UNIVERSITE MOHAMED BOUDIAFDE M'SILA

Faculté des Mathématiques et de l'Informatique

Département de Mathématiques

MEMOIRE DE FIN D'ETUDE

Présenté pour l'obtention du Diplôme de **MASTER**

Domaine : Mathématiques et Informatique

Filière: Mathématiques

Option : Equations aux dérivées partielles et applications

Par

AZOUZ SOURIA

Sujet

La résolution numérique de l'équation intégrale de Volterra de seconde espèce par les fonctions splines polynomiales et les fonctions splines non Polynomiales

Soutenu le 01/07/2019

Devant le jury :

Dr.Nouiri Brahim	MCA	Université de M'sila	Président
Dr. MerzouguiAbdelkrim	MCA	Université de M'sila	Rapporteur
Dr. Djaija Noui	MAA	Université de M'sila	Examinateur

Promotion : 2018 / 2019

Remerciements

Louagne à notre seigneur **"ALLAH"**qui nous a dotés de la merveilleuse faculté de raisonnement. Louagne à notre créateur qui nous incités à acquérir le savoir. C'est à lui que nous adressons toute notre gratitude en premier lieu.

Je tiens à remercier mon promoteur Mr le docteur **Merzougui Abdelkrim** pour la confiance qu'il m'a témoignée en me proposant ce sujet, ses encouragements et sa patience. Les discussions scientifiques qu'il a su générer, ses remarques et ses suggetions m'ont permis de finaliser ce modeste travail. Je souhaite lui transmettre ma reconnaissance et ma plus profonde gratitude.

Je remercie aussi tous les membres du Jury. Monsieur **Nouiri Brahim**, d'avoir accepter de présider ce mémoire. Monsieur **Djaija Noui**, d'avoir d'examiner ce mémoire.

Je tiens à remercier Monsieur **Zouarg Yahia**, pour tout l'aide qu'il ma apporté et sa patience , et ses conseils.

Je ne peux pas clôturer mes remerciments sans se retourner vers les êtres qui me sont les plus chers ; ma famille qui ont eu un rôle essentiel et continu dans ma réussite.

Table des matières

Notations

$\mathbb{R}$: Ensemble des nombres réels.

$\mathbb{C}$: Ensemble des nombres complexes.

H : Espace de Hilbert.

$\langle ., . \rangle$: Produit scalaire.

A : Opérateur intégrale

$K(x, t)$: Noyau de l'équation l'intégrale.

$f(x)$: Fonction donnée.

λ : Un paramètre non nul, réel ou complexe.

A^*: Opérateur adjoint.

$VIE's$: Les équations intégrales de Volterra.

LP : Fonction spline polynomiale de premier ordre.

QP : Fonction spline polynomiale de seconde ordre.

NLP : Fonction spline non polynomiale linéaire.

NQP : Fonction spline non polynomiale quadratique.

$u(x)$: Solution exacte.

$P_i(x)$: Solution approchée.

Introduction

Les méthodes numériques permettent de trouver la solution approchée des problèmes dont la solution exacte est difficile voir impossible a trouvé analytiquement, d'où la nécessité des méthodes numériques. Différents méthodes ont été développées et utilisées par plusieurs recherches pour trouver la solution approchée des équations intégrales..

Dans ce mémoire, On considère L'équation intégrale de Volterra de 2^{nd} espèce, de la forme:

$$u(x) = f(x) + \lambda \int_a^x k(x,t)u(t)dt \quad x \in [a,b]$$

Les équations intégrales de Volterra de 2^{nd} espèce jouent un rôle important dans l'étude des problèmes mathématiques, physiques, ingénieries, comme dans les problèmes de la mécaniques, les problèmes de diffusion, les théories potentiels, les problèmes de Dirichlet, électrostatiques, astrophysiques, et le problème de transport, aussi les équations différentielles fractionnaires, les système dynamiques Parmi les méthodes, on a la méthode de Galerkin, la méthode de collocation la méthode de l'expansion [14] , nous cherchons la solution approchée de VIE's de 2^{nd} espèce par les fonctions splines polynomiales de premier et de seconde ordre et par les fonctions splines non polynomiales linéaire et quadratique.

Notre but est d'essayer de montrer l'efficacité de ces méthodes à l'amélioration de solution.

La structure de mémoire est organisée comme suit:

Le chapitre 1 constitue des résultats préliminaires sur l'analyse fonctionnelle, et sur l'analyse numérique respectivement.

Dans le chapitre 2, on étudiera la classification des équations intégrales, l'existence et l'unicité de VIE's de 2^{nd} espèce, Ensuite nous présentons les définitions et propriétés des fonctions splines polynomiales et les fonctions spline non polynomiales.

Le chapitre 3 est consacré à l'Analyse numérique de VIE's de 2^{nd} espèce par les fonctions spline polynomiales de premier et seconde ordre et par les fonctions spline non polynomiales linéaire et quadratique. On donne des exemples numériques pour illustrer l'efficacité de la méthode. Une comparaison entre la solution exacte et la solution approchée sera présenté par des figures et des tableaux à la fin de chaque exemple suivi par une conclusion générale.

Chapitre 1

Concepts auxilaires

Dans ce chapitre, Nous introduisons plusieurs résultats préliminaires indispensables et des définitions nécessaires qui serons utilisées au long dans ce mémoire. On vise essentiellement à rappeler quelques définitions sur l'analyse fonctionnelle, Ensuite nous présentons les notions d'analyse numérique.

1.1 Rappels d'analyse fonctionnelle

1.1.1 Notions d'analyse fonctionnelle

Définition 1.1.1 *(**Espace vectoriel normé**): Une norme sur un espace vectoriel E est un application qui à tout vecteurs x associe un réel $\|x\|$ vérifiant :*

1. (positivité) $\|x\| \geq 0$, $\|x\| = 0$ si et seulement si x est le vecteur nul.
2. (homogénéité) $\|\alpha x\| = |\alpha| \, \|x\|$, $\alpha \in \mathbb{k}$, $x \in E$.
3. (inégalité triangulaire) $\|x + y\| \leq \|x\| + \|y\|$ x, $y \in E$.

Un espace E muni d'une norme est dit espace normé. Un vecteur de norme 1 est dit unitaire.

Définition 1.1.2 *(**Les espaces** $C^k(\Omega)$): Soit Ω un ouvert de $\mathbb{R}^n$. On désignera par $C^0(\Omega)$ (resp.$C^1(\Omega)$) l'espace des fonctions continues (resp. continûment différentiables) sur Ω à*

valeurs complexes puis, pour $k \in \mathbb{N}$, $k \geq 2$, *on pose*

$$C^k(\Omega) = \{u \in C^{k-1}(\Omega) : \frac{\partial u}{\partial x_i} \in C^{k-1}(\Omega), i = 1, ..., n\}.$$

C'est l'espace des fonctions k fois continûment différentiables sur Ω à valeurs dans $\mathbb{C}$. On notera

$$C^\infty(\Omega) = \bigcap_{k \in \mathbb{N}} C^k(\Omega)$$

l'espace des fonctions indéfinirnent différentiables sur Ω.

Exemple 1.1.1 *Sur des fonctions continues* $C([a,b])$ *sur l'intervalle borné* $[a,b]$ *, les fonctions suivantes:*

$\|f\|_1 = \int_a^b |f(t)|\, dt$, $\|f\|_\infty = \sup_{t \in [a,b]} |f(t)|$ définissent une norme.

Définition 1.1.3 ***(Espace de Banach)****: la suite* $(\varphi_n)_{n \in \mathbb{N}}$ *de vecteurs de* E *est dite de Cauchy si pour tout* $\varepsilon > 0$*, il existe,* $N > 0$*, tel que* $\|\varphi_p - \varphi_q\| < \varepsilon$*, pour* $p, q > N$*.*

l'espace normé $(E, \|.\|)$ est dit Complet ou de Banach si tout suite de Cauchy de E est convergente dans E.

Remarque 1.1.1 *:tout espace vectoriel normé et complet est dit de Banach.*

Définition 1.1.4 ***(Produit Scalaire)****: On appelle produit scalaire sur un espace vectoriel* E *(réel ou complex), une fonction* $\langle x, y \rangle$ *définie dans* $\Bbbk = \mathbb{R}$ *ou* $\mathbb{C}$ *, possédant les propriétés suivantes:*

$$\begin{aligned}
&\langle x, y \rangle \geq 0 \\
&\langle \lambda x, y \rangle = \lambda \langle x, y \rangle \\
&\langle x, y + z \rangle = \langle x, y \rangle + \langle x, z \rangle \\
&\langle x, y \rangle = \overline{\langle y, x \rangle} \\
&\langle x, x \rangle = 0 \Longrightarrow x = 0
\end{aligned}$$

la définition du produit scalaire nous donne les relations suivantes:

$.\langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$.

$.\langle x, \lambda y \rangle = \overline{\lambda} \langle x, y \rangle$.

Définition 1.1.5 *(**Espace Euclidien**) : un espace vectoriel E muni d'un produit scalaire est dit espace euclidien ou préhilbertien, on introduit la norme définie par*

$$\|x\| = \sqrt{\langle x, x\rangle}.$$

Définition 1.1.6 *(**Espace de Hilbert**): un espace de Hilbert H est un espace vectoriel induit par le produit scalaire, de tel sorte qu'il est complet par rapport à la norme $\|.\| = \sqrt{\langle x, x\rangle}$.*

Définition 1.1.7 *(**Espace** $L^2(a,b)$): On dit qu'une fonction f est de carré intégrable sur $[a,b]$ si l'intégrale*

$$\int_a^b f^2(x)dx.$$

existe (est finie). L'ensemble de toutes les fonctions carré intégrable sur $[a,b]$ sera noté $L^2(a,b)$ ou L^2. On muni L^2 du produit scalaire définie par :

$$\langle f, g\rangle = \int_a^b f(x)\ g(x)dx \qquad f, g \in L^2$$

On définie la norme d'une fonction f sur L^2 par:

$$\|f\| = \sqrt{(f,f)} = \sqrt{\int_a^b f^2(x)dx}.$$

1.1.2 Les opérateurs

Opérateur Intégrale linéaire

Définition 1.1.8 *Soit $K : C[a,b] \times C[a,b] \to \mathbb{R}$ une fonction continue, l'opérateur intégrale linéaire sur $C[a,b]$ est défini par la formule suivante:*

$$\begin{aligned} A &: \varphi \in C[a,b] \to A\varphi \in C[a,b] \\ A(\varphi)(x) &= \int_G K(x,t)\varphi(t)dt\ . \end{aligned}$$

Où : K s'appelle le noyau de l'opérateur intégral A .

Opérateur Adjoint

Théorème 1.1.1 *Soit A un opérateur intégrale de noyau $K(x,t)$, Alors l'adjoint A^* est un opérateur intégrale de noyau $K^*(x,t)$, tel que :*

$$K^*(x,t) = K(t,x)$$

Théorème 1.1.2 *Soit un opérateur intégral A défini à partir d'un noyau K continu sur $[a,b]\times[a,b]$ par la formule:*

$$A\varphi(x) = \int_a^b K(x,t)\varphi(t)dt \ , \ x \in [a,b]$$

Alors l'opérateur A admet un unique opérateur adjoint A^* pour le produit scalaire usuel de $L^2([a,b])$, défini par :

$$\forall x \in [a,b] \quad A^*\Psi(x) \ = \int_a^b K(x,t)\Psi(t)dt \ .$$

1.2 Notions d'analyse numérique

1.2.1 Intégration Numérique

Soit $f : [a,b] \to \mathbb{R}$ une fonction continue donnée sur un intervalle $[a,b]$, nous voulons calculer numériquement la quantité

$$\int_a^b f(x)dx.$$

Pour ce faire nous commençons par partitionner l'intervalle d'intégration en N intervalle de même longueur $[x_i, x_{i+1}]$, $i = 1, ..., N$ tels que:

$$a =< x_1 < .. < x_{N-1} < x_N = b.$$

Où

$$x_i = a + h(i-1), i = 1, ..., N$$

avec $h = \dfrac{b-a}{N}$

Alors

$$\int_a^b f(x)dx = \sum_{i=1}^{i=N-1} \int_{x_i}^{x_{i+1}} f(x)dx \ .$$

Il existe plusieurs méthodes pour calculer le second membre de cette égalité, parmi elles: la méthode du trapèze, la méthode de Simpson, la méthode du Bool et Weddle , Cubic Spline, Integration du Romberg, la formule d'intégration du Newton-Cotes. Pour plus de détailles (voir [17]).

1.2.2 L'interpolation

Interpolation polynomiale

Soit P_n un sous espace de $C([a,b])$ de dimension $(n+1)$ engendré par:
$\{1, x, x^2, ..., x^n\}$ et soit $x_0, x_1, ..., x_n \in [a,b]$, on considère les $(n+1)$ paire (x_i, φ_i).
Le problème consiste à trouver un polynôme $p_n \in P_n$ telle que:

$$p_n(x_i) = c_0 + c_1 x_i + c_2 x_i^2 + ... + c_n x_i^n = \varphi_i \ \ , i = 0, 1, ..., n \ .$$

Les points x_i , sont appelées les noeuds d'interpolation.

Il existe plusieurs méthodes pour determiner l'interpolation, parmi ces méthodes la méthode d'interpolation de Lagrange, de Neville, La thechnique des différences divisées de Newton. (Voir [7]).

Théorème 1.2.1 *Soit $(n+1)$ noeuds distincts $x_0, x_1, ..., x_n$ de $[a,b]$, et $(n+1)$valeurs correspondantes $\varphi_0, \varphi_1, ..., \varphi_n$, Alors il existe un polynôme unique $p_n \in P_n$ telle que*

$$p_n(x_i) = \varphi_i \ \ , i = 0, 1, ..., n \ .$$

A quoi l'interpolation sert-elle ?

On sait bien qu'il est facile d'évaluer un polynôme quand on connaît ses coefficients. En tant qu'outil théorique, l'interpolation résout le problème inverse et montre qu'un polynôme de degré n peut aussi bien se représenter par ses $n+1$ coefficients que par ses valeurs en $n+1$ points quelconques.

Cette technique est la base des algorithmes les plus rapides connus pour la multiplication de polynômes. L'interpolation apparaît aussi comme sous-problème dans de nombreux problèmes plus appliqués. Par exemple, pour intégrer une expression compliquée, il suffit de l'évaluer en $n+1$ points, d'approximer l'expression par un polynôme de degré n (interpolation) et d'intégrer ce polynôme. Cette technique est la base des schémas d'intégration numériques.

Erreur d'interpolation

Soit $x_0, x_1, ..., x_n \in [a,b]$, avec $\varphi \in C^{(n)}([a,b])$, et si $\varphi^{(n+1)}$existe dans $]a,b[$. Alors il existe $\zeta_x \in]a,b[$ dépend de x, telle que:

$$E_n(x) = \varphi(x) - p_n(x) = F(x)\frac{\varphi^{(n+1)}\zeta_x}{(n+1)!}. \qquad (*)$$

Où $F(x) = \prod_{i=0}^{n}(x - x_i)$

Preuve. Si $x = x_i$, alors $E_n(x) = 0$ et l'égalité (*) est trivialement vérifiée ; Supposons que $x \neq x_i \ \forall i = 0, 1, ..., n$ et considérons pour x fixé la fonction g définie par:

$$g(t) = E_n(t) - \frac{F(t)}{F(x)}E_n(x).$$

La fonction $g \in C^{(n+1)}([a,b])$ et s'annule en $(n+2)$ points distincts $x, x_0, x_1, ..., x_n$. Le théorème de Rolle montre que g' admet au moins $(n+1)$ racines dans I .

D'où, en procédant par récurrence sur l'ordre de dérivation de g , la fonction $g^{(n+1)}$ admet au moins une racine dans I .

Soit ζ_x cette racine. On a :

$$0 = g^{(n+1)}(\zeta_x) = \varphi^{(n+1)}(\zeta_x) - \frac{(n+1)!}{F(x)}E_n(x)$$

D'où

$$E_n(x) = F(x)\frac{\varphi^{(n+1)}\zeta_x}{(n+1)!}$$

■

Chapitre 2

Equation intégrale de Volterra et Les Fonctions Splines

Introduction

Les équations intégrales apparaissent de manière naturelle au cours de l'obtention de la solution mathématique à des problèmes aux limites mixtes de la physique mathématique. De nombreuses approches possibles pour réduire un problème de frontière mixte donné à une équation intégrale.

L'équation intégrale de Volterra présentée par Vito Volterra (1860-1940), dans sa célèbre théorie de l'équation intégrale.

Ce chapitre est organisé comme suit: En section (2.1), nous étudions la classification des équations intégrales et quelques concepts de base sont donnés, la théorie mathématique de l'existence et unicité de la solution de VIE's de 2^{nd} espèce seront considérés, En section (2.2), nous définissons quelques types des fonctions splines polynomiales, les fonctions splines classiques, et enfin les fonctions splines polynomiales et non polynomiales qui seront utilisées dans ce mémoire.

2.1 Equation intégrale de Volterra

2.1.1 Classification des équations intégrales

Une équation intégrale est une équation dans la fonction inconnue $u(x)$ apparaît sous le signe d'intégration. Le plus standard type de l'équation intégrale en $u(x)$ est de la forme:

$$u(x) = f(x) + \lambda \int_{a(x)}^{b(x)} k(x,t,u(t))dt \tag{2.1}$$

Où $a(x)$ et $b(x)$ sont les limites d'intégration, λ est un paramètre constant et $k(x,t)$ est une fonction connue, appelée le noyau de l'équation intégrale, la fonction inconnue $u(x)$ qui sera déterminé apparaît sous le signe d'intégration.

Dans d'autres cas, la fonction inconnue $u(x)$ apparaît à l'intérieur et à l'extérieur du signe d'intégration. les fonctions $f(x)$ et $k(x,t)$ sont données.

Il est à noter que les bornes d'intégration déterminées comme a(x) et b(x) peuvent être toutes les deux des variables, constantes, ou mixtes.

La classification des équations intégrales dépend de plusieurs caractéristiques.

La première est la linéarité du noyau $k(x,t,u(t))$ par rapport à la troisième variable.

1. si $k(x,t,u(t))$ est linéaire par rapport à la troisième variable i.e

$$k(x,t,u(t)) = k(x,t)u(t) \tag{2.2}$$

l'équation intégrale est appelée équation intégrale linéaire.

2. si $k(x,t,u(t))$ est non linéaire par rapport à la troisième variable i.e

si l'équation contient la fonction non linéaire $u(x)$, l'équation intégrale est appelée équation intégrale non linéaire.

deux autres cas distincts qui dépend des bornes d'intégration sont utilisés pour caractériser les équations intégrales, appelées:

1. **Equation intégrale de Fredholm** si les bornes d'intégration sont constants, donnée par la forme:

$$u(x) = f(x) + \lambda \int_a^b k(x,t)u(t))dt \tag{2.3}$$

Oû, a et b sont constantes.

2. si au moins une limite d'intégration est variable, l'équation intégrale est appelée **équation intégrale de Volterra** est donnée par:

$$u(x) = f(x) + \lambda \int_a^x k(x,t)u(t))dt \tag{2.4}$$

En plus, les deux autres façons distinctes, qui dépend de la fonction inconnue $u(x)$, sont définis comme suit:

1. si la la fonction inconnue $u(x)$ apparaît seulement sous le signe d'intégration de l'équation de Fredholm ou de Volterra, l'équation intégrale est appelée l'équation intégrale de Fredholm ou Volterra de première espèce respectivement.

2. si la la fonction inconnue $u(x)$ apparaît à l'intérieur et à l'extérieur de le signe d'intégration de l'équation Fredholm ou Volterra, l'équation intégrale est appelée l'équation intégrale de Fredholm ou de Volterra de seconde espèce respectivement.

dans toutes les équations intégrales de Fredholm ou de Volterra présentées ci-dessus, si $f(x)$ est égal zéro, l'équation résultante:

$$u(x) = f(x) + \lambda \int_a^b k(x,t)u(t)dt \tag{2.5}$$

$$u(x) = f(x) + \lambda \int_a^x k(x,t)u(t)dt \tag{2.6}$$

est appelée l'équation intégrale homogène de Fredholm ou de Volterra respectivement.

Maitenant nous résumons les types d'équations intégrales comme suit.

équation intégrales de Fredholm

Pour les équations intégrales de Fredholm, les limites d'intégration sont constants. En plus, la fonction inconnue $u(x)$ apparaît seulement à l'intérieur d'équation intégrale est de la forme:

$$f(x) = \int_a^b k(x,t)u(t)dt \tag{2.7}$$

est appelée l'équation intégrale de Fredholm de première espèce. Cependant, pour les équations intégrales de Fredholm de seconde espèce, la fonction inconnue $u(x)$ apparaît à l'intérieur et à l'extérieur du signe d'intégration, cette équation est représenté par :

$$u(x) = f(x) + \lambda \int_a^b k(x,t)u(t)dt \tag{2.8}$$

équation intégrale de Volterra

Pour les équations intégrales de Volterra, au moins une limite d'intégration est variable. Pour les équations intégrales de Volterra de première espèce, la fonction inconnue $u(x)$ apparaît seulement sous le signe d'intégration est de la forme:

$$f(x) = \int_a^x k(x,t)u(t)dt \tag{2.9}$$

Cependant, les équation intégrales de Volterra de seconde espèce, la fonction inconnue $u(x)$ apparaît à l'intérieur et à l'extérieur du signe d'intégration, prend la forme:

$$u(x) = f(x) + \lambda \int_a^x k(x,t)u(t)dt \tag{2.10}$$

Remarque 2.1.1 *l'équation intégrale deVolterra est un cas particulier de l'équation intégrale de Fredholm, Il suffit de prendre le noyau k vérifie la condition:*

$$k(x,t) = 0, \text{ pour } x < t. \tag{2.11}$$

Equation intégrale singulière

les équations intégrales de Volterra de première espèce

$$f(x) = \lambda \int_{a(x)}^{b(x)} k(x,t)u(t)dt \tag{2.12}$$

ou de seconde espèce

$$u(x) = f(x) + \lambda \int_{a(x)}^{b(x)} k(x,t)u(t)dt \tag{2.13}$$

sont appelées singulières si l'une des limites d'intégration $a(x), b(x)$ ou les toutes les deux sont infinies. En plus, les deux équations sont appelées singuliéres.

si le noyau $k(x,t)$ devient illimité à un ou plusieurs points de l'intervalle d'intégration, la forme la plus répandue est:

$$f(x) = \int_0^x \frac{1}{(x-t)^\alpha} u(t)dt \qquad 0 < \alpha < 1 \tag{2.14}$$

laquelle est de première espèce et elle est appelée équation intégrale d'Abel généralisée, l'équation de seconde espèce est de la forme:

$$u(x) = f(x) + \int_0^x \frac{1}{(x-t)^\alpha} u(t)dt \qquad 0 < \alpha < 1 \tag{2.15}$$

est appelée une équation intégrale faiblement singulière.

Dans l'équation intégrale d'Abel généralisée, si $\alpha = \frac{1}{2}$ l'équation devient l'équation intégrale d'Abel singulière. la singularité de ce type d'équation se produit à partir de la limite supérieur lorsque $t = x$

Remarque 2.1.2 *l'équation intégrale d'Abel est souvent deduite du problèmes mécaniques*

Equation intégrales du type convolution

Sont des équations intégrales linéaires dont le noyau $k(x,t)$ dépendant que la différence des arguments.i.e.

$$k(x,t) = \ k(x-t) \tag{2.16}$$

Donc l'équation intégrale de type convolution est de la forme suivante:

$$u(x) = f(x) + \lambda \int_0^x k(x-t)u(t))dt \tag{2.17}$$

cette classe d'équation comprend par exemple l'équation d'Abel généralisée.

Equations intégrales à noyau dégénéré:

Le noyau d'une équation intégrale linéaire est dit dégénéré s'il est la somme d'un nombre fini de produits de fonctions de x par des fonctions de t .i.e.

$$k(x,t) = \sum_{j=1}^{n} \ a_j(x) \ \ bj(t) \tag{2.18}$$

les fonctions $a_j(x)$ et $b_j(t)$ seront supposées continues dans $[a,b] \times [a,b]$, et linéairement indépendantes.

Dans ce mémoire, on considère le problème suivant:

l'équation intégrales de Volterra de seconde espèce avec $\lambda = 1$, c.à.d

$$u(x) = f(x) + \int_a^x k(x,t)u(t)dt \quad , x \in [a,b] \tag{2.19}$$

2.1.2 Existence et Unicité

On considére l'équation

$$u(x) = f(x) + \int_0^x k(x,t)u(t)dt \tag{2.20}$$

avec $f : [0,\alpha] \rightarrow \mathbb{R}^n$ fonction continue et $k(x,t)$ est une matrice $n \times n$ des fonction continues telles que $0 \leq x \leq \alpha$ et $\alpha \leq \infty$.

Lemme 2.1.1 ***(Granwall)***

Soit $f, g : [0, \alpha] \to [0, \infty]$ deux fonctions continues et soit c un nombre positif.

$$f(t) \leq c + \int_0^t g(s)\ f(s)ds \qquad , 0 \leq t \leq \alpha \tag{2.21}$$

Alors

$$f(t) \leq c + \exp \int_0^t g(s)\ ds \qquad , 0 \leq t \leq \alpha \tag{2.22}$$

Théorème 2.1.1 *Soit $0 < \alpha \leq \infty$, on suppose que $f : [0, \alpha] \to \mathbb{R}$ une fonction continue et $k(x, t)$ et un matrice $n \times n$ des fonctions telle que $0 \leq t \leq x \leq \alpha$, si $0 < t < \alpha$, alors il existe une unique solution $u(x)$ pour l'équation intégrale de Volterra suivante:*

$$u(x) = f(x) + \int_0^x k(x, t)u(t)dt \ , \ x \in [0, T] \tag{2.23}$$

Preuve.

Soit l'équation intégrale de Volterra

$$u(x) = f(x) + \int_0^x k(x, t)u(t)dt$$

On définie la suite $(\Psi_n(x))$ sur l'intervalle $[0, T]$ par la méthode d'approximation successive, on obtient:

$$\Psi_0(x) = f(x)$$

$$\begin{aligned} \Psi_1(x) &= \int_0^x k(x, t)\Psi_0(t)dt \\ &\vdots \end{aligned}$$

$$\Psi_{i+1}(x) = \int_0^x k(x, t)\Psi_i(t)dt$$

$$K = \max_{0 \leq t \leq T} k(x, t)$$

$$F = \max_{0 \leq t \leq T} \left| f\,|\Psi_1(x)| \leq \int_0^x |k(x, t)\ \Psi_0(t)|dt \ \leq \ K\ F\ x(x) \right|$$

$$|\Psi_2(x)| \leq \int_0^x |k(x,t)\Psi_1(t)|dt \leq \frac{1}{2}\ K^2 F x^2$$

$$\vdots$$

$$|\Psi_n(x)| \leq \int_0^x |k(x,t)\Psi_{n-1}(t)|dt \leq \frac{F(K\ x)^n}{n\ !}.$$

par récurrence la relation précédente est vraie $\forall\ n \in \mathbb{N}$, on a:

$$\sum_{i=0}^{\infty} \frac{F(K\ x)^n}{n\ !} = F\ e^{\ Kx}$$

est une série de Taylor qui converge uniformément et absolument sur $[0,T]$, Alors la suite $u_n(x) = \sum_{i=0}^{\infty}\ \Psi_i(x)$ converge vers $u(x) = \sum_{i=0}^{n}\ \Psi_i(x)$

$$\int_0^x k(x,t)u(t)dt = \int_0^x k(x,t)\ (\sum_{i=0}^{n}\ \Psi_i(t)dt\ )$$

$$= \sum_{i=0}^{n}(\int_0^x k(x,t)\ \Psi_i(t)dt)$$

$$= \sum_{i=0}^{n} \Psi_{i+1}(x)$$
$$= u(x) - f(x)$$

Donc il existe une solution $u(x)$ telle que

$$u(x) = f(x) + \int_0^x k(x,t)u(t)dt \tag{2.24}$$

Nous pouvons montrer l'unicité, on suppose qu'il existe deux solutions $X(x)$ et$Y(x)$ de l'équation (2.24), donc

$$X(x) - Y(x) = \int_0^x k(x,t)(X(t) - Y(t))dt$$

On obtient donc la suite suivante:

$$|X(x) - Y(x)\ | \leq\ k\int_0^x |X(t) - Y(t)|dt$$

cette relation est de la forme suivante

$$Z(x) \leq k \int_0^x |Z(t)|dt$$

telle que $Z(x) = X(x) - Y(x)$, on a $\forall c > 0$ la relation

$$Z(x) \leq c + k \int_0^x |Z(t)|dt$$

la relation précédente est vraie pour $c = 0$ telle c est une constante, d'après le lemme de Granwall

$$|Z(x)| \leq ce^{\,Kx} = 0$$

puisque $c = 0$, donc

$$\begin{aligned} Z(x) &= X(x) - Y(x) = 0 \\ X(x) &= Y(x) \end{aligned}$$

Alors la solution de l'équation intégrale (2.24) est unique. ■

2.2 Les Fonctions Spline

2.2.1 Introduction

Un polynôme est une expression mathématique contenant une somme de puissances d'une ou plusieurs variables multipliée par des coefficients. Les polynômes ont longtemps été utilisés pour approximer des fonctions, principalement à cause de leurs propriétés mathématiques simples. Cependant, il est bien connu que les polynômes de degré élevé ont tendance à osciller fortement et qu'ils sont susceptibles de produire une très mauvaise approximation.

Les fonctions splines sont des polynômes par morceaux de degré n qui se joignent au niveau des points de discontinités. Ces polynômes possèdent $n - 1$ dérivées continues. Les points de discontinités des splines sont appelés noeuds. Avec les fonctions splines, nous combinons un polynôme de faible degré et donc faiblement oscillant de manière à obtenir une fonction aussi lisse que possible au sens où elle possède des intervalles de continuité

maximaux sans être altérée globalement un polynôme. Les fonctions splines peuvent être intégrées et différenciées car elles sont des polynômes par morceaux.

Une fonction spline non polynomiale par morceaux est une combinaison de fonctions de base trigonométriques et polynomiales, qui forment un espace de Chebyshev complet. Cette approche assure une précision améliorée et une forme générale à la fonction spline ex istante.

$$s_n = span\{1, x, x^2, ..., x^n\}.$$

Une nouvelle base, appelée la base de c-Bezier, est construire pour l'espace

$$\lceil(n) = span\{1, x, x^2, ..., x^{n-2}, \cos x, \sin x\}.$$

dans lequel x^{n-1} et x^n ont été remplacé par $\cos x$ $\sin x$

2.2.2 Quelques types des fonctions spline polynomiales:

Définition 2.2.1 *Une fonction S est appelée une spline de degré k si:*

1. le domain de S est un intervalle $[a, b]$,
2. $S \in C^{k-1}[a, b]$.
3. Il existe t_i (les noeuds de S) t.q $a = x_0 < x_1 < x_2... < x\ _n = b$ et telle que S est un polynome de degré au plus k sur chaque sous-intervalle $[x_i, x_{i+1}]$.

Définition 2.2.2 *Une fonction spline est appelée* ***spline naturelle*** *s'il satisfait une autre condition suivant:*

$$S_k^m(x_0) = S_k^m(x_n) = 0$$

Il existe d'autres types des fonctions spline polynomiales par exemple:
M –spline, L-spline, G-spline, P-spline , B-spline,. . . , etc.
Maintenant, on définié quelques types de fonctions spline nécessite pour ce travail.

2.2.3 Fonction spline classique:

Fonction spline classique linéaire

Définition 2.2.3 *Une fonction L est appelée* ***spline linéaire*** *si il satisfait :*

1. Il existe un partion de l'intervalle $a = x_0 < ... < x_n = b$, telle que L est un polynomiale de degré 1 sur chaque sous-intervalle $[x_i, x_{i+1}]$.

2. L est continue sur $[a, b]$, i.e.

$$L(x) = \begin{cases} l_0(x) \ , x \in [x_0, x_1] \\ l_1(x), x \in [x_1, x_2] \\ \quad . \\ \quad . \\ l_{n-1}(x), x \in [x_{n-1}, x_n] \end{cases} \tag{2.25}$$

Où $x_0, x_1, ..., x_n$ sont appelées noeuds , et chaque morceau de $L(x)$ est de la forme:

$$l_i = a_i x + b_i \tag{2.26}$$

Où a_i, b_i sont des coefficientes de la fonction spline linéaire (2.26).

Fonction spline classique quadratique

Définition 2.2.4 *Une fonction Q est appelée* ***spline quadratique*** *si il satisfait :*

1. Q ,Q' sont continues sur $[a, b]$.

2. Q est un polynomiale de degré au plus 2 sur chaque sous-intervalle $[x_i, x_{i+1}]$,

Où $a = x_0 < ... < x_n = b$.
la fonction spline quadratique est de la forme:

$$q_i = a_i + b_i(x - x_i) + c_i(x - x_i)^2 \tag{2.27}$$

Fonction spline classique cubique

Définition 2.2.5 *Une fonction S est appelée une spline cubique si elle satisfait:*

1. $S_i(x), S_i'(x)$ et $S_i''(x)$ sont continues sur $[a, b]$
2. $S_i(x)$ est au plus cubique sur chaque sous-intervalle $[x_i, x_{i+1}]$, ou $a = x_0 < ... < x_n = b$

la fonction spline cubique est de la forme:

$S_i = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$

Les x_i sont les nœuds et sont supposés être classés par ordre croissant. La fonction spline S, que nous construisons, consiste en n polynomes cubiques par morceaux :

$$S(x) = \begin{cases} s_0(x) \ , x \in [x_0, x_1] \\ s_1(x), x \in [x_1, x_2] \\ \quad . \\ \quad . \\ s_{n-1}(x), x \in [x_{n-1}, x_n] \end{cases}$$

Ici s_i désigne le polynôme cubique qui sera utilisé sur le sous-intervalle $[x_i, x_{i+1}]$. La condition d'interpolation es:

$s_i(x) = u_i \ 0 \leq i \leq n$

2.2.4 Fonction spline polynomiale

On consider la partition $\triangle = \{t_0, t_1, t_2, .., t_n\}$ de $[a, b] \subset \mathbb{R}$.Soit $S(\triangle)$ noté l'ensemble des polynomes par morcaux sur le sous-intervalle $I_i = [t_i, t_{i+1}]$ de partition $\triangle$. Soit $u(t)$ est la solution exacte, chaque spline polynomiale de l'ordre n, on a $p_i(t)$ est de la forme:

$$p_i(t) = a_i(t - t_i) + b_i(t - t_i)^2 + ... + y_i(t - t_i)^{n-1} + z_i \tag{A.1}$$

Où $a_i, b_i, ...$et z_i des constantes.

La fonction spline polynomiale de premier ordre

La forme de la fonction spline polynomiale de premier ordre est :

$$p_i(t) = a_i(t - t_i) + b_i \quad i = 0, 1, ..., n \tag{A.2}$$

Où a_i,et b_i sont des constantes. Pour obtenir la valeur de a_i et b_i , on dériviée l'équation (A.2) on aura:

$$p_i'(t) = a_i \tag{A.3}$$

d'où en remplaçant t par t_i dans la relation (A.2) et (A.3) on obtient:

$p_i(t_i) = b_i$

$p_i'(t_i) = a_i$

dans l'équations ci-dessus, on a la valeur de a_i et b_i comme suit:

$$a_i = p_i'(t_i) \tag{A.4}$$

$$b_i = p_i(t_i) \qquad \text{pour} \quad i = 0, 1, ..., n \tag{A.5}$$

La fonction spline polynomiale de seconde ordre

La forme de spline polynomiale de seconde ordre est:

$$p_i(t) = a_i(t - t_i)^2 + b_i(t - t_i) + c_i \tag{A.6}$$

Où a_i, b_i et c_i sont des constantes des fonctions polynomiales . On a les relations suivantes:

$p_i(t_i) = c_i = u(t_i)$

$p_i'(t_i) = b_i = u'(t_i)$

$p_i''(t_i) = 2a_i = u''(t_i)$

Par substitution on obtient les valeur de a_i, b_i et c_i comme suit:

$$a_i = \frac{1}{2}u''(t_i) \tag{A.7}$$

$$b_i = u'(t_i) \tag{A.8}$$

$$c_i = u(t_i) \tag{A.9}$$

pour $i = 0, 1, ..., n$

2.2.5 Fonction spline non polynomiale

On considère la partition $\triangle = \{t_0, t_1, t_2, .., t_n\}$ de $[a, b] \subset \mathbb{R}$.Soit $S(\triangle)$notée l'ensemble des polynomes par morcaux sur chaque sous-intervalle $I_i = [t_i, t_{i+1}]$ de partition $\triangle$.Soit $u(t)$ est la solution exacte, cette nouvelle méthode fournit une approximation non seulement l'approximation de $u(t_i)$ dans les noeuds, mais aussi pour $u^n(t_i), n = 1, 2, ..$à chaque point de l'intervalle d'intégration aussi, la C^∞ différentiabilité de la partie trigonometrique des fonctions splines non-polynomiales compense la perte de lissage produite par les polynomes. Chaque fonction spline non polynomiale d'ordre n , $p_i(t)$ est de la forme:

$$p_i(t) = a_i \cos k(t - t_i) + b_i \sin k(t - t_i) + ... + y_i(t - t_i)^{n-1} + z_i \tag{B.1}$$

Où $a_i, b_i, .., y_i$ et z_i des constanets et k est de la fréquence de la fonction trigonométrique qui est utilisée pour augmenter la précision de la méthode.

La fonction spline non polynomiale linéaire

La forme de la fonction spline non-polynomiale linéaire est :

$$p_i(t) = a_i \cos k(t - t_i) + b_i \sin k(t - t_i) + c_i(t - t_i) + d_i \quad i = 0, 1, ..., n \tag{B.2}$$

Où a_i, b_i , c_i et d_i sont des constantes à determinées. Pour obtenir les valeurs de a_i, b_i , c_i et d_i, on dérive l'équation $(B.2)$ trois fois, alors on a:

$$\left.\begin{aligned} p_i'(t) &= -ka_i \sin k(t - t_i) + kb_i \cos k(t - t_i) + c_i \\ p_i''(t) &= -k^2 a_i \cos k(t - t_i) - k^2 b_i \sin k(t - t_i) \\ p_i^{(3)}(t) &= k^3 a_i \sin k(t - t_i) - k^3 b_i \cos k(t - t_i) \end{aligned}\right\} \tag{B.3}$$

d'où on remplace t par t_i dans la relation $(B.2)$et$(B.3)$ on a:

$p_i(t_i) = a_i + d_i$

$p_i'(t_i) = kb_i + c_i$

$p_i''(t_i) = -k^2 a_i$

$p_i^{(3)}(t_i) = -k^3 b_i$

dans l'équations ci-dessus, on a la valeur de a_i, b_i , c_i et d_i comme suit:

$$a_i = -\frac{1}{k^2} p_i''(t_i) \tag{B.4}$$

$$b_i = -\frac{1}{k^3} p_i^{(3)}(t_i) \tag{B.5}$$

$$c_i = p_i'(t_i) + kb_i \tag{B.6}$$

$$d_i = p_i(t_i) + a_i \qquad for\ i = 0, 1, ..., n \tag{B.7}$$

La fonction spline non polynomiale quadratique

La forme de la fonction spline non polynomiale quadratique est :

$$Q_i(t) = a_i \cos k(t - t_i) + b_i \sin k(t - t_i) + c_i(t - t_i) + d_i(t - t_i)^2 + e_i \tag{B.8}$$

Où a_i, b_i ,c_i ,d_i et e_i sont des constantes à determiniées. Pour obtenir les valeurs de a_i,b_i , c_i ,d_i et e_i , on dérivié l'équation $(B.8)$ quatre fois par rapport t, alors on a :

$$\left.\begin{aligned} Q_i'(t) &= -ka_i \sin k(t - t_i) + kb_i \cos k(t - t_i) + c_i + 2d_i(t - t_i) \\ Q_i''(t) &= -k^2 a_i \cos k(t - t_i) - k^2 b_i \sin k(t - t_i) + 2d_i \\ Q_i^{(3)}(t) &= k^3 a_i \sin k(t - t_i) - k^3 b_i \cos k(t - t_i) \\ Q_i^{(4)}(t) &= k^4 a_i \cos k(t - t_i) + k^4 b_i \sin k(t - t_i) \end{aligned}\right\} \tag{B.9}$$

d'où on remplace t par t_i dans la relation $(B.8)$ et $(B.9)$ on a:

$Q_i(t_i) = a_i + e_i$

$Q_i'(t_i) = kb_i + c_i$

$Q_i''(t_i) = -k^2 a_i + 2d_i$
$Q_i^{(3)}(t_i) = -k^3 b_i$
$Q_i^{(4)}(t_i) = k^4 a_i$
On a les valeurs de a_i, b_i , c_i , d_i et e_i :

$$a_i = \frac{1}{k^4} Q_i^{(4)}(t_i) \tag{B.10}$$

$$b_i = -\frac{1}{k^3} Q_i^{(3)}(t_i) \tag{B.11}$$

$$c_i = Q_i'(t_i) + \frac{1}{k^2} Q_i^{(3)}(t_i) \tag{B.12}$$

$$d_i = \frac{1}{2}[Q_i''(t_i) + \frac{1}{k^2} Q_i^{(4)}(t_i)] \tag{B.13}$$

$$e_i = Q_i(t_i) - \frac{1}{k^4} Q_i^{(4)}(t_i) \quad pour\ i = 0, 1, ..., n. \tag{B.14}$$

Chapitre 3

Analyse numérique pour l'équation intégrale de Volterra de seconde espèce

La résolution Analytique de plusieurs types d'équations intégrales est trés difficile de obtenir, alors on essaye de les résoudre numériqument.

Ce chapitre est consacré aux résultats relatifs aux méthodes décritent ci-dessu, notre but et de trouver une solution approchée de VIE's de 2^{nd} espèce

$$u(x) = f(x) + \int_a^x k(x,t)u(t)dt \quad , x \in [a,b] \tag{A.10}$$

Où $u(x)$ et $k(x,t)$ sont des fonctions connues, mais $u(t)$ est inconnue. Ensuite on va faire une comparaison entre la solution exacte et la solution approchée, Enfin on présente une analyse des résultats qui montre l'efficacité de cette thechnique.

3.1 La résolution de l'équation intégrale de Volterra de seconde espèce par les fonctions spline polynomiales

Pour résoudre l'équation (A.10), on dérive l'équation $(A.10)$ deux fois par rapport x, par utilisation de la formule de leibniz, on a:

$$u'(x) = f'(x) + \int_a^x \frac{\partial k(x,t)}{\partial x} u(t)dt + k(x,x)u(x) \tag{A.11}$$

$$\begin{aligned} u''(x) &= f''(x) + \int_a^x \frac{\partial^2 k(x,t)}{\partial x^2} u(t)dt + (\frac{\partial k(x,t)}{\partial x})_{t=x} u(x) + \\ &\quad (\frac{dk(x,t)}{dx})u(x) + k(x,x)u'(x) \end{aligned} \tag{A.12}$$

En substituant $x = a$ dans les équations $(A.10) - (A.12)$, on a:

$$u_0 = u(a) = f(a) \tag{A.13}$$

$$u'_0 = u'(a) = f'(a) + k(a,a)u(a) \tag{A.14}$$

$$\begin{aligned} u''_0 &= u''(a) = f''(a) + ((\frac{\partial k(x,t)}{\partial x})_{t=x})_{x=a} u(a) + (\frac{dk(x,t)}{dx})_{x=a} u(a) \\ &\quad + k(a,a)u'(a) \end{aligned} \tag{A.15}$$

3.1.1 Utilisation des fonctions spline polynomiales de premier ordre

Description de la méthode

On approxime la solution de l'équation intégrale de Volterra de seconde espèce par la fonction spline polynomiale de premier ordre. nous introduisons la méthode de solution dans cet algorithme.

L'Algorithme(VIE2PS1):

étape1: mettre $h = (b-a)/n, t_i = t_0 + ih$, $i = 0, .., n$ (où $t_0 = a, t_n = b$) et $u_0 = f(a)$.

étape2: evalué a_0 et b_0 par substitué (A.13)-(A.14) dans les équations (A.4)-(A.5).

étape3: calcule $p_0(t)$ on utilise l'étape 2 et l'équation (A.2).

étape4: approximant $u_1 = p_0(t_1)$.

étape5: pour $i = 1$à $n-1$ fait les étapes suivantes :

étape6: evalué a_i, b_i, c_i et d_i par l'utilisation des équations (A.4)-(A.5) et remplace $u(t_i), u'(t_i)$ par $p_i(t_i)$ $p_i'(t_i)$.

étape7: calcule $p_i(t)$ on utilise l'étape 6 et l'équation (A.2).

étape8: approximant $u_{i+1} = p_i(t_{i+1})$.

Exemples test

exemple01:

On considére l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = x + \int_0^x (t-x)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = \sin(x)$.

exemple02:

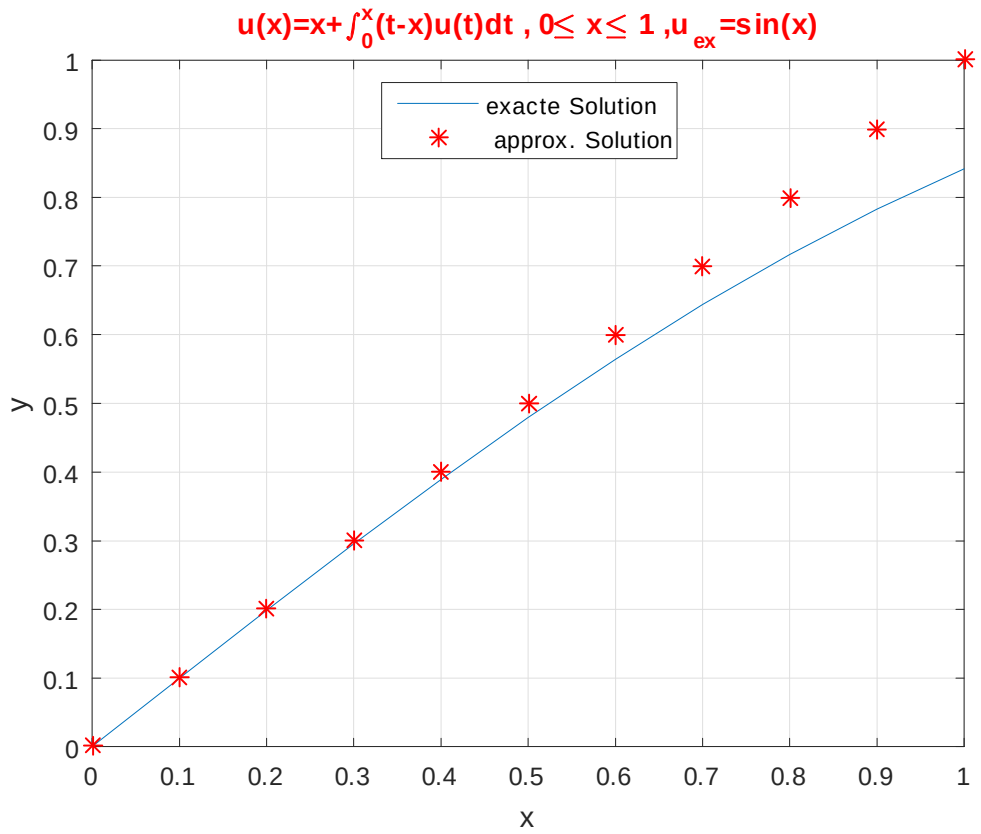
On considére l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = \sin(x) + \int_0^x \sin(x)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = x$.

n	x	sol exacte	sol approchée	Erreur
1	0.00	0.00000000000000	0.00000000000000	$0.00000000000000e+00$
2	0.10	0.09983341664683	0.10000000000000	$1.66583353171851e-04$
3	0.20	0.19866933079506	0.20000000000000	$1.33066920493879e-03$
4	0.30	0.29552020666134	0.30000000000000	$4.47979333866044e-03$
5	0.40	0.38941834230865	0.40000000000000	$1.05816576913495e-02$
6	0.50	0.47942553860420	0.50000000000000	$2.05744613957970e-02$
7	0.60	0.56464247339504	0.60000000000000	$3.535752660449646e-02$
8	0.70	0.64421768723769	0.70000000000000	$5.57823127623089e-02$
9	0.80	0.71735609908995	0.80000000000000	$8.26439091004771e-02$
10	0.90	0.78332269096274	0.90000000000000	$1.16673090372517e-01$
11	1.00	0.84147098480790	1.00000000000000	$1.58529015192103e-01$

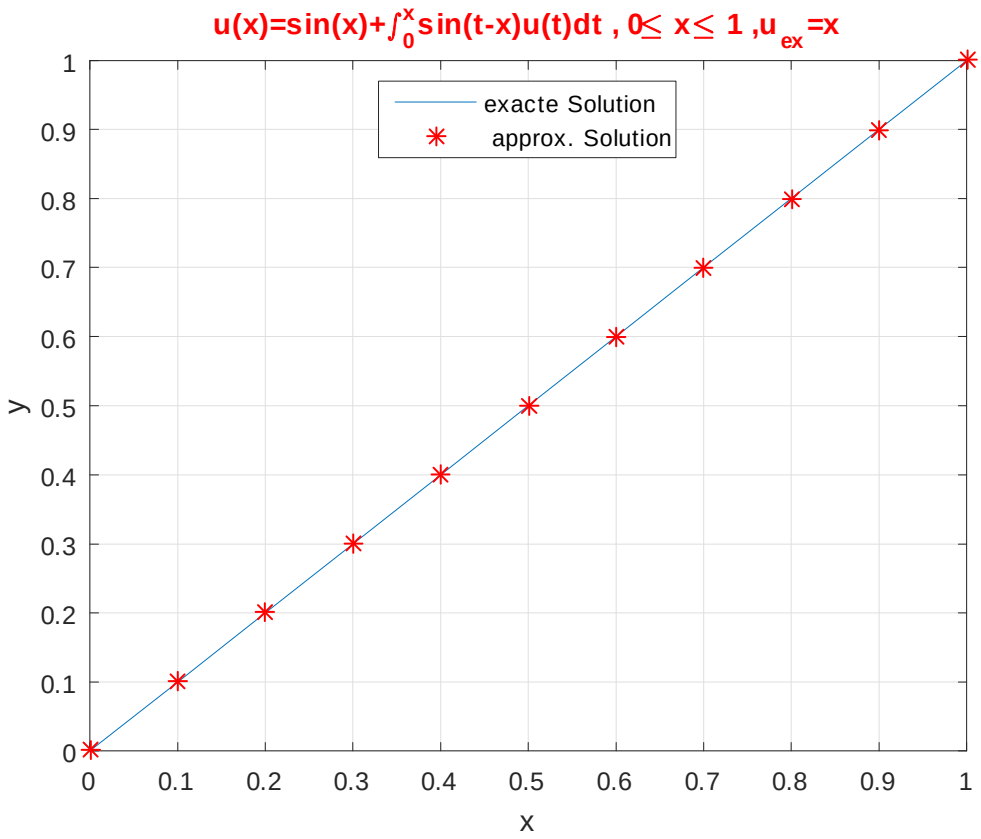
Tableau(1.1) la solution exacte et la solution approchée pour exemple1



Figure(1.1):Comparaison entre la solution exacte et la solution approchée pour l'utilisation des fonctions spline polynomiales de premier ordre pour exemple 1

n	x	sol exacte	sol approchée	Erreur
1	0.00	0.00000	0.00000000000000	$0.00000000000000e+00$
2	0.10	0.100000	0.10000000000000	$0.00000000000000e+00$
3	0.20	0.200000	0.20000000000000	$0.00000000000000e+00$
4	0.30	0.300000	0.30000000000000	$0.00000000000000e+00$
5	0.40	0.400000	0.40000000000000	$0.00000000000000e+00$
6	0.50	0.500000	0.50000000000000	$0.00000000000000e+00$
7	0.60	0.600000	0.60000000000000	$0.00000000000000e+00$
8	0.70	0.700000	0.70000000000000	$0.00000000000000e+00$
9	0.80	0.800000	0.80000000000000	$1.11022302462516e-16$
10	0.90	0.900000	0.90000000000000	$1.11022302462516e-16$
11	1	1.000000	1.00000000000000	$1.11022302462516e-16$

Tableau(1.2) la solution exacte et la solution approchée pour exemple2



Figure(1.2):Comparaison entre la solution exacte et la solution approchée pour l'utilisation des fonctions spline polynomiales de premier ordre pour exemple 2

3.1.2 Utilisation des fonctions spline polynomiales de seconde ordre

Description de la méthode

On approximier la solution de l'équation intégrale de Volterra de seconde espèce par la fonction spline polynomiale de seconde ordre . nous introduisons la méthode de solution dans l'algorithme.

L'Algorithme(VIE2PS2):

étape1: mettre $h = (b-a)/n, t_i = t_0 + ih$,$i = 0, .., n$ (où $t_0 = a, t_n = b$) et $u_0 = f(a)$.

étape2: evalué a_0 ,b_0 et c_0 par substitué (A.13)-(A.15) dans les équations (A.7)-(A.9)

étape3: calcule $p_0(t)$ on utilise l'étape 2 et l'équation (A.6).

étape4: approximant $u_1 = p_0(t_1)$.

étape5: pour $i = 1$ à $n - 1$ fait les étapes suivantes :

étape6: evalué a_i, b_i, c_i et d_i par l'utilisation des équations (A.7)-(A.9) et remplace $u(t_i), u'(t_i)$ et $u''(t_i)$ par $p_i(t_i)$, $p'_i(t_i)$ et $p''_i(t_i)$.

étape7: calcule $p_i(t)$ on utilise l'étape 6 et l'équation (A.6).

étape8: approximant $u_{i+1} = p_i(t_{i+1})$.

Exemples Test

exemple01:

On considère l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = x + \int_0^x (t - x)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = \sin(x)$.

exemple02:

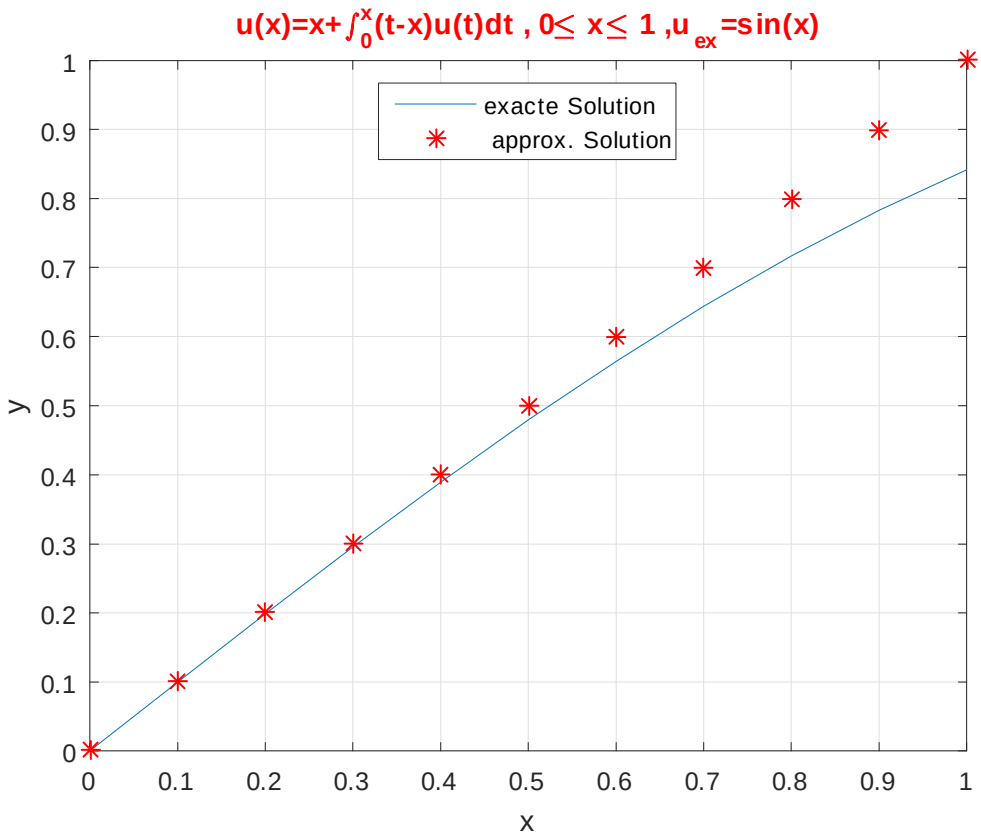
On considére l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = \sin(x) + \int_0^x \sin(x)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = x$.

n	x	sol exacte	sol approchée	Erreur
1	0.00	0.00000000000000	0.00000000000000	$0.00000000000000e+00$
2	0.10	0.09983341664683	0.10000000000000	$1.66583353171851e-04$
3	0.20	0.19866933079506	0.20000000000000	$1.33066920493879e-03$
4	0.30	0.29552020666134	0.30000000000000	$4.47979333866044e-03$
5	0.40	0.38941834230865	0.40000000000000	$1.05816576913495e-02$
6	0.50	0.47942553860420	0.50000000000000	$2.05744613957970e-02$
7	0.60	0.56464247339504	0.60000000000000	$3.535752660449646e-02$
8	0.70	0.64421768723769	0.70000000000000	$5.57823127623089e-02$
9	0.80	0.71735609908995	0.80000000000000	$8.26439091004771e-02$
10	0.90	0.78332269096274	0.90000000000000	$1.16673090372517e-01$
11	1.00	0.84147098480790	1.00000000000000	$1.58529015192103e-01$

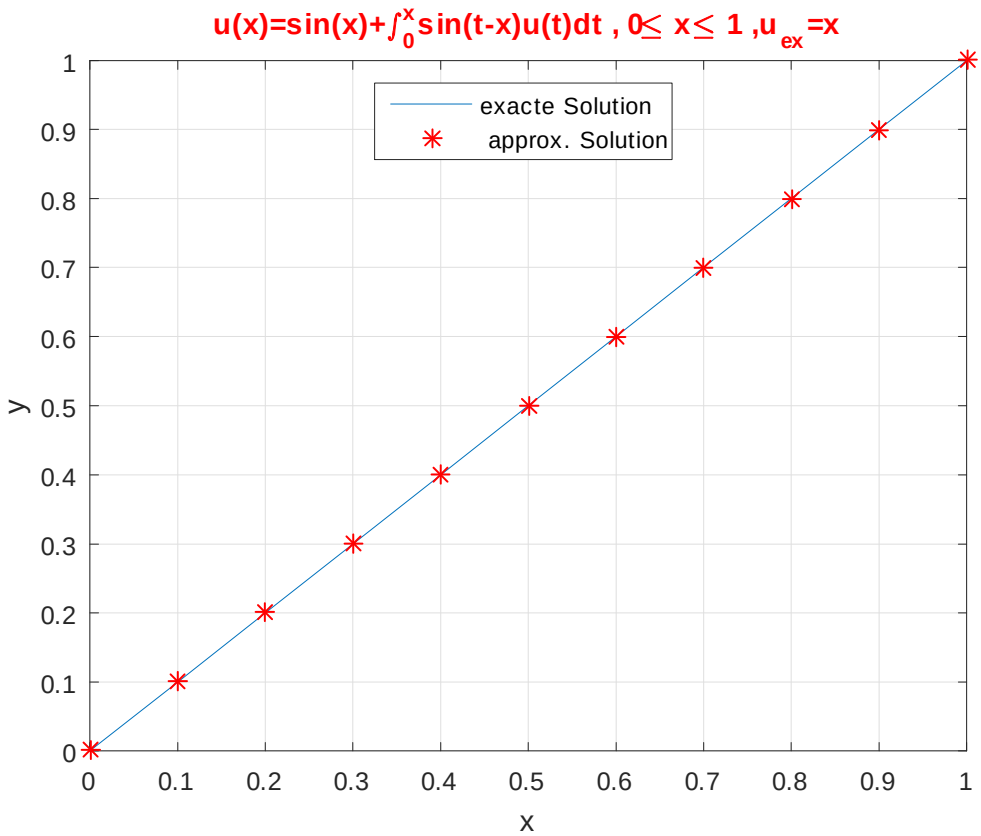
Tableau(1.3) la solution exacte et la solution approchée pour exemple1



Figure(1.3):Comparaison entre la solution exacte et la solution approchée pour l'utilisation des fonctions spline polynomiales de seconde ordre pour exemple 1

n	x	sol exacte	sol approchée	Erreur
1	0.00	0.000000	0.00000000000000	$0.00000000000000e+00$
2	0.10	0.100000	0.10000000000000	$0.00000000000000e+00$
3	0.20	0.200000	0.20000000000000	$0.00000000000000e+00$
4	0.30	0.300000	0.30000000000000	$0.00000000000000e+00$
5	0.40	0.400000	0.40000000000000	$0.00000000000000e+00$
6	0.50	0.500000	0.50000000000000	$0.00000000000000e+00$
7	0.60	0.600000	0.60000000000000	$0.00000000000000e+00$
8	0.70	0.700000	0.70000000000000	$0.00000000000000e+00$
9	0.80	0.800000	0.80000000000000	$1.11022302462516e-16$
10	0.90	0.900000	0.90000000000000	$1.11022302462516e-16$
11	1.00	1.000000	1.00000000000000	$1.11022302462516e-16$

Tableau(1.4) la solution exacte et la solution approchée pour exemple2



Figure(1.4):Comparaison entre la solution exacte et la solution approchée pour l'utilisation des fonctions spline polynomiales de seconde ordre pour exemple 2

3.2 Résolution de l'équation intégrale de Volterra de seconde espèce par les fonctions spline non polynomiales

Pour résoudre l'équation

$$u(x) = f(x) + \int_a^x k(x,t)u(t)dt \quad , x \in [a,b] \tag{B.14'}$$

On dérive l'équation$(B.14\prime)$ deux fois par rapport x, par l'utilisation de la formule de leibniz, on a:

$$u'(x) = f'(x) + \int_a^x \frac{\partial k(x,t)}{\partial x}u(t)dt + k(x,x)u(x) \tag{B.15}$$

$$\begin{aligned} u''(x) &= f''(x) + \int_a^x \frac{\partial^2 k(x,t)}{\partial x^2}u(t)dt + (\frac{\partial k(x,t)}{\partial x})_{t=x}u(x) + \\ &\quad (\frac{dk(x,t)}{dx})u(x) + k(x,x)u'(x) \end{aligned} \tag{B.16}$$

$$\begin{aligned} u^{(3)}(x) &= f^{(3)}(x) + \int_a^x \frac{\partial^3 k(x,t)}{\partial x^3}u(t)dt + (\frac{\partial^2 k(x,t)}{\partial x^2})_{t=x}u(x) + \\ &\quad \frac{d}{dx}(\frac{\partial k(x,t)}{\partial x})_{t=x}u(x) + (\frac{\partial k(x,t)}{\partial x})_{t=x}u'(x) + \frac{d^2k(x,x)}{d\ x^2}u(x) + \\ &\quad 2\frac{dk(x,x)}{dx}u'(x) + k(x,x)u''(x) \end{aligned} \tag{B.17}$$

$$\begin{aligned} u^{(4)}(x) &= f^{(4)}(x) + \int_a^x \frac{\partial^4 k(x,t)}{\partial x^4}u(t)dt + (\frac{\partial^3 k(x,t)}{\partial x^3})_{t=x}u(x) + \\ &\quad \frac{d}{dx}(\frac{\partial^2 k(x,t)}{\partial x^2})_{t=x}u(x) + (\frac{\partial^2 k(x,t)}{\partial x^2})_{t=x}u'(x) + \\ &\quad \frac{d^2}{dx^2}(\frac{\partial k(x,t)}{\partial x})_{t=x}u(x) + 2\frac{d}{dx}(\frac{\partial k(x,t)}{\partial x})_{t=x}u'(x) + \\ &\quad (\frac{\partial k(x,t)}{\partial x})_{t=x}u''(x) + \frac{d\ ^3k(x,t)}{dx^3}u(x) + \\ &\quad 3\frac{d\ ^2k(x,t)}{dx^2}u'(x) + 3\frac{d\ k(x,t)}{dx}u''(x) + \\ &\quad 3\frac{d\ ^2k(x,t)}{dx^2}u'(x) + 3\frac{d\ k(x,t)}{dx}u''(x) + \end{aligned} \tag{B.18}$$

Pour compléter cette méthode, on mettre $x = a$ dans les équations (B.14')-(B.18), ona:

$$u_0 = u(a) = f(a) \tag{B.19}$$

$$u_0' = u'(a) = f'(a) + k(a,a)u(a) \tag{B.20}$$

$$\begin{aligned} u_0'' &= u''(a) = f''(a) + ((\frac{\partial k(x,t)}{\partial x})_{t=x})_{x=a}u(a) + \\ &(\frac{dk(x,t)}{dx})_{x=a}u(a) + k(a,a)u'(a) \end{aligned} \tag{B.21}$$

$$\begin{aligned} u_0^{(3)} &= u^{(3)}(a) = f^{(3)}(a) + ((\frac{\partial^2 k(x,t)}{\partial x^2})_{t=x})_{x=a}u(a) + \\ &[\frac{d}{dx}(\frac{\partial k(x,t)}{\partial x})_{t=x}]_{x=a}u(a) + [(\frac{\partial k(x,t)}{\partial x})_{t=x}]_{x=a}u'(a) + \\ &(\frac{d^2k(x,x)}{dx^2})_{x=a}u(a) + 2(\frac{dk(x,x)}{dx})_{x=a}u'(a) \\ &+k(a,a)u''(a) \end{aligned} \tag{B.22}$$

$$\begin{aligned} u_0^{(4)} &= u^{(4)}(a) = f^{(4)}(a) + [(\frac{\partial^3 k(x,t)}{\partial x^3})_{t=x}]_{x=a}u(a) + \\ &[\frac{d}{dx}(\frac{\partial^2 k(x,t)}{\partial\ x^2})_{t=x}]_{x=a}u(a) + [(\frac{\partial^2 k(x,t)}{\partial x^2})_{t=x}]_{x=a}u'(a) + \\ &[\frac{d^2}{dx^2}(\frac{\partial k(x,t)}{\partial x})_{t=x}]_{x=a}u(a) + 2[\frac{d}{dx}(\frac{\partial k(x,t)}{\partial x})_{t=x}]_{x=a}u'(a) + \\ &[(\frac{\partial k(x,t)}{\partial x})_{t=x}]_{x=a}u''(a) + (\frac{d\ ^3k(x,t)}{dx^3})_{x=a}u(a) + \\ &3(\frac{d\ ^2k(x,t)}{dx^2})_{x=a}u'(x) + 3(\frac{d\ k(x,t)}{dx})_{x=a}u''(x) + \\ &k(a,a)u^{(3)}(a) \end{aligned} \tag{B.23}$$

3.2.1 Utilisation des fonctions spline non polynomiales linéaire

Description de la méthode

On approximier la solution de l'équation intégrale de Volterra de seconde espèce par la fonction splinenon polynomiale de linéaire. nous introduisons la méthode de solution dans l'algorithme.

L'Algorithme(VIE2NPS1):

étape1: mettre $h = (b-a)/n, t_i = t_0 + ih$,$i = 0, .., n$ (où $t_0 = a, t_n = b$) et $u_0 = f(a)$.

étape2: evalué a_0 ,b_0, c_0 et d_0 par substitué (B.19)-(B.22) dans les équations (B.4)-(B.7).

étape3: calcule $p_0(t)$ on utilise l'étape 2 et l'équation (B.2).

étape4: approximant $u_1 = p_0(t_1)$.

étape5: pour $i = 1$à $n-1$ fait les étapes suivantes :

étape6: evalué a_i, b_i, c_i et d_i par l'utilisation des équations (B.4)-(B.7) et remplace $u(t_i), u'(t_i)$, $u''(t_i)$ et $u^{(3)}(t_i)$ par $p_i(t_i)$, $p_i'(t_i)$, $p_i''(t_i)$ et $p_i^{(3)}(t_i)$.

étape7: calcule $p_i(t)$ on utilise l'étape 6 et l'équation (B.2).

étape8: approximant $u_{i+1} = p_i(t_{i+1})$.

Exemples test

exemple01:

On considère l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = x + \int_0^x (t-x)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = \sin(x)$.

exemple02:

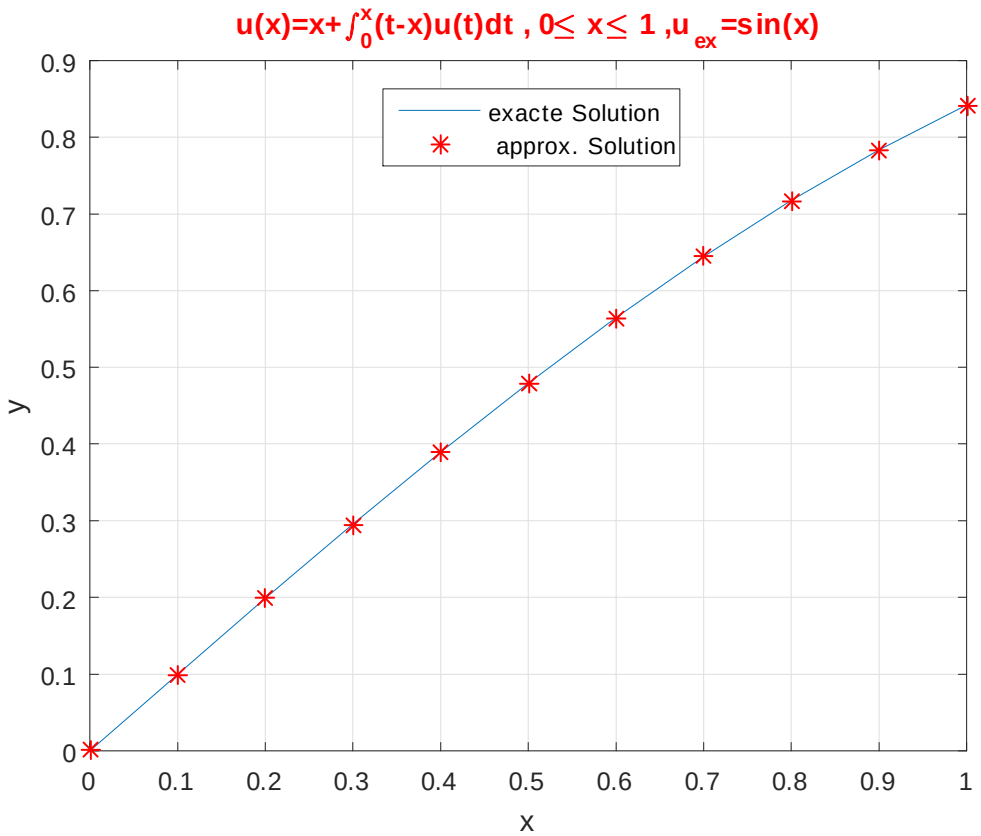
On considére l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = \sin(x) + \int_0^x \sin(x)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = x$.

n	x	sol exacte	sol approchée	Erreur
1	0.00	0.00000000000000	0.00000000000000	$0.00000000000000e+00$
2	0.10	0.09983341664683	0.09983341664683	$0.00000000000000e+00$
3	0.20	0.19866933079506	0.19866933079506	$0.00000000000000e+00$
4	0.30	0.29552020666134	0.29552020666134	$5.55111512312578e-17$
5	0.40	0.38941834230865	0.38941834230865	$1.11022302462516e-16$
6	0.50	0.47942553860420	0.47942553860420	$1.11022302462516e-16$
7	0.60	0.56464247339504	0.56464247339504	$1.11022302462516e-16$
8	0.70	0.64421768723769	0.64421768723769	$1.11022302462516e-16$
9	0.80	0.71735609908995	0.71735609908995	$2.22044604925031e-16$
10	0.90	0.78332269096274	0.78332269096274	$3.33066907387547e-16$
11	1.00	0.84147098480790	0.84147098480790	$4.44089209850063e-16$

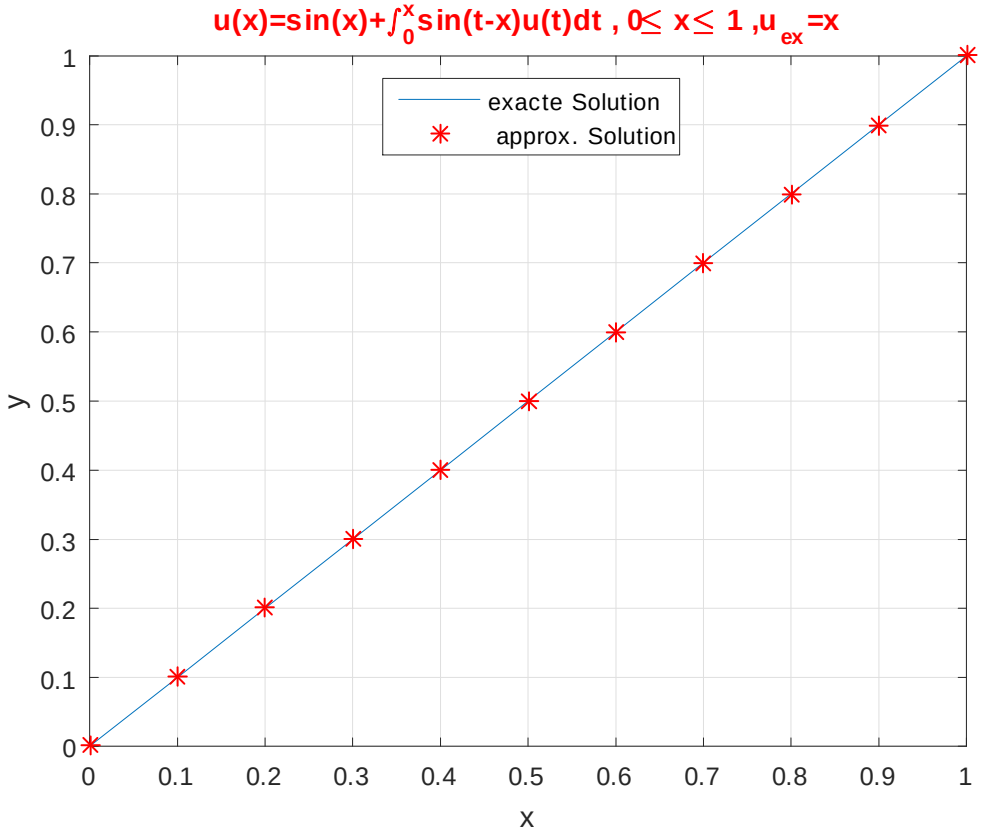
Tableau(2.1)la solution exacte et la solution approchée pour exemple1.



Figure(2.1):Comparaison entre la solution exacte et la solution approchée pour l'utilisation des fonctions spline non polynomiales linéaire pour exemple 1

n	x	sol exacte	sol approchée	Erreur
1	0.00	0.00000000000000	0.00000000000000	$0.00000000000000e+00$
2	0.10	0.10000000000000	0.10000000000000	$0.00000000000000e+00$
3	0.20	0.20000000000000	0.20000000000000	$0.00000000000000e+00$
4	0.30	0.30000000000000	0.30000000000000	$0.00000000000000e+00$
5	0.40	0.40000000000000	0.40000000000000	$0.00000000000000e+00$
6	0.50	0.50000000000000	0.50000000000000	$0.00000000000000e+00$
7	0.60	0.60000000000000	0.60000000000000	$0.00000000000000e+00$
8	0.70	0.70000000000000	0.70000000000000	$0.00000000000000e+00$
9	0.80	0.80000000000000	0.80000000000000	$1.11022302462516e-16$
10	0.90	0.90000000000000	0.90000000000000	$1.11022302462516e-16$
11	1.00	1.00000000000000	1.00000000000000	$1.11022302462516e-16$

Tableau(2.2) la solution exacte et la solution approchée pour exemple1.



Figure(2.2):Comparaison entre la solution exacte et la solution approchée pour l'utilisation des fonctions spline non polynomiales linéaire pour exemple 2

3.2.2 Utilisation des fonctions spline non polynomiales quadratique

Description de la méthode

On approximier la solution de l'équation intégrale de Volterra de seconde espèce par la fonction spline non polynomiale quadratique. nous introduisons la méthode de solution dans l'algorithme.

L'Algorithme(VIE2NPS2):

étape1: mettre $h = (b-a)/n, t_i = t_0 + ih$,$i = 0, .., n$ (où $t_0 = a, t_n = b$) et $u_0 = f(a)$.

étape2: evalué a_0 ,b_0, c_0 ,$d_0$et e_0 par substitué (B.19)-(B.23) dans les équations (B.10)-(B.14).

étape3: calcule $p_0(t)$ on utilise l'étape2 et l'équation (B.8).

étape4: approximant $u_1 = p_0(t_1)$.

étape5: pour $i = 1$ à $n-1$ fait les étapes suivantes :

étape6: evalué a_i, b_i, c_i ,d_i et e_i par utilisation des équations (B.10)-(B.14) et remplace $u(t_i)$, $u'(t_i)$, $u''(t_i)$, $u^{(3)}(t_i)$ et $u^{(4)}(t_i)$ par $p_i(t_i)$, $p_i'(t_i)$, $p_i''(t_i)$, $p_i^{(3)}(t_i)$ et $p_i^{(4)}(t_i)$.

étape7: calcule $p_i(t)$ on utilise l'étape 6 et l'équation (B.8).

étape8: approximant $u_{i+1} = p_i(t_{i+1})$.

Exemples test

exemple01:

On considère l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = x + \int_0^x (t-x)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = \sin(x)$.

exemple02:

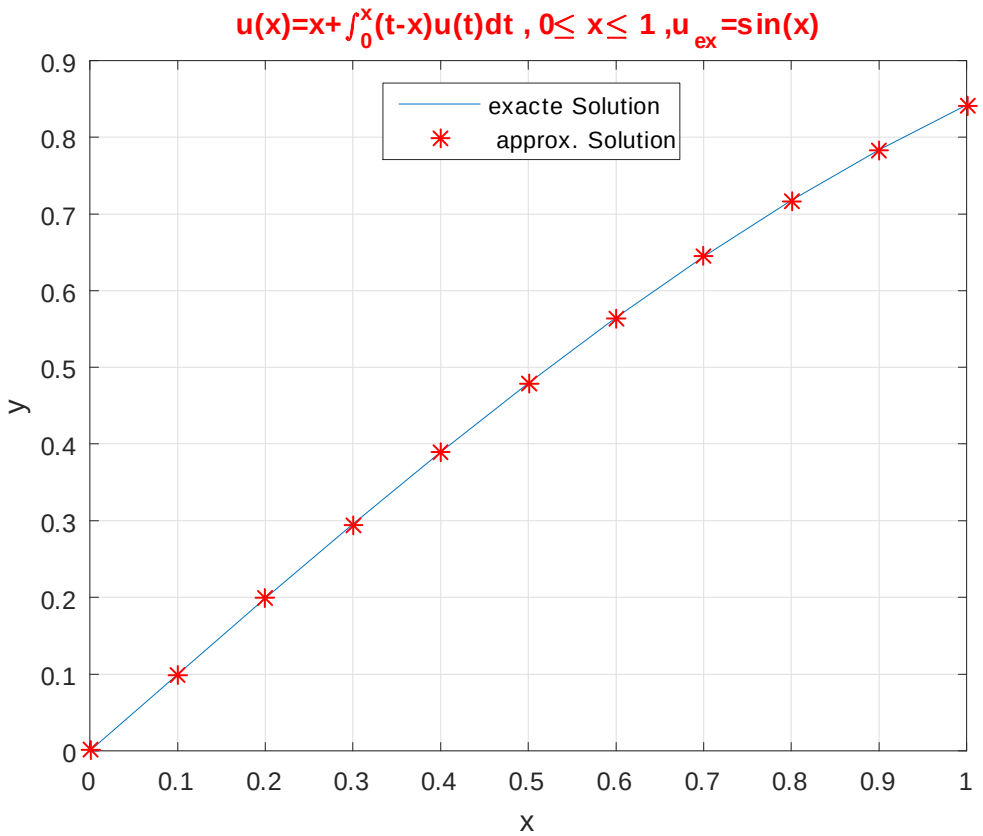
On considère l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = \sin(x) + \int_0^x \sin(x)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = x$.

n	x	sol exacte	sol approchée	Erreur
1	0.00	0.00000000000000	0.00000000000000	$0.00000000000000e+00$
2	0.10	0.09983341664683	0.09983341664683	$0.00000000000000e+00$
3	0.20	0.19866933079506	0.19866933079506	$0.00000000000000e+00$
4	0.30	0.29552020666134	0.29552020666134	$5.55111512312578e-17$
5	0.40	0.38941834230865	0.38941834230865	$1.11022302462516e-16$
6	0.50	0.47942553860420	0.47942553860420	$1.11022302462516e-16$
7	0.60	0.56464247339504	0.56464247339504	$1.11022302462516e-16$
8	0.70	0.64421768723769	0.64421768723769	$1.11022302462516e-16$
9	0.80	0.71735609908995	0.71735609908995	$2.22044604925031e-16$
10	0.90	0.78332269096274	0.78332269096274	$3.33066907387547e-16$
11	1.00	0.84147098480790	0.84147098480790	$4.44089209850063e-16$

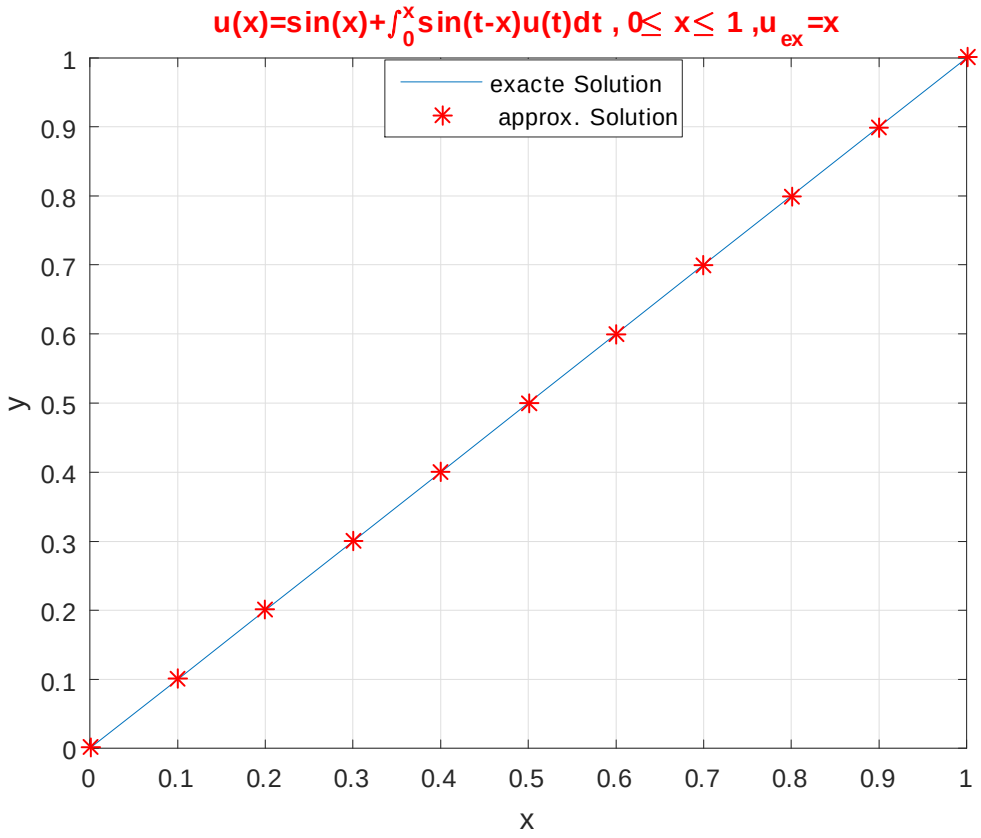
Tableau(2.3) la solution exacte et la solution approchée pour exemple1.



Figure(2.3):Comparaison entre la solution exacte et la solution approchée pour l'utilisation des fonctions spline non polynomiales quadratique pour exemple 1

n	x	sol exacte	sol approchée	Erreur
1	0.00	0.00000000000000	0.00000000000000	$0.00000000000000e+00$
2	0.10	0.10000000000000	0.10000000000000	$0.00000000000000e+00$
3	0.20	0.20000000000000	0.20000000000000	$0.00000000000000e+00$
4	0.30	0.30000000000000	0.30000000000000	$0.00000000000000e+00$
5	0.40	0.40000000000000	0.40000000000000	$0.00000000000000e+00$
6	0.50	0.50000000000000	0.50000000000000	$0.00000000000000e+00$
7	0.60	0.60000000000000	0.60000000000000	$0.00000000000000e+00$
8	0.70	0.70000000000000	0.70000000000000	$0.00000000000000e+00$
9	0.80	0.80000000000000	0.80000000000000	$1.11022302462516e-16$
10	0.90	0.90000000000000	0.90000000000000	$1.11022302462516e-16$
11	1.00	1.00000000000000	1.00000000000000	$1.11022302462516e-16$

Tableau(2.4)la solution exacte et la solution approchée pour exemple2.



Figure(2.4):Comparaison entre la solution exacte et la solution approchée pour l'utilisation des fonctions spline non polynomiales quadratique pour exemple 2.

3.3 L'Analyse des résultats

exemple03:

On considère l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = e^x + \int_0^x \emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = (1+x)e^x$.

exemple04:

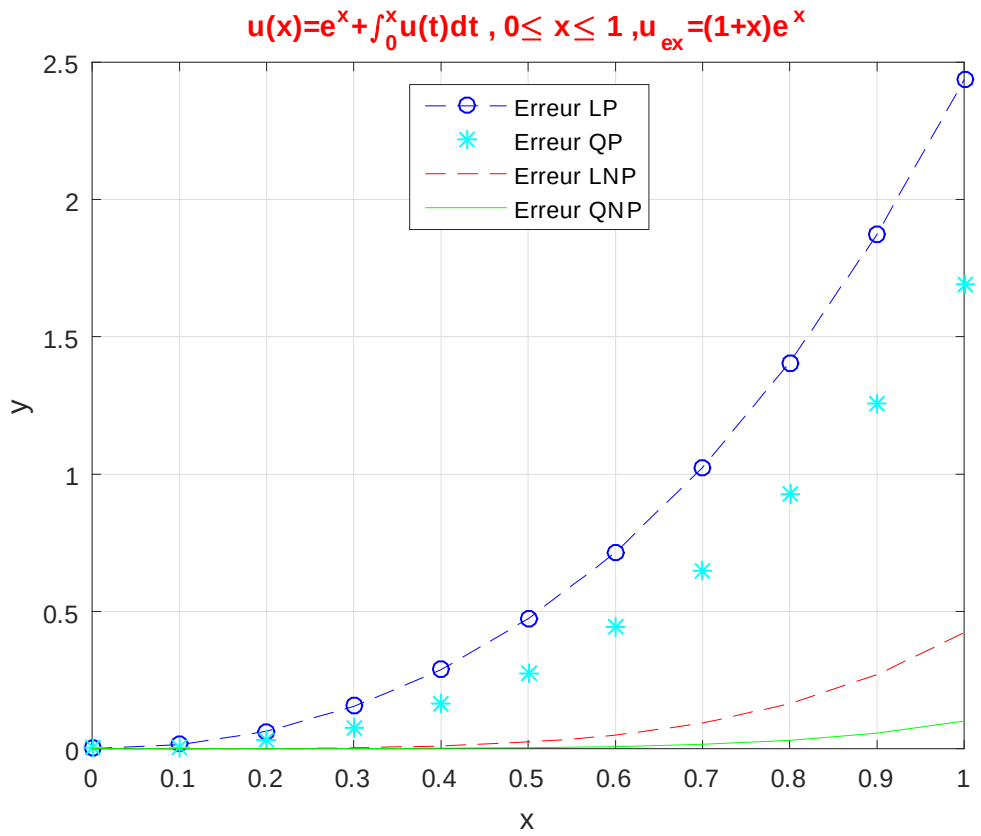
On considère l'équation intégrale de Volterra de seconde espèce

$$\emptyset(x) = x - 2 + 2e^x + \int_0^x (x-t)\emptyset(t)dt \quad , 0 \leq x \leq 1$$

Avec la solution exacte $\emptyset(x) = xe^x$.

n	x	sol exacte	Erreur de LP	Erreur de QP	Erreur deNLP	Erreur de NQP
1	0.00	1.0000000	$0.000000e+00$	$0.000000e+00$	$0.000000e+00$	$0.000000e+00$
2	0.10	1.2156880	$1.568800e-02$	$6.880098e-04$	$3.417230e-05$	$8.500803e-07$
3	0.20	1.4656833	$6.568330e-02$	$3.568330e-02$	$5.603664e-04$	$2.774376e-05$
4	0.30	1.7548164	$1.548164e-01$	$7.981644e-02$	$2.906743e-03$	$2.148308e-04$
5	0.40	2.0885545	$2.885545e-01$	$1.685545e-01$	$9.410927e-03$	$9.229759e-04$
6	0.50	2.4730819	$4.730819e-01$	$2.780819e-01$	$2.353174e-02$	$2.871251e-03$
7	0.60	2.9153900	$7.153900e-01$	$4.453900e-01$	$4.996681e-02$	$7.281899e-03$
8	0.70	3.4233796	$1.023379e+00$	$6.483796e-01$	$9.477691e-02$	$1.603941e-02$
9	0.80	4.0059736	$1.405973e+00$	$9.259736e-01$	$1.655181e-01$	$3.186448e-02$
10	0.90	4.6732459	$1.873245e+00$	$1.258245e+00$	$2.713834e-01$	$5.850370e-02$
11	1.00	5.4365636	$2.436563e+00$	$1.686563e+00$	$4.233545e-01$	$1.009360e-01$

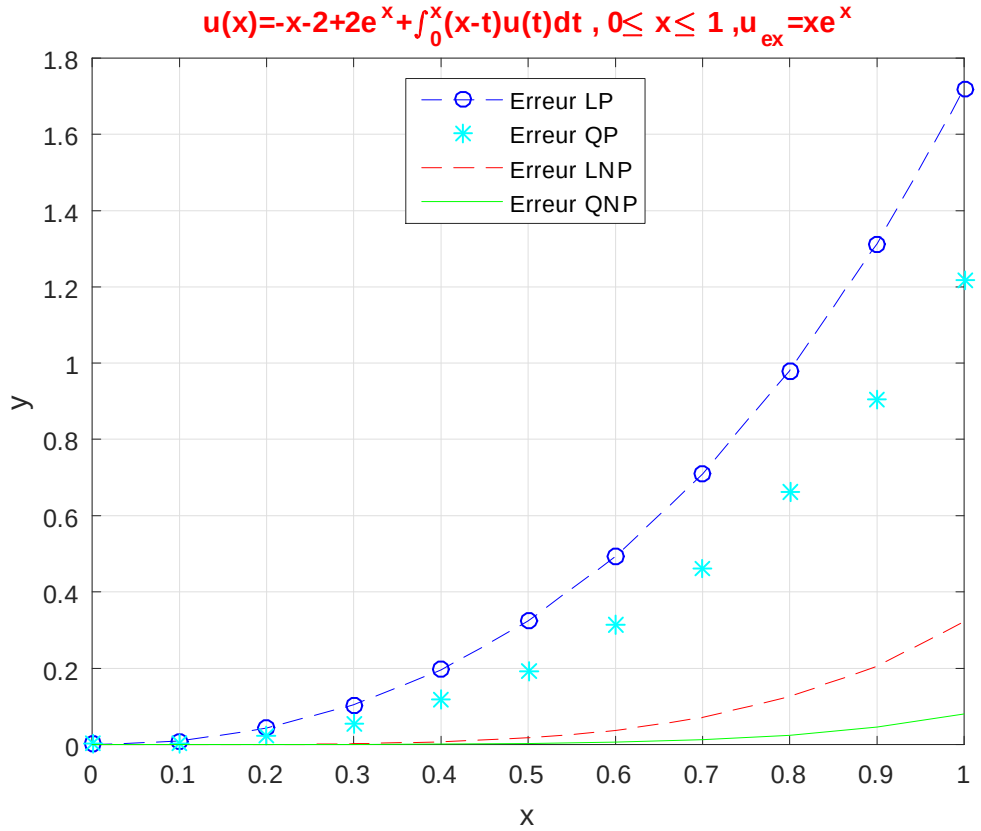
Tableau(3.1):Comparaison entre Les erreur obtenue par Les fonctions spline polynomiales et Les fonctions spline non polynomiales pour exemple 3



Figure(3.1):Comparaison entre Les erreur obtenue par Les fonctions spline polynomiales et Les fonctions spline non polynomiales pour Exemple 3

n	x	sol exacte	Erreur de LP	Erreur de QP	Erreur deNLP	Erreur de NQP
1	0.00	0.000000	$0.000000e+00$	$0.000000e+00$	$0.000000e+00$	$0.000000e+00$
2	0.10	0.110517	$1.051709e-02$	$5.170918e-04$	$2.567230e-05$	$6.806359e-07$
3	0.20	0.244280	$4.428055e-02$	$2.428055e-02$	$4.216996e-04$	$2.223265e-05$
4	0.30	0.404957	$1.049576e-01$	$5.495764e-02$	$2.191240e-03$	$1.723057e-04$
5	0.40	0.596729	$1.967298e-01$	$1.167298e-01$	$7.106893e-03$	$7.409299e-04$
6	0.50	0.824360	$3.243606e-01$	$1.943606e-01$	$1.780237e-02$	$2.307003e-03$
7	0.60	1.093271	$4.932712e-01$	$3.132712e-01$	$3.786993e-02$	$5.856240e-03$
8	0.70	1.409626	$7.096268e-01$	$4.596268e-01$	$7.196433e-02$	$1.291120e-02$
9	0.80	1.780432	$9.804327e-01$	$6.604327e-01$	$1.259144e-01$	$2.567417e-02$
10	0.90	2.213642	$1.313642e+00$	$9.036428e-01$	$2.068434e-01$	$4.718365e-02$
11	1.00	2.718281	$1.718281e+00$	$1.218281e+00$	$3.232993e-01$	$8.148555e-02$

Tableau(3.2) : Comparaison entre Les erreur obtenue par Les fonctions spline polynomiales et Les fonctions spline non polynomiales pour exemple 4



Figure(3.2):Comparaison entre Les Erreur obtenue par Les fonctions spline polynomiales et Les fonctions spline non polynomiales pour exemple 4

Après La présentation des figures, des tableaux, et à partir de la comparaison entre les erreurs, on peut remarquer:

- L'approximation par les fonctions spline non polynomiales donne meilleurs precision à comparée des fonctions spline polynomiales.
- L'approximation par les fonctions spline polynomiales de seconde ordre donne meilleurs precision à comparée des fonctions spline polynomiales de premier ordre.
- L'approximation par les fonctions spline non polynomiales quadratique donne meilleurs precision à comparée de les fonctions spline non polynomiales linéaire

Conclusion

Les méthodes utilisées dans ce mémoire montre leur efficacité pour la résolution de l'équation intégrale de Volterra de seconde espèce, l'Analyse des résultats obtenues montre que ces méthodes donnent bon Approximation

.

Bibliographie

[1] **Adhra 'a M.Muhammad,** Numerical Solution of Volterra Integral Equation With delay by using Non-Polynomial Spline Function, Misan Journal for Academic Studies.No.32,2017.

[2] **Azouz. I,** Etude sur la décomposition polaire des opérateur ,Mémoire de Master,Université de M'sila,2013.

[3] **Burden. R.L. and Faries, J.D**, Numerical Analysis .Ninth Edition.Brooks/cole, Cengage Learing,2010.

[4] **Benyoussef.s**, Resolution Numérique des Equations Intégro-différentielles de Fredholm,Mémoire de Magitère,Université de M'sila,2014.

[5] **Claude Zuily,** Elément de Distribution et D'équation aux Dérivées Partielles,Université de Paris XI-Oray,2002.

[6] **Dilmi.N,** Résolution numérique des équations intégrales, Mémoire de Master, Université de M'sila, 2018.

[7] **Fronçois Boulier,** Calcule Numérique,Interpolation,Intégration,Equation différentielle, Support de cours de GIS, Polytech 'Lille, France, 2016.

[8] **G.Ch**, Class of Bezier,Aided Geom,Design 20,29-39,2003.

[9] **Guilopé Laurent,**Analyse Fonctionelle,Polytech'Nantes,France,2008.

[10] **Haim Brezis**, Functional Analysis Sobolev Spaces and Partial differential Equation,Rutgers University,USA,2010.

[11] **H,Harbi Sarah**, Algorithms For Solving Volterra Integral Equations Using Non-polynomiale Spline Functions,Thesis,University of Baghdad,2013.

[12] Linear Volterra integral Equations.http/:WWW Cambridge.Org/Core.Binghanton University.on 2017.

[13] **Kh.Maleknegad,.Rashidina,H.Jalilian**, Non-polynomial spline Function and Quazi-linearisation to Approximate Non linear Volterra Integral Equation,University of Nis ,Serbia,2018.

[14] **Khirani. A,** Etude des équations Intégrales non linéaire de Volterra dans Les espaces fonctionnels, thèse de Doctorat ,Université de M'sila, 2016.

[15] **Soukeur. A,** Sur L'aspect numérique des solution des équations intégrales de Volterra lineaire et non linéaire au moyen d'une nouvelle méthode quadratique intégrale généralisé (QIG), Mémoire de Magitère, Université de Biskra,2004..

[16] **Seraidi.A**, Sur la Résolution Numérique des équations Integro-diffirentielles Via de Polynome D'Hermite, Mémoire de Master, Université de M'sila,2017.

[17] **S.S.Sastry**, Introductory Methods of numerical Analysis , New Delhi-110001,2012.

[18] **Wazwaz.A**, Linear and Non linear Integrale Equation Method and Application ,Higher Education Press ,Beijing,2011.

[19] **Zerbnia .M;Hoshyar .M; and Sedahti.M,** Non-polynomial Spline Method For The Solution of Problem in Calculus of variation,Word Academy Engendering Thechnology (51),2011.

ملخص

في هذا البحث، دوال الثلمة متعددة الحدود والغير متعددة الحدود طبقت لإيجاد حل تقريبي لمعادلات فولتيرا التكاملية من النوع الثاني.

علاوة على ذلك، تم تقديم أمثلة عددية لتوضيح كفاءة هذه الطريقة،

الكلمات المفتاحية:

معادلات فولتيرا التكاملية من النوع الثاني، دوال الثلمة متعددة الحدود، دوال الثلمة الغير متعددة الحدود.

Abstract

In this memory, polymonial spline functions and non- polynomial spline functions are used to find approximate solution of Volterra integral equation of the second kind.

Moreover , Numerical examples are presented to show the effectiveness of this method

Key- words :

Volterra integral equation of the second kind, polymonial spline functions, non-polynomial spline functions.

Résume

Dans ce mémoire, les fonctions spline polynomiales et les fonctions spline non polynomiales ont été appliquées pour trouver une solution approchée de l'équation intégrale de Volterra de seconde espèce.

En plus, des exemples numérique illustratifs ont été utilisés pour montrer l'efficacité de la méthode.

Mots Clé :

équation intégrale de Volterra de seconde espèce, fonctions splines

polynomiales, fonctions splines non polynomiales.
