Auteur: Elio BTEICH

Interpolation de Lagrange

Introduction

L'interpolation est un domaine mathématique et informatique, en pleine recherche, elle consiste à rapprocher une fonction complexe difficile à examiner avec une fonction beaucoup plus simple, facile à étudier. L'interpolation de lagrange est l'une des méthodes d'interpolation les plus utilisées grâce à sa simplicité et sa flexibilité.

Dans ce projet, j'ai implémenté l'algorithme d'interpolation de Lagrange en C++ pour permettre à l'utilisateur de trouver une courbe d'interpolation pour une série de points d'interpolation donnés. J'ai utilisé la bibliothèque graphique GnuPlot pour afficher les courbes d'interpolation. Les résultats de mon programme montrent que mon implémentation est efficace et précise pour résoudre des problèmes de calcul numérique. Dans les sections suivantes, je discuterai en détail de ma méthodologie, des résultats obtenus et des implications pour les utilisateurs potentiels de mon programme.

Contexte

L'interpolation de lagrange consiste à traiter chaque point d'interpolation à part, en lui trouvant un polynôme qui passe par lui même mais qui s'annule pour les autres points d'interpolation, et comme ça on pourra additionner tous ces polynômes pour obtenir le polynôme de lagrange qui passe par tous les points d'interpolations.

Le programme que j'ai écrit peut être utilisé pour approximer une fonction sur un intervalle de points d'interpolations équidistants. Il fournira aussi une représentation graphique de la courbe de la fonction et des courbes de l'erreur relative et absolue.
Il affichera aussi sur la console l'erreur maximale absolue et relative, qui permettra à l'utilisateur de voir l'efficacité de cet algorithme et de la visualiser grâce aux courbes d'erreurs.

Methodologie

Pour ce faire j'ai créé un fichier source Lagrange.cpp avec son fichier de définition que j'ai inclus dans le main.cpp et util.cpp.
Les fonction de ce fichier sont les suivantes:

- xvals: décompose un intervalle en n sous intervalle
- yvals: prends un tableau de points et retourne un tableau d'image de ces points par une fonction donnée
- base_lagrange: évalue le polynôme de base de lagrange d'indice i passée en paramètre pour un x donnée
- approx_lagrange: évalue le polynôme de lagrange pour un point donnée

En plus j'ai modifié quelques fonctions dans le fichier util.cpp pour que ça fonctionne bien avec mon programme et j'ai utilisé les fonctions de ce fichier pour calculer les erreurs maximaux de mon interpolation, et j'ai dessiné toutes les courbes dans mon programme grâce à la fonction plot qui utilise le logiciel gnuplot.

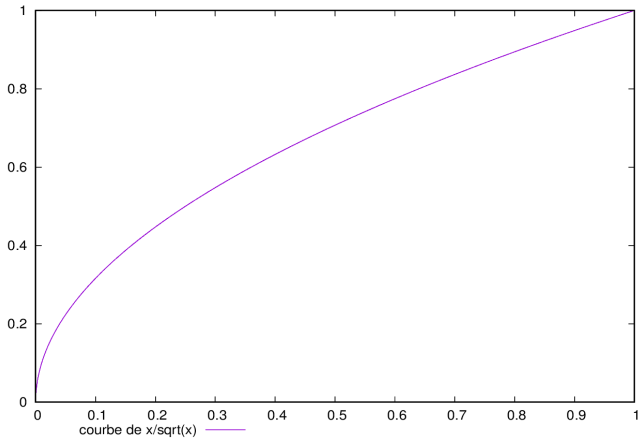
Par contre j'ai rencontré lors de l'implémentation de mon algorithme un défis qui était un problème de division par 0 qui produisait un NAN (not a number) qui traînait dans mon programme, et qui l'empêchait de s'exécuter normalement. C'était à cause de l'intervalle des points d'interpolation qui commençait par 0, vu que la fonction qu'on devait interpoler est x/sqrt(x) donc au moment du calcul de l'image de chaque point d'interpolation, le premier point à être évalué était le 0 pour lequel la fonction n'est pas défini et qui donnait comme image 0/sqrt(0).
Heureusement, j'ai pu gérer le problème, en se décalant un peu vers la droite pour le point 0 lors de la décomposition de l'intervalle vu que la fonction était défini pour $x > 0$.

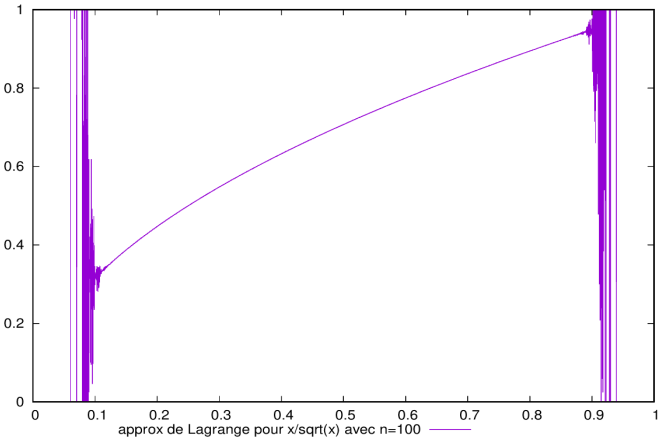
Dans notre cas, c'était juste pour le premier élément de l'intervalle que la fonction n'est pas défini, ce qui nous a permis de se décaler d'un très petit epsilon, car n'importe quel point après le 0 valide la condition $x>0$, mais ceci n'est pas un cas générique, donc par exemple si on avait choisi comme interval [-10, 10], se décaler juste d'un petit epsilon vers la droite n'aurait pas été suffisant. Cela veut dire que l'étude de chaque fonction se fait à part, trouvé un programme générique qui fait l'interpolation de n'importe quelle fonction est une tâche compliqué à faire en tant qu'étudiant de deuxième année de licence.

Résultats

Pour n = 10 dans l'intervalle [0, 1], le polynôme de Lagrange évalué pour 0.55 est égale à 0.7416107676346457, or f(0.55) = 0.7416198487095663

Voici ci-dessous la représentation de la courbe f(x) = x/sqrt ainsi que l'interpolation de Lagrange de cette dernière avec 100 points d'interpolations.

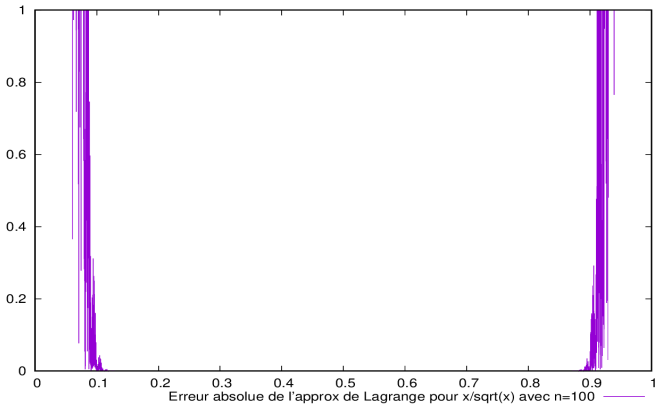


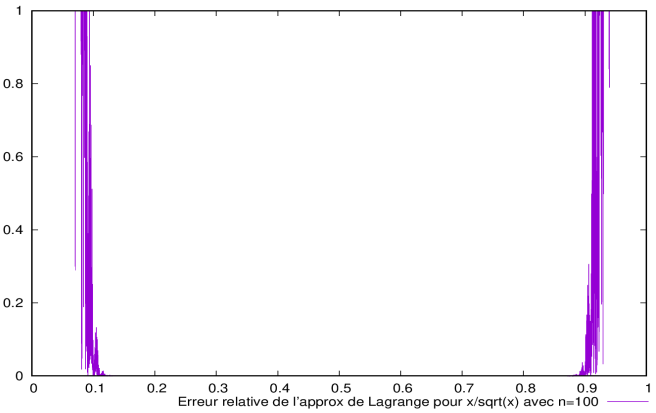


Discussion

En comparant, dans la section "Résultats", les valeurs de f(0.55) et l'image du polynôme de Lagrange évalué pour 0.55 on remarque que ces deux valeurs sont très proche l'une de l'autre, elles sont les mêmes jusqu'à 5 chiffres après la virgule pour un petit nombre de points d'interpolation (n=10).
Pour la courbe d'approximation de Lagrange ci-dessus, on remarque un comportement bizarre pour x<=0.1 et x>=0.9, mais malgré cela, les deux courbes présentées ci-dessus se ressemblent sur l'intervalle]0.1, 0.9[.

Ainsi, on peut déduire que l'approximation de Lagrange n'est pas efficace au voisinage de l'intervalle des points d'interpolations, mais par contre plus on s'éloigne des bornes plus notre résultat est précis. On peut visualiser l'efficacité de l'interpolation de Lagrange pour n=100 grâce à la courbe de l'erreur relative et absolue ci-dessous:





On remarque qu'à partir de 0.12 à peu près et avant 0.88 l'erreur est presque 0, et ça explose avant 0.12 et après 0.88. Donc on peut dire que si on choisit correctement notre intervalle en fonction de ce qu'on recherche, cet algorithme est très efficace.

Conclusion

En conclusion, j'ai présenté dans ce rapport mon projet sur l'interpolation de Lagrange en C++. J'ai d'abord donné un contexte sur l'interpolation et le choix de Lagrange en tant que méthode d'interpolation. J'ai ensuite présenté les résultats obtenus suite à la mise en œuvre de l'algorithme en C++. Enfin, j'ai discuté des résultats obtenus en comparant les approximations obtenues à l'aide de la méthode de Lagrange avec ce qui était attendu en fournissant les courbes d'erreurs.

En général, on peut dire que l'interpolation de Lagrange est une méthode fiable et efficace pour approximer des fonctions. Cependant, il est important de choisir judicieusement les points d'interpolation et les intervalles pour éviter les erreurs et obtenir des approximations de haute qualité.

En fin de compte, je peux conclure que ce projet a été une expérience enrichissante qui m'a permis de comprendre les concepts mathématiques liés à l'interpolation et de les implémenter en C++.