Finite Element Solution to Integral Equations

Dieudonne Ndong Ovono

Abstract—Some complex natural phenomena in physics, economics and engineering can be represented by Integral Equations (IE). But one may be short of general mathematical tricks to solve them analytically, which could quickly turn into a life time enterprise. In the contrary, the use of computers or numerical methods introduced some decades ago has brought a different approach that is innovative and effective.

In addition, the theory of measure, the study of polynomials and normed spaces, namely Banach, Hilbert, Lebesgue, H"older, Lipschitz and Sobolev spaces led to advanced modern numerical methods to solve these complex IEs. When using a computer, a generally continuous domain is divided into smaller pieces, or elements, so the calculation on any point of the continuous domain could be done using results from the elements; it is similar to approximate a circle by a regular polygon; as the number of side increases, the polygon become more and more close to a circle.

This method of solving IEs using subdivisions of the domain, along with discrete form of the equations, is referred to as the Finite Element Method (FEM). In this paper, we implement the advanced core theory of the finite element method into adaptive meshes for generic complex problems represented by IEs.

Index Terms—Finite Element, Integral Equation, Interpolation Operator, Adaptive Mesh.

I. INTRODUCTION

This paper covers the implementation of the FEM that we have developed from scratch, which helped us fully extend our understanding of the topic and program the actual computer codes in solving IEs and be able to compare various approximation methods. This paper is organized into three main parts.

The first part summarizes the theoretical foundation of the FEM as well as an overview of the Finite Element interpolation technique. It restates the results on normed vector spaces ([I], p.463-492; [3], p.13-17; [4]), Lebesgue integration, distributional derivatives and Sobolev spaces ([I], p.463-492; [3], p.13-17), essential to derive “error estimate”, convergence and well posedness.

Next, we review Integral Equations ([2], p.222-337; [3], p.31-42) and their approximation methods and results.

The last part, Implementation, covers practical implementation considerations such as meshing techniques, “a posterior error estimate” for mesh refinement or adaptive mesh ([I], p.337-457; [3], p.68-112), examples illustrating our simulator results.

II. THEORETICAL FOUNDATIONS OF THE FINITE ELEMENT METHOD

A. Theoretical Framework and Fundamental Analysis

Please refer to ([1], p.463-492; [3], p.13-17) for a complete overview.

Lebesgue Spaces: $L^p(\Omega)$ is the space of the scalar-valued functions that are Lebesgue-integrable. The space of locally integrable functions is denoted by $L^1_{\text{loc}}(\Omega)$ and is defined as:

For $1 \leq p \leq +\infty$, let $L^p(\Omega) = \{f \in M(\Omega) ; \|f\|_{0,p,\Omega} < +\infty\}$ where

$$\|f\|_{0,p,\Omega} = \left(\int_{\Omega} |f(x)|^p \, dx\right)^{1/p}$$

for $1 \leq p \leq +\infty$, and $\|f\|_{0,\infty,\Omega} = \text{ess sup}|f(x)| = \inf\{M \geq 0 ; |f(x)| \leq M, f \in \Omega\}$

Sobolev Spaces: Let $s$ and $p$ be two integers with $s \geq 0$ and $1 \leq p \leq +\infty$, the Sobolev space is defined as:

$$W^{s,p}(\Omega) = \{u \in D'(\Omega) ; \partial^\alpha u \in L^p(\Omega), |\alpha| \leq s\}$$

with distributional derivatives ([1], p.463-492). There follows that $W^{s,p}(\Omega)$ is a Banach space when equipped with the norm:

$$\|u\|_{W^{s,p}(\Omega)} = \sum_{|\alpha| \leq s} \|\partial^\alpha u\|_{L^p(\Omega)}$$

For $p = 2$, $W^{2,2}(\Omega)$ has an Hilbert structure and is denoted by $H^2(\Omega)$.

B. Overview of the Finite Element Interpolation

Please refer to ([1], p.3-58; [3], p.18-25) for a complete literature.

Polynomials Used for One-Dimensional Interpolation:

For $1 \leq p \leq +\infty$, a function $f \in L^p(\Omega)$ is approximated by a polynomial $p \in \mathbb{P}_k(\Omega)$ with $\|f - p\|_{L^p(\Omega)}$ being the $L^p$-norm of the error of approximation. The degree of the polynomial is usually denoted by $k$.

For $k \geq 1$ and $\delta \in (0,1)$, the Lagrange polynomial $p_k^\delta(\cdot)$ is defined by

$$p_k^\delta(x) = \sum_{i=0}^k \frac{f(x_i)}{\binom{k}{i} (x-x_i)^{k-i} (x-x_i)^\delta (x-x_i+\delta)^{k-i}}$$

for $x \in \Omega$. It is well known that $p_k^\delta(x)$ is the unique polynomial of degree at most $k$ that interpolates $f$ at the nodes $x_i$. The family of polynomials $\mathbb{P}_k(\Omega)$ is a space of polynomials of degree at most $k$.

The $p_k^\delta$ Lagrange finite element: Let $k \geq 1$ and let $\{s_0, \ldots, s_k\}$ be $k+1$ distinct numbers. The Lagrange polynomials $\{L_0^k, \ldots, L_k^k\}$ associated with the nodes $\{s_0, \ldots, s_k\}$ are defined to be $L_i^k(t) = \prod_{m \neq i} (t-s_m) / \prod_{m \neq i} (s_i-s_m)$, $0 \leq m \leq k$ and satisfy the important property $L_i^k(s_i) = \delta_{i,l}$, $0 \leq m, l \leq k$. For $j \in \{0, \ldots, k(N+1)\}$ with $j = ki + m$ and $0 \leq m \leq k - 1$, define the functions

$$\varphi_{ki+m}(x) = \begin{cases} L_i^k(x) & \text{if } x \in I_j \\ 0 & \text{otherwise} \end{cases}$$

Manuscript received September 29, 2014; revised Jan 02, 2015.

D.N.Ovono is an oil & gas engineer and researcher, he works for Halliburton, phone: 1-832-706-4036
(e-mail:arthur_ovono@yahoo.com).
The Crouzeix-Raviart finite element: \( \mathcal{P} \) family of points in \( s_h \leq n \)

Main definition: following Ciarlet, a finite element is \( i, j \leq n \) local degrees of freedom \( d \), this property is known as \( p=0 \) \( \sum \).

Then, there exists a basis \( \{ \varphi_1, ..., \varphi_{n_h} \} \) in \( P \), called the local shape functions, such that \( \varphi_i(\delta_j) = p(\delta_j), 1 \leq i, j \leq n_h \); dim \( P = \text{card}(\Sigma) = n_h \) and \( \forall p \in P, \varphi_i(p) = 0, 1 \leq i \leq n_h \), \( \varphi_i(p) \neq 0 \) (for \( p = 0 \)). This property is known as unisolvence.

Lagrange finite element: for all \( p \in P, \sigma_i(p) = p(\delta_j) \), \( 1 \leq i \leq n_h \); \( K, \Sigma \) is called a Lagrange finite element. The local shape functions \( \{ \vartheta_1, ..., \vartheta_{n_h} \} \), with \( \vartheta_i(\delta_j) = \delta_{ij}, 1 \leq i, j \leq n_h \), are called the nodal basis of \( P \).

Simplicial Lagrange finite element: Let \( \{ a_0, ..., a_d \} \) be a family of points in \( \mathbb{R}^d \), \( d \geq 1 \). Assume that the vector \( a_1 - a_0, ..., a_d - a_0 \) are linearly independent. Then, the convex hull of \( \{ a_0, ..., a_d \} \) is called a simplex, and the points \( \{ a_0, ..., a_d \} \) are called the vertices of the simplex. Let \( P_k \) be the space of polynomials in the variables \( x_1, ..., x_d \), with real coefficient and global degree at most \( k \):

\[
P_k = \left\{ p(x) = \sum_{0 \leq i_1 + i_2 + ... + i_d \leq k} a_{i_1, ..., i_d} x_1^{i_1} ... x_d^{i_d} \mid a_{i_1, ..., i_d} \in \mathbb{R} \right\},
\]

\( P_k \) is a vector space of dim \( P_k = (d+k) \)

There is also “Tensor Product” and “Prismatic” finite elements where the shape of cell is a cuboid, respectively a prism.

The Crouzeix-Raviart finite element:

\[
P = P_1 \text{ and take for the local degrees of freedom the mean-value over the } (d+1) \text{ faces of } K, \text{ i.e.}, \text{ for } 0 \leq i \leq d, \sigma_i(p) = \frac{1}{\text{meas}(F_i)} \int_{F_i} p.
\]

Let \( \Sigma = \{ \sigma_i \}_{0 \leq i \leq d} \) then \( \{ K, P, \Sigma \} \) is a finite element. The local shape functions are \( \vartheta_i(x) = d \left( \frac{1}{d} - \lambda_i(x) \right), 0 \leq i \leq d \). The local Crouzeix-Raviart interpolation operator is then defined as follows \( \mathcal{T}_k^R : \mathcal{V}(K) \ni v \mapsto \mathcal{T}_k^R v = \sum_{i=0}^{d} \left( \frac{1}{\text{meas}(F_i)} \int_{F_i} v \right) \vartheta_i \in P_k \)

Other finite element type are Raviart-Thomas, Nedelec, Hermite, etc.
are involved. The most frequent form is the Fredholm equation:
\[
(\alpha(x) y(x) - F(x) + \lambda \int_a^b K(x,e) y(e) \, de)
\]
If \(b = x\) is identified with the current variable, the equation is known as the Volterra equation. \(\lambda, a\) and \(b\) are constants and \(\alpha, F\) and \(K\) are given functions, continuous on \((a, b)\) as well as \(y(x)\). \(K(x, \epsilon)\) is known as the kernel. The integral equation is said to be of the first kind if \(a \equiv 0\), second kind if \(a \equiv 1\) and of the third kind if \(\alpha\) is a function. If \(\alpha\) is positive, the Fredholm equation can take the form
\[
\sqrt{\alpha(x)} y(x) = \frac{F(x)}{\sqrt{\alpha(x)}} + \lambda \int_a^b \sqrt{\alpha(x)} \sqrt{\alpha(e)} y(e) \, de
\]
and be considered of the second kind in the unknown function \(y(x) y(x)\) with a modified kernel. For a two-dimensional variable \(w(x, y)\), the Fredholm equation is of the form
\[
\alpha(x, y) w(x, y) = F(x, y) + \lambda \int_a^b \int_a^b \phi(x, \epsilon) \psi(\epsilon, y) \, d\epsilon \, dy
\]
where \(\phi(x)\) and \(\psi(\epsilon, y)\)

**B. Numerical Solution of Integral Equations**

**Approximation Methods of Undetermined Coefficients**

In this method the solution of
\[
y(x) = F(x) + \lambda \int_a^b \phi(x, e) y(e) \, de
\]
is approximated by
\[
y(x) = \lambda \sum_{k=1}^n c_k K(x, x_k) y(x_k)
\]
where \(y(x_1) = F(x_1) + \lambda \sum_{k=1}^n D_k K(x, x_k) y(x_k)\) or \(y_1 = f_1 + \lambda \sum_{k=1}^n K_i y_k\). If \(y_1 = \{y_1\}\), \(K = [K_{ij}]\), \(D = [D_{ij}]\) and \(f = \{f_i\}\), then \((I - \lambda K D) y = f\) where \(I\) is the unit matrix of order \(n\). This method is particularly useful when the kernel \(K\) is available as a table of values of an empirical influence function. In the other hand, the characteristic numbers of
\[
y(x) = \lambda \int_a^b K(x, \epsilon) y(\epsilon) \, d\epsilon
\]
dc are afforded by reciprocals of the characteristic numbers of the corresponding matrix formulation \(y = \lambda \sum_{k=1}^n K_{ik} y_k = \lambda K D y\).

**Approximation Methods of Undetermined Coefficients**

In this method the solution of
\[
y(x) = F(x) + \lambda \int_a^b K(x, e) y(e) \, de
\]
is approximated by
\[
y(x) = \sum_{k=1}^n c_k \phi_k(x)\]
leading to the form
\[
\Phi(x) = \lambda \sum_{k=1}^n c_k \phi_k(x)
\]
where \(\Phi(x) = \lambda \int_a^b K(x, \epsilon) \phi(\epsilon) \, de\) \((a \leq x \leq b)\) and the \(\phi_k\) are suitably chosen functions such as \(x^k\) and the \(c_k\)'s are determined by a set of \(n\) algebraic equations. If end values are known in advance, it may be desirable to use
\[
y(x) = \sum_{k=1}^n c_k \phi_k(x)
\]
with \(\phi(x)\) verifying the known end conditions and the remaining \(\phi\),'s vanishing at end points.

**The Method of Collocation**

Following the preceding section by setting \(s_i(x) = \phi_i(x)\) \((a \leq x \leq b)\) and by requiring that \(\sum_{k=1}^n c_k \phi_k(x) = F(x)\) be an equality at \(n\) distinct points in \((a, b)\), we obtain the \(n\) conditions
\[
\sum_{k=1}^n c_k s_k(x_i) = F(x_i) \text{ for } i = 1, \ldots, n
\]
leading to the \(n\) algebraic equations in \(c_k\), hence \(y(x)\).

**The Method of Weighting Functions or Galerkin**

This method uses the \(n\) orthogonality conditions of the form
\[
\sum_{x=1}^n c_k s_k(x_i) = \int_a^b s_i F dx \quad (i = 1, 2, \ldots, n)\]
or \(Mc = b\) where \(M = [m_{ij}]\) with \(m_{ij} = \int_a^b s_i s_j F dx\) and \(b_i = \int_a^b s_i F dx \quad (i = 1, 2, \ldots, n)\); a convenient choice of \(s_i\) is \(1, x, x^2; x^n\); it is desirable to choose the \(s_i\) as a complete set of functions. The system \(Mc = b\) leads to \(n\) algebraic equations in \(c_k\), hence \(y(x)\).

**The Method of Least Square**

This method requires that the integral of the square of the difference \(\sum_{x=1}^n c_k s_k(x) - F(x)\) is as small as possible or
\[
\int_a^b (\sum_{x=1}^n c_k s_k(x) - F(x))^2 dx \text{ - minimum and avoids dependence on the choice of the collocation points or weighting functions. There then follows that}
\]
\[
\sum_{x=1}^n c_k s_k(x) - F(x) = \int_a^b s_i F dx \quad (i = 1, 2, \ldots, n)\]
leading to \(n\) algebraic equations in \(c_k\). If \(\int_a^b s_i^2 dx = \int_a^b F dx\) \((i = 1, 2, \ldots, n)\); \(Mc = b\) with change in indices, we obtain
\[
\sum_{x=1}^n c_k s_k(x) = \int_a^b s_i F dx \quad (i = 1, 2, \ldots, n)\]
with \(P = \{p_{ij}\} = \sum_{x=1}^n s_i s_j F dx = S^T D S = S^T \text{diag} = S^T D S\) and \(q = \{q_i\} = \sum_{x=1}^n s_i D_k f_i = S^T D f\) where \(S = [s_{ij}]=\{s(x_i)\}\), \(D = [D_{ij}]\) and \(f = \{f_i\}\).

Since \(D\) is diagonal, \(S^T D = (DS)^T\), \(P = (DS)^T S\) and \(q = (DS)^T f\), this result leads to the following procedure for determining the \(n\) linear equations represented by
\[
\sum_{x=1}^n c_k s_k(x) = F(x) \quad (i = 1, 2, \ldots, N)
\]
1) Choose \(N\) points in \((a, b)\) and write down the \(N\) equations. 2) Denote by \(S\) the \(N \times n\) matrix of coefficients in the previous set of equations and form the “weighting matrix” \(S' = DS\) by multiplying the \(i^{th}\) row of \(S\) by the weighting coefficient \(D_i\), associated with the point \(x_i\) in an appropriate integration scheme involving the \(N\) points. 3) Pre-multiply the augmented matrix of \(\sum_{x=1}^n c_k s_k(x) = F(x) \quad (i = 1, 2, \ldots, N)\) by the transpose of the weighting matrix \(S'\). The resultant matrix is the augmented matrix of the required set of \(n\) linear equations which determines the constants \(c_1, c_2, \ldots, c_n\).

**Approximation of the Kernel**

It is sometimes convenient to approximate a kernel by a polynomial in \(x\) and \(e\) or by a separable kernel and solve the resultant equation according to the method of separable kernel. For instance, a kernel can be approximated by \(A_1 + A_2 x + A_3 x^2\) or \(x(1-x)(B_1 + B_2 x + B_3 x^2)\) which should verify the end conditions and where the \(A\)’s and \(B\)’s are determined as functions of \(e\) by three-point collocation, the use of appropriate weighting functions or the use of least-square techniques.
\[ y(x) = x + \int_a^x K(x, \epsilon) y(\epsilon) d\epsilon \quad \text{where} \quad K(x, \epsilon) = \begin{cases} x & \text{when} \ x < \epsilon \\ \epsilon & \text{when} \ x > \epsilon \end{cases} \]

a) Exact solution:

By writing \[ \int_a^x K(x, \epsilon) y(\epsilon) d\epsilon = \int_a^x \frac{\partial G(x, \epsilon)}{\partial x} (x, B(x)) \frac{dB(x)}{dx} - G(x, A(x)) \frac{dA(x)}{dx} \] we obtain

\[ y'(x) = 1 + \frac{\partial y(x)}{\partial x} + x. y(x) \frac{dy(x)}{dx} + \text{const} \]

b) Method of Collocation:

Assuming the approximation \[ y(x) = c_1 + c_2 x + c_3 x^2 \]

We have \[ \phi_1 = x, \phi_2 = x^2 \]

By \[ J_a^b K(x, \epsilon) \phi_2(\epsilon) d\epsilon \] we obtain

\[ \phi_1 = \int_a^b e \epsilon d\epsilon + x \epsilon dx = x - 2 x^2 \]

\[ s_1(x) = \phi_1(x) = 1 - x + \frac{x^2}{2} \]

\[ s_2(x) = \frac{x}{2} \]

\[ \phi_2(x) = \frac{x}{2} \]

\[ s_3(x) = 1 - x + \frac{x^2}{2} \]

Hence, the approximate equality \[ \sum_{n=1}^b c_1 s_1(x) + c_2 s_2(x) + c_3 s_3(x) \approx x \] produces

\[ c_1 (1 - x + \frac{x^2}{2}) + c_2 \frac{x}{2} + c_3 (\frac{x}{2} - x + \frac{x^2}{2}) \approx x \]

\[ c_1 x + c_2 x + c_3 x^2 \]

\[ \approx x \]

\[ \frac{1}{2} c_1 x + \frac{1}{2} c_2 x + \frac{1}{2} c_3 x = 1 \]

\[ \frac{1}{2} c_1 + \frac{1}{2} c_2 + \frac{1}{2} c_3 = 1 \]

1.97895 \times 0.43373 x^2

c) Method of Weighted Coefficient or Galerkin:

Assuming the weighting functions to be \[ \psi_1 = x, \psi_2 = x^2 \]

We have \[ \phi_3 = x^3 \]

Galerkin system of equations

\[ \sum_{n=1}^a c_1 \psi_1 \psi_1 d\epsilon = \int_a^b \psi_1 \psi_1 d\epsilon + \text{const} \]

\[ (i = 1, 2, 3; k = 1, 2, 3) \]

becomes:

\[ c_1 \int_a^b \left(1 - x + \frac{x^2}{2}ight) dx + c_2 \int_a^b (\frac{x}{2} + \frac{x^2}{6}) dx + c_3 \int_a^b (\frac{1}{3} - \frac{x^2}{2} + \frac{x^4}{12}) dx = \int_a^b x dx \]

\[ c_1 \int_a^b (\frac{1}{3} x^2) + c_2 \int_a^b (\frac{1}{3} x + \frac{1}{4} x^2) dx + c_3 \int_a^b (\frac{1}{3} x^2 + \frac{1}{4} x^3) dx = \int_a^b x dx \]

d) Method of Least Square:

We evaluate \[ c_1 (1 - x + \frac{x^2}{2}) + c_2 (\frac{x}{2} + \frac{x^2}{6}) + c_3 (\frac{x}{2} + \frac{x^2}{12}) \]

\[ c_3 (-\frac{x}{2} + \frac{x^2}{12}) = 0 \]

\[ x \text{ at 0, 0.25, 0.5, 0.75 and 1 (refer to the above point b) for the evaluation at 0, 0.5 and 1) and calculate the weighting matrix } S^b \text{ by:} \]

\[ S^b = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \]

\[ x \]

\[ 2 c_1 + \frac{c_2}{2} + c_3 = 1 \]

\[ \frac{1}{2} c_1 + \frac{1}{2} c_2 + \frac{1}{2} c_3 = 1 \]

\[ \frac{1}{2} c_1 + \frac{1}{2} c_2 + \frac{1}{2} c_3 = 1 \]

\[ c_1 = 0 \]

\[ c_2 = \frac{1}{2} \]

\[ c_3 = \frac{1}{2} \]

\[ S^b = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

Hence the approximation

\[ y(x) = -0.0123 + 2.0147 x - 0.4333 x^2 \]

e) Method of the Approximation of the Kernel:

Now, if we assume the kernel is such that \[ K(x, \epsilon) \approx c_1 + c_2 x + c_3 x^2 \]

The evaluation of \[ \int_0^1 K(x, \epsilon) d\epsilon \] gives \[ \int_0^1 (c_1 + c_2 x + c_3 x^2) dx = \int_0^1 x dx \]

\[ \int_0^1 x dx = \frac{1}{2} \]

\[ 0.46680 \quad 0.16800 \quad 0.05934 \quad 0.29167 \]

\[ 0.16800 \quad 0.20790 \quad 0.09600 \quad 0.20090 \]

\[ 0.20790 \quad 0.09600 \quad 0.08660 \quad 0.15289 \]

\[ c_1 = -0.0123 \]

\[ c_2 = 2.0147 \]

\[ c_3 = -0.4333 \]

\[ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

\[ 0.46680 \quad 0.16800 \quad 0.05934 \quad 0.29167 \]

\[ 0.09514 \quad 0.09509 \quad 0.03658 \quad 0.15289 \]

\[ 0.46680 \quad 0.16800 \quad 0.05934 \quad 0.29167 \]

\[ 0.16800 \quad 0.20790 \quad 0.09600 \quad 0.20090 \]

\[ 0.09514 \quad 0.09509 \quad 0.03658 \quad 0.15289 \]

\[ c_1 = -0.0123 \]

\[ c_2 = 2.0147 \]

\[ c_3 = -0.4333 \]
\[ \int_0^1 x \, dx = x - \frac{x^2}{2}. \]  
(a) The integral of the kernel must equal its approximation over \((0,1)\), hence \( c_1 + \frac{c_2}{2} + \frac{c_3}{3} = x - \frac{x^2}{2}. \)

Additionally, the kernel approximation is exact at ends points \(0\) and \(1\):  
(b) \( K(0, \epsilon) = x|_{x=0} = c_1 \) and \( c_1 = 0 \)

(c) \( K(1, \epsilon) = \epsilon|_{x=1} = \epsilon = c_1 + c_2 + c_3 \), Conditions (a), (b) and (c) lead to the system

\[
\begin{align*}
 c_1 + \frac{c_2}{2} + \frac{c_3}{3} &= x - \frac{x^2}{2} \\
 c_1 &= 0 \\
 c_1 + c_2 + c_3 &= \epsilon \\
 c_1 &= 0 \\
 c_2 &= \epsilon(4 - 3\epsilon) \\
 K(\epsilon, x) &= \epsilon(4 - 3\epsilon)x + 3\epsilon(\epsilon - 1)x^2 = 3\epsilon(\epsilon - 1) \\
 \end{align*}
\]

Using the method of *separable kernel*, the following steps lead to the approximation:

\[ y(x) = x + \int_0^1 K(x, \epsilon)y(\epsilon)\,d\epsilon \]

\[ x + \int_0^1 [\epsilon(4 - 3\epsilon)]y(\epsilon)\,d\epsilon + x^2 \int_0^1 [3\epsilon(\epsilon - 1)]y(\epsilon)\,d\epsilon \]

Let

\[ a_1 = \int_0^1 [\epsilon(4 - 3\epsilon)]y(\epsilon)\,d\epsilon \mathrm{ and } a_2 = \int_0^1 [3\epsilon(\epsilon - 1)]y(\epsilon)\,d\epsilon \]

Multiplying (I) successively by \(g_1(x) = x(4 - 3x)\) and \(g_2(x) = 3x(x - 1)\) and integrating over \((0,1)\), we obtain the system:

\[ \begin{align*}
 a_1 &= \frac{1}{12} \left( 5a_1 + \frac{7}{5} + \frac{3}{2}a_2 \right) \\
 a_2 &= \frac{1}{4} \left( \frac{3}{3}a_1 - \frac{1}{2} + \frac{1}{2}a_2 \right) \\
 a_1 &= \frac{137}{139} \approx 0.9856 \mathrm{ and } a_2 = -\frac{60}{139} \approx -0.4317 \\
 y(x) &= x + 0.9856x - 0.4317x^2 = 1.9856x - 0.4317x^2. \\
 \end{align*} \]

IV. IMPLEMENTATION

This section is based on ([3], p.68-112), inspired by ([1], p.337-457).

**Mesh Generator:**

We have developed our Delaunay Triangulations (interior of the circumscribed sphere does not contain any vertex of the triangulations) and non-Delaunay options with insertion of triangle centers.

**Alternative Mesh Generator**

There are many types of triangle centers. We use them to generate the mesh; this is a non-Bowyer-Watson mesh generation. However, it can be shown that a Delaunay mesh is easily reached from a square diagonally split, then by simple successive insertion of the circumcenter. This is our fundamental results for building a mesh.

**Modified Bowyer-Watson Algorithm:**

The algorithm inserts successive points inside a triangle and split is in finer pieces, refining the mesh to a better representation of the domain. The more points are inserted, the more represented is the domain and the more accurate is the solution. This is similar to approaching a circle with a polygon to which the number of sides is gradually increased.

**Modified Delaunay Algorithm**

To generate a mesh, first we construct a background mesh that will be attached to the domain at its interpolation points with that domain. The elements that overlap the boundary will be replaced by their parts that are interior to the domain when the parts that are exterior will be deleted from the mesh. And, if the part that is interior is not a triangle, it will be triangulated. The mesh can be initially variable in each direction (for instance, for 2D mesh, Ni intervals in \(x\) direction and \(Nj\) intervals in \(y\) direction).

**Illustrative Example of Mesh of Complex 2D Polygon Shape**

Fig. 1 - Meshing a U Shape Reservoir before Refinement

**Quadratures, Assembling and Storage Assembling**

This refers to the phase in the finite element program where the entries of the stiffness matrix and the right-hand side vector are computed.

**Storage and Sparse Matrices**

Let \( \mathcal{A} \in \mathbb{R}^{N \times N}. \) Denote by \( nnz \) the number of non-zero entries in \( \mathcal{A}. \) The matrix \( \mathcal{A} \) is said to be sparse if \( nnz \ll N \times N. \) CSR or CSC format or Ellpack-Itpack format methods are used to condition the matrix.

**Linear Algebra**

We used conditioning and reordering techniques to save memory and accelerate the convergence to the solution of the system. We used LU factorization, LDLT factorization and Choleski factorization.

**Posteriori Error Estimates and Adaptive Meshes**

The goal is to assess the error between the exact solution and its finite element approximation in term of known quantities only, i.e., the size of the mesh cells, the problem data and the approximate solution. The *a posteriori estimate* assesses the quality of the approximation solution and provides information to construct a new mesh, a process that can be repeated several times, thereby generating a sequence of so-called *adaptive meshes*.

**Adaptive Mesh Generation**

We use the local errors indicators to generate a new mesh:

\[ |\mathcal{E}| \leq (\sum_{K \in \mathcal{A}} \eta_K(u_K,f))^2 \]

The quantities \( \eta_K(u_K,f) \) are error indicators and can be readily evaluated when using residual–based or hierarchical
error estimates. For duality techniques, \( \eta_K(u_\infty, f) \) requires solving a dual problem.

Implementation of Iterative Techniques:
We used “Direct Methods” (Gaussian Pivot and Gauss-Jordan Pivot [2], p.1; Crout Method [2], p.339) and “Indirect Methods” (Jacobi, Gauss-Seidel [5]; Krylov Spaces: CG and Bi-CG Stabilized [1], p.401-413 and [6], p.11-64).

Example of Time-Marching Algorithms (Case of PDEs):
The Implicit Euler is given by the formulation
\[
\frac{1}{\Delta t} (u_h^{n+1} - u_h^n, v_h)_L + (A u_h^{n+1}, v_h)_L = (f^{n+1}, v_h)_L, \forall v_h \in V_h
\]
Other time-marching algorithms programmed are Explicit Euler, Leap-Frog and Backward-Differential. More details could be found in [1], p.279-334.

The Finite Element Calculator (FEC©)
A software was specifically developed to illustrate the application of the method of finite element ([3], p.68-112). It is named the “Finite Element Calculator©” which is self-explanatory. Our objective is to build a library of solutions of real life problems such as heat transfer, reservoir simulation, earthquake, economical predictions, quantum mechanics, etc.

Example 1: Simulated Atm. Pressure along a cockpit:
Using a 15x15 on a \([0,1] \times [0,1]\) background mesh, we obtain the following:

![Profile of Atm. Depression along the Cockpit](image)

Example 2: Water Saturation Profile in U shape Reservoir:

![U Shape Reservoir Layer – Water Saturation Profile, perspective view](image)

V. CONCLUSION

Complex natural phenomena and engineering applications can often be represented by Integral Equations (IE).

Because it could be challenging to solve some of these equations using conventional and/or analytical methods, we use numerical techniques by mean of computers. The most powerful method of the latter is the so called the Finite Element Method (FEM). The FEM refers not only to the partitioning of a domain in smaller pieces called cells or elements, constituting the mesh, but also the use of mathematical polynomials as shape functions, the discretization of the equations, leading to a system of linear equations in matrix format, more suitable for computer iterative solving techniques.

This paper presented our implementation of the FEM using enhanced adaptive meshes to solve and simulate a broad range of real life complex problems that can be formulated in the forms of Integral Equations.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L^p(\Omega) )</td>
<td>Functions whose p-th power is Lebesgue integrable</td>
</tr>
<tr>
<td>( | f |_X )</td>
<td>Norm of ( f ) in the normed space ( X )</td>
</tr>
<tr>
<td>( W^{s,p}(\Omega) )</td>
<td>Function whose derivatives up to order ( s ) are in ( L^p(\Omega) )</td>
</tr>
<tr>
<td>( \delta^{\alpha} u )</td>
<td>( \frac{\partial^\alpha}{\partial \xi^\alpha} u ) is a multi-index</td>
</tr>
<tr>
<td>( L^m_\infty(S) )</td>
<td>Lagrange Polynomial</td>
</tr>
<tr>
<td>( \hat{\psi}_i )</td>
<td>geometric reference shape functions</td>
</tr>
<tr>
<td>( \delta_{ijkl} )</td>
<td>Kronecker symbol</td>
</tr>
</tbody>
</table>

ACKNOWLEDGEMENT

This paper is submitted as a continuation of my works started during my PhD in Applied Mathematics related to finite element implementation. Therefore, I would like to take this opportunity to express my gratitude to those that have been very supportive and of great help, many of my great colleagues, friends, former teachers, parents and close family, my mother Clementine Avome Ndong and my uncle, Paul Ovono Ndong for their moral encouragements throughout my life and studies.

REFERENCES