ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 8, August 2017

Parallel Computation of Mathematical Functions in Fractional Calculus

[1] Ms. Priya Y. Khot, [2] Mrs. Shweta Ashtekar, [3] Mr. Vishwesh A. Vyawahare, [4] Mr. Mukesh D. Patil
[1][2][3] Department of Electronics Engineering,
[4] Department of Electronics and Telecommunication Engineering
[1][2][3][4] Ramrao Adik Institute of Technology

Abstract **- Generalization of classical calculus that is integral calculus to non-integer order is termed as Fractional calculus. Nowadays some special mathematical functions are having greater interest from the scientists of both mathematical and scientific background. This is because of their role as solutions of fractional order differential and integral equations, as the better mathematical models of phenomena of various physical, engineering, automatization, chemical, biological, Earth science, economics etc. nature. Principle objective of this work is to implement Special Functions of Fractional calculus with the help of Graphics Processing Unit (GPU) using MATLAB. GPU Computing makes use of CPU along with GPU for faster computation. This computation speed is achieved by offloading parallel portions of numerical algorithm to GPU, while simultaneously serial portions to be executed on CPU. Error Function and Complementary Error Function are implemented on MATLAB and execution speed for both CPU computing and GPU computing are compared for faster computation. From the results it is observed that around 30X speedup is achieved in the computation of special functions using GPU.**

Index Terms— Fractional calculus, Graphics Processing Unit, Special Functions, Parallel Computing Toolbox.

I.INTRODUCTION

Fractional Calculus is a domain of mathematical study that evolves from the traditional definitions of derivative operators and integral calculus, in similar manner fractional exponents is termed as an outgrowth of exponents with integer value. It is three centuries old as the conventional calculus. Fractional Calculus allows the description of a real object in a much more accurate way as compared to classical integer calculus [1] [3].

Fractional differential equations (FDEs) represent an important tool in technology, science and economics and engineering applications [2]. Complex problems in engineering are often treated with the help of important functions as their solution. These functions are defined by improper integral and series (or infinite products) and are generally called special functions. From application points of view, special functions as important mathematical tools rest on, due to their eminent properties, usefulness for the applied scientists and engineers [5]. Various special functions will provide solutions to integer order differential equations and systems.

Fractional calculus study have wide range of applications in different domains including power transmission units, automobiles, financial system design, image processing and various control system. In comparison to the longer history of CPU, GPU is a new and revolutionary device to accelerate computational performance. In this work, aim is to show how special functions can be computed in shorter duration of time using GPU. We will implement some SF of FC on both, a CPU as well as GPU and compare their execution speeds. Organization of paper is stated here. Section 2 presents the basic concepts of Error, Complementary Error function and Weber function in brief. Section 3 discusses the parallel computation of functions along with the hardware aspects. Results of functions implemented on CPU and GPU using MALAB and comparison of their execution times are given in section 4. Results are summarized and final conclusion is stated in section 5.

II. BASIC CONCEPTS OF MATHEMATICAL FUNCTIONS

A. Error Function

Error function is a non-elementary function, also called as Gauss error function. It has sigmoid shape which occurs in statistics, probability, and partial differential equations describing diffusion. It is defined as [11],

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \qquad (1)$$

Expanding the integrand of equation(1) as a power series of t and integrating each term, we obtain

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 8, August 2017

$$erf(x) = \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{n!(2n+1)} \quad (2)$$

which can also be expanded as

$$erf(x) = \frac{2}{\sqrt{\pi}} x e^{-x^2} \left(1 + \sum_{n=1}^{\infty} \prod_{k=1}^{n} \frac{x^2}{k+1/2}\right) \quad (3)$$

When $|x| \longrightarrow \infty$ and $|arg x| < 3\pi/4$, the error function can be expanded as,

$$erf(x) \sim 1 - \frac{e^{-x^2}}{\sqrt{\pi}x}\left[1 + \sum_{n=1}^{\infty}(-1)^n \frac{1 \cdot 3 \cdots (2n-1)}{(2x^2)^n}\right]$$

$$erf(x) = 1 - \frac{e^{-x^2}}{\sqrt{\pi}x}\left[1 + \sum_{n=1}^{\infty} \prod_{k=1}^{n} \left(-\frac{2k-1}{2x^2}\right)\right] \quad (4)$$

It is not difficult to see that

$$erf(0) = 0; \quad erf(\infty) = 1;$$

$$erf(-x) = -erf(x); \quad erf(\overline{x}) = \overline{erf(x)} \quad (5)$$

Error function does not have singularities excluding at infinity; it is an entire function. Taylor series of error function is always convergent. The defining integral of error function can be evaluated with the help of Maclaurin series of error function which can be obtained by expanding the integrand e^{-z^2} into its Maclaurin series and integrating term by term instead of evaluating it in closed form in terms of elementary functions.One obtains the error function's Maclaurin series is as follows [1]:

$$erf(x) = \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{n!(2n+1)} =$$

$$\frac{2}{\sqrt{\pi}}\left(x - \frac{x^3}{3} + \frac{x^5}{10} - \frac{x^7}{42} + \frac{x^9}{216} - \cdots\right) \quad (6)$$

When the results of a series of measurements are described by a normal distribution with standard deviation σ and expected value 0, then $erf(\frac{a}{\sigma\sqrt{\pi}})$ is the probability that the error of a single measurement lies between -a and +a, for positive a [4]. Error function can be used in digital communication system for determining the bit error rate. When the Heaviside step function is treated as boundary condition, error function can be used as a solution of the heat equation. The error function and its approximations can be used to estimate results that hold with high probability.

B. Complementary Error Function

The complementary error function (erfc) is defined as

$$erfc(x) = 1 - erf(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-t^2} dt = e^{-x^2} erfcx(x) \quad (7)$$

which also defines erfcx , the scaled complementary error function (which can be used instead of erfc to avoid arithmetic underflow [7] [8]). Craig's formula defines the another form of complementary error function for non-negative x as follows [6]:

$$erfc(x|x \geq 0) = \frac{2}{\pi} \int_0^{\frac{\pi}{2}} exp\left(-\frac{x^2}{\sin^2\theta}\right) d\theta \quad (8)$$

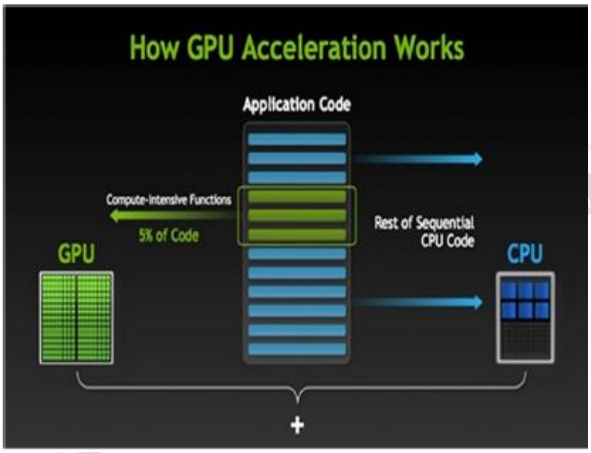

Fig. 1. GPU Computation.
[11]

III. PARALLEL COMPUTATION OF MATHEMATICAL FUNCTIONS

Parallel computation of functions is achieved with the help of Graphics Processing Unit. GPU-accelerated computing is the use of a graphics processing unit (GPU) together with a CPU to accelerate deep learning, analytics, and engineering applications. Acceleration is achieved by offloading parallel portions of the numerical algorithm to the GPU, while serial portions of the algorithm to the CPU [10].

A. Parallel Computing Toolbox:

In 2010, the feature of GPU computing was added into Matlab's parallel computing toolbox by a joint force of Mathworks and Nvidia. Build-in GPU enabled functions allow developers to take advantage on the powerful GPU computing simply by Matlab, a high-

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 8, August 2017

end programming language. MATLAB consists of many function toolboxes, and is very useful in solving system simulation and calculations. Computationally complicated and tasks which are much demanding can be resolved by using parallel computing toolbox's graphics processing unit, multicore structure and computer clusters. Paralle computation of MATLAB application is possible without using CUDA or MPI programming.

This computation can be simplified by the various attributes offered by toolbox like parallel for loops help in running the task simultaneously on multiple processors, CUDA-enabled NVIDIA GPUs supported by PCT, complete usage of multicore processors on the desktop, synergistic or section-wise execution of applications. All these are methods offered by PCT for parallel computation. Parallel Computing Toolbox lets you solve computationally and data-intensive problems using multicore processors, GPUs, and computer clusters. High-level constructs-parallel for-loops, special array types, and parallelized numerical Algorithms let you parallelize MATLAB applications without CUDA or MPI programming. The toolbox lets you use the full processing power of multicore desktops by executing applications on workers that run locally. Parallel computing is the execution of many processes simultaneously. The large problem is divided into smaller sections and solved in parallel.

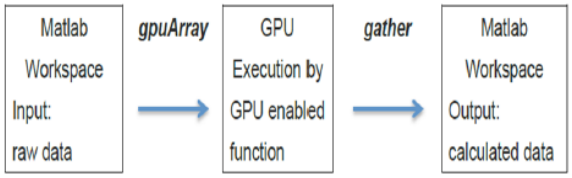

Fig. 2. GPU computing in Maltab.[9]

Benefits parallel computing are:

- Reduction in computational time and cost efficient: Multiple core structure in GPU computation of PCT will automatically reduces the execution time of optimized applications.
- Contemporaneous: Simultaneous execution of multiple tasks.
- Usage of non-local resources

The feature of GPU computing was added into Matlab's parallel computing toolbox by a joint force of Mathworks and Nvidia. Build-in GPU enabled functions allow developers to take advantage on the powerful GPU computing simply by Matlab, a high-end programming language. When Matlab's GPU enabled functions are executed on the GPU, data must be transferred from Matlab workspace to GPU device memory. The command gpuArray provides a specific array type for such data transfer, then GPU enabled Functions can run on these data. The command gather returns those calculated results, which are stored in GPU, back to Matlab workspace. The procedure of GPU computing in Maltab is as shown in figure (2):

The advantage of Matlab GPU programming is that users can easily utilize GPU computing by adding just few more commands to their original Matlab codes. Disadvantages include that only limited Matlab functions are GPU-enabled and the computing efficiency of Matlab GPU is less than those codes written in CUDA [2].

In this work, Leopard cluster is used for testing the results of functions. It has 2 Tesla K40 GPUs with 2880 cores and 2 Intel(R) Xeon(R) CPU E5-2620@ 2:00GHz with 24 cores 32 GB RAM and 30GB SWAP. Initially mathematical functions are implemented on CPU then on GPU and their execution speeds are compared for the speedup.

IV. RESULTS

Series approximation of error function and complementary error function represented using Maclaurin series as mentioned above are implemented on MATLAB. Optimized code in MATLAB is executed on GPU afterwards. After benchmarking, verified results for different parameters of both are tabulated in the table shown below. Two input parameters are given:

- 'z' is the array of input values of which error function is to be calculated
- 'N' is the number of upper limit of summation

A. Output plots obtained in MATLAB

1) Error Function:

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 8, August 2017

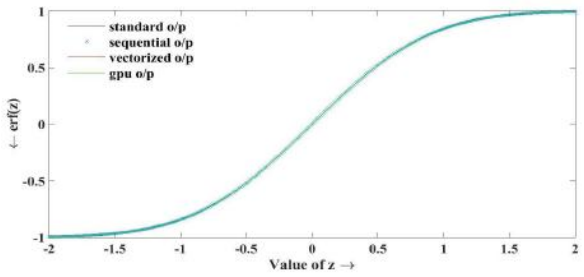

Fig. 3. Plot of Error Function in MATLAB.

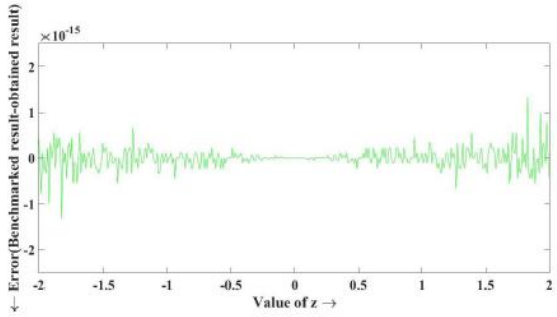

Fig. 4. Plot of error between benchmarked results and obtained results in MATLAB.

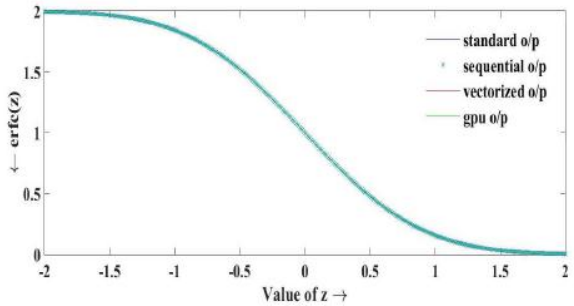

Fig. 5. Plot of Complementary Error Function in MATLAB.

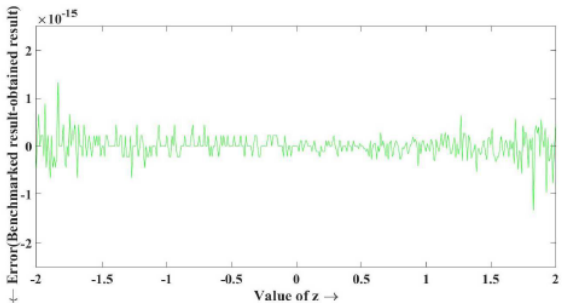

Fig. 6. Plot of error between benchmarked results and obtained results in MATLAB.

Figure(3) and (5) shows the outputs of error and complementary error functions respectively. Results obtained with sequential code, vectorized code, GPU code and benchmarked results are overlapped in these plots; they are perfectly matching. Outputs are benchmarked with the results obtained by MATLAB's inbuilt commands. Results for different input parameters are tabulated in the table I and II respectively. Speedup obtained is greater for large input samples. Highest speedup acheived here with GPU computation is around 27 times compared with CPU computation. Also we can observe from tables that error (MSE) obtained is very small.

Input parameters	CPU Computation Time (Tv sec)	GPU Computation Time (Tg sec)	Speedup (Tv/Tg)	Mean Square Error
z=-2:0.01:2; N=400	0.06	0.0093	6.5217	5.19E-32
z=-1:0.002:1; N=1000	0.041	0.009	4.5556	1.05E-32
z=-1:0.001:1; N=2000	0.13	0.011	11.8182	1.12E-32
z=-1:0.0004:1; N=5000	0.56	0.027	20.778	1.15E-32
z=-1:0.0002:1; N=10000	2.06	0.058	35.5173	1.09E-32

TABLE I
RESULTS OF ERROR FUNCTION.

Input parameters	CPU Computation Time (Tv sec)	GPU Computation Time (Tg sec)	Speedup (Tv/Tg)	Mean Square Error
z=-2:0.01:2; N=400	0.01	0.0073	2.1466	5.61E-32
z=-1:0.002:1; N=1000	0.04	0.009	4.7667	1.80E-32
z=-1:0.001:1; N=2000	0.12	0.0109	11.009	1.87E-32
z=-1:0.0004:1; N=5000	0.57	0.0218	26.514	1.90E-32
z=-1:0.0002:1; N=10000	2.08	0.05724	36.339	1.49E-32

TABLE II
RESULTS OF COMPLEMENTARY ERROR FUNCTION.

V. CONCLUSION

Mathematical functions in fractional calculus are implemented efficiently on MATLAB using Parallel Computing Toolbox (PCT). From the results it is observed that around 30X speedup is achieved in the computation of special functions using GPU. Work

offered us promising results with minimal errors. While testing codes it is observed that computation of complex functions leads to more speedup on GPU compared to CPU. On the hardware side, the execution speed offered by the GPU depends on its number of cores and Compute Capability. Higher is the number of these two parameters, logically higher will be the speedup offered by the GPU.

REFERENCES

[1] Monje, Concepcin A., et al. Fractional-order systems and controls: fundamentals and applications. Springer Science and Business Media, 2010.

[2] Cheng, Xiyou, and Chengkui Zhong. "Existence of positive solutions for a second-order ordinary differential system." Journal of mathematical analysis and applications 312.1 (2005): 14-23.

[3] Miller, Kenneth S., and Bertram Ross. "An introduction to the fractional calculus and fractional differential equations." (1993).

[4] "Error Function". En.Wikipedia.Org, 2017, https://en.wikipedia.org/wiki/Error-function.

[5] Lozier, Daniel W. "NIST digital library of mathematical functions." Annals of Mathematics and Artificial Intelligence 38.1 (2003): 105-119.

[6] Craig, John W. "A new, simple and exact result for calculating the probability of error for two-dimensional signal constellations." Military Communications Conference, 1991. MILCOM'91, Conference Record, Military Communications in a Changing World., IEEE. IEEE, 1991.

[7] Cody, William J. "Algorithm 715: SPECFUNa portable FORTRAN package of special function routines and test drivers." ACM Transactions on Mathematical Software (TOMS) 19.1 (1993): 22-30.

[8] Zaghloul, Mofreh R. "On the calculation of the Voigt line profile: a single proper integral with a damped sine integrand." Monthly Notices of the Royal Astronomical Society 375.3 (2007): 1043-1048.

[9] Chuan-Hsiang Han, Introduction to Matlab GPUA cceleration for Computational Finance. Pub. Date: December26,2013

[10] Abramowitz, Milton, and Irene A. Stegun. Handbook of mathematical functions: with formulas, graphs, and mathematical tables. Vol. 55. Courier Corporation, 1964.

[11] "NVIDIA On GPU Computing And The Difference Between Gpus And Cpus". Nvidia.Com, 2017, http://www.nvidia.com/object/what-isgpu-computing.html.

[12] Mathai, Arakaparampil M., and Hans J. Haubold. Special functions for applied scientists. Springer Science+ Business Media, 2008.