Les Projets,Des EDP à leur résolution par la méthode des éléments finis

F. Hecht

Année 2016-2017

1 Le Four à micro onde 3D

En version Helmholtz il s'agit de résoudre un problème du type:
Trouver u une fonction définit sur l'ouvert Ω à valeur dans $\mathbb{C}$ tel que

$$\omega^2 \mu u + \nabla . \frac{1}{\varepsilon} \nabla u = 0 \in \Omega$$

avec des données aux bord imaginaires.
En version Maxwell fréquentiel il faut résoudre

$$i\omega\mu H + \nabla \times E = 0; \quad -i\omega\varepsilon E + \nabla \times H = \sigma E \quad \text{dans}\Omega$$

des données aux bords du type $E \times n = g$ et en plus $\nabla . E = 0$ dans Ω .
Le couplage avec la température θ, ce fait comme suit:

$$-\nabla . K \nabla \theta = f$$

dans l'objet à cuire, et où f est nulle hors de l'objet à cuire, et est égal à $u\overline{u}$ dans l'objet à cuire.
Le projet est la résolution de 2 équations découples avec les données suivante $\varepsilon = 1$ dans l'air et $\varepsilon = 4$ dans la viande.
On pourra prendre des conditions de Dirichlet sur l'objet à cuire.

2 Navier-Stokes 2D

2.1 Methode de caracteristique

Soit un fluide incompressible de vitesse $u \in H^1(\Omega)^d$ et pression $p \in L^2(\Omega)$, solution des équations de Navier-Stokes incompressible dans un domaine Ω de $\mathbb{R}^d$ où $d = 2$ ou 3.

$$\frac{\partial u}{\partial t} + u . \nabla u - \nu \Delta u + \nabla p = 0$$

$$\nabla . u = 0$$

plus des conditions aux limites de type Dirichlet $u|_\Gamma = u_\Gamma$.
Un utilisant la dérivée particulaire pour approcher le terme $\frac{\partial u}{\partial t} + u . \nabla u$, on obtient le schéma suivant:

$$\frac{u^{n+1} - u^n \circ \chi^n}{\delta t} \approx \frac{\partial u}{\partial t} + u . \nabla u$$

où $\chi^n(x)$ est la fonction de transport qui donne la position de la particule x au temps $t^{n=1}$ et qui était en $\chi^n(x)$ au temps $t^n = t^{n+1} - \delta t$. Donc, on a $\chi^n(x) = \zeta_x(t^n)$ où $\zeta_x(t)$ est solution de l'équation différentielle rétrograde suivante $d\zeta_x/dt = u(\zeta_x(t))$ avec condition finale $\zeta_x(t^{n+1}) = x$.
Nous obtenons le schéma temporelle, suivant d'ordre 1.

$$u^{n+1} - \nu \delta t \, \Delta u^{n+1} + \delta t \, \nabla p^{n+1} = u^n \circ \chi^n$$

$$\nabla . u = 0$$

Donc, il ne reste plus qu'à discrétiser le problème de Stokes en utilisant une méthode standard.

2.2 Methode Newton

Soit un fluide incompressible de vitesse stationnaire $u \in H^1(\Omega)^d$ et pression $p \in L^2(\Omega)$, solution des équations de Navier-Stokes incompressible dans un domaine Ω de $\mathbb{R}^d$ où $d = 2$ ou 3.

$$u.\nabla u - \nu\Delta u + \nabla p = 0$$

$$\nabla.u = 0$$

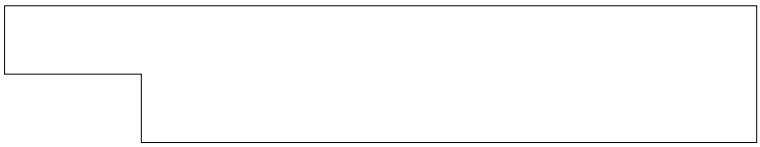
plus des conditions aux limites de type Dirichlet $u|_\Gamma = u_\Gamma$.
Rappel, la méthode de Newton pour résoudre une équation non linéaire est: défini par la suite partant de u^0 et définie par récurrence

$$u^{n+1} = u^n - DF(u^n)^{-1}F(u^n)$$

Comme le problème est de plus en plus difficile quand la viscosité diminue, il faudra peut être faire des itérations sur la viscosite en partant d'une viscosité de 1, pour atteindre la viscosité final.

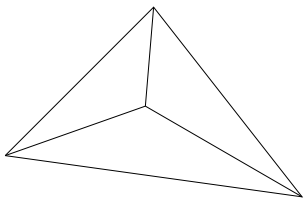
2.3 Domain de calcul

Le domaine de calcul pourra être une marche descendante de $\frac{1}{2}$ dans un canal hauteur 1 avec une viscosité ν de 1/100 et 1/400 avec un profile d'écoulement en entrant parabolique de vitesse maximal unitaire, une vitesse nulle sur les partie haute et base, et une condition de type Neumman en sortie c'est à dire $(\nu\partial_n u) - pn = 0$.



2.4 Mini-Élément

Soit V_h l'espace de fonctions P_1+bulle, c'est à dire, que les fonctions de bases sur un triangle T sont dans l'espace engendrer par les 4 fonctions suivante $\lambda_1^T, \lambda_2^T, \lambda_3^T, \lambda_b^T$ où λ_b^T est une fonction positive défini sur le triangle T, et nulle sur ∂T. par exemple λ_b est le produit des $\lambda_1^T, \lambda_2^T, \lambda_3^T$, où bien en découpant le triangle en 3 sous triangles ayant comme sommet commun le barycentre, on prend la fonction de base P_1 associée au barycentre.



L'intérêt est retiré la fonction bulle en faisant une élimination de gauss des noeuds internes associer aux fonctions bulles.

2.5 P1-P2

Ici l'espace V_h est obtenue en prenant un élément fini P_2 Lagrange sur chaque triangle, ou les degrés de libertés sont sur les sommets des triangles, et aux milieux des arêtes des triangles.
Les 6 fonctions des bases sont :
$4\lambda_i^T\lambda_j^T$ au milieu de l'arête i, j,
$\lambda_i^T(2\lambda_i^T - 1)$ au sommet i
le problème est la construction des tableaux des arêtes, et si possible la renumérotation des inconnues

2.6 P1-P1 iso P2

Ici l'espace V_h est obtenue en prenant un élément fini P_1 sur un maillage emboîté 2 fois plus fin. Le problème est entre autre la construction du maillage 2 fois plus fin, le problème d'injection d'un maillage dans l'autre.

2.7 P1 non conforme, P0+P1

Ici l'espace V_h est obtenue en prenant un élément fin i P_1 non conforme, c'est a dire que les fontions sont P_1 sur chaque élément et définies par leur valeur aux milieux des arêtes du maillage. La pression est plus bizarre puisque que c'est la somme de deux espaces d"element fini classique, l'espace de fonction P_1 continue et l'espace des fonctions constantes par triangles. On remarquera que l'intersection des deux espaces est réduit aux fonctions constante.

2.8 Les projets associé de la résolution des équation de Navier Stokes

Choisir entre la méthode des caractéristiques et la méthode de Newton, et choisir le type d'élément fini : Mini-Élément, P2-P1, P1-P1 iso P2, P1 non conforme, P0+P1. Soit 8 projets différent.
Bien sur il faut commencer par la résolution du problème de Stokes.

3 Modèle de turbulence

Le modèle de Smagorinsky propose une loi de viscosité fonction du maillage et du gradient de vitesse

$$\nu|_T = 0.01|T|\,|\nabla u + \nabla u^T|$$

couplé avec les équations de Navier-Stokes.
On pourra tester le modèle avec des conditions aux limites de non glissement ou de glissement:

$$u.n = 0, u.s = 0 \text{ ou } u.s + \beta\frac{\partial u.s}{\partial n} = 0$$

Et puis on pourra aussi tester le modèle $k - \varepsilon$:

$$\nu = c_\mu\frac{k^2}{\varepsilon}$$

$$\partial_t k + u\nabla k - \nu|\nabla u + \nabla u^T|^2 - \nabla.(\nu\nabla k) + \varepsilon = 0$$

$$\partial_t\varepsilon + u\nabla\varepsilon - c_i k|\nabla u + \nabla u^T|^2 - c_\varepsilon\nabla.(\nu\nabla k) + \varepsilon + c_2\frac{\varepsilon^2}{k} = 0$$

avec des conditions aux limites sur k et $\partial\varepsilon/\partial n$ (cf. Mohammadi-Pironneau (1994)).

4 Méthodes de Schwarz en 3D

On se propose de résoudre en parallèle l'EDP

$$-\Delta u = f \quad \text{dans } \Omega, \quad u_{|\Gamma} = g$$

On dispose pour cela d'un générateur de maillage emc2 et d'un solveur éléments finis. Le solveur lit une triangulation d'un fichier, construit la matrice du système linéaire et le résout par une méthode itérative (gradient conjugué préconditionné).

4.1 Méthode avec recouvrement

Le domaine Ω est l'union $\Omega_1 \cup \Omega_2$ telle que $\Omega_1 \cap \Omega_2$ soit non vide. La partie de frontière de $\Omega_1 \cup \Omega_2$ qui est strictement à l'intérieur de Ω_i sera notée $\Gamma_{ij}, (i, j = 1, 2)$. L'algorithme consiste à boucler en m sur

$$-\Delta u_i^m = f \quad \text{dans} \quad \Omega_i, \quad u_i^m|_{\Gamma-\Gamma_{ij}} = g, u_i^m|_{\Gamma_{ij}} = u_j^{m-1}$$

avec $i = 1, 2$.
On construira donc 2 triangulations, une pour chaque Ω_i , soit directement (plus facile), soit à partir d'une triangulation unique de (plus difficile mais plus élégant) avec deux frontières internes. On pourra utiliser l'indicateur de région, "region", défini par freefem pour chaque triangle (voir doc de freefem). Il faudra alors extraire les 2 triangulations de la triangulation globale. Dans les 2 cas Il faudra ensuite construire un tableau de correspondance des numéros des sommets de Γ_{ij} d'une triangulation à l'autre (utiliser le numéro des frontières et un test sur les coordonnées des points).

4.2 Méthode sans recouvrement

Le domaine est l'union de $\overline{\Omega_1 \cup \Omega_2}$ mais $\overline{\Omega_1} \cap \overline{\Omega_2}$ est réduit à une frontière Σ . L'algorithme consiste à boucler en m sur

$$-\Delta u_i^m = f \quad \text{dans} \quad \Omega_i, \quad u_i^m|_{\Gamma - \Gamma_{ij}} = g, u_i^m|_\Sigma = \lambda_j^{m-1}$$

avec i=1,2 et, pour un ρ fixe choisi à l'initialisation et un choix de normale n, en déterminant le signe de la formule suivante en faisant un petit calcul 1D.

$$\lambda^m = \lambda^{m-1} \pm \rho \left(\frac{\partial u_1^{m-1}}{\partial n} - \frac{\partial u_2^{m-1}}{\partial n} \right)$$

L'implémentation se fait comme dans la méthode avec recouvrement et là aussi on a le choix entre deux méthodes pour construire les triangulations de .
Amélioration du solveur
Si le temps le permet, on pourra remplacer les itérations de point fixe par des itérations de gradient conjugué pour λ. Et si vous avez encore plus de temps vous pouvez préconditionner le gradient conjugué.

4.3 Une autre alternative avec recouvrement

Sous les même hypothèse que la méthode avec recouvrement.

Algorithm is an alternative which does not require an interpolation on boundaries but in the interior of the domains.
Let

$$V_i = \{v \in L^2(\Omega) : \quad v|_{\Omega_i} \in H_0^1(\Omega_i), \quad v|_{\Omega \setminus \Omega_i} = 0\}$$

Choose $\rho > 0$.
Begin loop

- Compute $u_1^{n+1} \in V_1$ such that

$$\int_{\Omega_1 \cap \Omega_2} \rho(u_1^{n+1} - u_1^n) v_1 + \int_\Omega \nabla(u_1^{n+1} + u_2^n) \cdot \nabla v_1 = \int_\Omega f v_1, \quad \forall v_1 \in V_1.$$

- Compute $u_2^{n+1} \in V_2$ such that

$$\int_{\Omega_1 \cap \Omega_2} \rho(u_2^{n+1} - u_2^n) v_2 + \int_\Omega \nabla(u_1^n + u_2^{n+1}) \cdot \nabla v_2 = \int_\Omega f v_2, \quad \forall v_2 \in V_2$$

End loop
It is in fact a variation of Schwarz algorithm because if we call $v_1^{n+1} = u_1^{n+1} + u_2^n, \quad v_2^{n+1} = u_2^{n+1} + u_1^n$ we find that these verify a time dependant version of Schwarz algorithm. However the convergence is easier to show and the discretization is different. The convergence of this algorithm is given by the following result

Theorem 1(J.L.Lions)
Algorithm 1 converges in the sense that $u_i^n \to u_i^*$ with $u_1^* + u_2^* = u$ solution of (1) and the decomposition is uniquely defined by

$$(B + A)u_1 = \frac{1}{2}(B + A)(u + u_1^0 - u_2^0) \quad \text{in} \quad \Omega_{12}, \; u_1|_{S_1} = 0, \quad u_1|_{S_2} = u$$
$$(B + A)u_2 = \frac{1}{2}(B + A)(u + u_2^0 - u_1^0) \quad \text{in} \quad \Omega_{12} \; u_2|_{S_2} = 0, \quad u_2|_{S_1} = u$$

5 Hyper elasticity model

The Hyper elasticity problem is the minimization of the energy $W(I_1, I_2, I_3)$ where I_1, I_2, I_3 are the 3 invariants. For example The Ciarlet Geymonat energy model is

$$W = \kappa_1(J_1 - 3) + \kappa_2(J_2 - 3) + \kappa(J - 1) - \kappa \ln(J)$$

where $J_1 = I_1 I_3^{-\frac{1}{3}}$, $J_2 = I_2 I_3^{-\frac{2}{3}}$,$J = I_3^{\frac{1}{2}}$,
let $\boldsymbol{d}$ the deplacement field , when

- $F = I_d + \nabla \boldsymbol{d}$
- $C = {}^tFF$
- $I_1 = {}^t(C)$
- $I_2 = \frac{1}{2}({}^t(C)^2 - {}^t(C^2))$
- $I_3 = \det(C)$

The problem is find

$$u = \underset{u}{\texttt{argmin}}\, W(I_1, I_2, I_3)$$

Write the Newton method to find the minimal value of W. remark than I_1, I_2, I_3 are just polynom of d_i, and $\partial_j d_j$.
The test the problem a beam problem with the following constant:

- $\kappa_1 = \frac{1}{2}10^3$
- $\kappa_2 = \frac{3}{2}10^3$
- $\kappa = 10^5$
-

6 Bose Einstein Condensate

The problem is simple use the software `Ipopt` to solve the following problem:
Find a complex field u on domain $\mathcal{D}$ such that:

$$u = \underset{||u||=1}{\text{argmin}} \int_{\mathcal{D}} \frac{1}{2}|\nabla u|^2 + V_{trap}|u|^2 + \frac{g}{2}|u|^4 - \Omega i\overline{u}\left(\begin{pmatrix} -y \\ x \end{pmatrix}.\nabla\right)u$$

to code that in FreeFem++
All the data are define in the Freefem++ script

```
real Rtf = 3.4,Rcoef=1.7;
real Omega = 2.; // go to 5.  == Ω
func Vtrap = (x*x+y*y)*0.5 +square(x*x+y*y)*0.25;// == V_trap
real g = 1000;
real mu =50 ;
func rhoTF = max(0., (mu - Vtrap)/g ); // Initialisation μ such ||ρ_TF|| = 1
// computation of mu Nethon Method ||ρ_TF|| = 1
func real Computemu(real mmu)
{
    mu = mmu;
        // Compute mu
    for( int iter =0; iter < 20 ; ++ iter)
    {
        real nu = int2d(Th)(rhoTF) -1.  ;
        real dmu =  int2d(Th) (real( (mu - Vtrap) >0) )/g ;
        cout << " iter " << iter << " err = " <<nu << " " << mu << endl;

        mu -= nu / dmu ;
```

```
        if( abs(nu ) < 1e-15) break;
    }
    assert( abs(int2d(Th)(rhoTF) -1.) < 1e-10);
    return mu;
}

Computemu(50);
real R0 = sqrt(sqrt(1.+4.*mu)-1);
assert(R0 < Rmax);
```

Pour l'installations de IPOPT, premièrement installe MUMPS sans MPI confiture le fichier `Makefile.inc` de la distribution
Faire marche l'exemple en sequenteil.
Puis installer configure `Ipopt` avec

```
./configure \
            --disable-shared --enable-static \
           --with-mumps='Lib MUMPS' \
           --without-hsl \
           --with-mumps-incdir=' dir include MUMPS ' \
     CXXFLAGS='-I'dir include MUMPS' \
    CFLAGS='-I'dir include MUMPS' \
   --with-blas='blas libs' --with-lapack='lapack libs' --prefix=$HOME/soft'
```