Polynomial Finite Difference Technique to Generalize Helmholtz Type Equations

Abdul Ghafoor Shaikh[1,*], Abdul Hannan Sheikh[2], Wajid Ali Shaikh[3], Asif Ali[4]

[1]Department of Basic Sciencecs & Related Studies, QUEST, Nawabsah, Pakistan
[2]Department of Mathematics & Statistics, Institute of Business Management, Karachi, Pakistan
[3]Department of Mathematics, QUEST, Nawabshah, Pakistan
[4]Department of Basic Sciencecs & Related Studies, MUET, Jamshoro, Pakistan
[*]Corresponding author: agshaikh@ quest.edu.pk

Abstract

The Helmholtz differential equation is often used for efficient and dynamic modeling of wave scattering real-world problems. The time harmonic wave scattering phenomena find applications in several scientific areas such as acoustic, electromagnetism, sensors, seismic, radar, and solar technology. Lately, it has been vital in modeling medical imaging problems; an emerging need of humans. Helmholtz type's differential equation(s) also arises from a physical phenomenon, therefore, it is important to solve Helmholtz types of equation(s) for several purposes. Analytically, it is difficult, in some cases impossible, to find the solution of such equation(s). The numerical method can be used to find the solutions of such equations. Therefore, for the numerical method, the finite difference method is simple and is widely used as compared to other methods. The second order central finite difference is usually used in solving the differential equation(s), but sometimes in many scientific applications we do desire to have higher order approximations and that desire stems from two issues first the better accuracy. Suppose we have done computations of a problem with moderate size mesh, but if the error that we are getting is not acceptable for some unknown reasons, at the same time it is difficult to solve it with a further finer mesh that can happen at times and that is where we want a higher order approximation which gives us the same accuracy as a finer mesh but with fewer mesh points. This is the basic idea and memory storage is another important issue. The discretization of the space differential in the finite difference method is usually derived using the Taylor series expansion; however, if we use a method that adopts algebraic polynomial interpolations in the calculation around near-wall elements, all the calculations over irregular domains reduce to those over regular domains. However, if we use the polynomial interpolation systematically, exceptional advantages are gained in deriving high-order differences which is explained in detail in this paper with examples.

Keywords—Helmholtz Equation, Finite Difference, Acoustic, Wave Scattering, Discretization.

1 Introduction

The finite difference approximations [1-2] is a tool to solve hard differential equation(s), or Helmholtz types of a differential equation(s) [3-7]. The value of unknown can be found anywhere from the solution of the continuous function. These functions contain an infinite amount of information that we cannot store because of memory restrictions. We have to break space up into small cells/nodes, but even within one of those cells there is still an infinite amount of information. The infinite difference where we try to find the information at prescribed locations is called cell/node. This is like an experiment where we fix stick probes in certain locations and measure the desired measurement at those prescribed locations rather than anywhere within that body. The information is hidden in the nodes/cells which is important for scientific purposes and is to be determined having a digital computer to solve the problems. Therefore, we must convert the derivative into an algebraic equation which the goal of the finite difference method. The objective of finite difference is to express the derivative in terms of the nodal values so that we can treat those as unknowns and solve for them. In the basic philoso-

ISSN: *2523-0379 (Online), ISSN: 1605-8607 (Print)*
DOI: https://doi.org/10.52584/QRJ.1902.03

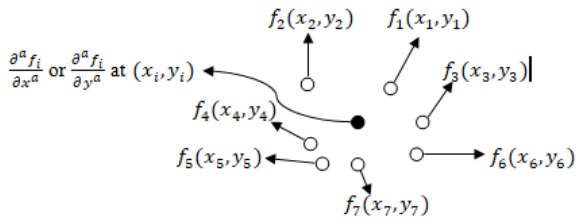

Fig. 1: Collecting information from scattering points.

phy of the finite difference, we replace the governing differential equation directly [8]. In other words, the differential equation could be replaced by an algebraic equation that is the most important feature of the finite difference approximations that makes the finite difference method simpler. We can easily write different expressions from the derivatives. The finite difference method is a way of obtaining a numerical solution of differential equations [9-10]. One of the disadvantages of finite difference approximation is that we cannot generalize finite difference approximations, but we have to determine finite difference approximations. There are several methods for deriving finite difference approximations. In in this paper, we present a polynomial technique show that this technique is simple to build the finite difference approximations. It builds on curve fitting polynomials to data and using MatLab codes to make it simple for finite difference approximations, presented with examples.

1.1 Polynomial Techniques for Finite Difference Approximations

The finite difference approximations do imply that we do not have a continuous smooth function stored in memory. We only know the function at discrete points [11-12]. The polynomial technique [9][13-16] is a lot more powerful to address the finite differences and presented with few examples. Let us consider a few random scattering points [17] which do not have to be distributed uniformly or anything just some random scattering of points in space. We want to generalize the concept of finite difference approximation to know what information is present in these points. Besides these points, where each point has x and y two dimensions, there will be a function $f_1, f_2, ..., f_7$ in which we have these random scattering of points. Now suppose that we want to estimate the function or one of the derivatives at some point. Amongst these open circle points, the close circle point is where we want to estimate the derivative of the open circle points. This is the information we have from which to derive the function or one of its derivatives at the close circle point [18]. It turns out that there is a way to do this. We always have a very similar form whether we interpolate the function or any of its derivatives. It is always just some weighted sum of the function values at each of those open circle points which is known as given follows,

$$\frac{\partial^a f_i}{\partial x^a} \; or \; \frac{\partial^a f_i}{\partial y^a}$$

$$\frac{\partial^a f_i}{\partial x^a} = a_1 f_1 + ... + a_7 f_7$$

where $a_1, a_2..., a_7$ are the finite difference coefficients. If we have a random scattering of points, we show these as an open circle, and we are interested to evaluate function or derivative at the close circle at right middle. Let's suppose we derive our finite difference coefficients and we take the same distribution of points but offset them. We will get the same distribution of the points at their relative position where open and close circle are the same. It turns out the same finite difference coefficients which means we are allowed to offset or shift coordinates and still get same finite difference coefficients which will be hugely useful in evaluating finite difference approximations. Again, if we take the same scattering of open circle points, but move the closed circle point, we can evaluate our function or one of its derivatives at a different point. It turns out completely finite difference coefficients of the different problem, but if we evaluate our finite difference at the same point but the points from which we are evaluating that change their position, so it turns out we get different coefficients in the end. Therefore, we only allow to shift coordinates and still get the same finite difference coefficients. If we do anything else, we have to redefine new finite difference coefficients.

1.2 Polynomial Technique for Deriving Finite Difference Approximations

The curve fitting to polynomials is known as polynomial technique for deriving the finite difference approximations. This technique is very simple. The general concept of how can we do this is as follows. Suppose we have a fit a polynomial to some set of points. To fit this n^{th} order polynomial, we needed $N + 1$ points,

such that;

$$\begin{aligned}
f(x) &= a_0 + a_1x + a_2x^2 + ... + a_Nx^N \\
f'(x) &= a_1 + 2a_2x + 3a_3x_2 + ... + Na_Nx^{N-1} \\
f''(x) &= 2a_2 + 6a_3x + 12a_4x^2 + ... + N(N-1)a_Nx^{N-2} \\
&\vdots \\
f(0) &= a_0 \\
f'(0) &= a_1 \\
f''(0) &= a_2 \\
f'''(0) &= a_3 \\
&\vdots
\end{aligned}$$

The function of any of its derivatives becomes very simple. We will shift our coordinates such that $x = 0$ is the point where we want to evaluate our function or finite difference. Suddenly, this polynomial is reduced to this very simple form, so we are using that property where we can shift points from which we are evaluating finite difference and it does not change the finite difference coefficients. We can reuse that anywhere no matter where those points are, so it is a very useful property.

1.3 Shifted Co-ordinates Technique for Finite Difference Approximations

Suppose we wish to evaluate $f(x)$ or one of its derivatives at the general point $x_n = x_{fd}$. To do this, we shift the x-axis by x_{fd} before fitting the polynomial. Recall that the finite difference coefficients depend only on the relative position of the points. An offset will not affect their values. Now we can rewrite our polynomial in terms of shifted coordinates.

$$\begin{aligned}
f(\tilde{x_1}) &= a_0 + a_1\tilde{x_1} + a_2{\tilde{x_1}}^2 + ... + a_N{\tilde{x_1}}^N \\
f(\tilde{x_2}) &= a_0 + a_1\tilde{x_2} + a_2{\tilde{x_2}}^2 + ... + a_N{\tilde{x_2}}^N \\
&\vdots \\
f(\tilde{x}_{N-1}) &= a_0 + a_1\tilde{x}_{N-1} + a_2\tilde{x}^2_{N-1} + ... + a_N\tilde{x}^N_{N-1}
\end{aligned}$$

It is very simple and will be presented with some examples. Let us take the x-coordinates of the points from which we approximate a derivative and store these x-coordinates in a column.

$$[x] = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N+1} \end{bmatrix}$$

Shifted function across the x-axis $\tilde{x} = 0$ corresponds to the point. We approximate the derivative as follows.

$$\frac{d^a}{dx^a} f(x = x_{fd} = fracd^a dx^a f(\tilde{x} = 0)$$

subtract x_{fd} from the column vector x to shift coordinates.

$$[\tilde{x}] = [x] - x_{fd} = \begin{bmatrix} x_1 - x_{fd} \\ x_2 - x_{fd} \\ \vdots \\ x_{N+1} - x_{fd} \end{bmatrix} = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \vdots \\ \tilde{x}_{N+1} \end{bmatrix}$$

Using the column vector $[\tilde{x}]$ to build the matrix $\left[\tilde{X}\right]$,

$$\left[\tilde{X}\right] = \left[\left[\tilde{X}\right]^0 \left[\tilde{X}\right]^1 \left[\tilde{X}\right]^2 \left[\tilde{X}\right]^3 \dots \left[\tilde{X}\right]^N\right]$$

Now we have a Vanderrmonde matrix where the first column is all ones, the second column is our values of x^2 and up to some value, which give a big square matrix.

$$\begin{bmatrix} 1 & \tilde{x}_1 & ... & \tilde{x}_1^N \\ 1 & \tilde{x}_2 & ... & \tilde{x}_2^N \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \tilde{x}_{N+1} & ... & \tilde{x}_{N+1}^N \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 - x_{fd} & ... & (x_1 - x_{fd})^N \\ 1 & x_2 - x_{fd} & ... & (x_1 - x_{fd})^N \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{N+1} - x_{fd} & ... & (x_{N+1} - x_{fd})^N \end{bmatrix}$$

Once we have the Vanderrmonde matrix $\left[\tilde{X}\right]$, its inverse will be

$$\left[\tilde{Y}\right] = \left[\tilde{X}\right]^{-1} = \begin{bmatrix} \tilde{y}_{1,1} & \tilde{y}_{1,2} & ... & (\tilde{y}_{1,N+1})^N \\ \tilde{y}_{2,1} & \tilde{y}_{2,2} & ... & (\tilde{y}_{2,N+1})^N \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{y}_{N+1,1} & \tilde{y}_{N+1,2} & ... & (\tilde{y}_{N+1,N+1})^N \end{bmatrix}$$

Once we have a $\left[\tilde{Y}\right]$ matrix, solve now polynomial equation, calculate the polynomial coefficients.

$$[a] = \left[\tilde{X}\right]^{-1} [f] = \left[\tilde{Y}\right] [f]$$

$$\begin{bmatrix} a_1 \\ a_2 \\ a_{N+1} \end{bmatrix} = \begin{bmatrix} \tilde{y}_{1,1} & \tilde{y}_{1,2} & ... & (\tilde{y}_{1,N+1})^N \\ \tilde{y}_{2,1} & \tilde{y}_{2,2} & ... & (\tilde{y}_{2,N+1})^N \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{y}_{N+1,1} & \tilde{y}_{N+1,2} & ... & (\tilde{y}_{N+1,N+1})^N \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_{N+1} \end{bmatrix}$$

$$\begin{aligned}
a_1 &= \tilde{y}_{1,1}f_1 + \tilde{y}_{1,2}f_2 + \tilde{y}_{1,3}f_3 + ... + \tilde{y}_{1,N+1}f_{N+1} \\
a_2 &= \tilde{y}_{2,1}f_1 + \tilde{y}_{3,2}f_2 + \tilde{y}_{3,3}f_3 + ... + \tilde{y}_{3,N+1}f_{N+1} \\
&\vdots \\
a_{N+1} &= \tilde{y}_{N+1,1}f_1 + \tilde{y}_{N+1,2}f_2 + \tilde{y}_{N+1,3}f_3 + ... \\
&\quad + \tilde{y}_{N+1,N+1}f_{N+1}
\end{aligned}$$

Recall how we interpolate the function or one of its derivatives to give the polynomial,

$$\begin{aligned}
f(\tilde{x}=0) &= a_0 \because a_0 = \tilde{y}_{1,1}f_1 + \tilde{y}_{1,2}f_2 + ... + \tilde{y}_{1,N+1}f_{N+1} \\
f'(\tilde{x}=0) &= a_1 \because a_1 = \tilde{y}_{2,1}f_1 + \tilde{y}_{2,2}f_2 + ... + \tilde{y}_{2,N+1}f_{N+1} \\
f''(\tilde{x}=0) &= a_2 \because a_2 = \tilde{y}_{3,1}f_1 + \tilde{y}_{3,2}f_2 + ... + \tilde{y}_{3,N+1}f_{N+1} \\
&\vdots
\end{aligned}$$

We can recognize that the rows of Y matrix are finite difference coefficients. We derive real finite difference approximations with the help of examples using the polynomial technique.

Example#1: Suppose we have three points distributed uniformly, and we would like to calculate a finite difference approximation first order and second order derivatives at the midpoint. Let we take shifted coordinates. We have a distribution of three points, we want them, $\tilde{X} = 0$ where we evaluate the finite difference, in this case, is the midpoint so that $\tilde{x} = 0$ value for midpoint is zero.

$$[\tilde{x}] = \begin{bmatrix} -h \\ 0 \\ h \end{bmatrix} = \begin{bmatrix} [\tilde{x}]^0 & [\tilde{x}]^1 & [\tilde{x}]^2 \end{bmatrix} = \begin{bmatrix} 1 & -h & h^2 \\ 1 & 0 & 0 \\ 1 & h & h^2 \end{bmatrix}$$

Remember that the rows of this Y matrix are essentially the finite difference coefficients.

$$\begin{aligned}
\begin{bmatrix} \tilde{Y} \end{bmatrix} &= \begin{bmatrix} \tilde{X} \end{bmatrix}^{-1} \begin{bmatrix} 1 & 1 & 0 \\ \frac{-1}{2h} & 0 & \frac{1}{2h} \\ \frac{1}{2h^2} & \frac{-1}{h^2} & \frac{1}{2h^2} \end{bmatrix} \\
\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ \frac{-1}{2h} & 0 & \frac{1}{2h} \\ \frac{1}{2h^2} & \frac{-1}{h^2} & \frac{1}{2h^2} \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_3 \end{bmatrix} \\
f(x_{fd}) \cong a_0 = f_1,\ \frac{df(x_{fd}}{dx} &\cong a_1 = f_2 = \frac{f_3 - f_1}{2h} \\
\frac{d^2 f(x_{fd})}{dx^2} &\cong a_2 = f_3 = \frac{f_1 - 2f_2 + f_3}{2h^2}
\end{aligned}$$

These are our finite difference approximations.

Example#2: Calculate a finite difference approximation of the same points, but now on the first point, we want to see how it behaves at the first point. Let us take shifted coordinates.

$$\begin{aligned}
[\tilde{x}] = \begin{bmatrix} 0 \\ h \\ 2h \end{bmatrix} &= \begin{bmatrix} [\tilde{x}]^0 & [\tilde{x}]^1 & [\tilde{x}]^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & h & h \\ 1 & 2h & 2h^2 \end{bmatrix} \\
\begin{bmatrix} a_0 \\ a_1 \\ a_3 \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ \frac{-1}{2h} & 0 & \frac{1}{2h} \\ \frac{1}{2h^2} & \frac{-1}{h^2} & \frac{1}{2h^2} \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix} \\
f(x_{fd}) \cong a_0 = f_1,\ \frac{df(x_{fd}}{dx} &\cong a_1 = f_2 \\
&= \frac{1.5f_1 - f_2 - 0.5f_3}{h} \\
\frac{d^2 f(x_{fd})}{dx^2} &\cong a_2 = f_3 = \frac{f_1 - 2f_2 + f_3}{2h^2}
\end{aligned}$$

Example # 3: Evaluate derivatives at the midpoints of four points, so our column vector of the x positions don't have a zero in it,

$$\begin{aligned}
[x] &= \begin{bmatrix} \frac{-3h}{2} & \frac{-h}{2} & \frac{h}{2} & \frac{3h}{2} \end{bmatrix} \\
[\tilde{x}] &= \begin{bmatrix} \frac{-3h}{2} \\ \frac{-h}{2} \\ \frac{h}{2} \\ \frac{3h}{2} \end{bmatrix} \\
&= \begin{bmatrix} [\tilde{x}]^0 & [\tilde{x}]^1 & [\tilde{x}]^2 & [\tilde{x}]^3 \end{bmatrix} = \begin{bmatrix} 1 & \frac{-3h}{2} & \frac{9h^2}{4} & \frac{-27h^3}{8} \\ 1 & \frac{-3h}{2} & \frac{9h^2}{4} & \frac{h^3}{8} \\ 1 & \frac{-3h}{2} & \frac{9h^2}{4} & \frac{h^3}{8} \\ 1 & \frac{-3h}{2} & \frac{9h^2}{4} & \frac{27h^3}{8} \end{bmatrix} \\
\begin{bmatrix} \tilde{Y} \end{bmatrix} &= \begin{bmatrix} \tilde{X} \end{bmatrix}^{-1} = \begin{bmatrix} \frac{-1}{16} & \frac{9}{16} & \frac{9}{16} & \frac{-1}{16} \\ \frac{1}{24h} & \frac{-9}{8h} & \frac{9}{8h} & \frac{-1}{24h} \\ \frac{1}{4h^2} & \frac{-1}{4h^2} & \frac{-1}{4h^2} & \frac{1}{4h^2} \\ \frac{-1}{6h^3} & \frac{1}{2h^3} & \frac{-1}{2h^3} & \frac{1}{6h^3} \end{bmatrix} \\
\Rightarrow \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} &= \begin{bmatrix} \frac{-1}{16} & \frac{9}{16} & \frac{9}{16} & \frac{-1}{16} \\ \frac{1}{24h} & \frac{-9}{8h} & \frac{9}{8h} & \frac{-1}{24h} \\ \frac{1}{4h^2} & \frac{-1}{4h^2} & \frac{-1}{4h^2} & \frac{1}{4h^2} \\ \frac{-1}{6h^3} & \frac{1}{2h^3} & \frac{-1}{2h^3} & \frac{1}{6h^3} \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix}
\end{aligned}$$

Solving the system, we get,

$$\begin{aligned}
f(x_{25}) \cong a_0 &= \frac{-f_1 + 9f_2 + 9f_3 - f_4}{16} \\
\frac{d}{dx} f(x_{2.5}) \cong a_1 &= \frac{f_1 - 27f_2 + 27f_3 - f_4}{24h}, \\
\frac{d^2}{dx^2} f(x_{2.5}) \cong a_2 &= \frac{f_1 - f_2 - f_3 + f_4}{2h^2}
\end{aligned}$$

These finite difference show that their evaluation is difficult.

1.4 Implementing the Polynomial Technique Using MatLab

Since for small matrices we have been using, the above technique, we do not need to use MatLab, but the reason MatLab is good to use is when matrix size

increases or maybe we do not have a uniform spacing grid [11], and we need to evaluate a completely different finite difference approximation for every point. We have so far derived finite difference approximations symbolically, but if we want 4^{th} order accurate finite differences. Recall our matrix equation representing polynomial written at each discrete point. It always had the following form where w's were just numerical constants. The h was symbolic.

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ & h & & & \\ & & h^2 & & \\ & & & \ddots & \\ & & & & h^N \end{bmatrix}^{-1}$$
$$\begin{bmatrix} 1 & w_{12} & w_{13} & \dots & w_{1N} \\ 1 & w_{22} & w_{23} & \dots & w_{2N} \\ 1 & w_{32} & w_{33} & \dots & w_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & w_{N+1,2} & w_{N+1,3} & \dots & w_{N+1,N} \end{bmatrix}^{-1} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix}$$

The key aspect here is that $[w]$ will be completely numerical, so it is easily inverted using MatLab. It is also important both separately inverted. This accommodates large matrices [18] and avoids symbolic manipulation, and letting $[v] = [w]^{-1}$.

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{N+1} \end{bmatrix} = \begin{bmatrix} 1 & 1.v_{12} & 1.v_{13} & \dots & 1.v_{1N} \\ 1 & \frac{1}{h}v_{22} & \frac{1}{h}v_{23} & \dots & \frac{1}{h}v_{2N} \\ 1 & \frac{1}{h^2}v_{32} & \frac{1}{h^2}v_{33} & \dots & \frac{1}{h^2}v_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \frac{1}{h^2}v_{N+1,2} & \frac{1}{h^2}v_{N+1,3} & \dots & \frac{1}{h^2}v_{N+1,N} \end{bmatrix} f$$

$$where\ f = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N+1} \end{bmatrix}$$

2 Finite Difference Approximations

We will write finite difference approximations of the polynomial coefficients.

$$f = a_0 = v_{11}f_1 + v_{12}f_2 + v_{13}f_3 + \ldots + v1NfN+!$$
$$\frac{df}{dx} = a_1 = \frac{v_{21}f_1 + v_{22}f_2 + v_{23}f_3 + \ldots + v_{2N}f_{N+1}}{h}$$
$$\frac{d^2f}{dx^2} = 2a_2 = \frac{v_{31}f_1 + v_{32}f_2 + v_{33}f_3 + \ldots + v_{3N}f_{N+1}}{h^2}$$

MatLab Example# 1: Here we need five points to calculate five polynomial coefficients,

$$[\tilde{x}] = \begin{bmatrix} -2h & -h & 0 & h & 2h \end{bmatrix}^T$$

Build $[w]$ matrix and $h = 1$,

$$[\tilde{x}] = \begin{bmatrix} -2 & -1 & 0 & 1 & 2 \end{bmatrix}^T$$
$$[W] = \begin{bmatrix} [\tilde{x}]^0 & [\tilde{x}]^1 & [\tilde{x}]^2 & [\tilde{x}]^3 & [\tilde{x}]^4 \end{bmatrix}$$
$$[W] = \begin{bmatrix} 1 & -2 & 4 & -8 & 16 \\ 1 & -1 & -1 & -1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 16 \end{bmatrix}$$
$$[V] = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ -0.0417 & -0.6667 & -1.2500 & 0.6667 & -0.0833 \\ -0.0417 & 0.6667 & -1.2500 & 0.6667 & -0.0417 \\ -0.0833 & 0.1667 & 0 & -0.1667 & 0.0833 \\ 0.0417 & -0.6667 & 0.2500 & -0.6667 & 0.0417 \end{bmatrix}$$

Here we write the finite difference approximations, to incorporate the symbolic h's back in.

$$f \cong a_0 = \frac{0.f_1 + 0.f_2 + 1.f_3 + 0.f_4 + 0.f_5}{1}$$
$$\frac{\partial}{\partial x}f \cong a_1 = \frac{0.0833f_1 - 0.6667f_2 + 0.f_3 - 0.6667f_4 - 0.0833f_5}{h}$$
$$\frac{\partial^2}{\partial x^2}f \cong a_2 = \frac{-0.042f_1 + 0.67f_2 - 1.25f_3 + 0.168f_4 + 0.042f_5}{h^2}$$
$$\vdots$$

MatLab Example# 2: Five points and five polynomial coefficients, i.e. build $[W]$ matrix, and $h = 0.5$.

$$[\tilde{x}] = \begin{bmatrix} -2h & -h & 0 & h & 2h \end{bmatrix} \Rightarrow [\tilde{x}]$$
$$= \begin{bmatrix} -1 & -0.5 & 0 & 0.5 & 1 \end{bmatrix}^T$$
$$[W] = \begin{bmatrix} [\tilde{x}]^0 & [\tilde{x}]^1 & [\tilde{x}]^2 & [\tilde{x}]^3 & [\tilde{x}]^4 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & -1 & 1 & -1 & 1 \\ 1 & -0.5 & -0.25 & -0.1250 & 0.0625 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0.5 & 0.25 & 0.1250 & 0.0625 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$
$$[V] = [W]^{-1}$$
$$= \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1.1667 & -1.3333 & 0 & -0.1667 & 1.3333 \\ -0.1667 & 2.6667 & -5,000 & -0.1667 & 2.6667 \\ -0.6667 & 1.3333 & 0 & 0.6667 & -1.3333 \\ 0.6667 & -2.6667 & 4.0000 & 0.6667 & -2.6667 \end{bmatrix}$$

Here we write the finite difference approximations, to incorporate the symbolic h's back in.

$$\begin{aligned} f &\cong a_0 = \frac{0.f_1 + 0.f_2 + 1.f_3 + 0.f_4 + 0.f_5}{1} \\ \frac{\partial}{\partial x} &\cong a_1 \\ &= \frac{1.167f_1 - 1.333f_2 + 0.f_3 - 1.67f_4 + 1.333f_5}{h} \\ \frac{\partial^2}{\partial x^2} &\cong 2a_2 \\ &= \frac{-0.167f_1 - 2.667f_2 + 5f_3 - 0.167f_4 + 2.667f_5}{h^2} \\ &\vdots \end{aligned}$$

MatLab Example# 3 Here, we need five points to calcu-late five polynomial coefficients,

$$\begin{bmatrix} \tilde{x} \end{bmatrix} = \begin{bmatrix} -2h & -h & 0 & h & 2h \end{bmatrix}^T$$

Build $\begin{bmatrix} w \end{bmatrix}$ matrix and $h = 0.25$,

$$\begin{bmatrix} \tilde{x} \end{bmatrix} = \begin{bmatrix} -0.5 & -0.25 & 0 & 0.5 & 0.25 \end{bmatrix}^T$$

$$\begin{bmatrix} V \end{bmatrix} = \begin{bmatrix} W \end{bmatrix}^{-1} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1.67 & -1.33 & 0 & -0.17 & 1.33 \\ -0.17 & 2.67 & -5.00 & -0.17 & 2.67 \\ -0.67 & 1.33 & 0 & 0.67 & -1.33 \\ 0.67 & -2.67 & 4.00 & 0.67 & -2.67 \end{bmatrix}$$

Here we write the finite difference approximations to incorporate the symbolic h back in.

$$\begin{aligned} f &\cong a_0 = \frac{0.f_1 + 0.f_2 + 1.f_3 + 0.f_4 + 0.f_5}{1} \\ \frac{\partial}{\partial x} &\cong a_1 \\ &= \frac{1.6667f_1 - 1.3333f_2 + 0.f_3 - 1.666f_4 + 1.3333f_5}{h} \\ \frac{\partial^2}{\partial x^2} &\cong 2a_2 \\ &= \frac{-0.167f_1 + 2.667f_2 + 5.00f_3 - 0.167f_4 + 2.667f_5}{h^2} \\ &\vdots \end{aligned}$$

3 Conclusion

The finite difference method by polynomial technique is used and is preferred on other methods because of its simplicity, which is built on curve fitting and using MatLab codes to make it simpler for finite difference approximations. In this paper, analysis of finite difference approximation is presented by polynomial technique with some examples, which showed that the polynomial technique is simpler as compared to other methods.

References

[1] Y. Huang and Z. Yin, "The Compact Finite Difference Method of Two-Dimensional Cattaneo Model", Journal of Function Spaces, 2020.

[2] J. M. G-Jordan, S. Rojas, M. F-Villegas, and J. E. Castillo, "A new second order finite difference conservative scheme", Divul-gaciones Matematica, vol. 13, no. 1, pp. 107–122, 2005.

[3] H. Calandra, S. Gratton, X. Pinel, and X. Vasseur, "An improved two-grid preconditioner for the solution of three-dimensional Helmholtz problems in heterogeneous media", CERFACS, Tou-louse, France, Technical Rport TR/-PA/12/2,2012. [Online].

[4] I. Duff, S. Gratton, X. Pinel, and X. Vasseur, "Multigrid Based Preconditioners for the Numerical Solution of Two-dimensional Heterogeneous Problems in Geophysics", Int. J. Comput. Math., vol. 84, no. 8, pp. 1167–1181, 2007.

[5] M. S. A Shafee, Sheikh o leslami, Mohsen,"Impact of Lorentz forces on Fe3O4-water ferrofluid entropy and exergy treatment within a permeable semi annulus", Journal of cleaner production, vol. 221, pp. 885-898, 2019.

[6] Sheikh, N.A. and Ching, D.L.C. and Khan, "A new model of fractional Casson fluid based on generalized Fick's and Fourier's laws together with heat and mass transfer", Alexandria Engineering Journal vol.59, no. 5, pp. 2865-2876, 2020.

[7] Nadeem Ahmad, Sheikh, Muhammad Jamil, "A generalized model for quantitative analysis of sediments loss: a Caputo time fractional model", Journal of King Saud University-Science, vol. 33, no. 1, 2021.

[8] M. A. Jones, "A Difference Equation Approach to Finite Differ-ences of Polynomials", The College Mathematics Journal, vol. 51, no. 5, pp. 375–377, 2020.

[9] S.Bilal, Rashid, Mahmood and A.H.Majeed, "Finite element method visualization about heat transfer analysis of Newtonian material in triangular cavity with square cylinder", Journal of Materials Research and Technology, vol. 9, no. 3, pp. 4904-4918, 2020.

[10] Musiliu, Folarin Farayol, and M. F. I Khan, "Mathematical modeling of radiotherapy cancer treatment using Caputo fractional derivative", Computer Methods and Programs in Biomedicine, vol. 188, 2020.

[11] M. B. Gijzen, Y. A. Erlangga, and C. Vuik, "Spectral Analysis of the Discrete Helmholtz Operator Preconditioned with a Shifted Laplacian", SIAM Journal on Scientific Computing, vol. 29, pp. 1942–1958, 2007.

[12] B. Hustedt, S. Operto, and J. Virieux, "Mixed-grid and staggered grid finite-difference methods for frequency-domain acoustic wave modelling", Geophysical Journal International, vol. 157, no. 3, pp. 1269–1296, 2004.

[13] C. W. Groetsch and J. T. King, "The Bernstein Polynomials and Finite Differences", Mathematics Magazine, vol. 46, no. 5, pp. 280–282, 1973.

[14] L. L. ESM Sherif, "Dual solutions and stability analysis of a hybrid nano fluid over a stretching/shrinking sheet executing MHD flow", Symmetry, vol. 12, no. 2, pp.276, 2020.

[15] S. A. KS Nisar, "Numerical modeling and theoretical analysis of a nonlinear advection-reaction epidemic system", Computer Methods and Programs in Biomedicine, vol. 193, pp. 105429, 2020.

[16] A. S. I Khan, "Analysis of the COVID-19 Pandemic Spreading in India by an Epidemiological Model and Fractional Differential Operator", Preprints 2020, 2020050266 (doi: 10.20944/preprints202005.0266.v1).

[17] F. Ihlenburg, "Finite element analysis of acoustic scattering". New York: Springer, Part of the Applied Mathematical Sciences book series, vol. 132, 1998.

[18] M. N. AH Soori, "Mathematical analysis of novel Coronavirus (2019-ncov) delay pandemic model", Comput. Mater. Continua, vol. 64, no. 3, pp. 1401-1414, 2020.