الجمهوريـة الجزائـريـة الديمقراطيـة الشعبيـة
République Algérienne Démocratique et Populaire
وزارة التعليـم العالـي والبحـث العلمـي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

N°Réf :……

Centre Universitaire
Abdelhafid Boussouf Mila

Institut des sciences et de la technologie **Département de Mathématiques et Informatiques**

Mémoire préparé en vue de l'obtention du diplôme de Master

En: Mathématiques
Spécialité : Mathématiques fondamentales et appliquées

APPLICATION DE L'APPROXIMATION PAR ELEMENTS FINIS AU LISSAGE DES TRAJECTOIRES D'OUTILS

Préparé par:
Bouguebina saliha

Soutenue devant le jury
Président : I.BOUFELGHA ……M.A.A
Examinateur : M.LEDRA……..... ….. …...……M .C.A
Encadrée par : H.BOUCHENITFA ……..M.C.B

Année universitaire : 2015/2016

Remerciements

Je remercie tout d'abord Allah le tout puissant de m'avoir aidé et donné la patience et le courage durant ces longues et dures années d'études. Je remercie aussi mon encadreur Monsieur H.Bouchenitfa
pour son orientation , sa confiance, ces précieux conseils, sa patience et son aide tout au long de la réalisation de ce mémoire.

Mes vifs remerciements au membre du jury Messieurs M.ledra et I.Boufelgha

Je tiens à remercier également l'ensemble de mes professeurs de master 2 MFA ainsi que toute personne ayant contribué de prêt ou de loin à l'élaboration de ce travail

A toutes ces personnes un grand MERCI

MARWA

Résumé

Avec l'augmentation de la concurrence, l'exigence en termes de qualité et performances augmente chaque jour. Le domaine de la fabrication mécanique n'est pas à l'égard de cette règle car il y a une demande pour des pièces de plus en plus précise, particulièrement pour des domaines comme l'aéronautique ou l'équipement médical. C'est dans cette idée que vient ce travail qui concerne l'amélioration de la précision de l'usinage des pièces mécanique. Cette amélioration concerne la trajectoire d'outils lors de l'usinage des surfaces. Pour cela, le principe est d'exploiter les techniques d'approximation mathématique pour effectuer un lissage de ces trajectoires afin de passer d'une trajectoire composée de segments de droites à une trajectoire composée de courbes polynomiales. Le résultat final est une surface lisse de bonne précision réalisée dans un temps minimal.

Abstract

With increasing competition, the requirement in terms of quality and performance is also increase every day. The field of mechanical manufacturing is not in opposition to this rule because there is a demand for more accurate parts, especially for areas such as aerospace or medical equipment. It is with this idea that this working is trying to improving the machining precision of mechanical parts. This improvement concerns the machining tool path of the surfaces. For this, the principle is to use mathematical approximation techniques to perform a smoothing of these trajectories end to pass from a path made up of line segments in to a path composed of polynomial curves. The end result is a good precision and a smooth surface made in a minimum time.

ملخص

مع المنافسة الكبيرة تتضاعف متطلبات النوعية كل يوم وميدان التصنيع الميكانيكي لا يخالف هذه القاعدة لأن بعض القطع أساسها الدقة وخصوصا في ميدان الطيران والتجهيز الطبي. وانطلاقا من هذه الفكرة يأتي هذا العمل لتحسين الدقة في تصنيع القطع ألميكانيكية هذا التحسن يتعلق بمسار الأداة التصنيع خلال عملية تشكيل أسطح القطع، والمبدأ هو استخدام تقنيات التقريب الرياضية لتحسين سلاسة المسارات بتحويلها من مسارات تتكون من قطع مستقيمة الى مسارات تتكون من منحنيات والنتيجة النهائية هي الحصول على سطح أملس جيد الدقة في أقصر وقت.

Table des figures

Table des matières

Introduction générale

Les exigences en termes de qualité et de précision, augmentent chaque jour. Cette augmentation touche tous les domaines. Dans le domaine de la fabrication de pièces mécaniques il est nécessaire aujourd'hui de maitriser les techniques de fabrication pour produire des pièces avec des défauts de forme et de dimensions très réduits et un état de surface parfait.

La maitrise de la précision de fabrication passe par l'utilisation de moyens modernes comme la fabrication assistée par ordinateur et l'usinage avec machine à commande numérique. Ces dernières utilisent pour leur fonctionnement des trajectoires définies par un ensemble de points par lesquelles passe l'outil de fabrication. La précision finale de l'usinage est liée directement à la précision de la réalisation de ces trajectoires.

Une trajectoire et composée de plusieurs segments de droite qui passe d'un point à un autre. Mais avec l'évolution des machines il est possible à présent, pour des machines modernes, d'utiliser des trajectoires sous forme de courbes.

Dans ce cas déterminer une trajectoire avec précision revient à calculer des courbes qui passent au mieux par des points définis dans un repère. Mathématiquement, cela revient à réaliser une interpolation ou une approximation des points de la trajectoire par des courbes. Ces dernières vont être traduites en instructions pour commander la machine et réaliser les pièces avec un maximum de précision.

Pour bien présenter ce travail, on propose de le partager en quatre chapitres. Dans le premier, des généralités sur l'usinage particulièrement l'usinage à commande numérique sera présenté pour pouvoir comprendre la problématique traité et la solution proposée. Dans le deuxième chapitre, différentes méthodes d'interpolation et d'approximation seront exposées. Dans le troisième, le nous allons détailler l'approximation par sous domaine et particulièrement l'approximation par éléments finis qui va être adoptée pour notre solution. Dans le quatrième chapitre, le détail de la solution à savoir le programme de calcul et de traitement des résultats sera présenté avec une application sur un cas pratique. A la fin on terminera par une conclusion générale.

Chapitre 1

Généralité sur l'usinage

1.1 Introduction *:*

L'usinage regroupe sous un même terme toutes les étapes de fabrication des pièces mécaniques composant les objets de notre quotidien. L'usinage d'une pièce mécanique fait appel á de nombreuses techniques diverses assistées ou non par ordinateur. Pour fabriquer les pièces mécaniques, la matière doit être façonnée de façon très précise selon les plans du bureau d'étude à l'origine de la conception. Cette phase de fabrication que l'on désigne globalement sous le terme d'usinage met en œuvre de nombreuses techniques particulières qui varient en fonction de la forme que l'on souhaite donner à la matière, le type de matière usinée, la taille des pièces á usiner et le nombre de pièces à réaliser (unitaire ou en série). Généralement, l'usinage de matière telle que la fonte ne nécessite pas une grande précision, par contre, les pièces réalisées en matières techniques comme l'aluminium, le téflon, le quartz...doivent être réalisées avec des tolérances de précision de l'ordre du micron. L'usinage des pièces mécaniques en bois, métal, céramique, plastique... se fait par étape successive sur des machines outils spécifiques ayant chacune une fonction particulière [1].

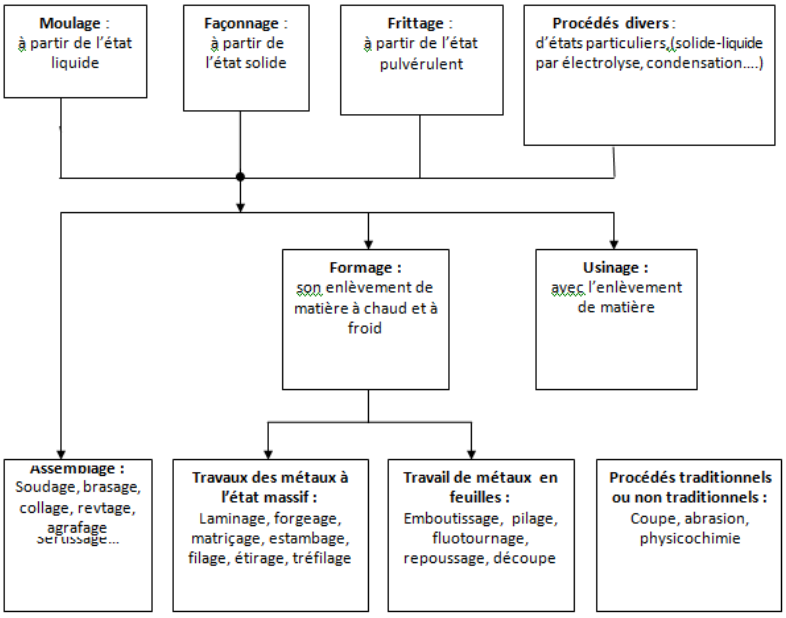


Figure 1.1 – Principaux procédés de mise en forme des matériaux métalliques

1.2 L'usinage par enlèvement de matière :

On appelle l'usinage ou mise en forme par enlèvement de matière destinée á conférer à une pièce une forme, des dimensions et un état de surface (écart de forme et rugosité) situés dans une fourchette de tolérance donnée.

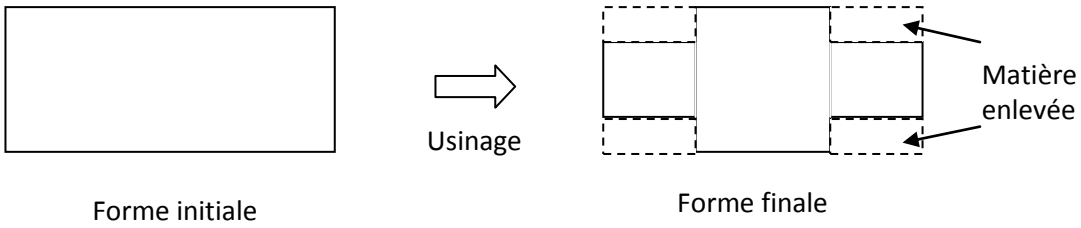


Figure 1.2 – Principe d'usinage par enlèvement de matière

On rencontre des difficultés analogues dans les substitutions de matériaux ; par exemple, les carrosseries de véhicules automobiles sont actuellement principalement constituées d'acier et leur réalisation en alliage d'aluminium ou en polymère, envisagée depuis des années pour alléger les véhicules et diminuer les consommations de carburant, se fait attendre [2]

1.2.1 L'usinage avec machine conventionnel :

Pour ce type d'usinage, les paramètres d'usinage et les réglages nécessaires ainsi la trajectoire d'outils sont faite manuellement sur la machine par l'opérateur. Pour cela, il utiliser les manivelles permettent de générer les mouvements suivent les axes ou directions. Les mouvements ne sont possibles que sur un seul axe á la fois.

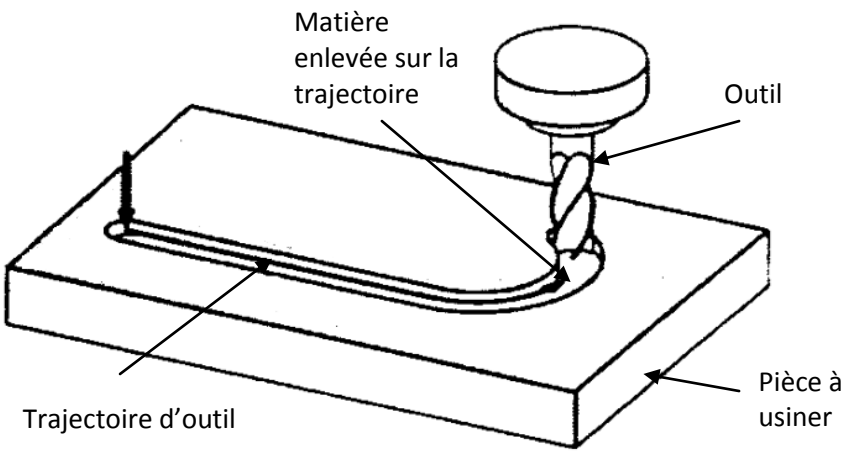


Figure 1.3 – Principe d'enlèvement de matière

Des moteurs permettent de choisir des vitesses d'avance suivant les axes de déplacement. Le choix de ces vitesse s'effectuer par l'intermédiaire d'une boite de vitesse mécanique

1.2.2 L'usinage avec machine à command numérique :

Une machine commande numérique, comme son nom l'indique, est composée en plus de la partie mécanique, d'une partie informatique qui la commande à travers des circuits électroniques. L'utilisation de ce type de machines nécessite de composer un codes alphanumériques pour représenter les instructions géométriques et technologiques nécessaires à sa conduite. C'est également une méthode d'automatisation des fonctions des machines ayant pour caractéristique principale une très grande facilité d'adaptation á des travaux différents.

A ce titre, la CN constitue l'un des meilleurs exemples de pénétration du traitement de l'information dans les activités de production. Exploitant au maximum les possibilités de la micro-informatique, toutes les données sont traitées en temps réel C'est-á-dire au moment où elles sont générées, de manière á ce que les résultats du traitement contribuent également á piloter le processus.
Après une première génération de CN à logique câblée sont apparues les commandes numériques par calculateur (CNC), ou par ordinateur, qui intègrent un ou plusieurs ordinateurs spécifiques pour réaliser tout ou partie des fonctions de commande.

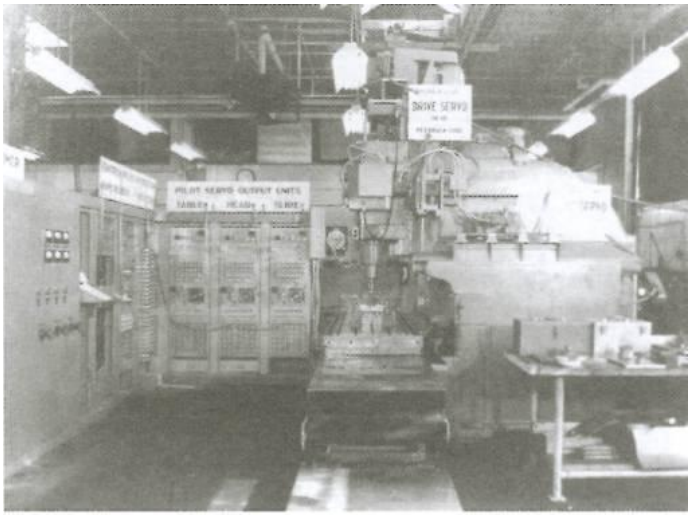


Figure 1.4 – La première machine à commande numérique qui occupe une salle [4]

Tous les systèmes de CN commercialisés actuellement contenant au moins un microprocesseur, les termes CN et CNC peuvent être considérés comme des synonymes. Le premier avantage d'une CN est d'offrir aux machines qui en sont équipées un très haut niveau d'automaticité. Sur de telles machines, l'intervention de l'opérateur nécessaire pour assurer la production de pièces peut être considérablement réduite voire supprimée.

De nombreuses MOCN peuvent ainsi fonctionner sans aucune assistance pendant toute la durée de leur cycle d'usinage, laissant l'opérateur libre d'accomplir d'autres tâches en dehors du poste de travail. Cette caractéristique présente par ailleurs un certain nombre d'avantages moins palpables mais tout aussi importants, tels qu'une diminution notable de la fatigue de l'opérateur, moins d'erreurs d'origine humaine et un temps d'usinage constant et prévisible pour chaque pièce d'une même série. Si l'on compare une MO conventionnelle et une MOCN, on peut considérer que le temps copeau est assez voisin sur les deux types de machines. En revanche, la productivité comparée de diverses catégories de machines de niveaux d'automatisation différents, c'est-á-dire ce même temps copeau ramené au temps effectif de production, est très différente compte tenu de la

réduction importante des temps non productifs que l'on enregistre sur les machines á fort taux d'automatisation.[4]

Figure 1.5 – Machine à commande numérique moderne

Le déplacement de l'outil sur la trajectoire d'usinage est d'écrit par l'opérateur á l'aide d'un programme. On utilise pour cela les coordonnées des différents points de passage de l'outil par rapport à la pièce. Les mouvements sont possibles sur plusieurs axes simultanément. Ces derniers sont générés par des moteurs qui permettent aussi de choisir des vitesses d'avances. Le choix de ces vitesses s'effectuer par un variateur. On dispose donc d'un large choix de vitesses [3].

Figure 1.6 – Pupitre de commande

Toujours dans une approche industrielle, il est intéressant d'observer la pièce de plus prés afin de visualiser l'enlèvement de matière proprement dit. Le but de cette simulation est de déterminer globalement le volume de matière enlevé pour chacune des phases d'usinage de la pièce.

La génération des trajectoires est réalisée le plus souvent á partir d'un logiciel de FAO. En fonction de la complexité des opérations à réaliser en usinage, il est difficile d'anticiper le comportement de la machine et, plus précisément, le comportement de l'outil coupant pendant ou en dehors de ces phases d'enlèvements de copeau.

Ainsi, grâce aux outils de simulation embarqués dans ces logiciels, l'utilisateur peut visualiser pas à pas ou de manière continue le déroulement de l'usinage de façon purement virtuelle. Plusieurs niveaux de simulation existent bien sûr :

– simulation uniquement de la trajectoire d'outil : dans ce cas, l'utilisateur voit défiler l'outil coupant dans l'environnement graphique superposé á la trajectoire qu'il souhaite réaliser

– simulation de l'enlèvement matière : dans ce cas, l'utilisateur voit l'outil coupant enlever de la matière au fur et á mesure que l'usinage se déroule. Des codes de couleurs permettent aussi de visualiser l'enlèvement matière d'une opération á une autre.

De nos jours, il existe des rendus de simulation beaucoup plus réalistes qui se concentrent plus sur la prédiction de l'aspect de surface en prenant en compte des phénomènes mécaniques très simples :

– point de rebroussement de l'outil coupant ;

– trajectoire d'outil avec angles vifs ;

– inversion des axes rotatifs...

Cette technique de simulation permet, entre autre, de visualiser et d'anticiper les défauts de surface liés purement à la stratégie programmée ou à la cinématique de la machine[6]

1.3 La programmation manuelle

La programmation manuelle consiste à écrire, ligne par ligne, les étapes successives nécessaires à l'élaboration d'une pièce donnée. Les instructions programmées doivent contenir toutes les données nécessaires à la commande et au séquencèrent des opérations à réaliser pour assurer l'usinage de la pièce sur la machine. Elles regroupent :

— les données géométriques, qui indiquent la forme et les dimensions de la pièce à usiner et permettent à la CN de calculer les positions successives de l'outil par rapport à la pièce pendant les diverses phases de l'usinage. Les positions sont définies par rapport à un repère connu. Certaines instructions viennent compléter les données géométriques en indiquant la nature du traitement numérique qu'elles doivent subir : le mode déplacement (d'interpolation linéaire ou circulaire), le choix du mode de cotation, absolue ou relative, le choix de l'outil, etc.

— les données technologiques, qui précisent les conditions de coupe optimales dans lesquelles pourront s'effectuer l'usinage. Elles concernent principalement la vitesse de rotation de la broche, les vitesses d'avance et la commande de l'arrosage.

Du point de vue structure un programme est composé de lignes qui constituent chacune un « Bloc » correspondant à une ou plusieurs instructions.[4]

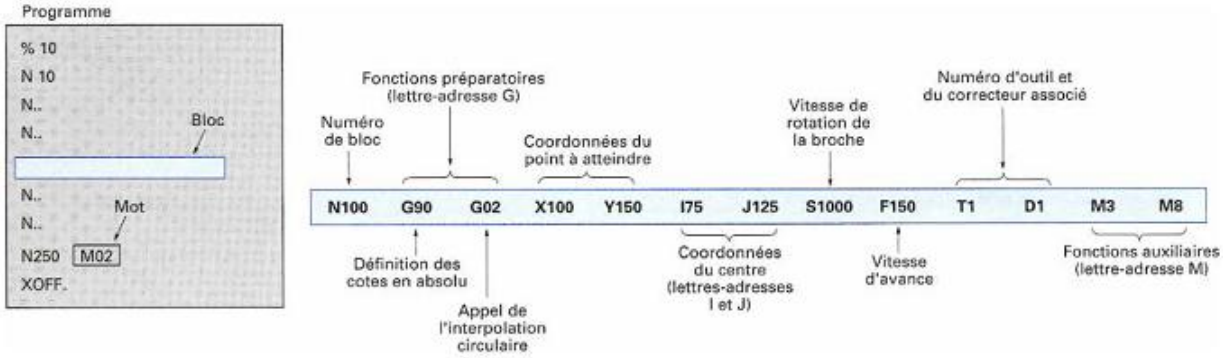


Figure 1.7 – composition d'un bloc [4]

1.3.1 Le bloc:

le «bloc» est l'unité de base du programme CN , il est vu sous forme imprimée comme une «ligne» du texte. Chaque bloc peut contenir un ou plusieurs «mots», qui consistent en une lettre, décrivant un réglage à effectuer, ou une fonction pour être effectuée, suivie d'un champ numérique, fournissant une valeur à cette fonction.

L'interpréteur accepte les mots commençant par n'importe qu'elle lettre, sauf N (qui désigne un numéro de ligne et doit être le premier).

Un exemple d'un bloc de programme :

/N0001 G0 X123.05

Ce bloc et construit avec les trois mots N0001, G0, et X123.05. La signification de chacun de ces mots sera décrite en détail ci-dessous. Puisque il commence avec une barre oblique « / », ce bloc sera considéré comme optionnel et ne sera pas pris en considération lors de l'exécution [4].

1.3.2 Axes et points de référence :

1.3.2.1 Convention d'axes

Conformément à la norme, les axes définissant les mouvements de translation principaux sont désignés par X, Y et Z.
L'axe Z Correspond à l'axe de la broche, le sens positif correspond à un accroissement de la distance entre la pièce et l'outil.
L'axe X Correspond à l'axe suivant ayant le plus grand déplacement, de sens direct (Règle des trois doigts de la main droite).
L'axe Y forme avec les deux autres un trièdre trirectangle, le sens positif correspond à un accroissement de la distance entre la pièce et l'outil.

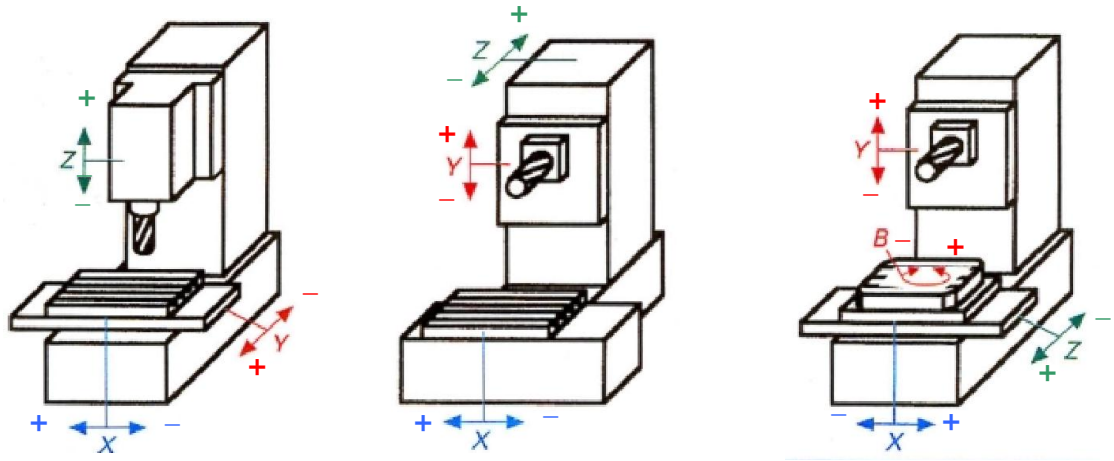


Figure 1.8 – Exemples de machines avec différentes configurations d'axes

Les axes de translations supplémentaires sont appelés ;

U parallèle à l'axe X
V parallèle à l'axe Y
W parallèle à l'axe Z

1.3.2.2 Points de référence :

Origine machine M :

L'origine machine M est fixée par le constructeur et ne peut être modifiée. Dans le cas du fraisage, il se trouve à l'origine du système de coordonnées machine et, dans le cas du tournage, sur la surface d'appui du nez de broche.

Origine pièce W :

L'origine pièce W, également appelée origine programme, est l'origine du système de coordonnées pièce. Il peut être choisie librement et devrait se trouver, en fraisage, là où la plupart des cotes ont leur origine sur le dessin. En tournage, l'origine pièce se trouve toujours sur l'axe de tournage et, généralement, sur la surface de dressage.

Point de référence R :

Le point de référence R est accosté pour définir l'origine du système de mesure, car l'origine machine ne peut, en général, pas être accosté. La commande et le système de mesure sont ainsi synchronisés.

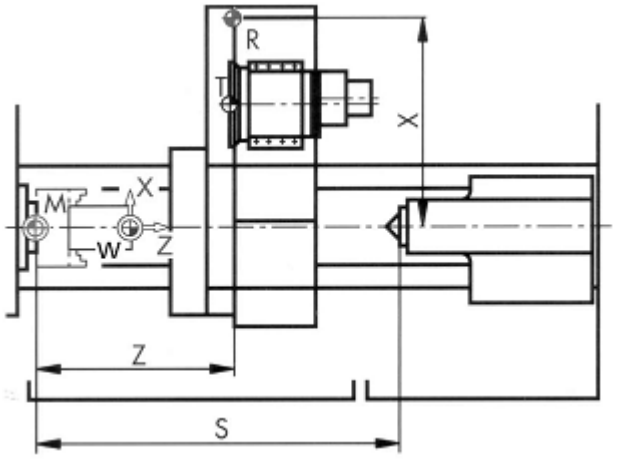


Figure 1.9 – points de références sur un tour

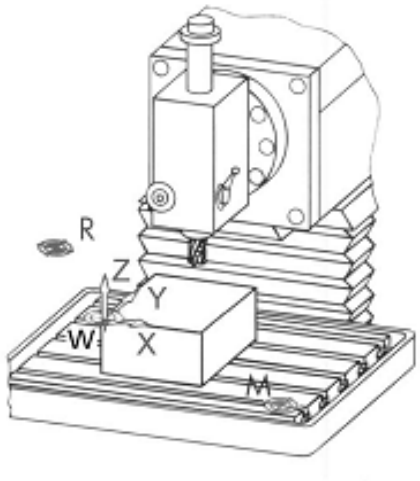


Figure 1.10 – points de référence sur une fraiseuse

1.3.3 Cotation relative et absolue :

1.3.3.1 Cotation absolue G90:

Les valeurs introduites se rapportent à l'origine pièce. Il faut toujours introduire les coordonnées absolues du point final dans le système de coordonnées actif (la position courante n'est pas prise en considération).

1.3.3.2 Cotation relative (incrémentales) G91:

Les valeurs introduites se rapportent à la position courante. Il faut toujours introduire les différences entre la position courante et le point final, en tenant compte du sens.

Exemple de programmation :

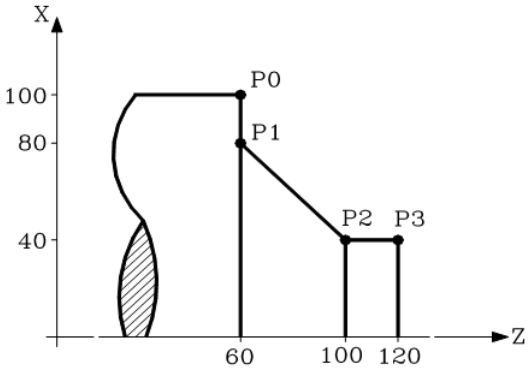


Figure 1.11 – parcours d'outil

Coordonnées absolues:

G90
X100 Z60 ; Point P0
X80 Z60 ; Point P1
X40 Z100 ; Point P2
X40 Z120 ; Point P3

Coordonnées relatives :

G90
X100 Z60 ; Point P0
G91
X-20 ; Point P1
X-40 Z40 ; Point P2
Z20 ; Point P3

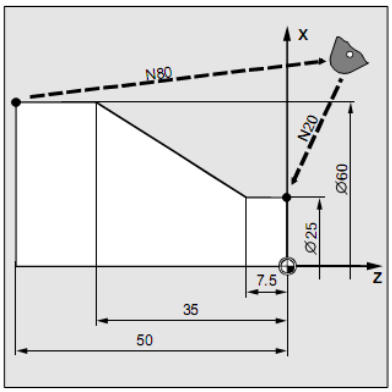
1.3.4 Instructions de déplacement

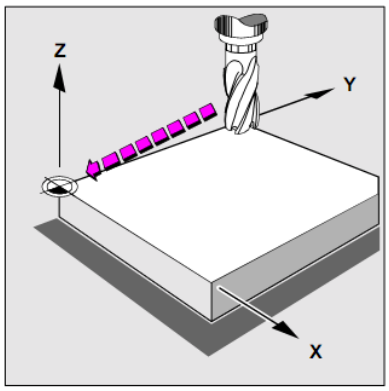
Cette partie décrit les instructions de positionnement et d'interpolation utilisées pour la commande de la trajectoire d'outil le long du contour programmé (par exemple une droite ou un arc de cercle).

1.3.4.1 Interpolation rapide (G00)

Le déplacement en rapide est utilisé pour le positionnement d'outil rapide, le contournement de pièce ou l'accostage des points de changement d'outil.

Le déplacement d'outil programmé avec G00 est exécuté à la plus grande vitesse de déplacement possible (vitesse rapide). La vitesse rapide est définie individuellement pour chaque axe dans le paramètre machine. Si le déplacement en rapide est exécuté simultanément dans plusieurs axes, la vitesse rapide sera déterminée par l'axe qui nécessite le temps le plus long pour effectuer sa part de trajectoire.
Format
G00 X... Y... Z...





N10 G00 X25 Z0

N10 G00 X0 Y0 Z3

Figure 1.12 – exemples de déplacements pour les cas tu tournage et fraisage

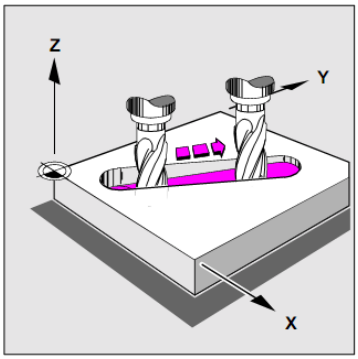
1.3.4.2 Interpolation linéaire (G01)

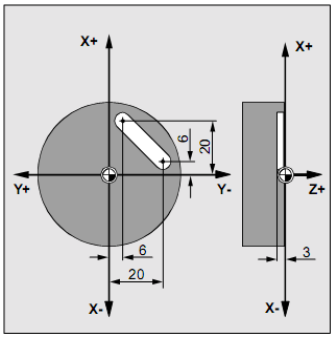
Avec G01, l'outil se déplace sur des droites parallèles aux axes, sur des droites obliques dans un plan ou sur des droites quelconques dans l'espace. L'interpolation linéaire permet, par exemple, de réaliser des surfaces 3D et des rainures.

Dans le cas de G01, l'interpolation linéaire est exécutée avec l'avance tangentielle. Les axes qui ne sont pas programmés dans le bloc contenant G01 ne se déplacent pas. La programmation de l'interpolation linéaire s'effectue comme dans l'exemple ci-dessus.
Format
G01 X... Y... Z... F... ;

1.3.4.3 Avance F pour axes à interpolation

La vitesse d'avance est indiquée sous l'adresse F. Une seule valeur F est programmable par bloc CN. L'avance F n'agit que sur les axes à interpolation et s'applique jusqu'à ce qu'une nouvelle valeur d'avance soit programmée





N10 G00 G90 X10 Y10 Z1
N20 G01 Z-12 F500
N30 X30 Y35 Z-3 F700

N10 G00 X40 Y-6 Z2
N20 G01 Z-3 F40
N30 X12 Y-20

Figure 1.13 – exemples de déplacements avec G01

1.3.4.4 *Interpolation circulaire (G02, G03)*

Avant la programmation du cercle (avec G02, G03), il est nécessaire de sélectionner le plan d'interpolation souhaité avec G17, G18 ou G19. Avec les 4ème et 5ème axes, l'interpolation circulaire est admise uniquement s'il s'agit d'axes linéaires.

Avec les fonctions énumérées ci-dessous, l'outil se déplace le long de l'arc de cercle spécifié dans le plan X-Y, Z-X ou Y-Z en respectant l'avance "F" spécifiée sur l'arc de cercle.

- Dans le plan X-Y :
  G17 G02 (ou G03) X... Y... R... (ou I... J...) F... ;
- Dans le plan Z-X :
  G18 G02 (ou G03) Z... X... R... (ou K... I...) F... ;
- Dans le plan Y-Z :
  G19 G02 (ou G03) Y... Z... R... (ou J... K...) F... ;

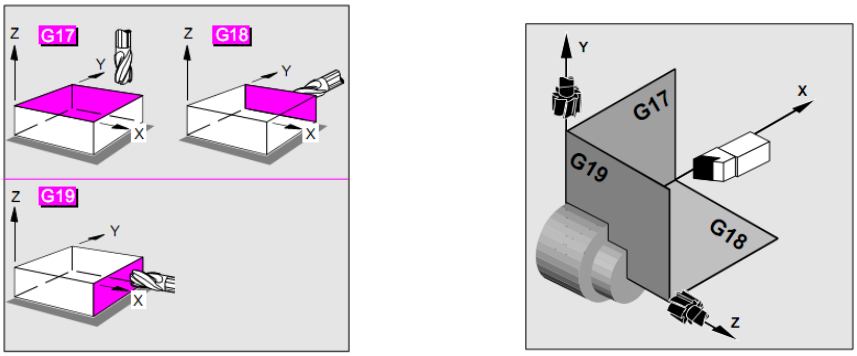


Figure 1.14 – Commande G pour la désignation des plans de travail

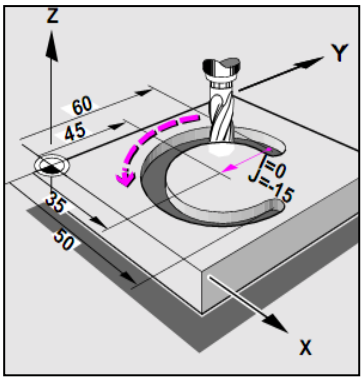
Format

L'exécution des fonctions figurant dans le tableau ci-dessous permet de démarre l'interpolation circulaire.

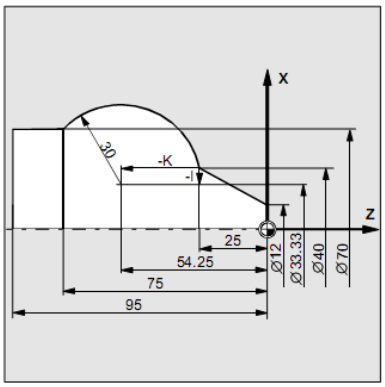
Elément	Fonction	Description
Désignation du plan	G17	Arc de cercle dans le plan X-Y
	G18	Arc de cercle dans le plan Z-X
	G19	Arc de cercle dans le plan Y-Z
Sens de rotation	G02	Sens horaire
	G03	Sens antihoraire
Position du point final	Deux axes parmi X, Y et Z	Position du point final dans un système de coordonnées pièce
	Deux axes parmi X, Y et Z	Distance entre le point de départ et le point final, avec signe
Distance entre le point de départ et le centre	Deux axes parmi I, J et K	Distance entre le point de départ et le centre du cercle, avec signe
Rayon de l'arc de cercle	R	Rayon de l'arc de cercle
Avance	F	Vitesse le long de l'arc de cercle

Tableau 1.1 : Fonctions pour l'exécution de l'interpolation circulaire

Avance
Pour l'interpolation circulaire, l'avance peut être indiquée de la même manière que pour l'interpolation linéaire (G01).



N5 G00 G90 X35 Y60
N10 G03 X50 Y45 I0 J-15 F500



N10 G00 X12 Z0
N20 G01 X40 Z-25 F0.2
N30 G03 X70 Z-75 I-3.335 K-29.25

Figure 1.15 – exemples de déplacements avec G03

1.3.5 Fonctions S, M, T

1.3.5.1 Fonction S et M de broche :

L'adresse S indique la vitesse de rotation de la broche en tr/min. M03 et M04 définissent le sens de rotation de la broche. M03 = sens de rotation de la broche à droite, M04 = sens de rotation de la broche à gauche M05 = arrêt de la broche.

- Les fonctions S ont un effet modal, ce qui signifie que, dès qu'elles sont programmées, elles restent actives jusqu'à la fonction S suivante. En cas d'arrêt de la broche avec M05, la fonction S est conservée. Si M03 ou M04 sont programmées ensuite sans indication d'une fonction S, la broche démarre à la vitesse de rotation programmée initialement.

- La limite inférieure de la fonction S (S0 ou fonction S proche de S0) dépend du moteur d'entraînement et du système d'entraînement de la broche et varie d'une machine à l'autre. Les valeurs négatives ne sont pas autorisées pour S.

1.3.5.2 Fonctions M utilisées pour interrompre des opérations (M00, M01, M02, M30) :

Ces fonctions déclenchent un arrêt de programme qui interrompt ou termine l'usinage. Selon les indications du constructeur de la machine, cet arrêt sera accompagné, ou non, d'un arrêt de la broche.

M00 (arrêt de programme)

Dans le bloc CN contenant M00, l'usinage s'arrête. Cet arrêt permet, par exemple, d'enlever les copeaux ou d'effectuer des mesures. Un signal est transmis à l'AP. Avec Départ programme, le programme reprend.

M01 (arrêt optionnel)

M01 arrête l'exécution du programme de la CN uniquement lorsque le signal correspondant de l'interface VDI a été mis à "1" ou qu'il a été activé dans HMI/boîte de dialogue "Influence sur le programme".

M30 ou M02 (fin de programme)

M30 ou M02 terminent le programme.

Fonction d'outil T :

La fonction T est utilisée pour faire l'appel de l'outil désigné par le nombre qui suit.

Exemple de programmation

N10 G17 G90 ; plan XY, cotation absolue

N20 F500 ; vitesse de coupe 500 mm/min

N30 G00 X10.00 Y5.00 ; déplacement rapide à (10,5)

N40 M03 ; démarrer la broche

N50 G04 P2.0 ; attendre 2 secondes

N60 G01 Z0 ; descendre l'outil

N70 X30.25 Y5.00 ; se déplacer en XY

N80 G03 X35.25 Y10.00 J5 ; déplacement séculaire anti-horaire

N90 G01 X35.25 Y50.10 ; déplacement linéaire

N100 G03 X30.25 Y55.10 I-5 ;déplacement séculaire anti-horaire

N110 G01 X10.00 Y55.10 ; déplacement linéaire

N120 G03 X5.00 Y50.10 J-5 ; déplacement séculaire anti-horaire

N130 G01 X5.00 Y10.00 ; déplacement linéaire

N140 G03 X10.00 Y5.00 I5 ; déplacement séculaire anti-horaire

N150 G01 Z5 ; sortir l'outil

N160 M05 ; arrêt de la broche

N170 G00 X0 Y0 ; retour rapide à l'origine

N180 M30 ; fin du programme

1.4 Génération automatique des programmes d'usinage:

Lorsque la géométrie des pièces devient complexe, qui est le cas le plus souvent rencontré, il sera difficile d'écrire manuellement son programme d'usinage. Pour cela on fait recoure généralement à des logicielle qui sont capable de calculer la trajectoire d'outil et générer le programme d'usinage en G code.

La stratégie généralement utilisée par ces logiciels, afin de simplifier le travail, est de convertir la surface continue de la pièce à une surface facettisée ou composée d'éléments de surface triangulaires puis déduire la trajectoire d'outil qui parcoure cette surface comme montré sur la figure.

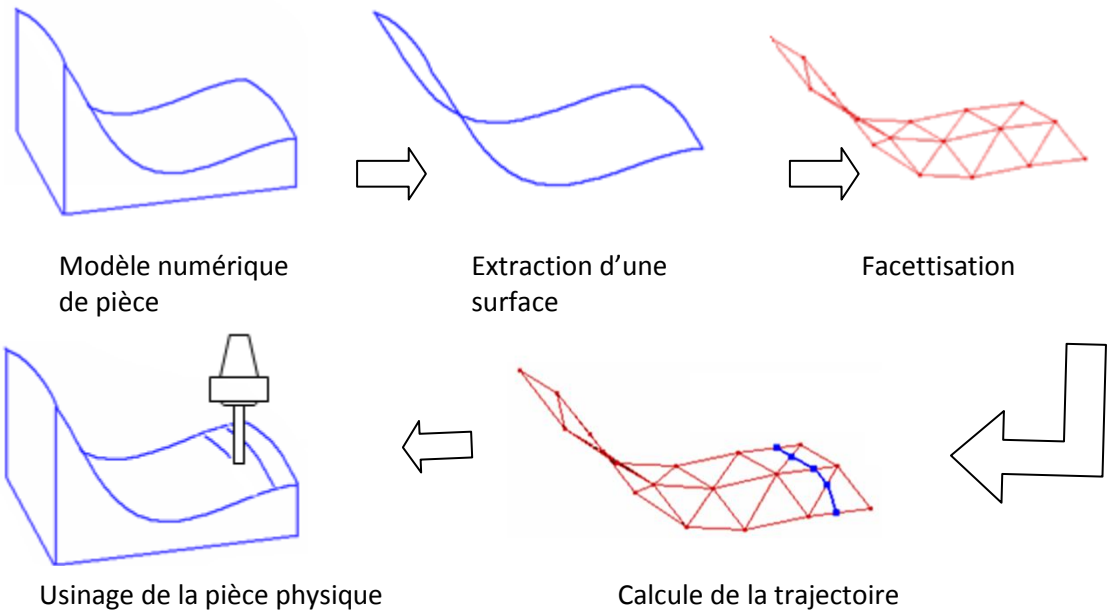


Figure1.16 Génération automatique d'une trajectoire d'outil pour une surface gauche

1.5 Interpolation polynomiale :

Les trajectoires d'outils et les programmes d'usinage déterminés manuellement ou générés automatiquement sont composés principalement de déplacement rectiligne en G01 c-à-dire composer de segment de droite jointe les un aux autres. Cette forme de trajectoire contient des discontinuités au niveau des points de jonctions. Ces discontinuité vont se répercuté sur la qualité finale de la surface usinée par des traces laissées sur cette dernière.
Pour faire face à ce problème, plusieurs machines modernes acceptent des trajectoires sous forme de courbes. Il s'agit de faire passer une courbe au plus prés des positions de l'outil posé sur la pièce.

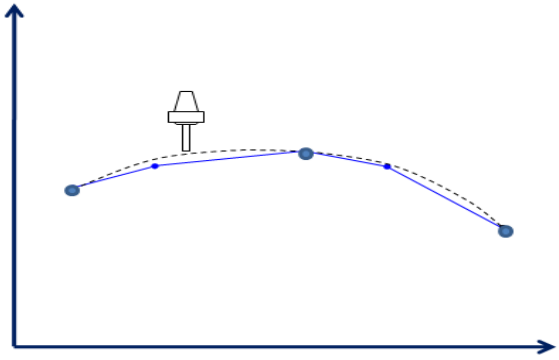
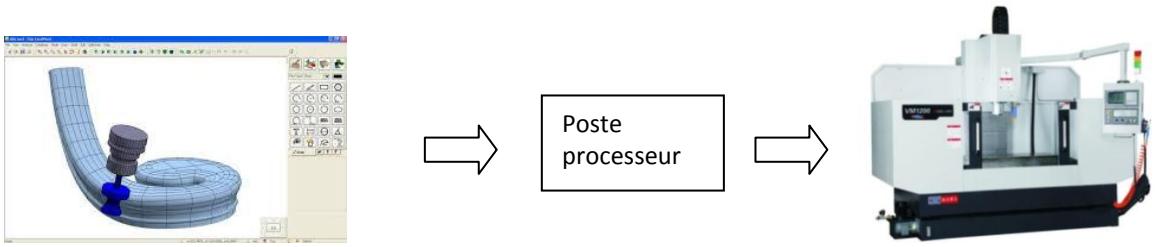


Figure 1.17 – Interpolation polynomiale

Les courbes sont généralement données sous forme de courbes BSplines, de NURS ou le polynôme canoniques de degré 3 ou 5. Mais ce format n'est pas défini dans le code ISO [ISO 6983] et chaque construction de CN a donc défini sa propre syntaxe.

1.5.1 L'interpolation polynomiale dans la chaine numérique :

Le recours au format polynomial peut se faire á différents niveaux dans la chaine numérique de programmation et d'exécution des trajectoires (figure). Quel que soit le processus adopté pour la génération des trajectoires, le modèle de référence est le modèle de CAO.
L'interpolation polynomiale peut alors être introduire à trois niveaux différents. Le premier niveau où on intègre directement le format polynomial lors de l'activité de génération de trajectoires. A ce niveau, les trajectoires ne sont décrites sous forme de lignes droites á aucun moment. La deuxième possibilité consiste à intégrer un post processeur qui modifier le format de description des trajectoires, passant d'une interpolation linéaire à une interpolation polynomiale, avant l'exécution sur la commande numérique [5].



Niveau 1
(Logiciel de génération de la trajectoire)

Niveau 2
(Logiciel ou programme poste processeur)

Niveau 3
(Machine à usiner)

Figure 1.18 – Les trois niveaux possibles où on intègre des trajectoires polynomiales

Enfin dans le troisième processus, une interpolation polynomiale à la volée est effectuée par la commande numérique à partir d'une trajectoire programmée en interpolation linéaire. La
CN détermine alors en temps réel la courbe d'interpolation d'un groupe de points. Cependant la géométrie pilotée lors de l'usinage ne peut être connue exactement. Pour augmenter la précision, il est donc préférable de réaliser l'association de courbes hors ligne c'est à dire suivant les niveaux 1 et 2.) [5].

1.5.2 Avantages du format polynomial :

Ce format permet de décrire une trajectoire sans discontinuités en tangence donc une meilleure trajectoire pour la dynamique de la machine. En effet, comme la courbe est continue en courbure, l'accélérateur de l'outil est continues et il n'ya plus de risques de vibrations ou de chocs dans l'asservissement. Des consignes de vitesse d'avance bien supérieures peuvent être atteintes, augmentant ainsi la productivité et la qualité de la coupe. Ce format et donc particulièrement adapté à l'usinage de formes complexes en grande vitesse.
Du point de vue de suivi de la trajectoire par la CN, la technique utilisée est la même que pour l'interpolation linéaire ou circulaire. Le trajet est décomposé en un ensemble de petits segments dont la longueur est déterminée par la fréquence d'échantillonnage de régulateur de position des axes de la machines. Les facettes engendrées par cette interpolation linéaire sont très petit et elles sont arasées par l'inertie de la masse de la machine et l'élasticité dans la chaine d'entrainement, ce

qui laisse un bon état de surface au niveau micrométrique. Notons également que la taille du programme réduite car le nombre d'instruction de déplacement pourra diminuer d'un facteur 5 à 30 [5].

1.5.3 Définition des formats d'interpolation :

Calculer un trajet sous la fourme des courbes polynomiales passe par le choix du format de la courbe. Pratiquement deux formats sont utilisables sur les commandes numériques: le format canonique et la format Bspline. Le format canonique permet de décrire tout arc de courbe en entier. [5]

Exemple de définition du Format canonique pour une commande Siemens Sinumerik 840D avec la fonction POLY :

POLY X=PO(xe,a2,a3,a4,a5) Y=PO(ye,b2,b3,b4,b5) Z=PO(ze,c2,c3,c4,c5) PL=n

POLY	Activation de l'interpolation polynomiale
PO[identificateur d'axes variables]=(_,_,_)	Point de départ de la courbe et les coefficients de polynômes
X, Y, Z	Nom d'axes
X_e, Y_e, Z_e	Point de départ de la courbe polynomiale
a2,a3,a4,a5	Coefficients du polynôme
PL	Largeur de l'intervalle dans lequel les polynômes sont définis.

1.5.4 Choix de la méthode d'association :

Nous avons d'abord abordé le problème sous la forme d'un problème d'interpolation ou d'approximation usuel d'un ensemble de points formant le trajet théorique. C'est alors un problème relativement simple de CAO, et la littérature propose un grand nombre de publications permettant de le résoudre selon différentes approches.

Le calcul de trajets d'usinage sous la forme de courbes polynomiales pose des problèmes selon les deux points de vue précédents. Il est en effet nécessaire de garantir une grande précision d'interpolation entre 1 et 10 µm et en même temps de contrôler l'évaluation de la courbure et l'absence d'oscillations.

De plus, la courbe calculée doit présenter un nombre de pôles minimum et des arcs de longueur maximum pour ne pas perturber artificiellement le comportement de la commande numérique. Enfin, l'algorithme retenu doit être robuste vis-à-vis de la géométrie du trajet, rapide et efficace. L'évaluation du résultat doit être automatique [5].

1.6 Problème à traiter et démarche de résolution proposée :

Pour ce travail, l'objectif est de réaliser un programme à intégrer dans un poste processeur qui va convertir les trajectoires d'un programme d'usinage, réalisées en G1, en une trajectoire polynomiale.

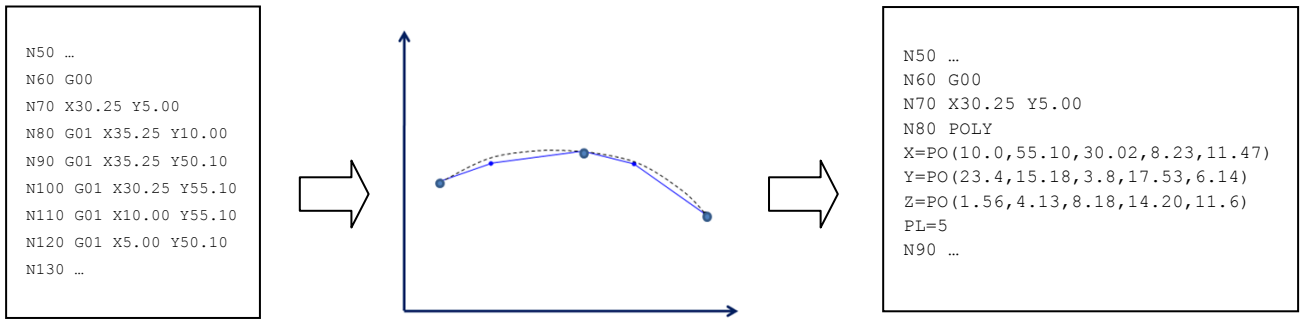


Figure 1.19 – Problème à traiter

Cela revient à convertir la trajectoire formée de segments de droite à une trajectoire lisse formée de courbes polynomiales. Ce qui consiste mathématiquement à effectuer une approximation ou une interpolation au sens mathématique des points G01 par une courbe mathématique.

Références :

[1] le site www.Usinage.pro

[2] Éric FELDER, procédés d'usinage-présentation, Techniques de l'Ingénieur, Edition technique 1997

[3] B.Vieille , cours d'usinage, Conservatoire nationale des Arts et Métiers

[4] Gilles PROD'HOMME, Commande numérique des machine outil, Techniques de l'Ingénieur, Edition technique 1996

[5] Christophe Tournier et coll, Usinage à grande vitesse, Edition technique 2010

[6] Fikret KALAY, Simulation numérique de l'usinage - Application à l'aluminium, Techniques de l'Ingénieur, édition technique, 2010

Chapitre 2

Méthodes d'interpolation et d'approximation des données

2.1 Introduction :

Dans les problèmes numériques, on substitue très souvent une fonction $f\ (x)$ connue en un nombre fini de points $x0, x1, \ldots, x_n$ par une fonction $P\ (x)$ plus simple et facilement calculable : c'est l'approximation. En termes mathématiques, l'approximation consiste à minimiser la distance qui sépare les fonctions $\boldsymbol{f}\ (x)$ et $P\ (x)$. L'interpolation impose de plus que les fonctions $\boldsymbol{f}\ (x)$ et $P\ (x)$ coïncident aux points x_j. Lorsque la fonction $P\ (x)$ représente la fonction $\boldsymbol{f}\ (x)$ décrite par un ensemble de points expérimentaux $(x_j, f\ (x_j))$, on parle de lissage. L'approximation d'une fonction est liée aux problèmes de représentation des fonctions comme limites de fonctions plus simples (développements en série, développements en série de Fourier, représentations intégrales, etc.). En pratique, on cherche à construire une suite de fonctions $f_n(x)$ qui converge vers la fonction de base $f\ (x)$. Lorsque les fonctions $f_n(x)$ sont des polynômes, on parle d'approximation polynomiale. L'approximation polynomiale est une des plus utilisées, car il est facile de rendre l'erreur d'approximation arbitrairement petite en augmentant le degré du polynôme.[1]

2.2 Méthodes d'interpolation :

2.2.1 Définition générale de l'interpolation :

Étant donner une fonction $\boldsymbol{f}$, une fonction $\boldsymbol{f}^*(x;\ a_0, a_1, \ldots a_n)$ dépond de x et $n + 1$ paramètre $a_0, a_1, \ldots a_n$ et un ensemble d'abscisses $x0, x1, \ldots x_n$ le problème d'interpolation consiste á déterminer les a_k de telle manière que les $n + 1$ équation :

$$\boldsymbol{f}^*(x;\ a_0, a_1, \ldots a_n) = f(\ x_k)\ =\ 1, 2, \ldots, n \tag{2.1}$$

Soient vérifiées.

Dans la suite nous utiliserons indifféremment les notations $\boldsymbol{f}^*\ (x_k)\ =\ \boldsymbol{f_k}$ chaque couple de valeur $(x_k, \boldsymbol{f_k})$ définit un point du plant qu'on l'appel un pivot ou un nœud.

il nous faut ensuite décider à quelle classe de fonctions appartient $\boldsymbol{f}$; le critère le plus important est la façon dont l'interviennent les paramètre a_k. On distingue les cas où f^* dépond linéairement des a_k de tous les autres. Une forme générale de f^* est

$$f^*(x) = \sum_0^n a_k \quad \varphi_k \tag{2.2}$$

Les qualités de l'interpolation (commodité de calcule, sensibilités aux erreurs d'arrondi par exemple) dépondront du choix des «fonctions de base» $\varphi_k(x)$.
L'interpolation polynomiale

$$\boldsymbol{f}^*(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \tag{2.3}$$

et l'interpolation trigonométrique

$$\boldsymbol{f}^*(x) = a0 + a_1 e^{ix} + a_2 e^{2ix} + \cdots + a_n e^{nix} \tag{2.4}$$

relèvent dans ce cas par contre l'interpolation rationnelle

$$f^*(x, a_0, a_1, \dots, a_m) = \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n}{b_0 + b_1 x + b_2 x^2 + \cdots + b_n x^n} \tag{2.5}$$

(qui par ailleurs fait appelé á $m + 2 + n$ coefficients) dépend non-linéairement des b_k
Bien que l'interpolation rationnelle soit parfois utilisé, nous nous limiterons ici á l'interpolation polynomiale qui n'implique que des calcules simples [2]

2.2.2 L'interpolation polynomiale :

Considérons *n* + 1 couples (x_i, y_i). Le problème est de trouver un polynôme $\Pi_m \in P_m$
appelé polynôme d'interpolation ou polynôme interpolant, tel que

$$\Pi_n(x_i) = a_0 + a1x_i + \cdots + a_m x^m = y_i \qquad i = 0, \dots, n \tag{2.6}$$

Les points x_i sont appelés nœuds d'interpolation Si $n \leq m$ le problème est sur ou sous- déterminé et si *n* = *m*, on a le résultat suivant :

Théorème 1. *Etant donné* $n + 1$ *points distincts* $x_0, \dots, x_n$ *et* $n + 1$ *valeurs correspondantes* $y_0, \dots, y_n$, *il existe un unique polynôme* $\Pi_n \in P_n$ *tel que :*

$$\Pi_n(x_i) = y_i \quad Pour \quad i = 0, \dots, n.$$

Preuve 1. *Pour prouver l'existence, on va construire explicitement* Π*.Posons*

$$l_i \in P_n : l_i(x) = \prod_{i=1, j \neq i} \frac{x - x_j}{x_i - x_j} \tag{2.7}$$
$$\text{Pour } i = 1, \dots, n$$

les polynômes $l_i, i = 0, \dots, n$ *forment une base de* P_n*. En décomposant* Π_n *sur cette base, on a* $\prod_n(x) = \sum_{j=0}^n b_i l_i(x)$

$$\prod_n(x) = \sum_{j=0}^n b_j \, l_j(x) = y_i \; i = 0, \dots, n \tag{2.8}$$

et comme $\boldsymbol{l_i(x_i) = \delta_{ij}}$*, on en déduit immédiatement que* $\boldsymbol{b_i = y_i}$*.*
Par conséquent, le polynôme d'interpolation existe et s'écrit sous la forme suivante :

$$\Pi_n(x) = \sum_{j=0}^n y_i l_i(x) \tag{2.9}$$

Pour montrer l'unicité, supposons qu'il existe un autre polynôme Ψ de degré m ≤ n, tel que $\Psi_m(x_i) = y_i$ Pour $i = 0, \dots, n$ La différence $\Pi_n - \Psi_m$ s'annule alors en n + 1 points distincts xi, elle est donc nulle. Ainsi $\Psi_m = \Pi_n$
On peut vérifier que

$$\prod_n(x) = \sum_{i=0}^{n} \frac{w_{n+1}(x)}{(x-x_j)w'_{n+1}(x)} y_i \qquad (2.10)$$

Où ω_{n+1} *est* le polynôme nodal de degré $n + 1$ défini par

$$\omega_{n+1}(x) = \prod_{i=0}^{n}(x - x_j) \qquad (2.11)$$

La relation 1.9 est appelée formule d'interpolation de Lagrange, et les polynômes $l_i(x)$ sont les polynômes caractéristiques de Lagrange. On a représenté sur le figure suivant les polynômes caractéristiques $l2(x), l3(x)$ et $l4(x)$, pour n = 6, sur l'intervalle [-1,1] avec des nœuds est qui repartis (comprenant les extrémités de l'intervalle).

Noter que | $l_i(x)$ | peut être plus grand que 1 dans l'intervalle d'interpolation.

Si $y_i = f(x_i)$ pour $i = 0, \ldots, n, f$ étant une fonction donnée, le polynôme interpolation $\Pi_n(x)$ sera noté $\Pi_n(x)f(x)$. [2]

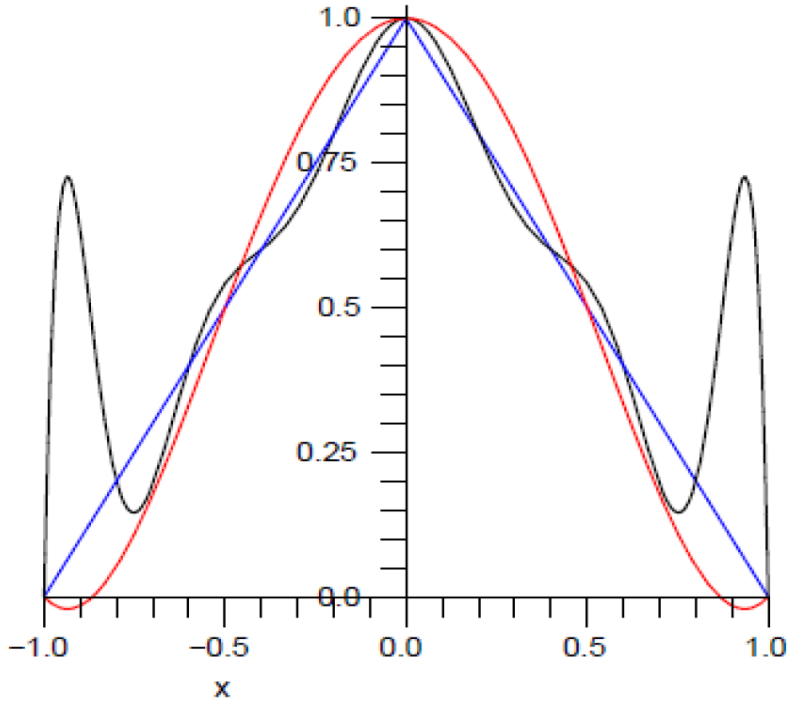


Figure 2.1 Tracée de la fonction "triangle" et des interpolations polynomiales de degré 4 (courbe rouge) et 10 (courbe noire).

2.2.3 Le polynôme d'interpolation de Lagrange :

L'approximation polynomiale, fondée en général sur le développement en série de Taylor, permet d'approcher une fonction f suffisamment régulière par un polynôme de degré n.

Rappelons que la série de Taylor d'une fonction peut ne pas converger et que, si elle converge, elle peut converger vers une quantité différente de la fonction initiale (par exemple, $f(x) = exp\left(-\frac{1}{x^2}\right)$ au voisinage de l'origine). Publiée pour la première fois par Brook Taylor (1685-1731) en 1715, puis reprise par Joseph-Louis Lagrange (1736−1813), et démontrée avec reste intégral par Augustin-Louis Cauchy (1789−1857), la formule de Taylor conduit á une estimation de l'erreur

dans l'approximation d'une fonction par un polynôme de Lagrange.[1]
Dans cette méthode nous faisons l'hypothèse que le polynôme d'interpolation est s'écrit par *n*+1 pivots (x_i, y_i)

$$\prod_n(x) = \sum_{i=0}^{n} l_i(x) y_i \qquad (2.12)$$

Chaque polynôme élémentaire de Lagrange $l_i(x)$ et de degré n et par conséquence Π est un polynôme de degré au plus égale á n. Vous voyez que le problème de l'interpolation sera résolu si nous pouvons former des aux conditions

$$l_i(x_k) = \delta_{i,k} \qquad i,k = 1,\dots,n \qquad (2.13)$$

$\delta_{,ij}$ est le symbole Kronecker, qui vaut 1 lorsque ses deux indices sont égaux et qui vaut 0 dans les autres. Si par exemple $x = x_2$ alors :

$$l_0(x_2) = l_1(x_2) = l_3(x_2) = \cdots = l_n(x_2) = 0$$

tandis que $l_2(x_2) = 1$.
Des $n+1$ terme de Π_n ne subsiste que y_2.
Ainsi pour le pivot x_2 le polynôme d'interpolation coïncide avec la fonction et il en est de même pour tout autre nœud. Le polynôme élémentaire l_i s'annule pour tous les pivots sauf x_i; il est donc proportionnel au polynôme

$$q(x) = (x - x_0)(x - x_1)\dots(x - x_{i-1}))(x - x_{i+1})\dots(x - x_n).$$

La constante proportionnalité s'obtient en

$$q(x_i) = (x_i - x_0)(x_i - x_1)\dots(x_i - x_{i-1})(x_i - x_{i+1})(x_i - x_{i+1}) : \; l_i$$

Est égale au polynôme normalisé

$$q(x)/q(x_i)$$

le polynôme d'interpolation de Lagrange est maintenant entièrement déterminer ;
il s'écrit

$$\Pi_n(x) = \sum_{i=0}^{n} l_i(x) y_i = \sum_{i=0}^{n} f_i \prod_{k=0 k\neq j}^{n} \frac{x - x_k}{x_i - x_k}$$

Le calcul numérique de $\Pi_n(x)$ se programme aisément á l'aide de deux boucles imbriquées; il est bien plus rapides que la résolution de système linéaire du précédent. Le fragment de pro- gramme ci-dessous réalise ce calcul.
Les deux premières instructions créent un vecteur d'abscisses et un vecteur de zéros, lequel va recevoir les valeurs successives du polynôme d'interpolation.
Á la ligne 4, nous remplissons la ligne i de la matrice L par des 1, le gros du travail est fait ligne 7,óu nous calculons á l'aide d'une itération les valeurs du polynôme élémentaire l_i,pour tout les abscisses simultanément. Nous avons utilisé ce programme pour interpoler les fonctions e^x sur le pivot d'abscisse [0,5;−1;1.5;2] le figure a montre le résultat.
Nous avons représenté la chacune des qualités$l_i(x)y_i(x)$, ainsi que leur somme $\Pi_n(x)$.Vous pouvez vérifier les $l_i(x)y_i(x)$ passent par un pivot et un seul alors que $\Pi_n(x)$ passe par tous les nœuds

```
x = linspace(xmin,xmax,npts);                                    1
pl = zeros(1,npts);                                              2
for i = 1:4                                                      3
   L(i,1:npts) = ones(1,npts);                                   4
   for k = 1:4                                                   5
      if k <> i                                                  6
         L(i,:) = L(i,:).*(x-xp(k))/(xp(i)-xp(k));               7
      end                                                        8
   end                                                           9
   L(i,:) = L(i,:)*fn(xp(i));                                    10
   pl = pl + L(i,:);                                             11
end                                                              12
```

Figure 2.2 calcul du polynôme de Lagrange de degré 3(4 pivots)

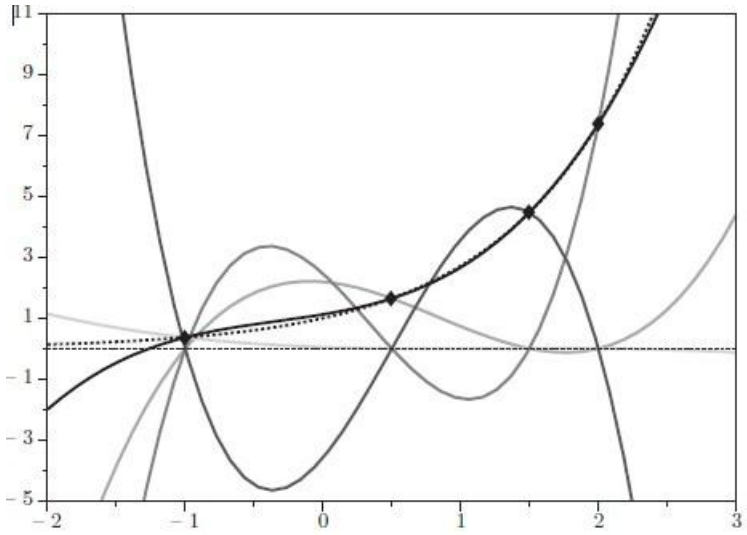


Figure 2.3 L'interpolation de Lagrange

Cependant, il serait pénible de développer de polynôme de Lagrange et de regrouper les termes pour le mètre sous la forme d'une somme ordonnée selon les puissances de x. [2]

2.2.4 Le polynôme d'interpolation d'Hermite :

Charles Hermite (1822-1901) a généralisé l'interpolation de Lagrange en faisant coïncider non seulement P_n et f aux points x_i, mais aussi leurs dérivées d'ordre k_i aux points x_i.[1]
Dans le but de construire une fonction d'interpolation f^* encore plus préciser nous pouvons imposer á celle-ci des conditions supplémentaire. Hermite a proposé que les dériver de f^* coïncident avec les dérivées correspondantes de f pour certains pivots. Nous allons suivre cette idées, en nous limitons á l'interpolation polynômiale et on ne considérera que les dériver premières. Pour tout les pivots nous imposons que :

$$\Pi_n(x_i) \equiv f_i \quad \Pi_n{}'(x_i) \equiv f_i{}'(x_i) = f_i{}' \tag{2.14}$$

En d'autres termes, $\Pi_n(x_i)$ interpole $f(x)$ et $\Pi_n{}'(x_i)$ interpole $f'(x)$ sur les mêmes pivots.
Nous supposons que le polynôme d'interpolation de Hermide, ainsi défini peux s'écrit sous une forme analogue á celle de Lagrange

$$\Pi_n(x) = \sum_{i=0}^{n} h_i(x) f_i + \sum_{i=0}^{n} \overline{h}_i(x) f_i' \qquad (2.15)$$

Où les hi et $\overline{h}$i sont des polynômes réels distincts. Quel est en le degré ?

Le polynôme $\Pi_n$doit satisfaire aux $2n + 2$. Condition (2.14) : il comporte donc $2n + 2$ coefficients ajustables est il de degré 2n + 1. Les polynômes élémentaire hi et $\overline{h}$i ont donc un degré au plus égale a 2n + 1.

Les polynômes hi sont tel que Π interpole f il obéisse donc à condition identiques á celle que nous avons écrites pour les polynômes élémentaire de Lagrange. Il parait de même raisonnable que les polynômes dérives soient choisis de façon á que Π'_n interpole f' d'où une nouvelle séries de condition. Mais cela ne se fait pas : il ne fait pas que les h'_i viennent perturber l'interpolation de f, ni que les $\overline{h_i}$ détruisent l'interpolation def' : ces polynômes doivent être nuls sur les pivotes nous arrivons ainsi á quatre séries de conditions

$$\begin{array}{rcll} h_i(x_k) & = & \delta_{i,k} & i,k = 0,1,\ldots,n \quad (a) \\ h_i'(x_k) & = & 0 & i,k = 0,1,\ldots,n \quad (b) \\ \overline{h}_i(x_k) & = & 0 & i,k = 0,1,\ldots,n \quad (c) \\ \overline{h'}_i(x_k) & = & \delta_{i,k} & i,k = 0,1,\ldots,n \quad (d) \end{array} \qquad (2.16)$$

Chaque polynôme élémentaire obéit á 2n + 2 conditions et peut être de degré 2n + 1 au plus. Commençons par construire le polynôme $\overline{h}_i$. Il s'annule ainsi que dérivée sur tous les pivots $k \neq i$, il admet donc des racines doubles en ces points et aussi une racine simple en x_i

Il s'exprime facilement á l'aide de polynôme élémentaire de Lagrange $l_i(x)$ sur les mêmes pivots et des facteurs $x - x_i$

$$\overline{h}_\iota(x) = C\,[l_i(x)]^2(x - x_i)$$

La dérivée $\overline{h'}_\iota(x_i)$ vaut 1 (condition$2.16, d$), ce qui impose$C \equiv 1$.

$$h_i(x) = C[l_i(x)]^2 t_i(x)$$

Le polynôme $h(x)$admet tous les pivots sauf x_i comme racines doubles il peut donc s'écrire :
en désignant par t_i un polynôme de premier degré tel que h_i respecte tous les conditions $(2.16, a, b)$ en x_i

$$h_i(x) = C[l_i(x)]^2 t_i(x) = t_i(x) = 1\ h_i'(x_i) = 2l_i(x_i)l_i'(x_i) + l_i^2(x_i)t_i'(x_i) = 0$$

Un calcule sans difficulté on montre que :

$$t_i(x) = 1 - 2(x - x_i)l_i'(x_i)$$

[2]

Exemple 1. (Polynôme d'interpolation osculateur)

Posons $\mathrm{m_i} = 1$ pour $\mathrm{i} = 0,..,\mathrm{n}$. Dans ce cas $\mathrm{N} = 2\mathrm{n} + 2$ et le polynôme d'Hermite est appelé polynôme osculateur. Π est donné par

$$H_{N-1}(x) = \sum_{i=0}^{n} \left(y_i A_i(x) + y_i^{(1)} B_i(x) \right)$$

Remarquer que

$$l_i'(x_i) = \sum_{k=0,k\neq 1}^{n} \frac{1}{x_i - x_k} \quad pour i = 0 \ldots n$$

On calcule les polynômes d'interpolation de Lagrange et d'Hermite de la fonction f(x) =sin (4πx) sur l'intervalle [0,1] en prenant quatre nœuds èquirèpartis (n=3).
La Figure montre que les graphes de la fonction f (trait discontinu) et des polynômes $\Pi_n f$(pointillés) et H_{N-1}(trais plein) [3].

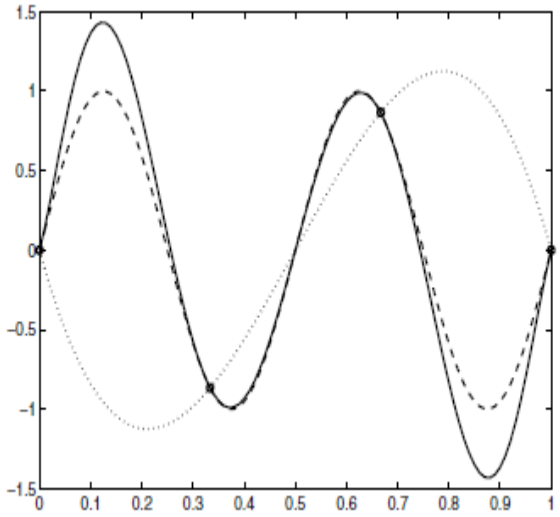


Figure 2.4 – Interpolation de Lagrange et d'Hermite de la fonction f (*x*) = *sin* (4π*x*) sur l'intervalle [0,1]

2.3 Méthodes d'approximation :

La théorie de l'approximation constitue une partie fondamentale de l'analyse numérique. De nombreuses questions étudiées dans les paragraphes précédents peuvent se formuler dans le cadre de cette théorie : approximation d'une fonction par un polynôme d'interpolation, d'une intégrale par une somme finie, de la solution d'une équation différentielle, etc. Nous allons don- ner les notions de base de la théorie de l'approximation ; elles feront appel à un minimum de connaissances en analyse fonctionnelle ; nous nous intéresserons plus á des exemples pour mon- très la richesse de cette théorie et la puissance de l'outil que constitue l'analyse fonctionnelle. Le premier mathématicien á attiré l'attention sur l'intérêt que pouvait présenter l'analyse fonctionnelle dans le développement de l'analyse numérique fut le Russe L.V. Kantorovitch en 1948. Les ides et les méthodes de l'analyse fonctionnelle jouent un rôle important, voire fondamental, en analyse numérique quand les mathématiques du problème dépendent beaucoup de l'analyse fonctionnelle (par exemple, dans les équations aux dérivées partielles),lorsque l'on cherche à traiter d'un seul coup une classe entière de méthodes (par exemple, les méthodes de quadrature de type interpolation) ou encore á démontrer l'existence de méthodes numériques présentant certaines caractéristiques. L'analyse fonctionnelle apporte alors une simplification importante ; elle joue, par contre, un rôle moins fondamental pour étudier un algorithme précis ou pour résoudre un problème spécifique et ne sera d'aucune utilité en ce qui concerne l'implémentation d'une méthode sur ordinateur. Actuellement, l'analyse fonctionnelle est un outil essentiel pour comprendre bon nombre de méthodes d'analyse numérique, mais on peut également trouver de nouvelles méthodes numériques sans son secours. Réciproquement, certains algorithmes dé- coulent directement des méthodes de l'analyse fonctionnelle. On doit la considérer comme un outil privilégié pour résoudre certains problèmes

d'analyse numérique, mais on ne peut pas, inversement, considérer l'analyse numérique comme une branche de l'analyse fonctionnelle ; ce sont deux domaines différents mais complémentaires puisque l'analyse numérique peut suggérer l'étude de nouvelles questions d'analyse fonctionnelle et que, inversement, celle-ci est le pivot de certains sujets d'analyse numérique [4].

2.3.1 Meilleure approximation :

Déterminer la meilleure approximation n'est pas toujours facile. Elle dépend de la topologie des espaces mis en œuvre. Soit E un espace métrique, A un sous-ensemble de E et f un élément de E. On dit qu'un élément φ de E est une meilleure approximation de f si

$$\| f - \varphi \| = d(f, A) = \inf_{a \in A} \| f - a \|$$

On démontre que si A est compact, alors pour toute fonction f de E il existe au moins une meilleure approximation. En effet, soit f un élément quelconque de E, notons d la distance de f á E et considérons la suite (a_n) d'éléments de A tels que

$$\lim_{n \to \infty} \| f - a_n \| = d(f, A) = d$$

D'où on déduit l'inégalité

$$\forall \varepsilon > 0 \quad \exists k_1 > 0 \quad n > k_1 \Longrightarrow \quad \| f - a_n \| \leq d + \varepsilon$$

Comme A est compact, cette suite admet une limite φ par conséquent

$$\forall \varepsilon > 0 \quad \exists k_2 > 0 \quad n > k_2 \Longrightarrow \quad \| a_n - \varphi \| \leq \varepsilon$$

En additionnant les deux inégalités, $\forall \varepsilon > 0 \exists k > 0$

$$n \geq k \Longrightarrow \| f - u \| \leq \| f - a_n \| + \| a_n - \varphi \| \leq d + \varepsilon$$

On en déduit que

$$\| f - \varphi \| \leq \| f - a \| \qquad \forall a \in A$$

La limite de la suite est donc une meilleure approximation.
De ce résultat, on déduit que si E est un espace vectoriel de dimension finie et A est un sous-espace vectoriel de E, alors pour toute fonction f de E, il existe au moins une meilleure approximation.
En effet, soit B le sous-ensemble de E

$$B = \{ b \in A \quad \| b \| < 2 \| f \| \}$$

Cet ensemble est non vide (car il contient au moins 0), fermé, borné : donc B est compact. D'après le résultat précédent, il existe au moins une meilleure approximation de f, notée ϕ et

vérifiant

$$\| f - \varphi \| \leq \| f - b \| \qquad \forall b \in B$$

Considérons maintenant un élément a de A qui n'appartient pas á B, on a

$$a \in A \backslash B \quad \| a - f \| \geq \| a \| - \| f \| > \| f \| \geq \| f - \varphi \|$$

ce qui permet d'étendre l'inégalité précédente á tous les éléments de A donc a est une meilleure approximation de f.

Soit E un espace vectoriel normé contenant l'espace P_n des polynômes de degré inférieur ou égal á n. Pour toute fonction f de E, il existe au moins un polynôme $p_n \in P_n$ que :

$$\| f - p_n \| = \inf_{p \in P_n} \| f - p \|$$

Le polynôme p_n est appelé polynôme de meilleure approximation pour la norme de E. Il est facile de vérifier ce résultat. Il existe une suite g_n telle que

$$\lim_n \| f - g_n \| = \inf_{p \in E} \| f - p \|$$

Une telle suite est bornée car $\|g_n\| \leq \|f - g_n\| + \|f\|$ et comme P_n est de dimension finie, on peut extraire une sous-suite convergeant vers un polynôme p_n qui minimise la distance de f á P_n.

Remarque 2. L'unicité de la meilleure approximation est un problème difficile. Cette unicité n'est pas toujours vérifiée. Par exemple, si E est l'ensemble des fonctions continues sur $[-1, 1]$ et intégrables sur cet intervalle $\mathrm{E} = \mathrm{l}^1$ et A est l'ensemble des polynômes de degré inférieur ou égal á 1 $\mathrm{A} = \mathrm{P}_1$.
La fonction $f(x) = 1 \ si \ x > 0, et \ -1 \ si \ x < 0$ *admet pour meilleure* approximation toutes les fonctions constantes $\varphi(x) = c$ avec $c \in [-1, 1]$ et dans ce cas :

$$\| f - \varphi \| = \int_{-1}^{1} | f(x) - \varphi(x) | \, dx = 2$$

On démontre que lorsque E est l'espace des fonctions continues sur un intervalle $[\mathrm{a}, \mathrm{b}]$, $\mathrm{E} = \mathrm{C}[\mathrm{a}, \mathrm{b}]$ et A est l'espace des polynômes de degré n, $\mathrm{A} = \mathrm{P_n}$ le polynôme de meilleure approximation est unique, mais lorsque E est l'espace de Lebesgue $\mathrm{E} = \mathrm{L}^2[\mathrm{a}, \mathrm{b}]$ et $\mathrm{A} = \mathrm{P_n}$, ce polynôme n'est pas toujours unique. Lorsque $\mathrm{E} = \mathrm{C}[\mathrm{a}, \mathrm{b}]$ est muni de la convergence uniforme, on parle de meilleure approximation uniforme, de Tchebychev, ou minimax. Lorsque $\mathrm{E} = \mathrm{L}^2[\mathrm{a}, \mathrm{b}]$ on parle de meilleure approximation quadratique ou meilleure approximation au sens des moindres carrés. On démontre le résultat suivant qui introduit la constante de Lebesgue. Soit E un espace vectoriel normé, A un sous-espace vectoriel de E de dimension finie. Soit u l'opérateur linéaire de $\mathrm{E} \rightarrow$ a défini par u(f) =φ oú φ est la meilleure approximation de f et vérifiantu$(\mathrm{a}) = \mathrm{a} \forall \ \mathrm{a} \in \mathrm{A}$. Pour toute fonction f de E, l'erreur d'approximation est majorée par

$$\| f - u(f) \| \leq (1 + \| 1 \|) d(f, A)$$

Le nombre $\|1\|$ est appelé constante de Lebesgue. Il mesure l'amplification de l'erreur. La vérification est immédiate. Soit ϕ la meilleure approximation de f sur A. Comme $\varphi \in A$ on a $u(\varphi) = \Phi$. Par conséquent

$$\| f - u(f) \| < \| f - \varphi \| + \| u(f - \varphi) \| \leq (1 + \| u \|) d(f, A)$$

2.3.2 Approximation au sens des moindres carrées :

Il faut distinguer le cas continu et le cas discret. Soit $C\ [a\ ,b\ ]$ l'espace des fonctions continues sur $[a\ ,b\ ]$ et soit ω une fonction poids strictement positive sur $]a\ ,b\ [$. Nous supposerons que existe pour toute fonction f de $C\ [a\ ,b\ ]$ et nous définirons le produit scalaire par :

$$(f, g) = \int_a^b = f(x) g(x) \omega(x) dx$$

Soit $g_0, \ldots, g_n$ des éléments linéairement indépendants de H et soit C le sous-espace vectoriel qu'ils engendrent. D'après ce qui précède, on a :

$$\overline{g} = a_0 g_0 + \cdots + a_n g_n$$

où les a_i sont solutions du système :

$$\sum_{i=0}^{n} a_i \int_a^b g_i(x) g_j(x) \omega(x) dx = \int_a^b f(x) g_j(x) \omega(x) dx \text{ pour } j = 0, \ldots, n$$

- Si $g_i(x) = x_i$, on retrouve l'approximation polynomiale, ce qui montre l'importance des familles de polynômes orthogonaux.

•Si $g_0(x) = 1, g_1(x) = \sin(x)$; $g_2(x) = cos(x), \ldots, g_{2n-1}(x) = sin(nx)$; $g_{2n}(x) = cos(nx); a = -\pi \text{ et } b = \pi$

On est dans le cas de l'approximation trigonométrique. $g_0, \ldots, g_{2n}$ forment un système orthonormée :

$$\overline{g}(x) = a_0 \sum_{k=1}^{n} (a_k cos(kx) + b_k sin(kx))$$

Étudions maintenant l'approximation discrète au sens des moindres carrés. Définissons le produit scalaire par :

$$(f, g) = \sum_{i_0}^{N} f(x_i) g(x_i) \omega(x_i)$$

où les x_i sont distincts les uns des autres et $\omega(x_i) > 0$.

Soit C le sous-espace engendré par des éléments $g_0, \ldots, g_n$ linéairement indépendants. On aura avec les a_i solutions du système :

$$\sum_{i=0}^{n} a_i \sum_{k=0}^{N} gi(x_k) g_j(x_k) \omega(x_k) = \sum_{k=0}^{N} f(x_k) g_j(x_k) \omega(x_k) \text{ pour } j = 0, \ldots, n$$

Posons :

$$b = \left(\sqrt{\omega(x_0)} f(x_0), \ldots \sqrt{\omega(x_N)} f(x_N) \right)^T$$

$a = (a_0, \ldots, a_n)^T$

$$\mathcal{A} = \begin{pmatrix} \sqrt{\omega(x_0)}g_0(x_0) \ldots \sqrt{\omega(x_0)}g_n(x_0) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \sqrt{\omega(x_N)}g_0(x_N) \ldots \sqrt{\omega(x_N)}g_n(x_N) \end{pmatrix}$$

Le système précédent s'écrit $A^T Aa = A^T b$. La matrice $A^T A$ étant symétrique définie positive, la méthode de Cholesky cf.

Prenons maintenant pour H l'espace vectoriel $H^q\,[a,b](q \geq 1)$ des fonctions définies sur [a, b] qui ont une dérivée $(q-1)^{iéme}$ absolument continue et une dérivée $q^{ème}$ de carré sommable, c'est-á-dire que :

$$\int_a^b (f^{(q)}(x))^2 dx < +\infty$$

On considère le produit scalaire :

$$(f,g) = \int_a^b f^{(q)}(x)g^{(q)}(x) + c\sum_{i=0}^{N} f(x_i)g(x_i)$$

Où les x_i sont distincts les uns des autres et oú c est un paramétre non négatif. Si C est un sous-espace vectoriel de $H^q\,[a,b]$, les résultats théoriques précédents s'appliquent. On dit alors que g est la fonction spline d'ajustement de f.[4]

2.3.3 Approximation uniforme

Le théorème de Karl Weierstrass (1815-1897), établi en 1885, affirme que toute fonction continue sur un intervalle I peut être approchée uniformément par une suite de polynômes. Le problème est de trouver un moyen de construire cette suite de polynômes. On considère l'espace vectoriel des fonctions continues de [a, b] dans R muni de la norme uniforme

$$\| f \| = \sup_{x\in[a,b]} | f(x) |$$

et on note

$$d(f, P_n) = \inf_{p\in P_n} \| f - p \|$$

la distance de f á l'ensemble des polynômes de degr'e inférieur ou égal á n.

Soit f une fonction de $C([a,b])$ continue sur l'intervalle $[a,b]$ et á valeurs dans R. Pour tout entier n naturel, il existe un et un seul polynôme q_n qui réalise le minimum de

$$\| f - q_n \| = d(f, P_n).$$

Ce polynôme est appelé polynôme de meilleure approximation uniforme. Dans le cas complexe, le théorème de Mergelyan établit que si A est un compact de C, f une fonction continue sur A et analytique sur l'intérieur de A, f admet une meilleure approximation polynomiale si et seulement si le complémentaire de A est connexe.

On dit que $f \in C(a,b])$ équioscille sur $(n+1)$points de l'intervalle [a , b] s'il existe des points $x_0 < x_1 < \cdots < x_n$ tels que $|f(x_i)| = \|f\|$

Pour i = 0, . . . , n et $f(x_{i+1}) = f(x_i)$ pour $i = 0, \ldots, n-1$.
On démontre que si $f \in C(a,b])$ est une fonction continue, le polynôme de meilleure approximation $q_n \in P_n$ de f est l'unique polynôme de degré n tel que $(f - q_n)$ équioscille sur au moins $(n+2)$ points de l'intervalle $[a,b]$ et que si f est une fonction analytique de série entière de rayon R centrée au point $(a+b)/2$ alors les polynômes d'interpolation p_n convergent uniformément vers f
si

$$R > (\frac{1}{c} + \frac{1}{2})(b-a)$$

la constante c étant donnée par $c = 1$ si les points p_n sont quelconques, $c = e$ si les points x_i sont équidistants, et $c = 4$ si les points x_i sont les points de Tchebychev. Rappelons qu'une fonction $f : I \to R$ est lipchitzienne de rapport k ou k-lipchitzienne si

$$\forall x, y \in I \mid f(x) - f(y) \mid \leq k \mid x - y \mid$$

L'ensemble des fonctions k-lipchitziennes est un espace vectoriel. Muni de la norme uniforme, l'espace $Lip(I)$ est un espace de Banach (espace vectoriel normé complet). Une fonction $f : I \to R$ est holderiénne de coefficient α $avec$ $0 < \alpha \leq 1$ et de rapport K si

$$\forall x, y \in I \mid f(x) - f(y) \mid \leq k(\mid x - y \mid)^{\alpha}$$

L'ensemble des fonctions höldériennes $Lip_{\alpha, k}(I)$ de coefficient α de rapport K est un espace vectoriel. Muni de la norme uniforme, c'est un espace de Banach.
Le module de continuité d'une fonction f est la fonction $\omega_f : R^+ \to R^+$ définie par

$$\forall x, y \in [a,b] \quad \mid f(x) - f(y) \mid \leq \omega_f(\mid x - y \mid)$$

Ce module vérifie différentes propriétés.

$$\forall x, y \in [a,b] \quad \mid f(x) - f(y) \mid \leq \omega_f(\mid x - y \mid)$$

La fonction $t \to \omega_f(t)$ est une fonction croissante et la limite vérifie

$$\lim_{t \to 0^+} \omega_f(t) = 0$$

La fonction $\omega_f(t)$ est une fonction croissante et la limite vérifie

$$\forall s, t \in \mathbb{R}^+ \quad \omega_f(s+t) \leq \omega_f(s) + \omega_f(t)$$

et vérifie

$$\forall n \in \mathbb{N} \quad \forall t > 0 \quad \omega_f(nt) \leq n\omega_f(t)$$

ainsi que la propriété suivante

$$\forall \alpha > 0 \quad , \forall t > 0 \omega_f(\alpha t) \leq (1+\alpha)\omega_f(t)$$

Le module de continuité permet de caractériser différentes classes de fonctions. La fonction f est uniformément continue sur I si et seulement si :

$$\lim_{t\to 0} \omega_f(t) = 0$$

La fonction f est lipchitzienne de rapport k si et seulement si

$$\omega_f(t) \leq k\omega_f(t)$$

La fonction f est höldérienne d'ordre α pour $0 < \alpha < 1$ si et seulement si

$$\omega_f(t) \leq kt^{\alpha}$$

Le module de continuité permet d'établir un résultat de Jackson. Soit $f \in C([a,b])$, il existe des polynômes p_n de degré n appelés polynômes de Jackson vérifiant

$$\| f - p_n \| \leq 3\omega_f(\frac{b-a}{n+2})$$

La constante de Lebesgue Λ_n est de l'ordre de

$$\Lambda_n \simeq \frac{2^{n+1}}{en\ln(n)}$$

si les points (x_i) sont équidistants et

$$\Lambda_n \simeq \frac{2}{\pi}\ln(n)$$

si les points (x_i) sont les points de Tchebychev. Ce résultat permet de montrer que si f est k-lipchitzienne, les polynômes d'interpolation de Tchebychev convergent vers f sur l'intervalle [a,b].[2]

2.3.4 Phénomène de Runge :

Dans le domaine mathématique de l'analyse numérique, le phénomène de Runge désigne le problème qui peut survenir lorsqu'on tente d'approcher une fonction á l'aide de polynômes d'interpolation. Contrairement á ce que l'on aurait pu penser, lorsque le degré du polynôme interpolateur augmente, les oscillations de ce dernier au bord du domaine s'intensifient. Il y a malgré tout convergence du polynôme vers la fonction sur un intervalle réduit (que nous expliciterons plus tard) par rapport au domaine de définition initial. Ce phénomène a été observé par le mathématicien Carle David Tomé Runge alors qu'il étudiait l'erreur d'approximation entre une fonction et ses polynômes interpolateurs. Considérons la fonction :

$$f(x) = \frac{1}{1-25x^2} \quad x \in [-1,1]$$

Runge (1856 – 1927) a montré que si cette fonction est interpolée aux points et équidistants x_i entre -1 et

$$x_i = -1 + (i-1)\frac{2}{n} \quad i = 0, \ldots, n$$

par un polynôme P_n de degré ≤ n alors

$$\lim_{n \to +\infty} \left(\max_{-1 \leq x \leq 1} \mid f(x) - P_n(x) \mid \right) = \infty$$

Lorsqu'on augmente le nombre de points, on constate que le polynôme se met à osciller forte- ment entre les points xi avec une amplitude de plus en plus grande.

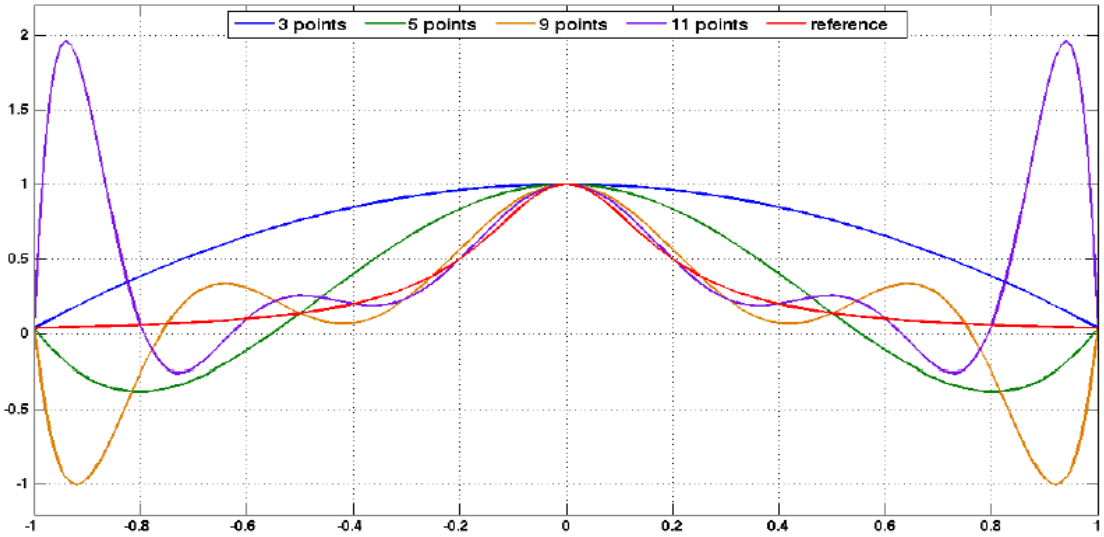


Figure 2.5 – Phénomène de Runge

2.3.5 Interpolation et approximation par morceaux :

Cherchons à améliorer la qualité de l'interpolation d'une fonction donnée, nous sommes tentés d'augmentée le nombre n de nœuds. Comme vous le savez cette solution est souvent vouée á l'échec, par ce que le terme d'erreur contient deux facteurs, la polynôme P(x) et la dérivée d'ordre n ou 2n de la fonction qui sont vrais semblablement rapidement croissent avec n.
On peut cependant augmenter à volonté le nombre de nœud sons faire croître l'ordre d'interpolation. La figure ci dessus montre le principe de ce tout petit miracle. Pour interpoler la fonction représentée en noirs, nous avons défini 5 pivots (d'abscisses -1.8,-0.5,0,0.5,1.4)et, ce faisant, 4 intervalles. Nous pratiquons une interpolation du premier degré dans chaque intervalle ; la fonction linéaire correspondante est définit pour tout x : elle représentée par un trait noir. Cependant, nous n'utilisons que la restriction de cette fonction à l'intervalle considéré, c'est à dire que nous assimilons l'arc de courbe au segment représenté en trait gris épais [2].

la fonction d'interpolation f^* est maintenant bien appliquée elle admet une définition différente dans chaque intervalle. De plus sa dérivée n'est plus continue. Ce type d'interpolation peut être rendus aussi précis que l'on veut en multipliant le nombre d'intervalles. On peut aussi perfectionner en choisissant une méthode plus précise dans chaque intervalle, interpolation parabolique ou interpolation de Hermite.

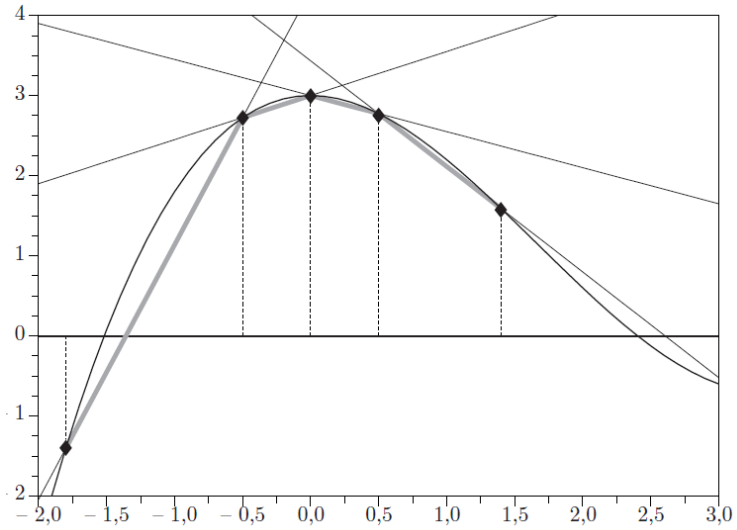


Figure 2.6 – l'interpolation par morceaux

Cependant, dans certaines application, la disparition de la continuité de la drivée qui pose problème. Il existe un formalisme qui combine l'avantage d'une interpolation par morceaux et la continuité des dérivées de la fonction d'interpolation : c'est la méthode de Splines, que nous expliquons dans la paragraphe ci-dessous.

2.3.5.1 Les Splines*:*

En introduisant les fonctions splines dans les années 40, Schoenberg apporta plus de souplesse dans l'approximation polynomiale. Il permit de diminuer le degré du polynôme approchant la fonction en considérant des fonctions polynomiales par morceaux appelées splines, du nom de la tige flexible qu'on fixait sur le papier pour tracer des courbes lisses. Les polynômes de Serge Bernstein (1880-1968) servent dans la construction paramétrique des B-splines. Nous envisageons ici le cas des splines cubiques.[1]
Définition : Soient$x_0, \ldots, x_n$, $n+1$ noeuds distincts de [a,b], avec $a = x_0 < x_1 < \ldots < x_n = b$.
La fonction $s_k(x)$ sur l'intervalle [a,b] est une spline de degré k relative aux noeuds x_j si

$$s_k\,|_{[x_j,x_{j+1}]} \in \mathbb{P}_k \quad i = 0, 1, \ldots, n-1 \tag{2.17}$$

$$s_k \in C^{k-1}[a,b] \tag{2.18}$$

Si S_k désigne l'espace des splines s_k définies sur [a,b] et relatives à $n+1$ nœuds distincts, alors $dim\, S_k = n + k$. Evidemment, tout polynôme de degré k sur [a, b] est une spline ; mais en pratique, une spline est constituée de polynômes différents sur chaque sous-intervalle. Il peut donc y avoir des discontinuités de la dérivée k-iéme aux nœuds internes $x_1, \ldots, x_{n-1}$. Les nœuds où se produisent ces discontinuités sont appelés nœuds actifs. On vérifie facilement que les conditions 2.17 et 2.18 ne sont pas suffisantes pour caractériser une spline de degré k.
En effet, la restriction $\boldsymbol{s_{k,j}=s_k}|[\boldsymbol{x_j}, \boldsymbol{x_{j+1}}]$ peut être écrite sous la forme

$$s_{k,j} = \sum_{i=0}^{k} s_{ij}(x - x_i)^i \;\; si\; x \in [x_j, x_{j+1}] \tag{2.19}$$

on doit donc déterminer (k +1)n coefficients $\boldsymbol{s_{ij}}$. D'autre part, d'après2.18,

$$s_{k,j-1}^{(m)}(x_j) = s_{k,j}^{(m)}(x_j), \;\; j = 1, \ldots, n \quad m = 0, \ldots, k-1$$

ce qui revient à fixer k(n-1) conditions. Il reste par conséquent $(k+1)n-k(n-1)=k+n$ degrés de liberté.
Même si la spline est une spline d'interpolation, c'est-à-dire telle que $\mathbf{s_K}(\mathrm{x_j}) = \boldsymbol{f_j}$ pour $j = 0,\ldots,n$, où $f_0,\ldots,f_n$ sont des valeurs données, il reste encore k-1 degrés de liberté à fixer.
Pour cette raison, on impose d'autres contraintes qui définissent :

1. Les splines périodiques, si

$$s_k^{(m)}(a) = s_k^{(m)}(b), m = 0, 1, \ldots, k-1 \qquad (2.20)$$

2. Les splines naturelles, si pour k=2l-1, avec l ≥ 2

$$s_k^{(l+j)}(a) = s_k^{(l+j)}(b) = 0 \quad j = 0, 1, \ldots, l-2 \qquad (2.21)$$

On déduit de 2.19 qu'une spline peut être représentée à l'aide de k+1 splines de base. Le choix le plus simple qui consiste à utiliser des polynômes convenablement enrichis n'est pas satisfaisant sur le plan numérique car le problème est alors mal conditionné. Dans les Sections 1.2.5.2 et 1.2.5.3, nous donnerons des exemples de splines de base : les splines cardinales pour k=3 et les B-spline pour k quelconque.[3]

2.3.5.2 Splines cubiques :

Les splines d'interpolation cubiques sont particulièrement importantes car :ce sont les splines de plus petit degré qui permettent une approximation C^2 elles ont de bonnes propriétés de régularité (á condition que la courbure soit assez faible)
Considérons donc, dans [a,b], n+1 noeuds $a = x_0 < x_1 < \cdots < x_n = b$ et les valeurs correspondantes $f_{i,i} = 0,\ldots,n.$
Notre but est de d'définir un procédé efficace pour construire la spline cubique interpolant ces valeurs. La spline étant de degré 3, sa dérivée seconde doit être continue.
Introduisons les notations suivantes

$$f_i = s_3(x_i), \quad m_i = s_3'(x_i), M_i = s_3''(x_i) \quad i = 1, \ldots, n$$

Comme $s_{3,i-1} \in P_3, s''_{3,i-1}$ est linéaire et

$$s''_{3,i-1}(x) = M_{i-1}\frac{(x_i - x)}{h_i} + M_i\frac{(x - x_{i-1})}{h_i} \text{ pour } x \in [x_{i-1}, x_i] \qquad (2.22)$$

où $h_i = x - x_{j-1}$ pour i=1,...,n En intégrant deux fois (2.22) on obtient

$$s_{3,i-1}(x) = M_{i-1}\frac{(x_i - x)^3}{6h_i} + M_i\frac{(x - x_{i-1})^3}{6h_i} + C_{i-1}(x - x_{i-1}) + \overline{C}_{i-1}$$

les constantes $C_i - 1$ et $\overline{c}_{i-1}$ sont déterminées en imposant les valeurs aux extrémités

$$s_3(x_i - 1) = f_i - 1 \; et \; s_3(x_i) = f_i$$

Ceci donne, pour $i = 1,\ldots,n-1$

$$\overline{C}_{i-1} = f_{i-1} - M_{i-1}\frac{h_i^2}{6}, C_{i-1} = \frac{f_i - f_{i-1}}{h_i} - \frac{h_i}{6}(M_i - M_{i-1}).$$

Imposons á présent la continuité de la dérivée première en x_i ; on obtient

$$s_3'(x_i^-) = \frac{h_i}{6}M_{i-1} + \frac{h_i}{3}M_i + \frac{f_i - f_{i-1}}{h_i}$$

$$= \frac{h_i + 1}{3}M_i + -\frac{h_i + 1}{6}M_{i+1} + \frac{f_{i+1} - f_i}{h_{i+1}} = s_3'(x_i^+)$$

Où $s_3'(x^{\pm}) = \lim_{t\to 0} s_3'(x_i \pm t)$ Ceci conduit au syst'eme lin'eaire suivant (appel'e syst'eme de M-continuités)

$$\mu_i M_{i-1} + 2M_i + \lambda_i M_{i+1} = d_i \quad i = 1, \ldots, n \tag{2.23}$$

On a pose:

$$\mu_i = \frac{h_i}{h_{i+1}}, \quad \lambda_i = \frac{h_{i+1}}{h_i + h_{i+1}},$$

$$d_i = \frac{6}{h_i + h_{i+1}}\left(\frac{f_{i+1} - f_i}{h_{i+1}} - \frac{f_i - f_{i-1}}{h_i}\right) i = 1, \ldots, n-1.$$

Le système (2.23) a $n+1$ inconnues et $n-1$ équations ; 2 conditions restent donc á fixer. En général, ces conditions sont de la forme

$$2M_0 + \lambda_0 M_1 = d_0, \mu_n M_n - 1 + 2M_n = d_n$$

Où $0 \leq \lambda_0, \mu_n \leq 1$ et d_0, d_n sont des valeurs données.
Par exemple, pour obtenir les splines naturelles (satisfaisant $s_3''(a) = s_3''(d) = 0$
On doit annule les coefficients ci-dessus. Un choix fréquent consiste á poser $\lambda_0 = \mu_n = 1$ et $d_0 = d_1$, $d_n = d_{n-1}$
Ce qui revient á prolonger la spline au déjà des points extrêmes de l'intervalle [a, b] et á traiter a et b comme des points internes. Cette stratégie donne une spline au comportement "régulier".
Une autre manière de fixer λ_0 et μ_n (surtout utile quand les valeurs $f'(a)$ et $f'(b)$ ne sont pas connues) consiste á imposer la continuité de $s_3'''(x)$ en x_1 et x_{n-1}. Comme les nœuds x_1 et $x_n - 1$ n'interviennent pas dans la construction de la spline cubique, celle-ci est appelée spline not-a-knot, avec pour nœuds "actifs" $\{x_0, x_2, \ldots, x_n - 2, x_n\}$ et interpolant f aux nœuds $\{x_0, x_1, x_2, \ldots, x_n - 2, x_n - 1, x_n\}$.
En général, le système linéaire obtenu est tri diagonal de la forme.

$$\begin{bmatrix} 2 & \lambda_0 & 0 & & & \ldots & 0 \\ \mu_1 & 2 & \lambda_1 & & & & \\ 0 & \ddots & \ddots & \ddots & & & \vdots \\ \vdots & & & & \mu_{n-1} & 2 & \lambda_{n-1} \\ 0 & & \ldots & & 0 & \mu_n & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ \vdots \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix} \tag{2.24}$$

Une approche complètement différente pour construire s_3 consiste á se donner une base $\{\varphi_i\}$ de l'espace S_3 (de dimension $n + 3$) des splines cubiques. Nous considérons ici le cas où les $n +$

3 fonctions de base ϕ_i ont pour support tout l'intervalle [a,b], et nous renvoyons á la 1.3.2 pour le cas où le support est "petit".
On définit les fonctions ϕ_i par les contraintes d'interpolation suivantes

$$\varphi_i(x_i) = \delta_{ij}, \quad \varphi_i'(x_0) = \varphi_i'(x_n) = 0 \text{ pour } i, j = 0, \ldots, n$$

et il faut encore définir deux splines φ_{n+1} et φ_{n+2}. Par exemple, si la spline doit satisfaire des conditions sur les dérivées aux extrémités, on impose

$$\begin{aligned} \varphi_{n+1}(x_j) &= 0 \quad j = 0, \ldots, n \\ \varphi_{n+2}(x_j) &= 0 \quad j = 0, \ldots, n \end{aligned}$$

$$\begin{aligned} \varphi'_{n+1}(x_0) &= 1 \quad \varphi'_{n+1}(x_n) = 0 \\ \varphi'_{n+2}(x_0) &= 0 \quad \varphi'_{n+2}(x_n) = 1 \end{aligned}$$

La spline a alors la forme suivante

$$s_3(x) = \sum_{i=0}^{n} f_i\varphi_i(x) + f_0'(x)\varphi_{n+1}(x) + f_n'\varphi_{n+2}(x)$$

où f_0' et f_n' sont deux valeurs données. La base obtenue $\{\varphi, i = 1, \ldots, n\}$ appelée base de splines cardinales, est souvent utilisée dans la résolution numérique d'équations différentielles ou intégrales. La Figure 7.9 montre une spline cardinale typique calculée sur un intervalle théoriquement) non borné où les nœuds d'interpolation x_j sont les entiers. La spline change de signe entre chaque intervalle $[x_j - 1, x_j]$ *et* $[x_j, x_j + 1]$ et décroît rapidement vers zéros.

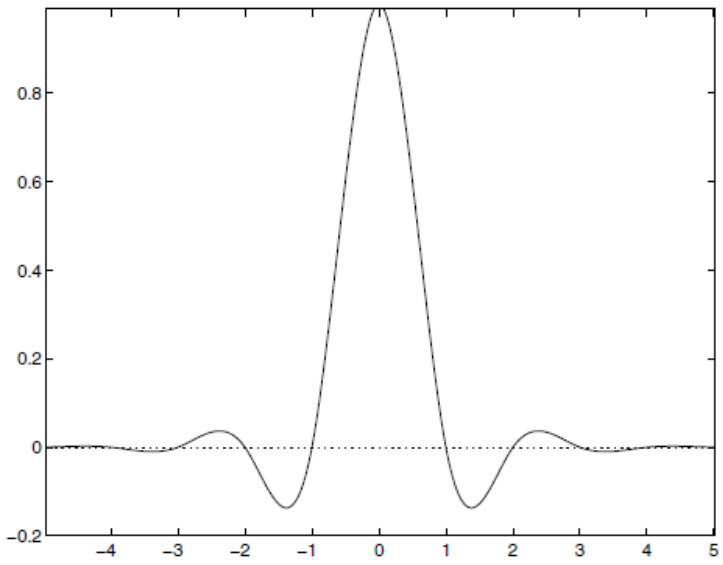


Figure2.7 – Spline cardinale

Propriété 1. *Soit* $f \in C^0$ *et soit* s_3 *la spline cubique naturelle interpolant f. Alors*

$$\int_a^b [s_3''(x)]^2(x)dx \leq \int_a^b [f''(x)]^2(x)dx \tag{2.25}$$

avec égalité si et seulement *si f* = s_3

Le résultat ci-dessus, appelé propriété de la norme du minimum, est encore valable si, au lieu des conditions naturelles, on impose des conditions sur la dérivée première de la spline aux extrémités (dans ce cas, la spline est dite contrainte). La spline d'interpolation cubique s_f d'une fonction $f \in C^2([a,b])$, avec $s_f(a) = f(a) et\ s_f(b) = f(b)$, satisfait également

$$\int_a^b [f''(x) - s_f''(x)]^2(x)dx \leq \int_a^b [f''(x) - s''(x)]^2(x)dx \quad \forall s \in S_3$$

En ce qui concerne l'estimation de l'erreur, on a le résultat suivant :

Propriété 2. Soit $f \in C^4([a,b])$ et soit une partition de $[a,b]$ en sous intervalles de longueur h_i. On note $h = \max_i h_i$ et $\beta = h/\min_i h_i$. Soit s_3 la spline cubique interpolant f. Alors

$$\| f^{(r)} - s_3^{(r)} \|_\infty \leq C_r h^{4-r} \| f^{(4)} \|_\infty \qquad r = 0,1,2,3 \tag{2.26}$$

avec $C_0 = 5/384$, $C_1 = 1/24$, $C_2 = 3/8$ *et* $C_3 = (\beta + \beta - 1)/2$. *Par conséquent, la spline* s_3 *ainsi que ses dérivées première et seconde convergent uniformément vers* f *et vers ses dérivées quand h tend vers zéro. La dérivée troisième converge également, à condition que* β *soit uniformément borné*

Exemple 2. *La Figure (2.8) montre la spline cubique approchant la fonction de l'exemple de Runge, et ses dérivées premières, seconde et troisièmes, sur une grille de 11 nœuds quirépartis. On a indiqué dans (la Table 1) l'erreur* $\|s_3 - f\|_\infty$ *en fonction de h ainsi que l'ordre de convergence p. Les résultats montrent clairement que p tend vers 4 (l'ordre théorique) quand h tend vers zéro.*

Table 1. *Erreur d'interpolation commise pour la fonction de Runge avec des splines cubiques :*

h	1	0.5	0.25	0.125	0.0625
$\|s_3 - f\|_\infty$	0.022	0.0032	2.7741e-4	1.5983e-5	9.6343e-7
p		2.7881	3.5197	4.1175	4.0522

2.3.5.3 B-splines:

Nous revenons maintenant aux splines quelconques de degré *k*, et nous allons définir la base de *B*-splines (ou bell-splines).

définition 2. *On définit la B-spline normalisée* $B_{i,k+1}$ *de degré k relative aux nœuds distincts* $x_i, \ldots, x_{i+k+1}$ *par :*

$$B_{i,k+1}(x) = (x_{i+k+1} - x_i)g[x_i, \ldots, x_{i+k+1}], \tag{2.27}$$

Où

$$g(t) = (t-x)_+^k = \begin{cases} (t-x)^k & si\ x \leq t, \\ 0 & sinon. \end{cases} \tag{2.28}$$

En substituant $\left(f[x_0,\dots,x_n]=\sum_{1=0}^{n}\frac{f(x_i)}{\omega_n'+1(x_i)}\right)$ (fonction de Newten) dans (1.27), on obtient l'expression explicite suivante :

$$B_{i,k+1}(x)=(x_{i+k+1}-x_i)\sum_{j=0}^{k+1}\frac{(x_{j+i}-x)_+^k}{\prod\limits_{l=0,l\neq j}^{k+1}(x_{i+j}-x_{i+l})} \qquad (2.29)$$

On déduit de (1.29) que les noeuds actifs de $B_{i,\,k\,+\,1}(x)$ sont $x_i,\dots,x_{i\,+\,k\,+\,1}$ et que $B_{i,\,k\,+\,1}(x)$ est non nulle seulement sur l'intervalle $[x_i, x_{i+k+1}]$.

On peut montrer que c'est l'unique spline non nulle de support minimum relative aux nœuds x_i, x_{i+k+1}.

On peut aussi montrer que

$$B_{\,i,k+1}(x)\geq 0 \;\; et \;\left|B^l_{i,\,k+1}(x_i)\right| = B^l_{i,k+1}(\,x_{i+k+1})$$

pour $l = 0,\dots,k-1$.

Les B-splines satisfont la relation de récurrence suivante :

$$B_{i,1}(x)=\begin{cases}1 & \text{si } x\in[x_i,x_{i+1}],\\ 0 & \text{sinon,}\end{cases} \qquad (2.30)$$

$$B_{i,k+1}(x)=\frac{x-x_i}{x_{i+k}-x_i}B_{i,k}(x)+\frac{x_{i+k+1}-x}{x_{i+k+1}-x_{i+1}}B_{i+1,k}(x),\quad k\geqslant 1.$$

qui est généralement préférée à (2.29) quand on évalue une *B*-spline en un point donné.

Remarque 1. En généralisant la définition des différences divisées, il est possible de définir des B-splines quand certains nœuds coïncident. On introduit pour cela la relation de récurrence suivante pour les différences divisées de Newton :

$$f[x_0,\dots,x_n]=\begin{cases}\dfrac{f[x_1,\dots,x_n]-f[x_0,\dots,x_{n-1}]}{x_n-x_0} & si\ x_0<x_1<\dots<x_n,\\ \dfrac{f^{(n+1)}(x_0)}{(n+1)!} & si\ x_0=x_1=\dots=x_n.\end{cases}$$

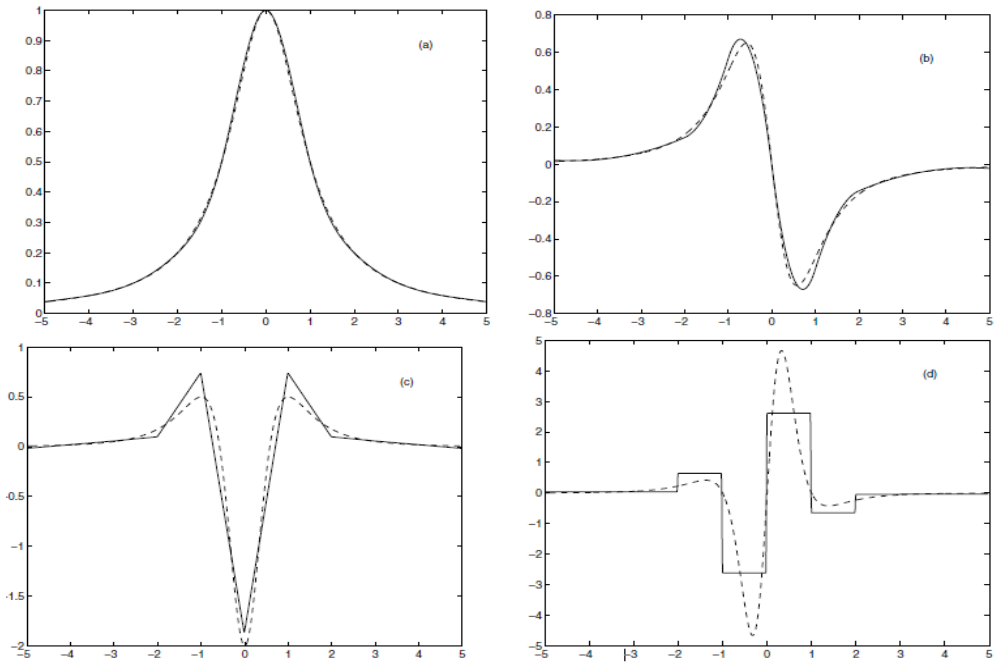


Figure 2.8 – Spline d'interpolation (a) et ses dérivées d'ordre un (b), deux (c) et trois (d) (traits pleins) pour la fonction de l'exemple de Runge (traits discontinus)

Si parmi les $k + 2$ nœuds $x_i,\ldots,x_i + k + 1$, m noeuds $(1 < m < k + 2)$ coïncident et sont égaux à λ, alors 2.19 contient une combinaison linéaire des fonctions$(\lambda - x)_+^{k+1-j}$, pour $j = 1,\ldots,m$. Par conséquent, la B-spline ne peut avoir des dérivées continues en λ que jusqu'à l'ordre $k - m$, et Elle est discontinue si $m = k + 1$. Si $x_{i-1} < x_i = \ldots = x_{i+k} < x_{i+k+1}$, alors

$$B_{i,k+1}(x) = \begin{cases} \left(\dfrac{x_{i+k+1} - x}{x_{i+k+1} - x_i}\right)^k & si\ x \in [x_i, x_{i+k+1}], \\ 0 & sinon, \end{cases}$$

et si $x_i < x_{i+1} = \ldots = x_{i+k+1} < x_{i+k+2}$, *alors*

$$B_{i,k+1}(x) = \begin{cases} \left(\dfrac{x - x_i}{x_{i+k+1} - x_i}\right)^k & si\ x \in [x_i, x_{i+k+1}], \\ 0 & sinon, \end{cases}$$

En combinant ces formules avec la relation de récurrence (2.30), on peut construire des B-splines relatives à des nœuds pouvant coïncider.

Exemple 3. *Examinons le cas particulier de B-splines cubiques sur les nœuds équirépartis* $x_{i+1} = x_i + h, i = 0,\ldots,n - 1$. *L'équation (2.29) devient*

$$6h^3 B_{i,4}(x) =$$
$$\begin{cases} (x-x_i)^3 & si\ x \in [x_i, x_{i+1}], \\ h^3 + 3h^2(x-x_{i+1}) + 3h(x-x_{i+1})^2 - 3(x-x_{i+1})^3 & si\ x \in [x_{i+1}, x_{i+2}], \\ h^3 + 3h^2(x_{i+3}-x) + 3h(x_{i+3}-x)^2 - 3(x_{i+3}-x)^3 & si\ x \in [x_{i+2}, x_{i+3}], \\ (x_{i+4}-x)^3, & si\ x \in [x_{i+3}, x_{i+4}], \\ 0 & sinon. \end{cases}$$

La Figure (2.9) représente le graphe de $B_{i,4}$ pour des noeuds distincts et partiellement confondus.
Etant donné n+1 nœuds distincts $x_j, j = 0, \ldots, n$, on peut construire $n - k$ B-splines de degré k linéairement indépendantes, mais il reste alors

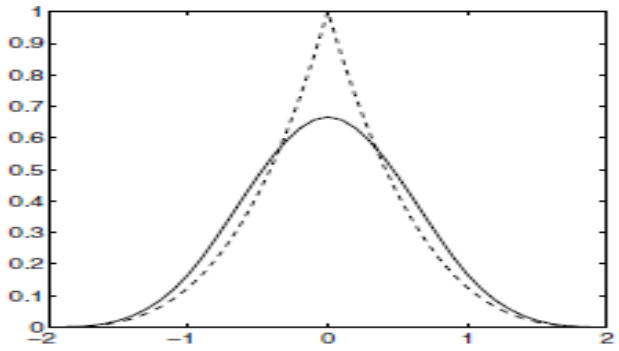


Figure2.9 – B-spline pour des nœuds distincts (en trait plein) et pour des nœuds dont trois coïncident à l'origine. Remarquer la discontinuité de la dérivée première

$2k$ degrés de liberté à fixer pour construire une base de S_k. Une manière de procéder consiste à introduire $2k$ nœuds fictifs

$$x_{-k} \leqslant x_{-k+1} \leqslant \ldots \leqslant x_{-1} \leqslant x_0 = a,$$
$$b = x_n \leqslant x_{n+1} \leqslant \ldots \leqslant x_{n+k}, \qquad (2.31)$$

auxquels on associe la *B*-splines $B_{i,k+1}$ pour $i = -k, \ldots, -1$ et $i = n-k, \ldots, n-1$. Ainsi, toute spline $s_k \in S_k$ s'écrit de manière unique

$$s_k(x) = \sum_{i=-k}^{n-1} c_i B_{i,k+1}(x). \qquad (2.32)$$

Les réels ci sont les coefficients de *B*-spline de s_k. On choisit généralement les nœuds (2.31) confondus ou périodiques.
Confondus : ce choix est bien adapté pour imposer les valeurs atteintes par une spline aux extrémités de son intervalle de définition. Dans ce cas en effet, d'après la Remarque (1), on a

$$s_k(a) = c_{-k}, \qquad s_k(b) = c_{n-1}. \qquad (2.33)$$

Périodiques, c est.-à-dire

$$x_{-i} = x_{n-i} - b + a, \quad x_{i+n} = x_i + b - a, \quad i = 1, \ldots, k.$$

C'est un bon choix si on doit imposer les conditions de périodicité (2.20)[3]

Références :

[1] Franck Jedrzejewski, Introduction aux méthodes numériques Deuxième édition, SPRINGER Springer- Verlag France 2001,2005

[2] Jean-Philippe Grivet, Méthodes numériques appliquées pour le scientifique et l'ingénieur, EDP Sciences, 2009

[3] Alfio Quarteroni, Riccardo Sacco, Fausto Saleri, Méthodes Numériques Algorithmes, analyse et applications, SPRINGER, Springer-Verlag Italia, Milano 2004

[4] Claude BREZINSK, Méthodes numériques de base – Analyse numérique, Techniques d'ingénieures, Edition technique, 2006

[5] Gouri Dhat,Gilbert Touzout, Une présentation de la méthode des éléments finis, Deuxième édition Maloine S.A.Editeur1984

Chapitre 3

L'approximation par éléments finis

3.1 Introduction :

La méthode d'approximation par sous-domaines simplifie la construction de u(x) est s'adapte très bien au calcul par ordinateur.

Elle consiste à :

- Identifier un ensemble de sous-domaines V^e du domaine V :
- Définir une fonction approchée $u_e(x)$ différente sur chaque sous-domaine V^e par la méthode des approximations nodales. Chaque fonction peut dépendre des variables nodales attachées à des nœuds situés sur d'autres sous-domaines comme c'est le cas dans l'approximation de type « spline »

la méthode d'approximation par éléments finis est une méthode d'approximation par sous-domaines qui présente les particularités suivantes [4]:

- l'approximation nodale sur chaque sous-domaine V^e ne fait intervenir que les variables nodales attachées à des nœuds situés sur V^e et sur sa frontière.
- Les fonctions approchées $u_e(x)$ sur chaque sous-domaine sont construites de manière à être continues sur V^e et elles satisfont des conditions de continuité entre les différent sous-domaines.
- Les sous-domaines V^e sont appelés des éléments.
- Les points en lesquels la fonction approchée $u_e(x)$ coïncide avec la fonction exacte $u_{ex}(x)$ sont les nœuds d'interpolation ou points nodaux.
- Les coordonnées xi de ces nœuds sont les coordonnées nodales.
- Les valeurs $u_i=u_e(x_i)=u_{ex}(x_i)$ sont les variables nodales.

Pour appliquer la méthode d'éléments finis il faut donc tout d'abord définir analytiquement la géométrie de tous les éléments, en suite construire les fonctions d'interpolations Ni(x) correspondant à chaque élément.

3.2 Approximation sur un élément de références :

3.2.1 Expression de la fonction approchée U(x) :

Nous choisissons sur le domaine V un ensemble de n nœuds d'interpolation de coordonnées Xi , confondus ou non avec les noeuds géométrique. Sur chaque élément *Ve* nous utilisons une approximation nodale de la fonction exacte $Uex(x)$:

$$U_{ex}(x) \approx U(x) =< N_1(x)N_2(x)\dots N_{n^e}(x) > \left\{ \begin{array}{c} U_1 \\ U_2 \\ \vdots \\ U_{n^e} \end{array} \right\} =< N(x) > \{U_n\} \tag{3.1}$$

ou : x appartient á *V e*.
u_1, u_2, u_{ne} sont les valeurs de *Uex* aux *ne* nœuds d'interpolation de élément ou variables nodales. N(x) sont les fonctions d'interpolation sur l'élément réel.
Remplaçons l'approximation sur l'élément réel par l'approximation correspondante sur l'élément de référence :

$$U_{ex}(\xi) \approx U(\xi) =< N(\xi) > U_n \tag{3.2}$$

Avec :

$$\tau : \xi \rightarrow x(\xi) = [\overline{N}(\xi)]\{x_n\} \tag{3.3}$$

ou : {*Un*} sont les variables nodales de l'élément de référence.
< N(ξ)> sont les fonction d'interpolation sur l'élément de référence

Remarque 2.

- En générale les fonctions N(x) ne sont utilisons que pour des éléments simples. Elles sont le plus souvent remplacées par les fonctions N(ξ) où x et ξ sont liés par la transformation τ définis par (3.3).
- Dans l'expression (3.1), les fonctions N(x) dépendent des coordonnées des nœuds de l'élément et sont donc différentes pour chaque élément par contre, dans l'expression (3.2). les fonctions N(ξ) sont indépendantes de la géométrie de l'élément réel V^e. les mêmes fonctions N(ξ) peuvent donc être utilisées pour tous les éléments possédant le même élément de référence caractérisé par :
    - sa forme.
    - ses nœuds géométriques.
    - ses nœuds d'interpolation.

3.2.2 Propriétés de la fonction approchée U(x)

a) Propriété fondamentale de l'approximation nodale :

Nous retrouvons les propriétés de l'approximation nodale du paragraphe **2.3.1** : la fonction approchée U(x) coïncide avec la fonction exacte $U_{ex}(x)$ en tous les nœuds d'interpolation de l'élément, de coordonnées
xi :

$$U_{ex}(x_i) = U(x_i) = U_i =< N_1(x_i) N_2(x_i) \cdots > \begin{Bmatrix} U_1 \\ U_2 \\ \vdots \\ U_{n^e} \end{Bmatrix}$$

D'où :

$$N_j(x_i) = \begin{cases} 0 \text{ si} & i \neq j \\ 1 \text{ si} & i = j \end{cases} \tag{3.4}$$

De même, en utilisant l'approximation sur l'élément de référence :

$$U_{ex}(\xi_i) = U(\xi_i) = U_i =< N_1(\xi_i) \quad N_2(\xi_i) \cdots > \begin{Bmatrix} U_1 \\ U_2 \\ \vdots \\ U_{n^e} \end{Bmatrix}$$

D'où :

$$N_j(\xi_i) = \begin{cases} 0 \text{ si} & i \neq j \\ 1 \text{ si} & i = j \end{cases} \tag{3.5}$$

b) Continuité sur l'élément :

Si nous désirons obtenir une fonction approchée U(x) continue sur élément. Ainsi que ses dérivées jusqu'á l'ordre s, nous devons utiliser des fonctions *Ni*(*x*) continues et á dérivées continues jusqu'á l'ordre s.

c) Continuité entre élément :

Si nous désirons que U(x) et ses dérivées jusqu'á l'ordre s soient continus sur une frontière commune á deux éléments, il faut que U(x) et ses dérivées jusqu'á l'ordre s dépendent de manière unique des seules variables nodales associées aux nœuds de cette frontière. Considérons d'abord la continuité de U(x) sur une fonction (continuité C^0)

$$U(x) =< N_1(x) \quad N_2(x) \cdots > \begin{Bmatrix} U_1 \\ U_2 \\ \vdots \\ U_{n^e} \end{Bmatrix}$$

Les produits $Ni(x)ui$ doivent être nuls si Ui n'est pas une variable nodale associée á un nœuds de cette frontières.
D'où :

$$N_i(x) = 0 \tag{3.6}$$

Lorsque *X* est situé sur une frontière et Ui n'est pas une variable nodale de cette frontière.
-De même sur l'élément de référence.

$$N_i(\xi) = 0$$

Lorsqu'est situer sur une frontière et U_i n'est pas une variable nodale de cette frontière
la condition pour que $\frac{\partial u(x)}{\partial x}$ soit continue sur une frontière s'écrit de manière similaire :

$$\frac{\partial U(x)}{\partial x} =< \frac{\partial N_1(x)}{\partial x} \quad \frac{\partial N_2(x)}{\partial x} > \cdots \begin{Bmatrix} U_1 \\ U_2 \\ \vdots \\ U_{n^e} \end{Bmatrix}$$

Ou :

$$\frac{\partial U(x)}{\partial x} = 0 \tag{3.7}$$

Lorsque x est situé sur une frontière et *Ui* n'est pas une variable nodale de cette frontière.
La condition précédente s'écrit sur l'élément de référence, á deux dimensions

$$\frac{\partial N_i(\xi)}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial N_i(\xi)}{\partial \eta}\frac{\partial \eta}{\partial x} = 0$$

Lorsque *ξ* est situé sur la frontière et *Ui* n'est pas une variable nodale de cette frontière. La notion de continuité sur les frontières entre les éléments est une notion clé de la méthode des éléments finis. Elle est liée á la notion d'élément conforme ou non conforme

3.3 Construction des fonctions $N(\xi)$ et $\bar{N}(\xi)$

Les fonctions de transformation géométrique $\bar{N}(\xi)$ et les fonctions d'interpolation sur l'élément de référence *N*(*ξ*) ont les mêmes propriétés. Elles peuvent parfois être construites directement á partir de polynômes.
Ceux-ci sont souvent des polynômes classique de type Lagrange ou Hermite ; cependant il n'existe pas de technique manuelle systématique pour les construite. Seule l'expérience a permis de trouver les fonctions *N*(*ξ*) correspondant á un certain nombre d'éléments classiques.
Nous proposerons dans les paragraphes suivants une méthode numérique générale valable pour tous les types d'éléments.

3.3.1 Méthode générale de construction

-**Choix de la base polynomiale** :

Exprimons $U(\xi)$ sur l'élément de référence sous la forme d'une combinaison linéaire de fonctions connues indépendantes $P1(\xi), P2(\xi), \ldots,$ qui sont le plus souvent des monômes indépendants. Le choix des fonctions $Pi(\xi)$ est l'une des opérations de base de la méthode des éléments finis.

$$U(\xi) =< P_1(\xi) \quad P_2(\xi) \cdots > \begin{Bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n^e} \end{Bmatrix} =< P(\xi) > \tag{3.8}$$

L'ensemble des fonctions P(ξ) constitue la base polynomiale de l'approximation. Son nombre de termes doit être égal au nombre de variables nodales ou nombre de degrés de liberté n_d de l'élément. Nous utilisons de plus souvent une base polynomiale complète ; ceci n'est possible que pour certaines valeurs de n_d. Le tableau suivant précise le nombre de monômes nécessaires pour construire des polynômes complets.

Degré du polynôme r	1 dimension	2 dimensions	3 dimensions
	n_d	n_d	n_d
1	2	3	4
2	3	6	10
3	4	10	20
4	5	15	35
5	6	21	56

Nombre de dimensions	Degré du polynôme r	Base polynomiale < P >	n_d
Base complètes			
1 1	1 2	< 1 ζ > (linéaire) < 1 ζ ζ^2 > (quadratique)	2 3
2 2	1 2	< 1 ζ η > (linéaire) < 1 ζ η ζ^2 ζη η^2 > (quadratique)	
3 3	1 2	< 1 ζ η ζ > (linéaire) < 1 ζ η ζ ζ^2 ζη η^2 ηζ ζ^2 ζξ > (quadratique)	4 10
Bases non complètes			
2 3	2 3	< 1 ζ η ηζ > (bi-linéaire) < 1 ζ η ζ ζ^2 ζη ηζ ζξ ξηζ > (tri-linéaire)	4 8

Exemple 3. Bases polynomiales complètes et incomplètes Pour construire les fonctions de transformation géométrique N, choisissons de la même manière des expression de x de la forme :

$$\begin{aligned} x(\xi) &= <\overline{P}(\xi)>\{a_x\} \\ y(\xi) &= <\overline{P}(\xi)>\{a_y\} \\ z(\xi) &= <\overline{P}(\xi)>\{a_z\} \end{aligned} \tag{3.9}$$

Le nombre de fonction $P(\xi)$ et de coefficients $\{ax\}, \{ay\}$, et $\{az\}$ est égal au nombre *ne* de nœuds géométriques de l'élément.

-Relations entre variables généralisées et variables nodales :

Exprimons qu'en chaque nœud d'interpolation de coordonnées $\{\xi_i\}$, la fonction $U(\xi)$ prend la valeur nodale $U_i = U_{ex}(\xi_i)$:

$$\begin{Bmatrix} U_1 \\ U_2 \\ . \\ . \\ . \\ U_{n_d} \end{Bmatrix} = \{U_n\} = \begin{bmatrix} <P_1(\zeta_1) & P_2(\zeta_1) \dots\dots\dots P_{n_d}(\zeta_1)> \\ <P_1(\zeta_2) & P_2(\zeta_2) \dots\dots\dots P_{n_d}(\zeta_2)> \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ <P_1(\zeta_1) & P_2(\zeta_{n_d}) \dots\dots\dots P_{n_d}(\zeta_{n_d})> \end{bmatrix} \{a\}$$

$$\{U_n\} = [P_n]\{a\} \tag{3.10}$$

Soit en inversant la matrice nodale $[P_n]$ d'ordre n_d

$$\{a\} = [P_n]^{-1}\{U_n\} \tag{3.11}$$

-Expression des fonctions N et $\overline{N}$

Reportons (3.11) dans (3.8) :

$$U(\zeta) = < P(\zeta)[P_n]^{-1}\{U_n\} >$$

Soit :

$$U(\zeta) = < N(\zeta) > \{U_n\}$$

D'où :

$$< N(\zeta) > = < P(\zeta) > [P]^{-1} \tag{3.12}$$

Nous obtenons de la même manière dans le cas des fonctions $\overline{N}$

3.4 Exemples :

3.4.1 Approximation à une dimension par éléments finis

Nous cherchons une approximation de la fonction de la figure 3.1 qui passe par les points $u(x_1)$, $u(x_2)$, $u(x_3)$, $u(x_4)$.

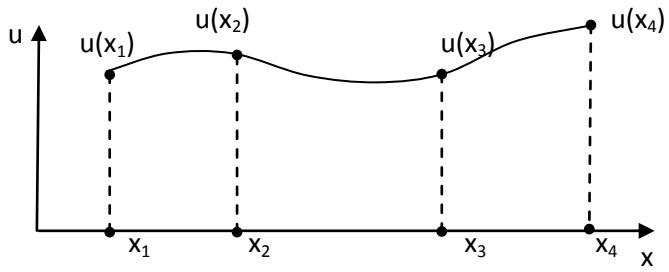


Figure3.1– Fonction continue.

-Définition géométrique des éléments :

Nœuds : 1, 2, 3, 4.
Coordonnées nodales : x_1, x_2, x_3, x_4
Domaine complet : $V : x_1 \leq x \leq x_4$
Eléments : $V^1 : x_1 \leq x \leq x_2$
$V^2 : x_2 \leq x \leq x_3$
$V^3 : x_3 \leq x \leq x_4$

-Construction des fonctions approchées $u^e(x)$:

Variables nodales : u_1, u_2, u_3, u_4
Fonctions approchées $u^e(x)$ linéaires sur chaque élément.

Elément 1 (domaine V^1) :

$$u^1(x) = N_1 u_1 + N_2 u_2 \tag{3.10}$$

Où N_1 et N_2 sont des fonctions linéaires en x

$$\begin{cases} N_1 = \dfrac{x - x_2}{x_1 - x_2} & N_1(x_1) = 1 \qquad N_1(x_2) = 0 \\ N_2 = \dfrac{x - x}{x_2 - x_1} & , N_2(x_1) = 0 \qquad N_2(x_2) = 1 \\ x_1 \leq x \leq x_2 \end{cases} \tag{3.11}$$

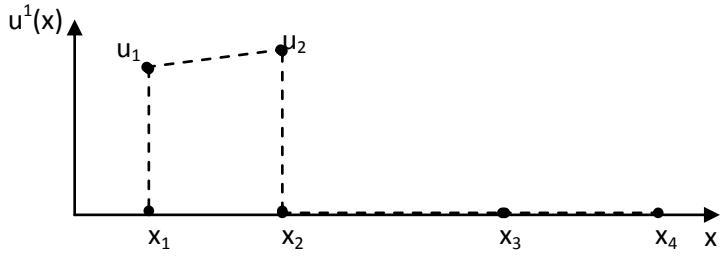


Figure 3.2—Elément 1(domaine V^1).

-Elément 2 (domaine V^2) :

$$u^2(x) = N_1 u_2 + N_2 u_3 \tag{3.12}$$

Où :

$$\begin{cases} N_1 = \dfrac{x - x_3}{x_2 - x_3} \; ; \\ N_2 = \dfrac{x - x_2}{x_3 - x_2} \\ x_2 \leq x \leq x_3 \end{cases} \tag{3.13}$$

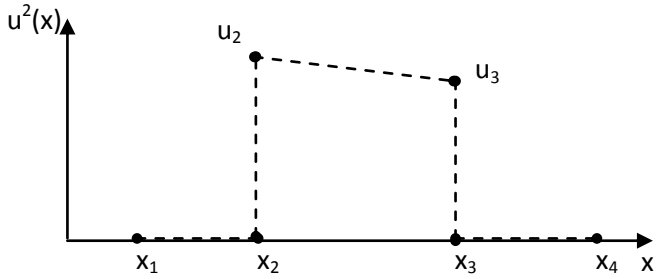


Figure3.3–Elément 2(domaine V^2).

-Eléments 3 (domaine V^3) :

$$u^3(x) = N_1 u_3 + N_2 u_4 \qquad (3.14)$$

Où :

$$\begin{cases} N_1 = \dfrac{x - x_4}{x_3 - x_4} \ ; \\ N_2 = \dfrac{x - x_3}{x_4 - x_3} \\ x_2 \le x \le x_4 \end{cases} \qquad (3.15)$$

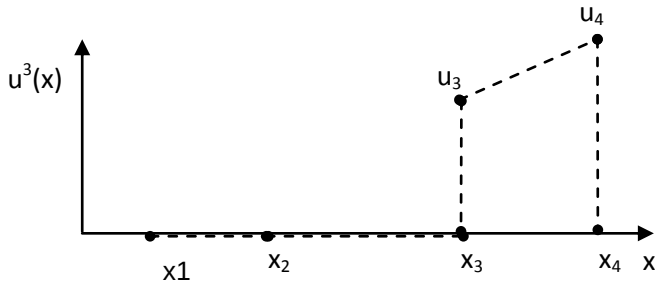


Figure3.4– Elément 3 (domaine V^3).

Les fonctions $u^e(x)$ et $N_i(x)$ sont différentes pour chaque élément V^e ; ces fonctions sont nulles en dehors de l'élément V^e. La somme des fonctions $u^1(x)$, $u^2(x)$ et $u^3(x)$ donne la fonction approchée u(x) sur l'ensemble du domaine V (figure IV.8).

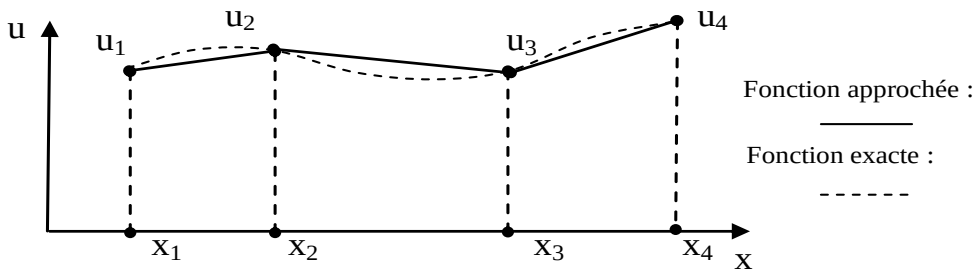


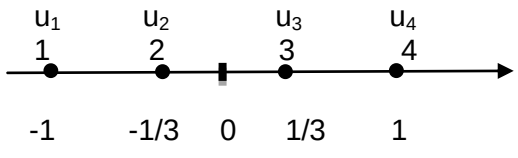
Figure 3.5–Fonction approchée obtenue après assemblage des éléments

3.5 Elément à une dimension

3.5.1 Elément de haute précision de type Lagrange :

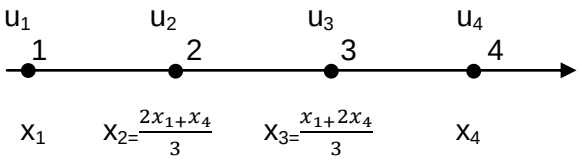
Cette famille d'élément est obtenue en augmentant le nombre de nœuds d'interpolation et en gardant une variable *Ui* par nœud.

3.5.1.1 Elément cubique á noeuds équidistants : (4 nœuds, C0) :



$-1 \leq \xi \leq 1$ n=4

V^r



n_d=4 $x_1 \leq x \leq x_4$

V^e

$$< P >=< 1 \quad \xi \quad \xi^2 \quad \xi^3 > \tag{3.16}$$

$$[p_n]^{-1} = \frac{1}{16}\begin{bmatrix} -1 & 9 & 9 & -1 \\ 1 & -27 & 27 & -1 \\ 9 & -9 & -9 & 9 \\ -9 & 27 & -27 & 9 \end{bmatrix} \tag{3.17}$$

	$\frac{1}{c}\{N\}$	$\frac{1}{c}\{\partial N/\partial\zeta\}$	C
1 2 3 4	-(1 - ζ) (1 - 9 ζ^2) 9(1 -ζ^2) (1 - 3ζ) 9(1 -ζ^2) (1 + 3 ζ^2) -(1 – ζ) (1 - 9 ζ^2)	1 + 18 ζ -27ζ^2 -27 - 18 ζ + 81 ζ^2 27 - 18 ζ + 81 ζ^2 -1 + 18 ζ + 27 ζ^2	1/16

$$e(\xi) \leq \frac{1}{9}(-1 + 10\xi^2 - 9\xi^4)\frac{1}{24}\max\left|\frac{\partial^4 U_e x}{\partial \xi^4}\right|_{v^r}$$

$$|e|_0 \leq \frac{I^4}{1944} \max \left| \frac{\partial^4 U_e x}{\partial X^4} \right|_{v^e} \tag{3.18}$$

$$|e|_I \leq \frac{I^3}{108} \max \left| \frac{\partial^4 U_e x}{\partial \xi^4} \right|_{v^e} \tag{3.19}$$

3.5.1.2 Elément général à n nœuds (n nœuds C0) :

les fonction N d'un élément á n nœuds d'interpolation sont des polynômes de Lagrange de degré n-1.Elles s'écrivent :

$$N_i(\xi) \prod_{j=1, j\neq i}^{n} \frac{(\xi_j - \xi)}{(\xi_j - \xi_i)} \tag{3.20}$$

Dans le cas où les nœuds sont régulièrement espacés :

$$\xi_i = -1 + 2\frac{i-1}{n-1}$$
$$N_i(\xi) = \prod_{j=1, j\neq i}^{n} \frac{(2j - n - 1) - (n-1)\xi}{2(j-1)} \tag{3.21}$$

Ces fonctions peuvent être obtenues par la méthode générale en utilisant :

$$< P >=< 1 \quad \xi \quad \xi^2 \ldots \xi^{n-1} > \tag{3.22}$$

$$< \xi_n >=< -1; -1 + \Delta; -1 + 2\Delta; \cdots - 1 + (n-1)\Delta >$$

Ou : $\Delta = \frac{2}{n-1}$

Les erreurs sont de la forme :

$$|e|_0 = C_0/^n \max \left| \frac{\partial^n U_e x}{\partial X^n} \right|_V^e \tag{3.23}$$

$$|e|_0 = C_1/^{n-1} \max \left| \frac{\partial^n U_e x}{\partial X^n} \right|_V^e \tag{3.24}$$

Tous les éléments présentés jusqu'ici offrent une continuité de type C^0 si la base $< P >$ est de degré n-1, la fonction u et ses dérivées jusqu'á l'ordre n-1 sont continues sur l'élément, et seule u est continu sur la frontière de l'élément.

3.5.2 Elément de haute pression de type de Hermite

Ces élément sont obtenus en augmentant le nombre de variables nodales attachées á caque nœud : aux variables Ui nous ajoutons les valeurs aux noeuds des dérivées de Uex :

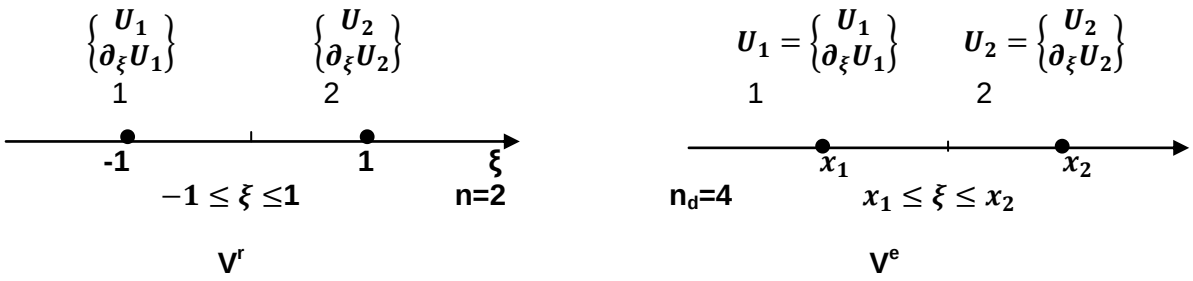
$$\left.\frac{\partial U_{ex}}{\partial x}\right|_{\mathbf{n\text{œ}ud}_i} \qquad \left.\frac{\partial^2 U_{ex}}{\partial x^2}\right|_{\mathbf{n\text{œ}ud}_i}$$

Nous noterons :

$$\begin{aligned} \partial_x U_i &= \left.\frac{\partial U_{ex}}{\partial x}\right|_{x=x_i} & \partial_x^2 U_i &= \left.\frac{\partial^2 U_{ex}}{\partial x^2}\right|_{x=x_i} \\ \partial_\xi U_i &= \left.\frac{\partial U_{ex}}{\partial \xi}\right|_{\xi=\xi_i} & \partial_\xi^2 U_i &= \left.\frac{\partial^2 U_{ex}}{\partial \xi^2}\right|_{\xi=\xi_i} \end{aligned}$$

Les nœuds géométriques, les fonctions *N* et la matrice jacobéennes [J] restent identiques á ceux de l'élément linéaire.

3.5.2.1 Elément cubique (2 nœudsC^1) :



Nombre de variables par nœuds : 2

$$< P >=< 1 \quad \xi \quad \xi^2 \quad \xi^3 >$$

$$[P_n] = \begin{bmatrix} < P(\xi_1) > \\ < \frac{\partial P}{\partial \xi} \xi_1 > \\ \cdots\cdots \\ < P(\xi_2) > \\ < \frac{\partial P}{\partial \xi} \xi_2 > \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 0 & 1 & -2 & 3 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} \qquad (3.25)$$

$$[P_n]^{-1} = \frac{1}{4}\begin{bmatrix} 2 & 1 & 2 & -1 \\ -3 & -1 & 3 & -1 \\ 0 & -1 & 0 & 1 \\ 1 & 1 & -1 & 1 \end{bmatrix}$$

$$U(\xi) =< N > \{U_n\}_\xi \quad \text{ou} \quad U(\xi) =< N > \{U_n\} \tag{3.26}$$

où chaque fonction Ni ne diffère, dans les deux cas, que par un facteur multiplicatif.

	$\frac{1}{c}\{N\}$	$\frac{1}{c}\ \partial N/\partial\zeta$	C pour $\{u_n\}_\xi$	C
1	$(1-\zeta)^2(2+\zeta)$	$-3(1-\zeta^2)$		1/4
2	$(1-\zeta^2)(1-\zeta)$	$(-1+\zeta)(1+3\zeta)$	1/4	1/8
3	$(1-\zeta)^2(2-\zeta)$	$3(1-\zeta^2)$		1/4
4	$(-1+\zeta^2)(1+\zeta)$	$(-1-\zeta)(1-3\zeta)$		1/8

Remarquons que ces fonctions, qui sont des polynômes d'Hermite.
Cependant il est possible de construire des relations du même type. Nous avons par Exemple :

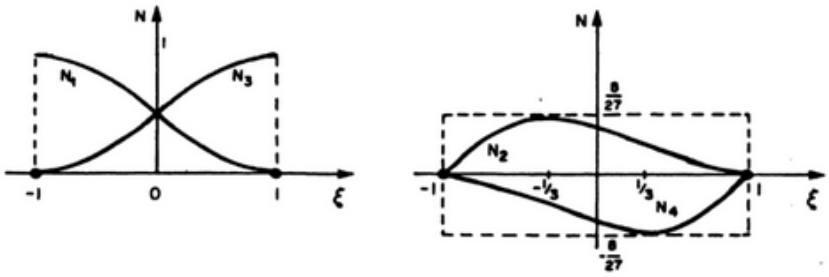
$$N_1 + N_3 = 1 \quad \text{et} \quad -N_1 + N_2 + N_3 + N_4 = \xi$$

Les variables nodales sur l'élémentes de référence et sur l'élément réel sont différent á cause des dérivations en ξ et x :

$$\{U_n\}_\xi = \begin{Bmatrix} U_1 \\ \partial_\xi U_1 \\ U_2 \\ \partial_\xi U_2 \end{Bmatrix}; \{U_n\} = \begin{Bmatrix} U_1 \\ U_2 \end{Bmatrix} = \begin{Bmatrix} U_1 \\ \partial_x U_1 \\ U_2 \\ \partial_x U_2 \end{Bmatrix};$$

$$\{U_n\}_n = \frac{1}{4}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{l}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{l}{2} \end{bmatrix}\{U_n\}; \quad l = x_2 - x_1 \tag{3.27}$$

En fait ce sont les variables nodales {*Un*} contenant les dérivées en x qui sont conservées comme variables finales du problème.
Les graphes des fonctions N sont les suivant :



Remarquons que l'ordre de l'erreur pour cet élément est le même que celui de l'élément de Lagrange á 4 nœuds. Cependant le coefficient $\frac{1}{384}$de $|e|0$ est plus grand que dans l'élément de Lagrange $\left(\frac{1}{1944}\right)$alors que le coefficient $\frac{1}{72\sqrt{3}}$ de $|e|_1$ est plus petit que pour l'élément de Lagrange$\left(\frac{1}{108}\right)$.
L'élément présenté est un élément à continuité C^1: u et $\frac{\partial u}{\partial x}$sont continues sur l'élément et à la frontière de l'élément.

3.6 Assemblage des éléments :

Le but de l'opération d'assemblage est de relier les différents éléments pour former un seul objet continu, mathématiquement ceci revient à assembler les systèmes d'équations élémentaires relatifs à touts les éléments pour construire un seul système contenant toutes les données est les inconnues du problème. Pour le cas de surfaces avec des éléments triangulaires, par exemple, il revient à assembler les surfaces élémentaires pour former une seule surface continue (figure3.6)

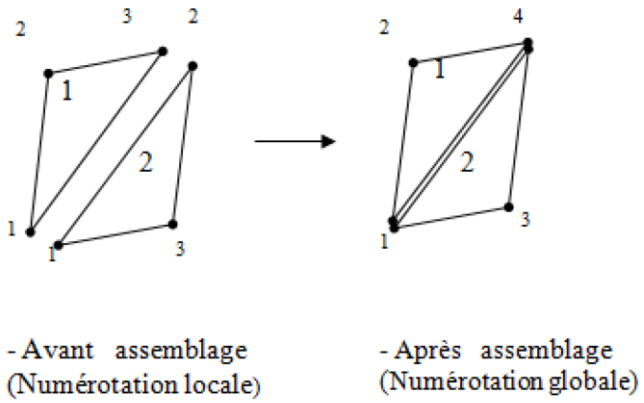


Figure3.6—Assemblage de deux éléments triangulaires

- le système élémentaire pour l'élément 1 est :

$$\begin{bmatrix} a_{11}^1 & a_{12}^1 & a_{13}^1 \\ a_{21}^1 & a_{22}^1 & a_{23}^1 \\ a_{31}^1 & a_{32}^1 & a_{33}^1 \end{bmatrix} \begin{bmatrix} z_1^1 \\ z_2^1 \\ z_3^1 \end{bmatrix} = \begin{bmatrix} b_1^1 \\ b_2^1 \\ b_3^1 \end{bmatrix} \quad (3.28)$$

- le système élémentaire pour l'élément 2 est :

$$\begin{bmatrix} a_{11}^2 & a_{12}^2 & a_{13}^2 \\ a_{21}^2 & a_{22}^2 & a_{23}^2 \\ a_{31}^2 & a_{32}^2 & a_{33}^2 \end{bmatrix} \begin{bmatrix} z_1^2 \\ z_2^2 \\ z_3^2 \end{bmatrix} = \begin{bmatrix} b_1^2 \\ b_2^2 \\ b_3^2 \end{bmatrix} \quad (3.29)$$

- le système global pour les deux éléments assemblés est :

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \quad (3.30)$$

D'après la figure3.6 et la condition présentée au paragraphe précédent, après assemblage on peut écrire :

$$\begin{aligned} z_1 &= z_1^1 = z_1^2 \\ z_2 &= z_2^1 \\ z_3 &= z_3^1 \\ z_4 &= z_3^1 = z_2^2 \end{aligned}$$

Prenons maintenant la première équation des deux systèmes élémentaires, et remplaçons les et par les coordonnées z globales à partir des égalités ci-dessus.

$$\begin{aligned} a_{11}^1 z_1 + a_{12}^1 z_2 + a_{13}^1 z_4 &= b_1^1 \\ a_{11}^2 z_1 + a_{12}^2 z_4 + a_{13}^2 z_3 &= b_1^2 \end{aligned}$$

En faisant la somme des deux cotés des deux équations deux à deux on trouve :

$$(a_{11}^1 + a_{11}^2) z_1 + a_{12}^1 z_2 + a_{13}^2 z_3 + (a_{13}^1 + a_{13}^2) z_4 = (b_1^1 + b_1^2)$$

Par analogie avec la première équation du système d'équation globale, on peut déduire que :

$$\begin{aligned} a_{11} &= (a_{11}^1 + a_{11}^2) \\ a_{12} &= a_{12}^1 \\ a_{13} &= a_{13}^2 \\ a_{14} &= (a_{13}^1 + a_{13}^2) \\ b_1 &= (b_1^1 + b_1^2) \end{aligned}$$

De la même façon on calcule les autres constantes pour aboutir au système d'équation globale des deux éléments assemblés :

$$\begin{bmatrix} (a_{11}^1 + a_{11}^2) & a_{12}^1 & a_{13}^2 & (a_{13}^1 + a_{13}^2) \\ a_{21}^1 & a_{22}^1 & a_{23}^1 & 0 \\ a_{21}^2 & a_{22}^2 & a_{23}^2 & 0 \\ (a_{31}^1 + a_{31}^2) & 0 & 0 & (a_{33}^1 + a_{33}^2) \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} (b_1^1 + b_1^2) \\ b_2^1 \\ b_3^2 \\ (b_4^1 + b_4^2) \end{bmatrix} \quad (3.31)$$

3.7 Approximation par élément finis au sens des moindres carrées :

Dans cette partie une combinaison entre l'approximation au sens des moindres carrés d'un nuage de points avec un polynôme du troisième degré est la méthode d'approximation par éléments finis.
En appliquant l'idée de base de la méthode des éléments finis, au lieu d'essayer de réaliser une approximation d'un nuage de points avec un seul polynôme sur tout le domaine, on va le diviser en sous-domaine et traiter le problème dans chaque sous-domaine. Ceci va nous donner un système d'équation élémentaire. L'assemblage de ce dernier avec les autres systèmes issu des autres domaines pour aboutir à un système globale dont la solution nous fourni la solution qui couvre tout le domaine du problème.
Si on cherche la solution de notre problème c-à-dire la définition de la trajectoire d'outils *z(x)* sous forme d'un polynôme du troisième degré, nous pouvons écrire :

$$z(x) = p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

On peut exprimer aussi Z en utilisant un élément d'Hermite cubique sous cette forme

$$z(x) = N_1(x) z_1 + N_2(x) z_2 + N_3(x) z_1' + N_4(x) z_2'$$

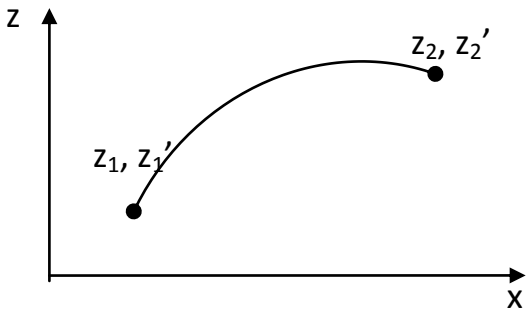
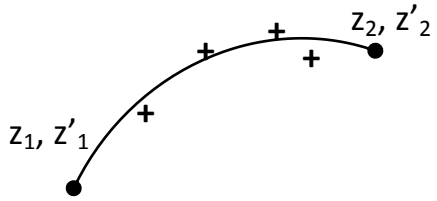


Figure 3.7 : élément d'Hermite cubique

Pour une approximation au sens des moindres carrées :

$$Min\left(\sum e_i^2\right)$$



$$\frac{\partial \sum_{i=1}^{n} e_i^2}{\partial z_n} = 0 \qquad e_i = z - z_i$$

En remplaçant z par sa valeur on obtient

$$e_i = N_1(x_i)z_1 + N_2(x_i)z_2 + N_3(x_i)z_1^{'} + N_4(x_i)z_2^{'} - z_i$$

Le calcul des dérivées nous donne :

$$\begin{cases} \frac{\partial \sum_{i=1}^{n} e_i^2}{\partial z_1} = \sum_{i=1}^{n} (N_1 z_1 + N_2 z_2 + N_3 z_1' + N_3 z_2' - z_i) N_1 = 0 \\ \frac{\partial \sum_{i=1}^{n} e_i^2}{\partial z_2} = \sum_{i=1}^{n} (N_1 z_1 + N_2 z_2 + N_3 z_1' + N_3 z_2' - z_i) N_2 = 0 \\ \frac{\partial \sum_{i=1}^{n} e_i^2}{\partial z_3} = \sum_{i=1}^{n} (N_1 z_1 + N_2 z_2 + N_3 z_1' + N_3 z_2' - z_i) N_3 = 0 \\ \frac{\partial \sum_{i=1}^{n} e_i^2}{\partial z_4} = \sum_{i=1}^{n} (N_1 z_1 + N_2 z_2 + N_3 z_1' + N_3 z_2' - z_i) N_4 = 0 \end{cases}$$

Sous forme matricielle :

$$\begin{bmatrix} \sum N_1 N_1 & \sum N_1 N_2 & \sum N_1 N_3 & \sum N_1 N_4 \\ \sum N_2 N_1 & \sum N_2 N_2 & \sum N_2 N_3 & \sum N_2 N_4 \\ \sum N_3 N_1 & \sum N_3 N_2 & \sum N_3 N_3 & \sum N_3 N_4 \\ \sum N_4 N_1 & \sum N_4 N_2 & \sum N_4 N_3 & \sum N_4 N_4 \end{bmatrix} \begin{bmatrix} Z_1 \\ Z_2 \\ Z_1' \\ Z_2' \end{bmatrix} = \begin{bmatrix} \sum N_1 Z_i \\ \sum N_2 Z_i \\ \sum N_3 Z_i \\ \sum N_4 Z_i \end{bmatrix}$$

-Détermination des fonctions N(x) :

$$z = p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

$$z = \begin{bmatrix} 1 & x & x^2 & x^3 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{Bmatrix} \qquad \text{Ou} \qquad z = [p]\{a\}$$

$$z(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

$$z'(x) = a_1 + a_2 x^2 + a_3 x^3$$

$$\begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 0 & 1 & 2x_1 & 3x_1^2 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 0 & 1 & 2x_2 & 3x_2^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} z_1 \\ z_1' \\ z_2 \\ z_2' \end{bmatrix}$$

$$[p_n]\{a\} = \{z_n\}$$

$$z = [p][p_n]^{-1}\{z_n\}$$

$$z = \begin{bmatrix} \mathrm{N}_1(x) & \mathrm{N}_2(x) & \mathrm{N}_3(x) & \mathrm{N}_4(x) \end{bmatrix} \begin{Bmatrix} z_1 \\ z_1' \\ z_2 \\ z_2' \end{Bmatrix}$$

Où

$$z = [N]\{z_n\}$$

$$[N] = [p][p_n]^{-1}$$

Avec :

$$[P] = \begin{bmatrix} 1 & x & x^2 & x^3 \end{bmatrix}$$

$$[p_n] = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 0 & 1 & 2x_1 & 3x_1^2 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 0 & 1 & 2x_2 & 3x_2^2 \end{bmatrix}$$

Référence:

[1] Gouri Dhat, Gilbert Touzout, Une présentation de la méthode des éléments finis, édition Maloine, 1984

Chapitre 4

Détail de résolution et cas d'application

4.1 Introduction :

La méthode adoptée pour résoudre notre problème est la méthode d'approximation au sens des moindres carrée telle quelle est décrite au chapitre précédent (section 3.7). Le présent chapitre décrit les étapes de résolution de la méthode de lissage avec un cas pratique d'application après avoir présenter les détails de conception du programme de résolution. A la fin, la discussion des résultats obtenus est effectuée.

4.1.1 Etapes de résolution :

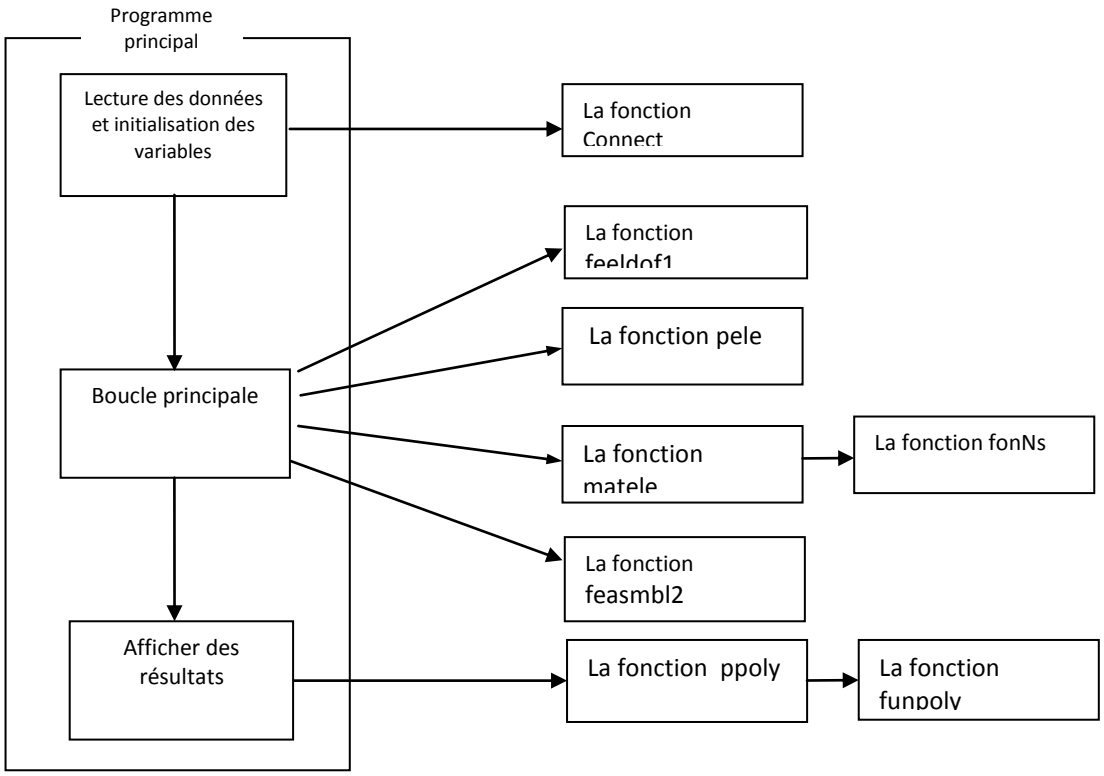
Au début du programme les données du problème sont introduites à savoir le nuage des points issu de la trajectoire G1 ainsi que les paramètres de résolution tel que le nombre de nœuds par éléments et le nombre de degrés de libertés par éléments. La génération du maillage durant laquelle nous déterminons les coordonnées des nœuds et la connectivité des éléments, à savoir la numérotation globale pour les nœuds de chaque élément.
En suite le calcul à l'échelle élémentaire est effectué pour déterminer les matrices et vecteurs élémentaires en utilisant les fonctions de forme choisies au préalable. Ainsi obtenu ces derniers vont être assemblées pour obtenir les matrices et vecteurs globaux qui nous donnent le système d'équation globale. La résolution de ce système va être utilisée pour déduire les paramètres des polynômes qui composent la courbe d'approximation qui est la solution de notre problème.
Les principales étapes de résolution de la méthode peuvent êtres résumés comme suit:
1-lecture des données.
2- initialisation des matrices et des vecteurs.
3- génération du maillage.
4- calcul des matrices et des vecteurs élémentaires et leur assemblage.
5- résolution du système d'équations matricielles globales
6-affichage des résultats

4.1.2 Structure du programme :

En plus des étapes résolution, pour déterminer la structure du programme qu'il faut adopter, il faut prendre en considération d'autres points :

1- prévoir une structure qui facilite le test du programme et la détection des sources d'erreurs.
2- Utiliser des fonctions qui peuvent être utilisées pour d'autres programmes.
3- Réaliser une structure ouverte permettant une intégration facile de nouvelles fonctions pour un développement éventuel de l'algorithme.

Nous pouvons résumer la structure du programme par le schéma suivant :



Comme nous pouvons le constater la structure du programme est assez simple, elle est constituée du programme principal qui contient une boucle qui parcour tous les éléments.
Dans cette boucle on trouve aussi des appels à quatre fonctions qui assurent le calcul et l'assemblage des matrices élémentaires, et une dernière partie qui s'occupe de l'affichage numérique et graphique des résultats.

4.1.3 Langage de programmation utilisé

Pour la réalisation du programme nous avons utilisé MATLAB qui est un environnement de programmation et de calcul très puissant comprenant des outils de calcul pour des domaines très variés allant du calcul algébrique aux réseaux de neurones. Il dispose aussi d'un interpréteur permettant le développement de programme et d'outils personnels. Nous pouvons justifier ce choix par les deux points suivants :

- Le but essentiel de ce travail n'est pas la réalisation d'un logiciel mais de tester un algorithme de calcul ce qui rend l'utilisation d'un interpréteur suffisante au lieu d'un compilateur qui fourni un exécutable.

- Le nombre considérable de fonctions disponibles sous MATLAB auxquelles on peut faire appel directement ce qui permet d'économiser beaucoup d'efforts et de temps.

L'inconvénient majeur de ce choix réside dans la lenteur d'exécution avec MATLAB qui ne nous donne pas une idée précise sur la vraie vitesse d'exécution du programme.

4.2 Détail de l'implémentation :

Dans ce qui suit nous présenterons le détail du programme principal ainsi que celui des différentes fonctions :

4.2.1 Le programme principal :

k : matrice élémentaire
f : vecteur élémentaire
kk : système matricielle
ff : systéme vectorielle
index : un systéme contenant du vecteur dofs associé à chaque élément

```
clc; clear all;
% lecture des données depuis un fichier
fid = fopen('trajectoire2.dat','r');
i=0;
while ~feof(fid)
  i=i+1;
  A(i,:) = fscanf(fid,'%*s %*s %f %*s %f %*s %f %*s %f %*c %f %*s %f',[1, 6]);
end
xx=A(:,2);
zz=A(:,3);

%initialisation des  matrices et  vectors
%
ff=zeros(sdof,1);                %initialisation de systém vectorielle
kk=zeros(sdof,sdof);             %initialisation de systém matricielle
index=zeros(nnel*ndof,1);        % initialisation de vecteur d'index
%

nel =3;                          % nombre des éléments
nnel =2;                         %nombre des noeuds par elements
ndof =2;                         %nombre de  dofs par nœud

% Generer la coordonnée nodale estime et la connectivité nodale pour chaque élément
[nnode,gcoord,nodes]=connect(nel,xx(1),xx(end));% determination de la connectivité
sdof = nnode*ndof;                        % Système total dofs
%
% Calcul de matrices et vecteurs élémentaire et leur assemblage
for iel=1:nel                    %
%
nl=nodes(iel,1); nr=nodes(iel,2);       %
x1=gcoord(nl);x2=gcoord(nr);                %
index=feeldof1(iel,nnel,ndof);           %
[xi,zi]=pele(xx,zz,x1,x2);
[k,f,u]=matele(x1,x2,xi,zi);

[kk,ff]=feasmbl2(kk,ff,k,f,index);%assemblage des matrices et des vecteurs élémentaires
```

```
%
end                                % %
% solution d'équation matricielles
%
Z=kk\ff;

% affichage des résultats

[xp,zp,xn,zn,ai,Ez]=ppoly(Z,zz,xx,nel,nnel,ndof,gcoord,nodes);

plot(xx',zz,'+b',xp,zp,'-g',xn,zn,'.r');
daspect([1 1 1])

fprintf('Les coordonnées des noeuds:\n')
fprintf('    x          z\n')
fprintf(1, '  %6.3f %12.3f\n', [xn' zn'])
fprintf('\n')
fprintf('Les paramètres ai des  polynomes:\n')
fprintf('    a0          a1          a2          a3\n')
fprintf(1, '  %6.3f %12.3f %12.3f %12.3f\n',ai)
fprintf('\n')
% fprintf('Erreur d approximtion pour chaque point:\n')
% fprintf(1, '  %6.3f\n',Ez)
% fprintf('\n')
fprintf('Erreur globale au moindres carrées :%6.3f\n',sum(Ez.^2))
```

4.2.2 Les fonctions utilisées dans le programme :

1. La fonction connect :

Cette fonction génère le maillage autrement dit calcul le nombre des nœuds et la connectivité des éléments à savoir les nœuds relatives à chaque élément et les coordonnées de chaque nœud :

Nel : nombre des éléments
X1 : le premier nœud
X2 : le dernier nœud

```
function[nnode,gcoord,nodes]=connect(nel,x1,x2)
pas=(x2-x1)/nel;
nnode=nel+1;
for i=1:nnode;
   gcoord(i)=x1+((i-1)*pas);
end
for i=1:nel
  nodes(i,1)=i;
  nodes(i,2)=i+1;
end
end
```

2. la fonction *feeldof1*:

Cette fonction nous donne en sortie le tableau *index* qui contient la position des variables de la matrice élémentaire dans la matrice globale pour chaque élément, et ce en utilisant les données suivantes :

iel : élément désigné

nnel : nombre des nœuds par élément

ndof : le degré de liberté associé à chaque nœud (le nombre des variables)

```
function [index]=feeldof1(iel,nnel,ndof)
%
edof = nnel*ndof;
start = (iel-1)*(nnel-1)*ndof;
%
for i=1:edof
index(i)=start+i;
end
end
```

3. la fonction *pele* :

Cette fonction extrait le nuage de points associé à chaque élément du nuage de point global en utilisant les données suivantes :

xx ,zz : les coordonnées des points du nuage global

x1 : la coordonnée x du premier nœud de l'élément concerné

x2 : la coordonnée x du deuxième nœud de l'élément concerné

```
function[xi,zi]=pele(xx,zz,x1,x2)
if x1<x2
 px=xx(find(xx>x1));
 xi=px(find(px<x2));
 pz=zz(find(xx>x1));
 zi=pz(find(px<x2));
else
 px=xx(find(xx>x2));
 xi=px(find(px<x1));
 pz=zz(find(xx>x2));
 zi=pz(find(px<x1));
end
end
```

4. la fonction *matele* :

Cette fonction calcule les matrices et les vecteurs élémentaires ainsi que la solution du système matriciel élémentaire, elle utilise les données suivantes :

X1, X2 : les coordonnées x des 2 nœuds de l'élément concerné
Xi, Zi : les coordonnées des points de nuage associés à l'élément concerné

```
function [k,f,u]=matele(x1,x2,xi,zi)
k=zeros(4,4);
f=zeros(4,1);
nx=length(xi);
for i=1:nx;
   N=fonNs(x1,x2,xi(i));
   k(1,1)=k(1,1)+ N(1)* N(1) ;
   k(1,2)=k(1,2)+ N(1)* N(2) ;
   k(1,3)=k(1,3)+ N(1)* N(3) ;
   k(1,4)=k(1,4)+ N(1)* N(4) ;
   k(2,1)=k(2,1)+ N(2)* N(1) ;
   k(2,2)=k(2,2)+ N(2)* N(2)   ;
   k(2,3)=k(2,3)+ N(2)* N(3)  ;
   k(2,4)=k(2,4)+ N(2)* N(4)  ;

   k(3,1)=k(3,1)+ N(3)* N(1);
   k(3,2)=k(3,2)+ N(3)* N(2);
   k(3,3)=k(3,3)+ N(3)* N(3);
   k(3,4)=k(3,4)+ N(3)* N(4);

   k(4,1)=k(4,1)+ N(4)* N(1) ;
   k(4,2)=k(4,2)+ N(4)* N(2);
   k(4,3)=k(4,3)+  N(4)* N(3);
   k(4,4)=k(4,4)+N(4)* N(4);

   f(1)=f(1)+ N(1)* zi(i);
   f(2)=f(2)+ N(2)* zi(i);
   f(3)=f(3)+ N(3)* zi(i);
   f(4)=f(4)+ N(4)* zi(i);
end
u=k\f;
end
```

5. la fonction fonNs :

Cette fonction calcule la fonction les fonctions de forme N pour chaque élément en utilisant pour chaque élément et pour chaque point (désigné par sa coordonnée x).

```
function [N]=fonNs(x1,x2,x)
        pn=[1 x1 x1^2 x1^3 ; 0 1 2*x1 3*x1^2 ; 1 x2 x2^2 x2^3 ; 0 1 2*x2 3*x2^2];
        p=[1 x x^2 x^3];
        N=p*(inv(pn));
end
```

6. la fonction *feasmble* :

Cette fonction effectue l'assemblage des matrices élémentaires en se basant sur les données ci-dessous :

kk : la matrice global

ff : le vecteur global

k : la matrice élémentaire

f : le vecteur élémentaire

Index : tableau fourni par la fonction *feeldof1*

```
function [kk,ff]=feasmbl2(kk,ff,k,f,index)

edof=length(index);
for i=1:edof
ii=index(i);
ff(ii)=f(i)+ff(ii);
for j=1:edof
jj=index(j);
kk(ii,jj)=kk(ii,jj)+k(j,i);
end
end
end
```

7. la fonction *ppoly* :

Utilisé pour l'affichage graphique des résultats, cette fonction nous donne les coordonnées xp et zp d'un ensemble de points tirés de la courbe d'approximation finale avec les coordonnées finales xn et zn des nœuds des différents éléments, et ce en utilisant le vecteur zz solution du système d'équation globale.

```
function [xp,zp,xn,zn,ai,Ez]=ppoly(Z,zz,xx,nel,nnel,ndof,gcoord,nodes)
for i=1:nel
   index=feeldof1(i,nnel,ndof);
   z=Z(index);
   x1=gcoord(nodes(i,1));
   x2=gcoord(nodes(i,2));
   [x,z,z1,z2,a]=funpoly(x1,x2,z);
   [xi,zi]=pele(xx,zz,x1,x2);
   E=zi-(a(1)+a(2)*xi+a(3)*xi.^2+a(4)*xi.^3);
    if i==1
      xp=x;
      zp=z;
      xn(1)=x1;
      zn(1)=z1;
      xn(2)=x2;
      zn(2)=z2;
      ai=a';
      Ez=E;
    else
      xp=[xp x];
      zp=[zp z];
      xn=[xn x2];
      zn=[zn z2];
      ai=cat(1,ai,a');
      Ez=[Ez;E];
    end
end
end
```

8. la fonction *funpoly* :

Utilisée par la fonction *ppoly*, elle permet de déduire les coordonnées d'un ensemble de points appartenant à la courbe polynomiale les coordonnées z1 et z2 des points de départ et d'arrivé ainsi que les paramètres ai qui définis le polynôme. Cette fonction est appelée pour chaque élément.

```
Function [xp,zp,z1,z2,ai]=funpolz(x1,x2,u)
pn=[1 x1 x1^2 x1^3 ; 0 1 2*x1 3*x1^2 ; 1 x2 x2^2 x2^3 ; 0 1 2*x2 3*x2^2];
ai=pn\u;
for i=1:11
   pas=(x2-x1)/10;
   xp(i)=x1+pas*(i-1);
   zp(i)=ai(1)+ai(2)*xp(i)+ai(3)*xp(i)^2+ai(4)*xp(i)^3;
end

z1=ai(1)+ai(2)*x1+ai(3)*x1^2+ai(4)*x1^3;
z2=ai(1)+ai(2)*x2+ai(3)*x2^2+ai(4)*x2^3;
end
```

4.3 Cas d'application :

Pour essayer d'évaluer les performances du programme réalisé, nous avons considéré le cas de fabrication du plateau tibiale d'une prothèse du genou. C'est un cas qui correspond bien à notre problématique car c'est cas où les exigences en termes de qualité de surface sont élevées. Pour que cette surface soit lisse pour, éviter un frottement élevé qui produira des débris dans le genou du malade.

Comme expliqué précédemment la première étape de résolution est celle de lecture de données depuis le programme d'usinage en G1 (GOTO) présenté dans la figure. La visualisation de la trajectoire nous donne la figure … .

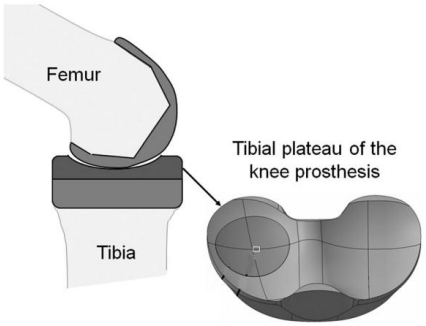


Figure 4.1 — Plateau tibial d'une prothèse médicale du genou

```
$$ ------------------------------------------------------------
-----
$$     Généré le lundi 26 novembre 2012 15:20:22
$$     CATIA APT VERSION 1.0
$$ ------------------------------------------------------------
-----
$$ Programme de fabrication.1
$$  Phase d'usinage.1
$$*CATIA0
$$ Programme de fabrication.1
$$     0.00000    -1.00000     0.00000    -7.50000
$$     0.00000     0.00000     1.00000    -0.70000
$$    -1.00000     0.00000     0.00000     0.00000
PARTNO Phase d'usinage.1
$$ OPERATION NAME : Changement outil.2
$$  Début de génération de: Changement outil.2
MULTAX
$$ TOOLCHANGEBEGINNING
CUTTER/ 10.000000,  5.000000,  0.000000,  5.000000,  0.000000,$
         0.000000, 50.000000
TOOLNO/1,MILL,1,0,   10.000000,  100.000000,$
 100.000000,  100.000000,,   60.000000,,   50.000000,$
 1000.000000,MMPM,   70.000000,RPM,CLW,$
ON,,NOTE
TPRINT/T1 Fraise 2 tailles D 10,T1 Fraise 2 tailles D 10,T1
Fraise 2 ta$
illes D 10
LOADTL/1,1,1
$$ TOOLCHANGEEND
$$  Fin de génération de: Changement outil.2
$$ OPERATION NAME : Plans parallèles.1
$$  Début de génération de: Plans parallèles.1
LOADTL/1,1
SPINDL/   70.0000,RPM,CLW
RAPID
GOTO  /   22.83368,  -11.00506,    5.05575, 0.000000,-0.707107,
0.707107
RAPID
GOTO  /   22.83368,   -7.54027,    1.59096, 0.000000,-0.707107,
0.707107
FEDRAT/  300.0000,MMPM
```

Figure 4.2—Programme d'usinage avec une trajectoire en G1

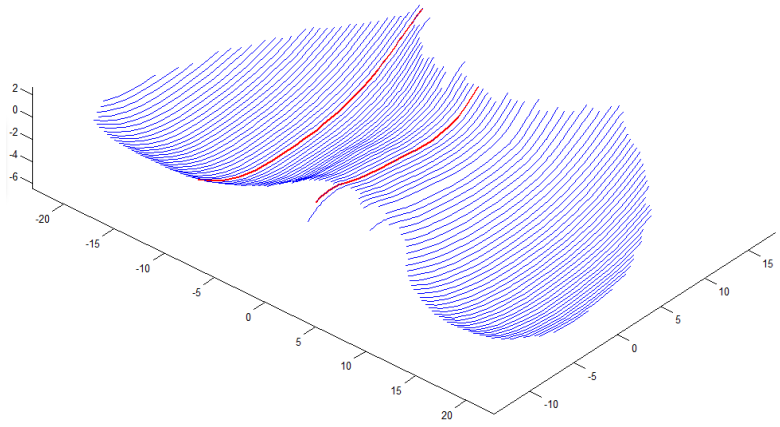


Figure 4.3 — Trajectoire d'outil pour la réalisation de la prothèse

A partir de la trajectoire globale nous avons sélectionné deux passes de forme et emplacements différents sur lesquelles nous avons appliqué la méthode et nous avons obtenu les résultats suivants :

4.3.1. Premier cas de passe :

a) Avec une seule courbe (un seul polynôme):

En premier lieu l'approximation des points est effectuée avec une seule courbe (figure 4.4). C'est l'équivalant d'une approximation polynomiale normale sans éléments finis.

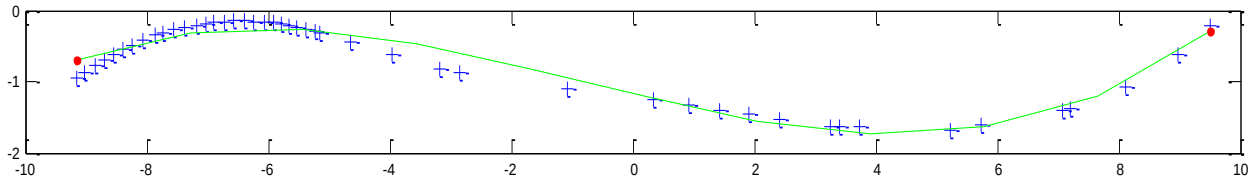


Figure 4.4 — Approximation avec une seule courbe polynômiale

L'exécution du programme nous a donné les résultats suivants :

Les coordonnées et les dérivés aux nœuds pour chaque élément:

x1	x2	z1	z2	z1'	z2'
9.5154	-9.1540	-0.2971	-0.7080	0.6257	0.3206

Les paramètres ai des polynômes:

a0	a1	a2	a3
-1.1776	-0.2063	0.0068	0.0026

L'erreur globale au moindres carrées est égale à : 0.495

b) Avec 4 éléments finis :

Pour cette foi la méthode des éléments finis est utilisée avec 4 éléments (figure 4.5)

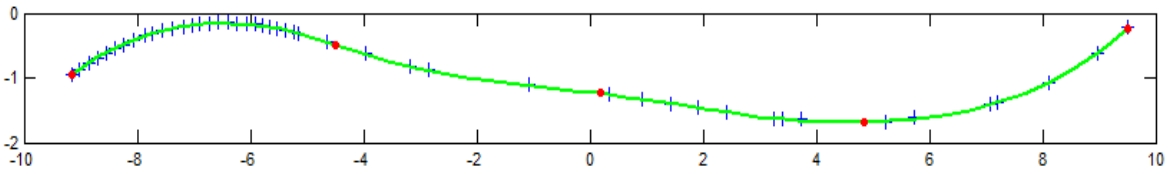


Figure 4.5 — Approximation par éléments finis avec 3 éléments

Les résultats obtenus sont les suivants :

Les coordonnées et les dérivés aux nœuds pour chaque élément :

x1	x2	z1	z2	z1’	z2’
9.5154	4.8481	-0.2384	-1.6813	0.7738	0.0361
4.8481	0.1807	-1.6813	-1.2379	0.0361	-0.1044
0.1807	-4.4867	-1.2379	-0.4894	-0.1044	-0.3174
-4.4867	-9.1540	-0.4894	-0.9492	-0.3174	0.6349

Les paramètres ai des polynômes:

a0	a1	a2	a3
-2.4479	0.4869	-0.1104	0.0088
-1.2198	-0.0951	-0.0271	0.0056
-1.2193	-0.1013	-0.0071	-0.0046
-2.6883	-0.5515	0.0111	0.0055

Erreur globale au moindres carrées est égale à : 0.002

1.1.1. Deuxième cas :

Pour tester le programme pour différentes formes de trajectoires ou de passes ce deuxième cas est traité. Comme pour le premier cas l’approximation est réalisée au départ avec une seule courbe (ou élément), ce qui représente la méthode classique, puis avec 4 éléments finis afin de faire une comparaison.

a) Avec un seul polynôme :

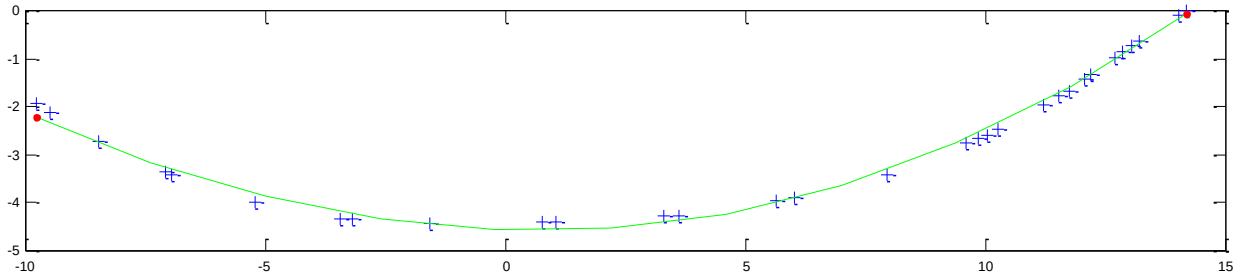


Figure 4.5 — Approximation une seule courbe polynômiale

Les coordonnées et les dérivés aux nœuds pour chaque élément:

x1	x2	z1	z2	z1’	z2’
14.1851	-9.7778	-0.0695	-2.2079	0.6949	-0.4383

Les paramètres ai des polynômes:

a0	a1	a2	a3
-4.5735	-0.0325	0.0227	0.0001

Erreur globale au moindres carrées est égale à: 0.351

b) Avec 4 éléments :

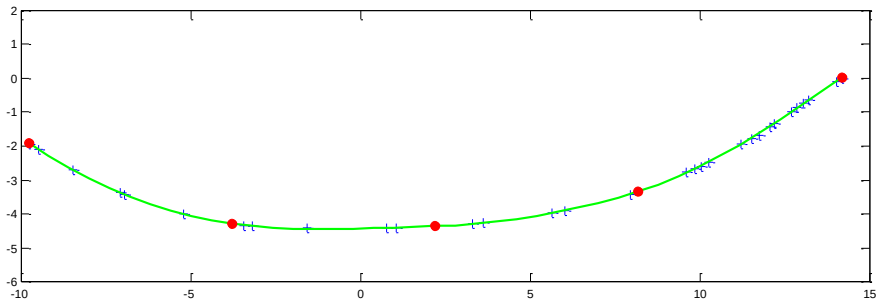


Figure 4.7—Approximation avec 4 éléments finis

Les coordonnées et les dérivés aux nœuds pour chaque élément:

x1	x2	z1	z2	z1’	z2’
14.1851	8.1944	0.0341	-3.3349	0.7109	0.2984
8.1944	2.2037	-3.3349	-4.3562	0.2984	0.0348
2.2037	-3.7871	-4.3562	-4.2737	0.0348	-0.1347
-3.7871	-9.7778	-4.2737	-1.9129	-0.1347	-0.6623

Les paramètres ai des polynômes:

a0	a1	a2	a3
0.2414	-1.3871	0.1424	-0.0032
-4.3144	-0.0738	0.0254	-0.0002
-4.4306	0.0229	0.0094	-0.0020
-4.1976	0.1714	0.0390	-0.0002

Erreur globale au moindres carrées est égale à : 0.009

4.3.3. Influence du nombre d’éléments sur l’erreur d’approximation :

Le tableau ci-dessous nous montre l’influence du nombre d’éléments utilisés sur l’erreur d’approximation.

Nombre d’éléments	Erreur globale (mm)	
	Cas 1	Cas 2
1	0,495	0,351
2	0,061	0,022
3	0,018	0,014
4	0,002	0,009
5	0,002	0,006

Le contenu du tableau est visualisé sur le graph de la figure 4.8 pour voire l'évolution globale de l'erreur.

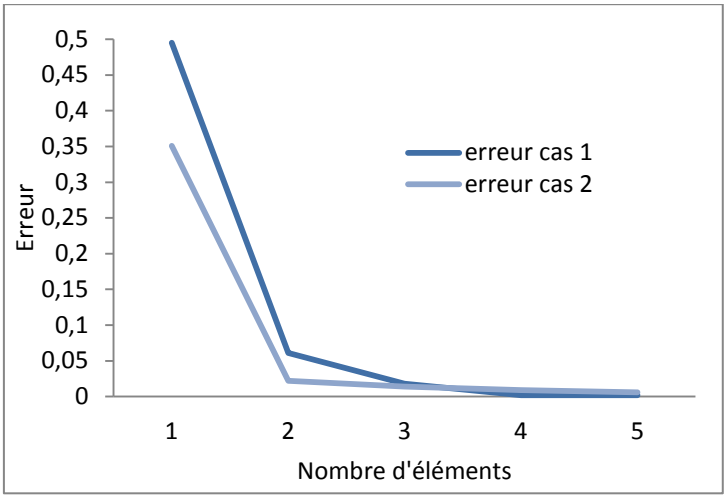


Figure 4.8 : Approximation par la méthode des élément finis avec 4 éléments

4.3. Discussion des résultats :

L'analyse des résultats obtenus pour le cas pratique traité nous permet de déduire que :

1- Le programme nous fourni une bonne solution en éléments finis à savoir un ensemble de courbes polynomiales attachées les une aux autres avec une continuité au niveau des points de jonction. Cette continuité peut être vérifiée à travers les valeurs des coordonnées et des dérivés aux nœuds calculés sur les deux éléments voisins séparément qui sont égaux.
2- L'erreur d'approximation globale au sens des moindres carrées est très réduite même avec un très petit nombre d'éléments qui est 4 éléments où elle est de 0,002 et 0,009.
3- Comme pour n'importe quelle application de la méthode des éléments finis, l'erreur finale d'approximation dépend du nombre d'éléments utilisés. Selon le graphe réalisé, elle diminue rapidement pour atteindre la valeur de 0,003. Cette variation va nous permettre de choisir le nombre d'éléments pour obtenir une précision qui correspond à une application donnée.
4- Les résultats montrent que la méthode garde ses performances pour différentes formes de trajectoires.
5- L'évolution de l'erreur globale varie d'une trajectoire à une autre mais reste très petite.

Conclusion générale et perspectives

L'objectif de ce travail est d'améliorer la qualité d'usinage des pièces mécaniques. Pour le faire, l'idée adoptée était de réduire les discontinuités dans la trajectoire d'outil en passant d'une trajectoire formée de plusieurs segments G1 à une trajectoire composée de courbes polynomiales.

Après une introduction sur l'usinage numérique et la présentation de la problématique, les différentes méthodes d'interpolation et d'approximation ont été exposées. Le choix a porté sur l'utilisation de d'approximation par éléments finis. Cette dernière à été détaillée dans un chapitre à part. Pour exploiter la méthode des éléments finis, comme n'importe quelle méthode numérique, il faut passer par un programme, chose qui a été faite par Matlab. Pour valider et tester ce programme, nous l'avons exécuté avec des données issues d'un cas réel ou la précision est très exigée qui est l'usinage d'une prothèse du genou.

Les résultats obtenus sont très encourageants est ont permet d'éliminer les segments de droite qui compose initialement la trajectoire d'outil et de les remplacés par un nombre très réduit de courbes polynomiales. La précision atteinte était très importante même avec un nombre de courbes polynomiales qui ne dépasse pas 4 courbes. Même avec ces résultats, ce travail accepte toujours des améliorations et peut être développé pour être plus performant et couvrir un maximum de cas pratique d'application. Comme perspective d'amélioration de ce travail on peut citer :

- Le développer une partie pour déterminer le nombre optimale de courbe qu'il faut utiliser
- Le développer une partie pour trouver une répartition optimale des nœuds sur la trajectoire ce qui va permettre de réduire d'avantage le nombre de courbe polynomiale à utiliser.
- L'améliorer la méthode pour effectuer un lissage dans la direction perpendiculaire à l'avance pour améliorer d'avantage l'état de surface obtenu.
- La réalisation de mesures sur les surfaces usinées pour vérifier la qualité réelle obtenues et évaluer les améliorations apportées.