PUBLICATIONS MATHÉMATIQUES ET INFORMATIQUES DE RENNES

ALAIN PONCET

Écriture d'un code d'éléments finis

Publications des séminaires de mathématiques et informatique de Rennes, 1975, fascicule S3
« Journées « éléments finis » », , p. 1-16

<http://www.numdam.org/item?id=PSMIR_1975___S3_A8_0>

© Département de mathématiques et informatique, université de Rennes, 1975, tous droits réservés.

L'accès aux archives de la série « Publications mathématiques et informatiques de Rennes » implique l'accord avec les conditions générales d'utilisation (http://www.numdam.org/conditions). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM
Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques
http://www.numdam.org/

ECRITURE D'UN CODE D'ELEMENTS FINIS

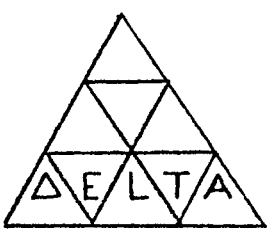
Alain PONCET*

Nous présentons ici un système interactif pour la résolution d'équations aux dérivées partielles sur des domaines bornés du plan.

Ce système se compose d'un ensemble de programmes qui permettent à ses utilisateurs de formuler puis de résoudre de tels problèmes par des méthodes d'éléments finis triangulaires sans avoir à programmer ou à préparer des données sur fichiers. En particulier, il fournit des triangulations automatiques et permet d'améliorer une triangulation existante. Se présentant sous la forme d'un langage de commandes, il prévoit et favorise les interventions humaines en cours de traitement, pour décrire ou modifier :

- les domaines d'intégration et leur discrétisation,
- les équations et conditions aux limites,
- les procédés d'interpolation,

ou pour conduire des itérations, les résultats obtenus pouvant être visualisés au fur et à mesure sur l'écran d'un terminal graphique.



* Attaché de recherche au CNRS, Université Scientifique et Médicale de Grenoble

- 0 - INTRODUCTION

Le traitement sur ordinateur d'un problème aux dérivées partielles avec conditions aux limites par des méthodes d'éléments finis, peut se scinder en 4 étapes :

1) l'obtention d'un domaine d'intégration discrétisé,
2) la définition d'un procédé d'interpolation,
3) la description des équations et des conditions aux limites,
4) la résolution du problème approché ainsi posé.

Les 3 premières étapes sont totalement indépendantes et la dernière peut être abordée dès que les 3 autres l'ont été au moins une fois avec succès. C'est pourquoi notre système fait correspondre une commande, donc un programme, à chacune de ces étapes ; ainsi :

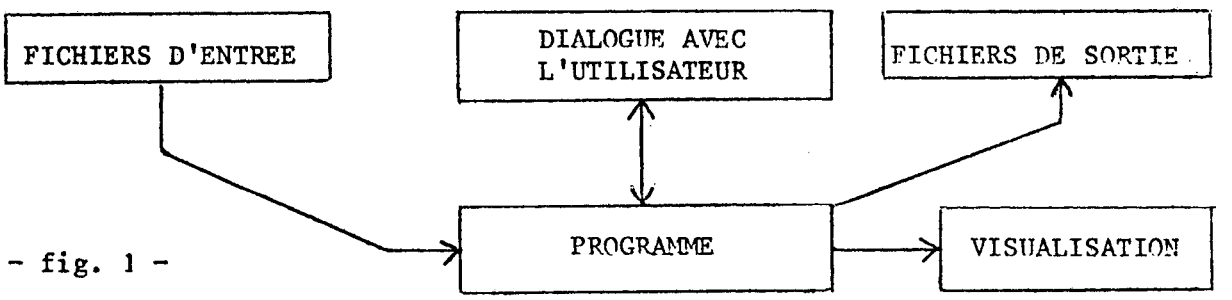
- la commande DOMAINE permet de décrire un domaine borné du plan (par ses frontières), de le trianguler grâce à un algorithme de triangulation frontale automatique (contrôlant la taille des triangles et l'évolution du front et évitant une re-numérotation) ou de le corriger "à la main" en utilisant les techniques graphiques ;

- la commande EQUATION codifie les formulations variationnelles et les conditions aux limites (coefficients variables et paramétrés) ;

- la commande APPROCHE construit et visualise les polynômes de base pour l'interpolation au sens de Lagrange ou Hermite ;

- la commande SOLUTION résoud les problèmes formulés lors de l'activation des trois commandes précédentes (le système linéaire est résolu par le procédé d'élimination frontale) et visualise les résultats approchés : valeurs ponctuelles, courbes de niveau, contraintes, déplacements ; elle permet de reprendre le traitement après corrections de frontières ou de paramètres (exemple : frontières libres, petits déplacements successifs, certains problèmes d'évolution, etc...).

Chacune de ces commandes a la stucture suivante :



- fig. 1 -

Les fichiers d'entrée de la commande SOLUTION seront les fichiers de sortie des 3 autres commandes et les fichiers d'entrée de celles-ci auront la même structure que les fichiers de sortie correspondant mais ils seront facultatifs.

- I - LA COMMANDE "DOMAINE"

-I-1- Description des domaines d'intégration. Frontières, bornes, ceinture.

Soit Ω un ouvert borné de $\mathbb{R}^2$, domaine d'intégration pour un problème aux limites.

Soit Γ sa frontière, découpée en parties γ_i (encore appelées frontières) : $\Gamma = \bigcup_i \gamma_i$; on désigne par Γ_E le sous-ensemble des frontières extérieures et par Γ_I celui des frontières intérieures. On introduit un ensemble Γ_{II} de courbes bornées dans que l'on nomme "fronts intermédiaires" et on note $\tilde{\Gamma} = \Gamma \cup \Gamma_{II}$. Γ_{II} sera spécifié par l'utilisateur et lui permettra de partager Ω en sous-domaines Ω_i plus simples et admissibles pour les algorithmes de triangulation (au minimum fortement connexes).

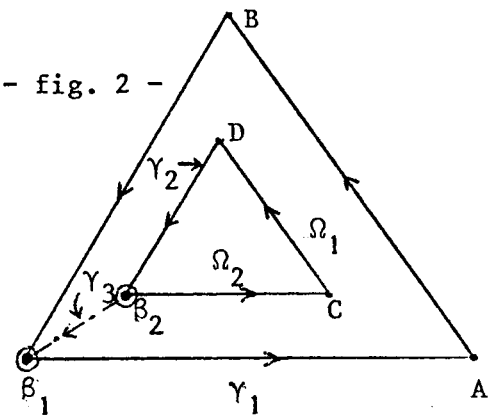
On appellera <u>borne</u> chaque extrémité de frontière.

On appellera <u>ceinture</u> la liste ordonnée des frontières orientées formant une boucle simple fermant un sous-domaine.

Chaque frontière sera décrite par ses 2 bornes et une liste de coordonnées fournies soit explicitement, soit par l'intermédiaire d'un "tracé" manuel sur un écran cathodique ou une tablette connectés à l'ordinateur dont on prélèvera un maximum de points. Cette liste ordonnée fixera une orientation de la frontière. (L'implémentation d'une description analytique de frontières est à l'étude).

<u>Exemple</u> : Problème à milieu hétérogène .

- fig. 2 -



$\gamma_1 \in \Gamma_E$ défini par (β_1, A, B, β_1)
$\gamma_2 \in \Gamma_I$ défini par (β_2, C, D, β_2)
$\gamma_3 \in \Gamma_{II}$ défini par (β_2, β_1)
$\Omega = \Omega_1 \cup \Omega_2 \cup \gamma_3$
ceinture de Ω_1 : $(\gamma_1, -\gamma_3, -\gamma_2, \gamma_3)$
ceinture de Ω_2 : (γ_2)

A chaque frontière sera attribué un "type" caractérisant la condition portant sur celle-ci et indiquant, par son signe, son appartenance à Γ_E, Γ_I ou Γ_{II} : par convention, les types seront positifs sur Γ_E, négatifs sur Γ_I, nuls sur Γ_{II}.

-I-2- <u>Discrétisation des frontières</u>

La discrétisation de chaque sous-domaine débute par la discrétisation de sa ceinture, ce qui fournit les premières bases des triangles à construire à l'intérieur. Le problème est donc ici de choisir sur celle-ci un certain nombre de points dont les distances entre eux soient contrôlées par l'utilisateur.

Celui-ci attribuera donc, à chaque borne, une valeur indicative pour le pas de discrétisation et les noeuds-frontières seront répartis entre 2 bornes, de sorte que les distances entre eux suivent une progression arithmétique entre les 2 valeurs indicatives, ceci permettant d'avoir des pas très variables d'une région à une autre.

Les bornes, quant à elles, seront systématiquement des noeuds de discrétisation.

La visualisation simultanée de chaque frontière "brute" et de la frontière discrétisée correspondante permet à l'utilisateur de critiquer cette dernière et éventuellement de la reprendre (cf. fig. 5).

-I-3- Algorithme de triangulation automatique

Nous présentons dans ce paragraphe un algorithme qui permet de construire dynamiquement des triangles sur un sous-domaine dont la ceinture a été discrétisée comme indiqué ci-dessus. Cette construction dynamique, proche de celle de George |5|, s'opère cependant dans un ordre différent, bien adapté à la technique d'élimination frontale (cf. Irons |6|, Aussems |1|), de telle sorte que l'on n'ait pas de renumérotation à effectuer.
Dans la même perspective, un calcul d'adresse est fait en parallèle, afin de déterminer l'emplacement en mémoire des inconnues discrètes et de leurs équations, lors de la résolution approchée d'un problème aux limites.

Chaque pas de l'algorithme détermine les coordonnées d'un nouveau noeud de la triangulation ainsi que ses liaisons avec les noeuds déjà construits (noeuds-frontières au départ).

-I-3-1- Notion de front

On appellera front à un pas donné de l'algorithme la liste ordonnée des noeuds situés à la frontière entre la partie de Ω_i qui est déjà triangulée et celle qui ne l'est pas encore. La largeur du front sera le nombre de noeuds qu'il contient.

-I-3-2- Critères d'optimisation

- Un front de largeur minimum fournit un encombrement minimum pour la mémorisation des équations en les inconnues discrètes (à rapprocher de la largeur de la bande de la matrice du système linéaire).

- Des triangles, dont les plus petits angles soient maximaux donc les plus proches possible de triangles équilatéraux, donnent le meilleur encadrement de l'erreur (cf. Raviart |3| |10|).

Sans être formalisés d'avantage ici, ces deux critères nous guiderons dans la construction des éléments, sans toutefois nous conduire à une triangulation "optimale" (cf. I-4).

-I-3-3- Nombre de liaisons libres en un noeud du front

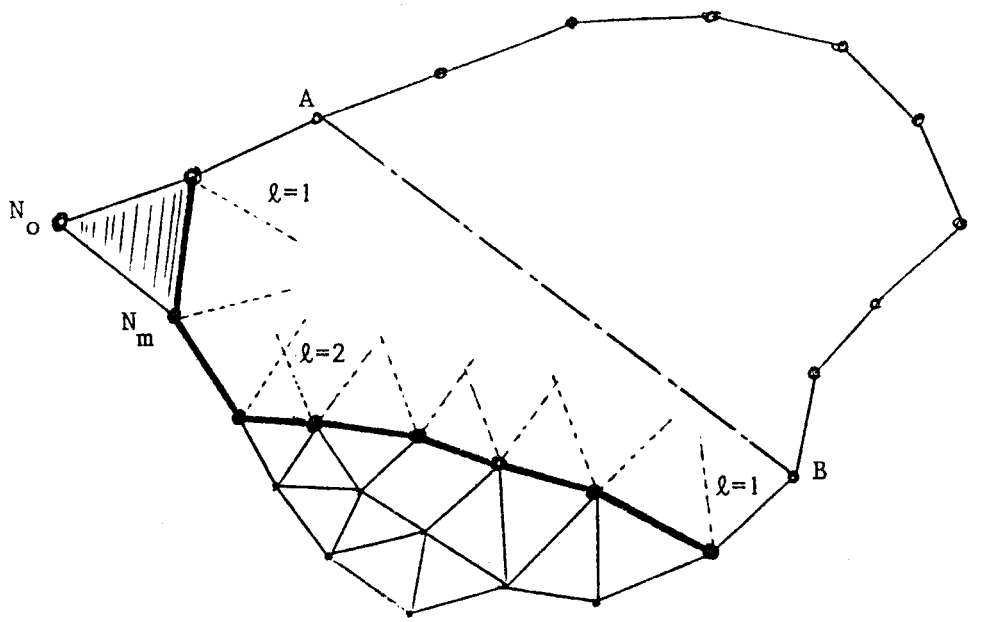
Soit θ l'angle que fait un noeud avec ses deux voisins sur le front ; déterminons le nombre ℓ de liaisons qu'aura ce noeud avec les noeuds à construire, de la manière suivante :
$\pi/3$ étant l'angle optimal au sens de la majoration de l'erreur sur un triangle, on comparera θ et ses sous-multiples à $\pi/3$ et ℓ sera tel que :

$$\left|\frac{\theta}{\ell+1} - \frac{\pi}{3}\right| = \inf_{i \geq 0} \left|\frac{\theta}{i+1} - \frac{\pi}{3}\right|$$

ce qui nous permet de représenter ℓ par rapport à θ dans le tableau suivant :

θ	0	$4\pi/9$	$4\pi/5$	2π
ℓ	0	1	$\ell \geq 2$	

Fig. 3 : Exemple



-I-3-4- Corde et flèche

Soient A et B les noeuds extrêmes du front, donc situés sur $\tilde{\Gamma}$; on appellera "corde" le segment AB.
A, B et un pas de discrétisation étant fixés, il est bien évident que lorsque les noeuds du front sont sur la corde AB, ils sont en nombre minimum, c'est pourquoi on cherchera à rapprocher le front de sa corde.

Soit N un noeud du front, on appellera "flèche pondérée" en N, la quantité :

$$f_N = \frac{\vec{AB} \wedge \vec{AN}}{S_N \cdot (\ell_N + 1)}$$

où S_N est la surface souhaitée pour les triangles voisins de N et ℓ_N le nombre de liaisons libres en N, S_N étant interpolés entre les deux valeurs connues : S_A et S_B.
Le signe de f_N donne la place et sa valeur absolue l'éloignement de N par rapport à la corde AB. La notion de distance ainsi introduite prend en compte d'une part la taille des triangles et d'autre part la priorité à donner aux zones très "fermées" du front. Si la corde rencontre une frontière ailleurs qu'en A et B, l'algortihme devient défaillant.

-I-3-5- Progression du front sur les frontières

L'extrémité A ou B du front est avancée jusqu'au prochain noeud-frontière ayant au moins 2 liaisons libres dans les 3 cas suivants :

1) Tous les noeuds du front ont une flèche négative.
2) Un triangle admet cette extrémité comme sommet.
3) L'utilisateur désire faire progresser le front dans cette direction et l'indique à l'aide du pointeur optique sur l'écran du terminal graphique où la triangulation est visualisée au fur et à mesure de son élaboration.

-I-3-6- Description d'un pas de l'algorithme

1 - Progression du front sur les frontières, construction d'un triangle pour chaque noeud du front sans liaison libre : N_o.

2 - Recherche du noeud à flèche pondérée maximum (= N_m)

3 - Recherche des noeuds N_i voisins de N_m ayant une liaison libre = détermination des bases des nouveaux triangles.

4 - Calcul des coordonnées du nouveau point :
on construit sur chaque segment $N_i N_{i+1}$ les triangles isocèles de surface S_{N_i} et l'on prend pour nouveau noeud le barycentre des sommets des triangles isocèles.

Ces coordonnées ainsi grossièrement calculées peuvent être optimisées dans une phase ultérieure (cf. paragraphe I-4).

5 - Diverses mises à jour du nouveau front :
Calcul des surfaces S_N et des liaisons ℓ_N modifiées par la création du nouveau point, stockage en mémoire, visualisation, etc...

-I-3-7- Calcul d'adresses pour l'assemblage frontal des équations

Parallèlement à cette construction, l'algorithme prépare le calcul des adresses qu'occuperont les inconnues discrètes lors de l'assemblage du système linéaire : lorsqu'un noeud entre dans le front, un pointeur lui est attribué (première valeur trouvée dans la liste des pointeurs libres) ;

lorsqu'un noeud quitte le front, son pointeur est libéré et ceci est indiqué par un changement de signe dans la codification du dernier triangle qui le compte comme sommet. (Nous savons qu'alors, l'équation en ce noeud peut être complétement formée et être éventuellement stockée en mémoire secondaire).

-I-3-8- Exemple de triangulation automatique

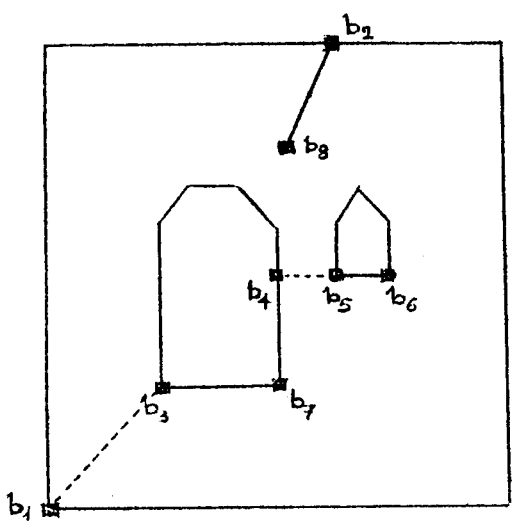
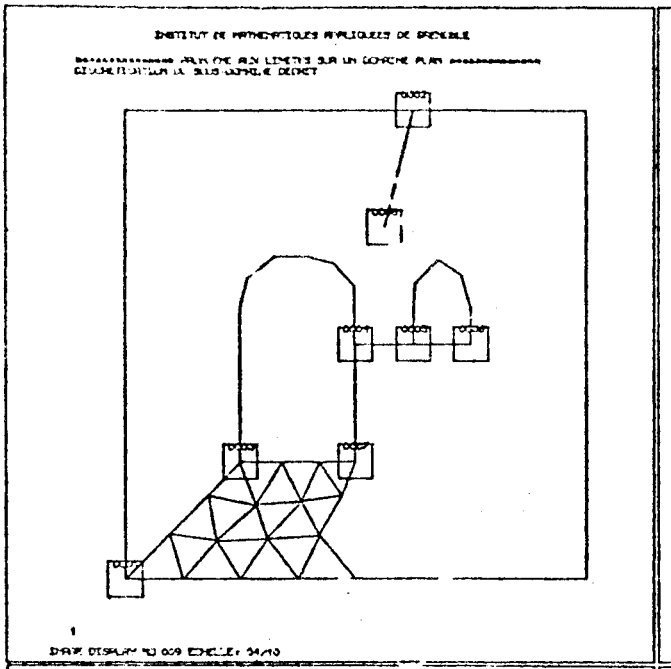


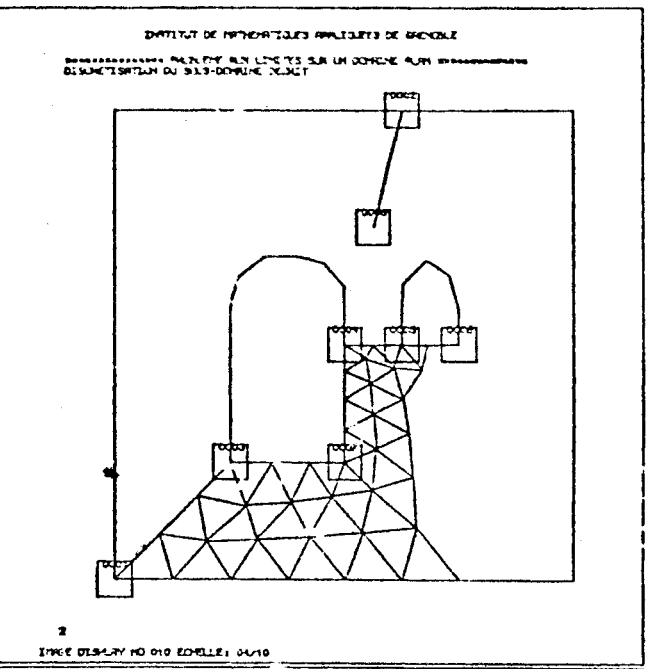
fig. 4

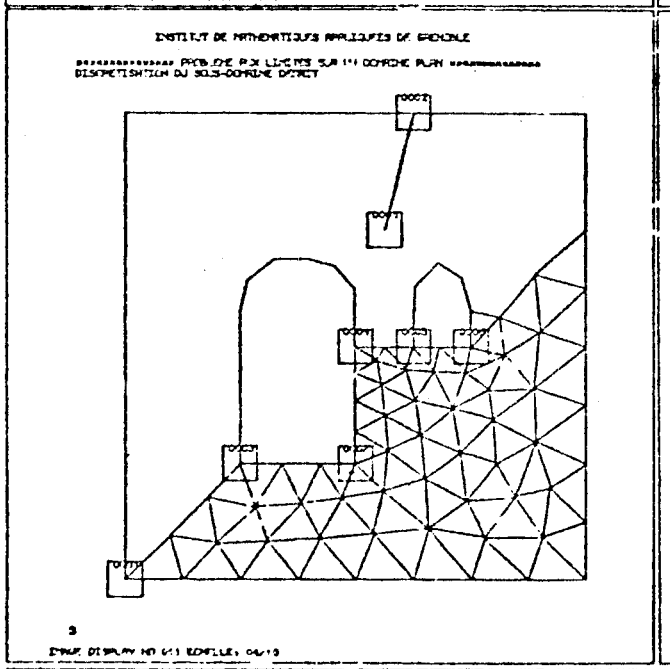
Considérons le domaine suivant qui n'est pas directement triangulable.
On lui ajoute deux frontières fictives b_1b_3 et b_4b_5 et on lui applique l'algorithme décrit précédemment.

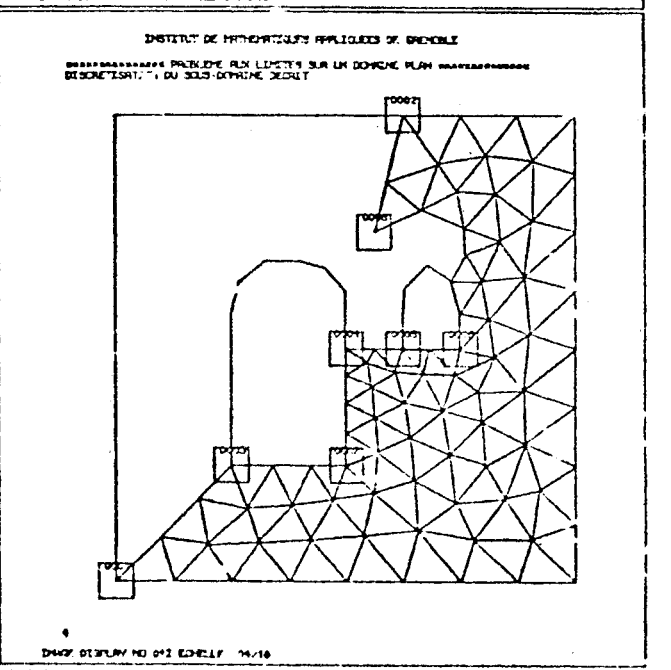
On trouvera en page suivante quelques phases de la triangulation de ce nouveau domaine.

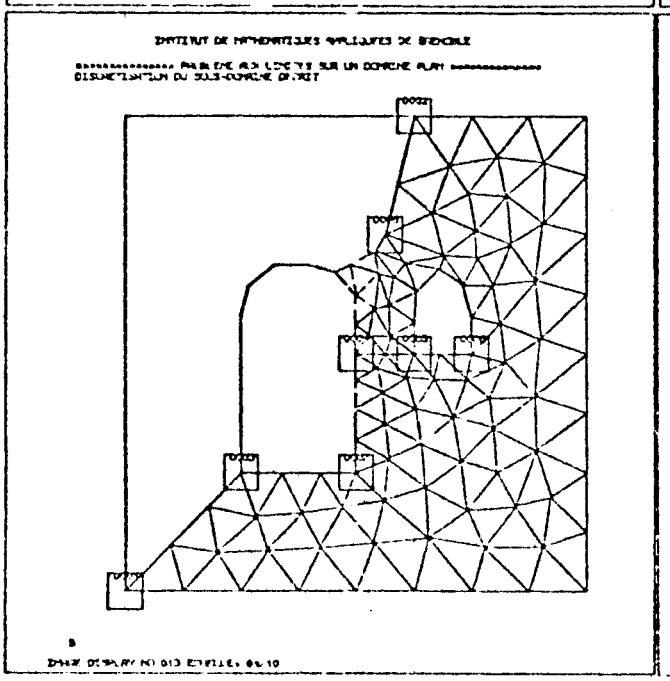
On obtient (en 5 sec. IBM 360) un découpage en 232 triangles, représenté page 6.

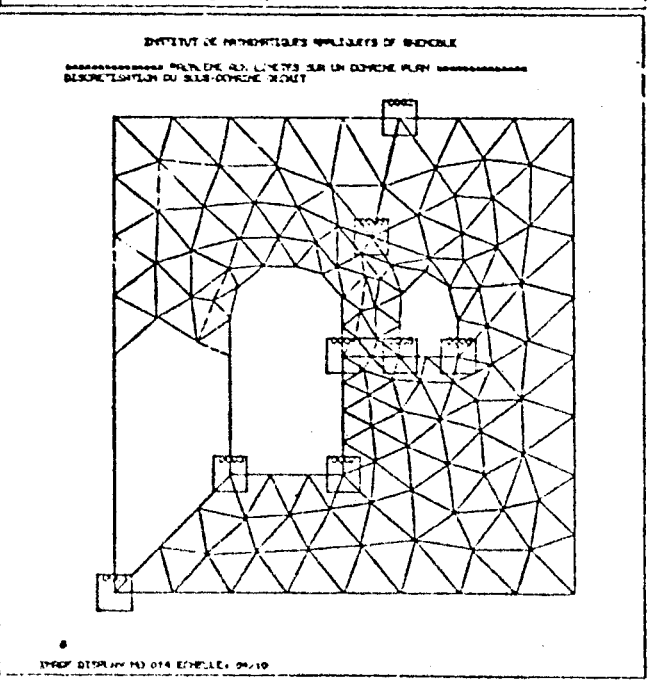

INSTITUT DE MATHEMATIQUES APPLIQUEES DE GRENOBLE
1

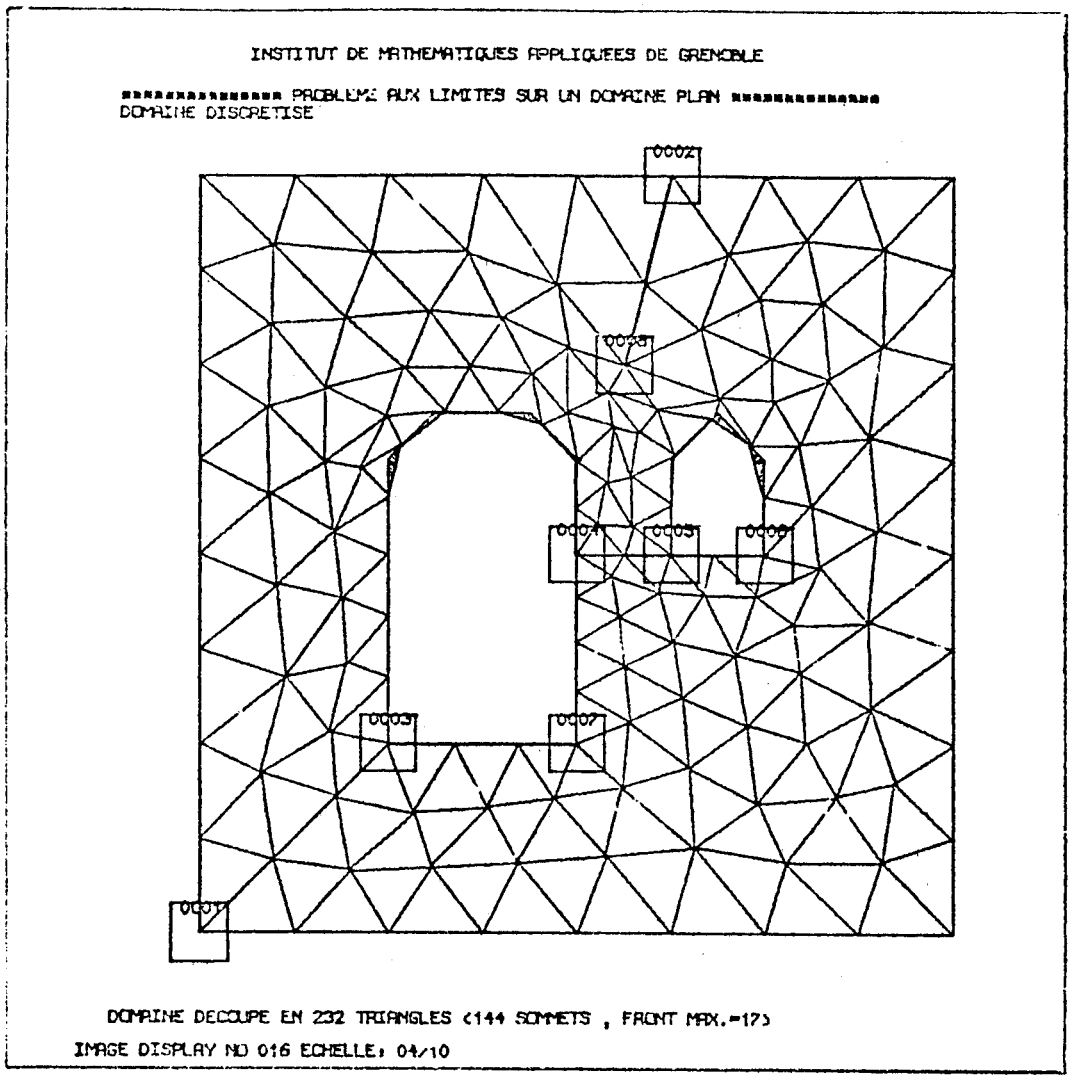

INSTITUT DE MATHEMATIQUES APPLIQUEES DE GRENOBLE
2


INSTITUT DE MATHEMATIQUES APPLIQUEES DE GRENOBLE
3


INSTITUT DE MATHEMATIQUES APPLIQUEES DE GRENOBLE
4


INSTITUT DE MATHEMATIQUES APPLIQUEES DE GRENOBLE
5


INSTITUT DE MATHEMATIQUES APPLIQUEES DE GRENOBLE
6



- Fig. 5 -

Exemple de triangulation

Les carrés numérotés symbolisent les bornes ;

l'espace entre frontière brute et frontière discrétisée est hachuré.

-I-4- Le problème de l'optimisation d'une triangulation

-I-4-1- Triangulation optimale d'un domaine borné

Dégageons nous du contexte frontal et examinons les propriétés géométriques d'une triangulation T_h, d'un ouvert Ω de $\mathbb{R}^2$, définie par

$$T_h = (F_h, N_h, L_h)$$

où F_h est la liste des coordonnées des noeuds-frontières,

N_h la liste des coordonnées des noeuds intérieurs,

L_h la liste des liaisons entre les noeuds, à laquelle on peut faire correspondre un graphe qui sera le graphe de la triangulation T_h.

Les triangles dont la réunion forme le domaine discrétisé Ω_h sont les plus petits polygones que l'on peut construire avec ces 3 listes.

Soit E un ensemble quelconque de triangulation de Ω et f une fonction réelle sur l'ensemble des triangles du plan (isomorphe à $\mathbb{R}^6$).

Définition : T_h sera dite optimisée pour f sur E si

$$\inf_{T \in \Omega_h} f(T) \text{ est maximum sur } E.$$

Exemple : Soit T un triangle de Ω_h, f(T) le plus petit angle de T ;
$\inf_{T\in\Omega_h} f(t)$ est donc le plus petit angle de la triangulation.

Si T_h est optimisé pour ce choix de f, la majoration de l'erreur pour la solution approchée d'un problème aux limites sur Ω donnée dans Raviart |10| sera la meilleure possible ; on dira alors que T_h est une triangulation optimale.

Le choix le plus général pour E est l'espace $(\mathbb{R}^{2N}, \mathbb{R}^{2N}, \mathbb{N}^{2xN})$ mais, outre la grande complexité du problème situé dans ce cadre, l'absence de contrôle de la taille des éléments, donc de leur nombre, nous conduit à commencer beaucoup plus modestement par considérer le cas :

$$E = (F_h, \mathbb{R}^{2N}, L_h)$$

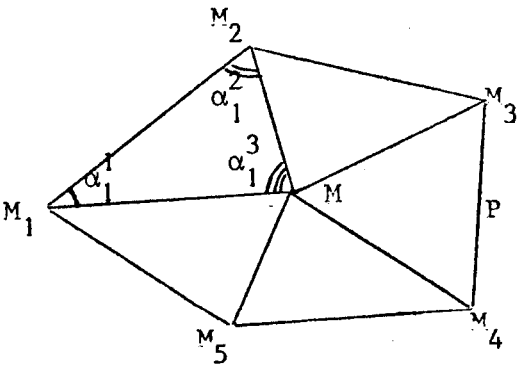
où les noeuds-frontières et le graphe de la triangulation sont connus et où N est le nombre de noeuds.

Nous souhaitons donc élaborer un algorithme qui permette d'approcher une triangulation optimale sur E donné ci-dessus en partant de l'idée suivante : Considérons un noeud de la pré-triangulation qui a fourni F_h et L_h et cherchons sa position optimale lorsque ses voisins sont fixés : cela nous ramène à une optimisation dans $E = (F_h, \mathbb{R}^2, L_h)$ et parcourons itérativement tous les noeuds de la triangulation.

Problème de l'optimisation locale

Soient M le noeud dont on cherche la position optimale et P le polygone que forment ses n voisins.

Soit $\{\alpha_i^j\}$ l'ensemble des angles que M détermine avec P, chaque α_i^j est une fonction des coordonnées x et y de M :



- fig. 6 -

Soient a une valeur d'angle comprise entre 0 et $\pi/3$ et

$$a_+ = \{(x,y) \mid \inf_{i,j} \alpha_i^j \geq a\}.$$

Par construction, a_+ est un convexe borné dont la frontière est l'enveloppe convexe des courbes de niveau $\alpha_i^j = a$ pour tout i de 1 à n et tout j de 1 à 3.

La position optimale de M est donc obtenue en cherchant le plus grand a^* tel que $a_+^* \neq \emptyset$.

Proposition (évidente) : a_+^* est d'intérieur vide ; il est :

- soit réduit à un point, c'est alors le point M optimal,
- soit réduit à un segment de droite auquel cas la position optimale n'est pas unique.

a^* est inférieur ou égal à la moitié du plus petit angle de P.

Recherche d'algorithmes fournissant une position localement optimale du point M (à ε près).

Considérons un algorithme itératif qui permette de construire un point M_{n+1} sur la frontière d'un convexe a_+^{n+1} à partir d'un M_n et d'un a_+^n ; $a_+^{n+1} \subset a_+^n$ et la suite croissante des angles a_n est destinée à converger vers a^* .

Proposition : Soit r_n le rayon du plus petit cercle inscrit dans a_+^n .
Tout algorithme tel que pour tout n, il existe K indépendant de n tel que $\sin(a_{n+1} - a_n) \geq K\, r_n > 0$ converge vers un point M^* de l'ensemble a_+^* des solutions optimales.

Considérons maintenant l'itération qui consiste à chercher successivement une position optimale pour chaque noeud : on obtient une suite d'angles minimaux qui est croissante et bornée donc convergente. La question qui reste ouverte est celle de savoir si les angles obtenus à la limite fournissent bien le maximum pour f(T).

Un problème "dual"

Ayant constaté (contre-exemple) que l'on n'avait pas, en général, unicité du noeud M(x,y) localement optimal à P fixé, on peut se poser le problème de la recherche des points qui engendrent avec P des angles dont le plus grand soit minimum : on sait par exemple que si le plus grand angle d'une triangulation est aigu, la matrice du problème de Dirichlet discrétisé est non-négative |4|.

On construit $A_- = \{x,y \mid \alpha_i^j(x,y) \leq A \quad \forall i = 1, n \quad \forall j = 1, 2, 3\}$

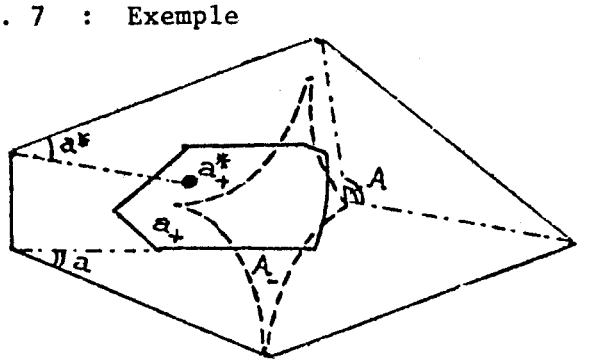
où A est une valeur d'angle entre $\pi/3$ et π.

A_- a de moins bonnes propriétés que a_+, il n'est pas convexe et n'est même pas toujours connexe ; cependant, en cherchant le minimum du plus grand angle seulement sur a_+^*, on obtient la :

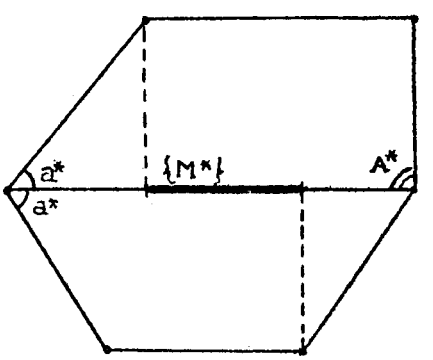
Proposition : S'il existe une droite contenant a_+^* ne coupant pas P en deux de ses sommets, alors il existe un point $M^*(x^*,y^*)$ unique tel que

$$M^* \in a_+^* \text{ et } \operatorname*{Sup}_{i,j} \alpha_i^j(x^*,y^*) = \operatorname*{Inf}_{M \in a_+^*} \operatorname*{Sup}_{i,j} \alpha_i^j(x,y)$$

Fig. 7 : Exemple



contre exemple



- II - LA COMMANDE "EQUATION"

-II-1- Un cadre général pour les formulations variationnelles et les conditions aux limites

Soit un problème aux limites dont la solution est supposée être dans un espace E de fonctions définies sur Ω, plus précisément dans le sous-espace V des fonctions de E vérifiant les conditions aux limites.

V sera défini implicitement par un système d'équations linéaires entre les inconnues u_i, leurs dérivées normales $\frac{\partial u_i}{\partial N}$ et tangentielles $\frac{\partial u_i}{\partial T}$ sur la frontière Γ de Ω, système de la forme :

$$(1)\quad \sum_i \left(b_{ik}\, u_i + c_{ik} \frac{\partial u_i}{\partial N} + d_{ik} \frac{\partial u_i}{\partial T}\right) = g_k$$

sur les frontières de type k.

La fonctionnelle J(u) à minimiser sur V sera de la forme

$$(2)\quad J(u) = \int_\Omega \sum_{i,j,k,h} a_{ijkh} \frac{\partial u_i}{\partial x_k} \cdot \frac{\partial u_j}{\partial x_h} + \sum_i f_i\, u_i + \int_\Gamma \sum_{i,j} u_j \left(b'_{ij} u_i + c'_{ij} \frac{\partial u_i}{\partial N} + d'_{ij} \frac{u_i}{\partial T} + g'_i\right)$$

avec, par convention d'écriture : $\frac{\partial u_i}{\partial x_o} \equiv u_i$, $\frac{\partial u_i}{\partial x_1} \equiv \frac{\partial u_i}{\partial x}$, $\frac{\partial u_i}{\partial x_2} \equiv \frac{\partial u_i}{\partial y}$.

Cette formulation recouvre les problèmes classiques : Dirichlet, Neumann, mélés élasticité linéaire, ainsi que des problèmes que l'on résoud itérativement et dont chaque itération se ramène à un tel problème (par exemple les problèmes d'évolution où la dérivée par rapport au temps est approchée par une différence non centrée d'ordre 1).

-II-2- Génération des sous-programmes calculant les coefficients a, b, c, d, f et g en tout point de Ω

Il s'agit de permettre au programme de résolution approchée(cf. IV) de calculer tous les coefficients de (1) et (2) en tous les points d'intégration numérique sur les triangles, en admettant qu'ils puissent dépendre des coordonnées de ces points et de paramètres. Pour cela, la commande EQUATION interroge l'utilisateur, puis génère et compile les sous-programmes à activer lors de la commande SOLUTION pour effectuer ces calculs.

Ainsi seront succesivement demandés à l'utilisateur :
- les noms des inconnues du problème,
- les noms des deux variables d'espace,
- les noms des paramètres,
- les expressions (FORTRAN) des coefficients en fonction des noms précédents.

Les temps de calcul seront considérablement réduits en faisant la convention suivante : lorsque l'expression d'un coefficient sera vide (ligne blanche) cela signifiera que celui-ci est identiquement nul.

- III - LA COMMANDE "APPROCHE"

Le but de cette commande est de construire les polynômes de base pour l'interpolation au sens de Hermite sur un triangle quelconque, non applati, T.

Si l'utilisateur désire une autre interpolation polynomiale (cf. Chenin |2|), il devra fournir à la commande SOLUTION le fichier correspondant.

-III-1- Rappel sur l'interpolation au sens de Hermite sur un triangle

Supposons que la fonction à interpoler soit connue, ainsi que certaines de ses dérivées partielles, en N points, appelés noeuds, du triangle T.

Soit $D = \{D_j\}$ un ensemble fini d'opérateurs différentiels (pour condenser l'écriture, on notera D_o l'opérateur identité).

Soient (x_i, y_i), i = 1, N les coordonnées des noeuds, D_i une partie de D attaché au $i^{ème}$ noeud et J_i l'ensemble des indices dans D correspondant.

On se pose le problème d'interpolation suivant :

Trouver des polynômes $P_k(x,y)$ vérifiant les équations :

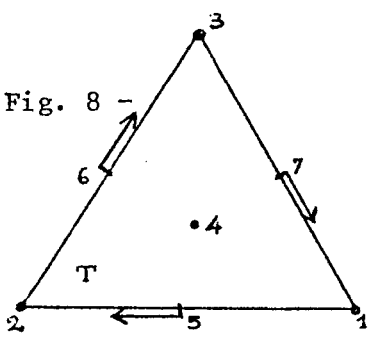
$D_j P_k(x_i,y_i) = \delta_{ijk\ell}$, i et k étant pris entre 1 et N, j étant pris dans J_i et ℓ dans J_k avec :

$\delta_{ijk\ell} = 1$ si i = k et j = ℓ, $\delta_{ijk\ell} = 0$ sinon.

Ces polynômes sont au nombre de : $\sum_{i=1}^{N} \mathrm{Card}(J_i)$.

Si D se réduit au seul opérateur identité, on retrouve l'interpolation au sens de Lagrange.

Exemple :



- Fig. 8 -

Prenons pour noeuds de T ses sommets, les milieux de ses côtés et son centre de gravité. Supposons que la fonction à interpoler soit connue en ces 7 noeuds et que ses dérivées tangentielles sur le bord le soient aux milieux des côtés ; nous avons alors :

$D = \{1, \frac{\partial}{\partial\tau}\}$; $D_i = \{1\}$ et $J_i = \{1\}$, i=1à4

$D_i = \{1, \frac{\partial}{\partial\tau}\}$ et $J_i = \{1,2\}$, i = 5 à 7.

$P_{5,2}$ est, par exemple, le polynôme de degré 3, nul au 7 noeuds et dont la dérivée tangentielle au bord vaut 0 aux nœuds 6 et 7 et 1 au noeud 5. (Cf. fig. 10).

-III-2- Calcul des polynômes de base : $P_{k\ell}(x,y)$

Dans un problème aux dérivées partielles, les valeurs de la (des) fonction(s) inconnue(s) et des dérivées aux noeuds de chaque triangle sont justement les inconnues discrètes du problème approché, c'est pourquoi on doit effectuer un calcul formel des polynômes $P_{k\ell}(x,y)$; ce calcul ne peut se faire que sur la base

des monômes canoniques : 1, x, y, x^2, xy, y^2, etc, ... où, fort heureusement, les dérivations formelles sont simples.

Choisissons parmi ces monômes une base finie $E = \{e_m\}$, m = 1 à M, et écrivons $P_{k\ell}(x,y) = \sum_{m=1}^{M} \alpha_{mk\ell}\ e_m$.

Le calcul se ramène donc à la résolution du système linéaire :

$$\sum_{m=1}^{M} \alpha_{mk\ell}\ D_j\ e_m(x_i,y_i) = \delta_{ijk} \qquad \begin{cases} \text{i et k de 1 à N} \\ j \in J_i,\ \ell \in J_k \end{cases}$$

Propriété : Si E engendre l'espace des polynômes de degré inférieur ou égal à k et si ce système admet une solution unique, alors l'élément fini ainsi construit est dit : "P_k-unisolvant" (cf. |10|).

Pour un autre choix de E, il faudra introduire des relations supplémentaires entre les monômes canoniques (implémentation à l'étude).

Remarque : Ce calcul ne sera pas fait nécessairement par l'ordinateur sur tout triangle de Ω_h. En effet :

1°) On peut d'abord faire ce calcul sur un élément de référence $\hat{T}$ puis se ramener à T par une transformation affine : ceci est efficace pour les "gros" éléments de type Lagrange, mais pas pour les éléments de type Hermite, les directions de dérivations étant bouleversées.

2°) Le calcul sur l'élément de référence lui-même peut, dans les cas simples, être mené à la main en choisissant convenablement $\hat{T}$.

-III-3- Codification

- Pour les éléments de type Lagrange, nous avons choisi la famille de Nicolaïdes |8| donc, si l'utilisateur donne à l'ordre maximum de dérivation pour les opérateurs de *D* la valeur 0, il lui suffira de donner le nombre de noeuds entre les sommets de chaque triangle.

Dans le cas contraire, on lui demandera de remplir 2 tableaux qui permettent de codifier l'interpolation au sens de Hermite :

1er tableau : codification de *D* ; à chaque ordre de dérivation, on associe les entiers suivants :

- l'ordre de dérivation,
- le nombre de directions de dérivations à cet ordre,
- les directions de dérivations codées ainsi : 0 pour ∂/∂_x, 1 pour ∂/∂_y, 2 pour ∂/∂_N, 3 pour ∂/∂_T.

Les dérivations aux différents ordres sont enchaînées grâce à des pointeurs.

2ème tableau : codification des J_i ; à chaque noeud correspond un élément d'un deuxième tableau dont la valeur est le premier pointeur pour ce noeud dans le premier tableau ; ainsi on réalise une codification dont l'encombrement est minimal.

Exemple : Reprenons l'exemple de la fig. 8 ; nous avons alors :

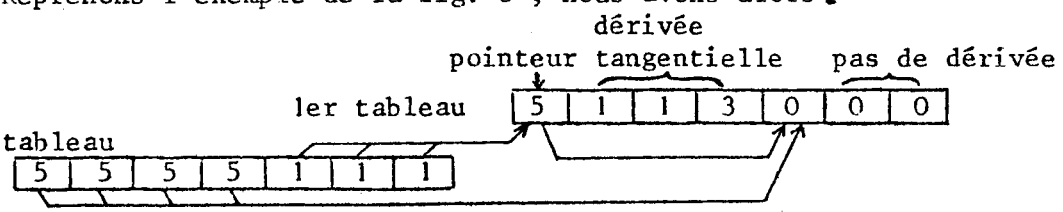


Fig. 9 : Courbes de niveau pour les polynomes p_{11} et p_{52} du même exemple.

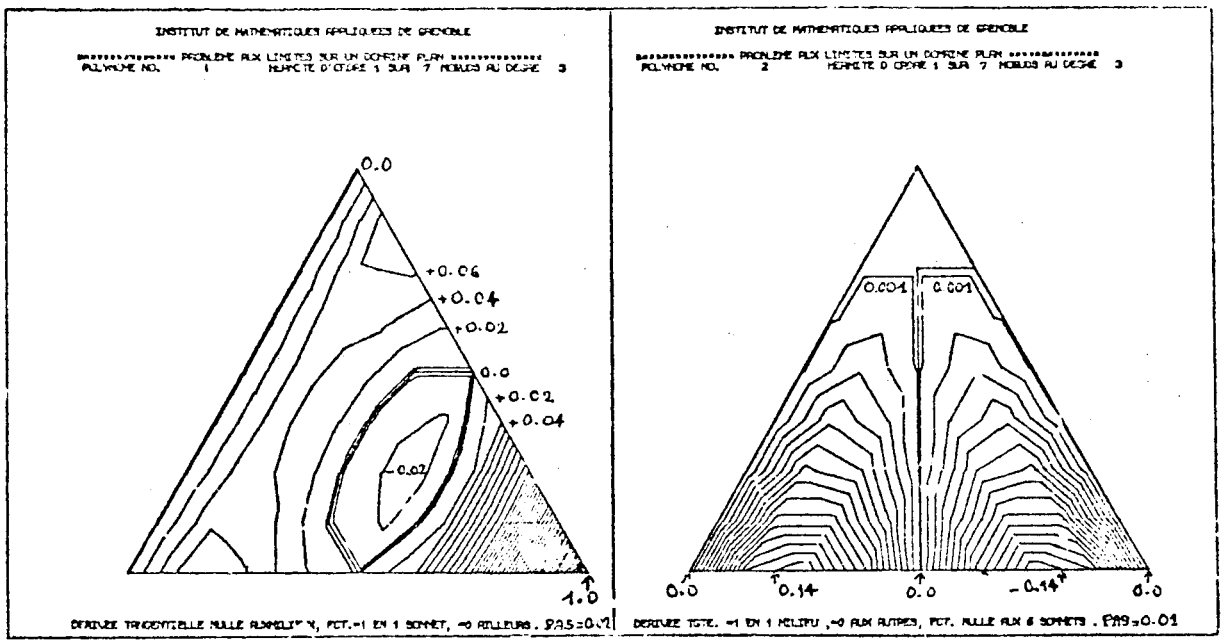
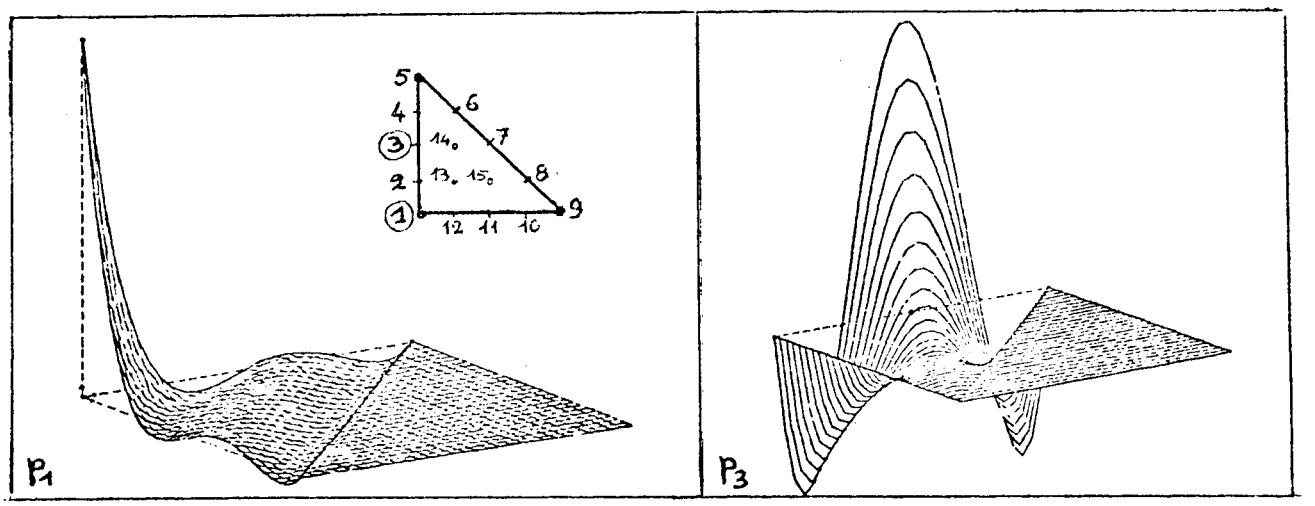


Fig. 10 : Elément de Lagrange de degré 4, polynomes p_1 et p_3



- IV - LA COMMANDE "SOLUTION"

La solution d'un problème approché par la méthode des éléments finis qui a été codifiée grâce aux trois commandes précédentes est celle d'un système linéaire dont la commande SOLUTION effectue l'élaboration et la résolution ; celles-ci peuvent être reprises (lors de la même commande), autant de fois qu'on le désire, avec différentes valeurs pour les paramètres des équations ou avec des frontières modifiées.

A cette fin, tous les résultats partiels nécessaires à cette reprise sont sauvegardés, de nombreux aiguillages sont possibles et enfin une visualisation variée des résultats a été mise au point :

- sur l'écran cathodique : courbes de niveau, valeurs ponctuelles, directions principales des contraintes en mécanique, (représentation en perspective à l'étude), avec reproduction possible sur table traçante ;
- sur le papier ("mouchard") : valeurs ponctuelles des fonctions approchées et de leurs dérivées, tableau de toutes les valeurs recherchées aux noeuds de Ω_h.

-IV-1- La technique frontale

La construction et la triangulation du système linéaire s'opère en considérant successivement tous les triangles de Ω_h, dans l'ordre établi lors de la triangulation (cf. I-3).

En écrivant que la restriction à chaque triangle, de la fonctionnelle J(u) à minimiser, est minimale sur le sous-espace de dimension finie correspondant à l'interpolation sur ce triangle et à la prise en considération des conditions éventuelles aux bords, on obtient une matrice appelée "matrice-élément". La matrice du système linéaire complet, qui n'est jamais explicitée, serait obtenue en regroupant toutes les matrices-éléments par des sommations après changements d'indice.

Schématiquement, la technique frontale consiste à effectuer, pour chaque triangle successivement considéré, la triangulation sur un maximum d'indices, c'est à dire sur tous les noeuds que l'on rencontre pour la dernière fois dans ce triangle. On ne travaille donc jamais sur toute la matrice mais seulement sur les lignes et colonnes correspondant aux noeuds du front, c'est pourquoi il est nécessaire de faire des changements d'adresse pour passer des numéros de noeuds à leurs numéros sur le front, et vice-versa.
Cette technique est donc bien adaptée aux grands systèmes linéaires ou aux ordinateurs à petite mémoire centrale.

On trouvera dans Irons |6| et Aussens |1| la description de programmes mettant en oeuvre cette technique (1).

-IV-2- Un exemple de traitement itératif : problème de frontière libre pour l'écoulement en milieu poreux

Soit à calculer l'écoulement de l'eau dans une digue poreuse.

Soit H le potentiel du liquide ; l'équation de potentiel est $\Delta H = 0$ avec H fixé de part et d'autre de la digue. A l'intérieur, la surface du liquide infiltré est inconnue et vérifie l'équation H = hauteur du liquide.

Supposons la digue suffisamment longue pour que l'on puisse se ramener à un problème dans $\mathbb{R}^2$ en prenant une section plane.

L'équipe du Pr Magenes |7| a étudié une nouvelle méthode, justifiée théoriquement, et qui permet de se ramener à un domaine connu.

La méthode traditionnelle consite à rechercher itérativement la frontière libre de la manière suivante :

Partant d'une frontière initiale, on résoud $\Delta H = 0$ pour obtenir sur celle-ci des valeurs H(x,y) qui en général, ne satisferont pas l'équation : H(x,y) = y.

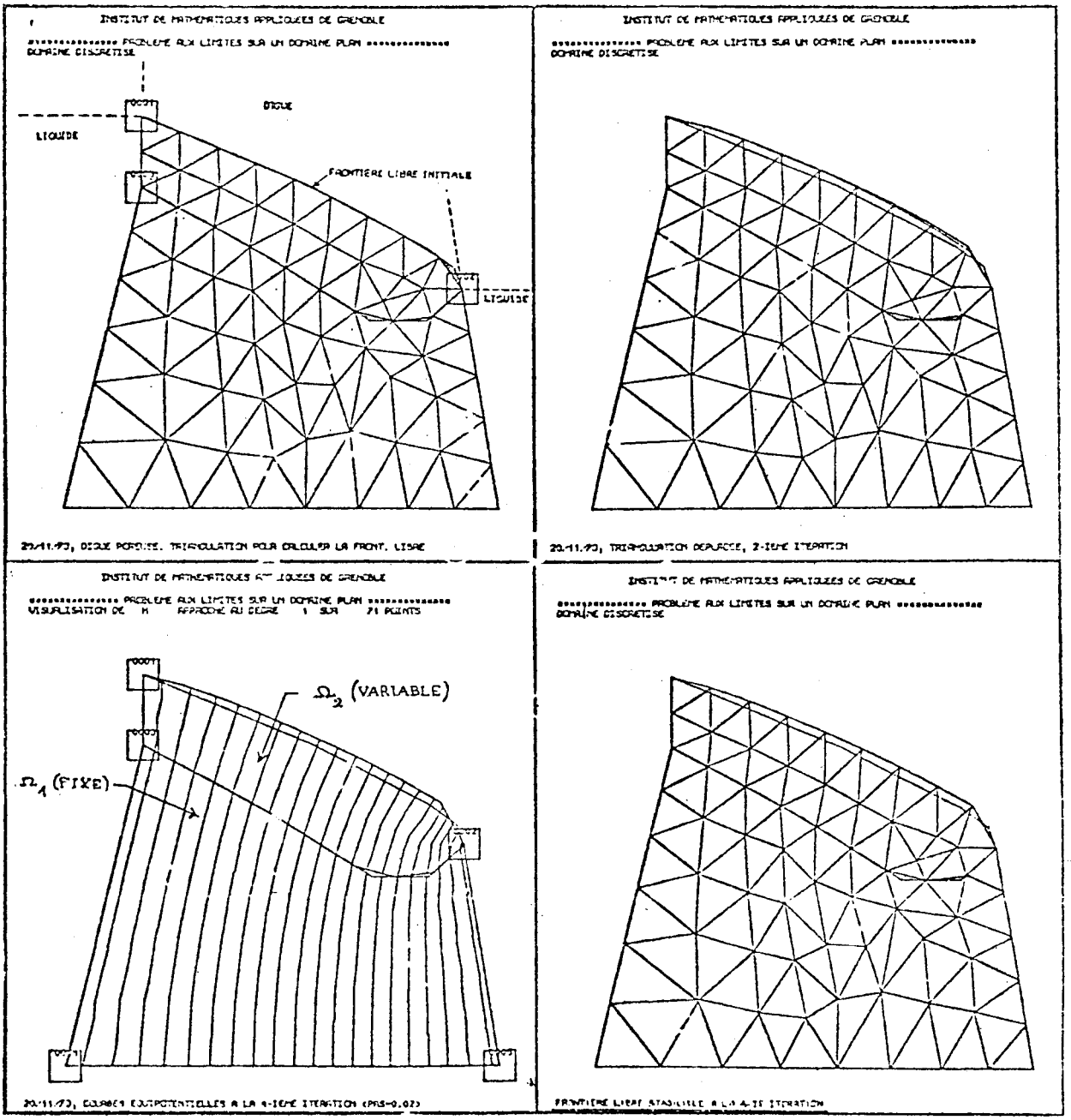
Les itérations consistent à effectuer sur la frontière variable des déplacements verticaux d'amplitude : H(x,y) - y jusqu'à ce qu'ils soient négligeables.

Dans le contexte de notre système, ces itérations sont faciles et peu coûteuses car, en découpant Ω en deux sous-domaines Ω_1 et Ω_2 où Ω_1 est fixe et où Ω_2 contient la frontière variable, le système linéaire peut être partiellement sauvegardé et, de plus, la frontière variable est déplacée automatiquement grâce à un court sous-programme spécifique à ce problème.

---

(1) Le programme utilisé dans notre système est dû à M. A. Aussems, à qui nous renouvelons, à cette occasion, l'expression de notre profonde gratitude.

Fig. 11 - Reproduction de 4 images parmi celles obtenues sur l'écran cathodique lors du traitement itératif du problème ci-dessus.



Lorsque ces itérations convergent (ce que l'on constate dans la pratique lorsque la frontière initiale est choisie "raisonnablement"), la frontière obtenue est bien la frontière libre cherchée ; elle est obtenue avec plus de précision que par la méthode directe citée plus haut.
L'implémentation dans notre système de résolutions itératives des systèmes linéaires nous permettra une comparaison plus complète des performances de ces deux méthodes.

IV-3- Un problème d'évolution à frontière variable

Considèrons une équation de type dhaleur :

(I) $a\frac{\partial c}{\partial t} - \Delta c = f$ sur un ouvert borné Ω

(a et f peuvent dépendre de x,y et t)

avec les conditions :

$$\left| \begin{array}{l} c = 0 \text{ sur la frontière } \Gamma \text{ de } \Omega \\ c = 0 \text{ pour } t = 0. \end{array} \right.$$

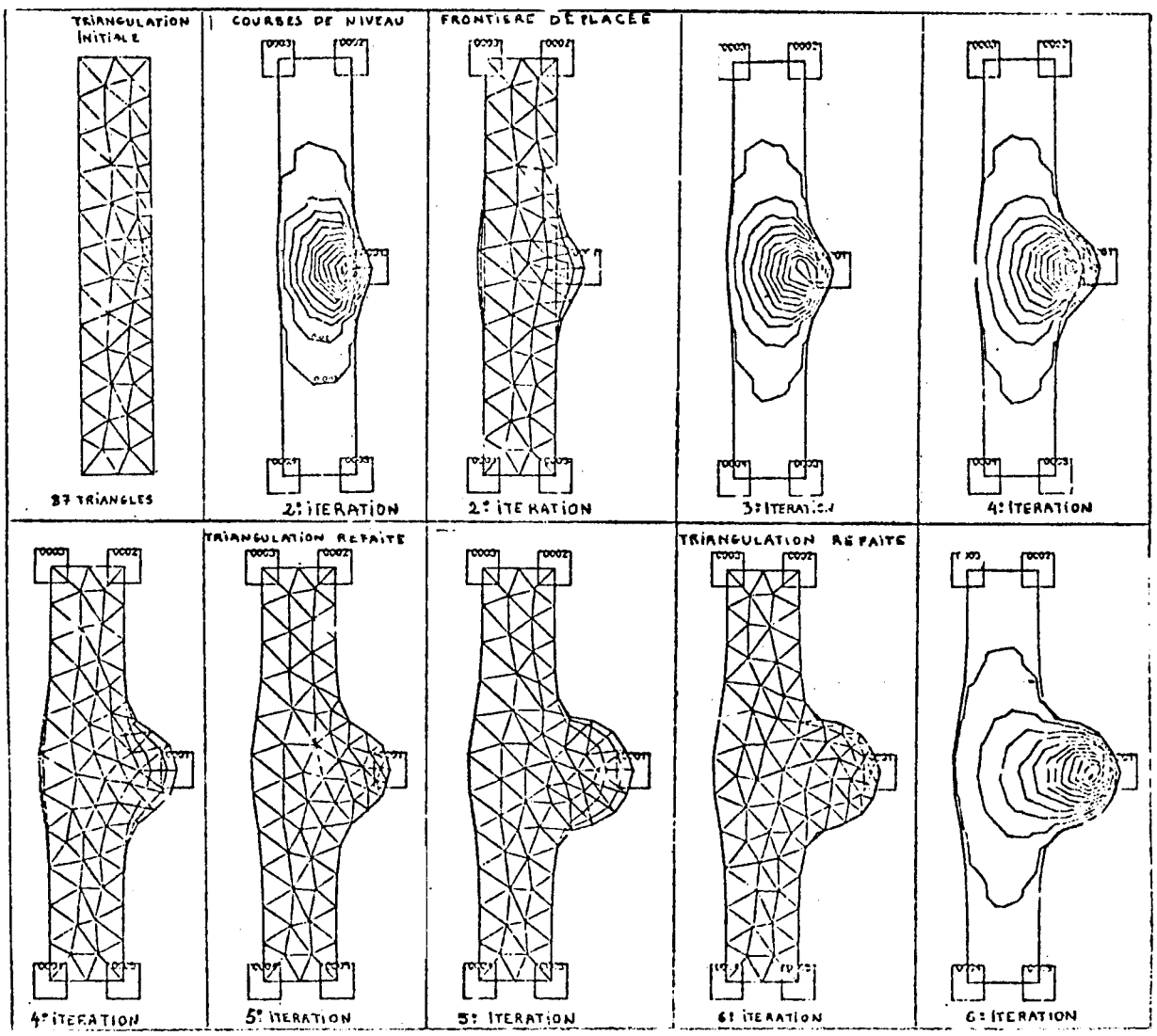
Supposons que la frontière se déforme au cours du temps en fonction de c (par exemple proportionnellement à la dérivée normale extérieure à Γ : $\frac{\partial c}{\partial \nu}$.

Discrétisons le problème en temps par un schéma décentré implicite :

(I) => (II) : $\frac{a}{\Delta t} c(t) - \Delta c(t) = f + \frac{a}{\Delta t} c(t - \Delta t)$

Nous donnons en fig. 12 quelques étapes du traitement par éléments finis de ce problème pour un choix convenable de plusieurs paramètres permettant de rendre compte de phénomènes observés en embryologie.

fig. 12



- V - LES TEMPS DE CALCUL

Le temps que l'utilisateur d'un tel système peut économiser en n'ayant aucune programmation, ni préparation très minutieuse de données à effectuer pour résoudre complétement son problème devrait largement compenser les temps de calcul qui paraîtront longs à ceux qui disposent de programmes très spécifiques et donc très performants dans leur spécialité.

Le temps moyen de calcul par inconnue discrète est essentiellement fonction du degré des polynômes d'interpolation, de la largeur maximum des fronts et des visualisations demandées.

Donnons deux chiffres extrêmes, pour une même largeur de front : (10 noeuds)

Pour un problème de potentiel, une interpolation linéaire, le temps moyen sans visualisation sera de 0,17 sec.

Pour un problème d'élasticité, des éléments cubiques de type Lagrange avec visualisation, il pourra dépasser 0,6 seconde.

- BIBLIOGRAPHIE -

|1| AUSSEMS : Une mise en oeuvre frontale des méthodes d'éléments finis. Thèse, Grenoble 1972.

|2| CHENIN : Quelques problèmes d'interpolation. Thèse, Grenoble, 1974.

|3| CIARLET,RAVIART : General Lagrange and Hermite Interpolation in R^n. Arch. Rat. Mech. Analysis.

|4| CIARLET,RAVIART : Maximum principle and uniforme convergence for the finite element method.

|5| GEORGE : Computer Implementation of finite element method. Thèse, Stanford, 1971.

|6| IRONS : A frontal solution program for finite element analysis. Int. J. Num. Meth. Engineering, Vol. 2, 5-32, 1970.

|7| MAGENES, BAIOCCHI, COMINCIOLI, GUERRI, POZZI, VOLPI : Free boundary Problems in the theory of fluid flow through prous media. CNR Pavie (1973).

|8| NICOLAIDES : On a class of finite elements generated by Lagrange interpolation. SIAM. J. Num. Analysis, Vol. 9, n° 3, Sept. 1972.

|9| NORRIE & DE VRIES : The finite element method. Academic Press (1973).

|10| RAVIART : Methode des éléments finis. Cours D.E.A., 1971-1972, Université Paris 6.

|11| STRANG & FIX : Analysis of the finite element method. Prentice Hall.

|12| ŽENIŠEK : Interpolation polynomials on the triangle. Num. Math. 15, 283-296, 1970.

|13| ZIENKIEWICZ : La méthode des éléments finis. Ed. Science, 1973.

|14| ZLAMAL : On some finite element procedures for solving second order boundary value problems. Num. Math., 14-42-48, 1969.

Alain PONCET
Université Scientifique et Médicale
Tour de Mathématiques
38041 - GRENOBLE-CEDEX