NATIONAL RADIO ASTRONOMY OBSERVATORY
SOCORRO, NEW MEXICO
VERY LARGE ARRAY PROJECT

VLA COMPUTER MEMORANDUM NO. 138

EFFECT OF INPUT TRUNCATION ON THE OUTPUT OF AN FFT

Sumant Krishnaswamy

August 1977

The subject of truncation errors in digital FFT transforms has been extensively explored. The general result has been to suggest that errors rise at a rate of about 0.4 to 0.5 bits per stage. A map problem arising with the VLA is, for instance, a 2048x2048 map, which is a 22 stage FFT suggesting that 9 to 11 bits of precision will be lost from whatever word length the computation is done with. However, little is known about a sudden truncation in the middle of the FFT process. In the large map case, an expensive component is the transposing memory used to hold the intermediate results between row transforming and column transforming. To hold down costs, we would like to hold the word length as small as possible. This, then is an attempt to determine the necessary word length.

Rather than doing a two-dimensional FFT, with different distributions in the two dimensions, the problem is simplified to a transform, truncate, inverse transform case, instead of a transform, truncate, transpose, transform sequence. The consequence of this simplification is the dependence on the assumption that the effect of the truncation error is not strongly dependent on the distribution of the numbers being truncated. In order to find a conservative worst case several distributions were investigated.

The FFT routine used was a single-precision FORTRAN program taken from the IBM System/360 Scientific Subroutine Package. This routine performs discrete, complex Fourier Transforms using the Cooley-Tukey algorithm on a complex, three-dimensional array with each dimension equal to a power of 2. After the program was set up, the routine was tested by transforming a given one-dimensional array, then inverse transforming it and checking that the original array was obtained. The accuracy of the FFT routine was typically 1 part in 10^7 or better.

The method used in this investigation was to generate a complex (Hermitian) array by transforming a given real function, then taking the inverse transform of the array before and after truncation and calculating the resulting error in the output. Only one-dimensional arrays were considered. The procedure is outlined in more detail by the following steps (sketched in Figure 1):

1) Choose a function f(x) to represent the one-dimensional brightness distribution of a source.
2) Generate a one-dimensional array of size $N(=2^n)$ by sampling the above function at N equally spaced points in some interval. The real part of the i^{th} element of the array equals the value of the function at the i^{th} sample of the function and the imaginary parts are all set equal to zero. In this investigation the interval within which the function is sampled has been taken to be $[-1,1]$.
3) Fast Fourier Transform this array to simulate the visibility data for the source. Since the array generated in step (2) is real, its transform is Hermitian.
4) Take the inverse transform of the visibility data to get back the brightness distribution. This will differ from the original function by about 1 part in 10^7 because of the error inherent in the FFT procedure. For the same reason, the imaginary parts will also differ from zero by a similar amount. The real part of this output array is referred to as the signal. Calculate the peak value of this signal.
5) Truncate the real and imaginary parts of each element of the visibility data to a specified number of bits of precision.

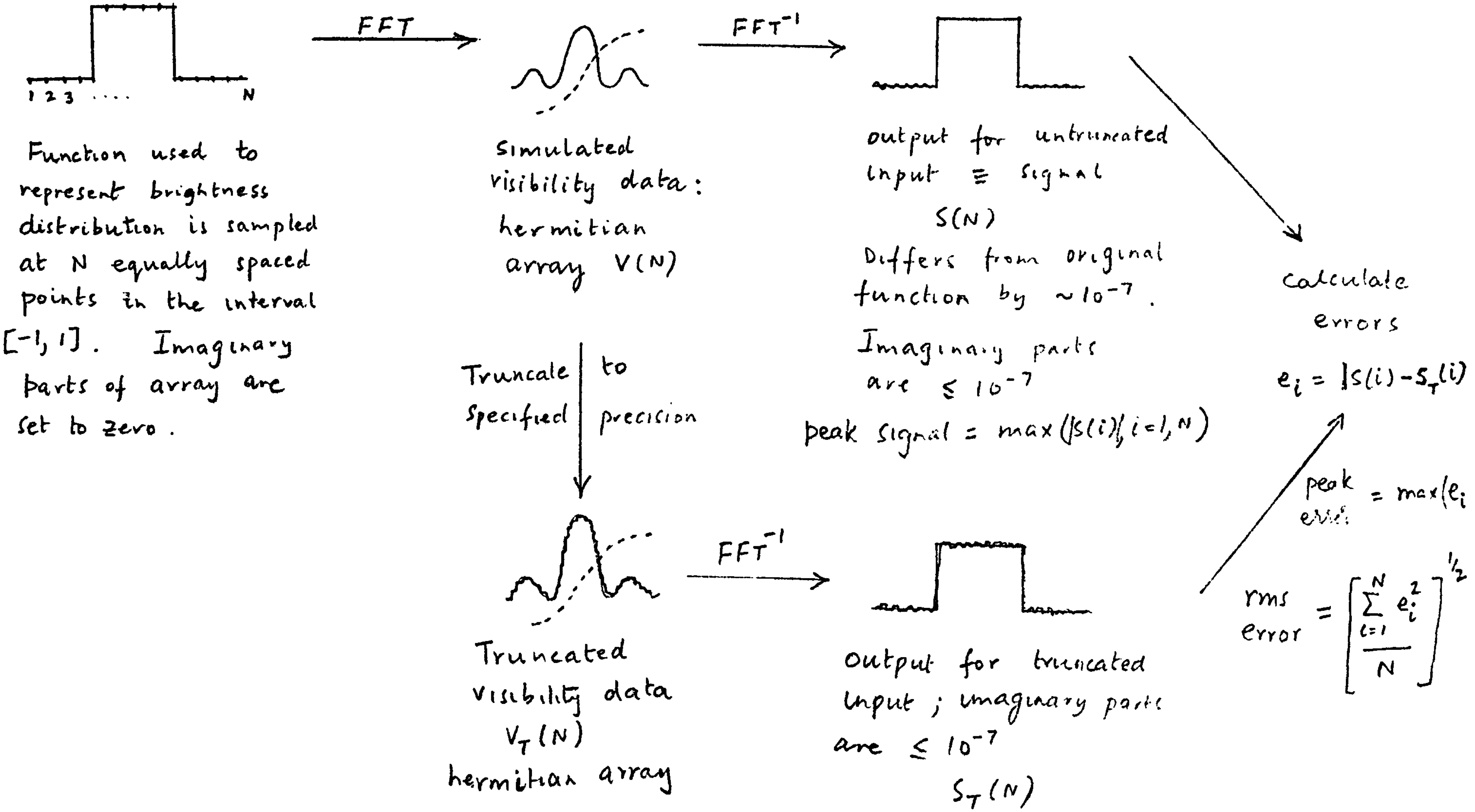
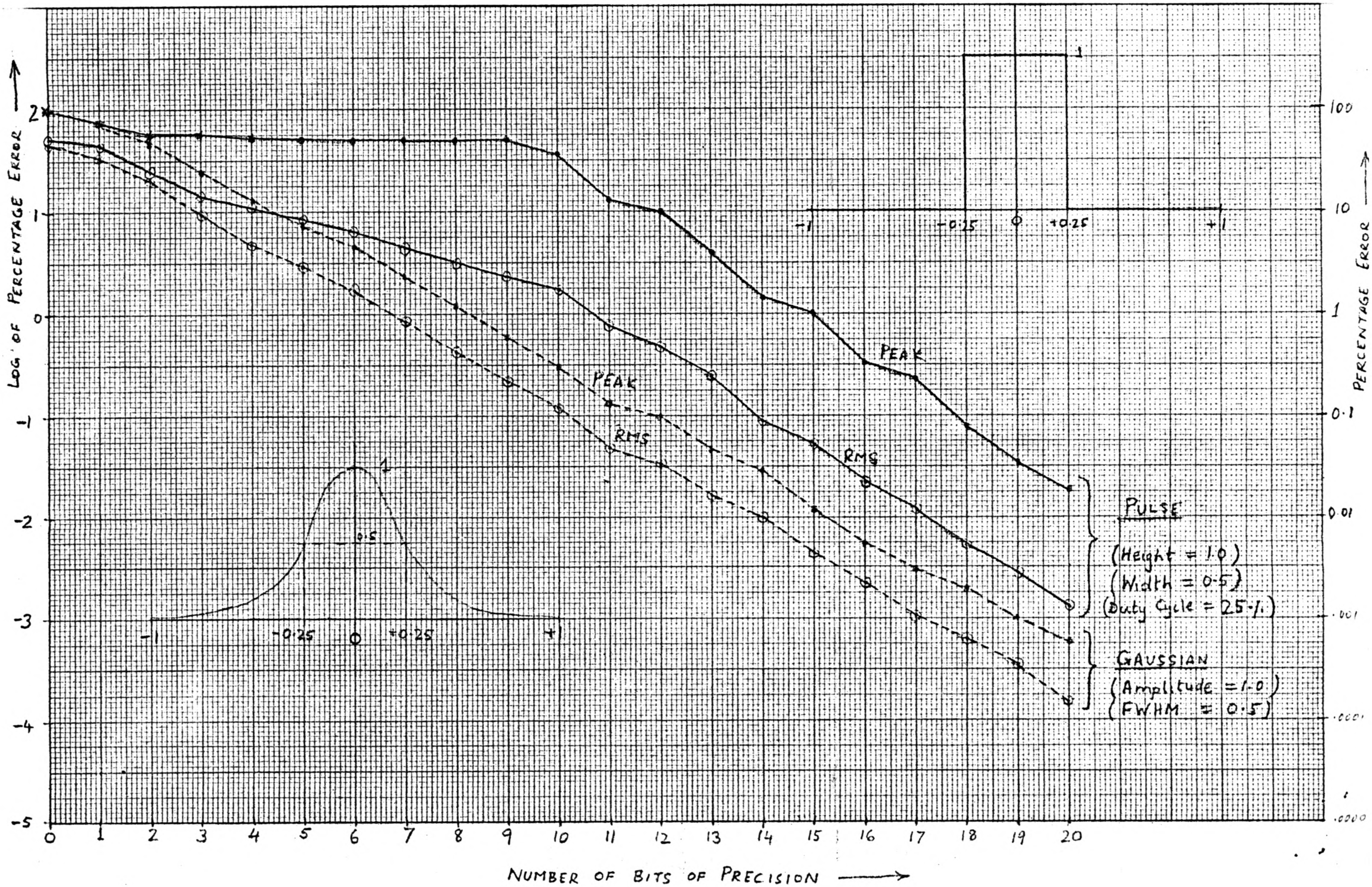

1 2 3 . . . . N
Function used to represent brightness distribution is sampled at N equally spaced points in the interval [-1, 1]. Imaginary parts of array are set to zero.
FFT
simulated visibility data: hermitian array V(N)
Truncate to specified precision
Truncated visibility data $V_T(N)$ hermitian array
FFT^{-1}
output for untruncated input ≡ signal S(N)
Differs from original function by $\sim 10^{-7}$.
Imaginary parts are $\leq 10^{-7}$
peak signal = max(|S(i)|, i=1,N)
FFT^{-1}
output for truncated input; imaginary parts are $\leq 10^{-7}$
$S_T(N)$
calculate errors
$e_i = |S(i) - S_T(i)$
peak error = max(e_i
rms error = $\left[\frac{\sum_{i=1}^{N} e_i^2}{N}\right]^{1/2}$

Figure 1

Procedure used for calculating error in FFT output

FIGURE 2

N = 2048 ERROR VS BITS PULSE & GAUSSIAN



6) Now inverse transform the truncated visibility data. Since the truncated data is still Hermitian, this output also is essentially real. Compare this output with the output for the untruncated input (i.e., with the signal) and calculate the peak value of the error and the rms value of the error. Express these errors as a percentage of the peak value of the signal:

$$\% \text{ peak error} = \frac{\text{peak value of error}}{\text{peak value of signal}} \times 100$$

$$\% \text{ rms error} = \frac{\text{rms value of error}}{\text{peak value of signal}} \times 100$$

7) See how these errors vary with the precision, the array size and the type of function used to generate the visibility data.

These steps are shown schematically in Figure 1.

Some tests were made to check that the procedure was working correctly:

1) Since the truncation is done in terms of the number of bits, scaling the simulated visibility data V by powers of 2 should have no effect on the percentage errors. In other words, if the real and imaginary parts of each element of V are multiplied by some power of 2, then the percentage errors should remain unchanged. This was verified.

2) The truncation routine works in a fashion such that if the precision specified is zero bits, the truncated visibility data will be identically zero and its FFT will also be identically zero. The peak error will then equal the peak value of the signal and so the percentage peak error will be 100% irrespective of the array size and the type of function. This was also verified.

RESULTS:

Figure 2 shows the variation in the percentage errors as a function of the precision from 0 to 20 bits for an array size of 2048 = 2^{11}.

Both the percentage peak and rms errors are plotted on a logarithmic scale. Two functions have been considered here:

1) A pulse with a height of unity and a width of 0.5. Since the interval is $[-1,1]$, this means a duty cycle of 25%,

$$\text{i.e.,} \quad f(x) = \begin{cases} 1 & |x| \le 0.25 \\ 0 & 0.25 < |x| \le 1 \end{cases}$$

2) A Gaussian with an amplitude of unity and a full width at half-maximum (FWHM) of 0.5,

$$\text{i.e.,} \quad f(x) = e^{-x^2/\sigma^2}, \; |x| \le 1$$

$$\text{where} \quad \sigma^2 = (\text{FWHM})^2/4 \ln 2$$

The data for Figure 2 are given in Table 1. Some points of interest are:

1) Percentage errors increase roughly by a factor of 2 for every bit cut off. This is true for both functions for a precision $\ge$ 10 bits.

2) For the pulse: a) the peak error stays constant until approximately 9 bits, then falls off.

   b) the peak error is roughly 20 times the rms error at the same precision ($\ge$10 bits).

   c) peak error $\le$ 1% for precision $\ge$15 bits and rms error $\le$ 1% for precision $\ge$11 bits.

3) For the Gaussian: a) both errors fall off relatively smoothly as the precision goes up.

   b) the peak error is only about 3 times the rms error.

   c) peak error $\le$ 1% for precision $\ge$8 bits and rms error $\le$ 1% for precision $\ge$7 bits.

Figure 3 shows the variation of the errors with pulse width for a unit height pulse, a precision of 16 bits and an array size N=2048.

The data are given in Table 2.

Figure 4 consists of plots of errors as a function of width for four different functions. In addition to the pulse and the Gaussian, the functions investigated are:

1) 3 identical Gaussians uniformly spaced and added, i.e.,

$$f(x) = e^{-(x-s)^2/\sigma^2} + e^{-x^2/\sigma^2} + e^{-(x+s)^2/\sigma^2}$$

where s is the separation between the axes of adjacent Gaussians. The value of s has been taken to be a constant at 0.5. For small values of σ, the curves are essentially distinct but as σ increases (and so does the FWHM), the curves overlap to an increasing extent.

2) the sinc-squared function, i.e.,

$$f(x) = \left[\frac{\text{Sin } m\pi x}{m\pi x}\right]^2$$

where m = the frequency of the function. It has zeros at $x = \pm \frac{1}{m}, \pm \frac{2}{m}$, etc. and the number of maxima in the interval $[-1,1]$ is 2m-1. As the value of m increases, the peaks of the function become narrower and in order to be able to compare this function with the others, m has been plotted in decreasing order.

In all four plots, the function is narrowest at the left and broadest at the right. There is a general tendency for the errors to increase from left to right for all functions, but the variation is not always monotonic. Furthermore, the increase in error from narrowest to broadest is within a factor of 10 for all functions except for the peak error in the case of the three Gaussians. The data are given in Table 2.

Figure 5 displays essentially the same data as Figure 4 in a somewhat different manner in order to see easily how the errors depend on the type of function. The functions have been arranged in order such that the errors increase from one function to the next. It is

FIGURE 3

N = 2048 ERROR vs PULSE WIDTH 16 BITS

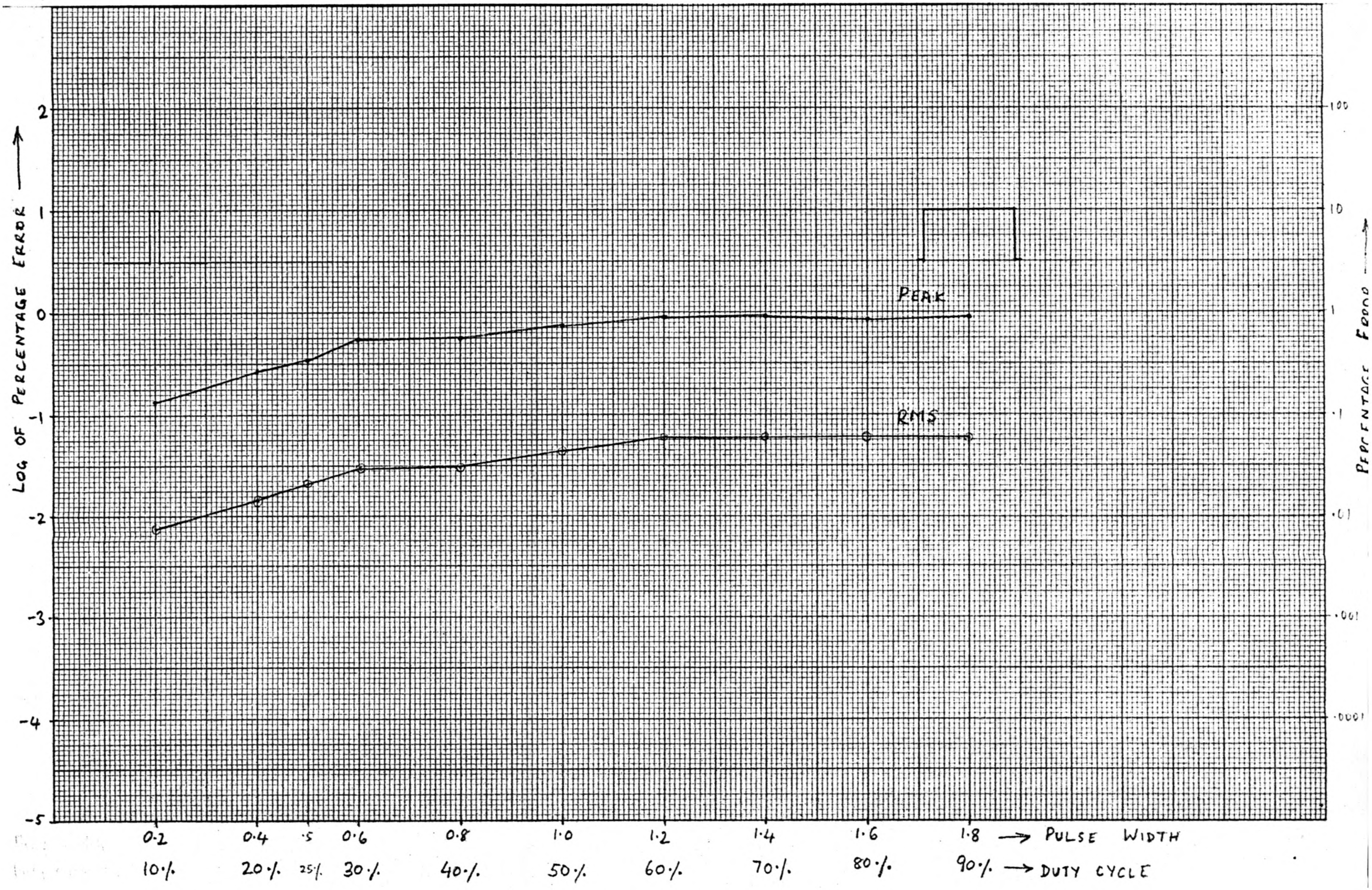


FIGURE 4

N = 2048 ERROR VS WIDTH FOR DIFFERENT FUNCTIONS 16 BITS

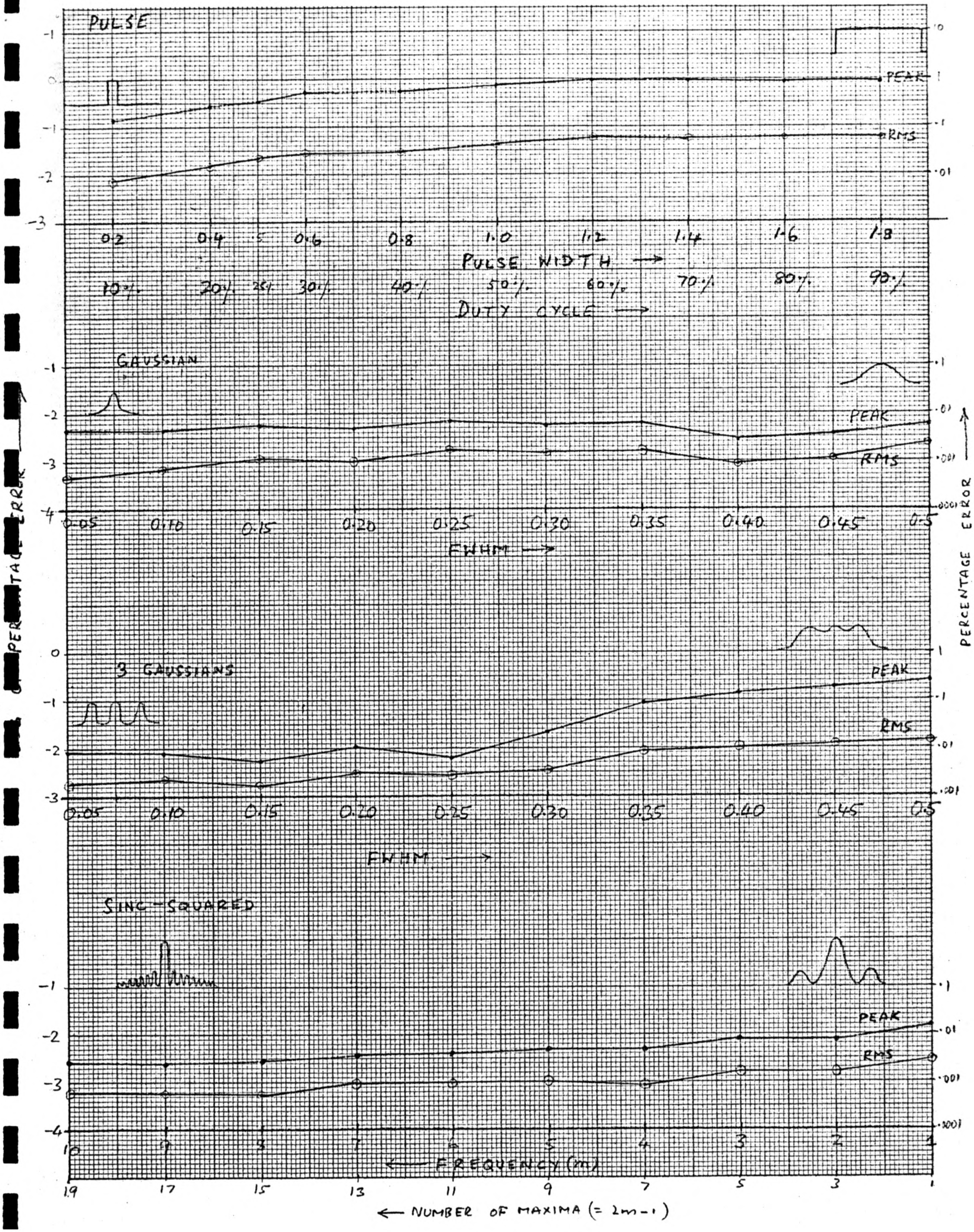
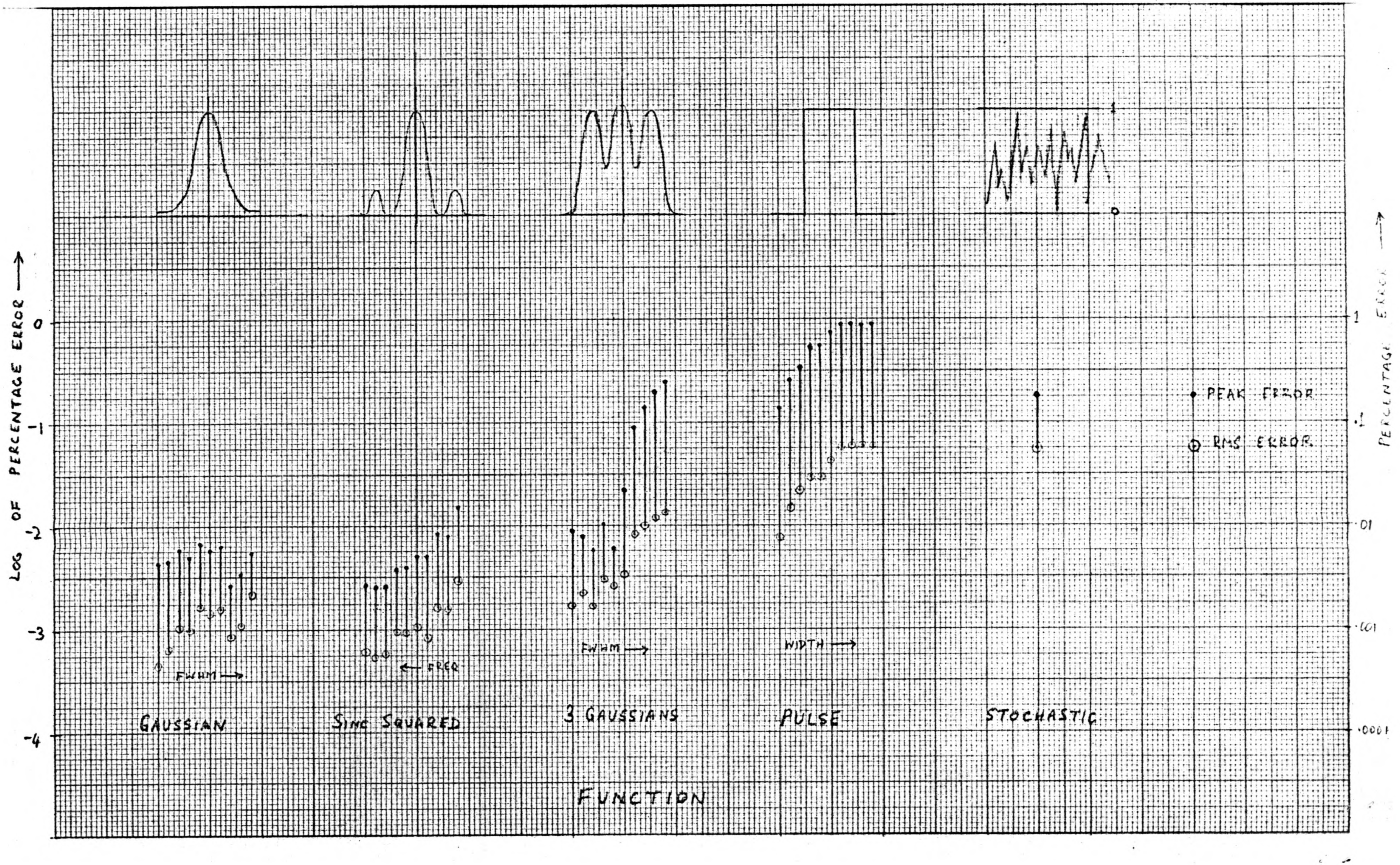


FIGURE 5

N = 2048 ERROR VS SHAPE 16 BITS



evident that the Gaussian is the least error-prone and the pulse the most error-prone. However, all errors are below the 1% level for 16 bits precision. The errors due to a stochastic function are also plotted in the same Figure. These were calculated by using a pseudo-random number generator from the FORTRAN library. This routine returns a random number between 0 and 1 each time it is called. This data is also given in Table 2.

The last figure, Figure 6, is a plot of the errors as a function of the array size N for 16 bits precision. Curves are plotted for a 25% duty-cycle pulse and a Gaussian with a FWHM of 0.5. In the case of the pulse, the error rises steadily as N increases, with the peak error rising somewhat more rapidly than the rms error. However, for the Gaussian, the errors change very little and in fact appear to decrease somewhat for some values of N. When interpreting this curve, it should be kept in mind that as N increases, the function is sampled more accurately since the interval of consideration remains the same. The data for this figure are given in Table 3.

FIGURE 6

16 BITS

ERROR VS ARRAY SIZE

PULSE & GAUSSIAN

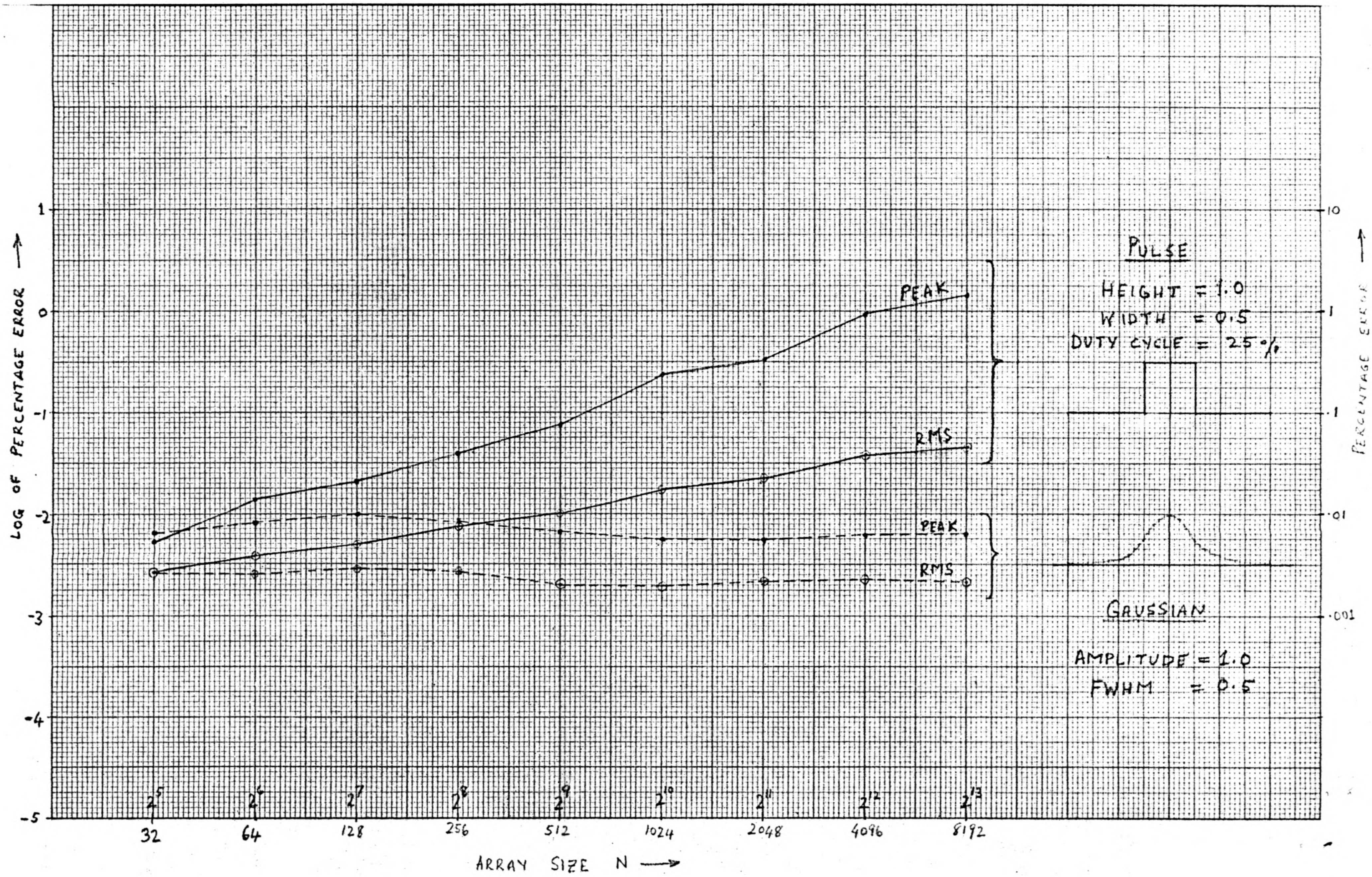


TABLE 1

N=2048 ERROR vs. BITS PULSE AND GAUSSIAN

No. of Bits	PULSE (25% DUTY CYCLE)		GAUSSIAN (FWHM=0.5)	
	% rms error	% peak error	% rms error	% peak error
20	.0013	.0194	.00015	.00062
19	.0026	.0387	.00034	.00107
18	.0053	.0793	.00063	.0020
17	.0122	.2413	.00104	.003
16	.0221	.3338	.00214	.005
15	.0519	1.084	.00436	.012
14	.0883	1.482	.0100	.029
13	.2059	4.448	.016	.048
12	.4624	10.44	.035	.103
11	.7245	13.38	.049	.136
10	1.662	37.24	.120	.314
9	2.312	50.13	.221	.596
8	3.104	50.14	.418	1.18
7	4.344	50.69	.833	2.35
6	6.251	50.69	1.708	4.69
5	8.570	50.69	2.903	7.81
4	11.05	52.90	4.692	12.5
3	17.24	57.32	9.30	25.0
2	24.06	57.32	20.58	50.0
1	43.30	75.0	34.3	75.0
0	50.0	100.0	43.3	100.0

TABLE 2

N=2048; ERROR vs. SHAPE FOR 5 FUNCTIONS; 16 BITS

PULSE

Duty Cycle	10%	20%	25%	30%	40%	50%	60%	70%	80%	90%
% rms error	.0073	.0143	.022	.029	.029	.043	.058	.058	.058	.057
% peak error	.133	.268	.334	.547	.552	.726	.899	.893	.861	.899

GAUSSIAN

FWHM	.05	.1	.15	.2	.25	.3	.35	.4	.45	.5
% rms error	.0004	.0006	.0010	.0010	.0016	.0014	.0016	.0008	.0010	.0021
% peak error	.0042	.0044	.0057	.0048	.0068	.0056	.0062	.0026	.0033	.0054

3 GAUSSIANS

FWHM	.05	.1	.15	.2	.25	.3	.35	.4	.45	.5
% rms error	.0016	.0022	.0017	.0030	.0026	.0033	.0082	.010	.012	.014
% peak error	.0088	.0077	.0057	.010	.006	.022	.087	.139	.194	.243

SINC -SQUARED

Freq.(m)	10	9	8	7	6	5	4	3	2	1
% rms error	.0006	.0005	.0006	.001	.001	.001	.0008	.0016	.0016	.0029
% peak error	.0027	.0025	.0026	.0037	.0039	.0048	.0048	.0084	.0077	.0153

STOCHASTIC

% rms error	0.0566
% peak error	0.1820

TABLE 3

ERROR vs. ARRAY SIZE; 16 BITS

ARRAY	PULSE		GAUSSIAN	
SIZE	(25% DUTY CYCLE)		(FWHM=0.5)	
N	% rms error	% peak error	% rms error	% peak error
32	.0026	.0053	.0023	.0061
64	.0037	.0137	.0025	.0080
128	.0048	.0204	.0029	.0100
256	.0073	.0397	.0027	.0079
512	.0103	.0745	.0020	.0068
1024	.0175	.2307	.0020	.0055
2048	.0221	.3338	.0021	.0054
4096	.0365	1.066	.0022	.0059
8192	.0443	1.429	.0021	.0059