periodica polytechnica

Electrical Engineering
53/3-4 (2009) 179–185
doi: 10.3311/pp.ee.2009-3-4.09
web: http://www.pp.bme.hu/ee
© *Periodica Polytechnica 2009*

RESEARCH ARTICLE

Limitations of the noise model of roundoff for the FFT

Vilmos Pálfi / *István* Kollár

Received 2010-05-25

Abstract

The general assumptions made about roundoff noise are that its samples form a signal-independent white sequence, and they are uniformly distributed between $\pm q/2$, where q equals the least significant bit (LSB). While these are often true, strange cases may appear, e.g. misleading peaks can occur in the spectrum. In this paper the roundoff error of the fixed-point and floating point fast Fourier transform is investigated. It reproduces the results of Welch (1969) with modern tools, revisits his simulations, and investigates the consequences of the violation of the above assumptions for almost pure sine waves. The maximum amplitude of spurious peaks is determined and the amount of the decrease in the dynamic range is given.

Keywords

floating-point · fixed-point · FFT · roundoff error

Acknowledgement

This work has been supported by the Hungarian Scientific Research Fund (OTKA), grant number TS-73496.

Vilmos Pálfi

Dept. of Measurement and Information Systems, BME, H-1521 Budapest, Magyar tudósok krt. 2, Hungary
e-mail: palfi@mit.bme.hu

István Kollár

Dept. of Measurement and Information Systems, BME, H-1521 Budapest, Magyar tudósok krt. 2, Hungary
e-mail: kollar@mit.bme.hu

1 Introduction

The Fast Fourier transform (FFT) is one of the most effective algorithms of digital signal processing. It decreases the need for complex additions/multiplications from the order of magnitude of N^2 to $N \cdot \log_2 N$. This is why it has been so popular since the sixties [2]. The basis is the so-called butterfly (see Fig. 1). The FFT was executed in the sixties with the then-available fixed-point arithmetic. High-speed FFTs are still done in fixed point. However, for the FFT, number representation needs to be modified: the inherent amplitude increase [1] requires that a so-called block-float exponent (common to all samples) is used to continuously utilize the full bit length (avoid both overflow and roundoff error too large with respect to the signal). Therefore, "Fixed-point FFT" means block-float FFT. The popularity of block-float FFT required a thorough analysis of the roundoff errors, e.g. [4, 5, 8, 9]. This was done partly theoretically (with the noise model), partly experimentally, using simulation. This latter was done e.g. in [9] by direct computation in Fortran. Today, simulation has become possible in modern software (Matlab) in a reproducible way, using general quantization tools.

2 Reproducing classical results

In his paper, Welch analyzed the radix-2 DIT algorithm[1] of FFT. He studied the effect of roundoff on the output of operations at each stage of the FFT. Block-float number representation was used. In this, the numbers are stored in fix bit length. An exponent is common for all numbers, so it is stored only once, not for every number. The mantissa is $1+B$ bits long where B is the number of the fraction bits, and one bit is used for the sign. The largest number is downscaled until its absolute value is in (0.5,1). All the other numbers are downscaled by the same amount, thus a common exponent can be used:

$$X_{\text{input}} = [0.07\ 0.334\ 0.707\ 0.043 \ldots 0.002] \times 2^{13} \qquad (1)$$

In this example the common exponent is 13, and the biggest represented sample equals $0.707 \times 2^{13} \approx 5.79 \times 10^3$.

[1] DIT is an abbreviation decimation-in-time, the way of reducing the size of DFTs. Radix-2 stands for the size of the butterflies (they can also be radix-4, radix-3 etc.).

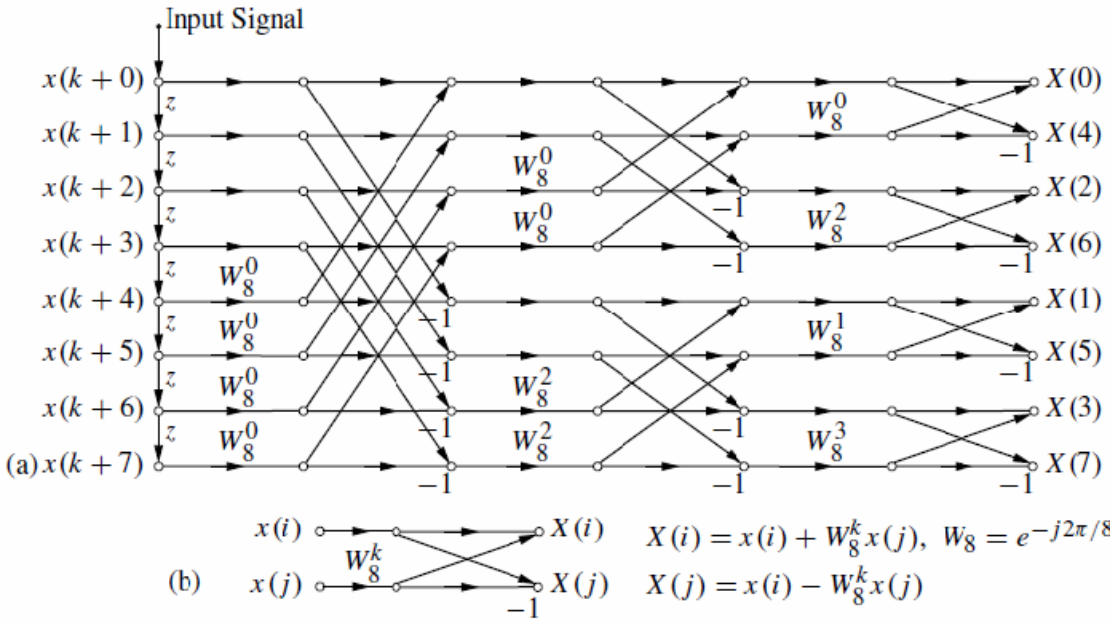

Fig. 1. FFT a) decimation-in-time (DIT) flowchart for N = 8; b) the basic butterfly

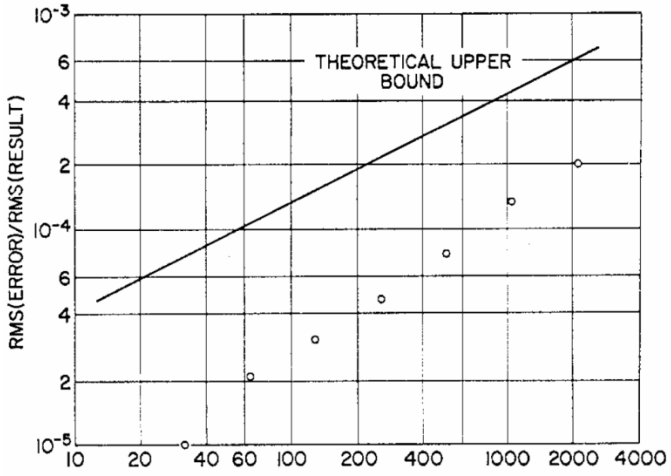

Fig. 2. Welch's result for clear sine wave input

During the algorithm, due to the summations and multiplications, the numbers have to be downscaled. For example if the second element of (1) is added to the third one, the result is $(0.707 + 0.334) \times 2^{13} = 1.041 \times 2^{13}$, but then the mantissa is larger than 1 and this would cause an overflow, so before the addition the numbers are downscaled:

$$X = [0.035\, 0.167\, 0.354\, 0.022 \ldots 0.001] \times 2^{14} \qquad (2)$$

Now $(0.167+0.354)\times 2^{14} = 0.521 \times 2^{14}$ so the result's mantissa is less than 1. The roundoff due to downscalings (discarding or rounding the rightmost bit) can be considered as adding noise (the error term) to the exact value.

Welch gave an upper bound for the error of the fixed-point FFT output, assuming rescaling at each stage:

$$\frac{RMS(\text{error})}{RMS(\text{result})} \le h_{\max} = \frac{2^{\frac{M+3}{2}} \times 2^{-B} \times C}{RMS(\text{input})} \qquad (3.a)$$

In this equation $M = \log_2 N$, B is the number of fraction bits, and C is a constant between 0.4 and 0.9 (depending on the signal shape). Therefore, $h_{\max}$ is equal to:

$$h_{\max} = \frac{\sqrt{8N} \times 2^{-B} \times C}{RMS(\text{input})} \qquad (3.b)$$

This formula was verified by using simulations, with B=17. We have reproduced these simulation results for the FFT of a series of random numbers in (0, 1), then in (−1, 1). These two cases are significantly different, because in the first case the input has a nonzero mean value and this causes many more downscalings than zero mean noise does. The runs confirmed the theoretical formula as an upper bound. In the first case, the noise-to-signal ratio (*NSR*) increased proportionally with log(N), as it is expected from (3). Welch defined the *NSR* for the whole spectrum (a possible alternative is to calculate the *NSR* only for the bin where the sine appears).

3 New results for a pure sine wave

Simulation was also done with the pure sine wave also used by Welch (the input signal was $x[k] = \sin(2 \cdot \pi \cdot k/8)$, $k = 0 \ldots N-1$). To our surprise, the simulation gave significantly different results from Welch's. Fig. 2 shows Welch's result, as the noise-to-signal ratio increases with N. However, in our simulation h did not increase with $\sqrt{N}$, it was even constant. We speculate that Welch's simulations were probably done with a sine wave somewhat different from the text (maybe some noise was added).

In order to understand in detail what happens for the FFT of a pure sine wave, look into Fig. 3. It illustrates the 8-point FFT of a sine wave. Compare this now to the FFT of N=16 samples (Fig. 4). For this, the samples for k= 0...7 are repeated for k = 8...15. Here the second eight outputs of the first stage are exactly zero because the corresponding sample pairs have to be

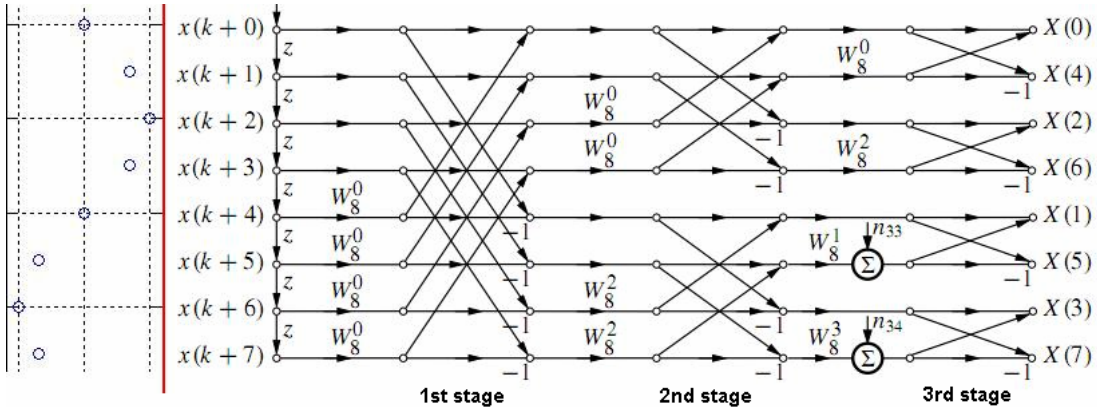

Fig. 3. Sine wave input and 8-point FFT

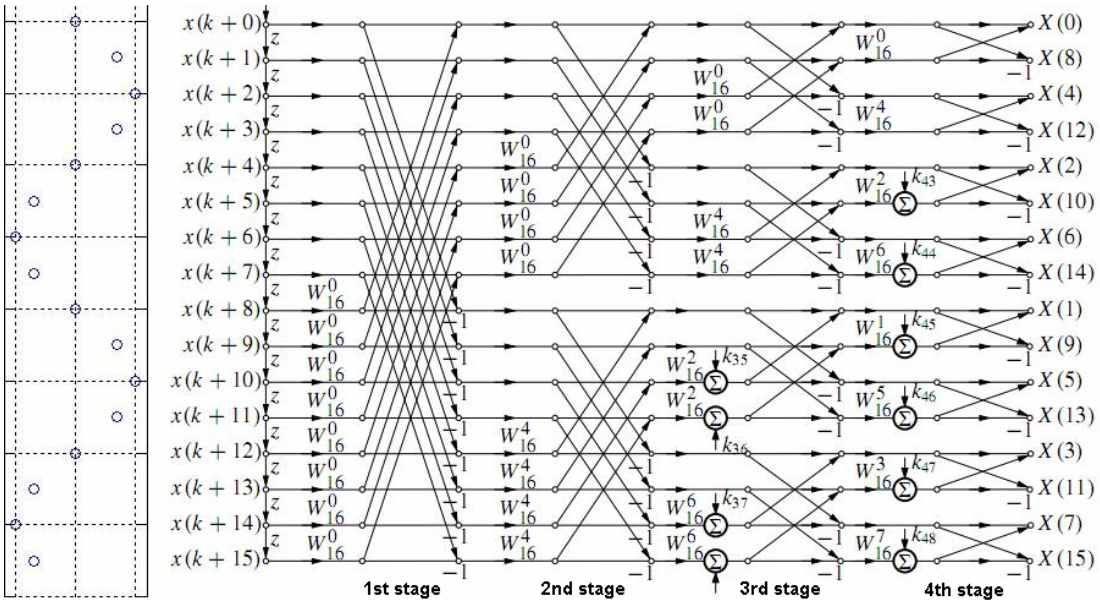

Fig. 4. Sine wave input and 16-point FFT

subtracted from each other:

$$X_1[k+8] = X_0[k] - X_0[k+8] \tag{4}$$

In the equation $X_0[k]$ is an input of the first stage (a sample of the input signal), and $X_1[k]$ is an output of the first stage (and also an input of the second stage). In the next stages in the lower half part of the FFT all operations are made on these zero elements, so the second eight outputs are all exactly equal to zero.

The first eight outputs of the first stage equal two times the input samples, because the corresponding sample pairs have to be added to together:

$$X_1[k+8] = X_0[k] + X_0[k+8] = 2 \times X_0[k] \tag{5}$$

The addition causes an overflow because $X_0[2] = X_0[10] = 1$ and $X_1[2] = X_0[2] + X_0[10]$, so the algorithm has to downscale the samples, and the block exponent increases by one. At the first eight outputs of stage 1 the samples are multiplied by two, then they are divided by two, thus nothing happened with their value.

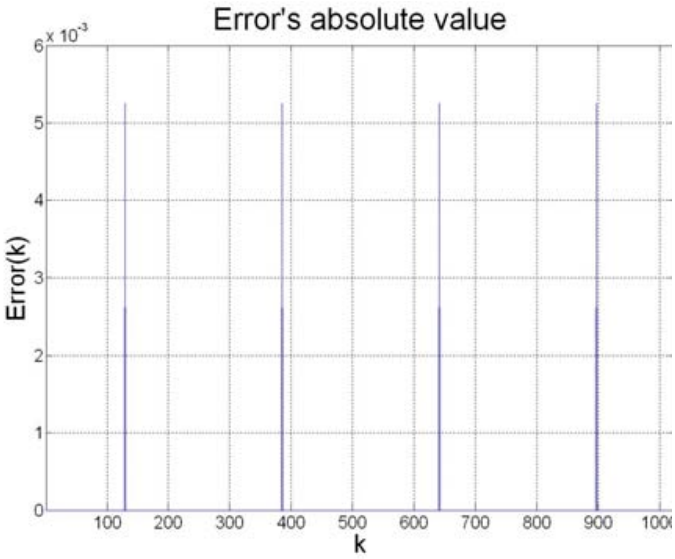

Fig. 5. Error pattern for pure sine wave and fixed-point arithmetic

At the next stages the same operations are made on the first eight samples as in the case of the 8-point FFT in Fig. 2. The 16-point FFT's first eight output samples are equal to the 8-point

FFT's output before rescaling (multiplying the outputs value with 2^{be} where 'be' is the block exponent's value), the second 8 outputs are equal to zero, only the block exponent is larger by 1 than in the 8 point case. Therefore, the same operations cause the same roundoff errors. After rescaling, the output's RMS for the N=16 case is $\sqrt{2}$ times the output's RMS for the N=8 case. The noise's RMS for N=16 is also $\sqrt{2}$ times the RMS for N=8, so the noise-to-signal ratio remains unchanged. It does not depend on the number of samples.

We also checked the pattern of the roundoff error and we found it very regular (Fig. 5).

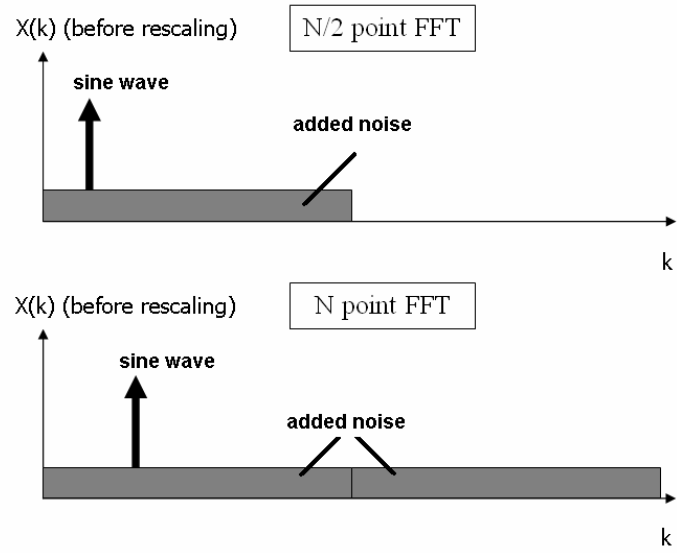

Fig. 6. N/2 and N point FFT result for a sine wave input, and the added noise at the last stage

Our simulations and considerations indicate that strange and until now not predicted – although not very large – things may happen during the FFT of pure or close-to-pure sine waves. Therefore, we have investigated such cases in more detail. We start it here with a brief overview of the results of the noise model.

4 Investigating the noise model of roundoff for FFT of a sine wave

Rounding errors are usually modeled with quantization noise. This noise is signal-independent and consists of uncorrelated samples, so it is also called independent white noise. Its samples are uniformly distributed in (−LSB/2, LSB/2). In order to investigate the effect of this noise model, our simulation of roundoff allows substitution of each rounding operation by the addition of independent white noise. This means that instead of applying rounding as it happens in a digital signal processor, a random value is generated in (−LSB/2, LSB/2) and added to the result's exact value. By this way we can see how similar is the effect of the noise model of quantization for a pure sine wave's FFT with block-float number representation.

Applying the noise model of quantization as above, it is found that the noise-to-signal ratio increases indeed with $\log(N)$. This result corresponds to Welch's (Fig. 2), in contrast with our results for the true fixed-point roundoff of a sine wave. An explanation for the *NSR*'s behavior for the noise model can be given as follows. Let us check how the roundoff noise, added at each stage to the signal, applies to the FFT's output. Due to the downscalings throughout the algorithm, the noise added to signal in earlier stages has much less effect to the output than the noise added in later stages. So it is a reasonable approximation to consider only the noise which is added in the last stage, and we neglect the noises added earlier. The *NSR* clearly does not depend on scaling (the DFT of a coherently sampled sine is the same in the mantissa, independently of N). The total power of the noise added to the mantissas is proportional to N. Thus, the NSR is proportional to log(N). In conclusion, the results expected from (1) have been obtained. This gives rise to the following implications:

- Welch's calculations assuming the noise model of roundoff are correct,
- for a pure sine wave the noise model (PQN) assumption is not correct,
- the simulations of Welch probably included an unknown factor (probably some input noise was added).

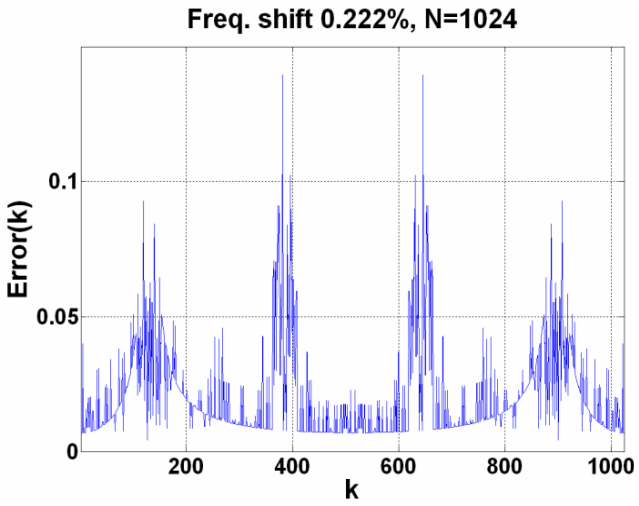

Fig. 7. Absolute value of the error in a very bad case, N=1024

5 Strange cases for noncoherently sampled pure sine wave

The previous simulations were made with coherent sampling, so we got an integer number of cycles. For a pure sine wave the roundoff error behaves regularly, its pattern is absolutely not noise-like (Fig. 5). If the sampling is not coherent, the pattern of the error is more noise-like. In further simulations we study how the errors behave when the sampling is almost coherent. It is the border-land between the above mentioned two cases. The sampling frequency was slightly shifted. On the next figures the amount of the frequency shift is given in bin percent. For example if the input is the $x[i] = \sin(2\pi i/8)$ and an $N = 1024$

point FFT is performed, the power of the sine wave appears in the k=128th bin. Shifting the sampling frequency by "5 bin%" means that the sine wave would "appear" on the k=128.05th "bin".

Simulations were executed to determine how the error's pattern depends on the value of frequency shift. These simulations were run using an N=1024-point FFT to collect more detailed information. We found that for certain sampling frequencies the errors are accumulating at some of the outputs, and these error peaks can be unexpectedly high. A very bad case can be seen in Fig. 7.

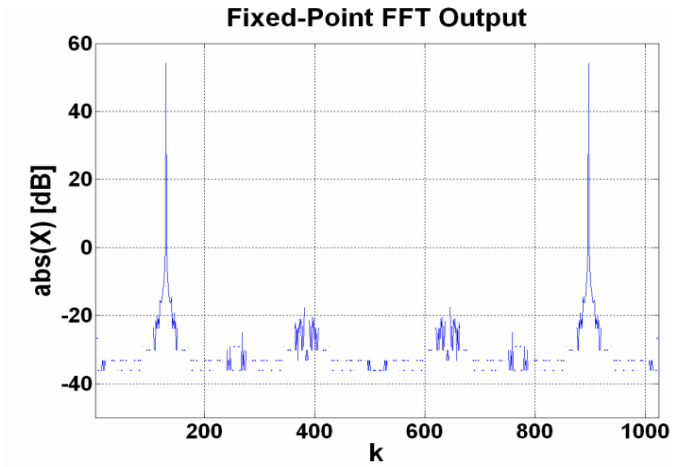

Fig. 8. Output for the fixed-point FFT

Fig. 8 shows the output of the FFT in the previous very bad case.

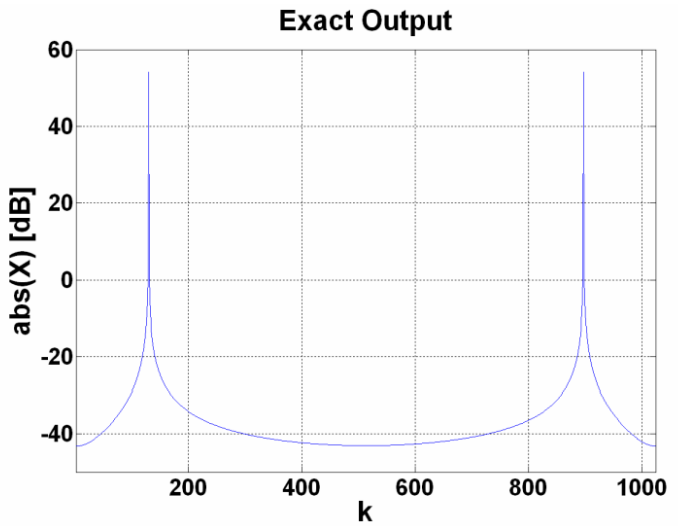

Fig. 9. The exact FFT result

The correct output can be seen in Fig. 9. (We assumed that Matlab's built-in floating point FFT provides the correct result). If the two figures are compared, one can see two peaks which are above −20 dB around k=400 and k=600 in the fixed-point case, due to roundoff errors. Fig. 10 shows the error for deviations from coherent sampling. The deviation is given in bin percent. The other two axes are the bins of the FFT and the error's absolute value. For small deviation levels the errors are small and they are concentrated to a few points (compare with Fig. 5, the two patterns are similar). As the deviation level increases, the error appears at more points. Unfortunately at some points the errors are accumulating, and become so high that sine wave-like peaks appear in the output (0.1875%). This can cause that one can observe sine waves which are actually not there. For larger deviation levels this accumulation behavior disappears, the errors spread to all points of the output, so the noisy-like pattern is restored (3%).

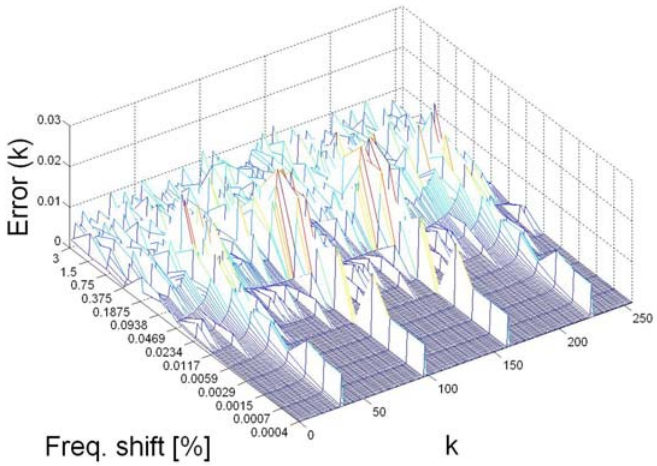

Fig. 10. Error's absolute value on different frequency shift values

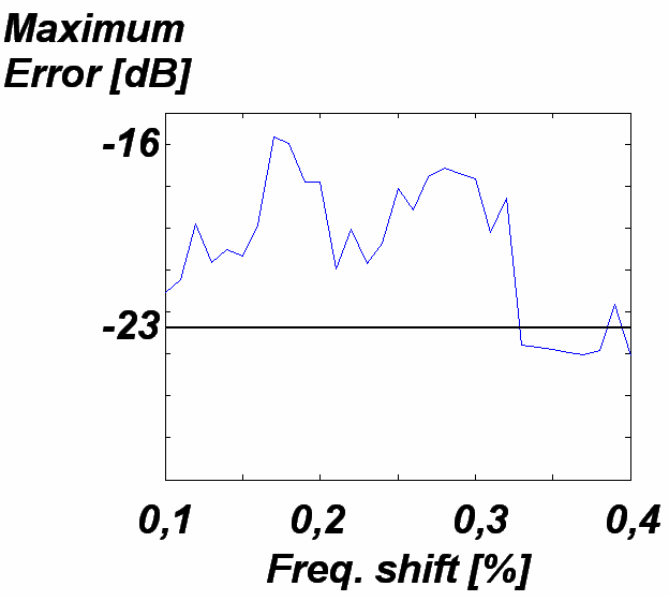

Fig. 11. Maximum value of error peaks compared with the estimated value using the noise model of roundoff

Next we compared these results with results assuming the noise model of roundoff. The absolute value of the FFT's output is Rayleigh distributed. Let us find the value X for which the probability is 99% that every peak of the noise FFT is smaller than X. The FFT's result is symmetric, so it contains about N/2 independent random variables. We neglect the variable at the sine wave's frequency, and also the DC component, so we calculate with 510 variables. P is the probability of one peak belonging to the quantization noise less then X. We also assumed

independent variables. This means:

$$P^{510} = 0.99,\ P = \sqrt[510]{0.99} \tag{6}$$

$$P = 1 - e^{-\frac{x^2}{2\sigma^2}} \tag{7}$$

$$X^2 = 2\sigma^2 \times \ln(\frac{1}{1 - \sqrt[510]{0.99}}) \tag{8}$$

In Eqs. (7) and (8) σ^2 is the variance of the random variables. This value X can be seen in Fig. 11 as a horizontal black line. In a very bad case the dynamic range (the range of a properly detectable sine wave) can be less by 7 dB than its estimated value using the noise model of roundoff.

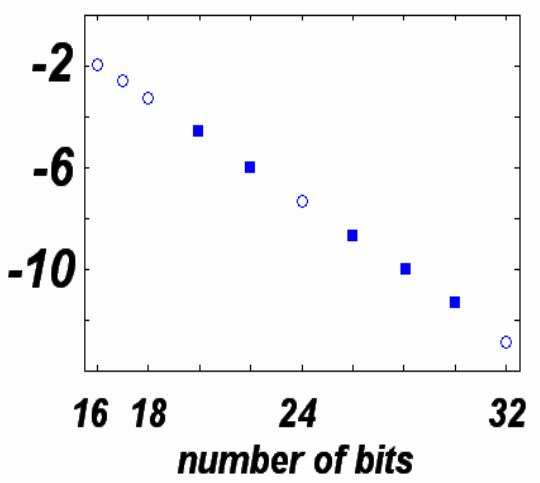

Fig. 12. The maximum of the error for different bit lengths

In previous simulations we stored the numbers on 16 bits. We also checked behavior of the roundoff error when the number of fraction bits is increased. We found that if the number of bits is increased by one, a smaller frequency shift value is needed to reach a similar "very bad case", and the maximum value of the error caused by the error accumulation is halved. When the number of fraction bits is increased by one, the value of LSB is less. So if we apply the noise model in simulations the amount of the added noise is also halved. This means that the 7 dB loss in the dynamic range does not depend on the number of fraction bits. Fig. 12 shows the logarithm of the error's maximum for different bit lengths. Simulations were run for 16, 17, 18, 24, and 32 bits (denoted by the circles). It is easy to fit a line to these points, so the error's maximum can be calculated easily for other bit lengths (denoted with the squares).

6 Noncoherently sampled sine wave with floating-point arithmetic

After these unexpected results we studied the pattern of the roundoff error with floating-point number representation. Floating-point numbers consist of three parts: sign, mantissa, exponent. We ran simulations using IEEE single precision number format. An IEEE single-precision number has 1 sign bit, 8 exponent bits and 23 mantissa bits, its precision is 24 because of the hidden leading bit [1]. N=1024 point FFTs were executed to collect more detailed information. Before the simulations we expected less roundoff error than in the fixed point case, because the floating point arithmetic stores the exponent for all numbers. The unique exponent for every number means there is no need to downscale all numbers when an element of the array is re-quantized (e. g. rounded after a multiplication). So first we thought that the error would be distributed (more or less) uniformly among the 1024 outputs, and there would not be spectacularly outstanding peaks.

Simulation results are surprising again. The error pattern was very similar to the fixed point case (Fig. 13). On the figure i is the ordinal number of the simulations. Instead of uniformly distributed error power there are still special output bins where the significant part of the error is accumulating. It is also noticeable that two of these special bins (the two in the middle) are far away from the frequency of the sine wave. To find a really bad case with higher error peaks we generated random frequency shift values and ran another 3000 simulations. Next we compared these results with result assuming the noise model of roundoff. A floating point number's variance is about $\mathrm{var}\{\upsilon\} \approx 0.180 \cdot 2^{-p} \cdot var\{x\}$, with p the precision and x the signal [1], and it is uncorrelated with the signal, and it is white. The absolute value of the Fast Fourier Transform's roundoff error is Rayleigh distributed. X is the value for which the probability is 99.9% that every peak of the roundoff error is smaller than X. In the case of N=1024 point FFT this means that only one error peak is expected to be larger than the X value. This X value can be seen on Fig. 14 as the horizontal line. In a very unfortunately situation 15 dB loss can be experienced in the dynamic range when compared to the noise model. It is really surprising, mainly because the maximum loss found in the fixed-point case is 7 dB.

The conclusion is that for pure or almost pure sine waves the roundoff error could behave irregularly if the input is noiseless. This special behavior does not depend on the number format (floating point or fixed point). The high peaks due to quantization noise can appear far away from the frequency of the sine wave. The exact frequency cannot be predicted, so the FFT's output should be used with some extra care. To our experience, these unwanted peaks disappear even for proper input dither. By adding noise in $[-8LSB,+8LSB]$ (where LSB is the least significant bit) to the signal the high error peaks disappear from the output. So sometimes noisy input causes less disturbing roundoff error. Another noticeable conclusion is that the noise model of quantization is not as universal as we thought.

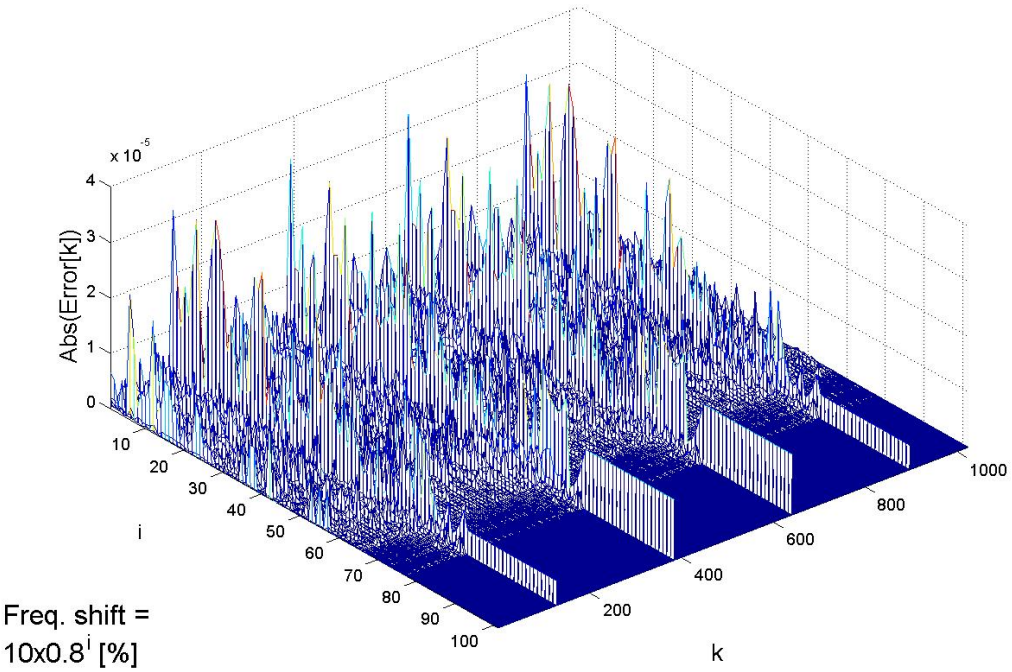

Fig. 13. Error's absolute value on different frequency shift values for N=1024 point FFT with IEEE single precision arithmetic

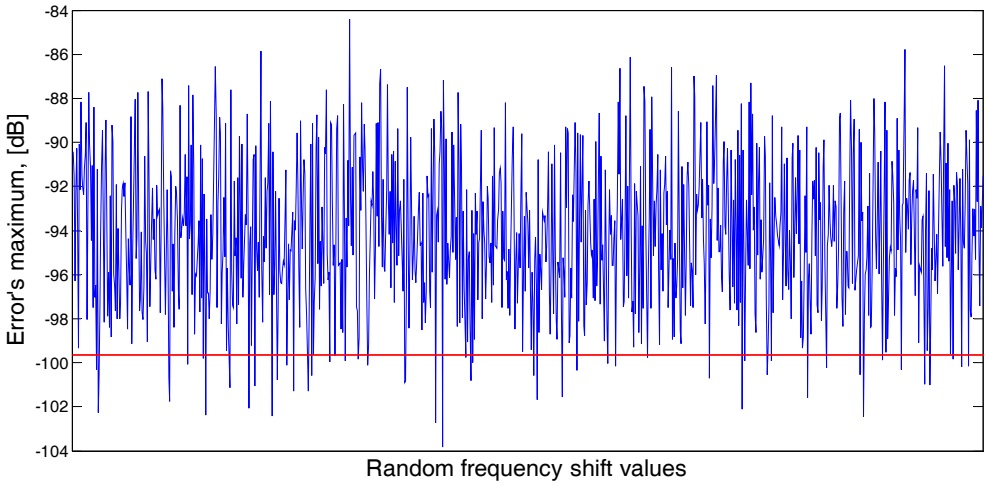

Fig. 14. Maximum value of error peaks compared with the estimated value using the noise model of roundoff

References

1 **Widrow B, Kollár I**, *Quantization Noise: Roundoff Error in Digital Computation, Signal Processing, Control, and Communications*, Cambridge University Press, Cambridge, UK, 2008, available at `http://www.mit.bme.hu/books/quantization/`.

2 **Brigham E O**, *The Fast Fourier Transform*, Prentice-Hall, Inc., Englewood Cliffs, 1974.

3 **Jones D L**, *Decimation-in-time (DIT) Radix-2 FFT*, 2006, available at `http://cnx.org/content/m12016/latest/?format=pdf`. Version 1.7: Sep 15, 2006, Connexions Project.

4 **Horváth G**, *Kvantálási hibák FFT alapú spektrumbecsléseknél*, 1987. Cand. Sci. Thesis.

5 **Kaneko T, Liu B**, *Accumulation of round-off error in fast Fourier transforms*, Journal of the Association for Computing Machinery **17** (1970), no. 4, 637-654.

6 **Kollár I**, *Simulation of roundoff in Matlab*, available at `http://www.mit.bme.hu/book/quantization/Matlab-files.html`.

7 **Nguyén Quang Hung**, *A gyors Fourier-transzformáció kerekítési hibáinak vizsgálata*, 1990. Cand. Sci. Thesis.

8 **Thong T, Liu B**, *Fixed-point fast fourier transform error analysis*, IEEE Trans. ASSP ASSP-24 (1976), 563-573.

9 **Welch P D**, *A fixed-point fast Fourier transform error analysis*, IEEE Transactions on Audio and Electroacoustics **17** (1969), no. 2, 151-57.