

**PUBLICATIONS**

**MATHEMATIQUES**

**D'ORSAY**

N° 81.07

Résolution des équations

de Navier Stokes

dans le cas stationnaire

P. GUILLAUME

F. CREPEL - D. SELIGMANN

Université de Paris-Sud  
Département de Mathématique

Bât. 425

91405 **ORSAY** France

Code matière : Equations aux dérivées partielles 35  
Analyse Numérique 65  
Equations de Navier-Stokes 35.Q10  
(Equations et Problèmes spéciaux)  
Eléments finis 65.N30

Mots clés : Eléments finis  
Equations de Navier-Stokes

**PUBLICATIONS**

**MATHEMATIQUES**

**D'ORSAY**

N° 81.07

Résolution des équations

de Navier Stokes

dans le cas stationnaire

35.858

P. GUILLAUME

F. CREPEL - D. SELIGMANN



Université de Paris-Sud  
Département de Mathématique

Bât. 425

91405 ORSAY France

## Table des matières

Introduction

Partie I : Théorie et approximation du problème.

- hypothèses de travail et mise en équation
- formulation variationnelle mixte du problème
- problème perturbé
- méthode d'éléments finis

Partie II : Structure informatique.

- Structure des principaux programmes établis

Partie III : Résultats numériques et commentaires.

Bibliographie

Appendice : Documentation du programme par A. LICHNEWSKY.

Cet ouvrage constitue la poursuite du travail  
de Monsieur Guillaume décédé accidentellement pendant  
la préparation de sa thèse.

O. Introduction :

Nous nous proposons d'étudier le mouvement d'un fluide contenu dans une cavité carrée. C'est le problème : "WALL-DRIVEN SQUARE".

Nous nous donnons :

- la géométrie de la cavité, sous forme d'un maillage représentant la section plane
- les constantes physiques du fluide
  - masse volumique
  - viscositéou le Reynolds du fluide
- la vitesse initiale du fluide
- les conditions aux limites (aux bords de la cavité)
- les forces volumiques, éventuellement fonctions du temps.

I - Hypothèses de travail et mise en équation :

L'objet de cette étude est le traitement numérique des équations de Navier-Stokes dans le cas stationnaire, par une méthode de pénalisation.

Nous avons employé des éléments finis mixtes, et seul le cas de la dimension 2 en espace a été envisagé.

Le modèle proposé est établi pour une température donnée uniforme et constante dans le temps.

Equations du mouvement :

On donne ici les équations d'un fluide incompressible visqueux à température constante et uniforme.

L'état du fluide est caractérisé par :

- le champ des vitesses  $\vec{u}$
- le champ des pressions hydrostatiques  $p$

a/ Equations de conservation :

MASSE :

L'équation de continuité dans le cas incompressible est :

$$(1) \quad u_{i,i} = 0 \quad \text{dans } \Omega \quad [\text{ceci en adoptant la convention d'Einstein}]$$

QUANTITE DE MOUVEMENT :

$$(2) \quad \sigma_{ij,j} + \rho f_i = \rho \dot{u}_i \quad \text{dans } \Omega$$

ceci pour  $i = 1,2$  et  $j = 1,2$

$\sigma_{ij}$  étant le tenseur des contraintes du fluide, de composantes  $\sigma_{ij}$  et  $\rho$  la masse volumique du fluide

$\dot{u}_i$  désignant la dérivée particulaire de  $u_i$ , et  $f$  le champ des forces extérieures.

b/ Loi de comportement :

$$(3) \underline{\underline{T}} = -p \underline{\underline{1}} + \mu_0 (\underline{\underline{\text{grad}}} u + \underline{\underline{\text{grad}}} u^T) \text{ dans } \Omega$$

$\mu_0$  étant le coefficient de viscosité.

Cette équation s'écrit pour chaque élément du tenseur :

$$T_{ij} = -p \delta_{ij} + \mu_0 (u_{i,j} + u_{j,i})$$

où  $\delta_{ij}$  est le symbole de Kronecker.

Nous faisons l'hypothèse de fluide visqueux, ce qui nous fait choisir comme conditions aux bords (aux limites) les valeurs de  $u_N$  et  $u_T$  sur la frontière  $\Gamma$  de  $\Omega$ .

En combinant les relations (1), (2) et (3) nous obtenons la forme traditionnelle des équations de Navier-Stokes (pour plus de détails on pourra se référer au livre de P. GERMAIN cf [3]).

$$(S_1) \left\{ \begin{array}{l} \rho_0 \frac{\partial u_i}{\partial t} - \mu_0 u_{i,jj} + p_{,i} + \rho_0 u_j u_{i,j} = \rho_0 f_i \\ u_{i,i} = 0 \text{ dans } \Omega \\ \left. \begin{array}{l} u_N = u_N^d \\ u_T = u_T^d \end{array} \right\} \text{ sur } \Gamma \end{array} \right. \quad \begin{array}{l} i = 1, 2 \text{ dans } \Omega \\ \text{où } u^d \text{ est une vitesse donnée} \\ \rho_0 = \rho(0, x_1, x_2) \text{ est une constante} \end{array}$$

Remarquons que la condition (1) qui est primordiale n'est pas toujours satisfaite dans les approximations utilisées.

Le système  $(S_1)$  peut être mis sous forme adimensionnelle en posant :

$$u = V_c \bar{u} ; x_i = L_c \bar{x}_i ; p = p_c \bar{p}$$



ce qui nous donne :

$$\rho_0 v_c \frac{\partial \bar{u}_i}{\partial t} - \mu_0 \frac{v_c}{L_c^2} \bar{u}_{i,jj} + \frac{p_c}{L_c} \bar{p}_{,i} + \rho_0 \frac{v_c^2}{L_c} \bar{u}_j \cdot \bar{u}_{i,j} = \rho_0 f_i$$

$$\Leftrightarrow \frac{\partial u_i}{\partial t} - \frac{\mu_0}{\rho_0 L_c^2} \bar{u}_{i,jj} + \frac{p_c}{\rho_0 v_c L_c} \bar{p}_{,i} + \frac{v_c}{L_c} \bar{u}_j \cdot \bar{u}_{i,j} = \frac{1}{v_c} f_i$$

$$\Leftrightarrow \frac{L_c}{v_c} \frac{\partial \bar{u}_i}{\partial t} - \frac{\mu_0}{\rho_0 L_c v_c} \bar{u}_{i,jj} + \frac{p_c}{\rho_0 v_c^2} \bar{p}_{,i} + \bar{u}_j \cdot \bar{u}_{i,j} = \frac{L_c}{v_c^2} f_i$$

Nous choisissons un temps caractéristique  $T_c$  tel que  $\frac{L_c}{T_c v_c} = 1$

et une pression caractéristique  $p_c$  telle que  $\frac{p_c}{\rho_0 v_c^2} = 1$

Nous obtenons alors :

$$\frac{\partial \bar{u}_i}{\partial t} - \frac{\mu_0}{\rho_0 L_c v_c} \bar{u}_{i,jj} + \bar{p}_{,i} + \bar{u}_j \cdot \bar{u}_{i,j} = \frac{L_c}{v_c^2} f_i = \bar{f}_i$$

Le nombre de Reynolds  $\mathcal{R}$  du fluide étant égal à  $\frac{\rho_0 L_c v_c}{\mu_0}$  nous obtenons finalement le système adimensionnel suivant :

$$(S^i) \left\{ \begin{array}{l} \frac{\partial u_i}{\partial t} - \frac{1}{\mathcal{R}} u_{i,jj} + p_{,i} + u_j \cdot u_{i,j} = f_i \quad \text{dans } \Omega \\ u_{i,i} = 0 \quad \text{dans } \Omega \\ \left. \begin{array}{l} u_N = u_N^d \\ u_T = u_T^d \end{array} \right\} \quad \text{sur } \Gamma \end{array} \right.$$

II - Formulation variationnelle mixte du problème :

Soit  $\Omega$  un ouvert borné de  $\mathbb{R}^2$  et soit  $\Gamma$  sa frontière, nous désirons calculer une solution approchée du problème sans dimension :

$$\left\{ \begin{array}{l} \text{Trouver } u \text{ et } p \text{ tels que} \\ \frac{\partial u}{\partial t} + u \cdot \nabla u - \frac{1}{\mathcal{R}} \nabla^2 u + \nabla p = f \quad \text{dans } \Omega \\ \operatorname{div} u = 0 \quad \text{dans } \Omega \\ u = \alpha \quad \text{avec } \alpha = \alpha(x, y, t) \quad \text{sur } \Gamma \\ u|_{t=0} = u_0 \quad \text{avec } u_0 = u_0(x, y, 0) \quad \text{dans } \Omega, \\ \text{où } \mathcal{R} \text{ est le Reynolds du fluide.} \end{array} \right.$$

Nous ne ferons l'étude que dans le cas stationnaire et avec des conditions limites homogènes, si celles-ci ne le sont pas on pourra se ramener au cas étudié par translation.

Avant d'introduire la formulation variationnelle mixte des équations de Navier-Stokes nous définissons :

- la forme bilinéaire sur  $H^1(\Omega)^4$  :

$$a(u, v) = \int_{\Omega} u_{j,i} v_{j,i} dx$$

- les formes trilinéaires sur  $H^1(\Omega)^6$  :

$$\left\{ \begin{array}{l} b(u, v, w) = \int_{\Omega} u_i v_{j,i} w_j dx \\ \tilde{b}(u, v, w) = \frac{1}{2} [b(u, v, w) - b(u, w, v)] \end{array} \right.$$

La forme trilinéaire  $u, v, w \rightarrow \tilde{b}(u, v, w)$  est continue sur  $H^1_0(\Omega)^6$ .

En effet :

D'après les théorèmes d'inclusion sur les Sobolev, nous avons  $(H^1_0(\Omega))^N \subset (L^4(\Omega))^N$  pour  $N \leq 3$

donc  $u, w \in (H^1_0(\Omega))^2 \Rightarrow u, w \in (L^4(\Omega))^2$

et  $v \in (H^1_0(\Omega))^2 \Rightarrow D_i v_j \in L^2(\Omega) \quad \forall i, j$  (où  $D_i v_j = v_{j,i}$ ).

$$\Rightarrow \left| \int_{\Omega} u_i v_{j,i} w_j dx \right| \ll \| u_i \|_{L^4(\Omega)} \| D_i v_j \|_{L^2(\Omega)} \| w_j \|_{L^4(\Omega)} < +\infty$$

et donc la forme trilinéaire  $u, v, w \rightarrow \tilde{b}(u, v, w)$  est continue sur  $H_0^1(\Omega)^6$ .

Remarque :

$$\bullet \tilde{b}(u, v, w) = b(u, v, w) \quad \forall v, w \in (H_0^1(\Omega))^2 \quad \text{et} \quad \forall u \in \{v \in (H_0^1(\Omega))^2 / \operatorname{div} v = 0\}$$

$$\bullet \tilde{b}(u, v, w) = 0 \quad \forall u, v \in (H_0^1(\Omega))^2$$

Notation :

$$V = \{v \in (H_0^1(\Omega))^2 / \operatorname{div} v = 0\}$$

Nous définissons alors la formulation variationnelle mixte des équations de Navier-Stokes, qui est :

$$(P) \left\{ \begin{array}{l} \text{Etant donné } f \in V', \text{ trouver } (u, p) \in (H_0^1(\Omega))^2 \times L^2(\Omega)/\mathbb{R} \text{ tels que} \\ \tilde{b}(u, u, v) - (p, \operatorname{div} v) + \frac{1}{\mathcal{R}} a(u, v) = (f, v) \\ (\operatorname{div} u, q) = 0 \end{array} \right. \quad \text{ceci } \forall (v, q) \in (H_0^1(\Omega))^2 \times L^2(\Omega)/\mathbb{R}$$

$$\text{Définissons } N = \sup \frac{|\tilde{b}(u, v, w)|}{\|u\|_{(H_0^1(\Omega))^2} \|v\|_{(H_0^1(\Omega))^2} \|w\|_{(H_0^1(\Omega))^2}} \quad u, v, w \in (H_0^1(\Omega))^2$$

et introduisons l'hypothèse suivante :

$$(H.1) \quad f \text{ est telle que } N \mathcal{R}^2 \|f\| \leq 1 - \delta \quad \delta > 0$$

Nous avons alors le théorème suivant ( cf LIONS [4] )

Pour tout  $f$  donné dans  $V'$ , vérifiant l'hypothèse (H.1) le problème (P) admet une solution unique  $(u, p)$ .

Remarque : L'unicité n'est possible que dans un espace  $\Omega$  de dimension  $n \leq 3$ . Elle est montrée grâce à la continuité de  $\tilde{b}$  et à (H.1).

### III - Problème perturbé:

Pour obtenir une solution approchée du problème sans dimension nous allons introduire une famille d'équations avec une petite perturbation.

Nous remplacerons la condition  $\operatorname{div} u = 0$  par le terme de pénalisation  $\frac{1}{\varepsilon} \nabla(\operatorname{div} u_\varepsilon)$ . Le problème perturbé est alors :

$$\left\{ \begin{array}{l} \text{Trouver } u_\varepsilon \text{ tel que :} \\ \frac{\partial u_\varepsilon}{\partial t} + u_\varepsilon \cdot \nabla u_\varepsilon - \frac{1}{\mathcal{K}} \nabla^2 u_\varepsilon + \frac{1}{2} (\operatorname{div} u_\varepsilon) \cdot u_\varepsilon \frac{1}{\varepsilon} \nabla(\operatorname{div} u_\varepsilon) = f \\ u_\varepsilon = \alpha \quad \text{sur } \Gamma \\ u|_{t=0} = u_0 \quad \text{dans } \Omega \end{array} \right.$$

Remarquons que le terme  $\frac{1}{2} (\operatorname{div} u_\varepsilon) \cdot u_\varepsilon$  est nécessaire pour obtenir un système de Cauchy - Kowaleska, la démonstration est due à R. TEMAM (cf [5]).

En utilisant les mêmes notations que celles du chapitre précédent nous obtenons la formulation variationnelle mixte du problème perturbé:

$$(P_\varepsilon) \left\{ \begin{array}{l} \text{Etant donné } f \in V' , \text{ trouver } (u^\varepsilon, p^\varepsilon) \in (H_0^1(\Omega))^2 \times L^2(\Omega) \text{ tels que} \\ \tilde{b}(u^\varepsilon, u^\varepsilon, v) - (p^\varepsilon, \operatorname{div} v) + \frac{1}{\mathcal{K}} a(u^\varepsilon, v) = (f, v) \\ + \varepsilon (p^\varepsilon, q) + (\operatorname{div} u^\varepsilon, q) = 0 \\ \text{ceci } \forall (v, q) \in (H_0^1(\Omega))^2 \times L^2(\Omega) \end{array} \right.$$

Et nous avons le théorème suivant :

Sous l'hypothèse (H.1) et pour  $f$  donné dans  $V'$ , le problème  $(P_\varepsilon)$  admet une solution unique  $(u_\varepsilon, p_\varepsilon)$ .

Nous avons de plus :

$$\|u - u_\varepsilon\|_V + \|p - p_\varepsilon\|_{L^2(\Omega)} \leq C\varepsilon$$

où  $(u, p)$  est la solution du problème  $(P)$ .

En utilisant la seconde équation du problème  $(P_\varepsilon)$  on peut éliminer le terme  $p^\varepsilon$  et nous obtenons :

(P'\_\varepsilon) Etant donné  $f \in V'$ , trouver  $u^\varepsilon \in (H_0^1(\Omega))^2$  tel que

$$\tilde{b}(u^\varepsilon, u^\varepsilon, v) + \frac{1}{\varepsilon} (\operatorname{div} u_\varepsilon, \operatorname{div} v) + \frac{1}{\mathcal{R}} a(u_\varepsilon, v) = (f, v)$$

ceci  $\forall v \in (H_0^1(\Omega))^2$

La pression  $p^\varepsilon$  pouvant être calculée à l'aide de l'équation :

$$p^\varepsilon = - \frac{1}{\varepsilon} \operatorname{div} u^\varepsilon$$



IV - Méthode des éléments finis :

1/ Formulation variationnelle mixte

Soit  $\mathcal{V} = (H_0^1(\Omega))^2$  et soit  $\mathcal{W} = L^2(\Omega)$ . La méthode des éléments finis consiste à rechercher la solution  $(u_\varepsilon, p_\varepsilon)$  du problème  $(P_\varepsilon)$  dans  $\mathcal{V}_h \times \mathcal{W}_h$  où  $\mathcal{V}_h$  est un sous espace de dimension finie de  $\mathcal{V}$   
 $\mathcal{W}_h$  est un sous espace de dimension finie de  $\mathcal{W}$

Nous définirons sur  $\mathcal{V} \times \mathcal{W}_h$  l'opérateur bilinéaire  $(\text{div}_h \cdot, \cdot)$

tel que :  $(\text{div}_h u, p_h) = (\text{div } u, p_h) \quad \forall u \in \mathcal{V} \text{ et } p_h \in \mathcal{W}_h$

Soit  $f_h = \text{Proj}_{\mathcal{W}_h} f$  nous devons alors résoudre le problème :

$$(P_{H_\varepsilon}) \left\{ \begin{array}{l} \text{Etant donné } f_h, \text{ trouver } (u_h^\varepsilon, p_h^\varepsilon) \in \mathcal{V}_h \times \mathcal{W}_h \text{ tels que} \\ \tilde{b}(u_h^\varepsilon, u_h^\varepsilon, v_h) - (p_h^\varepsilon, \text{div}_h v_h) + \frac{1}{\mathcal{R}} a(u_h^\varepsilon, v_h) = (f_h, v_h) \\ + \varepsilon (p_h^\varepsilon, q_h) + (\text{div}_h u_h^\varepsilon, q_h) = 0 \\ \forall (v_h, q_h) \in \mathcal{V}_h \times \mathcal{W}_h \end{array} \right.$$

Le problème  $(P_H)$  correspondant à  $(P_{H_\varepsilon})$  :

$$(P_H) \left\{ \begin{array}{l} \text{Etant donné } f_h, \text{ trouver } (u_h, p_h) \in \mathcal{V}_h \times \tilde{\mathcal{W}}_h \text{ tels que} \\ \tilde{b}(u_h, u_h, v_h) - (p_h, \text{div}_h v_h) + \frac{1}{\mathcal{R}} a(u_h, v_h) = (f_h, v_h) \\ (\text{div}_h u_h, q_h) = 0 \\ \forall (v_h, q_h) \in \mathcal{V}_h \times \tilde{\mathcal{W}}_h \text{ avec } \tilde{\mathcal{W}}_h \subset \mathcal{W}/\mathcal{R} \end{array} \right.$$

Définissons  $N_h = \text{Sup} \frac{|\tilde{b}(u_h, v_h, w_h)|}{\|u_h\|_{\mathcal{V}_h} \|v_h\|_{\mathcal{V}_h} \|w_h\|_{\mathcal{W}_h}} \quad u_h, v_h, w_h \in \mathcal{V}_h$

Faisons les hypothèses suivantes :

$$\left. \begin{array}{l}
 \text{i)} \\
 \text{(H.1')} \quad N_h \mathcal{R}^2 \quad \|f_h\|_{V_h} \leq 1 - \delta \quad \delta > 0 \\
 \text{ii) l'hypothèse fondamentale (cf BREZZI [8])} \\
 \text{(H.2)} \quad \exists_k > 0 \quad \sup_{u_h \in \mathcal{V}_h} \frac{(\operatorname{div}_h u_h, p_h)}{\|u_h\|_{V_h}} \geq k \|p_h\|_{W_h} \quad \forall p_h \in W/P
 \end{array} \right\}$$

Nous obtenons alors le théorème :

Théorème : Sous les hypothèses (H.1), (H.1'), (H.2), si  $(u_h^\xi, p_h^\xi)$  est l'unique solution du problème  $(P_{H_\xi})$ , si  $(u, p)$  est l'unique solution du problème  $(P)$  et si  $(u_h, p_h)$  est l'unique solution du problème  $(P_H)$  alors :

$$\|u - u_h^\xi\|_{V_h} + \|p - p_h^\xi\|_{W_h} \leq C_1 \xi + \|u - u_h\|_{V_h} + \|p - p_h\|_{W_h}$$

(pour la démonstration de ce théorème, cf M. BERCOVIER, Thèse de Doctorat d'Etat, Rouen, [1]).

En éliminant finalement la pression dans  $(P_{H_\xi})$  nous obtenons le problème :

$$(P_{H_\xi}') \left\{ \begin{array}{l}
 \text{Etant donné } f_h, \text{ trouver } u_h^\xi \in \mathcal{V}_h \text{ tel que} \\
 \tilde{b}(u_h^\xi, u_h^\xi, v_h) + \frac{1}{\mathcal{R}} a(u_h^\xi, v_h) + \frac{1}{\xi} (\operatorname{div}_h u_h^\xi, \operatorname{div}_h v_h) = (f_h, v_h) \\
 \text{ceci } \forall v_h \in \mathcal{V}_h
 \end{array} \right.$$

2/ Méthode de résolution :

Le problème  $(P_{H_\xi}')$  est non linéaire. Pour le résoudre nous utiliserons l'algorithme suivant ; en faisant la convention d'omettre les indices  $h$  et  $\xi$  pour simplifier l'écriture :

Soit  $u^0$  la valeur initiale ( $u^0 \equiv 0$ ), soit  $u^n$  la solution à la  $n^{\text{ième}}$  itération, on calcule  $u^{n+1}$  solution de :

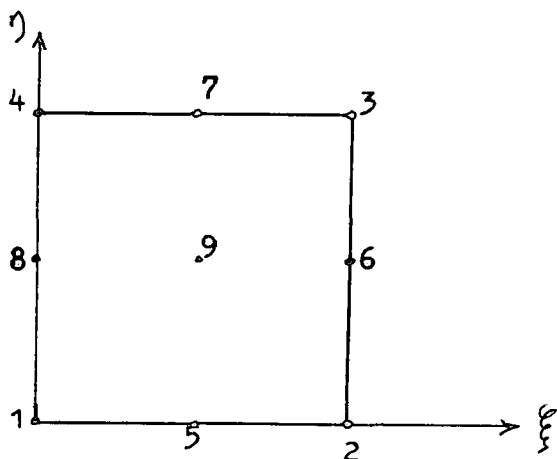
$$\frac{1}{\mathcal{K}} a(u^{n+1}, v_h) + \tilde{b}(u^n, u^{n+1}, v_h) + \frac{1}{\varepsilon} (\operatorname{div}_h u^{n+1}, \operatorname{div}_h v_h) = (f_h, v_h)$$

Le test d'arrêt de la méthode itérative portera sur  $|u^{n+1} - u^n|$

### 3/ Approximation par éléments finis :

Nous considérerons dans ce qui suit une approximation par des éléments finis de type  $Q_2$ .

Sur l'élément de référence, les fonctions de bases sont :



$$\left\{ \begin{aligned} \varphi_1(\xi, \eta) &= (\xi - \frac{1}{2})(\xi - 1)(\eta - \frac{1}{2})(\eta - 1)4 \\ \varphi_2(\xi, \eta) &= (\xi - \frac{1}{2})\xi(\eta - \frac{1}{2})(\eta - 1)4 \\ \varphi_3(\xi, \eta) &= (\xi - \frac{1}{2})\xi(\eta - \frac{1}{2})\eta 4 \\ \varphi_4(\xi, \eta) &= (\xi - \frac{1}{2})(\xi - 1)(\eta - \frac{1}{2})\eta 4 \\ \varphi_5(\xi, \eta) &= -\xi(\xi - 1)(\eta - \frac{1}{2})(\eta - 1)8 \\ \varphi_6(\xi, \eta) &= -\xi(\xi - \frac{1}{2})\eta(\eta - 1)8 \\ \varphi_7(\xi, \eta) &= -\xi(\xi - 1)(\eta - \frac{1}{2})\eta 8 \\ \varphi_8(\xi, \eta) &= -(\xi - \frac{1}{2})(\xi - 1)\eta(\eta - 1)8 \\ \varphi_9(\xi, \eta) &= (\xi - 1)\xi(\eta - 1)\eta 16 \end{aligned} \right.$$

Tout  $u_h \in U_h$  s'écrira alors sous forme de combinaison linéaire des fonctions de base  $\varphi_i$   $i=1, \dots, 9$

Les matrices A et B(u) associées à  $a(u^{n+1}, v_h)$  et  $\tilde{b}(u^n, u^{n+1}, v_h)$  seront calculées à l'aide d'une intégration numérique portant sur les  $3 \times 3$  points de Gauss.

Par contre la matrice associée au terme  $(\operatorname{div}_h u_h^{n+1}, \operatorname{div}_h v_h)$  sera calculée à l'aide d'une intégration numérique portant sur les  $2 \times 2$  points de Gauss, (cf 2, M.S ENGELMAN).



## I - Structure des principaux programmes :

D'une manière générale, les programmes débutent par une définition des plus complètes (du moins nous le souhaitons) des paramètres et quantités nécessaires aux calculs.

Une vérification des dimensions allouées est également faite d'une manière systématique dès qu'il y a création d'un tableau s'inscrivant dans notre zone de travail.

### 1/ Structure du programme principal :

- Initialisation par défaut des divers namelist, des unités de lecture et d'impression ainsi que du commun contenant les largeurs des différents tableaux.

- lecture du namelist "EXEC" définissant les principaux paramètres du travail du programme.

- appel des sous programmes O7QAD 1 et O7INQD en vue du remplissage des tableaux nécessaires à la partie géométrique de la méthode d'éléments finis.

- appel de O7PREP en vue du remplissage des tableaux donnant les fonctions de bases.

- appel de O7XSOL en vue d'initialiser le secteur "solution" en tenant compte des conditions limites.

- lecture du namelist "STRAT" définissant la stratégie employée dans la frontale.

- lecture du namelist "DISK" donnant les numéros des fichiers utilisés dans la frontale.

- appel de OOFRT1 définissant les tableaux nécessaires à la frontale.

- boucle pour la résolution itérative des différentes équations.

- initialisation du 2<sup>nd</sup> membre.

- appel de OOFRT2 effectuant la "descente" de la méthode frontale, ce sous programme appelle O7MTLM.

- Test éventuel de vérification de l'assemblage, entraîne la fin du programme (ie : renvoie au namelist EXEC).

- appel de OOFRT3 : remontée de la frontale.
- calcul des erreurs.
- appel de O8ERR qui dresse un tableau de l'erreur entre deux itérations.
- calcul de la composante horizontale de la vitesse le long de la médiane.
- calcul de la norme  $L^2$  de la divergence grâce aux sous programme O7DIV.

Tests d'arrêt :

- nombre maximum d'itérations (donné dans le namelist EXEC).
- différences entre deux itérations,
- différences relatives entre deux itérations, (le choix se faisant à l'aide du namelist EXEC).
- Impression éventuelle du vecteur solution XSOL.
- Tracé éventuel des champs des vitesses et de la composante horizontale de la vitesse le long de la médiane.

2/ Structure de O7INQ D : "Remplissage du tableau inquad décrivant la méthode d'éléments finis".

- remplissage intermédiaire de 'inquad', numérotation des noeuds de la façon suivante :

4	7	3
8	9	6
1	5	2

- impression du nombre de noeuds  
de noeud de base  
de fonctions de base.
- remplissage définitif de 'inquad' :  
les noeuds libres sont au début ;  
les noeuds du bord sont à la fin.

- 3/ Structure de O7PREP : "Définition de la méthode d'intégration numérique sur le quadrangle de référence".
- lecture du nombre de points d'intégrations numériques.
  - lecture des coordonnées, et du poids de chacun des points de Gauss.
  - calcul des fonctions de bases du  $Q^2$  et de leurs gradients:
    - avec 3 x 3 points de Gauss
    - avec 2 x 2 points de Gauss.
- 4/ Structure de O7XSOL : "Initialisation du tableau XSOL"
- On appelle les fonctions FU1 et FU2 qui donnent les composantes horizontale et verticale de la vitesse sur un bord, en un point.
- 5/ Structure de O7MFLM : "Création de la matrice élémentaire de l'élément inq", méthode de Bercovier.
- Détermination des noeuds du quadrangle, ainsi que de leurs composantes.
  - Calcul des quantités nécessaires au changement de base.
  - Initialisation des divers tableaux.
  - Boucle sur les éléments d'intégration numérique d'ordre supérieur
    - Détermination des coefficients du point de Gauss.
    - calcul du Jacobien au point de Gauss.
    - calcul des coordonnées des points de Gauss dans l'élément inq.
    - calcul de la dérivée du changement de variable : (elt qcque → elt de référence).
    - calcul de la dérivée du changement de variable inverse.
    - calcul des fonctions de base et de leurs dérivées dans le système réel aux points d'intégration numérique.
    - vecteur vitesse calculé à l'itération précédente en vue du remplissage du terme non linéaire.

- remplissage des tableaux élémentaires
  - i/ correspondant à la forme bilinéaire 'a'
  - ii/ correspondant au terme non linéaire 'b'
  - iii/ correspondant au second membre, avec calcul des fonctions F1 et F2.
- fin de la boucle sur les éléments d'intégration numérique d'ordre supérieur.
- boucle sur les éléments d'intégration numérique d'ordre inférieur: pour le remplissage du terme en divergence.

"l'approche est la même que précédemment".
- fin de la boucle sur les éléments d'intégration numérique d'ordre inférieur.
- remplissage de la matrice élémentaire.
- modification du second membre, dûe aux conditions aux bords.
- impressions éventuelles.

Méthode frontale :

L'approche utilisée nous a été suggérée par A. LICHNEWSKY [9].  
On pourra trouver d'autres approches (cf [10] → [12] ).

Structure de OOFRT1 : "Création des tableaux nécessaires  
à la méthode frontale".

- détermination du front de sortie d'un noeud.
- boucle sur les éléments d'un front
  - remplissage des tableaux
  - écriture sur fichier du descripteur et du nombre de degrés de liberté figés, restants, sortants.
- écriture de la taille nécessaire à la matrice : ie longueur maximale du front.

Structure de OOFRT2 : "Assemblage de la matrice et du second membre  
ainsi que 'descente de la méthode frontale".

- lecture de la taille de la matrice.
- boucle sur les éléments d'un front
  - appel du sous programme O7MTLM, qui calcule la matrice élémentaire et le second membre élémentaire en tenant compte des conditions limites.
  - remplissage de la matrice et du second membre.
  - lecture du descripteur.
  - méthode de Gauss sur la matrice et le second membre correspondant aux sortants.
  - écriture sur fichier de la matrice.
  - initialisation des lignes et colonnes de la matrice correspondant aux sortants.

Structure de OOFRT3 : "Remontée de la frontale"

- lecture de la taille de la matrice.
- boucle sur les éléments du front
  - lecture du descripteur
  - lecture de la matrice
  - remontée de la méthode de Gauss sur la matrice des noeuds sortants.

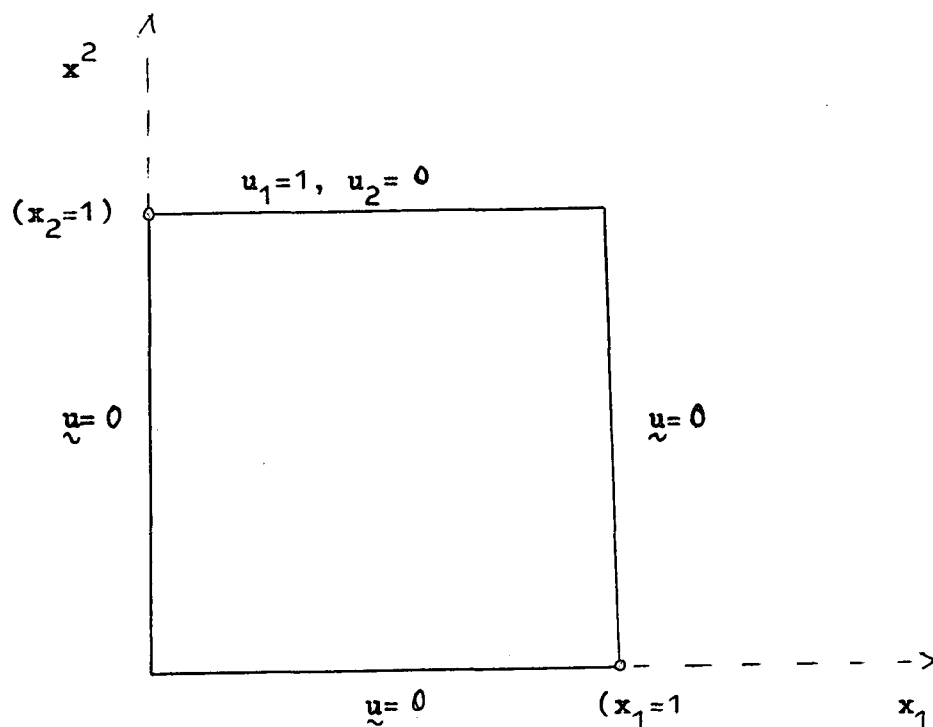
III - Résultats numériques et commentaires :

Présentation du cas traité : la cavité carrée

Le problème classique de la cavité carrée a été choisi pour traiter les équations de Navier-Stokes, pour sa relative simplicité.

De nombreux travaux lui sont consacrés, (cf, par exemple ENGELMAN [2], BENAZETH [6], THOMASSET [7]).

- La cavité est un carré de côté unité
- les conditions aux limites sont :



- Nous utiliserons une "triangulation" en  $N^2$  carrés égaux.

Evolution de la taille du problème en fonction du maillage :

Le maillage est toujours le plus simple et le plus régulier qui soit, le nombre des subdivisions est le même quel que soit l'axe,

nombre de subdivisions sur un axe	4	8	12	16
nombre de quadrangles	16	64	144	256
nombre de noeuds	81	289	625	1089
nombre de degrés de liberté	162	578	1250	2178
nombre de noeuds de base	49	225	529	961
nombre de fonctions de base <sup>(*)</sup>	98	450	1058	1922

(\*) en chacun des noeuds de base nous définissons deux fonctions de base :

- i) l'une correspondant à une intégration numérique en 3 x 3 points de GAUSS.
- ii) l'autre à l'intégration numérique en 2 x 2 points de GAUSS.

Evolution de la taille de la matrice issue de la méthode frontale en fonction du maillage et de la stratégie choisie :

nombre de subdivisions sur un axe	4	8	12	16
<u>stratégie quadrangle par quadrangle</u>				
nombre d'étapes	16	64	144	256
dimension de la matrice (nombre d'éléments)	30 x 30 (900)	46 x 46 (2116)	62 x 62 (3844)	78 x 78 (6084)
<u>stratégie : 1/2 ligne par 1/2 ligne</u>				
nombre d'étapes	8	16	24	32
dimension de la matrice (nombre d'éléments)	38 x 38 (1444)	70 x 70 (4900)	102x102 (10404)	134x134 (17956)
<u>stratégie : ligne par ligne</u>				
nombre d'étapes	4	8	12	16
dimension de la matrice (nombre d'éléments)	54 x 54 (2916)	102x102 (10404)	150x150 (22500)	198x198 (39204)
taille de la matrice "pleine" (nombre d'éléments)	162x162 (26244)	578x578 (334084)	1250x1250 (1562500)	2178x2178 (4743684)



Interprétation des différents tracés :

Deux maillages différents ont été traité :

i/ 16 carrés : ce maillage primaire ne donne pas des résultats très satisfaisants comme le montrent les différents tracés qui suivent.

ii/ 64 carrés : ce maillage nous a permis de constater

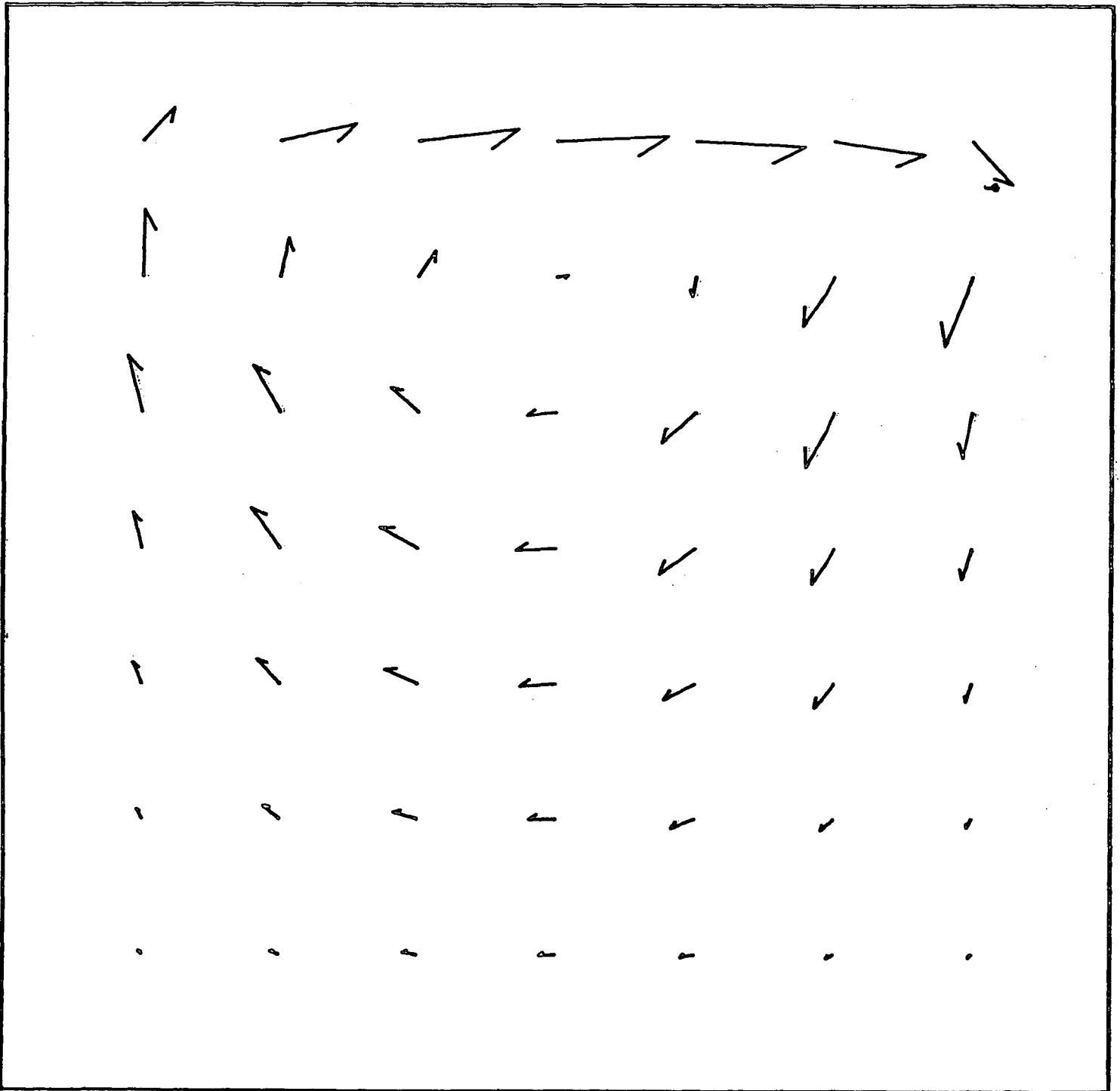
- le décentrement du tourbillon principal qui diminue quand le Reynolds augmente.

- l'apparition de tourbillons secondaires (contre-tourbillons) dans la partie inférieure droite de la cavité pour les deux Reynolds étudiés (400 et 1000),

et dans la partie inférieure gauche de la cavité pour Reynolds égal à 1000 .

D'autre part l'allure générale de la composante horizontale de la vitesse le long de la médiane est conforme aux résultats déjà établis, (cf par exemple : F. THOMASSET [7] ).

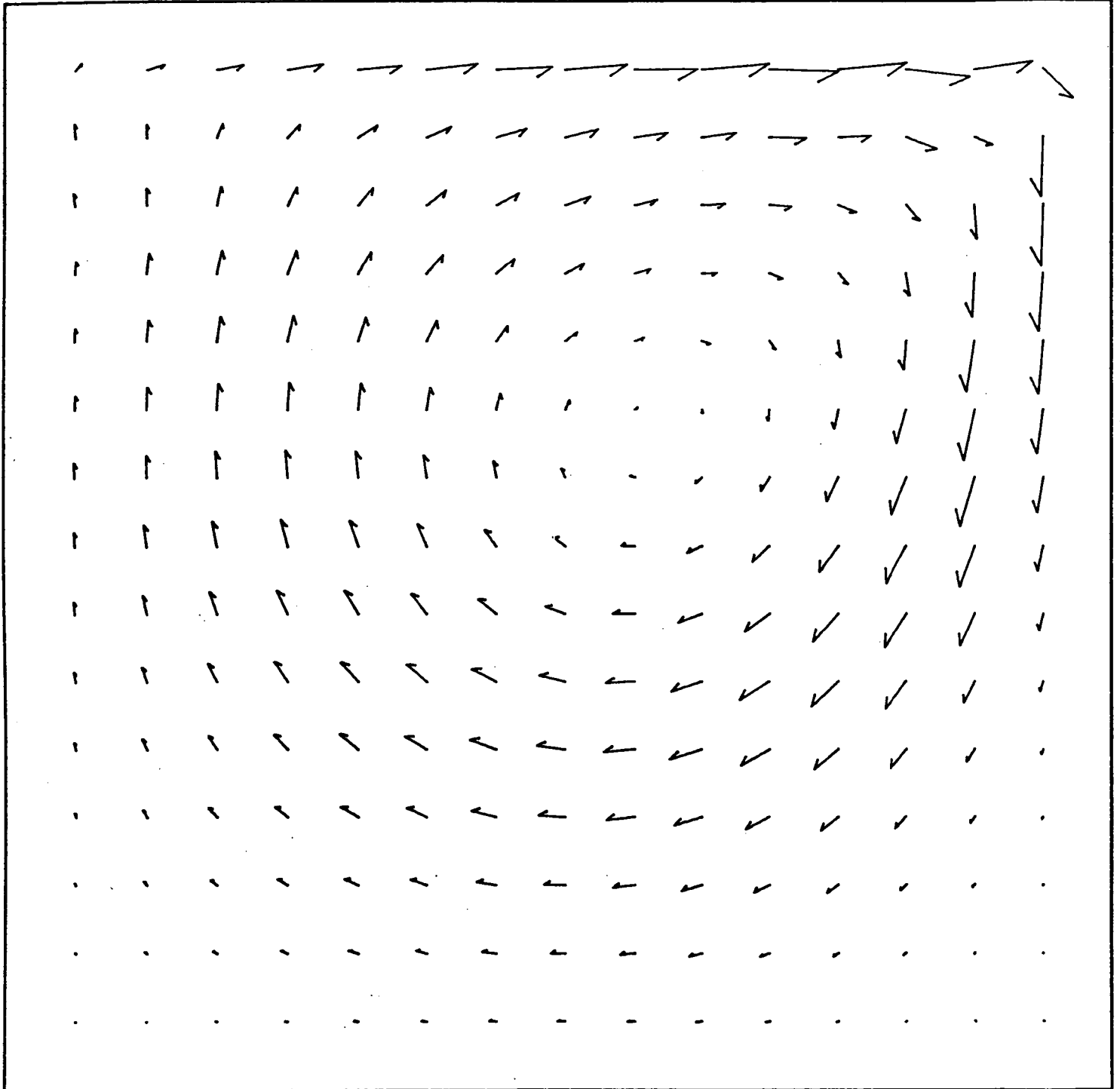
CHAMP DES VITESSES ,ECH=1



$NX, NY=4, REYNO=400, EPS=1E-5$

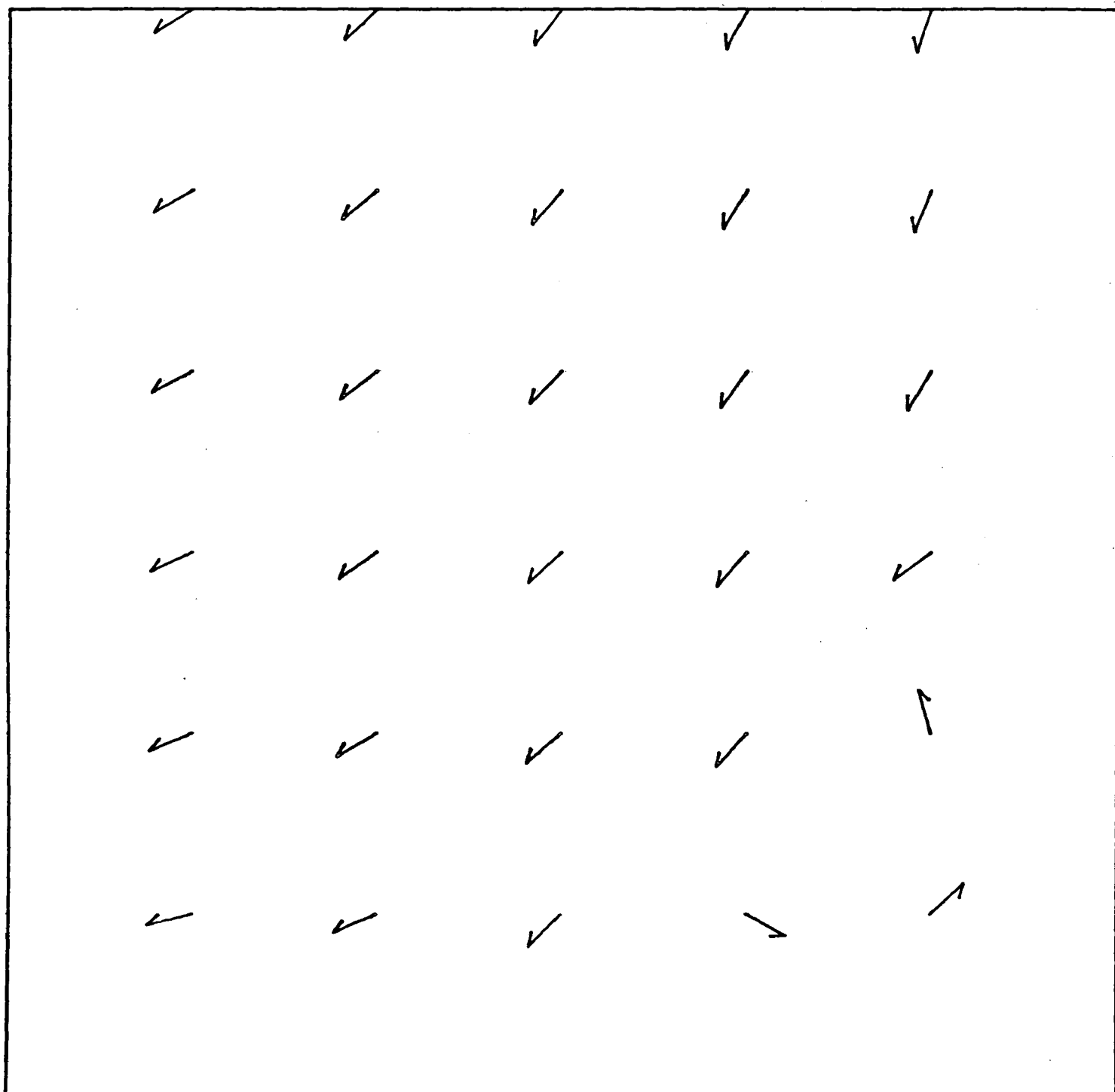
fig. 1

### CHAMP DES VITESSES ,ECH=1



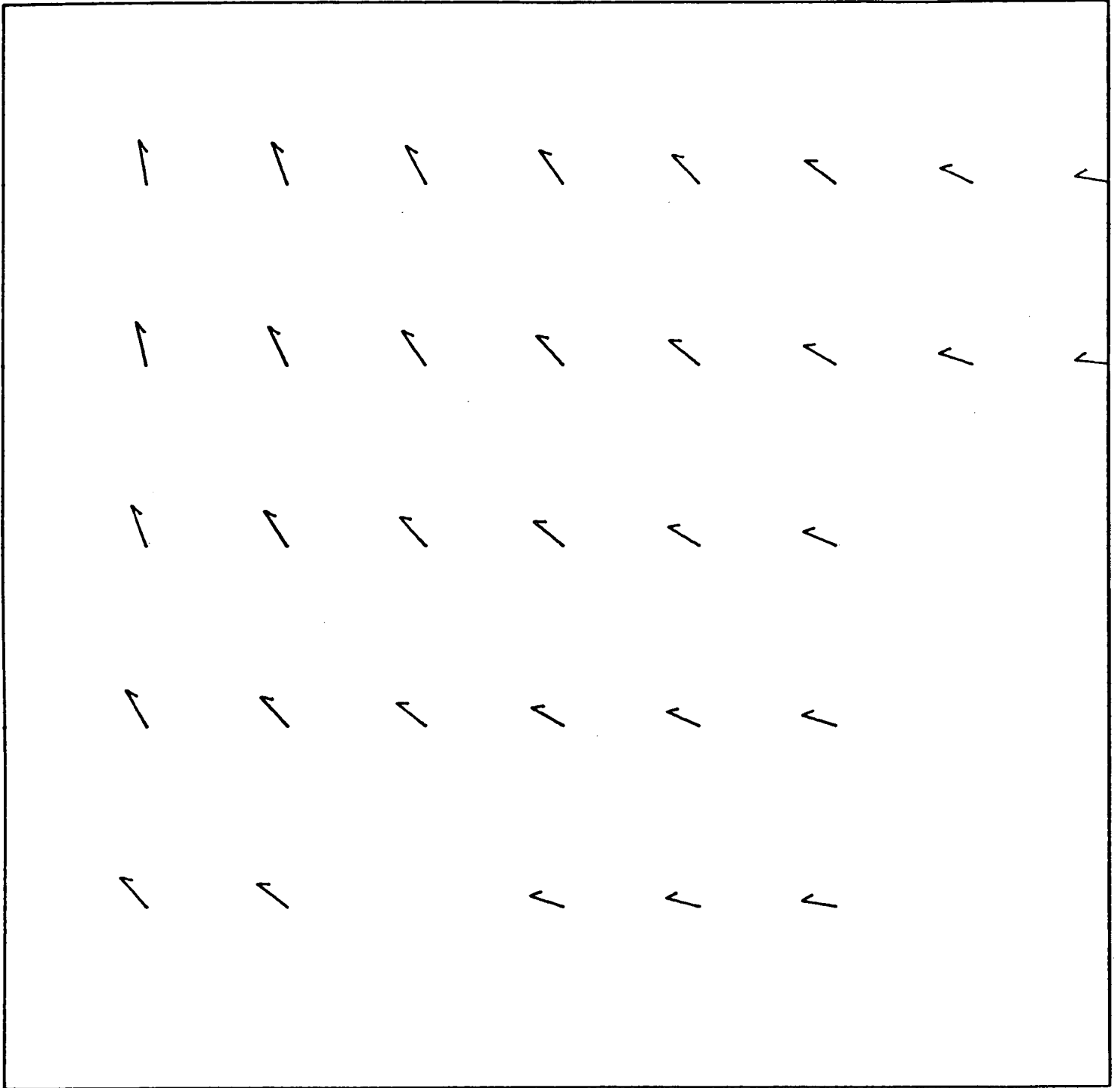
$NX, NY = 8, REYNO = 400, EPS = 1E-5$

# CHAMP DES VITESSES ,COIN BAS-DROITE



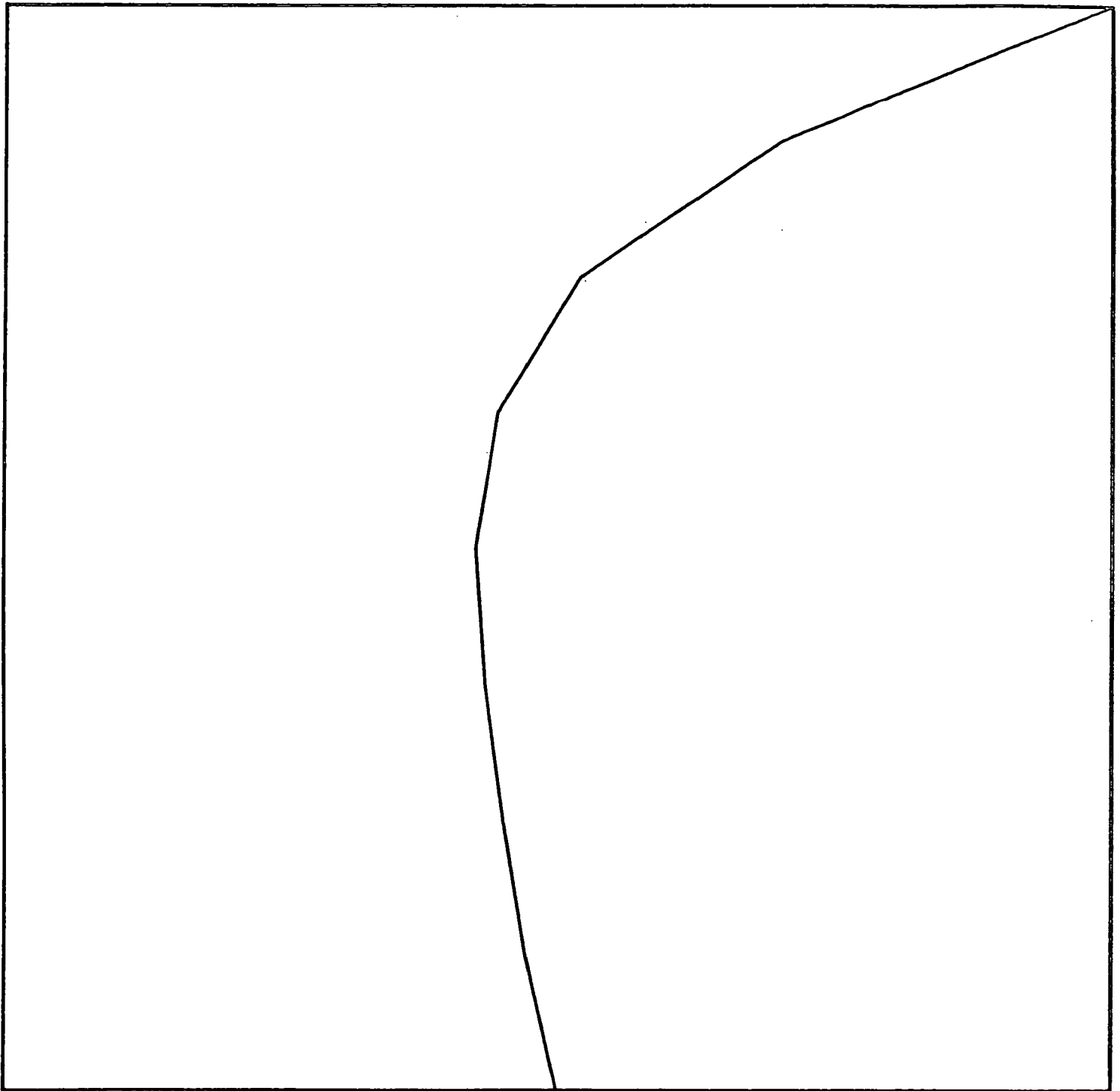
$NX, NY = 8, REYNO = 400, EPS = 1E-5$

## CHAMP DES VITESSES ,COIN BAS-GAUCHE



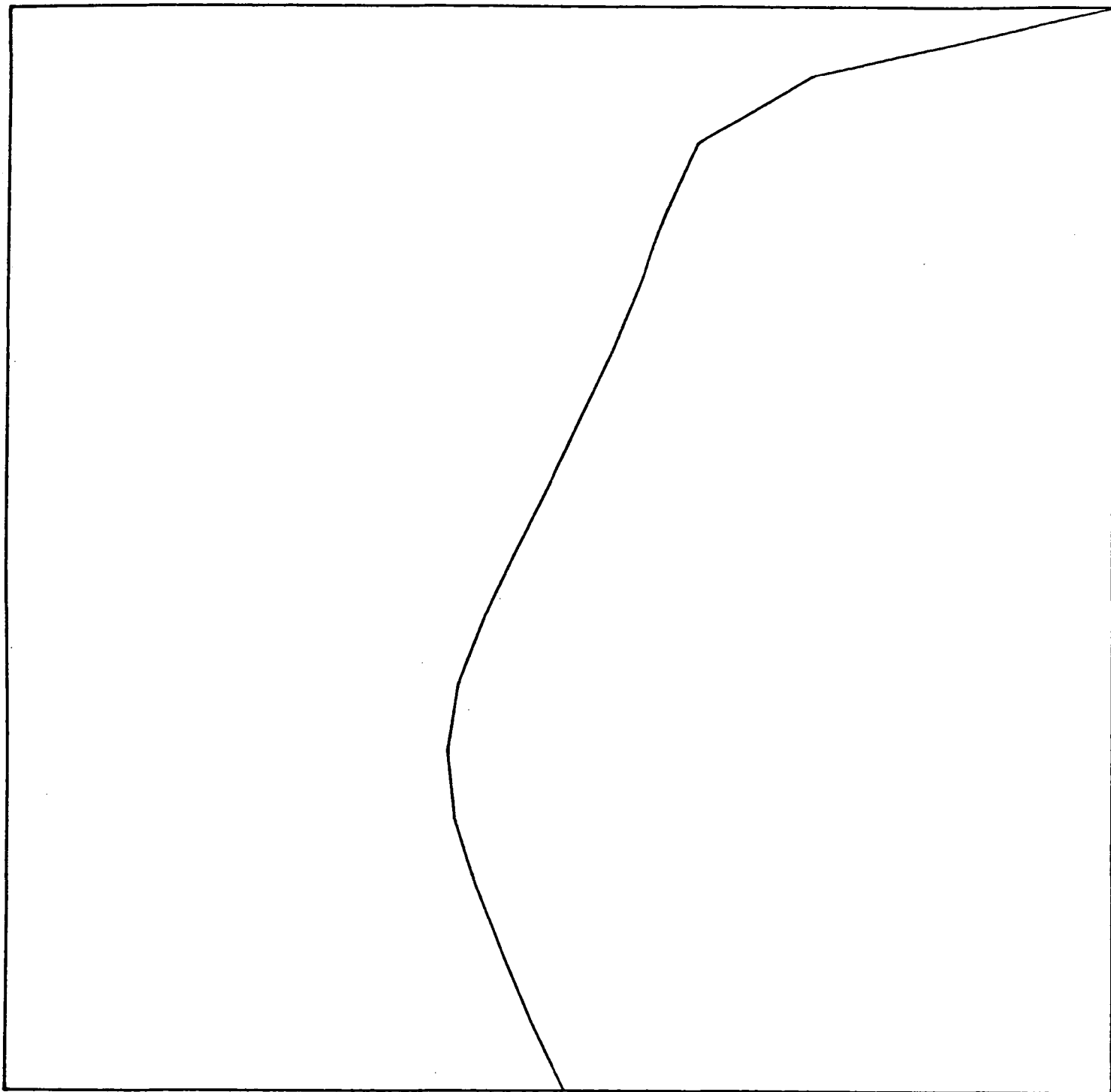
$NX, NY = 8, REYNO = 400, EPS = 1E-5$

COMPOSANTE HORIZONTALE DE LA VIT LE LONG DE LA MEDIANE



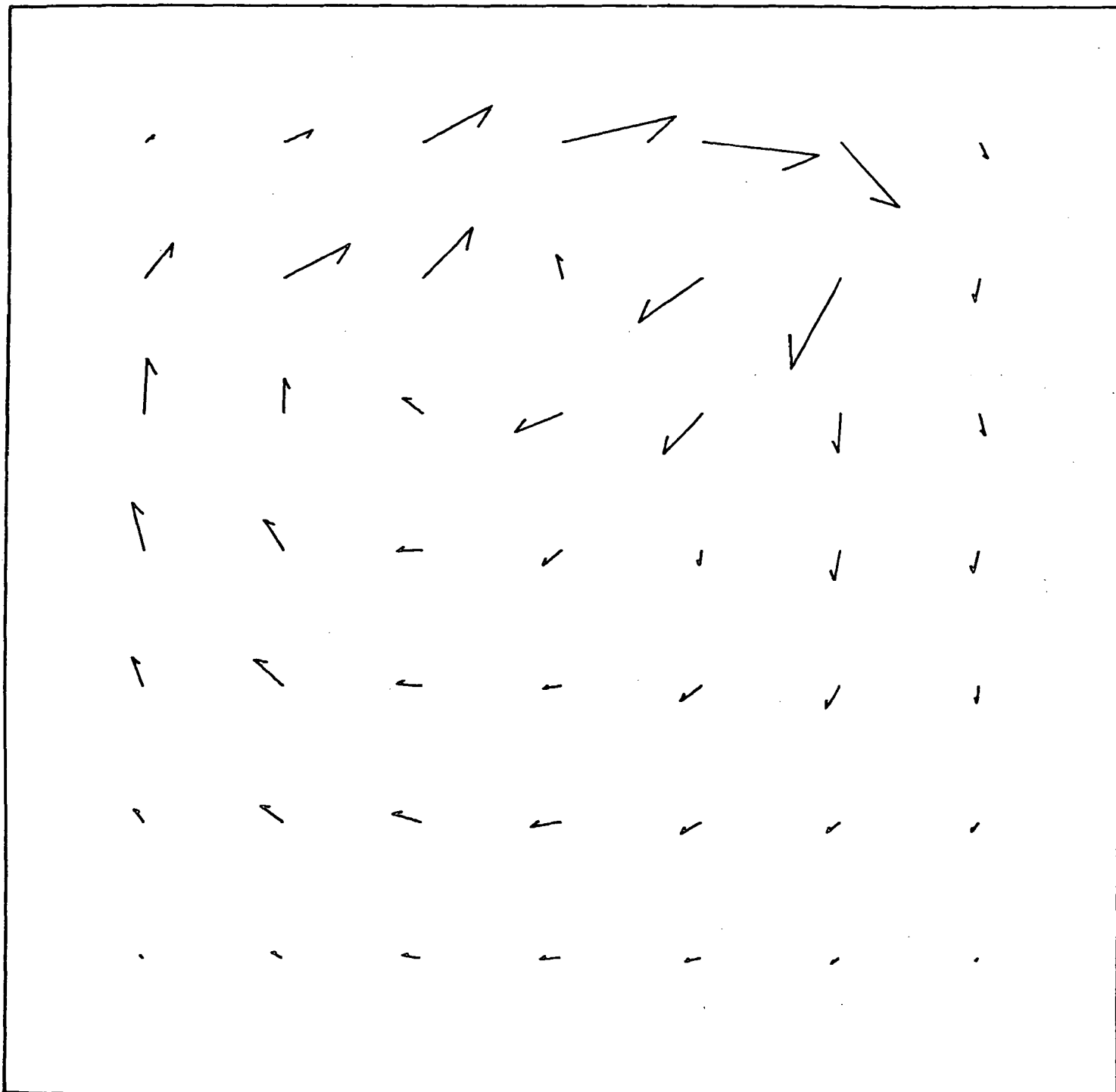
$NX, NY = 8, REYNO = 400, EPS = 1E-5$

COMPOSANTE HORIZONTALE DE LA VIT. LE LONG DE LA MEDIANE



$NX, NY = 8, REYN0 = 400, EPS = 1E - 5$

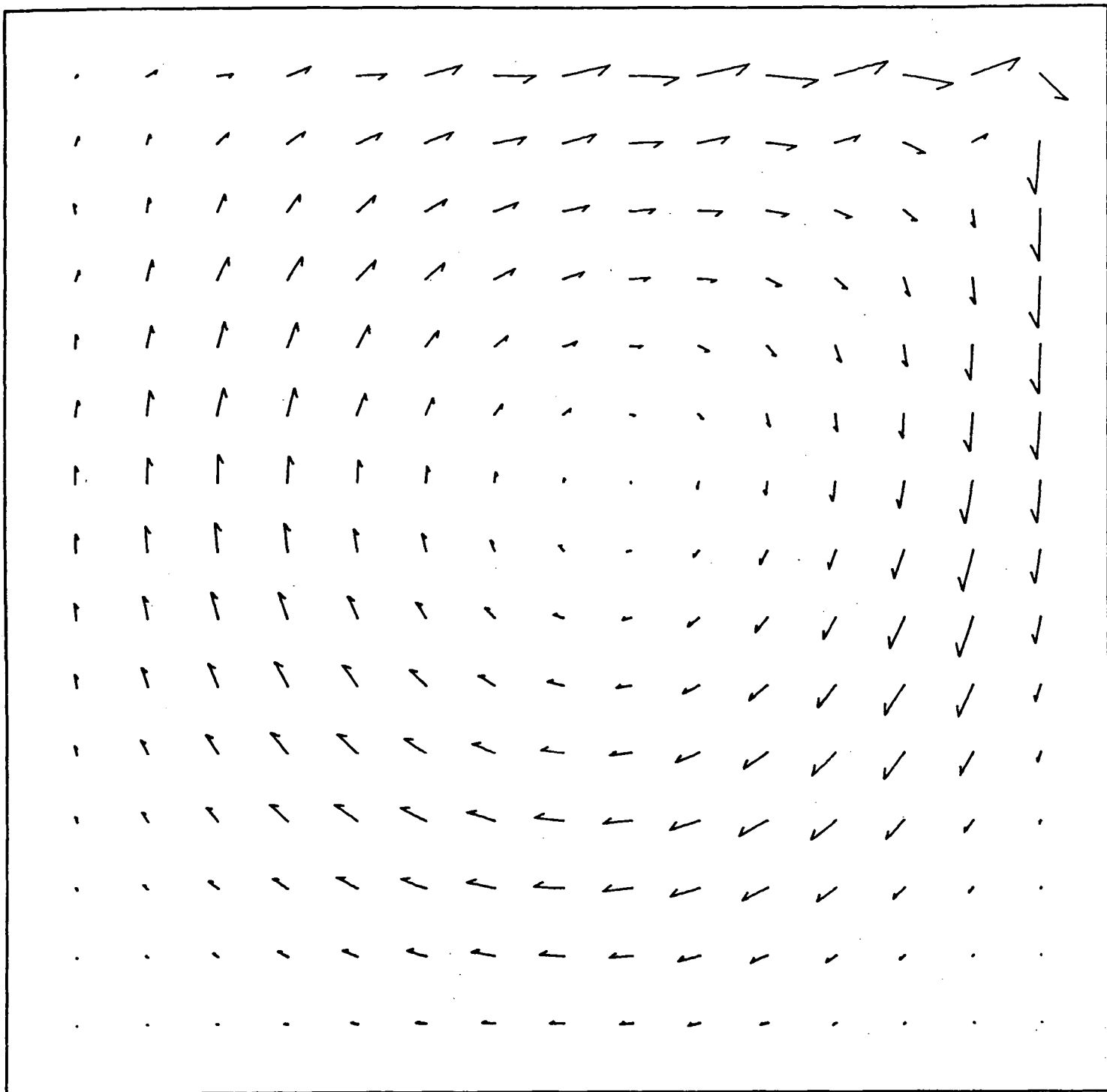
CHAMP DES VITESSES ,ECH=1



$NX, NY = 4, REYNO = 1000, EPS = 1E-5$

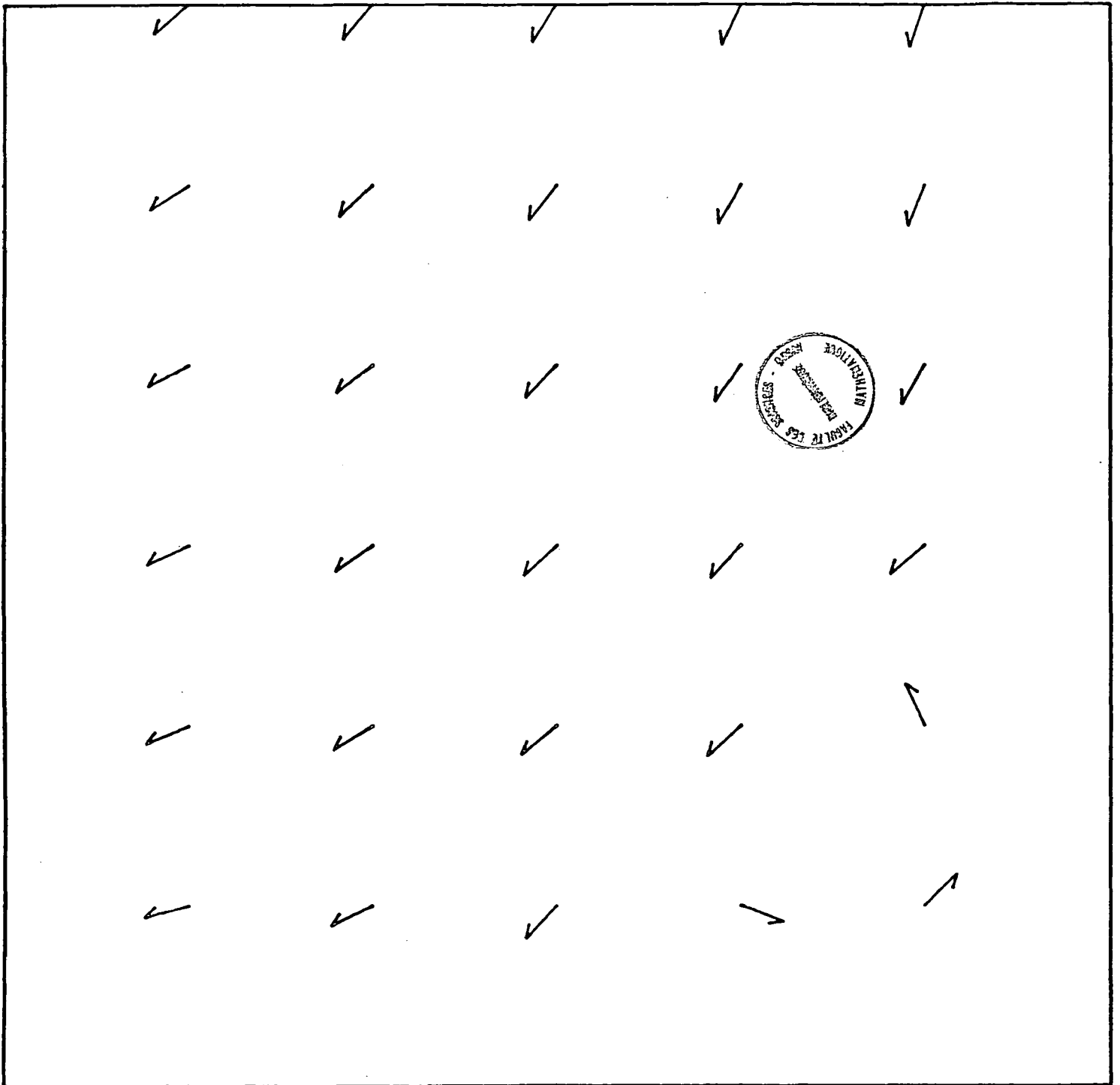


CHAMP DES VITESSES, ECH=1



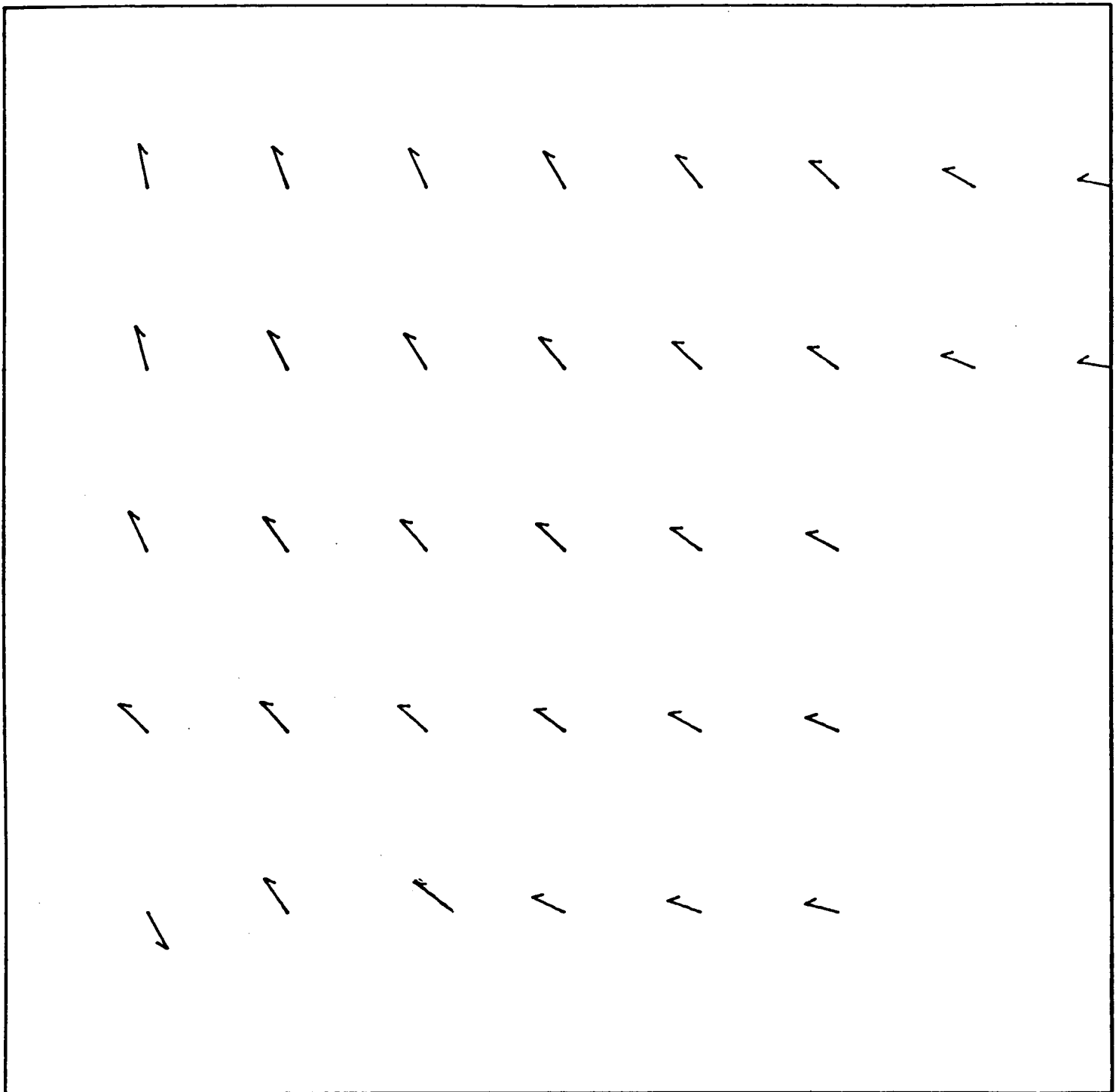
$NX, NY = 8, REYNO = 1000, EPS = 1E-5$

# CHAMP DES VITESSES ,COIN BAS-DROITE



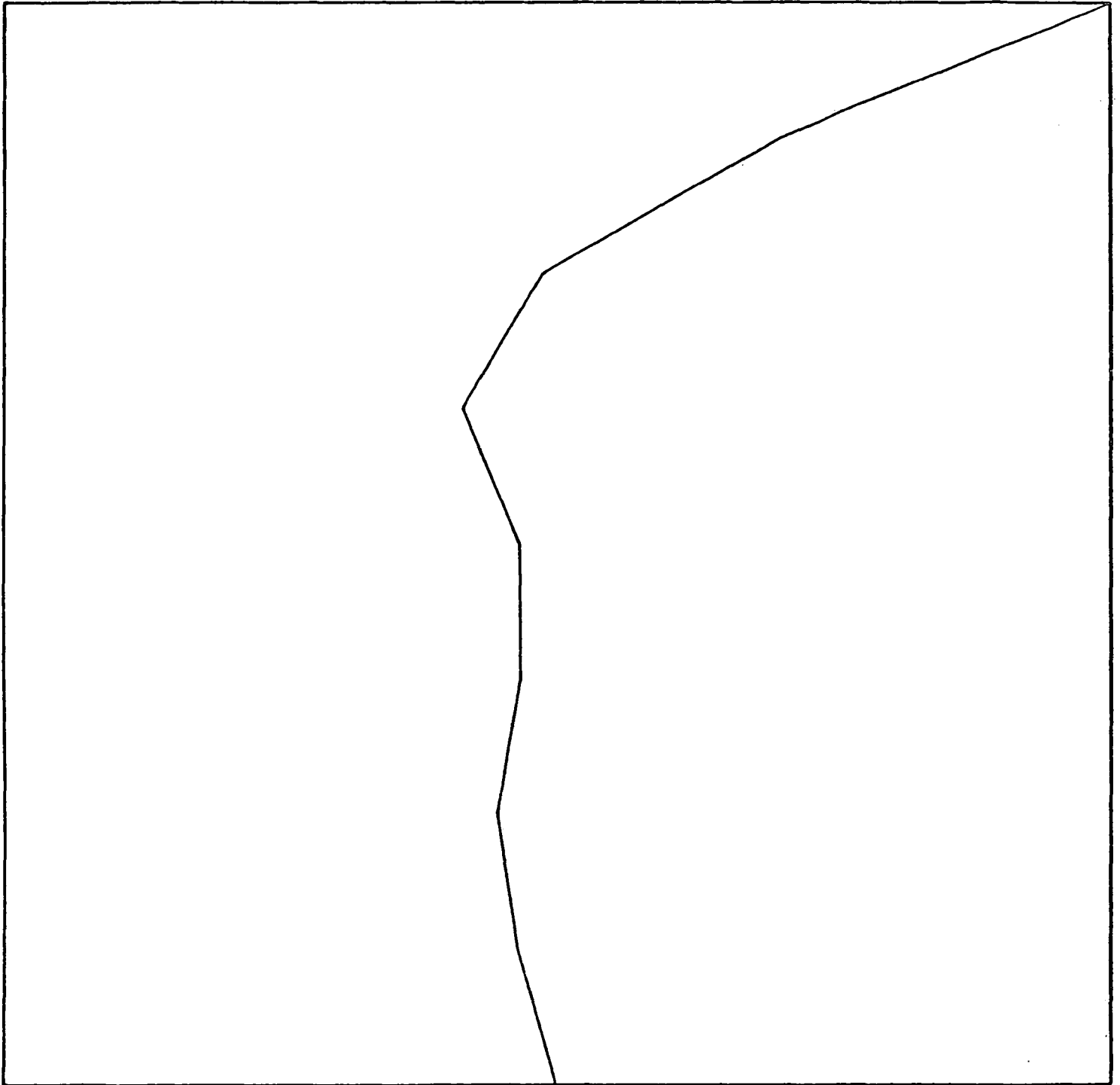
$NX, NY = 8, REYNOLD = 1000, EPS = 1E-5$

# CHAMP DES VITESSES, COIN BAS-GAUCHE



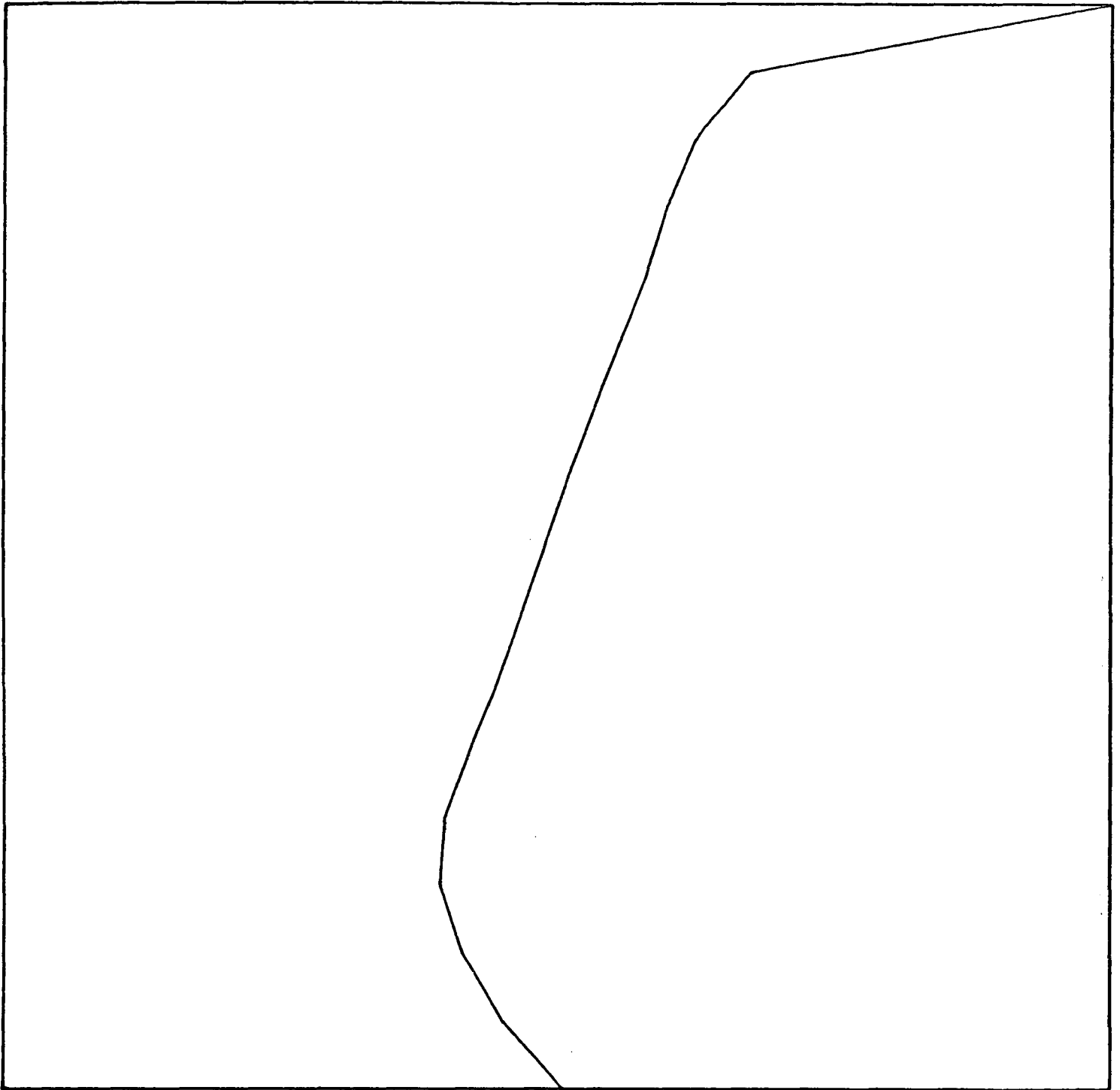
$NX, NY = 8, REYNOLD = 1000, EPS = 1E-5$

COMPOSANTE HORIZONTALE DE LA VIT. LE LONG DE LA MEDIANE



$NX, NY = 8, REYN0 = 1000, EPS = 1E-5$

COMPOSANTE HORIZONTALE DE LA VIT. LE LONG DE LA MEDIANE



$NX, NY = 8, REYNO = 1000, EPS = 1E-5$

Résultats numériques sur la convergence - calcul d'erreur :

L'erreur due à l'approximation par des éléments finis de type "Q2" est en  $O(h^2)$ ,

dans notre cas pour  $NX = NY = 4$  :  $\approx 5 \cdot 10^{-1}$

pour  $NX = NY = 8$  :  $\approx 10^{-2}$

- sur les tableaux qui suivent nous nous apercevons que l'on peut considérer la convergence en :

6 itérations pour Reynolds 400,

8 itérations pour Reynolds 1000.

- les tableaux d'erreur donnent un coefficient de proportionnalité entre les itérations (de 5 en 5). Pour lire ces tableaux il faut savoir que "plus le nombre est élevé, plus l'erreur est faible".
- la comparaison des deux courbes donnant "la différence entre deux itérations" semble confirmer le fait qu'il existe un lien entre "EPS" et le pas de discrétisation, ceci influençant la convergence.

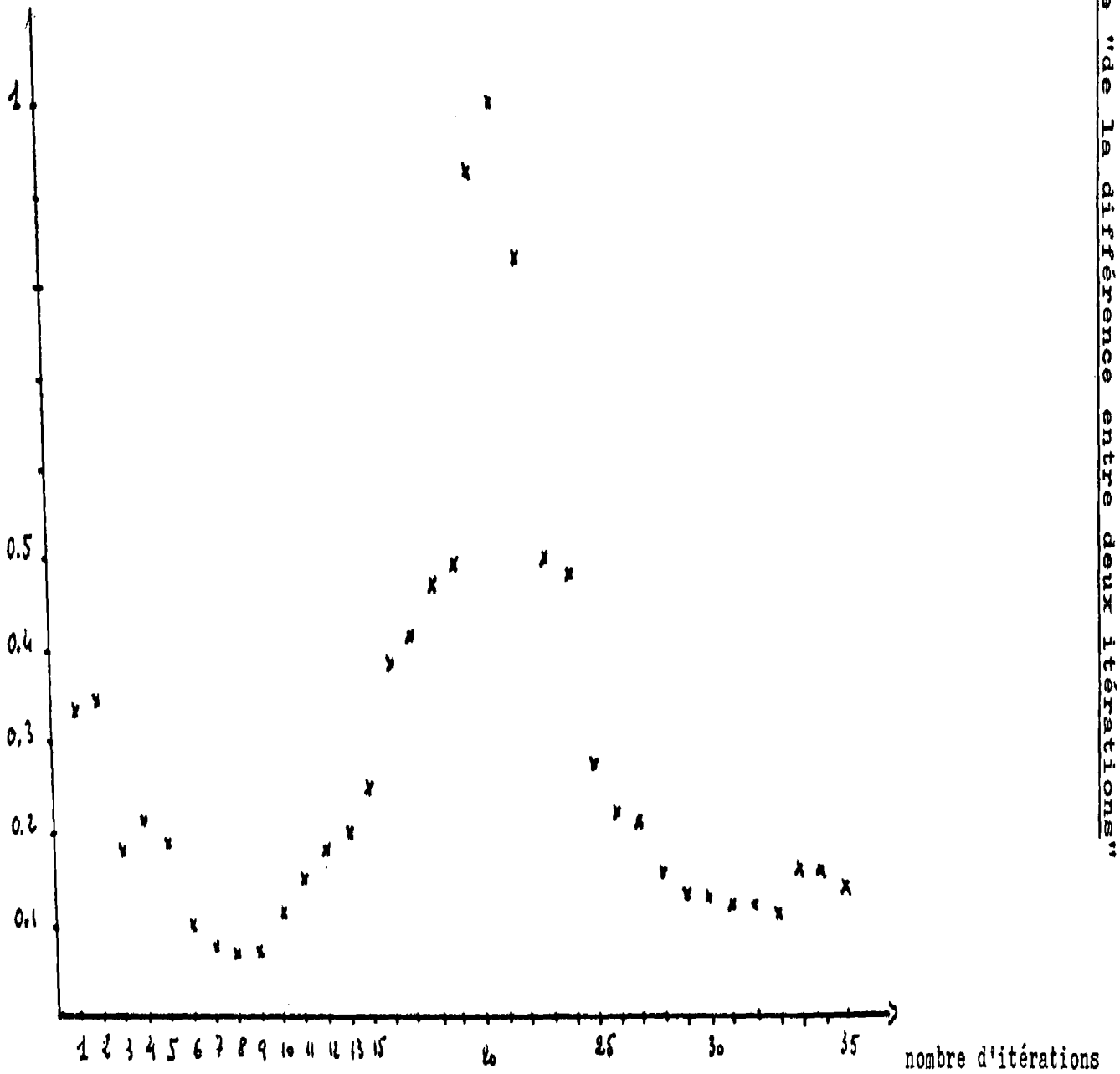
maillage : 16 quadrangles (NX=NY 4)

Reynolds : 1000

EPS :  $10^{-5}$

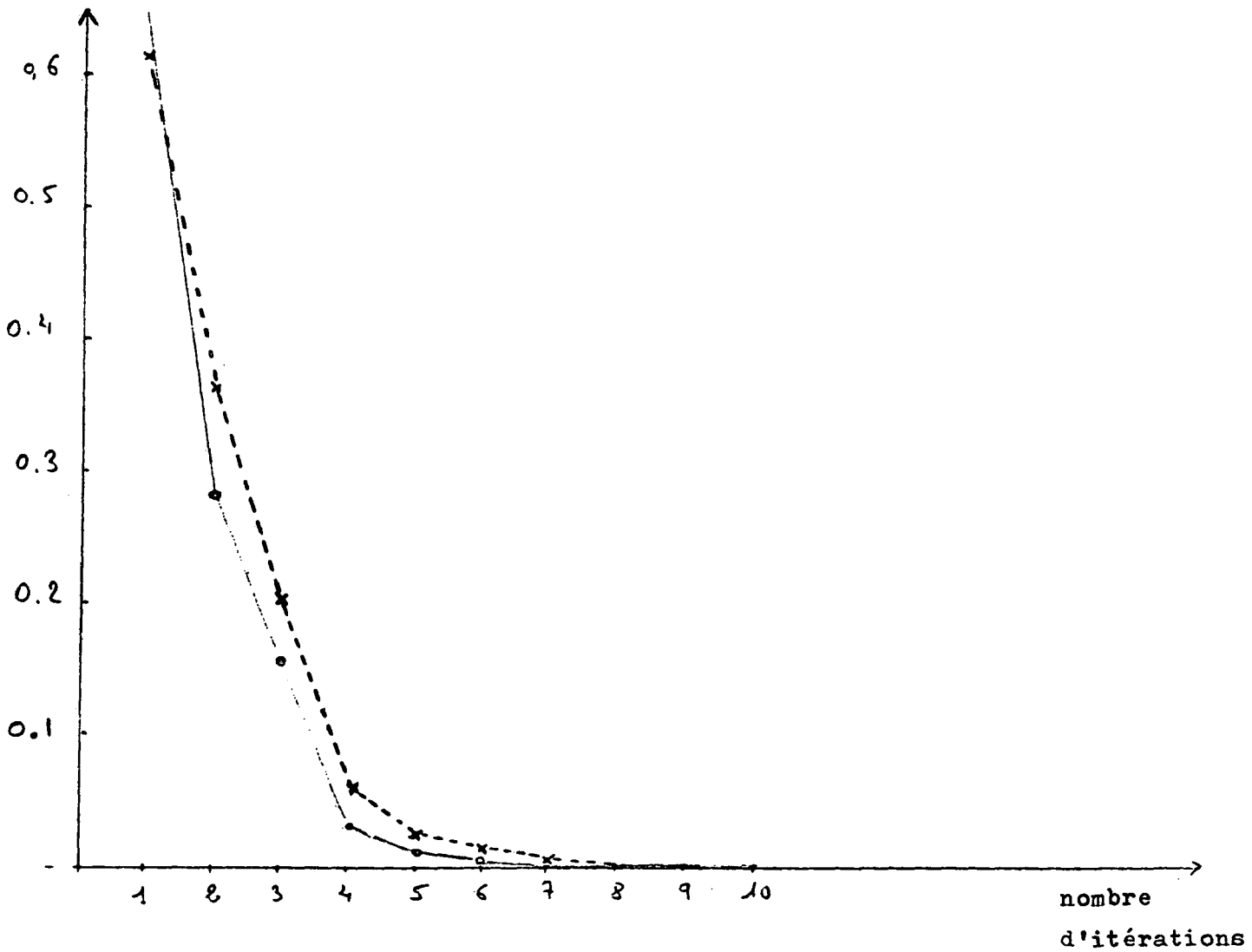
différence entre les  
deux itérations

erreur due à la méthode d'éléments finis  $\approx 5 \cdot 10^{-2}$



maillage : 64 quadrangles (NX = NY = 8).  
Reynolds : 1000 (x) ; 400 (o)  
EPS =  $10^{-5}$

différence entre les  
deux itérations





Reynolds : 400  
nombre de subdivisions par axe : 8  
second membre nul  
initialisation au vecteur nul.

numéro de l'itération	différence entre les deux itérations	erreur relative entre deux itérations
1	0.6618	1
2	0.2843	120.20
3	0.1563	3.840
4	0.0337	8.9316
5	0.0118	4.241
6	0.0073	0.1604
7	0.0021	0.0750
8	0.0012	0.0286
9	0.0007	0.0168
10	0.0002	0.0113





ORDONNEES	VITESSE HORIZONTALE
0.0	0.0
0.062500	-0.060371
0.125000	-0.112516
0.187500	-0.161767
0.250000	-0.201964
0.312500	-0.215730
0.375000	-0.196545
0.437500	-0.149973
0.500000	-0.091884
0.562500	-0.030559
0.625000	0.026831
0.687500	0.085207
0.750000	0.136175
0.812500	0.182938
0.875000	0.238926
0.937500	0.450743
1.000000	1.000000

LA NORME L2 DE LA DIVERGENCE EST : 0.305622D-06

Reynolds : 1000

nombre de subdivisions par axe : 8

second membre nul

initialisation au vecteur nulle

numéro de l'itération	différence entre les deux itérations	erreur relative entre deux itérations
1	0.63471	1
2	0.36025	167.85
3	0.20462	19.483
4	0.0610	19.306
5	0.0193	1.5943
6	0.0126	2.1193
7	0.0063	0.6326
8	0.0030	0.2485
9	0.0041	0.1583
10	0.0027	0.1147





ORDONNEES	VITESSE HORIZONTALE
0.0	0.0
0.062500	-0.108826
0.125000	-0.181224
0.187500	-0.221471
0.250000	-0.213592
0.312500	-0.169230
0.375000	-0.120310
0.437500	-0.077520
0.500000	-0.036327
0.562500	0.006114
0.625000	0.051592
0.687500	0.097573
0.750000	0.150842
0.812500	0.188235
0.875000	0.239815
0.937500	0.339727
1.000000	1.000000

LA NORME L2 DE LA DIVERGENCE EST : 0.2071160-06



Bibliographie

références générales :

- [ 1 ] M. BERCOVIER : Thèse de Doctorat d'état, Rouen, 1976.
- [ 2 ] M.S. ENGELMAN : Simulation of glow models of Newtonian and non-Newtonian viscous incompressible glows, Thesis, Hebrew University of Jerusalem, 1979.
- [ 3 ] P. GERMAIN : Mécanique des milieux continus, Masson, 1962.
- [ 4 ] J.L. LIONS : Quelques méthodes de résolution des problèmes aux limites non linéaires, Dunod, 1969.
- [ 5 ] R. TEMAM : - Une méthode d'approximation de la solution des équations de Navier-Stokes, Bull. Soc. Math. France 96, 1968.  
- Navier Stokes Equations, North Holland, 1977.
- [ 6 ] J.C. BENAZETH : Résolution des équations de Navier-Stokes stationnaires par une méthode d'éléments finis mixte, Thèse, Paris, 1978.
- [ 7 ] F. THOMASSET : F.E.M. for Navier-Stokes equations, INRIA, 1980.
- [ 8 ] F. BREZZI : On the existence, uniqueness and approximation of saddle point problems arising from Lagrangian multipliers , R.A.I.R.O., vol. 8, R2 , 1974.

références pour la méthode frontale :

- [9] A. LICHNEWSKY : Cours de DEA, Université Paris XI, Orsay, 1980.
- [10] B.M. IRONS : A frontal solution program for finite elements analysis, Int.J.Num.Meth. Eng., 2, 1970.
- [11] P. HOOD : Frontal solution program for unsymmetric matrix, Int.J.Num.Meth.Eng., 10, 1976.
- [12] O.C. ZIENKIEWICZ : The finite element method, 3<sup>rd</sup> edition, McGraw-Hill, 1977.
-

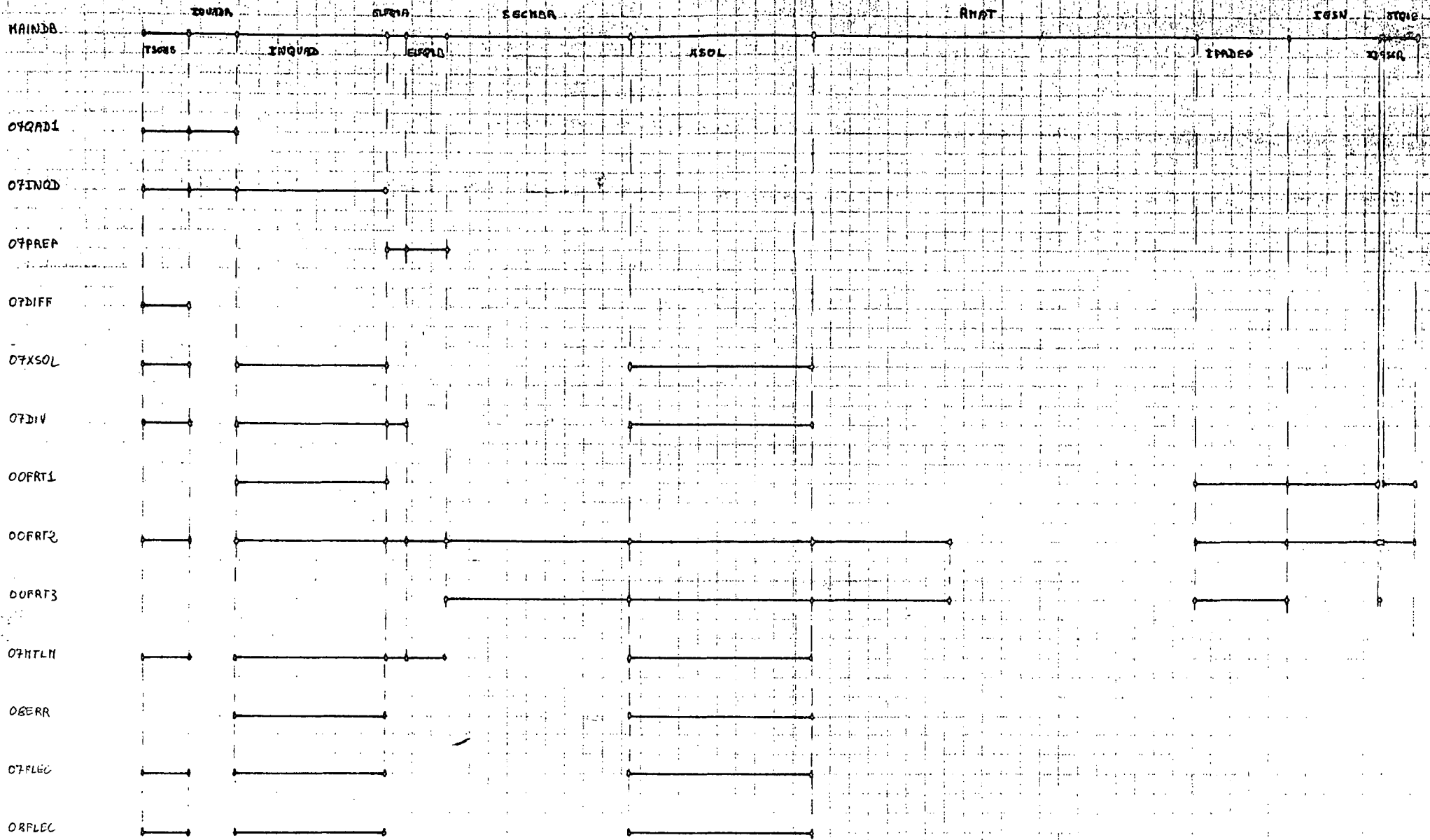
Répartition de la zone mémoire occupée par les principaux tableaux :

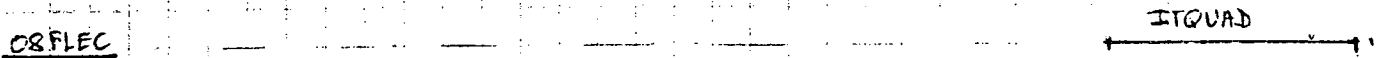
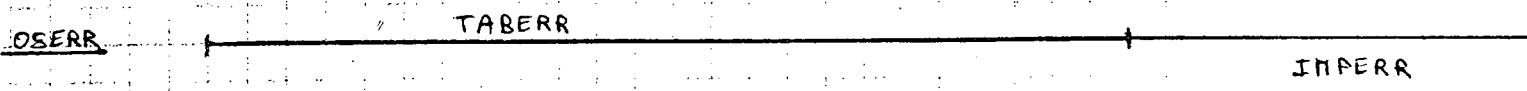
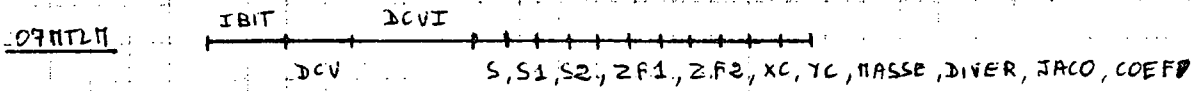
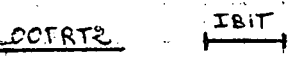
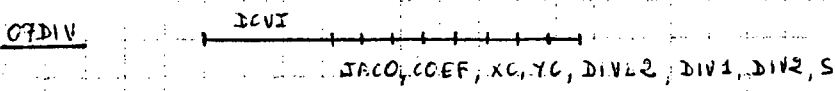
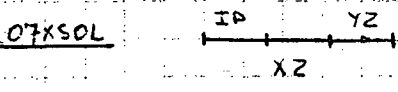
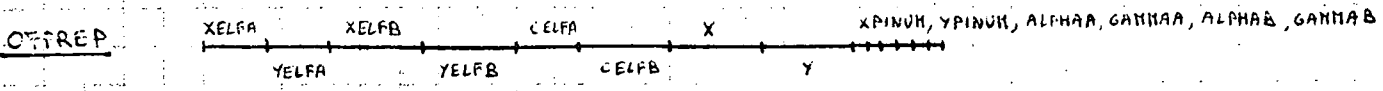
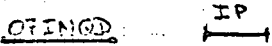
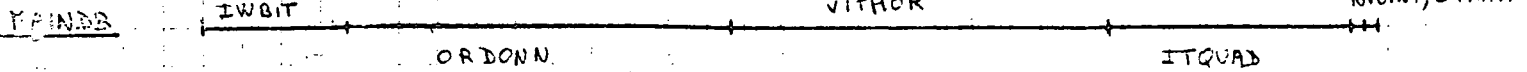
On remarquera que la taille des tableaux `RMAT` et `IDESCR` dépend de la stratégie utilisée pour la méthode frontale. Il faut donc faire attention à déclarer dans le programme principal une dimension suffisante. Il faut la répéter dans le commun `ZLFTAB` (`LFMAT = ...`). Le programme calcule lui-même la taille mémoire nécessaire (`LNMAT = ...`) et s'arrête si celle-ci est trop grande (c'est-à-dire `LNMAT > LFMAT`).

Dans la version actuelle `RMAT(N,N)` et `IDESCR(N)` sont dimensionnés pour `N = 80`, ceci permet par exemple de traiter le cas d'un carré subdivisé en `16 x 16` éléments avec une stratégie quadrangle par quadrangle.

Sur la figure suivante la longueur utilisée réellement correspond à `NX = NY = 4` avec la même stratégie.

250 mli memoire





(cf programme principal)

Structure du programme "d'élément fini et résolution" :

Programme principal

<u>07 init</u>	"initialisation des communs".	
<u>07 qad 1</u>	"quadrangulation d'un rectangle".	
<u>07 inqd</u>	"remplissage du tableau décrivant la méthode d'elt. fini".	
<u>07 prep</u>	"définition de la méthode d'intégration numérique sur le quadrangle de référence".	
<u>07 diff</u>	"modification du maillage par difféomorphisme en vue de tester 07 MTLM".	
<u>07 Xsol</u>	" initialisation de la solution, en tenant compte des conditions de Dirichlet au bord".	FU1 FU2 07 PRTD
<u>07 Div</u>	"calcul de la norme $L^2$ de la divergence de la vitesse (en Q1, points de Gauss(4))".	
<u>00 FRT1</u>	"création des tableaux nécessaires à la méthode frontale".	07 PRTI 00 TABI

[ F1  
          ↑  
          F2

00 FRT2  
"assemblage de la matrice et du second  
  membre et descente de la méthode  
  frontale".

07 MTLM création de  
          la matrice  
          élémentaire

07 PRTD  
07 PRTI  
00 TABD

00 FRT3  
"remontée de la méthode frontale".

07 PRTI  
00 TABD

08 err  
"erreur entre deux itérations".

07 div  
"cf précédemment".

07 Flec  
"représentation d'un champ de vecteur  
  par des flèches".

GRFBE 2

08 Flec  
"agrandissement de quelques quadrangles,  
  du progamme 07 Flec, et donne la  
  direction du champ".

GRFBE 2

GRFBE 1

Programmes d'impressions et fonctions :

Impressions :

O7 PRTD  
imprime un vecteur double précision.

O7 PRFI  
imprime un vecteur d'entiers.

OO TABI  
imprime un tableau d'entiers.

OO TABD  
imprime un tableau double précision.

Fonctions :



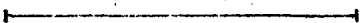

FU1  
donne la composante horizontale de la vitesse  
sur un bord et en un point.

FU2  
donne la composante verticale de la vitesse  
sur un bord et en un point.

F1  
donne la composante horizontale de la force  
en un point.

F2  
donne la composante verticale de la force  
en un point.



ONES	Z WORK	Z LEFTAB	Z UNITS	Z CONTX	Z QAD1	ZISK	Z QUA
MAINDB	 ZW(1600) (DP)	LZW, LFTSON, LFQUAD LFXSOL, LFSM, NFRONT LFPADG, LFHAT	ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR, NBOARD, NCOTE NNCEUD, NNBASE, NFNBAS NPINUA, NPINUB, NITER	XDIM, YDIM NX, NY	NDESCR NTABLE	
07INIT			ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR, NBOARD, NCOTE NNCEUD, NNBASE, NFNBAS NPINUA			
07QAD1		LZW, LFTSON, LFQUAD	ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR	XDIM, YDIM NX, NY		
07INQD	 ZW(3200)	LZW, LFTSON, LFQUAD	ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR, NBOARD, NCOTE NNCEUD, NNBASE, NFNBAS			
07PREP			ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR, NBOARD, NCOTE NNCEUD, NNBASE, NFNBAS NPINUA, NPINUB			
07DIFF		LZW, LFTSON	ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC NSOMS			
07XSOL		LZW, LFTSON, LFQUAD, LFXSOL	ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR, NBOARD, NCOTE NNCEUD, NNBASE, NFNBAS NPINUA, NPINUB			
07DIV	 DUAL(1600) (DP)	LZW, LFTSON, LFQUAD, LFXSOL	ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR, NBOARD, NCOTE NNCEUD, NNBASE, NFNBAS NPINUA, NPINUB			MUL, NZ(8) XZ(4), YZ(4) POL(2,4) POL I(4)
07PAT1	 XDIM(1600)	LZW, LFTSON, LFQUAD LFXSOL, LFSM, NFRONT LFPADG, LFHAT	ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR, NBOARD, NCOTE NNCEUD, NNBASE, NFNBAS NPINUA, NPINUB		NDESCR	
07TAB		LZW, LFTSON, LFQUAD LFXSOL, LFSM, NFRONT LFPADG, LFHAT	ULECT1, ULECT2 UPAT1, UPAT2, UPAT3	IPRINT, ITEST, IPREC, NSOMS NQVADR, NBOARD, NCOTE NNCEUD, NNBASE, NFNBAS NPINUA, NPINUB		NDESCR NTABLE	MUL, NZ(8) XZ(4), YZ(4) POL(2,4) POL I(4) MATH(15,15)

→ 250 meta  
(DP) double,

COMMUNS PAGES	ZWORK	ZLRTAB	ZUNITE	ZCONTX	ZQAD1	DISK	ZQUA
00FRT3		LZW, LRTSON, LQQUAD LFXSOL, LFSM, NFRADMT LFRPDG, LFMAT	ULECT1, ULECT2 UPRT1, UPRT2, UPRT3	I PRINT		NDESCR NTABLE	
00FRTLH	$\overbrace{\text{SONGAD}(18,18), \text{SONDIV}(18,18)}^{\text{DP}}$ $\text{A}(18,18), \text{PHI}(9), \text{DPH1}(9,2)$	LZW, LRTSON, LQQUAD LFXSOL, LFSM, LFMATR LFMAT, LFMTR	ULECT1, ULECT2 UPRT1, UPRT2, UPRT3	I PRINT, ITEST, IPREC, NSONS NQVADR, NBORD, NCOTE NNOEUD, NNBASE, NFNBAS NFINUA, NFINUB			$\overbrace{\text{NUL}, \text{NZ}(9), \text{XZ}(4), \text{YZ}(4), \text{POL}(2,4), \text{POLJ}(4), \text{MATLN}(18,18), \text{SM}(18)}^{\text{DP}}$
00BEAR	$\overbrace{\text{XSOL1} \text{ (liste d'appel) } (1200)}^{\text{DP}}$	LZSW, LRTSON, LQQUAD LFXSOL	ULECT1, ULECT2 UPRT1, UPRT2, UPRT3		XDIM, YDIM NX, NY		
00FLEC	$\overbrace{\text{ZW} (2200)}$	LZSW, LRTSON, LQQUAD LFXSOL	ULECT1, ULECT2 UPRT1, UPRT2, UPRT3	I PRINT, ITEST, IPREC, NSONS NQVADR, NBORD, NCOTE NNOEUD, NNBASE, NFNBAS			
00FLEC	$\overbrace{\text{ZZW} (2500)}$	LZW, LRTSON, LQQUAD LFXSOL, LFMATR, LFMAT		I PRINT, ITEST, IPREC, NSONS NQVADR, NBORD, NCOTE NNOEUD			

← 250 mots mémoire

DP double précision

PROGRAMME DE RESOLUTION DES EQUATIONS DE  
NAVIER-STOKES PAR UNE METHODE  
D'ELEMENTS FINIS MIXTE DE BERCOVIER

AUTEUR :Patrick GUILLAUME

REVISE :Mr SELIGMAN  
HDe CREPEL

CETTE DOC: Alain LICHNEWSKY

<<< LABORATOIRE D'ANALYSE NUMERIQUE >>>  
<<< Universite de PARIS-SUD >>>  
<<< 91405 ORSAY >>>

VERSIONS:

- (1) Resolution des Systemes lineaires par  
Methode "LU" Matrice profil  
disp:Juin 79
- (2) Resolution par Methode de Gauss/Frontale  
disp:Juin 80

REMARQUES 1/Les programmes documentes ici ont ete realises  
par divers stagiaires debutants, le lecteur  
excusera donc les erreurs de jeunesse dans les  
programmes.  
2/Pour la documentation, les remarques permettront  
de l'ameliorer. M E R C I.

## T A B L E D E S M A T I E R E S

1	Notations
2	Methode de Bercovier, Matrice Profil: programme principal
2.1	types de donnees
2.2	communs & variables globales
2.3	description de l'algorithm global
3	o7init
4	o7gad1 quadrangulation domaine
5	o7inqd identification degres de liberte
6	o7prep definition des fonctions de forme
7	o7mtlm calcul des matrices elementaires
7.1	definition des matrices elementaires, et des applications entre element de reference et element courant.
7.2	description o7mtlm
8	o7xsol initialisation solution
9	o7div calcul norme de la divergence
10	NOTE stockage des matrices profil symetrique
11	o8asmr assemblage
12	Methode FRONTALE
12.1	principe
12.2	structures de donnees
12.3	schema des indexations/indirections
13	Navier_Stokes_frontal programme principal methode frontale
13.1	sous programmes utilises
13.2	structures de donnees
13.3	communs & variables globales
13.4	description
14	o0frt* description des programmes
14.1	o0frt1 preparation
14.2	o0frt2 descente/assemblage
14.3	o0frt3 remontee

## &lt;&lt;NOTATIONS&gt;&gt;

- 1/ On utilise l'écriture :: <var>:+=<var1>  
pour indiquer :: <var>=<var>+<var1>
- 2/ On utilise "≈" pour indiquer "approximativement"
- 3/ On utilise le FOR\_EACH ("élément d'un ensemble") DO..  
dans une situation où l'ordre des opérations n'est pas importante et où un parallélisme pourrait exister.
- 4/ On a cherché à expliciter les indirections et autres correspondances. Dans cette documentation les correspondances sont notées par  
(T\_indx1 -> T\_indx2)  
ou T\_indx sont des types d'indices relatifs à des numérotations distinctes. Ce sont des FONCTIONS.

```

-----
| Navier_Stokes_Profil:
|   MAIN_PROGRAM:
|   DECLARE      SUBROUTINE
|
|       o7init   (INTEGER),           --Sous Programmes
|       o7qad1   (                    --Initialisations o7..
|               T_tsoms,              --Construction maillage
|               T_elem,               --Tableau sommets
|               T_deglib,             --tableau elements
|       o7inqd   (                    --Construction table de
|               T_tsoms,              -- degres de liberte
|               T_elem,               --
|               T_deglib,             --
|       o7prep   (                    --Construction table
|               T_descr,T_descr,     -- decrivant les elem-
|               ,                    -- -de reference
|       o7diff   (T_tsoms),           --Deformation maillage
|       o7xsol   (                    --Initialisation du
|               T_tsoms,              -- vecteur "solution"
|               T_deglib,             --
|               T_vect(lnxsol),       --vecteur solution
|       o7div    (                    --Calcul norme
|               T_tsoms,              -- divergence vitesse
|               T_deglib,             --
|               T_descr,              --Descr. pour int.num.
|               T_vect(lnxsol)),      --Solution
|       o8asmr   (                    --Assemblage de la
|               T_tsoms,T_deglib,     -- matrice profil
|               T_descr,T_descr,     --pour o7mtlm
|               T_mat_prs,            --Matrice profil
|               T_mat_prs_ptr,        --"adressage matrice
|               ,                    -- creuse "
|               T_vect,               --Second membre
|               T_vect,               --Solution (pour calcul
|               ,                    -- des non-linearites)
|               ,                    --
|               ,                    --
|       mulmat   (T_mat_prs,          --Produit matrice profi
|               T_mat_prs_ptr,        -- par le vecteur:
|               ,T_vect),             -- ((*).0)
|       crmc3d   (T_mat_prs_ptr,      --Resolution methode
|               T_mat_prs,            -- de Crout pour matr.
|               ,                    -- profil sym. non
|               ,                    -- sym / (C)MODULEF/
|               T_mat_prs,            --
|               OPTION (ARGS(2,7) MAY_BE_OVERLAPPED)
|               ,                    --regarder avant d'
|               ,                    -- optimiser
|       drcr3d   (T_mat_prs_ptr,      --Resolution m.Crout
|
|-----

```

DOCUMENTATION: A. LICHNEWSKY -- UNIVERSITE PARIS-SUD ORSAY  
 RESOLUTION EQ. DE NAVIER-STOKES -- METHODE DE BERCOVIER --

```

                                T_mat_prs,      -- /(C) MODULEF/
                                T_vect(*) ,      --Zone de travail
                                ,,,T_vect) ,     --Solution
o7prtd  (T_vect,,) ,                --Impression
o8err   (T_deglib,                  --Mise en forme et
                                -- impression de la
                                -- difference entre
                                -- iteres
                                T_vect,T_vect,  --Solutions a comparer
                                ,,,) ,
(startg,o7flec,o8flec,            --Pgms. graphiques
grfbel,contrl)                  -- sans interet ici
                                IN "graphics" PACKAGE; -- vu? ..

```

```

*****
*
*      Voici les principaux types de tableaux
*      manipules
*
*****
      DECLARE      TYPE
      ( T_vect:REAL(*) ),          --Vecteurs
      ( T_tsoms: 1 sommet (nsoms), --Pour chaque sommet:
        2 (x,y) REAL , --les coordonnees
        2 code INTEGER)--suivant que sur le
                        , -- bord du domaine...
      ( T_element:
        1 element (nquadr),--
        2 nsom INTEGER(4),-- numero de
                        -- sommet.(Ceci sert
                        -- a definir la geo-
                        -- metrie.)
      ( T_deglib:1 element (nquadr)--Pour chaque element:
        2 nsom INTEGER(4), -- num. sommets
        2 ndeglib_sommets --num. des degres de
                        -- liberte correspond.
                        INTEGER(4), -- aux sommets
        2 ndeglib_milieu_cotes
                        --degres de liberte
                        -- associes aux milieu
                        -- des cotes
                        INTEGER(4),
        2 ndeglib_centre
                        --degre de liberte
                        INTEGER)-- au centre
        2 ndeglib --au total 9 degres de
                        --liberte par element
                        REDEFINES ndeglib_sommet
                        THRU ndeglib_centre
                        INTEGER(9)
      (T_descr:
        --description de
        -- l'element de ref.
        1 point_integr_num --par point d'
                        \ -- integration
                        -- numerique sur l'
                        (npinu*) , -- element reference
        2 (x,y) REAL , --coordonnees
        2 coet REAL , --coeff.integration
        2 code INTEGER,--

```



```

2 (val_fonc_base--fonctions de base
  , val_ddx , -- leurs d/dx
  val_ddy ) -- leurs d/dy
REAL(9) ,

--MATRICES dont le profil est symetrique mais
-- dont les coefficients ne le sont pas forcement.
--Stockees a l'aide de deux tableaux
-- l'un decrivant la structure
-- l'autre contenant les coefficients

(T_mat_prs_ptr: --pour chaque ligne l
                -- de la matrice
                INTEGER(nfbas+1)) , -- position dans le
                -- vecteur T_mat_prs
                -- de val(mat(l,l))
(T_mat_prs : REAL(taille);--coefficients .

-- 1/ nfbas=nombre d'inconnues du probleme.
-- On represente 2 composantes de la vitesse
-- par elements finis. A chaque degre de liber-
-- te correspondent 2 fonctions de base inter-
-- venant chacune dans l'approximation d'une
-- composante. (Le nombre de degres de liberte
-- est designe par 'nnoeud' dans le programme)
-- Du fait des conditions au bord , il y a
-- 'nbase' degres de liberte non figes
-- et 'nfbas' inconnues .
-- 2/ Heuristiques pour la taille.
-- Pour un probleme bidim.
-- taille # c2*(nsons)**1.5
-- Pour un probleme tridim.
-- taille # c3*(nsons)**1.67
-- (Ici seuls les exposants sont a considerer
-- )
-- On a privilegie l'acces par ligne pour la partie
-- de la matrice en dessous de la diagonale, et
-- l'acces par colonnes pour le reste de la matrice
-- dans cette representation.
-- La FUNCTION ncmuns permet
-- l'adressage des elements representes a partir
-- de leurs indices (l,k). Cet adressage individuel
-- ne tire pas parti de la possibilite d'adresser

```

-- "plus directement" les portions de lignes ou  
-- colonnes. Dans Navier\_Stokes\_profil  
-- on adresse les elements un par un sans chercher  
-- ce gain en efficacite, en particulier dans  
-- 'mulmat'. (Cette remarque ne s'applique pas aux  
-- programmes 'crmc3d' et 'drcc3d')

```

*****
*
*      Organisation des Communs D'Utilisation Globale
*
*      (sont exclus les communs, qui ne sont connus que localement)
*
*****

```

```

DECLARE      COMMON      --
              zcontx      --options globales pour les
                          -- programmes o7XXXX o8XXXX
                          --dimension des tableaux
              WRITTEN(o7init,o7inqd,o7prep,o7qad1,
                      Navier_Stokes_profil )
              KNOWN  (o8asmr,o7div,o7mtlm,o7flec,o7xsol,
                      o8flec )
              zlftab      --longueur maximales disp.
                          -- pour les tableaux.
              WRITTEN(Navier_Stokes_profil)
              KNOWN  (mulmat,ncmuns,o7flec,o7inqd,o7mtlm,
                      o7xsol,o8asmr,o8flec)
              zunite      --unites e/s
              WRITTEN(Navier_Stokes_profil) -- P.princ.
              KNOWN  (*) , --sert partout
              zqad1      --parametres de o7qad1
              WRITTEN(o7qad1)--
              KNOWN  ("graphics"),-- sert a faciliter
                          -- la lecture des resultats
                          -- (mise en page .) .
              zwork      --zone de travail
              WRITTEN(o7div,o7flec,o7inqd,o8flec )
              KNOWN  (Navier_Stokes_profil);

```

```

*****
DECLARE LOCAL_VAR      --Principaux tableaux pour tout
                          -- le programme
              xsol :T_vect, --solution apres l'iteration
              zw  :T_vect, -- a l'iteration preced.
              tsoms :T_tsoms, --Sommets
              iquadr:T_element, --Elements
              inquad:T_deglib , --Degres de liberte
              (elfq2a,
              elfq2b):T_descr , --Formules integration numerique
              mat :T_mat_prs , --| M A T R I C E

```

DATE=05:11:80

PAGE= 10

DOCUMENTATION: A. LICHNEWSKY -- UNIVERSITE PARIS-SUD ORSAY

RESOLUTION EQ. DE NAVIER-STOKES -- METHODE DE BERCOVIER --

pntr :T\_mat\_prs\_ptr:--| -----

DOCUMENTATION: A. LICHNEWSKY -- UNIVERSITE PARIS-SUD ORSAY

```

-----
| initialiser:                                --Initialisation des communs ( options
|   DO;                                        -- et dimensions variables)
|   CALL startg;                              --Pour les graphiques ....
| etiq_1000:                                  --Passage au cas suivant si 1°
|   CALL o7init;                              -- utilisateur est content et presse.
|   END initialiser;                          --
|
| geometrie :                                --Definition de la geometrie par les
|   do;
|   CALL o7qad1(tsoms,iquadr);-- tableaux des sommets et elements
|   CALL o7inqd(tsoms,iquadr,--Definition des degres de liberte
|     inquad);                                -- suivant geom. elements finis et
|                                           -- fonctions approchees:
|                                           --
|                                           --         vitesse ux,uy
|   IF(.)CALL o7diff(tsoms); --Deformation du domaine?
|
| END geometrie;
|
| CALL o7prep(elfq2a,elfq2b)                 --Definition fonctions de base de la
|                                           -- methode d°elements finis sur ref.
|
| -- Les donnees "statiques" sont maintenant pretes.On donne les
| -- valeurs initiales aux quantites ux,uy sur lesquelles on va
| -- iterer pour la resolution du probleme NON-LINEAIRE).
|
| CALL o7xsol(tsoms,inquad,xsol); --Remplit xsol:point de depart
| CALL o7div (tsoms,inquad,elfq2a, --Impressions seulement
|   xsol);
|
| -- <<<< I T E R O N S   M A I N T E N A N T   >>>>
|
| iteration_non_lineaire:
|   DO UNTIL("convergence");                 --On utilise la difference entre
|                                           -- iteres pour apprecier.
|
|   ...
|   CALL o8asmr (tsoms,                      --Assemblage de la matrice et du
|     inquad,                                -- second membre.
|     elfq2a,elfq2b,
|     mat,                                    --Matrice creuse sera remplie
|     pntr,                                  --T_mat_prs_ptr associe sera rempli
|                                           -- une fois pour toute au premier

```

```

-- appel.
secabr, --Second membre (terme non-lineaire)
-- a calculer
xsol --Vecteur solution (iteration prec.
-- sert ici a calculer termes non
); -- lineaires.
--Test sans interet en exploitation
IF ("test") THEN
test_assemblage_et_resolution:
DO;
CALL mulmat(...);CALL crnc3d(...);
CALL drcr3d(...);CALL o7prtd(...);
END test_assemblage_et_resolution;

-- Sauver valeur de xsol a l'iteration precedente pour tests
zw = xsol ;
-- Decomposer la matrice sous forme L*U
CALL crnc3d(pntr,mat,,,,mat,) ; -- 2 fois mat (pgm
-- (C) MODULEF )
-- Resoudre le systeme lineaire
CALL drcr3d(pntr,mat,secabr,,,,xsol); --xsol:solution ecrit
--secabr:verifier
-- Calculer la difference entre iteres
erreur=MAX(ABS(XSOL-ZW) (*));
IF (erreur<eps ) THEN "convergence=oui" ;
...
IF ("option_d_impression")
THEN DO;.....;END;
END iteration_non_lineaire;

IF ("option") THEN DO;
CALL o7prtd; -- impression solution
CALL o7flec;...CALL grfbel; -- Graphiques
END;

IF ("..") GO TO etiq_1000; -- passer au cas suivant si l'utili
-- -sateur n'est pas degoute.
fin:...
CALL contrl; -- n'interesse que les Graphiques
END Navier_Stokes_profil; -- voila !!!!

```

```

-----
o7init: SUBROUTINE ;
        DECLARE ARGUMENT IN
            ifile:integer          --fichier sur lequel on
                                   -- trouve les parametres
        DECLARE COMMON
            zcontx                 --options globales pour les
                                   -- programmes o7XXX o8XXX
                                   --dimensions tableaux
            WRITTEN(o7init,
                    )
            KNOWN (o8asmr,o7div,o7mtlm,o7flec,o7xsol,
                  o8flec,o7inqd,o7prep,
                  Navier_Stokes_profil),
            zunite                 --unites liees aux divers fichier
            WRITTEN(o7init)
            KNOWN (o7qad1,o7inqd,o7prep,o7diff,
                  o7xsol,o7div,o8asmr,o7prtd,
                  o8err,o8flec,o7flec);
        ...
        --initialisation des communs
END o7init;
-----

```

```

o7qad1:SUBROUTINE;
  DECLARE ARGUMENT IN
    iunit:INTEGER ; --unite lecture parametres
  DECLARE ARGUMENT OUT
    tsoms:T_tsoms ; --Table des sommets
    iquadr:T_elem ; --Table des elements(quadrangl)
  DECLARE COMMON
    zlftab          --Longueur disponibles pour
                   -- les tables crees.
    WRITTEN(Navier_Stokes_profil)
    KNOWN (o7qad1,
          )
    zcontx          --Options generales
                   --Nombre elements et sommets
    WRITTEN(o7init,o7qad1)
    KNOWN (o7****,o8****),
    zunite          --unites logiques fichiers
    WRITTEN(o7init) --initialisation
    KNOWN(*****) , --utilisation generale
    zqad1          --parametres maillage
    WRITTEN(o7qad1) --suivant demande
    KNOWN("graphics");--
DO;
  ...
  FOR_EACH(isomet)
    DO;
      tsoms.sommet(isomet)=...; -- code '0' :Interieur du
                                --          domaine
                                --          '1' :bord verticaux
                                --          '2' :bord horizont.
                                --          '-2' :bord horizont.
                                --          vitesse =0
    END;
  FOR_EACH(ielemt) --Boucle pour remplir table des elements
    iquadr.element(ielemt)=...; --Numeros des sommets
  IF("option") THEN ... ; --Impressions eventuelles
END o7qad1;

```



```

o7inqd:SUBROUTINE;                                --Identification et numerotation
                                                    -- des degres de liberte.
  DECLARE ARGUMENT IN
    tsoms :T_tsoms, --Table des sommets
    inquadr :T_element;--Table des elements
  DECLARE ARGUMENT OUT
    inquad :T_deglib ;--Table des degres de liberte
  DECLARE COMMON
    zcontx WRITTEN(o7init,o7qad1,o7inqd)
            KNOWN (o7****,o8****)
            :: (o...o, nnoeud, -- deglib figes ou non
               nnbas, -- deglib non figes
               nfbas,,) -- inconnues
    zunite WRITTEN(o7init)
            KNOWN(*****),
    zlftab WRITTEN(Navier_Stokes_profil)
            KNOWN(o7qad1,o7inqd),
    zwork WRITTEN(o7inqd) --zone de travail
            ; -- 'volatile'
    ... -- Initialisations
  FOR_EACH(element) -- Placer dans inquad les degres
    DO; -- de liberte correspondant aux
        inquadr(element).nson=...; -- sommets du maillage
        inquadr(element).ndeglib_sommet=...;
        -- Pour l'instant le ndeglib
        -- est egal au nson.Mais on
        -- renumerotera pour avoir la
        -- structure bande.
    END;

id_deglib: --Numeroter les degres de lib.
  DO; -- ceci necessite leur identifi-
      -- cation en tenant compte des
      -- proprietes 'globales' de la
      -- geometrie.
  -- Numerotation des degres de libertes portes par les sommets
  -- Il se trouve qu'il y a 1 degre de liberte par sommet et il
  -- suffit donc de lui attribuer le meme numero qu'au sommet.
  --
  -- Numerotation des degres de liberte portes par les cotes
  -- On emploie la methode suivante basee sur l'observation qu'un
  -- cote figure dans 2 elements au plus (1 si au bord)
  -- La structure liste_cotes ,implementee a l'aide de
  -- II (dans le COMMON ZW)

```

```

-- IDEP,JDEP pointeurs
-- represente la liste des cotes qui ont ete trouves une fois en
-- explorant le maillage element par element.Un cote est repere
-- par la cle formee des numeros des deux sommets dans l'ordre
-- croissant.
-- Lorsque l'on trouve un nouveau cote on lui affecte un numero
-- et l'on identifie les degres de liberte associes;on introduit
-- ce cote dans la liste liste_cotes.On tient a jour le premier
-- numero de degre de liberte disponible.
-- Lorsque l'on retrouve le cote on connait les degres de liberte
-- associes ,ceci permet de leur affecter le meme numero.On retire
-- alors le cote de la liste.
-- Ceci pourrait etre rendu plus performant en remplaçant la
-- liste par une structure permettant des recherches sur cle
-- plus efficaces.Toutefois la liste des cotes reste relativement
-- courte,d'ou la methode retenue.
--
-- Numerotation des degres de liberte a l'interieur des elements.
-- Ces degres de liberte ne sont rencontres qu'une fois lorsqu'
-- on parcourt la liste des elements.On leur affecte le premier
-- numero de degre de liberte disponible,que l'on met a jour.
--
--<<CECI documente les lignes 800 a 1920 de o7inqd>>

-- Mise a jour du commun zcontx contenant les dimensions
  nnoeud=      ; -- nombre degres de liberte
  nnbas =      ; -- ... non figes
  nfbas = 2*nnbas; -- nombre incounues

--Renumerotation pour obtenir une structure bande refletant la
--numerotation initiale des elements.
--on attribue les nouveaux numeros aux degres de libertes non figes
--dans l'ordre dans lequel ils apparaissent lorsque l'on parcourt
--le maillage element par element.
--les degres de libertes sur le bord sont places a la fin avec
--cette numerotation.(Ceci permet de tenir compte des donnees
--de Dirichlet non homogenes dans o7mtlm,o8asmr,o0frt2).
--pour ce faire on emploie une structure de liste implementee
-- a l'aide du tableau jj (dans le COMMON ZWORK).Ceci est
-- plus complique qu'efficace.(Une table donnant pour chaque
-- ancien numero le nouveau et permettant d'identifier les

```

-- degres de liberte non encore renumerotes suffit...)  
--<<CECI documente les lignes 1940 a 2750 >>

--Rappel sur degres\_de\_liberte et fn\_base

- 1/ Les degres de liberte sont lies a la methode d'element finis :ils 'parametrent' l'espace de fonctions dans lequel on represente chacune des composantes de la vitesse.  
Un degre de liberte est dit 'fige' s'il correspond a une donnee au bord (de type 'Dirichlet' non homogene).
- 2/ Le terme fonction de base fait ICI reference a la base de l'espace vectoriel ou evolue la solution du probleme representant les 2 composantes de la vitesse.CECI n'est pas une TERMINOLOGIE CLASSIQUE mais a ete utilise ICI de facon suffisamment systematique pour ne pas poser de probleme.
- 3/ Du fait de 1/ et 2/ une fonction de base pourra etre figee ou non.
- 4/ Au degre\_de\_liberte i on fait correspondre les fn\_base id\_1,id\_2 avec  $id_1=2*i-1$   $id_2=2*i$  pour maintenir une structure profil creuse au systeme global , du fait des COUPLAGES entre les 2 composantes.

END o7inqd;

```

o7prep: SUBROUTINE;                                --Definition des fonctions de
                                                    -- base (i.e. 'de FORME ') sur
                                                    -- l'element de reference, ainsi
                                                    -- que des formules d'integration
                                                    -- numerique.

    DECLARE ARGUMENT IN
        (idima, idimb) : --longueur fournies pour les
                        INTEGER ; -- tables
    DECLARE ARGUMENT OUT
        (elfq2a, elfq2b) : --tables decrivant les elements
                        T_descr ;
    DECLARE COMMON
        zcontx
        WRITTEN (o7prep),
        zunite;

-- Determiner si formules standart ?
READ(...); ..... ;IF ("dimension_insuffisante") THEN STOP;
...
IF ("nonstandart")
    THEN
        READ elfq2*.x(*), --Lire coordonnees des points et
            .y(*), -- coefficients associes.
            .coef(*) ; --L'utilisateur donne ici les
                        -- formules souhaitees.
    ELSE
        DO; --Calculer les coefficients pour
            elfq2*.x(*)=... -- les formules usuelles.
            elfq2*.y(*)=... --On dispose de plusieurs formule
            elfq2*.coef(*)=... -- les unes donnant des coeff.
        END; -- faciles a verifier, les autres
                -- visant la precision.

--Maintenant que les points et les coefficients sont connus, on
-- calcule UNE FOIS POUR TOUTES les valeurs en ces points des
-- fonctions de base et de leurs derivees.

    FOR_EACH (pinum) --operer par point d'integration
        DO; -- sur elmt.ref.
            elfq2*.val_fonc_base= ...; --dans le programme on remplit
            .val_ddx= ...; --separement les 2 tables bien
            .val_ddy= ...; --que les formules soient ident.
        END;
    IF ("impression_desiree") THEN .... --Pour verifications

```

```
-----  
| -- Toute la partie "METHODE D'APPROXIMATION / CHOIX DE L'ELEMENT |  
| -- FINI " est concentree dans 3 sous programmes :o7prep,o7mtlm et |  
| -- o7div.(o7div est une specialisation de certains 'sous-algorithmes'|  
| -- de o7mtlm destinee a calculer et imprimer la norme de la |  
| -- divergence .) |  
| -- Le reste est largement independant de l'approximation. |  
| -- |  
| END o7prep; |  
|-----
```

```

o7mtlm:SUBROUTINE;
--Calcul des matrices elementai-
-- res.Ces matrices elementaires
-- representent la contribution
-- d'un element aux matrices
-- representant le systeme
-- 'global'.(Il s'agit de
-- contributions ADDITIVES ).
DECLARE ARGUMENT IN
    inq      :INTEGER, --Numero de l'element considere
    tsoms    :T_tsoms, --Table des sommets
    inquad   :T_deglib, --Table des degres de liberte
    (elfq2*) :T_descr , --Description elmt. reference
    xsol     :T_vect  , --Vecteur vitesse a l'iteration
-- precedente,pour calcul des
-- NON-LINEARITES.
    (reyno,eps)
        :REAL , -- nb. de Reynolds,penalisation
-- divergence.
    icode    :BIT(4) --Options:permet de selection-
; -- ner les calculs desires.
DECLARE ARGUMENT OUT ()
--Tous les resultats sont places
-- dans le COMMON /ZQUA/ qui est
-- entierement calcule a chaque
-- appel.Ces resultats ne sont
; -- utilises "qu'une fois".
DECLARE COMMON
    zcontx ,
    zunite ,
    zlftab ,
    zwork  WRITTEN (o7mtlm) -- zone de travail
    KNOWN  (*****) -- Attention sert de zone
    , -- de travail "volatile".
    zqua   WRITTEN (o7mtlm) --Resultats :matrices
    -- elementaires.
    KNOWN  (CALLING_PROC)
    (o8asmr,front2);
--
-- Ce programme contient tout ce qui permet de calculer une quantite
-- sur un element de la discretisation du domaine.
--

```

## -- &lt;&lt; DESCRIPTION SIMPLIFIEE &gt;&gt;

-- De maniere generale on opere comme suit

- 1) le degre\_de\_liberte\_sur\_el\_reference ir 'correspond' au degre de liberte  $i = \text{inquad}(\text{inq}) . \text{ndeglib}(\text{ir})$  pour le systeme global.
- 2) On construit une application "changement de variable" CV envoyant l'element de reference sur l'element  $\text{inquad}.\text{element}(\text{inq})$ , de telle sorte que  $\text{ir} \rightarrow i$ . Cette application doit etre un diffeomorphisme assez regulier pour que l'on puisse effectuer le changement de variables dans les integrales, et calculer les gradients des fonctions de base.
- 3) Les fonctions sur l'element sont construites comme images de fonctions sur l'element de reference.
- 4) Le changement de variable est utilise pour ramener a des integrations numeriques sur l'element de reference.
- 5) On obtient des matrices elementaires indexees par degre de liberte 'ir' (c.f. 1/ ). Si bien que  $\text{mat}_{\text{lb}}(\text{ir1}, \text{ir2})$  est la contribution de l'element au terme  $\text{mat}(\text{ir1} \rightarrow i1, \text{ir2} \rightarrow i2)$  de la matrice "complete" (dite aussi "conceptuelle" ou "virtuelle" parceque TROP GRANDE POUR ETRE STOCKEE)

## -- &lt;&lt; DESCRIPTION PLUS PRECISE &gt;&gt;

-- En fait, tout ce qui vient d'etre dit plus haut est essentiellement valable dans le cas d'UNE SEULE EQUATION. Ici a un degre de liberte de la methode d'elements finis correspondent 2 fonctions de base associees au meme parametre (degre de liberte) dans le parametrage de chacune des composantes de la vitesse. Le systeme etant COUPLE il y a lieu d'en considerer les inconnues dans leur ensemble.

```

--
--
-- 1-bis/ Le degre_de_liberte_sur_el_reference ir ocorrespond
--         au degre_de_liberte i =inquad(ing).ndeglib(ir) dans
--         la description GLOBALE de l'approximation d'UNE
--         composante.
--         Il correspond aux fn_base
--         id = 2*i-1      pour la premiere composante
--         id = 2*i       pour la deuxieme .
--
-- 2-bis/ <cf. 2/ > .
--
-- 3-bis/ <cf. 3/ > .On travaille fonction inconnue par fonction
--         inconnue,pour chaque fonction on revient a la
--         definition de l'approximation et donc aux degres de
--         libertes et aux CV decrits.Les termes de couplage
--         sont des produits <de derivees> des 2 fonctions inc.
--
-- 4-bis/ <cf. 4/ > .
--
-- 5-bis/ On obtient des matrices elementaires indexees par
--         fn_base_sur_el_reference idr.
--         La correspondance
--         fn_base_sur_el_reference <-> fn_base
--                                     idr <-> id
--         s'effectue en passant par l'application
--         degre_de_liberte_sur_el_reference <-> deglib
--                                             ir <-> i
--         si bien que matlm(idr_1,idr_2) est la contribution
--         au terme mat(id_1,id_2) avec
--         idr -> id :: id = 2*inquad(ing).ndeglib(idr/2)-1+
--                               +MOD(idr,2)
--
--

```



```

--
construire_CV:                                -- construction changement vars.
DO;                                           --
  FOR_EACH (ir)                                -- pour chaque degre de liberte
  DO;                                         -- correspondant a un sommet
                                             -- sur l'elmt.ref.
      nz(ir)=inquad.ndeglib_sommet(inq,ir);--num "global"
      xz      =tsoms.x(inquad.nsom(inq,ir)) ;--coordonnee
      ...
    END;
    pol(*,*)= ....                          -- coefficients des polynomes
                                             -- representant CV.
    polj(*)= ....                            -- polynome representant |dCV|
  END construire_CV;                          --

--Integration numerique  termes ordre inferieur
calcul_1:FOR_EACH(pinum)                    -- on travaille point par point
DO;                                         -- de facon a ne calculer qu'une
  jaco=...                                  -- fois les donnees 'geometriques
  (x,y)=CV(elfq2a.point_integr_num.(x,y)(pinum))
  dcv(*,*)=(dCV)(x,y)  -- Derivees de CV en x,y
  dcvi="inverse(dcv)"  -- L'inverse sert aussi,mais on
                       -- simplifie un peu...
  --Fonctions de base au point x,y.On utilise aussi
  -- leurs derivees ,un peu plus complexes a calculer.
  ...
  --Interpole du vecteur vitesse xsol au point (x,y)
  --Ce vecteur est represente comme somme de fonctions
  -- de base,avec coefficients dans xsol.
  (s1,s2)=SUM(xsol((idr->id)(*)) *
              elfq2a.val_fonc_base((id->ir)(*)));
  --
  --remplissage de la contribution de ce point d'integra
  -- -tion numerique a toutes les quantites calculees.
  songrd(*,*)+=                               -- par produit de fonctions de
  b(*,*)+=                                     -- base en numerotation 'idr'
  sm(*) +=                                     -- Second membre
END calcul_1;

-- Termes de degre plus eleve      les formules se compliquent,mais

```

```

-----
-- ceci releve des memes idees que "calcul1"
calcul_2:DO;
    somdiv(*,*)= --
    END calcul2;
--
-- Matrices elementaires.
--
matlm=somgrd/reyno + somdiv/eps + (b-TRANSPPOSE(b))/2.;
--
-- Influence coaditions aux limites sur second membre
--
if ("option") then -- ce traitement peut etre
                    -- effectue au choix par o7mtlm
                    -- ou lors de l'assemblage
report_figes_sm:
DO; -- voir (*) ci dessous
FOR_EACH(idr_1) -- boucle sur colonnes matlm
IF((idr->id)(idr_1)>nfbas) -- ne concerne que les figes.
THEN
FOR_EACH(idr_2) -- boucle sur les lignes
-- reporter au second membre
-- les contributions de la
-- fn_base figee.
sm(idr_2):+=-matlm(idr_2,idr_1)*xsol((idr->id)(idr_1));
END report_figes_sm;
-- (*) NOTE
-- Les fonctions de base figees ne correspondent pas a
-- des inconnues du probleme :la matrice et le second membre ne
-- possedent donc pas de ligne les concernant. (Au moins dans leur
-- representation mathematique).
-- Toutefois le vecteur solution compte
-- 1/les inconnues effectives : de 1 a nfbas
-- 2/les valeurs figees :donnees de Dirichlet non homogenes de
-- nfbas+1 a 2*nnoeud.
IF("option_d_impression") THEN ...;
END o7mtlm;
-----

```

```

o7xsol:SUBROUTINE;                                --Initialisation solution
                                                    -- pour iterations non lineaires
                                                    -- permettant les donnees de
                                                    -- Dirichlet non homogenes

  DECLARE ARGUMENT IN
    tsoms: T_tsoms,                                --Geometrie
    inquad:T_deglib;                               --les degres de liberte indexent
                                                    -- naturellement xsol

  DECLARE ARGUMENT OUT
    xsol : T_vect(lnxsol), -- Vecteur vitesse
    lnxsol:INTEGER ; --longueur necessaire

  DECLARE SUBROUTINE
    --Voici les points d'entree
    -- par lesquels l'utilisateur
    -- donne ses conditions aux
    -- limites
    fu1 ((x,y):REAL IN -- coordonnees d'un sommet
        ,code:INTEGER IN ) -- code du sommet
    RETURNS (REAL), -- Composante horizontale vitesse
    fu2 LIKE fu1 ; -- ----- verticale -----

END o7xsol;                                       -- le reste ne presente pas
                                                    -- d'interet particulier pour
                                                    -- etre plus documente..

```

```

o7div :SUBROUTINE;                --Calcul de la norme L2 de la
                                   -- divergence de la vitesse.
--Cette quantite sert a juger du resultat,n'etant pas fixee a
-- 0, mais simplement l'objet d'une PENALISATION.
--
--Ce programme est une combinaison/specialisation d'algorithmes
-- de o7mtlm et o8asmr.

DECLARE ARGUMENT IN
    tsoms:T_tsoms,                -- Sommets
    inquad:T_deglib,             -- Degres de liberte
    elfq2a:T_descr,              -- Un seul type integration
    xsol :T_vect;                -- On calcule ||div(xsol)||
DECLARE ARGUMENT OUT ();         -- Sans influence exterieure
                                   -- ne fait qu'imprimer
DECLARE COMMON    zqua            -- Zone de travail "volatile"
                  WRITTEN(o7div)
                  zwork           -- Autre zone de travail
                  zlftab          ;
                  zcontx          ;

div2=0.;
FOR_EACH(element)                -- Balayage par elements
DO;
    ...                           -- Construire CV comme o7mtlm
    FOR_EACH(pinum)               -- Par point d'integration
    DO;
        ...                       -- Valeur CV au point,jacobien,
        dcvi(*,*)=                 -- dCV pour calcul des derivees.
        FOR_EACH (inquad(element).ndeglib(*))
        DO;
            div2:+="contribution"(xsol((i ->id) (inquad(element).ndeglib
            (*)))));
        END;
    END;
END;
END o7div;

```



L'interet de ce mode de stockage est que l'on ne represente que la partie de la matrice risquant de se remplir lors de :

- 1/Decomposition "LU" / "LDL" / "LL"
- 2/Resolution par methode d'elimination (GAUSS...)

Ceci signifie que si A est de type T\_mat\_prs et est associe a P\_A:T\_mat\_prs\_ptr, alors

1/A=LU avec L : triangulaire inferieure  
U : superieure  
et la matrice L+U est de type T\_mat\_prs et est aussi associee a P\_A.

Si on impose par exemple une des conditions  $\text{diag}(L) = (*) 1$  ou bien  $\text{diag}(U) = (*) 1$  on peut représenter tous les autres coefficients de L et U dans une matrice de type T\_mat\_prs associee au meme descripteur P\_A.

On peut aussi realiser la decomposition LU ou LL\* "en place".

Du fait du mode de stockage, on a privilegie l'acces aux colonnes de la partie de la matrice au dessus de la diagonale, et l'acces aux lignes en dessous de la diagonale.

Ceci permet certaines optimisations aussi bien pour le calcul des decompositions LU que pour la resolution des systemes de la forme  $LUX=b$ .

#### PLUS PRECISEMENT :

##### 1/DECOMPOSITION

L U = A

On identifie les coefficients de L et U un par un. Pour un coefficient a identifier on accede a une ligne de L et une colonne de U. Des optimisations sont possibles car on identifie en operant dans un ordre fixe: ligne de L, puis colonne de U. Le mode de representation assure une bonne localite des acces pour cet algorithme.

##### 2/RESOLUTION de $LUX=b$

On effectue successivement  $LY=b$  et  $UX=Y$

La resolution de  $LY=b$  ("descente") se fait naturellement en operant par lignes. Ceci permet l'accumulation de resultats partiels en double precision, sans stocker ni L ni U, ni X, ni b en double precision.

Pour la resolution de  $UX=Y$  l'acces par colonne est efficace en reordonnant le calcul:

```
DO i=DIM(U) TO 1 STEP -1
```

```
  Y(i) = ...
```

```
  FOR_EACH (j| j<i ) Y(j)=Y(j)-U(j,i)*Y(i);
```

DATE=05:11:80

PAGE= 29

DOCUMENTATION: A. LICHNEWSKY -- UNIVERSITE PARIS-SUD ORSAY

RESOLUTION EQ. DE NAVIER-STOKES -- METHODE DE BERCOVIER --

END ;

Ceci permet l'accès par colonnes de U. Toutefois, pour pouvoir accumuler les résultats partiels en double précision il faut stocker Y en double précision.



DOCUMENTATION: A. LICHNEWSKY

-- UNIVERSITE PARIS-SUD ORSAY

```

o8asmr:SUBROUTINE ;          -- Assemblage en vue de la
                              -- resolution par la methode "LU
                              -- La matrice est a profil syme-
                              -- trique, mais n'est pas nec.
                              -- symetrique.

  DECLARE ARGUMENT IN
    tsoms :T_tsoms ,          -- Table des sommets -> o7mtlm
    inquad :T_deglib,        -- Degres de liberte
    (elfq2a,elfq2b)          -- A l'intention de
    :T_descr ,              -- -> 07mtlm
    xsol :T_vect ,          -- Calcul NON LINEARITES ->o7mtl
    icode :INTEGER ,        -- Options
    reyno :REAL ,           -- Parametre "physique" du fluid
    eps :REAL ;            -- Parametre de PENALISATION div

  DECLARE ARGUMENT OUT
    mat :T_mat_prs,         -- Matrice profil symetrique
    pntr :T_mat_prs_ptr,   -- et pointeurs associes.
    secnbr :T_vect;        -- Second membre assemble aussi

  DECLARE COMMON
    zelftab,
    zcontx,
    zunate,
    zqua;                  -- Zone de communications avec
                              -- o7mtlm.Cette zone est utilise
                              -- par o7mtlm en ECRITURE seulem
                              -- ./Tenir compte de ceci pour
                              -- etudier la parallelisation de
                              -- appels a o7mtlm.

  -- IF ("option_de_icode") THEN
  rempli_pntr:              -- Remplissage de pntr:descripti
    DO;                      -- de la structure de la matrice
      ...                    -- verification dimension pntr.
                              -- pour verifier mat il faut
                              -- attendre de connaitre pntr.

      -- Premiere etape :pntr(j)=min(i|Vmat(i,j)≠0 & i<=j)
      DO i=1 TO LENGTH(pntr) ; -- indiquer que le terme diagona
        pntr(i)=i ;          -- doit etre represente.Sinon io
      END;                   -- decomposition impossible.

      -- Boucle sur les elements pour rechercher les contributions
      -- aux divers termes de Vmat.Ces contributions sont consider
      -- comme non nulle et on ne TIENT PAS COMPTE de l'annullatio
      -- de contributions entre elles.

```



```

FOR_EACH (element)          -- on reconstruit les positions
                             -- dans la matrice: fn_base id
                             -- a partir des degres de liberte.
DO;                          -- (Cf. o7mtlm)
  nummin=MIN((i->id) (inquad(element).ndeglib(*)))
                             -- voici la premiere colonne a
                             -- représenter pour les lignes et
                             -- ou intervient cet element.
  FOR_EACH (deglib IN inquad(element).deglib(*))
    IF (deglib<=nnbase) -- Seulement les deglib non figes
      THEN              -- correspondent a des inconnues e
      DO;              -- donc a des lignes de matrice.
        (lig_1,lig_2)=(i->id) (deglib); -- num. fn_base
        pntr(lig_1)=MIN(pntr(lig_1,nummin) -- voila
        END;
      END;
  -- Maintenant on a toute l'information necessaire pour
  -- positionner les coefficients diagonaux de Vmat et remplir
  -- pntr de maniere definitive.
DO i=1 TO nfbas+1;          -- Ce truc est fondamentalement
  pntr(i)=...              -- sequentiel.
END;
...                          -- verifier dimension de mat
END remplir_pntr;

remplir_mat:                -- calcul des coefficients
DO;
  FOR_EACH (element)        -- Par element:les contributions
                             -- sont des integrales calculees
DO;                          -- element par element.
  CALL o7mtlm(element,      -- ref. element concerne
    tsoms,tyquad,inquad,   -- geometrie
    elfq2a,elfq2b,         -- formules integration
    xsol,                  -- pour calcul non linearites
    icode,reyno,eps);
  -- reporter les contributions dans la matrice
  FOR_EACH (idr_1)          -- idr_1:fn_base_sur_el_referenc
    IF ((idr->id) (idr_1)<=nnbase) -- seulement si uon fi
      THEN
      DO;

```

```
FOR_EACH(idr_2) -- boucle sur les colonnes
  IF((idr->id)(idr_2)<=nnbase) -- si non fige.
    THEN
      DO;
        -- accumuler la contribution a
        -- Vmat(id_1,id_2)
        mat(ncmuns((idr->id)(idr_1),(idr->id)(idr_2)
          :+=matlm(idr_1,idr_2);
      END;
    IF("option_icode") -- accumuler second membre
      THEN
        secmbr((idr->id)(idr_1)):+=sm(idr_1);
      END;
    END;
  END remplir_mat;
END o8asmr;
```

<< M E T H O D E F R O N T A L E >>  
<< Principes >>

Telle qu'implementee ici la methode frontale consiste a effectuer la resolution de la matrice "conceptuelle" Vmat (qui sera dite matrice VIRTUELLE) , en operant etape par etape, et en ne representant, a chaque etape que les intersections des lignes et de colonnes ACTIVES (utiles) a cette etape dans Rmat.

En outre , on parvient a conduire simultanement a la descente dans la methode de Gauss, l'accumulation des contributions des elements (dans Rmat).

A chaque etape on effectue:

DESCENTE DANS LA METHODE DE GAUSS, ACCUMULATION DES CONTRIBUTIONS

1/ Assemblage d'une liste d'elements. Les fn\_base qui y interviennent sont soit deja actives, soit le deviennent (fn\_base "ENTRANTES"). On doit accumuler les contributions elementaires pour chaque coefficient de Vmat. Toutes les positions necessaires sont representees dans Rmat.

2/ On trie les fn\_base actives en 3 categories

a/ les fn\_base figees et qui n'interviennent plus

par la suite (sortantes) elles sont dites FIGEES\_S

b/ les fn\_base non figees et qui n'interviennent plus

dans les etapes suivantes; elles sont dites SORTANTES

c/ les autres sont dites RESTANTES

Normalement il y a toujours des restantes sauf a la derniere etape sinon le systeme se decouplerait ... et on en profiterait...

3/ On procede aux operations suivantes:

a/ On modifie le second membre pour tenir compte des fn\_base FIGEE\_S. (Ne correspondent pas a des inconnues).

b/ On elimine les inconnues des lignes SORTANTES.

Ceci revient a conduire la phase de descente pour les pivots en Vmat(SORTANTE, SORTANTE).

c/ On garde Rmat(SORTANTE, \*) pour calculer les inconnues eliminees lors de la remontee. (Ici on simplifie les transferts en gardant tout Rmat.

d/ On recupere les positions de Rmat qui viennent d'etre liberees par les fn\_base SORTANTES et FIGEE\_S, en y mettant des '0.' pour les utiliser a accumuler d'autres coefficients.

## ETAPE DE REMONTEE

Les etapes sont parcourues en sens inverse de la descente.

A chaque etape on effectue:

a/On restore  $R_{mat}$  dans l'etat ou il etait a la fin de l'elimination dans l'etape correspondante de la descente.

b/On calcule les composantes de la solution correspondantes aux  $fn\_base$  SORTANTES

On remarque qu'a la derniere etape tous les  $fn\_base$  sont SORTANT ce qui permet de passer a la remontee.

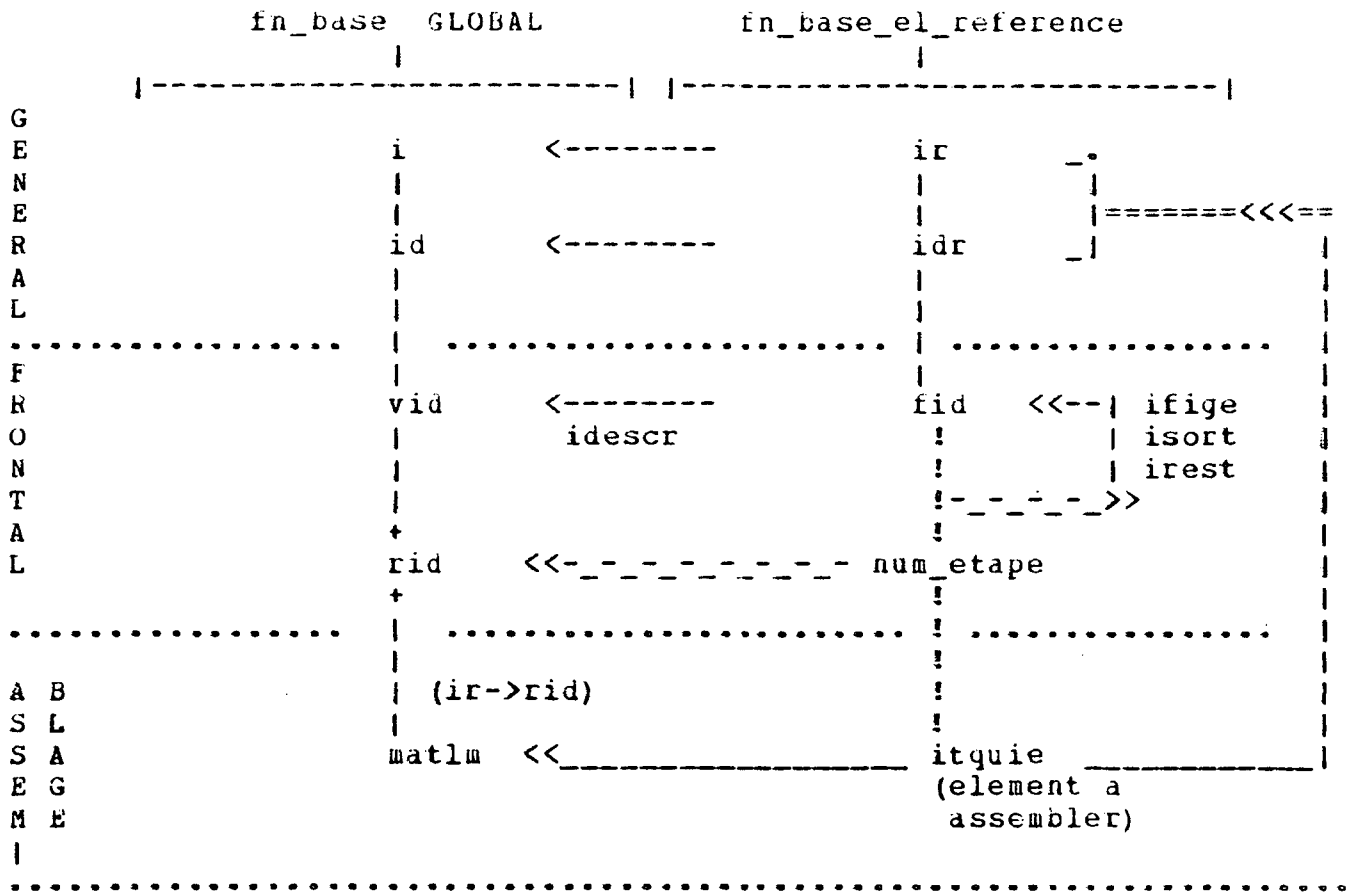
-----  
STRUCTURES DE DONNEES UTILISEES

Vmat            indexe par fn\_base: c'est la matrice conceptuelle du  
                  probleme. Elle est bien sur trop importante pour etre  
                  stockee. De plus elle est bordee par les colonnes  
                  correspondant aux fn\_base FIGES, ce qui permet de  
                  tenir compte des donnees de Dirichlet.

Rmat            matrice (npos, npos) dans laquelle on represente les  
                  coefficients utiles (ACTIFS) a une etape donnee. D'une  
                  etape a l'autre on modifie la correspondance Rmat->Vmat  
                  represente la partie utilisee, a une etape donnee dans  
                  Rmat. Les lignes et les colonnes de Fmat sont ordonnees  
                  suivant l'ordre logique pour les operations de la metho  
                  de de Gauss:

                  FIGES\_S        de 1 a ifige  
                  SORTANT        de ifige+1 a ifige+isort  
                  RESTANT        de ifige+isort+1 a ntot=ifige+isort+irest

ON a le SCHEMA GLOBAL decrivant les INDEXATIONS :



-----  
REALISATION de ces APPLICATIONS

Par ETAPE: ifige, isort, irest, ntot

calculés par o0frt1 servent  
à structurer Fmat logiquement.  
application: (fid->vid=id).

idescr

GLOBALEMENT

iprdeg

application id=vid-> rid  
Cette application est globale  
car on ne modifie pas la position  
rid affectée à un id dans Rmat  
d'une étape à l'autre.

id=vid

on identifie ces indices

```

Navier_Stokes_Frontal;
  MAIN_PROGRAM;
  DECLARE SUBROUTINE
    o7init (INTEGER), --Sous Programmes
    o7qad1 ( --Initialisations o7..
      T_tsoms, --Construction maillage
      T_elem, ) --Tableau sommets
    o7inqd ( --tableau elements
      T_tsoms, --Construction table de
      T_elem, -- degres de liberte
      T_deglib, ) --
    o7prep ( --Construction table
      T_descr, T_descr, -- decrivant les elem-
      ) -- -de reference
    o7diff (T_tsoms), --Deformation maillage
    o7xsol ( --Initialisation du
      T_tsoms, -- vecteur "solution"
      T_deglib, --
      T_vect(lnxsol), ) --vecteur solution
    o7div ( --Calcul norme
      T_tsoms, -- divergence vitesse
      T_deglib, --
      T_descr, --Descr. pour int.num.
      T_vect(lnxsol), ) --Solution
    o0frt1 ( --Preparer frontale
      T_deglib, --geometrie
      T_strat, --description etapes
      T_vect_i, --rend (vid->rid)
      T_vect_i, --zone de travail
    o0frt2( -- Descente/Assembl.
      T_deglib, -- geometrie pour
      T_tsoms, -- mtlm et assembl.
      (2) T_descr, -- pour o7mtlm
      T_vect_i, -- pour o7mtlm
      T_strat, -- (vid->rid)
      T_vect, -- pour assemblage
      T_vect, -- second membre
      T_vect_i, -- Rmat
      T_vect, -- zone de travail
      T_vect, -- solution a l'etape
      ) -- precedente (NON
      -- LINEARITES et Dirichlet non homo-
      -- gene.
    o0frt3 ( INTEGER, (2) REAL), -- parametres
      ) -- remontee methode

```



```

-- frontale
T_vect, -- rend solution
T_vect, -- second membre tel
-- que rendu par frt2
T_vect_i, -- vid->rid
Rmat,T_vect_i, -- zones de travail
INTEGER), -- non utilise

o7prtd (T_vect,,), --Impression
o8err (T_deglib, --Mise en forme et
-- impression de la
-- difference entre
-- iteres
T_vect,T_vect, --Solutions a comparer
),
(startg,o7flec,o8flec, --Pgms. graphiques
grfbel,contrl) -- sans interet ici
IN "graphics" PACKAGE; -- vu? ..

```

```

*****
*
*      Voici les principaux types de tableaux
*      manipules
*
*****
DECLARE   TYPE
          ( T_vect:REAL(*) ),           --Vecteurs
          ( T_tsoms: 1 sommet (nsoms)  --Pour chaque sommet:
            2 (x,y) REAL                --les coordonnees
            2 code INTEGER)            --suivant que sur le
                                         -- bord du domaine...
          ( T_element:
            1 element (nquadr),--
            2 nsom  INTEGER(4),-- numero de
                                         -- sommet. (Ceci sert
                                         -- a definir la geo-
                                         -- metrie.)
          ( T_deglib: 1 element (nquadr)--Pour chaque element:
            2 nsom  INTEGER(4),  -- num. sommets
            2 ndeglib_sommets --num. des degres de
                                         -- liberte correspond.
                                         INTEGER(4),  -- aux sommets
            2 ndeglib_milieu_cotes
                                         --degres de liberte
                                         -- associes aux milieu
                                         -- des cotes
                                         INTEGER(4),
            2 ndeglib_centre
                                         --degre de liberte
                                         INTEGER) -- au centre
            2 ndeglib
                                         --au total 9 degres de
                                         --liberte par element
          REDEFINES ndeglib_sommet
          THRU ndeglib_centre
          INTEGER(9)
          (T_descr:
            --description de
            -- l'element de ref.
            1 point_integr_num --par point d'
            -- integration
            -- numerique sur l'
            (npinu*) , -- element reference
            2 (x,y) REAL , --coordonnees
            2 coef REAL , --coeff.integration
            2 code INTEGER,--

```

```
                2 (val_fonc_base--fonctions de base
                  , val_ddx , -- leurs d/dx
                    val_ddy ) -- leurs d/dy
                    REAL(9)),
T_strat:        -- description des elements
                -- entrants a une etape donnee
1 front(nfront), -- indice par etape
  2 de:INTEGER, -- premier element
  2 a :INTEGER, -- dernier element
  2 pas:INTEGER, -- pas d'increment de numero
                -- d'element
-- de maniere generale on aurait pu associer a
-- chaque etape la liste des elements entrants.
-- Ceci est NECESSAIRE pour operer sur un
-- maillage quelconque.
T_vect_i:INTEGER(*): -- vecteur d'indices .Sert
                    -- a realiser les indirections
                    -- vid->rid , etc...
```

```

*****
*
*      Organisation des Communs D°Utilisation Globale
*
*      (sont exclus les communs, qui ne sont connus que localement)
*
*****

```

```

DECLARE      COMMON
              zcontx      --
                      --options globales pour les
                      -- programmes o7XXXX o8XXXX
                      --dimension des tableaux
              WRITTEN(o7init,o7inqd,o7prep,o7gad1,
                      Navier_Stokes_Frontal
                      )
              KNOWN  (o0frt*,o7div,o7mtlm,o7flec,o7xsol,
                      o8flec
                      ),
              zlftab      --longueur maximales disp.
                      -- pour les tableaux.
              WRITTEN(Navier_Stokes_Frontal)
              KNOWN  (o0frt*,o7flec,o7inqd,o7mtlm,
                      o7xsol,o8flec),
              zunite      --unites e/s
              WRITTEN(Navier_Stokes_Frontal) -- P.princ.
              KNOWN  (*)      --sert partout
              zgad1      --parametres de o7gad1
              WRITTEN(o7gad1)--
              KNOWN  ("graphics"),-- sert a faciliter
                      -- la lecture des resultats
                      -- (mise en page .)
              zwork      --zone de travail
              WRITTEN(o7div,o7flec,o7inqd,o8flec,o0frt*)
              KNOWN  (Navier_Stokes_Frontal),
              disk      WRITTEN (Navier_Stokes_Frontal)
                      KNOWN(o0frt*) ;
                      -- fichiers utilises par la
                      -- methode frontale pour garder
                      -- le contexte des etapes.

```

```

*****
DECLARE LOCAL_VAR      --Principaux tableaux pour tout
                      -- le programme
              xsol      :T_vect,      --solution apres l°iteration
              zw        :T_vect,      --
                      a l°iteration preced.
              tsoms     :T_tsoms,     --Sommetts

```

```
iquadr:T_element, --Elements
inquad:T_deglib , --Degres de liberte
(elfq2a,
elfq2b):T_descr , --Formules integration numerique
Rmat :T_vect(lnmat),--| M A T R I C E R E E L L E
iesn :T_mat_vect_i, --zone de travail pour o0frt1
--il est indique de mettre cette
--variable et Rmat en recouvre-
-- ment.
itquie:T_strat, -- definition des etapes
iprdeg:T_vect_i; -- utilise par o0frt* :vid->rid
```

```

-----
| initialiser:                                --Initialisation des communs ( options
|   DO;                                        -- et dimensions variables)
|   CALL startg;                              --Pour les graphiques ....
| etiq_1000:                                  --Passage au cas suivant si l°
|   CALL o7init;                              -- utilisateur est content et presse.
|   END initialiser;                          --
|
| geometrie :                                --Definition de la geometrie par les
|   do;
|   CALL o7qad1(tsoms,iquadr); -- tableaux des sommets et elements
|   CALL o7ingd(tsoms,iquadr, --Definition des degres de liberte
|     inquadr); -- suivant geom. elements finis et
|     -- fonctions approchees:
|     -- vitesse ux,uy
|   IF(.)CALL o7diff(tsoms); --Deformation du domaine?
|
| END geometrie;
|
| CALL o7prep(elfq2a,elfq2b) --Definition fonctions de base de la
| -- methode d°elements finis sur ref.
|
| -- Les donnees "statiques" sont maintenant pretes.On donne les
| -- valeurs initiales aux quantites ux,uy sur lesquelles on va
| -- iterer pour la resolution du probleme NON-LINEAIRE).
|
| CALL o7xsol(tsoms,inquad,xsol); --Remplit xsol:point de depart
| CALL o7div (tsoms,inquad,elfq2a, --Impressions seulement
|   xsol);
|
| --
| -- P R E P A R A T I O N des D O N N E E S pour F R O N T A L E
| --
| itquie(*).*= ...; -- definition des etapes
| --
| -- o0frt1 est appele UNE FOIS POUR TOUTES pour construire la
| -- description des etapes qui sera utilisee par o0frt2,o0frt3
| --
| CALL o0frt1(inquad, -- degres de liberte
|   itquie, -- description etapes
|   iprdeg, -- construit par o0frt1
|   iesn); -- zone de travail
| -- resultats dans le fichier
| -- ndescr et iprdeg
|
-----

```

```

--
--
-- <<<< I T E R O N S   M A I N T E N A N T   >>>>

iteration_non_lineaire:
  DO UNTIL ("convergence");      --On utilise la difference entre
                                -- iteres pour apprecier.

    ...
    secmbr=0.;                  --le second membre sera rempli
                                --par o0frt2 qui lui fera subir
                                --les combinaisons de la descente.
    CALL o0frt2(                -- premiere phase methode frontale
      inquad,                   -- pour assemblage et calculs mtlm
      tsoms,elfq2a,elfq2b,-- pour 07mtlm
      secmbr,                   --second membre
      Rmat, idescr,            -- zones de travail
      xsol,                   -- solution a l'iteration precedente
                                -- pour calcul NON LINEARITES et
                                -- donnees Dirichlet non homogenes
      icode,eps,reyno );      -- pour 07mtlm:parametres divers

    ...
    IF ("test") THEN          --Test sans interet en exploitation
test_assemblage_et_resolution:
      DO;
        CALL o0frt3(..,icode);-- le test est selectionne
        CALL o7prtd(..);      -- par icode
      END test_assemblage_et_resolution;

-- Sauver valeur de xsol a l'iteration precedente pour tests
zw = xsol ;
-- remontee de la methode frontale
CALL o0frt3(xsol,             -- solution
            secmbr,          -- construit par o0frt2
            iprdeg,         -- construit par o0frt1
            Rmat,           -- zone de travail
            idescr,         -- --- -- ----
            icode);        -- code inutilise pour l'instant

-- Calculer la difference entre iteres
erreur=MAX(ABS(XSOL-ZW) (*));
IF (erreur<eps ) THEN "convergence=oui" ;
...
IF ("option_d_impression")

```

```
                THEN DO;.....;END;
END iteration_non_lineaire;

IF ("option") THEN DO;
                CALL o7prtd;      -- impression solution
                CALL o7flec;...CALL grfbel; -- Graphiques
                END;

IF ("..") GO TO etiq_1000;      -- passer au cas suivant si l'utili
                                -- -sateur n'est pas degoute.
fin:...
CALL contrl;                    -- n'interesse que les Graphiques
END Navier_Stokes_Frontal;      -- voila !!!!!
```



```

o0frt1:SUBROUTINE;
-- Phase preparatrice de la
-- methode frontale.Preparation
-- des tableaux iprdeg et idescr.
-- Identification des Etapes.

    DECLARE ARGUMENT IN
        inquad : T_deglib,
        itquie : T_strat ;
-- Identif. fn_base
-- determination des elements a
-- faire entrer a chaque etape.

    DECLARE ARGUMENT OUT
        iprdeg : T_vect_i,
        iesn : T_vect_i;
-- (vid->rid)
-- zone de travail:on y marque,
-- pour chaque fn_base l'etape ou
-- sort.

    DECLARE COMMON
        zwork
            :: (idescr:
                1 list_alloc,
                -- 2 rid
                --
                2 fid,
                2 vid,
                2 rid )
            WRITTEN (o0frt1),
            zlftab,
            zunite,
            zcontx,
            disk WRITTEN (Navier_Stokes_frontal)
            KNOWN (o0frt1,o0frt2,o0frt3)
            :: (ndescr:INTEGER,
                --unite logique fichier descrip-
                --tion des etapes construit frt1
                ntable:INTEGER) ; --fichier servant a garder Rmat

```

```

--remplir iesn                pour chaque fn_has determiner
remplir_iesn:                  -- 1e etape ou elle sort
  DO etape=1 TO nfront;        -- etape par etape
    DO element=itquie(etape).de TO .a BY .pas; -- voici les elements
                                      -- a inserer.
      FOR_EACH(inquad(element).deglib)
        iesn((i->id)(.deglib))=etape;
      END;
  END remplir_iesn;

```

```

-- Maintenant, on sait a quelle etape les fn_base sortent. On peut donc
-- leur affecter des positions (:rid) dans Rmat, etape par etape.
-- On obtient aussi les positions dans Fmat en les classant.

```

```

par_etape:
  DO etape=1 TO nfront      ;
    ifige, irest, isort=0    ; -- pour compter les divers types

    -- collecter les informations element par element.
    DO element=itquie(etape).de TO .a BY .pas;
      DO idr_1=1 TO 18;      -- par fn_base_sur_el_reference
        IF (iprdeg((idr->id)(idr_1))<=0)
          THEN
            -- fn_base entrante
            -- lui attribuer une position dan
            -- Rmat: rid.
            ipos, iprdeg((idr->id)(idr_1))=ilibre;
            list_alloc(ipos).vid=(idr->id)(idr_1);
            ilibre=list_alloc(ipos).ptr;
            END fn_base_entrante;
          END;
        END;
      --

```

```

-- Compter les figes, les sortants et les sortants en decrivant la
-- liste LISTE_ACTIF. Attribuer les position dans Fmat au tur et a
-- mesure

```

```
list_alloc(*).fid= ...;
```

```

--(*) NOTE .Dans le programme on a prefere decrivre list_alloc(*) en
--      reperant les elements non dans la LISTE_ACTIF par le
--      fait que .rid=0
--

```

```

-- On a maintenant determine la situation et les tables de correspon

```

```
-- -dance pour cette etape.On ecrit la description sur le fichier
-- ndescr .La seule information est .fid qui deviendra idescr :
-- et determinera pour cette etape l'application fid->rid.

WRITE ntot,ifige,irest,isort,list_alloc(*).fid ON ndescr;
--
--
-- Preparer pour l'etape suivante
-- Exclure les sortants de LISTE_ACTIF
--
-- (*) NOTE .Dans le programme on opere en 2 etapes qui gagneraient
-- a etre confondues .
END par_etape;
--
--On connait maintenant seulement la dimension maximale de Rmat
-- necessaire .
lnmat=...;
REWIND ON ndescr; -- on va utiliser ce fichier pour
-- la descente.

END o0frt1;
```

```

oofrt2:SUBROUTINE;
-- Phase de descente dans la
-- methode de Gauss et d'accumu-
-- lation des coefficients de la
-- matrice.

  DECLARE ARGUMENT IN
    inquad  :T_deglib, -- pour assemblage
    tsoms   :T_tsoms,  -- pour o7mtlm
    (elfq2a,elfq2b) -- formules integration numerique
    :T_descr, -- pour o7mtlm
    iprdeg  :T_vect_i, -- definition de (vid->rid)
    itquie  :T_strat,  -- definition liste des elements
    -- entrants par etape.pour assemb
    icode   :INTEGER,  -- pour o7mtlm
    eps     :REAL,     -- parametre de penalisation
    reyno   :REAL ;    -- parametre "physique"

  DECLARE ARGUMENT OUT
    secmbr  :T_vect,   -- second membre .Il est modifie
    -- pendant la descente.ON le
    -- rend dans cet etat:pret pour
    -- remontee.
    rmat    :T_vect(lnmat), -- zone de travail on y rend
    -- la valeur de rmat prete pour l
    -- remontee.
    idescr  :T_vect_i, -- zone de travail .
    xsol    :T_vect ;  -- solution .on y trouve les
    -- de Dirichlet non homogenes

  DECLARE COMMON
    zlftab,
    zunite,
    zcontx,
    disk    ::(ndescr, --fichier prepare par oofrt1
              ntable), --fichier de sauvegarde de Rmat
              -- en attendant la remontee.
    zqua   WRITTEN (o7mtlm) -- zone de communication
           KNOWN(oofrt1); -- avec o7mtlm.(cf. o8asmr)

```

```

-----
| Rmat (*,*)=0.           ;           -- initialiser Rmat (les
|                               -- contributions sont additives
| descente_par_etape:      -- par etape accumuler,eliminer..
|   DO etape=1,nfront;
| accumulation_contrib:
|   DO element=itque(element).de TO .a BY .pas;
|                               -- accumuler les contributions pa
|                               -- element.
|   CALL o7mtlm(element,.,.,.); -- remplir matlm dans zqua
|                               -- par ligne de matlm:idr_1 est
|                               -- un numero de fonction de base
|                               -- sur element de reference
|   DO idr_1=1 TO 18;
|     DO idr_2 =1 TO 18 ;       -- par colonne
|       Rmat((idr->id->rid) (idr_1,idr_2)) :+=
|         matlm(idr_1,idr_2) ;
|     END;
|     secnbr((idr->id) (idr_1)) :+=sm(idr_1);
|   END;                       -- accumuler aussi second membre
| END accumulation_contrib;
| --
| -- on va passer a l'elimination (descente dans la methode de Gauss
| -- il faut d'abord lire la definition de (fid->vid)
| --
| READ ntot,ifige,isor,irest,idescr(*) FROM ndescr;
| --
| elimination_fn_base_figees_sortantes:
|   DO;                         --Ici ceci est effectue par o7mtl
|     ...                       -- ,ce programme pourrait aussi
|   END;                         -- faire.
| descente_gauss:
|   DO;                         -- Eliminer les inconnues SORTANT
|     DO f_ipiv=ifige TO ifige+isor; -- ligne du pivot=num. inconnue a
|       ...                     -- eliminer.
|     DO f_il=f_ipiv+1 TO ifige+isor+irest ;
|       ...                     -- On elimine l'inconnue sur
|       ...                     -- TOUTES les lignes SORTANTES et
|       ...                     -- RESTANTES.
|     -- On DEMONTRE que ceci est correct. En remarquant, en parti-
|     -- culier que les colonnes correspondant aux fn_base SORTANTES
|     -- sont aussi totalement accumulees, et que l'on effectue sim-
|     -- plement une permutation dans l'ordre d'accumulation de
|     -- contributions ADDITIVES.
|     --
|     DO f_ic=f_ipiv+1 TO ifige+isor+irest; -- par colonne sur

```

```

-- une ligne.
Rmat((fid->vid->rid) (f_il, f_ic)) =
Rmat((fid->vid->rid) (f_il, f_ic)) -
Rmat((fid->vid->rid) (f_ipiv, f_ic)) *
Rmat((fid->vid->rid) (f_il, f_ipiv)) /
Rmat((fid->vid->rid) (f_ipiv, f_ipiv));
END;
--combinaison avec le second membre
secmbr((fid->vid) (f_il)) =
secmbr((fid->rid) (f_il)) -
secmbr((fid->rid) (f_ipiv)) *
Rmat((fid->vid->rid) (f_il, f_ipiv)) /
Rmat((fid->vid->rid) (f_ipiv, f_ipiv));
END;
END;
END descente_gauss;
-- Sauver la matrice Rmat (on utilisera les lignes
-- correspondant aux fn_base sortantes lors de la remontee.
WRITE Rmat ON ntable;
END descente_par_etape ;
END o0frt2;
```

```
o0firt3:SUBROUTINE;                                -- Remontee methode frontale.
                                                    -- L'algorithme revient a
                                                    -- effectuer la remontee de
                                                    -- la methode de Gauss.

  DECLARE ARGUMENT IN
    secmbr: T_vect, -- second membre deja modifie
                                                    -- par o0firt2 pendant la descent
    iprdeg: T_vect_i, --realise l'application vid->rid
    icode : INTEGER; --option sans objet actuellement

  DECLARE ARGUMENT OUT
    xsol : T_vect, --solution du systeme lineaire
    idescr: T_vect_i, --zone de travail
    rmat : REAL(lnmat); -- zone de travail

  DECLARE COMMON
    zlftab,
    zunite,
    zcontx,
    disk;

                                                    -- definit les fichiers construit
                                                    -- par o0firt1 et o0firt2 resp.
```

```

--on opere etape par etape mais dans l'ordre INVERSE
DO etape=nfront TO 1 BY -1;
  READ ntot,ifige,irest,isort,idescr(*) FROM ndescr BACKWARDS;
  READ Rmat                                FROM ntable BACKWARDS;
                                          --restaurer le contexte de cette
                                          -- etape

remontee_gauss:
                                          -- remontee dans la methode de
                                          -- Gauss
DO f_ipiv=ifige+isort TO ifige BY -1;
  -- on calcule les inconnues
  -- dans l'ordre INVERSE de leur
  -- elimination.
  xsol((fid->vid) (f_ipiv))=secmbr((fid->vid) (f_ipiv));
DO f_icol=f_ipiv+1 TO ifige+isort+irest;
  xsol((fid->vid) (f_ipiv)):+=
    - Rmat((fid->vid->rid) (f_ipiv,f_icol)
    * xsol((fid->vid) (f_icol));
END;
  xsol((fid->vid) (f_ipiv))=
    xsol((fid->vid) (f_ipiv))
    / Rmat((fid->vid->rid) (f_ipiv,f_ipiv));
END remontee gauss;
END;
END oofrt3;                                -- Fin de l'etape

```





