en 0
            theo_var = (A^2 + B^2 + C^2 - A*B - A*C - B*C)/18;
            rms = 20*log10(sqrt(theo_var)/x0);

            fprintf('Theorical mean: 0; emp mean: %.2f\n',emp_mean)
;
            fprintf('Theorical var: %.2f; emp var: %.2f\n',theo_var
,emp_var);
            fprintf('Sigma value: %.2f\n',sqrt(theo_var));
            fprintf('rms value in dBFS: %.2f\n',rms)

            % ver histograma y pdf teorica
            xgrid = linspace(A,C,500)';
            figure
            histogram(samples,100,'Normalization','pdf')
            hold on
            plot(xgrid, pdf(pd,xgrid), 'LineWidth',1.5)
            title('Triangular (media 0) -- muestras vs PDF')
            hold off
```

- Option 2: Another way we can generate samples of a random variable following a symmetric triangular pdf as the sum of two independent random variables X_1 and X_2 from a uniform distribution. When two independent random variables with uniform distributions are added, the resulting probability density function (PDF) becomes triangular. This can be understood both intuitively and mathematically.

  Intuitively, if each variable is uniform on $[-a, a]$, there are many pairs that sum near zero but only a few that produce sums near the extremes $\pm 2a$. Hence the PDF peaks at zero and decreases linearly towards the edges.

  Mathematically, let $Z = X_1 + X_2$ with X_1, X_2 independent and uniform on $[-a, a]$. The PDF of Z is the convolution of the two uniform PDFs:

$$f_Z(z) = (f_{X_1} * f_{X_2})(z) = \int_{-\infty}^{\infty} f_{X_1}(t)\, f_{X_2}(-t+z)\, dt.$$

  Carrying out the convolution yields the triangular PDF supported on $[-2a, 2a]$:

$$f_Z(z) = \frac{2a - |z|}{4a^2}, \qquad |z| \leq 2a.$$

  If we want the triangular distribution to have support $[-x_0, x_0]$, we must choose $a = x_0/2$. In that case the PDF simplifies to

$$f_Z(z) = \frac{x_0 - |z|}{x_0^2}, \qquad |z| \leq x_0,$$

  Adding two uniform random variables with 0 mean, results in another random variable with 0 mean.

$$E[Z] = E[X_1 + X_2] = E[X_1] + E[X_2] = 0 + 0 = 0.$$

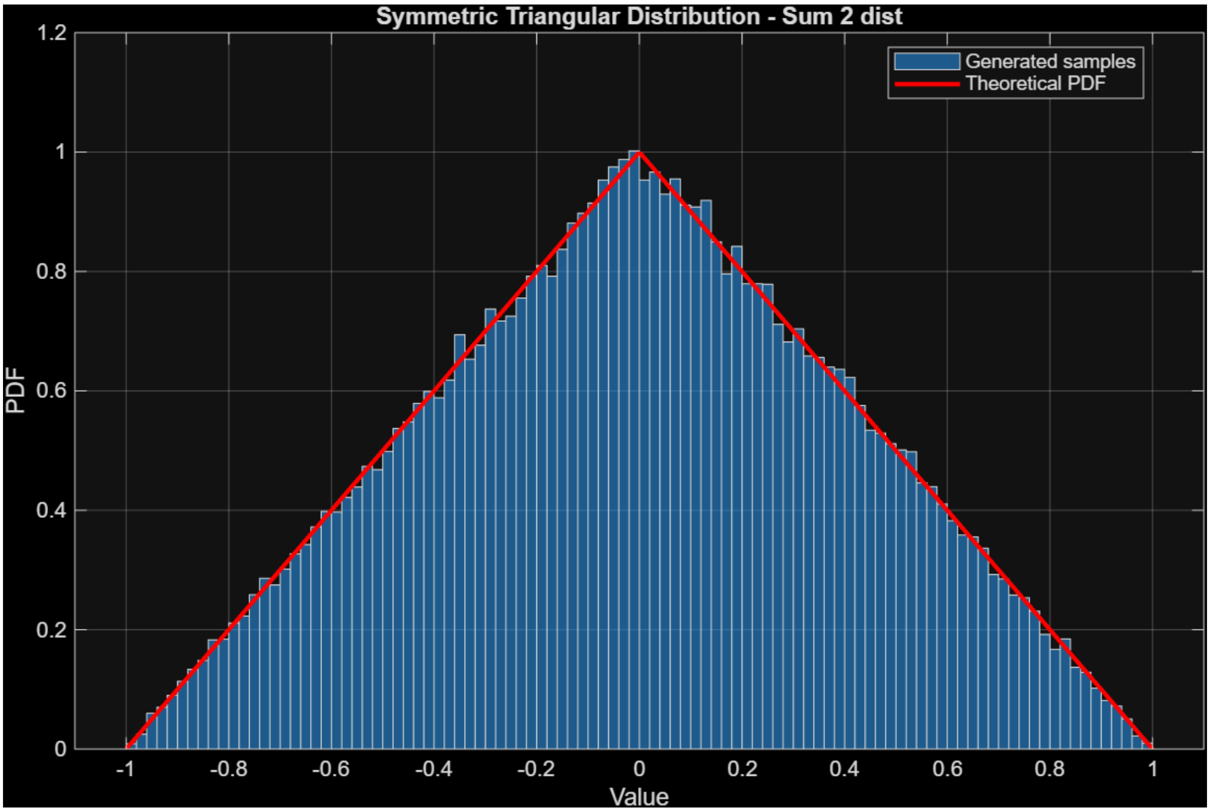
The variance of the sum of two independent random variables is the sum of their variances. So if we want a triangular distribution with variance σ_x (in dBFS), we need to set the variance of each uniform variable to $\sigma_x/2$.

$$Var(Z) = Var(X_1 + X_2) = Var(X_1) + Var(X_2) = \sigma_x/2 + \sigma_x/2 = \sigma_x.$$

For a uniform on $[-a, a]$ we have $\mathrm{Var}(X_i) = a^2/3$. Taking $a = x_0/2$ gives $\mathrm{Var}(Z) = 2 \cdot (x_0/2)^2/3 = x_0^2/6 = \sigma_x$, as required.

The result of the mean and var commands are as follows:

- Empirical mean: 0.00003, wich is close to 0, very close to our target mean.
- Empirical variance [dB]: -7.80701, wich is very close to our target variance.



we can doit as follows: REVISAR!!

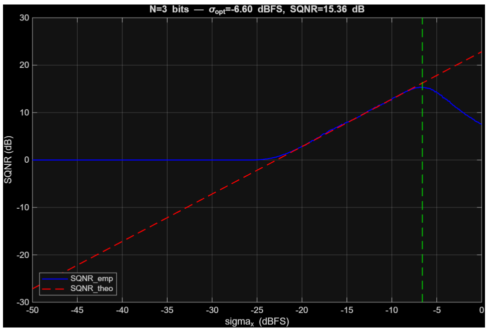
```
x0=2;
sigma0 = x0/sqrt(2);
N = 100000;

c = sigma0 * sqrt(3/2);

x1 = (2 * rand(N, 1) - 1) * c;
x2 = (2 * rand(N, 1) - 1) * c;
```

```
            y = x1 + x2;

            sample_mean = mean(y);
            sample_var = var(y);
            sample_rms = std(y);

            fprintf('--- Validation ---\n');
            fprintf('Target Mean: 0.0\n');
            fprintf('Sample Mean: %f\n\n', sample_mean);

            fprintf('Target Variance (sigma0^2): %f\n', sigma0^2);
            fprintf('Sample Variance: %f\n\n', sample_var);

            fprintf('Target RMS (sigma0): %f\n', sigma0);
            fprintf('Sample RMS: %f\n\n', sample_rms);

            figure;
            histogram(y, 100, 'Normalization', 'pdf', 'DisplayName'
, 'Generated Samples');
            grid on;
            hold on;

            a = 2*c;
            x_pdf = linspace(-a, a, 400);
            y_pdf = (1/a) * (1 - abs(x_pdf)/a);
            plot(x_pdf, y_pdf, 'r-', 'LineWidth', 2.5, 'DisplayName
', 'Theoretical PDF');

            title('Symmetric Triangular Distribution');
            xlabel('Random Variable Value');
            ylabel('Probability Density Function (PDF)');
            legend;
            hold off;
```

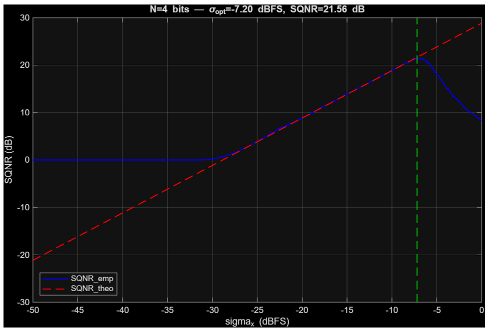
Question: Take $10 \cdot 2^{10}$ of these triangularly distributed samples, quantize them, and estimate the SQNR empirically for $N =$ 3, 4, 5 and 6 bits. Do this for σ_x varying in the range $[-50, 0]$ dBFS and in steps of $0{,}1$ dBFS. Plot the resulting curves (SQNR in dB vs. σ_x in dBFS) along with the theoretical expression

$$\mathrm{SQNR} = 6{,}02N + 4{,}77 - 20\log_{10}\frac{\mathrm{FS}}{\sigma_x} \qquad \textbf{(dB).} \tag{1}$$

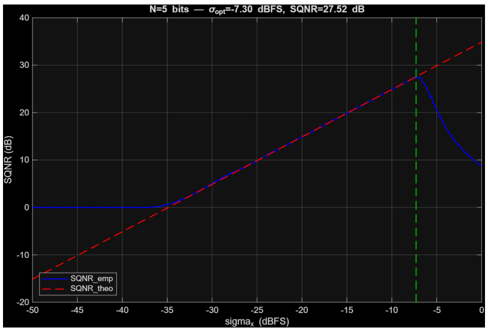
Are there any differences between the theoretical and empirical curves? If so, how do you explain them?



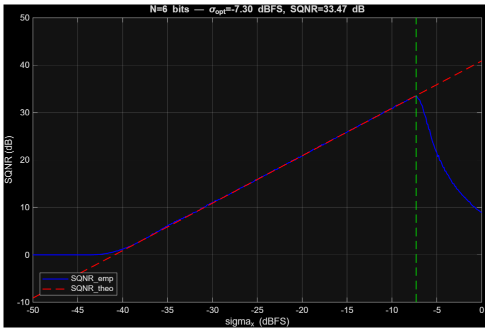
(a) N=3 bits - σ_{opt}=-6.60 dBFS, SQNR=15.36 dB



(b) N=4 bits - σ_{opt}=-7.20 dBFS, SQNR=21.56 dB



(c) N=5 bits - σ_{opt}=-7.30 dBFS, SQNR=27.52 dB



(d) N=6 bits - σ_{opt}=-7.30 dBFS, SQNR=33.47 dB

Figura 1: SQNR vs σ_x (dBFS) for triangularly distributed input at different quantization resolutions.

The comparison between theoretical and empirical SQNR curves for triangularly distributed inputs is shown in Figure 1. The red line represents the theoretical SQNR curve, while the blue line represents the empirical SQNR values obtained from quantizing the triangularly distributed samples.

Empirical curves deviate from the straight theoretical line for two practical reasons. At very small σ_x values, quantization noise is no longer uniformly distributed and we have no enough bits for quantization. At large σ_x values, clipping occurs, distorting the signal and reducing SQNR below theoretical predictions. Otherweise, in the mid-range of σ_x values, empirical results closely follow theoretical expectations and reaches the maximum SQNR (marked in the description of each image).

When we increase the number of bits N, the curve starts to follow the theorical curve with smaller values of σ_x. We reach a point where optimum σ_x value (where SQNR is maximized) approaches the theoretical value of -7.78 dBFS calculated earlier.

Question: In view of your results, what are the optimum values (regarding SQNR) of σ_x (in dBFS), and for the different resolutions analyzed (3 to 6 bits)? Does this agree with your intuition (see first point above)?

ADD

Question: Repeat the previous points, but now using normally distributed input

samples with zero mean and standard deviation σ_x.

ADD

4 Task 4

Assume a full-scale sinusoidal input with $f_0 = 37{,}1094MHz$, and let the FFT size be M = 1024. Generate $15 \cdot M$ samples of $x(t)$ (at fs = 100 MHz) and quantize them to N = 12 bits. Break the vector xq of quantized samples into 15 size-M blocks using, e.g., the command reshape:

```
xqblocks = reshape(xq, M, 15);
```

so that each column of the $M \times 15$ matrix xqblocks will contain the corresponding block of size M . Now, since the fft command computes the FFT columnwise, in order to apply an M -point FFT to each block, we simply make

```
X = fft(xqblocks, M);
```

Average the squared magnitude of the DFT coefficients over the 15 blocks and plot the results between 0 and fs/2, in dBFS. Observe the location and peak value of the principal frequency component, as well as the value of the noise floor. Do your observations agree (quantitatively) with what you would expect?

5 Task 5

Question: Plot $g_\gamma(x)$ vs. x in the range $x \in [-\mathrm{FS}, \mathrm{FS}]$ for $\gamma = 0$, 1 and 2. For input signals whose values are always much smaller than FS (in absolute value), what will be the effect of the nonlinearity?

xd

Question: Modify the code in `quanti.m` and write a Matlab function `dquanti.m` implementing this nonuniform quantizer. The format should be similar to that of `quanti.m`, but including an additional input parameter gama:

```
xq = dquanti( x, FS, Nbits, gama );
```

xd

Question: Generate samples (at 100 MHz) of a full-scale sinusoid with $f_0 = 6{,}8359$ MHz. Quantize them to $N = 11$ bits using $\gamma = 0{,}003$ in `dquanti`. Determine the SFDR in dBFS using an FFT size $M = 2048$, and then with $M = 512$. Does the SFDR depend on the FFT size? Does the noise floor depend on the FFT size? How do you explain this?

xd

Question: Using $M = 2048$, repeat the previous step for $\gamma = 0{,}01$ and $0{,}1$. Are the spectral spurs located where you would expect?

xd

Question: Set now the amplitude to $\frac{\mathrm{FS}}{3}$. Using $M = 2048$, measure the SFDR and express it in both dBFS and dBc for $\gamma = 0{,}005$, $0{,}05$ and $0{,}1$. Will these values change if you repeat the analysis with $M = 512$?

xd

Question: Consider now samples (at 100 MHz and with 11-bit resolution) of a sinusoid with frequency $3{,}3202$ MHz and amplitude $\frac{\mathrm{FS}}{2}$. Obtain the THD for this nonuniform ADC with $\gamma = 0{,}3$ under the IEEE 1241-2000 specification, expressed in both dB and percentage.

xd

6 Task 6

Question: If the rms value of the aperture jitter is 20 ps, and the input signal is a full-scale sinusoid with frequency f_c, for which values of f_c will the aperture error power dominate the quantization noise power?

xd

Question: If the rms value of the aperture jitter is 20 ps, and the input signal is a 3-MHz sinusoid, for which values of the amplitude (in dBFS) will the aperture error power dominate the quantization noise power?

xd

Question: Simulate the effect of aperture jitter on a full-scale sinusoid with frequency $40{,}03905$ MHz. Consider two cases: $\sigma_\tau = 10$ ps and $\sigma_\tau = 0{,}1$ ps respectively. Perform a 1024-FFT analysis of your data and check whether the perceived noise floor is at the expected level.

xd

Question: Neglecting other possible sources of distortion, the total SNR is given by the ratio of the signal power to the sum of the powers of the noises due to jitter and quantization. Plot the theoretical total SNR (in dB) vs. input frequency over the range 0.1–100 MHz, assuming a full-scale sinusoid and for $\sigma_\tau \in \{10, 20, 40\}$ ps,

$N \in \{10, 14\}$ **bits (so that you should have six graphs in a single plot, whose x-axis should be in log scale). Comment on your results.**

xd