Comparison of 1-D interpolation techniques to measure the gaussian pulse amplitude accurately using low sampling data for radiation detector readout

Arpit Patel, Himanshu Mazumdar

Abstract— Pulse amplitude measurement for the analog signal is now widely used in applications like spectroscopy, IR / laser receiver, RF signal receiver, etc. Accurate Amplitude measurement requires a high-speed analog to digital converter which consumes high power. In this paper we are aiming to detect analog pulse amplitude with low sampling ADC and interpolation method. The comparative study of seven interpolating methods namely Near interpolation, Linear interpolation, Cubic interpolation, LaGrange's interpolation, Newton Raphson interpolation, Whittaker Shannon interpolation and Neville's algorithm were conducted for measurement of analog pulse peak height. The interpolation method is selected based on the relative error, mean square error and mean absolute deviation. The performance of each method and the overall system is discussed in this paper.

Index Terms— gaussian pulse, charge sensitive pre-amplifier, interpolation, radiation detector, field-programmable gate array, pulse height measurement.

---

1 Introduction

Many applications require measurements using an analog-to-digital converter (ADC) to convert the analog input information in to the digital form. Such applications will have resolution requirements based on the dynamic range of the signal, the minimum measurable change in a parameter, and the signal-to-noise ratio (SNR). Many systems use a better resolution off-chip ADC as a result. However, the cost of the ADC increases with the level of desired accuracy. By creating hardware to quantize the analog signal amplitude into the digital signal with a longer code-word length, better ADC accuracy can be attained. Word lengths in practical ADCs are limited. Higher conversion accuracy is attained by calculating additional samples in order to efficiently strike a balance between system cost and accuracy. Analyzing the input signal is necessary in order to improve the required ADC resolution. Digital interpolation methods can be used to increase the resolution of digital data. Through an FPGA, the digital signal can be processed. This technique, which processes samples and increases the 12-bit ADC conversion's accuracy by extra bits, is examined in this study. As the work's application, peak height detection is used. The function generator's gaussian pulse is delivered to the ADC for sampling before the data is sent to the FPGA for interpolation. The interpolation will be implemented by FPGA, which will use the interpolated data to determine the maximum sample value. To obtain data from the FPGA and display it on the computer screen with the necessary graphs and numbers, a national instrument brand called NI DAQ is employed.

- *Arpit Patel is currently pursuing Ph. d program from Dharmsinh Desai University, Nadiad 387001, Gujarat, India. PH-+91 8460890415. E-mail: arpitp@prl.res.in*
- *Prof. Himanshu Mazumdar is head of The Research and development center at Dharmsinh Desai University, Nadiad 387001, Gujarat, India. E-mail: hsmazumdar@hotmail.com.*

2 Interpolation Methods

With the help of a few precise sample data points, interpolation can be used to estimate the unknown values of a nonlinear or linear function [1]. Theoretically, more data points are needed for better fitting accuracy. However, it is not practical to evaluate ideas based on big data sets due to the constraints of the difficulties of hardware computing. The initial objective of the problem at hand is to sample the input pulse into uniformly spaced data points. Any of the following interpolation techniques can be used to determine the analog pulse's unknown peak value

2.1 Nearest neighbor Interpolation

The simplest interpolation technique is nearest neighbor interpolation. The closest interpolation simply determines the closest data value at an integer position by rounding the expected position's value.

The value of the interpolated function f at point X is given for the samples F(k) that are given:

$$f(x) = F\left(\lfloor x + \tfrac{1}{2} \rfloor\right) \quad (1)$$

The closest sample point to x is simply chosen. The greatest integer number that is less than or equal to x in this case is called the floor function, or ⌊x⌋. This results in an interpolated function that looks like a staircase. The function f(x) is in reality defined on the entire real axis, but it is neither continuous (since it contains jumps) nor differentiable. Take a look at Figure-1's interpolated closest neighbor representation of the function f(x)=sin(x) as an example.

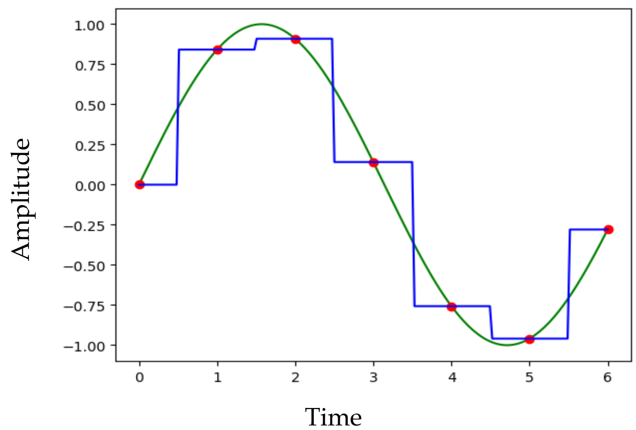


Fig. 1. The continuous function f(x) = sin(x) in green, the sampled function in red and the nearest interpolated function in blue.

2.2 Linear Interpolation

The simplest technique for determining the value of a function between any two known values is the linear interpolation formula. A technique for fitting curves with linear polynomials is the linear interpolation formula. In essence, the interpolation method uses the collection of values to discover new values for any function. The linear interpolation formula is used to determine the unknown values in the table. In this part, let's study more about the linear interpolation formula. Data forecasting, data prediction, mathematical and scientific applications, market research, etc. all employ the linear interpolation algorithm. The unknown values in the table can be found using the linear interpolation formula. The formula for linear interpolation formula is given by:

$$y = y_1 + (x - x_1)\frac{(y_2 - y_1)}{(x_2 - x_1)} \quad (2)$$

where,

- x_1 and y_1 are the first coordinates
- x_2 and y_2 are the second coordinates
- x is the point to perform the interpolation
- y is the interpolated value

2.3 Lagrange's interpolation

The Lagrange interpolation formula may be used to locate a polynomial known as a Lagrange polynomial that assumes certain values at every location. Lagrange's interpolation is an approximation to f using a N^{th} degree polynomial (x). Given n distinct real values $X_1, X_2, \ldots, X_n$ and n real values $Y_1, Y_2, \ldots, Y_n$ (not necessarily seperate), there is a unique polynomial P with real coefficients satisfying $P(X_i) = Y_i$ for $i \in \{1, 2, \ldots, n\}$, such that $\deg(P) < n$. The following is the Lagrange interpolation formula for a different order of polynomials:

$$y = f(x) = \frac{(x-x_1)(x-x_2)\ldots(x-x_n)}{(x_0-x_1)(x_0-x_2)\ldots(x_0-x_n)}y_0 + \frac{(x-x_0)(x-x_2)\ldots(x-x_n)}{(x_1-x_0)(x_1-x_2)\ldots(x_1-x_n)}y_1 + \cdots + \frac{(x-x_0)(x-x_1)\ldots(x-x_{n-1})}{(x_n-x_0)(x_n-x_1)\ldots(x_n-x_{n-1})}y_n \quad (3)$$

2.4 Newton Raphson Interpolation

The Newton-Raphson technique, commonly referred to as Newton's method, is a rapid approach to approximate the real-valued function f(x) = 0's root. It makes advantage of the idea that a straight line parallel to a continuous, differentiable function can serve as a rough approximation. A continuous, differentiable function f(x) needs to have a root, and you know the root you're seeking for is close to the point $x = x_0$. Then, according to Newton's approach, a more accurate approximation for the root is

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \quad (4)$$

To get the requisite precision, this procedure may be done as many as necessary. In general, for every x-value x_n, the next value is provided by below equation.

$$x_{(n+1)} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (5)$$

2.5 Cubic interpolation

Finding a curve that connects data points with a degree of three or less is possible through the use of cubic spline interpolation. The closest samples are utilized to determine the interpolated value in nearest neighbor and linear interpolation [1]. We examine two data points on the left and two on the right for cubic interpolation in order to increase the accuracy of the fitting. We fit a cubic polynomial over the range of x=k to x=k+1 to interpolate [3].

$k \le x \le k+1$:

$$f(x) = a(x)^3 + b(x)^2 + c(x) + d$$

to the 4-sample points k−1, k, k+1 and k+2. For the 4 points we have:

$$F(k-1) = a(-1)^3 + b(-1)^2 + c(-1) + d$$
$$F(k) = d$$
$$F(k+1) = a(1)^3 + b(1)^2 + c(1) + d$$
$$F(k+2) = a(2)^3 + b(2)^2 + c(2) + d \quad (6)$$

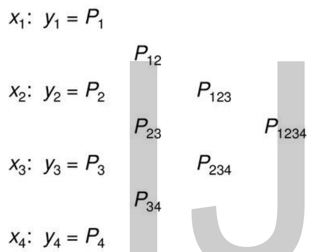
2.6 Whittaker–Shannon interpolation

One way to create a continuous-time bandlimited function from a list of real values is to use the Whittaker-Shannon interpolation formula, often known as sinc interpolation. Similar to the Lagrange polynomial interpolation, the procedure is used. The key to polynomial interpolation is to identify a collection of component polynomials, each of which traverses a certain set of provided points. The component polynomials must also equal zero whenever there is a different point in order to be able to combine them all into a single polynomial that traverses the whole collection of points. By doing this, it is ensured that the polynomials won't interfere with one another at the specified set of points when they are added together. Instead of using polynomials as the component, Whittaker-Shannon interpolation uses the Sinc function. The Sinc functions are periodic functions that are as below:

$$sinc(x) = \begin{cases} 1, & x = 0 \\ \frac{\sin(x)}{x}, & Otherwise \end{cases} \tag{7}$$

2.7 Neville's Algorithm for interpolation

An interpolation procedure called Neville's algorithm starts by fitting a polynomial of degree 0 across the point (x_k, y_k) for k=1,..., n, or $P_k(x)=y_k$. Then, in a second iteration, pairs of points are fitted using P_i and $P_{(i+1)}$, resulting in P_{12}, P_{23}, and so on. A "pyramid" of approximations is produced by repeating the process until the desired outcome is obtained.



The final result is

$$P_{i(i+1)....(i+m)} = \frac{(x-x_{i+m})P_{i(i+1)....(i+m-1)}}{(x_i-x_{i+m})} + \frac{(x_i-x)P_{(i+1)(i+2)....(i+m)}}{(x_i-x_{i+m})} \tag{8}$$

3 Implementation and Result

In the present work, All of the above-mentioned interpolation methods are simulated in this work, and hardware implementation is carried out using ADC, FPGA, and the LabVIEW environment. The initial finding from this research was that the precision of fitting rises as the number of interpolation points grows, reducing error. The 3-point interpolation method is used to compare all of the above interpolation methods. [6]. The 3-point interpolation method is used to compare all of the above interpolation methods. The values of MSE calculated for 1000 equally amplitude pulses for selected sample sizes of sections are shown in Table 1.

$$Relative\ error = \frac{Peak_{\ actual} - Peak_{\ Measured}}{Peak_{\ actual}} \tag{9}$$

$$MSE = \frac{\sum_1^n (Relative\ Error)^2}{n} \tag{10}$$

For comparing the efficiency of various interpolating methods based on fitting accuracy, same number of pieces were considered to provide a common platform for comparison. Nowadays FPGA's have found lots of applications in Instrumentation and Control Engineering. FPGA can perform basic mathematical operations of addition, subtraction, multiplication and division in real-time mode [7]. FPGA used in the present work uses fixed point operations and hence optimal bit size usage for each operation needs to be worked out in order to avoid any overflow leading to inaccurate measurement. In Present work National instrument (NI) USB 6343 and A3PE1500 (Microsemi make) FPGA is used to implement the methods. The FPGA is operated at 40 MHz clock frequency. The gaussian pulse is sampled using 12-bit ADC AD7492 at 1 MHz sampling rate. The Digital data is given to FPGA for interpolation. Data from the FPGA is given to NI module where LabVIEW code is written to find the maximum value from the sample data and the errors are estimated from the sample data. The compilation summary for various algorithms on FPGA is given in table-1 and discussed.

TABLE 1
MSE AND RESOURCE REQUIREMENT VALUES FOR VARIOUS INTERPOLATION METHODS

	Performance criteria		
		FPGA Resource utilization	
Method	**MSE**	**Number of Combinational cells (21504)**	**Number of sequencial cells (10752)**
Nearest	34.39E-06	15%	10%
Linear	34.39E-06	15%	14%
Lagrange's	2.34E-06	30%	28%
Newton Raphson	2.16E-06	40%	35%
Cubic	11.59E-06	20%	14%
Whittaker–Shannon	5.95E-06	32%	30%
Neville's Algorithm	6.47E-06	45%	30%

From the table -1 data, we can see that the MSE is less for the Lagrange's and Newton Raphson methods. In terms of FPGA utilization, Neville's Algorithm requires highest number of logic cells.

4 Conclusion

In this paper, A comparative study for the seven interpolating methods namely Nearest, Linear, Lagrange's, Newton Raphson, Cubic, Whittaker-Shannon, Neville's Algorithm was conducted for gaussian peak height measurement, performance comparison is carried out on the basis of the Mean square error and FPGA resource utilization. The Lagrange's and newton Raphson interpolation method has less MSE compared to other methods but the hardware utilization is more for newton Raphson method compared to Lagrange's method. It was also observed that the nearest interpolation method require lowest hardware as the function is less complex. For future work these interpolation techniques can be compared using another application.

Acknowledgment

This work is a part of the Ph.D. work of Arpit Patel. Director PRL, Head of Planetary Science Division, PRL, and Dean DDU, Nadiad are gratefully acknowledged for constant encouragement during the work.

References

[1] Bartels, R. H.; Beatty, J. C.; and Barsky, B. A. "Hermite and Cubic Spline Interpolation." *Ch. 3 in An Introduction to Splines for Use in Computer Graphics and Geometric Modelling*. San Francisco, CA: Morgan Kaufmann, pp. 9-17, 1998.
[2] Burden, R. L.; Faires, J. D.; and Reynolds, A. C. Numerical Analysis, 6th ed. Boston, *MA: Brooks/Cole,* pp. 120-121, 1997.
[3] Press, W. H.; Flannery, B. P.; Teukolsky, S. A.; and Vetterling, W. T. "Cubic Spline Interpolation." in *Numerical Recipes in FORTRAN: The Art of Scientific Computing,* 2nd ed. Cambridge, England: Cambridge University Press, pp. 107-110, 1992.
[4] M. J. Marsden, "Quadratic Spline Interpolation", *Bulletin of the American mathematical society*, vol. 80, no. 5, pp. 903-906, 1974.
[5] K.P.S. Rana, N. Mittra, N. Pramanik, P. Dwivedi and P. Mahajan, "A Virtual Instrumentation Approach to Neural Network-Based Thermistor Linearization on Field Programmable Gate Array", *Experimental Techniques*, vol. 39, no. 2, pp. 23-30, 2013.
[6] D. Sonowal and M. Bhuyan, "Linearizing Thermistor Characteristics by Piecewise Linear Interpolation in Real Time FPGA", *International Conference on Advances in Computing Communications and Informatics (ICACCI) IEEE*, pp. 1976-1980, 2013.
[7] Z. Nenova and T. Nenov, "Linearization circuit of the thermistor connection", *IEEE Transactions on Instrumentation and Measurement,* vol. 58, no. 2, pp. 441-449, 2009.
[8] James Tsui, "Digital Techniques for Wideband Receivers", *Artech House Inc.,* Norwood, Massachusetts, 1995.
[9] B. G. Quinn, "Estimating Frequency by Interpolation Using Fourier Coefficients," *IEEE Trans. Signal Processing,* Vol. 42, pp. 1264-1268, May 1994.
[10] P. Voglewede, "Parabola Approximation for Peak Determination" *Global DSP Magazine*, 3(5):13-17, May 2004
[11] M.A. Richards, "Fundamental of Radar Signal Processing", *New York; Hill*, 2005